UNIVERSIDADE DO ESTADO DO RIO GRANDE DO NORTE

UNIVERSIDADE FEDERAL RURAL DO SEMIÁRIDO

MESTRADO EM CIÊNCIA DA COMPUTAÇÃO

DAYANNE KELLY FREIRE DA ROCHA ESCALE

UMA ABORDAGEM FORMAL PARA MODELAGEM DE QoS EM REDES EM CHIP

MOSSORÓ – RN

Setembro, 2011

DAYANNE KELLY FREIRE DA ROCHA ESCALE

UMA ABORDAGEM FORMAL PARA MODELAGEM DE QoS EM REDES EM CHIP

Dissertação apresentada ao Mestrado de Ciência da Computação – associação ampla entre a Universidade do Estado do Rio Grande do Norte e a Universidade Federal Rural do Semiárido - para a obtenção do título de Mestre em Ciência da Computação.

Orientadora: Karla Darlene Nepomuceno Ramos

Co-Orientadora: Cláudia Mª. Fernandes A. Ribeiro

MOSSORÓ – RN

Setembro, 2011

Catalogação da Publicação na Fonte.

Universidade do Estado do Rio Grande do Norte.

Escalé, Dayanne Kelly Freire da Rocha.
Uma abordagem formal para modelagem de QoS em redes em chip. / Dayanne Kelly Freire da Rocha Escalé. – Mossoró**, RN, 2011.**

87 f.

Orientador(a): Profª. Karla Darlene Nepomuceno Ramos

Dissertação (Mestrado em Ciência da Computação). Universidade do Estado do Rio Grande do Norte. Curso de Mestrado em Ciência da Computação.

1. Redes em chip - Dissertação. 2. Métodos formais - Dissertação. 3. Qualidade de serviço (QoS) - Dissertação. I. Ramos, Karla Darlene Nepomuceno. II.Universidade do Estado do Rio Grande do Norte. III.Título.

Bibliotecária: Elaine Paiva de Assunção CRB 15 / 492

DAYANNE KELLY FREIRE DA ROCHA ESCALE

UMA ABORDAGEM FORMAL PARA MODELAGEM DE QoS EM REDES EM CHIP

Dissertação apresentada ao Mestrado em Ciência da Computação para a obtenção do título de Mestre em Ciência da Computação.

APROVADA EM: ___ / ___ / _____.

BANCA EXAMINADORA

_____

Prof[a]. Karla Darlene Nepomuceno Ramos (Orientadora)
Universidade do Estado do Rio Grande do Norte (UERN)

_____

Prof[a]. Claudia Maria Fernandes Araujo Ribeiro(Co-Orientadora)
Universidade do Estado do Rio Grande do Norte (UERN)

_____

Prof. Ivan Saraiva Silva
Universidade Federal do Piauí (UFPI)

_____

Prof. Márcio Eduardo Kreutz
Universidade Federal do Rio Grande do Norte (UFRN)

“Nem olhos viram, nem ouvidos ouviram, nem jamais penetrou no coração humano o que Deus tem preparado para aqueles que o amam.” I Coríntios 2.9

AGRADECIMENTOS

A Deus pela capacitação intelectual e por ser uma fonte inesgotável de motivação. Obrigada, Senhor, por suavizar o fardo.

Às professoras Karla Ramos e Claudia Ribeiro, por todas as horas dedicadas à orientação deste trabalho. Pelo zelo e profissionalismo, pelo exemplo de mulheres fortes, pelo respeito às minhas limitações, serei sempre grata a vocês.

Ao meu esposo, Alex Escale, pelo total apoio, paciência e carinho.

Aos meus pais Ivaniro Rocha e Anailde Rocha e ao meu tio Ivanilson Rocha, que me tratou como filha. Obrigada pelo suporte na minha mudança e estadia em Mossoró.

À minha irmã Anne Kelly pela amizade de sempre.

Às pessoas queridas que encontrei durante o período do mestrado e aos amigos que se tornaram mais próximos. Agradeço por cada momento de descontração, pelos "corujões", pelas companhias em refeições e viagens, pelas caronas. Vocês foram responsáveis por todos os momentos que tornaram esses dois anos de curso inesquecíveis. Agradeço especialmente à Christiane Nobre, Clezio Azevedo,Felipe Costa,Karla Haryana, Mailson Couto, Marianna Araujo, Ruth Drebes e Selma Pontes.

Aos meus sempre líderes, Daniel e Daihani, David Jr. e Raquel Praxedes, que deram a mim e ao meu esposo cobertura espiritual, intercedendo, incentivando e se alegrando com nossas vitórias.

A todos os professores e funcionários da UERN e da UFERSA.

Ao CNPq por financiar os custos do projeto “Uma Abordagem Formal para Modelagem de QoS em Redes em Chip”.

RESUMO

Um Sistema Embarcado pode ser definido como todo e qualquer sistema computacional completo com elementos integrados em um único chip, interconectados por uma infraestrutura de comunicação. Para compor esta infraestrutura, pesquisadores na academia e na indústria têm proposto o uso de Redes em Chip, visto que estas tratam limitações das arquiteturas de barramentos compartilhados. Os Sistemas Embarcadosbaseados em Redes em Chip suportam aplicações com requisitos cada vez mais rigorosos. Para isso, estas redes de interconexão devem ser capazes de atender requisitos de Qualidade de Serviço (QoS) relacionados à latência, vazão, consumo de energia e área de silício. Para lidar com a complexidade dos componentes embarcados bem como das infraestruturas de comunicação, tem-se utilizado descrições com linguagens em alto nível de abstração como forma de auxiliar a análise qualitativa do sistema. Estas descrições facilitam a exploração do espaço de projeto, suas síntese e verificação e permitem o aumento da produtividade por tratarem quatro aspectos chave: abstração; reuso; automação e exploração. Estes aspectos são contemplados pelos chamados métodos formais. Este trabalho utiliza o formalismo Z para especificar um Modelo de QoS capaz de auxiliar a análise do comportamento dos mecanismos de comunicação e das topologias de uma Rede em Chip antes de sua implementação. O modelo proposto estende a especificação formal que compõe a metodologia CADZ de forma a adicionar o tratamento de requisitos de QoS.

Palavras-Chave:Métodos Formais, Notação Z, Redes em Chip, Qualidade de Serviço

ABSTRACT

An embedded system can be defined asany computer system complete with integrated elements in a single chip, interconnected by communication infrastructure. Researchers in academia and industry have proposed the use of Networks on Chip to make up this infrastructure, since they deal with limitations of shared bus architectures. The Embedded Systems Networks on Chip-based support applications that require increasingly stringent. For this reason, these interconnection networks must be able to meet requirements of Quality of Service (QoS) related to latency, throughput, power consumption and silicon area. To deal with the complexity of embedded components and the communication infrastructure has been used languagesdescriptions at a high level of abstraction as a way of helping the qualitative analysis of the system. These descriptions facilitate the exploration of design space, their synthesis and verification and allow for increased productivity by addressing four key aspects: abstraction, reuse, automation and exploitation. These aspects are covered by so-called formal methods. This work uses the Z formalism to specify a QoS model can help analyze the behavior of the communication and topology of a network on chip before implementation. The proposed model extends the formal specification that composes the CADZ methodology in order to add the treatment of QoS requirements.

Keywords: Formal Methods, Z notation, Networks on Chip, Quality of Service.

LISTA DE FIGURAS

SUMÁRIO

1. INTRODUÇÃO

Esta dissertação trata da modelagem de Qualidade de Serviço em infraestrutura de comunicação para Sistemas Embarcados. Neste trabalho, considera-se um Sistema Embarcado ou SoC (do inglês S*ystem on Chip*) todo e qualquer sistema computacional completo, formado por elementos funcionais integrados em um único chip.

Usualmente, os projetos de SoCs baseiam-se no reuso de núcleos de propriedade intelectual, também chamados blocos IPs (do inglês *Intelectual Property*). Estes são módulos de hardware pré-projetados e pré-verificados, que podem formar sistemas complexos (Gupta, 1997). A estrutura de um SoC deve incluir elementos de computação e armazenamento, bem como interfaces para dispositivos periféricos e uma infraestrutura de comunicação (Goossens *et al*., 2005a; Martin, 2001).

Estas infraestruturas podem ser do tipo barramento compartilhado ou Rede em Chip(do inglês *Network-on-Chip*) (Guerrier, 2000; Jantsch, 2003). O paradigma de comunicação baseado em barramento, do qual deriva modelos tais como o PCI (do inglês *Peripheral Component Interconnect*), mostra-se adequado para sistemas embarcados de pequena e média escala. Para lidar com as limitações das arquiteturas de barramentos compartilhados, pesquisadores na academia e na indústria têm proposto o uso de Redes em Chip. Este paradigma de comunicação substitui as interconexões *adhoc* de blocos IP por uma abordagem que provê comunicação que pode ser comparada, em alguns aspectos, com o modelo de referência ISO/OSI. É exatamente no contexto da infraestrutura de Redes em Chip que esta pesquisa está inserida.

As Redes em Chip são conjuntos de roteadores e canais ponto-a-ponto, que emergem como uma solução para as restrições existentes em arquiteturas de interconexão do tipo barramento, devido às seguintes características: (i) eficiência de energia e confiabilidade; (ii) largura de banda escalável quando comparadas à arquitetura de barramento; (iii) reusabilidade; (iv) decisões distribuídas de roteamento (Benini, 2001; Benini e De Micheli, 2004; Guerrier e Greiner, 2000).

Os sistemas em um único chip baseados em Redes em Chip devem suportar aplicações de tempo real, sistema multimídia, algoritmos de codificação e decodificação de vídeo, ambientes para jogos 3D e reconhecimento de voz, computação em nuvens, entre outras. Para isso, estas redes devem ser capazes de atender requisitos de Qualidade de Serviço ou QoS (do inglês *Quality of Service*) relacionados à latência, custo, consumo de energia e área de silício utilizada (Agarwal et al., 2009).

Essencialmente, prover QoS implica na reserva de recurso, tais como *buffers* e *links*, para determinada aplicação. Esse processo envolve dois aspectos: (i) definição do serviço por meio de uma quantificação e (ii) negociação dos serviços (Bjerregaard e Mahadevan, 2006). Esses serviços podem ser baixa latência, alta vazão, consumo de energia reduzido, limite de *jitter*, entre outros, e devem ser alcançados por meio da combinação de mecanismos de comunicação e topologias.

Para prover estes serviços, alguns projetos de redes em chip apresentam mecanismos, tais como: (i) chaveamento por circuito; (ii) chaveamento por pacote com o uso de canais virtuais; (iii) arbitragem baseada em prioridades para transmissão de pacotes, entre outras. Dentre estes projetos, alguns destacam a necessidade do uso de mecanismos de comunicação capazes de diferenciar o tráfego de acordo com classes pré-estabelecidas. Com isso, busca-se priorizar determinado fluxo de dados, principalmente os pacotes procedentes de aplicações com tarefas de rígidos *deadlines*.

1.1. Motivação

Para validar e avaliar se uma configuração de Rede em chip atende a determinados requisitos de QoS, é comum o uso de técnicas de simulação com linguagem de descrição de hardware, tais como VHDL e VERILOG. Esta técnica oferece benefícios relacionados à análise quantitativa da Rede em Chip, tais como as avaliações da área utilizada e do consumo de energia. Porém, a dependência do método quantitativo de uma dada amostra de entrada e a crescente complexidade dos SoCs dificultam a representação de todo o espaço amostral do sistema.

Em contrapartida, descrever componentes e infraestrutura de comunicação de SoCs em níveis de abstração mais altos facilita a exploração do espaço de projeto destes sistemas, bem como suas síntese e verificação. Por isso, linguagens de alto nível de abstração têm sido introduzidas no projeto de SoCs e são usadas, por exemplo, no *projeto* em nível de sistema eletrônico ou ESL (do inglês *Eletronic System Level*) (Coussy, 2009).

Por diminuírem o tempo de lançamento do produto no mercado, estas linguagens aumentam a produtividade. Segundo (Rigo *et. al*., 2011), este aumento envolve a combinação de quatro aspectos chave: abstração; reuso; automação e exploração. Estes aspectos estão presentes também nos chamados métodos formais.

Os formalismos, nas quais se baseia o método qualitativo para projeto de hardware (Goossens, 2004), apresentam como principal vantagem a possibilidade de provar a validade dos requisitos essenciais do projeto antes da geração de código e da simulação. Os projetistas de Redes em Chip podem, então, se beneficiar, utilizando de forma complementar as técnicas de análise quantitativa e qualitativa. Ramos (2007) apresenta a Metodologia CADZ (*Computer Aided Design based on Z*), que utiliza o formalismo Z para analisar qualitativamente as propriedades do sistema de comunicação, de modo a identificar possíveis falhas nas fases iniciais do ciclo de projeto.

Além de permitir tratar aspectos como sincronização, integridade do sinal, consumo de energia, entre outros, os métodos para projeto de Redes em Chip devem auxiliar os projetistas a definirem configurações que atendam aos requisitos de QoS das aplicações. Para isso, é importante que a metodologia de projeto em uso permita avaliar a influência dos mecanismos de comunicação e do tipo de topologia de uma Rede em Chip no atendimento aos requisitos de QoS das aplicações.

Portanto, com base no paradigma ESL, na metodologia CADZ e considerando que o projeto de Redes em Chip, com suporte aos requisitos de QoS, é uma tarefa complexa, torna-se necessário o desenvolvimento de um Modelo de QoS capaz de facilitar a análise do comportamento dos mecanismos de comunicação e da topologia de uma Rede em Chip antes de sua implementação.

1.2. Objetivos

Mediante a necessidade de se obter métodos que auxiliem os projetistas a tratarem qualidade de serviço em redes em chip, o objetivo dessa dissertação consiste em: desenvolver um modelo baseado em métodos formais, que permite identificar como mecanismos de comunicação e topologias podem ser combinados para atender requisitos de QoS.

Deste modo, esta dissertação tem como foco principal o uso de métodos formais para modelar mecanismos de comunicação que suportam QoS e analisar qualitativamente as propriedades do sistema de comunicação que precede o desenvolvimento do próprio sistema. Para isso, a metodologia CADZ foi estendida para que os mecanismos de comunicação especificados suportem QoS.

Com o modelo proposto pretende-se prover ao projetista da Rede em Chip meios de melhor conhecer o comportamento do sistema antes de sua implementação, o que minimiza riscos e reduz custos do projeto. A especificação possibilita mapear os requisitos de uma aplicação, também especificada formalmente, em uma configuração de Rede em Chip. Para atingir o objetivo principal, foram definidos os seguintes objetivos específicos:

- Identificar topologias e mecanismos de comunicação que, integrados, são capazes de prover Qualidade de Serviço nas Redes em Chip;
- Conhecer o formalismo Z a fim de definir como melhor especificar topologias e mecanismos de comunicação identificados anteriormente;
- Selecionar ferramentas que automatizem o processo de análise léxica, sintática, semântica das especificações em Z, bem como, o processo de animação;
- Especificar os componentes do modelo de QoS proposto;
- Animar, provar e refinar especificações desenvolvidas;
- Criar cenários e estudo de caso para validar o modelo.

1.3. Metodologia

O modelo de QoS proposto consiste em uma série de especificações, definidas de acordo com a notação Z. Outra etapa no desenvolvimento do modelo de QoS proposto é a Prova das especificações, que garante a consistência da especificação, bem como evita que propriedades sejam omitidas ou descritas inapropriadamente. Neste trabalho, as etapas de especificação e de prova foram desenvolvidas com o apoio da ferramenta Z/Eves (Saaltink, 1997a; Saaltink, 1997b), que automatiza tarefas como: checagem de sintaxe, de tipos e domínios; cálculo de precondições e prova de teoremas.

Para complementar as etapas de especificação e prova, foi utilizada a ferramenta de animação Possum (Hazel, 1997), que permite observar a corretude das operações especificadas. Para o uso desta ferramenta é necessário que a especificação esteja descrita segundo a notação SUM (Cogito, 1999), uma versão simplificada da linguagem Z com suporte aos conceitos de operação, estado e inicialização.

1.4. Organização da Dissertação

Além da seção introdutória, esta dissertação inclui: o capítulo 2 que apresenta conceitos fundamentais relacionados às Redes em Chip, bem como à aplicação de mecanismos de comunicação que suportam qualidade de serviço em Redes em Chip; o capítulo 3 que apresenta o modelo formal de QoS; o capítulo 4 que mostra a especificação formal do modelo; o capítulo 5 que apresenta a validação por meio de um estudo de caso; e, finalmente, a capítulo 6 que traz os resultados, a conclusão, bem como, propõe trabalhos futuros.

2. REDES EM CHIP E MÉTODOS FORMAIS

Redes em chip são redes de interconexão chaveadas como as encontradas em computadores paralelos e, similarmente a estas, se caracterizam pela suas topologias e pelos mecanismos de comunicação utilizados. A topologia de uma rede define como os canais físicos ou os *links* interconectam os nós da rede, descrevendo a disposição destes canais. Em geral, estas interconexões são modeladas por meio de grafos, onde os vértices representam roteadores e as arestas, os *links*. Os mecanismos de comunicação definem a forma como as mensagens são transferidas pela rede, sendo os principais: controle de fluxo, roteamento, arbitragem, chaveamento e memorização (Duato *et al.*, 2002).

Desde que as Redes em Chip foram apresentadas, vários grupos de pesquisas vêm propondo diferentes configurações, que combinam topologias, arquiteturas de roteadores e esquemas de comunicação diversos. Guerrier e Greiner (2000) apresentam um dos primeiros trabalhos com resultados efetivos sobre redes em chip e propõem a NoC SPIN. Além da SPIN, outros exemplos de redes em chip são Æthereal (Goossens *et al.* 2005b, Goossens e Hansson, 2010), QNoC (Bolotin *et al.* 2003), HQNoC (Dobkin *et al.*, 2005), asynchronous QNoC (Dobkin *et al.*, 2009), Hermes (Moraes *et al*., 2004), Hermes-A (Pontes *et al.*, 2010a), Hermes-AA (Pontes *et al.*, 2010a) e SoCIN (Zeferino, 2003).

Em alguns projetos, as configurações das Redes em Chip são validadas por meio do uso de métodos formais. Por exemplo, Schmaltz e Borrione (2004a, 2004b e 2005) apresentaram a especificação formal da Rede em Chip Octagon (Karim *et al.*, 2002), segundo a lógica *A Computational Logic for Applicative Common Lisp* – ACL2 (Kauffman *et al.*, 2000). Tsiopoulos e Waldén (2006) utilizaram o formalismo *B Action System* (Waldén e Sere, 1998) para criar um método de desenvolvimento composicional de esquemas de roteamento assíncrono para projeto de sistemas de Redes em Chip.

Formalismos que usam um método analítico para computar latência, vazão e requisitos de armazenamento para a rede Æthereal são apresentados em (Gangwal *et al.*, 2005). E, para a mesma rede, Gebremichael *et al.* (2005) apresentaram um modelo formal modular em *Prototype Verification System* – PVS, no qual um critério de corretude foi estabelecido para conseguir ausência de *deadlock*.

Salaun *et al.* (2007) apresentam uma abordagem formal para verificação de arquiteturas assíncronas usando descrições segundo o formalismo *Communicating*

Hardware Processes – CHP (Martin, 1986), que são posteriormente mapeadas para a notação LOTOS, para que estas sejam verificadas automaticamente pelo *toolbox* CADP (*Construction and Analysis of Distributed Processes*).Os autores validam a estratégia verificando duas arquiteturas: uma implementação do Padrão *Data Encryption* (NIST, 1999) e a Rede em Chip ANoC (Beigne *et al.*, 2005), utilizada como *backbone* da Rede em Chip FAUST (Beigné e Vivet, 2006).

Coste *et al*. (2010) utilizam o formalismo LOTOS para analisar o impacto,sobre a latência e a vazão, do protocolo de controle de fluxo utilizado no MPSoC xSTream baseado em Rede em Chip. Para enriquecer os modelos LOTOS com atrasos probabilísticos, os autores integram a especificação com um modelo IPC (*Interactive Probabilistic Chain*), que facilita a descrição de transições de estados.

Palaniveloo e Sowmya (2011) apresentam um modelo formal para a Rede em Chip Hermes e seu esquema de comunicação, utilizando a linguagem *Heterogeneous Protocol Automata*– HPA, proposta pelos autores. O modelo em HPA foi manualmente mapeado para uma descrição em PROMELA (*a PROcess MEta LAnguage*) e verificado pela ferramenta SPIN.

Os métodos formais oferecem a possibilidade de provar a validade dos requisitos essenciais do projeto antes da geração de código e da simulação. Além disso, é possível aproveitar a capacidade incremental destas técnicas, de forma que a complexidade de um dado projeto possa ser administrada pelo uso de estratégias de refinamento sucessivo. Ademais, a base matemática dos métodos formais permite que detalhes específicos de um sistema possam ser considerados irrelevantes durante uma especificação formal abstrata preliminar (Woodcock e Davies, 1996). Desta forma, por meio do uso de métodos formais em projetos de redes em chip é possível descrever a infraestrutura de comunicação, destacando informações que possam facilitar, por exemplo, a análise da eficiência de determinado mecanismo na garantia de Qualidade de Serviço.

As próximas subseções tratam dos conceitos básicos relacionados às redes em chip e dos vários aspectos envolvidos no projeto de configurações que objetivam atender requisitos de QoS, principalmente de aplicações multimídia. As subseções também tratam do uso de formalismos na avaliação e validação das configurações de redes em chip.

2.1 Redes em Chip: Topologias, Mecanismos Básicos de Comunicação e Métricas de Desempenho

As Redes em Chip, que são fortemente baseadas nas redes de interconexão chaveadas e utilizadas em computadores paralelos, utilizam apenas um subconjunto do espaço de projeto dessas redes de interconexão, pois apresentam maior número de fios e pinos e menos espaços de armazenamento (Dally e Towles, 2001). Por exemplo, algumas alternativas de mecanismos de comunicação utilizadas nas Redes em Chip são: controle de fluxo do tipo *handshake* e baseado em crédito; roteamento determinístico XY; chaveamento por pacote do tipo *wormhole*, entre outras que serão apresentadas após a subseção de topologias.

2.1.1 Topologias

A seleção da topologia ou do *layout* físico de uma rede de interconexão é uma importante fase de projeto, visto que os mecanismos de controle de fluxo e de roteamento dependem desta escolha (Dally e Towles, 2004). A escolha da topologia em geral é motivada pelos seguintes aspectos: desempenho; escalabilidade; simplicidade; confiabilidade; manutenção; capacidade de expansão e particionamento; e restrições físicas e de custo (Duato *et al*., 1997; Dally e Towles, 2004). Dally e Towles (2004) relacionam os fatores custos e desempenho a dois componentes, largura de banda e latência, que podem ser determinadas pela combinação de topologias e determinados mecanismos de comunicação.

Duato *et al*. (1997) classifica as redes de interconexão de acordo com a topologia em quatro grupos principais: rede compartilhada (barramento), rede direta, rede indireta e rede híbrida. Em (Benini e De Micheli, 2004), foi mostrado que essa taxonomia, utilizada na computação paralela tradicional, pode ser aplicada ao domínio dos Sistemas Embarcados. Várias destas topologias de rede de interconexão foram propostas para diferentes projetos Redes em Chip, respeitando as peculiaridades dos SoCs.

As topologias diretas mais utilizadas em Redes em Chip são a Malha (*Mesh)* e o Toróide (*Torus*). Enquanto que as topologias que mais se destacam nas redes indiretas são o *Crossbar* e as redes *Multiestágio*, como a *Butterfly* Bidirecional e a Árvore Gorda. A Figura 2.1 mostra a representação gráfica das topologias citadas.

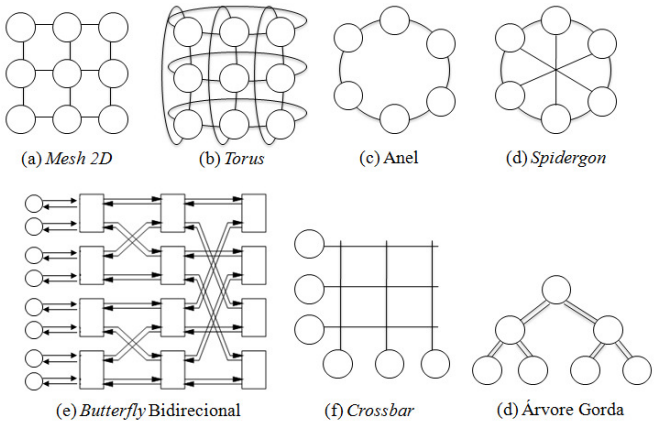


Figura 2.1. Algumas topologias utilizadas em Redes em Chip (Baseado em (Duato e Ni, 1997))

A escolha da topologia foi apontada por Dally e Towles (2001) como um desafio no projeto de Redes em Chip. Os autores analisam as topologias *Mesh* e *Torus* e provam que quando a dissipação de energia é um requisito crítico a escolha das redes *Mesh* são mais indicadas do que as *Torus*. Similarmente, o estudo apresentado em (Mirza-Aghatabar *et al.*, 2007) compara essas topologias e conclui que a *Torus* oferece menor latência do que a *Mesh*, porém por meio de maior consumo de energia.

Bononi e Concer (2006) realizaram um estudo comparativo entre as topologias *Mesh*, *Ring* e *Spidergon* com o objetivo de analisar qual arranjo provê menores níveis de latência em determinados cenários. Os autores observaram que a topologia *Spidergon*, em determinado cenário, oferece um bom *compromisso* entre desempenho, escalabilidade, simplicidade na manutenção, requisitos de área e energia utilizadas em SoCs.

As topologias *Mesh*, *Ring, Spidergon* e *Crossbar* são avaliadas e comparadas em Bononi *et al*. (2007), que utilizaram Redes em Chip na transmissão de dados gerados por uma aplicação MPEG4. Os resultados encontrados pelos autores indicam que num cenário com fluxo uniforme, a topologia *Crossbar* oferece melhor desempenho se comparada às demais topologias, devido a seu mapeamento de núcleos otimizado e às distâncias dos nós. Em contrapartida, a topologia *Spidergon* apresentou-se mais escalável se comparada à rede *Mesh*, mantendo o mesmo desempenho desta.

2.1.2 Mecanismos Básicos de Comunicação

Para atender os requisitos exigidos por determinada aplicação, além do desafio de definir uma topologia para a rede de interconexão, o projetista deve enfrentar o desafio de escolher mecanismos de comunicação tais como chaveamento, controle de

fluxo, roteamento, arbitragem e memorização. Esta escolha é um fator determinante no desempenho da rede, visto que estes mecanismos definirão como os recursos de comunicação serão compartilhados entre os componentes da rede.

Chaveamento

O mecanismo de chaveamento define como uma mensagem é transferida da entrada de um roteador para um de seus canais de saída. Se para a transmissão de uma dada mensagem todos os canais de saída entre o nó origem e destino forem reservados, tem-se o Chaveamento por Circuito. Do contrário, o chaveamento é classificado como Chaveamento por Pacote.

A Rede em Chip Æthereal (Goossens *et al*., 2005) utiliza chaveamento por circuitos para evitar a interferência de outros tráfegos na transmissão de determinado fluxo de dados, visto que por meio deste mecanismo um caminho completo é estabelecido da origem até o destino. O chaveamento por circuito, em geral, não utiliza *buffers*, pois não ocorre a contenção da mensagem na rede. Em contrapartida, os canais reservados não podem ser utilizados por outra mensagem, gerando menor utilização da rede. Essa é uma das razões pelas quais a maioria das Redes em Chip atuais, incluindo a SoCIN (Zeferino e Susin, 2003b) e QNoC (Bolotin *et al*. 2003), utilizam o chaveamento por pacote, em especial, o chaveamento *Wormhole*.

No chaveamento por pacote a mensagem é dividida em partes, que informam ao roteador qual caminho seguirão na rede. Neste caso, não há um caminho preestabelecido, ou seja, não há reserva de recursos e, consequentemente, a rede é melhor utilizada.

Quando no chaveamento por pacote, um pacote é completamente armazenado em cada roteador intermediário antes de ser encaminhado, tem-se a estratégia de chaveamento *Store-And-Forward* (SAF). Porém, para otimizar o uso de *buffers* e *links*, um roteador atual pode encaminhar os dados para o próximo roteador assim que a decisão de roteamento for concluída. Esta técnica é chamada de chaveamento *Virtual Cut-Through* (VCT).

As características que diferem o chaveamento por pacote *Wormhole* das outras técnicas de chaveamento por pacotes é que um roteador pode encaminhar um pacote tão logo recebe o seu cabeçalho e o canal de saída necessário esteja livre. E, em caso de indisponibilidade desse canal, o roteador não precisa absorver todas as partes do pacote.

Além disso, nesta técnica um pacote é particionado em unidades menores chamadas *flits*, o que possibilita a redução do custo dos *buffers* nos roteadores, que passam a ser menores, mais rápidos e mais baratos (Duato, 1997). O chaveamento *wormhole* possibilita o uso de controle de fluxo baseado em canais virtuais, que será abordado a seguir.

Controle de Fluxo

O controle de fluxo é um protocolo de sincronização para transmissão e recebimento de unidades mínimas de informação na rede. Este mecanismo possibilita a alocação de recursos, tais como *buffers* e *links*, para um determinado fluxo de dado. Este fluxo acontece tanto de um nó fonte para um nó destino como de uma porta de entrada para uma porta de saída num mesmo nó. Então, pode-se afirmar que o controle de fluxo ocorre em dois níveis distintos, no nível de pacote e no nível de enlace.

Uma alternativa para efetuar o controle de fluxo é por meio da técnica baseada em crédito, utilizada em Redes em Chip como a QNoC. No controle de fluxo baseado em crédito (Dally e Towles, 1997; Tanenbaum, 1996), representado graficamente pela Figura 2.2, cada nó fonte mantém um contador que controla a disponibilidade de *buffer* no contador receptor. Na Figura 2.2 o contador no nó fonte possui valor dois, indicando que existe espaço suficiente para armazenar dois *flits* no nó receptor. Ainda na Figura 2.2(a), o nó fonte possui o cabeçalho de uma mensagem armazenado, aguardando o encaminhamento. Quando o nó fonte encaminha o *flit* cabeçalho (Figura 2.2(b)), parte do *buffer* no nó destino passa a ser ocupado e o contador é decrementado. O nó fonte emite *flits* até que o valor do contador se anule (Figura 2.2(c)) e aguarda enquanto o contador mantém esse valor (Figura 2.2(d)). Quando o espaço ocupado anteriormente pelo *flit* é liberado (Figura 2.2(e)), o nó destino sinaliza para o nó fonte, que tem o contador incrementado e envia o último *flit* do pacote.

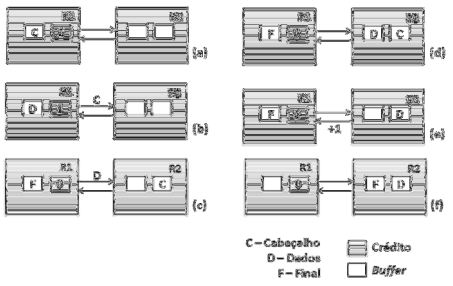


Figura 2.2. Estados do Controle de Fluxo Baseado em Crédito (Baseado em Duato *et al.*, 1997)

Outra técnica de controle de fluxo é o *Handshake*, utilizada pelas Redes em Chip HERMES e SoCIN. Esta técnica consiste no envio, por parte do nó fonte, de uma informação referente à intenção desse nó enviar uma mensagem para o nó alvo. Essa informação é enviada através de um sinal de validação, e a confirmação de disponibilidade de espaço no nó destino é enviada através de um sinal de reconhecimento (*acknowledge*).

Um terceiro método que realiza o controle de fluxo é a implementação de canais virtuais, que são pequenas filas de *buffers* associadas a um canal físico ou link (Dally, 1992). Com várias filas associadas a um *link*, em vez de apenas uma, pode-se aumentar a vazão de uma rede. Segundo Patterson e Davie (2003), essa técnica consiste em multiplexar um canal físico em *n* canais lógicos (canais virtuais), compartilhando a largura de banda entre diferentes fluxos.

Dally e Towles (2001) observam que um desafio na escolha do mecanismo de controle de fluxo de uma Rede em Chip é a busca por menores *buffers* e pela consequente redução de *overhead* nos roteadores. Esta preocupação deve-se ao fato de que o espaço ocupado por *buffers* está diretamente relacionado ao *overhead* de área de uma rede de interconexão. É importante que os métodos de controle de fluxo considerem o *compromisso* entre requisitos de armazenamento, desempenho, complexidade e consumo de energia.

Roteamento

Uma vez definido como os nós da rede estão interconectados, existe um conjunto de possíveis caminhos que uma mensagem pode atravessar para alcançar o seu destino. O mecanismo de roteamento define em qual desses possíveis caminhos ou rotas a mensagem deve seguir.

Uma boa decisão de roteamento otimiza a utilização dos recursos da rede e, em geral, busca minimizar o caminho percorrido pela mensagem (Dally e Towles, 2004). Logo, o algoritmo de decisão de rotas influencia diretamente na latência da rede ou no tempo necessário para uma mensagem chegar ao nó destino.

Segundo Duato *et al*. (1997) outras caraterísticas da rede de interconexão diretamente relacionadas aos algoritmos de roteamento são: conectividade; adaptatividade; ausência de *deadlock* e *livelock*; e tolerância à falha.

A característica de conectividade diz respeito à habilidade da rede de encaminhar uma mensagem ou partes de uma mensagem de um nó fonte a um nó destino. Enquanto que a adaptatividade é a propriedade de encontrar caminhos alternativos quando existe uma situação de contenção na rede. A capacidade de garantir ausência de *deadlock* e *livelock* evita, respectivamente, que mensagens nunca alcancem seus destinos por bloqueios e que estas busquem indefinidamente por recursos disponíveis. E, por fim, a tolerância à falha garante a entrega de mensagens mesmo com presença de falhas na rede.

Duato *et al*. (1997) ainda apresentam uma taxonomia para os algoritmos de roteamento de acordo com os seguintes critérios: número de destinos, distribuição de decisão, formas de implementação, adaptatividade, progressão, minimização e número de caminhos. Este trabalho abrange os algoritmos de roteamento, classificados de acordo com a adaptatividade, determinísticos e parcialmente adaptativos.

O algoritmo de roteamento determinístico mais utilizado no projeto de Redes em Chip é o XY, que roteia pacotes primeiramente na direção horizontal. Quando o pacote alcança a coluna do nó destino, é encaminhado na direção vertical até o receptor. Logo, este algoritmo é idealmente aplicado em redes com topologia *Mesh* e *Torus*. Com a proibição de determinadas rotas, evitam-se os ciclos mostrados na Figura 2.3 (a), permitindo apenas os encaminhamentos da Figura 2.3 (b). Como consequência, são evitadas situações de *deadlock* e *livelock*.

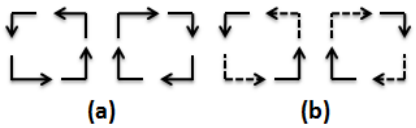


Figura 2.3. (a) Possíveis ciclos numa Rede Mesh; (b) Ciclos não permitidos pelo Algoritmo XY (em tracejado)(Baseado em (Glass e Ni, 1997))

Na literatura encontram-se Redes em Chip que implementam algoritmos de roteamento adaptativos. Por exemplo, a rede HERMES utiliza os algoritmos de roteamento parcialmente adaptativos para redes com topologia *Mesh 2D*: *West First* (WF), *North Last* (NL) e *Negative First* (NF). Estes algoritmos compõem o *Turn Model* (Glass e Ni, 1992), um modelo de roteamento projetado para redes com chaveamento *Wormhole*. O *Turn Model* busca garantir ausência de *deadlock* e *livelock*, caminhos mínimos ou não mínimos e adaptatividade máxima para as redes de interconexão. Sua principal característica é a análise das direções por onde um pacote pode ser encaminhado e a proibição dos ciclos, que esses encaminhamentos podem formar. A

Figura 2.4 representa graficamente as voltas permitidas e proibidas nos algoritmos do *Turn Model*.

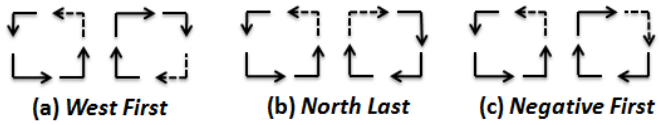


Figura 2.4. Rotas não permitidas pelos Algoritmos*Turn Model*(em tracejado)(Glass e Ni, 1997)

Chiu (2000) apresenta o modelo de roteamento parcialmente adaptativo denominado *Odd-Even*, projetado para redes com topologia *Mesh* com garantia de ausência de *deadlock*. Diferentemente do *Turn Model*, que proíbe o encaminho da mensagem em determinada direção, o *Odd-Even* restringe os locais exatos onde estes encaminhamentos não podem ocorrer. As avaliações realizadas pelos autores mostraram que a adaptatividade oferecida por este modelo é importante principalmente em tráfegos não uniformes.

Um algoritmo totalmente adaptativo encaminha uma mensagem para o seu destino por meio de qualquer uma das rotas existentes entre os nós fonte e destino. Em geral, as implementações deste tipo de algoritmo utilizam canais virtuais como forma de aumentar a adaptatividade da rede. Schonwald et al. (2007) apresentam um algoritmo de roteamento completamente adaptativo e tolerante a falha para Redes em Chip com chaveamento *wormhole*. A proposta foi desenvolvida por meio de um Modelo em Nível de Transação para redes com topologias *mesh*, *torus* e *hypercube*. Com isso os autores demonstraram que o algoritmo é capaz de distribuir o tráfego uniformemente por toda a rede, evitando a sobrecarga dos *links*.

Arbitragem

Uma Rede em Chip capaz de suportar diferentes tipos de fluxos de dados, com diferentes prioridades, deve implementar algum mecanismo de arbitragem, que defina em qual ordem estes fluxos devem ter acesso a um determinado recurso da rede. Este mecanismo de busca resolve os problemas de colisão de pacotes.

Alguns exemplos de arbitragens são os algoritmos *Round Robin* (RR), o *First Come First Serve* (FCFS), o Baseado em Prioridade (BP) e o *Round Robin* Baseado em Prioridade (RRBP). A Rede em Chip SPIN e o roteador RASoC implementam a arbitragem RR, enquanto a QNoC, XPIPES e a Æthereal utilizam o árbitro RRBP. Berejuck e Zeferino (2009) adicionam o uso do escalonamento por idade de pacotes à

Rede em Chip SoCIN, como forma de evitar que um pacote ou *flit* de baixa prioridade espere indefinidamente na rede pelo recurso requerido. Estes mecanismos são detalhados no capítulo 4.

Além dos métodos já citados, o acesso aos recursos pode acontecer por meio da multiplexação por divisão síncrona de tempo ou STDM (*Synchronous Time-Division Multiplexing*). Este método, utilizado pelas Redes em Chip Nostrum, aSoC e Æthereal,assume que: (1) roteadores compartilham da mesma noção de tempo e possuem a mesma frequencia de *clock;* (2) a alocação de *buffers* e *links* são acopladas, isto é, a reserva do *link* garante a reserva de *buffers* e vice-versa (Wang, Y. *et al*., 2008).

Memorização

Em virtude da possível indisponibilidade de canais de saída, quando do recebimento de pacotes na entrada dos roteadores, existe a necessidade de implementar algum esquema de memorização a fim de manter os pacotes bloqueados pelo roteador. A forma de implementação desses *buffers* interfere no desempenho do roteador. Sendo assim, a política de memorização utilizada numa Rede em Chip pode determinar a eficiência de outros mecanismos de comunicação, como a arbitragem.

Segundo (Bjerregaard e Mahadevan, 2006) afirmam que os *buffers* podem constituir a maior parte da área ocupada dos roteadores. Logo, equilibrar a quantidade de *buffers* com os requisitos de desempenho e aspectos como o tamanho e a localização destes dispositivos torna-se crucial no projeto de uma Rede em Chip.

Os *buffers* podem ser organizados de forma independente ou compartilhada, bem como, podem se posicionar na entrada, saída ou serem centralizados (Tamir e Frazier, 1992). Estas diferentes classificações são consideradas no modelo de QoS proposto.

2.2 Métricas de Desempenho

Durante o projeto de Redes em Chip, deve ser considerado o *compromisso* entre diferentes parâmetros que permitem mensurar o desempenho da rede. Na ausência de falha, as mais importantes métricas de uma rede de interconexão são latência e vazão.

Por esta razão, o modelo proposto neste trabalho combina diferentes mecanismos e topologias para tratar requisitos de QoS associados a estas métricas. Mas existem outras métricas a serem consideradas no projeto de Redes em Chip, como por exemplo o consumo de energia tratado a seguir.

Consumo de Energia

Consumo de energia é um aspecto crítico no projeto de roteadores para Redes em Chip e, na ausência de contenção na rede, é um eficiente indicador do nível de latência (Chen-Ling Chou e Marculescu, 2008). O baixo consumo de energia pode reduzir custos relacionados a testes e integração de sistemas, por exemplo, contribuindo para um alto desempenho computacional ser obtido (Huaxi Gu*et al.*, 2009). Além disso, os sistemas embarcados baseados em Redes em Chip suportam a realização em hardware das funcionalidades de aplicações dedicadas complexas. Estas são aplicações que requerem alto poder computacional e possuem sérias restrições de consumo de energia (Kreutz, 2005).

Por isso, muitas iniciativas preocupam-se em propor técnicas para aumentar o desempenho de Redes em Chip por meio da otimização do consumo de energia, como demonstra o trabalho apresentado em (Das *et al*., 2008). Os autores aplicam a compressão de dados apresentada em (Alameldeen e Wood, 2004) para reduzir o fluxo na rede e, consequentemente, diminuir a latência média e o consumo de energia.

Lee e Bagherzadeh (2009) apresentam um *framework* para estimação de energia que é capaz de prover precisos perfis de ciclos de energia, facilitando a análise do consumo de energia em nível de sistema para um roteador de Redes em Chip. Para isso, os autores propõem um modelo de alto nível e demonstram sua utilização por meio de diferentes mapeamentos de núcleos. Com o auxílio do *benchmark* SPLASH-2 (Woo *et al.*, 1995), observam a influência destes mapeamentos sobre o consume de energia de uma Rede em Chip com topologia *mesh* 7x7.

O trabalho proposto por (Xiaohang Wang *et al*., 2010) investiga as restrições impostas pela largura de banda e latência sobre o mapeamento de núcleos de uma Rede em Chip com topologia mesh, a fim de minimizar o consumo de energia entre os núcleos. Para isso, os autores utilizam o modelo de energia apresentado em (Hu e Marculescu, 2005), onde o consumo médio de energia para enviar um bit de dado entre os nós ti e tj pode ser representado como:

$$E^{ti,\,tj}_{bit} = \eta_{hops} \times E_{Sbit} + (\eta_{hops} - 1)\, E_{Lbit},$$

onde η_{hops} é a quantidade de roteadores entre os nós *ti* e *tj*, E_{Sbit} é a energia consumida no chaveamento e E_{Lbit} é a energia consumida nos *links* entre *ti* and *tj*.

Latência

A latência de uma rede chaveada por pacote, sem contenção, segundo (Ni, 1993) é definida pelo tempo necessário para uma mensagem de comprimento L passar através de um canal de largura de banda B, vezes a quantidade de nós (D) entre a origem e o destino, ou seja, (L/B) D, onde L é o comprimento (Length) do pacote, B é a largura de banda (*Bandwidth*) e D é a distância ou comprimento do caminho entre os nó de origem e destino. Esta definição é adequada para o chaveamento *Store-and-Forward* – SAF, e para os demais tipos de chaveamento esta definição é ajustada.

Duato, em (Duato, 1997), esclarece que a latência das técnicas de chaveamento envolve o tempo que um roteador leva para rotear um pacote, o tempo de propagação do pacote no canal (*link*) entre os roteadores e o tempo de um pacote ser enviado do *buffer* de entrada para o buffer de saída do roteador, este último depende do esquema de memorização utilizado. Esta definição de Duato é aplicada às Redes em Chip.

Diversas iniciativas preocupam-se em propor diferentes metodologias de projeto e configurações de Redes em Chip que possibilitem maximizar a latência de uma Rede em Chip, como mostra (Rose *et al.*, 2011). Os autores propõem um algoritmo de roteamento determinístico para uma topologia em chip 3D, combinando o roteamento XY nas camadas 2D e garantindo menor latência na rede do que outros algoritmos já conhecidos para topologia 3D.

O presente trabalho considera a infraestrutura de hardware da rede, bem como, filas de fluxos de dados na rede, esta métrica pode ser definida como o tempo necessário desde a injeção do cabeçalho da mensagem na rede até a última informação da mensagem ser recebida no nó receptor. E o tempo que a mensagem leva em filas de *buffers* também é adicionado ao valor da latência.

Vazão

A vazão pode ser definida como o máximo de informação entregue por unidade de tempo ou o máximo de tráfego suportado pela rede. Logo, está métrica pode ser representada em bits ou em *flits* por nó. Diferentes valores de latência e vazão podem ser alcançados pela combinação de topologias, mecanismos de comunicação.

Para alcançar uma configuração de Rede em Chip que produza alta vazão é necessário eleminar pontos de contenção na rede, como demonstra (Chen-Ling Chou e

Marculescu, 2008). Estes autores propõem um algoritmo de mapeamento de núcleos que objetiva minimizar a latência e maximizar a vazão, por meio da eliminação de pontos de contenção. Para isso, são considerados as topologias *Mesh* 2D e outras como *ring*, *torus* e *3D-grid*, o chaveamento *wormhole* e algoritmos determinísticos.

Diversos trabalhos relacionados às Redes em Chip exploram o compromisso entre Latência e Vazão, como (Pande *et al.*, 2006) que propõem o projeto baseado nestas métricas. Os autores apresentam uma abordagem que obtém resultados que permitem comparar diferentes topologias de Redes em Chip. Para isso, uma plataforma multiprocessada foi mapeada em diferentes configurações de Redes em Chip e demonstram como o sistema é afetado pelas métricas de latência e vazão.

Enquanto que (Pestana *et al.*, 2004) apresenta uma estratégia de projeto que busca equilibrar o custo da área ocupada com as métricas latência e vazão. Para a obtenção de números relacionados a custo e desempenho, os autores desenvolveram um simulador que parametriza configurações de Redes em Chip por meio de componentes em XML.

2.3 Qualidade de Serviço (QoS)

O termo "Qualidade de Serviço" se refere à capacidade de uma rede de controlar restrições de tráfego a fim de atender determinados requisitos de uma aplicação ou de um módulo específico da rede (Felicijan, 2004). Este controle acontece por meio dos mecanismos de comunicação discutidos anteriormente neste capítulo.

Ao se implementar mecanismos que suportam QoS, busca-se, primeiramente, garantir a disponibilidade de recursos requeridos tais como largura de banda. De acordo com (Felicijan, 2004) os métodos para prover QoS em Redes em Chip são:(1) oferta redundante de recursos; (2) esquemas de memorização; (3) modelagem do fluxo de dados; (4) controle de admissão; (5) e reserva de recursos.

A reserva de recurso, em geral, ocorre por meio de três técnicas: (i) chaveamento por circuito; (ii) chaveamento por pacote com o uso de canais virtuais; (iii) e arbitragem baseada em prioridades para transmissão de pacotes. Para realizar esta reserva, é importante implementar meios de distinguir fluxos de dados bem como os serviços oferecidos a estes, visto que uma Rede em Chip pode prover comunicação entre módulos heterogêneos.

Algumas implementações de Redes em Chip, como a Æthereal (Gangwal*et al.*, 2005), classificam os serviços oferecidos para uma aplicação em serviço garantido (em inglês, *Guaranteed Service- GS*) e em melhor esforço (em inglês, *Best Effort - BE*). Para isso, a Æthereal utiliza chaveamento por circuito baseado na multiplexação por divisão do tempo para o GS. Embora, nesta rede, todos os fluxos possuam a mesma prioridade, estes podem requerer diferentes reservas de largura de banda.

Redes em Chip que implementam chaveamento por pacote, em geral, utilizam canais virtuais para distinguir fluxos de dados. A Rede QNoC, por exemplo, classifica os fluxos de acordo com quatro tipos diferentes de requisitos de comunicação e define os seguintes serviços: Sinalização, Tempo Real, Leitura e Escrita e Bloco de Transferência. Cada um destes serviços está associado a uma fila de canais virtuais, bem como a uma prioridade, de forma que pacotes com maior prioridade são sempre transmitidos antes dos pacotes com prioridades menores o que assegura determinado nível de latência para este fluxo.

Li *et al.* (2011) afirmam que muitos trabalhos que tratam QoS focam em requisitos de forma individual, não considerando as restrições que um requisito impõe a outros. Por isso, eles propõem um *framework*, baseado em classes de serviços, para suportar a gerência coordenada de dois recursos críticos compartilhados em sistemas embarcados: memória e a rede de interconexão, neste caso as Redes em Chip. O *CoQoS framework* consiste em três etapas fundamentais: a atribuição de classes para cada aplicação; o mapeamento de classe em nível de serviço de acordo com os requisitos de memória, largura de banda e processamento; e a gerência coordenada de recursos compartilhados.

A preocupação com o compartilhamento de recursos em sistemas embarcado com Redes em Chip, também fundamenta a abordagem apresentada em (Grot *et al.*, 2011), que isola os recursos compartilhados em uma ou mais regiões dedicadas da rede. Para isso, os autores propõem uma abordagem baseada em Qualidade de Serviço para definir topologias de Redes em Chip. Uma vez inserido numa destas regiões, o pacote é regulado por mecanismos que suportam QoS.

2.4 Métodos Formais

Os métodos formais consistem num conjunto de técnicas e ferramentas baseado em modelos matemáticos e na lógica formal, que podem ser usados para especificar e

verificar requisitos e projetos de sistemas computacionais. Os métodos formais executam um importante papel em muitas atividades incluindo certificação, reuso e segurança (NASA, 1995).

A natureza qualitativa de uma análise formal é definida pela avaliação das propriedades do sistema que precede a geração do código. Considerando a alta complexidade dos SoCs, principalmente aqueles baseados em Redes em Chip, o poder de abstração de métodos formais representa uma ferramenta valiosa, contribuindo para a minimização de erros nas fases iniciais do projeto.

Essa minimização de erros influencia diretamente na redução de custo do projeto, do tempo de lançamento do produto no mercado e, consequentemente, no aumento da produtividade. Segundo (Rigo *et. al*., 2011), este aumento envolve a combinação de quatro aspectos chave: abstração; reuso; automação e exploração.

No projeto das Redes em Chip, o nível de representação pode ser elevado para diminuir a dificuldade em representar elementos complexos. Formalismos permitem descrições em níveis adequados de abstração, com detalhes que possibilitam refinamentos posteriores para linguagens em níveis mais baixos de abstração. A realização manual destes refinamentos pode aumentar o custo do projeto, por isso existe o esforço de grupos de pesquisa no desenvolvimento de ferramentas que apoiem estes refinamentos.

Além disso, os módulos gerados a partir de uma especificação formal podem ser reutilizados para a validação de diferentes configurações de Redes em Chip. Este reuso permite uma exploração mais ampla de soluções alternativas para problemas relacionados ao desempenho, área e consumo de energia. Assim, a probabilidade de, após o refinamento para níveis mais baixos de abstração, os requisitos exigidos pelo sistema não serem atendidos diminui, evitando o retrabalho.

2.4.1 Formalismos em Projeto de Hardware

Além dos trabalhos citados no início deste capítulo, outras iniciativas demonstram que a base matemática dos formalismos garante especificações concisas, completas, livres de ambiguidade, de fácil manutenção e adequadas para o projeto de hardware (Woodcock e Davies, 1996).

Por exemplo, Dodge *et al.* (1996) utilizam a notação Z para especificar um processador de sinal digital ou DSP (*Digital Signal Processor*) dedicado ao cálculo da transformada rápida de Fourier ou FFT (*Fast Fourier Transform*).

Wezeman (1995) demonstra o uso da notação Z, apresentando um novo método para modelar componentes de rede a partir do ponto de vista do gerenciador da rede. Em (Butterfield *et al*., 2009), o formalismo Z é utilizado para modelar a memória *flash* NAND. A notação Z é utilizada em trabalhos que demonstram que o formalismo é apropriado para descrever complexos sistemas de comunicação ou sistemas com limitações críticas de recursos, como os SoCs. Enquanto, Freitas e Woodcock (2007) especificam em Z, provam e refinam o sistema eletrônico que compõe o *smartcard* Mondex. Ramos (2007) apresenta em a Metodologia CADZ, que utiliza o formalismo Z para analisar qualitativamente as propriedades do sistema de comunicação de uma Rede em Chip, de modo a identificar possíveis falhas nas fases iniciais do ciclo de projeto.

2.4.2 Notação Z

A notação Z é uma linguagem de especificação formal baseada fortemente na teoria dos conjuntos e na lógica matemática, padronizada pela ISO em (ISO, 2002). A linguagem Z é suportada por ferramentas computacionais, como Z-Eves (Saaltink, 1997), que auxilia o processo de criação e verificação e por ferramentas para animação da especificação formal, como Possum (Hazel, 1997).

A forma básica utilizada pela notação Z é denominada esquema (em inglês *schema*), que é usado para introduzir axiomas de funções. Além disso, o esquema é usado para estruturar e compor descrições: unindo, encapsulando e nomeando partes da especificação de forma que estas possam ser reusadas (Woodcock e Davies, 1996).

O modelo de QoS proposto neste trabalho é construído pela especificação de uma série de esquemas usando tipicamente um estilo de transição de estado. Os esquemas são compostos por declaração e predicado, conforme exibe a Figura 2.5. A parte da declaração de um esquema declara as entidades que serão melhor definidas na parte do predicado.

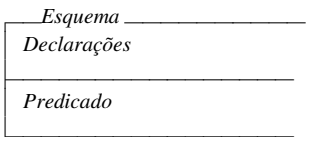


Figura 2.5. Esquema da Notação Z

A teoria dos conjuntos, como mencionado anteriormente, é um dos fundamentos da notação Z onde os tipos básicos são conjuntos definidos no início da especificação. Um dado conjunto inicial, denominado em inglês *given set*, é um poderoso elemento abstrato de Z, representado por nomes dentro de colchetes, do qual uma especificação formal é construída. Conjuntos enumerados também são permitidos na notação Z. Relações e funções são elementos chaves para a modelagem abstrata e estruturas de programação de dados tais como tabelas, listas, matrizes, etc. A lógica de primeira ordem e as operações da teoria dos conjuntos são utilizadas para especificar propriedades dinâmicas e invariantes de dados.

Em geral, a notação Z é utilizada para especificar elementos estruturais de um sistema, podendo a descrição em Z ser complementada com outros formalismos desenvolvidos especificamente para descrever aspectos relacionados a tempo e a concorrência, como o formalismo LOTOS utilizado em(Rocha *et al.*, 2010).

Porém, além de ser notoriamente adequado para descrever aspectos estruturais de um sistema, o formalismo Z pode abranger a descrição de outras propriedades, tais como temporização (Coombes e McDermid, 1993). O tratamento de propriedades temporais de um sistema é uma poderosa ferramenta, visto que o aspecto temporal pode determinar o sucesso ou a falha de um sistema (Kuhma,1989) e deve ser considerado quando se deseja prover requisitos de QoS relacionados à latência.

Coombes e McDermid (1993) apresentam uma estratégia para descrever em Z o comportamento temporal de um sistema por meio da descrição de intervalos, eventos e *time grids*, sendo este último conceito suficiente para prover múltiplos pontos de vista necessários na descrição, por exemplo, de um sistema distribuído.

Percebe-se, então, que este formalismo pode ser utilizado para descrever os aspectos estáticos e dinâmicos de um sistema. Os aspectos estáticos estão relacionados com o estado global do sistema, bem como com os relacionamentos entre seus componentes, também denominado invariante. Os aspectos dinâmicos incluem todas as operações que manipulam os elementos dos estados. Normalmente, sequencias de estados são representadas em Z por estado anterior e posterior, não havendo a representação da transição. Para suprir esta necessidade, Johnson (1995) especifica uma relação que representa operações entre os estados iniciais e finais.

3. MODELO DE QoS PARA PROJETOS DE REDES EM CHIP

O paradigma das Redes em Chip substitui a estrutura física *ad-hoc* dos blocos IP, provendo comunicação que pode ser comparada com ao padronizado pelo Modelo de Referência OSI/ISO ou RM-OSI, representado graficamente na Figura 3.1. Devido às camadas deste modelo permitir tratar separadamente questões como integridade do sinal, confiabilidade dos canais e QoS, o modelo de QoS proposto especifica topologias e mecanismos de comunicação baseado no RM-OSI. Apesar deste modelo de referência não padronizar a comunicação em Redes em Chip, permite agrupar os mecanismos de comunicação especificados e organizá-los em camadas.

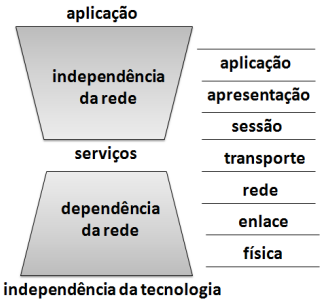


Figura 3.1. Pilha de Serviços do Modelo de Referência ISO/OSI (Baseada em Kogel et al.2006)

No contexto das Redes em Chip, as camadas do RM-OSI de interesse são: Física, Enlace, Rede e Transporte. Estas camadas são descritas brevemente a seguir:

- A camada Física lida com os aspectos elétricos da transmissão de dados, como formato de pulso, voltagens do sinal ou mecanismos para a transmissão de um *bit* de informação. Nesta camada, um dos principais parâmetros é o consumo de energia.
- A camada de Enlace provê, entre outras coisas, o controle de fluxo em nós adjacentes. Este mecanismo determina como um recurso da rede, tais como largura de banda e capacidade de memorização, são alocados para fluxos de dados. O modelo proposto aborda duas técnicas de controle de fluxo: *Handshake* e Baseado em Crédito.
- A camada de Rede implementa estratégias de memorização, mecanismos de chaveamento e arbitragem, bem como algoritmos de roteamento. O modelo proposto inclui técnicas de chaveamento por circuito e por pacote; algoritmos de roteamento determinísticos e adaptativos; árbitros *Round Robin* por Prioridade e por Envelhecimento; e esquema de memorização FIFO.

- A camada de Transporte gerencia conexões fim-a-fim, garantindo, por exemplo, a ordenação de pacotes e a identificação de classes de serviço de fluxos advindos de camadas superiores.

Os mecanismos especificados formalmente para compor o modelo de QoS proposto neste trabalho podem ser visualizados na Figura. A integração destes mecanismos permite alcançar determinados níveis de desempenho, que podem ser analisados por métricas como Latência e Vazão.

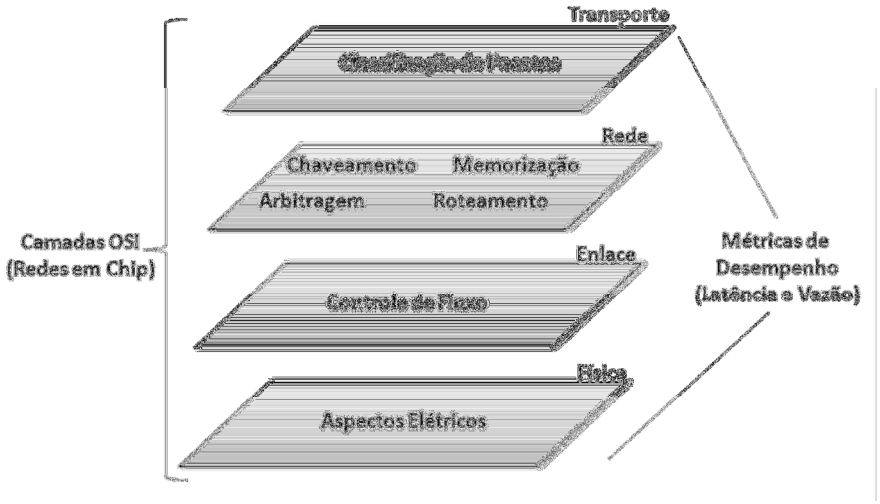


Figura 3.2. Componentes do Modelo Formal de QoS em Camadas

Este trabalho, que estende a metodologia CADZ apresentada em (Ramos, 2007), adiciona especificações de mecanismos essenciais para atender requisitos de Qualidade de Serviço, de modo a facilitar a análise qualitativa destes mecanismos nas fases iniciais do ciclo de projeto.

A Figura 3.3 descreve graficamente as fases da metodologia CADZ e sua inclusão no fluxo tradicional de projeto. Esta metodologia consiste em duas partes, sendo uma formal, constituída das etapas de Modelagem, Integração e Análise Qualitativa, e outra não formal. A especificação proposta por este trabalho está inserida não só nas etapas formais da metodologia CADZ, mas também aprimora a análise qualitativa proposta por e o formalismo adotado permite o refinamento para geração de código executável.

Como demonstra a Figura 3.3, a metodologia CADZ divide a etapa de modelagem em Modelagem da Aplicação e da Arquitetura. Isto permite que a especificação, que compõe o modelo de QoS proposto, aborde os principais problemas a serem considerados no projeto das Redes em Chip.

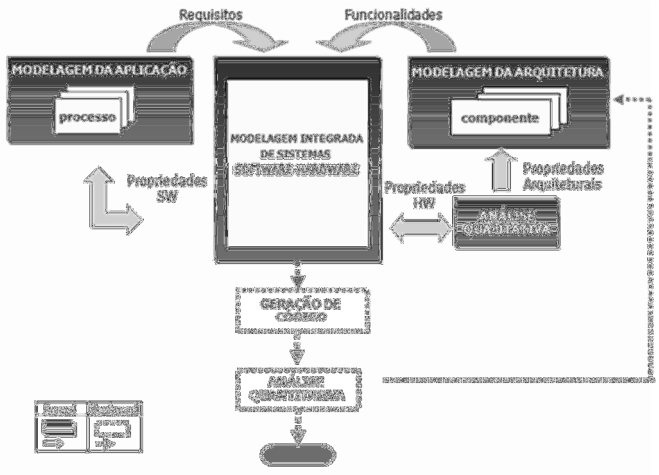


Figura 3.3. Metodologia CADZ (Ramos, 2007)

Segundo Marculescu *et al.* (2009) os principais problemas são: (*i*)Modelagem da aplicação; (*ii*)Definição da infraestrutura de comunicação; (*iii*)Definição dos mecanismos de comunicação; (iv) e Avaliação e validação do projeto.

Com o tratamento destes problemas busca-se contemplar três aspectos fundamentais de um modelo de QoS, a saber:

- *Identificação dos requisitos de QoS*. Para prover serviços diferenciados para determinado fluxo de dados, é necessário implementar meios que diferenciem este fluxo dos demais. Como exemplo, adicionando informação aos pacotes sobre qual classe de serviço ele pertence.
- *Atendimento dos requisitos de QoS pelos mecanismos da rede*. Cada mecanismo de comunicação da rede – roteamento, controle de fluxo, chaveamento, arbitragem, memorização – deve ser capaz de atender individualmente os requisitos de QoS dos tráfegos.
- *Política e Gerenciamento de QoS*. A rede deve ser capaz de identificar se novas conexões com requisitos de QoS podem ser estabelecidas. E, ao prover QoS, a rede deve testar se os requisitos foram efetivamente alcançados. As políticas e gerenciamento de QoS são definidos, em geral, durante o projeto das Redes em Chip.

Considerando que o modelo proposto enfatiza o projeto de hardware, mais especificamente o desenvolvimento de Redes em Chip com QoS, a fase de modelagem da aplicação foi simplificada. De forma que a aplicação é representada por meio de grafos, onde os nós representam processos e as arestas representam os requisitos da aplicação. A definição do grafo que caracteriza uma aplicação pode ser formalizada como abaixo:

Definição 1: Um grafo de caracterização da aplicação G = G(V, A) ou APCG é um grafo direcionado, onde cada vértice $v_{iu} \in V$ representa um processo da aplicação e cada arco direcionado $a_{i,j} \in A$ caracteriza a comunicação de um vértice v_i com um vértice v_j. Cada $a_{i,j}$ pode ser marcado com uma informação específica da aplicação ou uma restrição específica de projeto (por exemplo, latência e largura de banda).

Sendo assim, podem existir diferentes grafos para representar visões específicas dos requisitos da aplicação, ou seja, um grafo pode se referir ao requisito de consumo de energia, outro referente à latência, e assim por diante. A Figura 3.4 exemplifica a representação em grafo de um decodificador de vídeo MPEG4 apresentado em (Pastrnak *et al*., 2006).

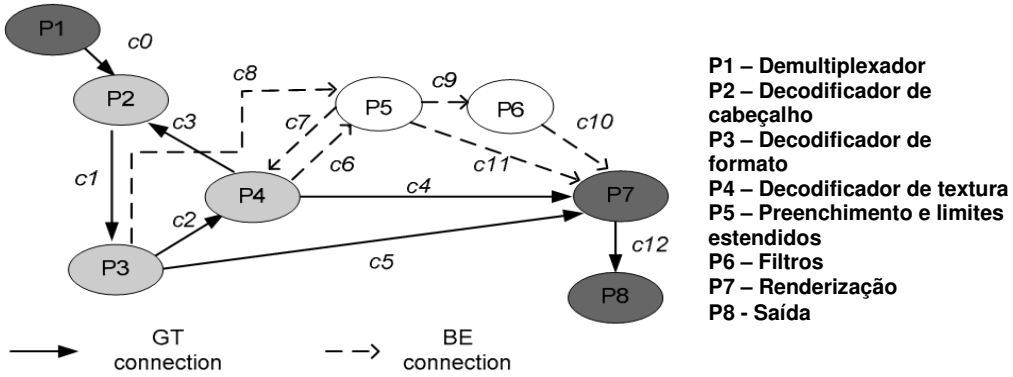


Figura 3.4. Grafo do decodificador de vídeo MPEG4 (Pastrnak et al., 2006)

3.1 Definição da infraestrutura e dos mecanismos de comunicação

A modelagem da arquitetura envolve a especificação formal da infraestrutura de comunicação da Rede em Chip. O espaço de projeto de uma arquitetura de Rede em Chip pode ser definida como em (Ogras *et al*., 2005), de acordo com dimensões distintas, tais como infraestrutura e paradigmas de comunicação:

Definição 2. Diferentes arquiteturas de Redes em Chip podem ser descritas pela tripla $\aleph$ ($A(R,Lk)$, $\Re$, Ω (C)), onde os componentes possuem os seguintes significados:

• O grafo direcionado $A(R,Lk)$ descreve a infraestrutura de comunicação; os roteadores (R) e os *links*(Lk) na rede possuem atributos tais como:

$\forall$ (lk) $\in Lk$, $S(lk)$ informa o *status* do canal da rede.

$\forall$ $r \in R$, $P(r)$ especifica a posição do roteador r.

• $\Re$ (RD (r, s, d, $\rho(n)$), Sw) descreve o paradigma de comunicação adotado pela rede. RD (r, s, d, $\rho(n)$), s, d, $r \in R, n \subseteq R$, define a política de Roteamento no roteador r para todos os pacotes com fonte s e destino d. Nesta função, $\rho(n)$ denota roteadores adjacentes, conceito que pode ser utilizado tanto pelos algoritmos de roteamento quanto pela definição de topologia. E Sw especifica a técnica de chaveamento (*switching*) adotada pela rede.

• $\Omega: C \rightarrow R$ mapeia cada núcleo $c_i \in C$ num roteador r. Para topologias diretas, cada roteador é conectado a um núcleo. Enquanto que para topologias indiretas apenas alguns roteadores são conectados apenas a outros roteadores.

A Figura 3.5representa um modelo conceitual que relaciona todos os mecanismos de comunicação que compõem o modelo de QoS proposto. O esquema informa que uma Rede em Chip é caracterizada por sua topologia e seus mecanismos de comunicação. Por meio destes mecanismos, a rede pode atender requisitos de QoS relacionados à latência e vazão.

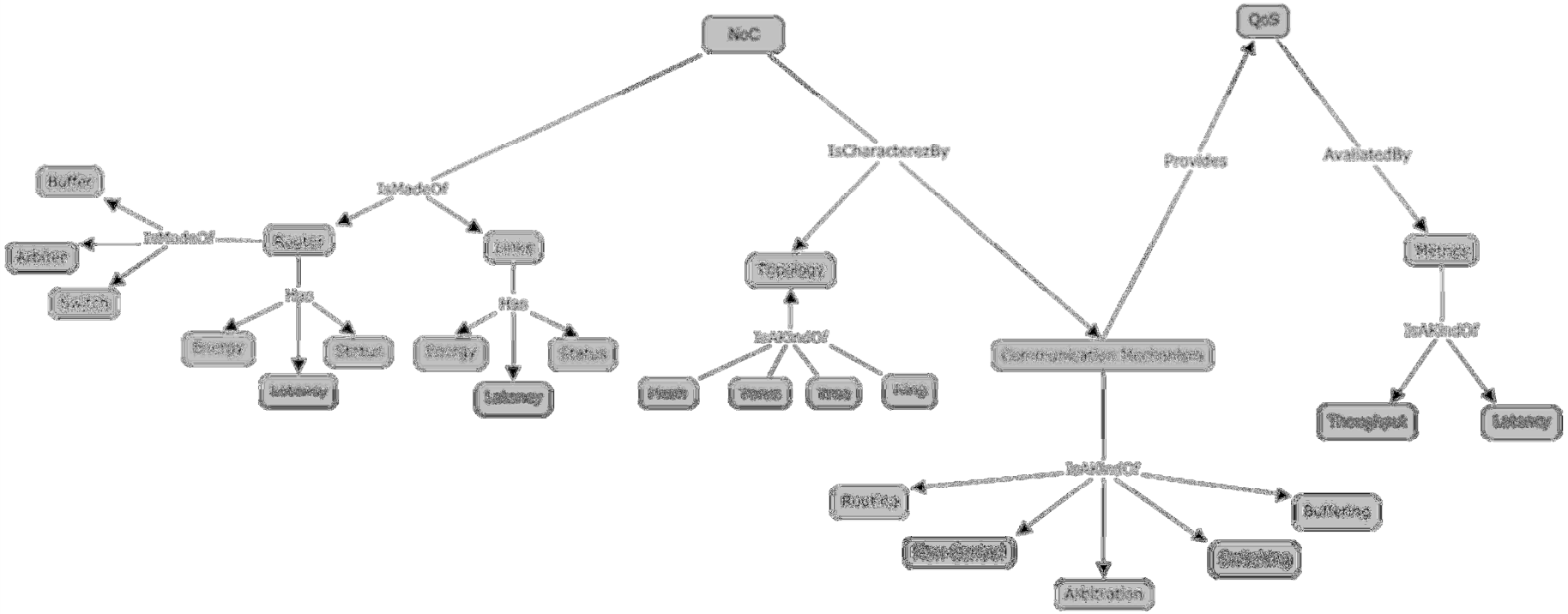


Figura 3.5. Modelo Conceitual

Os projetistas de Redes em Chip lidam com o problema de mapear uma aplicação em uma topologia, como abordam (Murali e De Micheli, 2004). Com o objetivo de auxiliar este mapeamento e a análise de *compromissos*, o modelo de QoS disponibiliza a especificação das topologias mais utilizadas em projetos de Redes em Chip, a saber: *Mesh* Regular e Irregular, *Torus*, Árvore Gorda, Anel, além de permitir a adição de topologias com três dimensões. Uma topologia, bem como o problema de mapear os processos de uma aplicação numa topologia de Rede em Chip,podem ser definidos como abaixo:

Definição3: O grafo da topologia de uma Rede em Chip é um grafo direcionado T(R, Lk) onde cada vértice r ∈ R representa um roteador numa topologia e as arestas direcionadas formam pares (r_i,r_j) ,que representam *links*. As arestas são denotadas por *lk* ∈ Lk, representando uma comunicação direta entre os vértices r_ie r_j.O peso de uma aresta *lk*, denotado por *bw*, representa a largura de banda disponível em *lk*.

O mapeamento do grafo da aplicação apresentado na Definição 1 em um grafo da topologia formalizado na Definição 3 pode ser escrito como:

$$map\ V \rightarrow R\ ,\ map(v_i,r_j)\ ,\ \forall v_i \in V,\ \exists r_j \in R \qquad (1)$$

Durante o projeto de Redes em Chip, um importante aspecto é a escolha do algoritmo de roteamento, visto que este mecanismo interfere diretamente em métricas como latência, vazão, dissipação de energia e confiabilidade. Sendo assim, o modelo de QoS proposto oferece ao projetista a especificação de algoritmos determinísticos e parcialmente adaptativos. Por meio destas especificações pretende-se facilitar a análise de aspectos como otimização de recursos e tolerância à falha. O problema da escolha do algoritmo de roteamento pode ser assim escrito:

Dado um grafo de aplicação APCG (ou CTG) e uma topologia T(R, Lk);

Encontrar o Protocolo de Roteamento PR(r, Src,Dst) que determina o canal de saída no roteador r ∈ R para todos os pacotes que são transmitidos de um roteador fonte Src a um roteador destino Dst;

Tal que O(Arch,G) é otimizado de acordo com Const(Arch,G).

As escolhas dos mecanismos de controle de fluxo e chaveamento também se caracterizam como importantes etapas no projeto das Redes em Chip, visto que estes mecanismos juntos definem a forma em que as mensagens serão encaminhadas na rede. O mecanismo de chaveamento permite, por exemplo, que as mensagens transmitidas

numa Rede em Chip sejam particionadas em unidades menores. Enquanto que o controle de fluxo decide se e quando um pacote deve ser armazenado ou encaminhado.

O modelo de QoS proposto oferece ao projetista as especificações de dois tipos de controle de fluxo: *Handshake* e Controle de Fluxo Baseado em Crédito. O modelo oferece também dois tipos de chaveamento, Por Circuito e Por Pacote, sendo o segundo dividido em SAF, VCT e *Wormhole*. O problema resolvido pelos mecanismos de chaveamento e controle de fluxo pode ser escrito como abaixo:

> **Dada** uma Política de Roteamento PR(r, Src,Dst) no roteador *r*, tal que *r* e r_i formam um *link lk*, que está ao longo da rota seguida por um *flit*;
>
> **Encontrar** um protocolo que multiplexe *flits* advindos de um roteador *r* e encaminhados pelo *link lk*;
>
> **Tal que** *lk* seja maximamente utilizado e o *overhead* em *r* sejam baixos.

Redes em Chip que provêem serviços em diferentes níveis, precisam implementar mecanismos de arbitragem capazes de priorizar fluxos de dados. Por isso, o modelo de QoS especifica a arbitragem *Round Robin* com Prioridade e Arbitragem por envelhecimento. Isto porque a arbitragem *Round Robin* comum não faz diferenciação em relação a níveis de serviços. O problema da priorização de fluxos de dados pode ser assim descrito:

> **Dada** uma arquitetura de Rede em Chip Arch (T(R,Lk),PR,Ω(C)) e um grafo da aplicação APCG (ou CTG);
>
> **Encontrar** a melhor estratégia de escalonamento para priorizar a alocação de recursos para os fluxos gerados pela aplicação.

Ao definir a infraestrutura e os mecanismos de comunicação, o projetista deve levar em consideração a dificuldade em estabelecer *compromissos*, por exemplo, entre vazão e atrasos. Esta preocupação deve existir principalmente em aplicações com *deadlines* de Tempo Real, onde garantias quanto aos atrasos devem ser precisas. Estas garantias estão relacionadas à capacidade de a rede prover QoS para a aplicação. Afim de auxiliar o projetista a definir como melhor prover QoS para uma aplicação, o modelo de QoS especifica cenários que definem possíveis configurações de Redes em Chip e como estas alcançam determinado nível de latência. O problema do suporte a QoS pode ser descrito como:

> **Dada** uma arquitetura de Rede em Chip Arch(T(R,Lk),PR,Ω(C)) e um grafo da aplicação APCG;
>
> **Encontrar** uma estratégia de alocação de recurso;

Tal que o desempenho da rede seja maior ou esteja dentro do limite especificado em {Lmax, Bmax}, onde Lmax é a latência especificada e Bmax é a largura de banda especificada.Esses limites podem ser especificados como funções de diferentes classes de serviço, por exemplo, {Lmax,RT} e {Lmax, nRT}, onde RT representa Tempo Real e nRT representa Tempo não Real.

Depois de tomadas as decisões relacionadas às escolhas do paradigma de comunicação e da infraestrutura da Rede em Chip, é necessária uma avaliação da configuração escolhida para garantir que sua implementação atenderá aos requisitos especificados. O problema de buscar meios de avaliar a configuração de uma Rede em Chip pode ser definido como:

Dada uma arquitetura de Rede em Chip Arch(T(R,Lk),PR,Ω(C)) e um grafo da aplicação APCG (ou CTG);

Encontrar os valores das variáveis de interesse (ex.: latência do pacote, consumo de energia, etc) como uma função das características da arquitetura e da aplicação.

O modelo proposto contribui nesta avaliação possibilitando uma Análise Qualitativa de uma configuração de uma Rede em Chip. Esta etapa corresponde à análise de métricas de desempenho que devem ser atendidas pelo projeto integrado para alcançar QoS. As métricas abordadas pelo modelo proposto são latência, vazão e consumo de energia.

Neste trabalho, as etapas de especificação e prova foram desenvolvidas com o apoio da ferramenta Z/Eves (Saaltink, 1997a; Saaltink, 1997b), que automatiza tarefas como: checagem de sintaxe, de tipos e domínios; cálculo de precondições e prova de teoremas. Para complementar as etapas de especificação e prova, foi utilizada a ferramenta de animação Possum (Hazel, 1997), que permite observar a corretude das operações especificadas. Para o uso desta ferramenta é necessário que a especificação esteja descrita segundo a notação SUM (Cogito, 1999), uma versão simplificada da linguagem Z com suporte aos conceitos de operação, estado e inicialização.

4. ESPECIFICAÇÃO FORMAL DO MODELO PROPOSTO

Este capítulo apresenta as especificações em Z que modelam a arquitetura de uma Rede em Chip, os mecanismos de comunicação, a aplicação, as métricas de desempenho e os cenários que permitem avaliar quais configurações melhor atendem determinados requisitos de QoS. Como mencionado anteriormente, estas especificações passaram pelos processos de prova e animação. Sendo assim, este capítulo também apresenta os resultados obtidos a partir destes processos.

O modelo de QoS proposto estende a especificação de uma Rede em Chip apresentada em (Ramos, 2007). Logo algumas descrições foram mantidas, como os roteadores que são representados pelo *given set* denominado *Router*, os canais ou *links* são representados por pares de roteadores (*Router*, *Router*) e os dispositivos de armazenamento ou *buffers* são representados pelo *given set Buffer*.

O esquema *NoC*, representado na Figura 4.1, contém em sua parte declarativa: o elemento *routers* que define o conjunto de roteadores da *NoC* ($\mathbb{P}$*Router*) e a relação *links* que define o canal entre roteadores ($\mathbb{P}$(*Router*$\times$*Router*)).

Para melhor tratar os requisitos de latência relacionados aos componentes da Rede em Chip, o modelo de QoS proposto associa cada roteador a: um ou vários dispositivos de armazenamento (*Router* $\leftrightarrow$ *Buffer)*; a um valor do tipo enumerado *status* (*Router* $\rightarrow$ *status*); a um valor do tipo Natural ($\mathbb{N}$) que representa o tempo de processamento do roteador (*routerLatency: Router* $\rightarrow\mathbb{N}$); a um valor do tipo Natural ($\mathbb{N}$) que representa a energia gasta para transportar internamente um *bit* entre as portas de um roteador (*routerEnergy: Router* $\rightarrow\mathbb{N}$).

A parte do predicado mostrada na Figura 4.1 exemplifica as restrições para declarar roteadores como sendo componentes dessa *NoC*. Além disso, a figura exemplifica as funções de associação de latência e *status* a um componente, neste caso um roteador.

De acordo com o esquema *NoC*, os dispositivos de memorização e os canais físicos também estão associados a valores de *status*, latência e energia, visto que estes componentes estão diretamente relacionados às métricas de desempenho.

NoC

routers: $\mathbb{P}$ *Router*
links: $\mathbb{P}$(*Router* × *Router*)
routerBuffer: Router ↔ *Buffer*
routerStatus: Router ⇸ *status*
routerLatency: Router ⇸ $\mathbb{N}$
routerEnergy: Router ⇸ $\mathbb{N}$
linkStatus: Router × *Router* ⇸ *status*
linkLatency: Router × *Router* ⇸ $\mathbb{N}$
linkEnergy: Router × *Router* ⇸ $\mathbb{N}$
bufferStatus: Buffer ⇸ *status*
bufferLatency: Buffer × *Buffer* ⇸ $\mathbb{N}$
bufferEnergy: Buffer × *Buffer* ⇸ $\mathbb{N}$

∀ *r1, r2: Router; bf1: Buffer; st1: status;*
ltR1, enR1: $\mathbb{N}$
| *r1* ∈ *routers*
∧ *r2* ∈ *routers...*
∧ *st1 = live...*
• (*r1*, *r2*)∈ *links*
∧(*r1*, *bf1*)∈*routerBuffer*
∧(*r1*, *st1*)∈ *routerStatus*
∧(*r1*, *ltR1*)∈ *routerLatency*
∧(*r1*, *enR1*)∈ *routerEnergy ...*
∧ *r1* ≠ *r2*

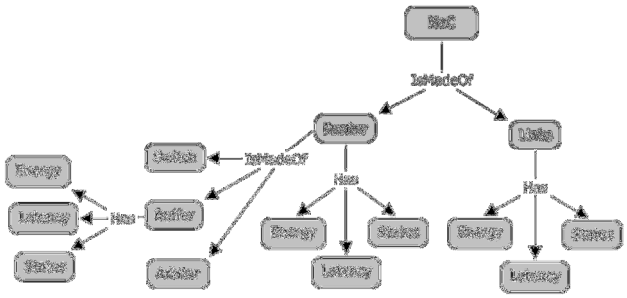


Figura 4.1. Esquema NoC

Como mencionado anteriormente, roteadores, *links* e *buffers* são relacionados a um único valor de *status* que é representado pelo tipo enumerado mostrado na Figura 4.2. Os valores que representam possíveis estados de roteadores são *idle* e *live*. Enquanto que os *links* podem estar associados aos valores *free* ou *busy* e os *buffers* aos valores *full* ou *empty*. Os demais valores representam estados de pacotes.

status ::= *free*| *busy*| *blocked*| *waiting*| *transmited*| *forwarded*| *full*| *empty*| *idle*| *live*

Figura 4.2. Tipo enumerado *status*

4.1 Especificações Formais: Topologias, Mecanismos Básicos de Comunicação e Métricas de Desempenho

O modelo de QoS proposto oferece especificações que permitem avaliar topologias com diferentes valores de métricas de *desempenho* cruciais no atendimento a requisitos de QoS, tais como: largura de banda, latência e diversidade de caminhos (Dally e Towles, 2004). Por meio das especificações também é possível comparar mecanismos de comunicação e outras características da rede de interconexão.

4.1.1 Topologias

Embora o modelo de QoS proposto contemple diferentes topologias, algumas características são comuns a todas elas. Estas características estão especificadas no esquema *Topology* (Figura 4.3), uma extensão do esquema *NoC* que define: as dimensões da rede por meio das variáveis *dimy*, *dimx* e *dimz*; o conjunto de locais possíveis para cada roteador (*placex*, *placexy*, *placexyz*); e a associação entre roteadores e suas posições numa Rede em Chip (*routerplacex*, *routerplacexy*, *routerplacexyz*). Este esquema facilita o reuso da especificação formal e reúne elementos essenciais para determinar como os roteadores estão interconectados.

Topology

NoC
dimx: $\mathbb{N}$
dimy: $\mathbb{N}$
dimz: $\mathbb{N}$
placex: $\mathbb{P}\mathbb{N}$
placexy: $\mathbb{P}(\mathbb{N}\times\mathbb{N})$
placexyz: $\mathbb{P}(\mathbb{N}\times\mathbb{N}\ \times\mathbb{N})$
routerplacex: Router $\rightarrowtail\!\!\!\!\!\!\rightarrow\mathbb{N}$
routerplacexy: Router $\rightarrowtail\!\!\!\!\!\!\rightarrow\mathbb{N}\times\mathbb{N}$
routerplacexyz: Router $\rightarrowtail\!\!\!\!\!\!\rightarrow\mathbb{N}\times\mathbb{N}\ \times\mathbb{N}$

---

dom *routerplacex* = *routers*
dom *routerplacexy* = *routers*
dom *routerplacexyz* = *routers*
ran *routerplacex* = *placex*
ran *routerplacexy* = *placexy*
ran *routerplacexy* = *placexyz*
placex = 0 .. *dimx*
placexy = 0 .. *dimx* - 1×0 .. *dimy* – 1
placexyz = 0 .. *dimx* - 1×0 .. *dimy* – 1 ×0 .. *dimz* - 1

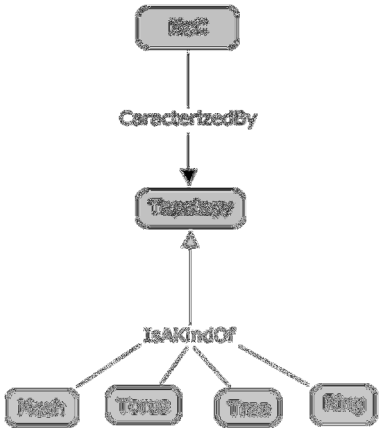


Figura 4.3.Esquema *Topology*

As restrições constantes na parte do predicado da Figura 4.3, as quais devem ser sempre satisfeitas, sentenciam que o domínio das funções *routerplacex*, *routerplacey* e *routerplacexyz* devem ser equivalentes ao conjunto *routers* do Esquema *NoC* e suas imagens devem ser iguais às relações *placex*, *placexy* e *placexyz*.

Essas relações devem respeitar o intervalo entre 0 e as dimensões da rede. Esse esquema pode ser estendido para topologias específicas em uma, duas (2D) ou três dimensões (3D).

A topologia *mesh* de dimensão *n*, por exemplo, é definida formalmente como consistindo de $k_0 \times k_1 \times \cdots \times k_{n-2} \times k_{n-1}$ nós, onde $k_i \geq 2$ é o número de nós ao longo da dimensão *i*. Um nó *X* é representado por *n* coordenadas, $(x_{n-1},x_{n-2},...,x_1,x_0)$, $0 \leq x_i \leq k_i - 1$, $0 \leq i \leq n - 1$. Se *X* e *Y* são vizinhos, então o canal da dimensão *i* do nó *X* está na direção positiva em relação ao nó *Y* quando $x_i = y_i - 1$, ou na direção negativa quando $x_i = y_i + 1$.

A Figura 4.4 mostra a extensão do esquema *Topology* que representa uma rede *Mesh Regular 2D*. Uma rede *Mesh2D* forma-se a partir de no mínimo quatro roteadores (*# routers* ⩾4) e pode ser representada como uma matriz com *x* linhas e *y* colunas, onde cada elemento é definido por uma coordenada que pertence ao conjunto *placexy* ((*x*, *y*) ∈ *placexy*). Logo, cada roteador $R_{(x,y)}$ está posicionado em uma dessas coordenadas, bem como, está conectado a outros roteadores $R_{(x+/-1,y)}$ e $R_{(y,x+/-1)}$. A situação de ausência de nó adjacente caracteriza a *Mesh* Irregular.

RegularMesh2D

Topology

 # routers ⩾4
∀ *x, y:* ℕ| *x* >0∧(*x, y*)∈ *placexy*
• (*routerplacexy* ~(*x, y*), *routerplacexy* ~((*x* - 1), *y*))∈ *links*
∀ *x, y:* ℕ| *x* < *dimx* - 1∧(*x, y*)∈ *placexy*
• (*routerplacexy* ~(*x, y*), *routerplacexy* ~((*x* + 1), *y*))∈ *links*
∀ *x, y:* ℕ| *y* >0∧(*x, y*)∈ *placexy*
• (*routerplacexy* ~(*x, y*), *routerplacexy* ~(*x*,(*y* - 1)))∈ *links*
∀ *x, y:* ℕ| *y* < *dimy* - 1∧(*x, y*)∈ *placexy*
• (*routerplacexy* ~(*x, y*), *routerplacexy* ~(*x*,(*y* + 1)))∈ *links*

Figura 4.4. Esquema *RegularMesh2D*

A prática de customizar uma Rede em Chip de acordo com determinada aplicação, a fim de alcançar melhor desempenho e menor custo, em geral resulta em topologias irregulares, como a *Mesh Irregular*. Isso acontece principalmente devido aos tamanhos e formatos dos módulos interligados pela rede (Bolotin *et al*., 2007).O modelo de QoS proposto especifica a variação irregular da topologia *Mesh*, como mostra a Figura 4.5.

A definição de uma *Mesh Irregular* é idêntica ao da *Mesh* Regular, exceto pelo fato de ser permitida a ausência de alguns roteadores e *links*. O esquema *IrregularMesh2D*, apresentado simplificadamente na Figura 4.5, especifica, por exemplo, que os roteadores $R_{(x,y)}$ e $R_{(x-1,y)}$ podem não estar conectados por meio de um *link*, embora sejam adjacentes.

IrregularMesh2D

Topology

 # routers ⩾4

∃ *x, y:* ℕ| *x* >0∧(*x, y*)∈ *placexy*

• (*routerplacexy* ˜(*x, y*), *routerplacexy* ˜((*x* - 1), *y*))∉ *links ...*

Figura 4.5. Esquema *IrregularMesh2D*

Apesar das vantagens já mencionadas no uso da topologia, segundo Mirza-Aghatabar *et al.* (2007), um dos problemas apresentados nesta disposição é a longa distância entre os nós, que trazem efeitos negativos à latência de comunicação. A topologia toróide ou *Torus* foi proposta com o objetivo de reduzir essas distâncias e consequentemente a latência. Para isso, nesta topologia, os nós localizados em extremidades opostas estão interconectados. Logo, cada nó sempre possui quatro nós adjacentes, como especifica o esquema *Torus2D* do modelo de QoS proposto, mostrado resumidamente na Figura 4.6. Por meio da Figura 4.6, pode ser observado que os roteadores em extremidades, como os $R_{(0,y)}$ e $R_{(x,0)}$, estão interconectados, respectivamente, às suas extremidades opostas, $R_{(dimx\ -1,y)}$ e $R_{(x,\ dimy\ -1)}$.

Torus2D

Topology

 # routers ⩾4

∀ *x, y:* ℕ| *x* = 0∧(*x, y*)∈ *placexy*

• (*routerplacexy* ˜(*x, y*), *routerplacexy* ˜((*dimx* - 1), *y*))∈ *links*

∀ *x, y:* ℕ| *y* = 0∧(*x, y*)∈ *placexy*

• (*routerplacexy* ˜(*x, y*), *routerplacexy* ˜(*x*,(*dimy* - 1)))∈*links ...*

Figura 4.6. Esquema *Torus*

Duato e Ni (1997) apresentam uma classificação para a topologia toróide de acordo com as dimensões da rede. Enquanto a Figura 4.6 especifica a *Torus* com duas dimensões, a Figura 4.7 define a topologia *Torus* unidirecional também chamada de topologia em Anel ou *Ring*, em inglês. O esquema *Ring* define que para a formação desta estrutura é suficiente que o conjunto *routers* possua três roteadores (*# routers* ⩾3) e limita as localizações dos roteadores ao intervalo de zero ao número de nós decrementado de uma unidade (*placex* = 0 .. *# routers* - 1). A disposição dos nós em anel permite a aplicação de estratégias de roteamento *straightforward*, ou seja, o encaminhamento de mensagens na direção horária ou anti-horária, dependendo do caminho mais curto.

Ring

Topology

$\#\ routers \geqslant 3$

$placex = 0\ ..\ \#\ routers - 1$

$\forall\ x: \mathbb{N} \mid\ x > 0 \land x \neq \#\ routers - 1 \land x \in placex$

$\bullet\ (routerplacex\ ^{\sim}\ x,\ routerplacex\ ^{\sim}(x - 1)) \in links$

$\land (routerplacex\ ^{\sim}\ x,\ routerplacex\ ^{\sim}(x + 1)) \in links$

$\forall\ x: \mathbb{N} \mid\ x > 0 \land x = \#\ routers - 1 \land x \in placex$

$\bullet\ (routerplacex\ ^{\sim}\ x,\ routerplacex\ ^{\sim}(x - 1)) \in links$

$\land (routerplacex\ ^{\sim}\ x,\ routerplacex\ ^{\sim}0) \in links$

$\forall\ x: \mathbb{N} \mid\ x = 0 \land x \in placex$

$\bullet\ (routerplacex\ ^{\sim}\ x,\ routerplacex\ ^{\sim}(x + 1)) \in links$

$\land (routerplacex\ ^{\sim}\ x,\ routerplacex\ ^{\sim}(\#\ routers - 1)) \in links$

Figura 4.7. Esquema *Ring*

Além das topologias apresentadas, existem projetos de Redes em Chip que adotam a interconexão multiestágio ou MINs (*Multistage interconnection networks*). Nesta topologia as entradas e saídas dos roteadores conectados pertencem a um determinado nível ou estágio. De forma que a quantidade de estágios e as conexões entre eles determinam a capacidade de roteamento da rede.

As MINs podem ser classificadas em unidirecional e bidirecional, dependendo do tipo de canal utilizado. As MINs bidirecionais ou BMINs (*Bidirectional* MINs) possibilitam que uma informação seja transmitida em direções opostas, suportando três tipos de encaminhamento: *forward*, *backward* e *turnaround* (Duato *et al.*, 1997). A topologia *Fat Tree* pode ser classificada como BMIN, onde os roteadores são representados pelas folhas e pelos vértices internos. Neste caso, a largura de banda de transmissão entre os nós aumenta na medida em que novos *links* são adicionados.

Para especificar a topologia *Fat Tree*, o modelo de QoS altera a especificação da árvore binária apresentada em (Ramos, 2007), como mostra parcialmente a Figura 4.8, de forma que o novo esquema descreva a topologia apresentada em (Gomez *et al.*,2007; Adriahantenaina *et al.*, 2003). Estes autores informam que uma *k-ary n-tree* é uma árvore com *k links* entre um par de nós e com *n* níveis.

No esquema *FatTree*, variáveis do tipo $\mathbb{N}_1$ representam o nível em que um nó se encontra e a ordem do nó na árvore. No predicado são definidas restrições como, por exemplo, relacionados à quantidade mínima de nós para o uso desta topologia. Para o modelo de QoS proposto, o esquema *FatTree* utiliza também as restrições definidas no predicado do esquema *Topology*.

FatTree

Topology
level: $\mathbb{N}_1$
nodes: $\mathbb{N}_1$ …

Figura 4.8. Esquema *FatTree*

4.1.2. Mecanismos de Comunicação

A seguir serão apresentadas as especificações dos mecanismos de comunicação: roteamento, controle de fluxo, chaveamento, arbitragem e memorização, como representam a Figura 4.9 com um trecho do modelo conceitual apresentado no capítulo anterior.

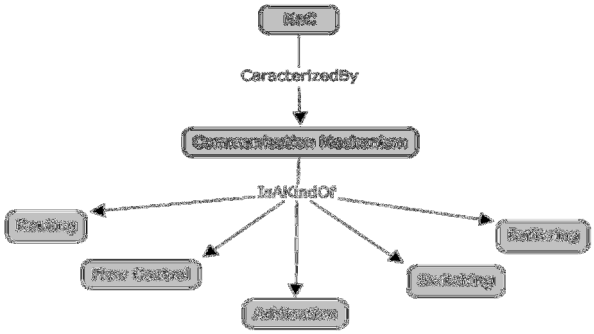


Figura 4.9. Trecho do Modelo Conceitual – Mecanismos de Comunicação

Roteamento

Como mencionado no Capítulo 2, o mecanismo de roteamento pode ser classificado, de acordo com o número de rotas permitidas entre nós destino e origem, em: Determinístico e Adaptativo. A seleção da estratégia de roteamento deve levar em consideração a complexidade de implementação e os requisitos de desempenho que podem ser alcançados.

O modelo de QoS proposto neste trabalho especifica algoritmos de roteamento determinísticos, parcialmente adaptativos e ainda um algoritmo que define um caminho mínimo. As características comuns a todos estão definidas nos esquemas *Routes* e *Routing*, mostrados na Figura 4.10(a) e Figura 4.10(b), respectivamente.

As rotas, especificadas pelo esquema *Routes* (Figura 4.10 (a)), são caminhos entre pares de roteadores que pode incluir *links* intermediários. Sendo assim, a função *routeInfo* identifica cada rota por um identificador exclusivo do tipo *ID*, e a função *routeElements* define o conjunto de todos os *links* intermediários que constituem a rota.

O predicado do esquema define que os identificadores dos pares origem/destino devem ser o mesmo da rota propriamente dita.

No esquema *Routing* (Figura 4.10(b)), a primeira parte inclui as declarações contidas no esquema *Routes* e as variáveis *routingThroughput*, *routingComplexity* e *routingLatency*, as quais representam, respectivamente, o nível de vazão de uma rota determinada por um algoritmo de roteamento, o nível de complexidade de implementação e o nível de latência do roteamento.

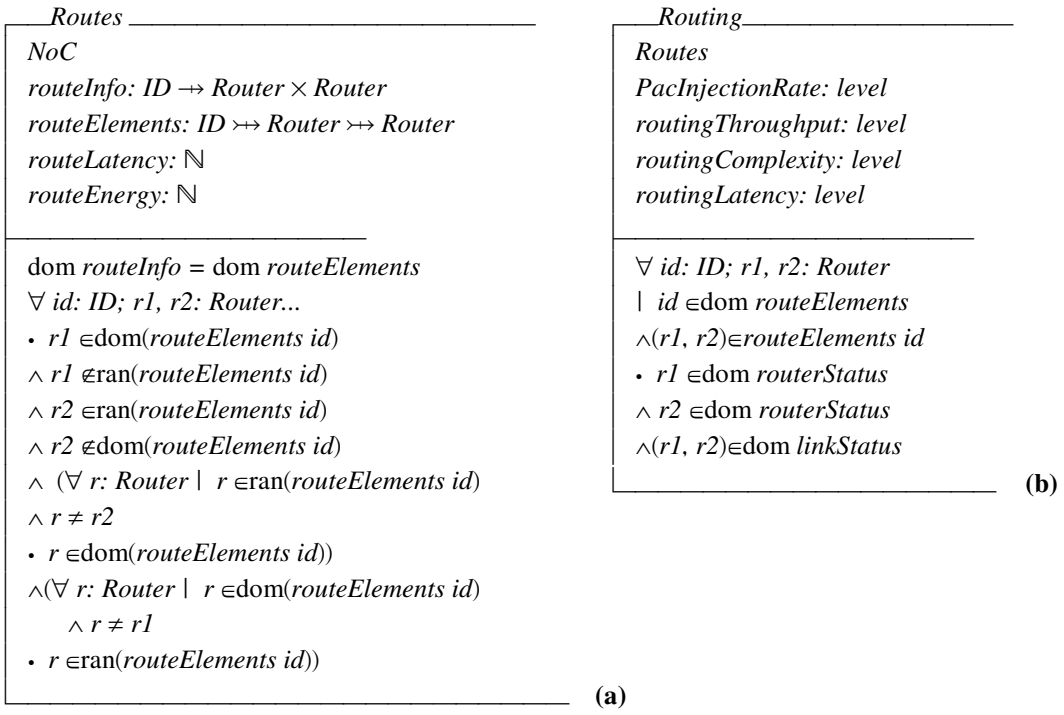


Figura 4.10. (a) Esquema *Routes*; (b) Esquema *Routing*

Assim como em (Ramos, 2007), o esquema *Routing* é estendido de forma que algoritmos determinísticos e adaptativos possuam restrições distintas. As Figura 4.11(a) e Figura 4.11(b) mostram estas extensões que consistem nos esquemas *DeterministicRouting* e *AdaptiveRouting*. Esses esquemas se diferenciam, principalmente, pelos valores atribuídos às variáveis e pela quantidade de rotas entre os nós origem e destino.

O predicado do esquema *DeterministicRouting* restringe o domínio da função *routeInfo* com tamanho unitário (# (dom *routeInfo*) = 1), indicando que entre pares de nós destino/origem é permitido o uso de apenas uma rota. O roteamento determinístico torna-se adequado para as Redes em Chip, principalmente, devido à limitação de recursos neste tipo de rede e aos rigorosos requisitos de latência que esta deve atender.

Em comparação aos algoritmos adaptativos, os determinísticos são menos complexos (*routingComplexity = low*) e exigem menos recursos para a implementação. Por exemplo, na situação aonde os pacotes roteados adaptativamente chegam desordenados em um roteador destino, é necessária memorização adicional para reordená-los. Porém, o uso de uma única rota diminui a adaptatividade da rede em situação de contenção. Neste caso, quando a taxa de injeção de dados é elevada, aumenta-se a quantidade de *links* ocupados e, consequentemente, a vazão da rede diminui e sua latência aumenta (*PacInjectionRate = high* ⇒ *routingThroughput = low* ∧ *routingLatency = high*).

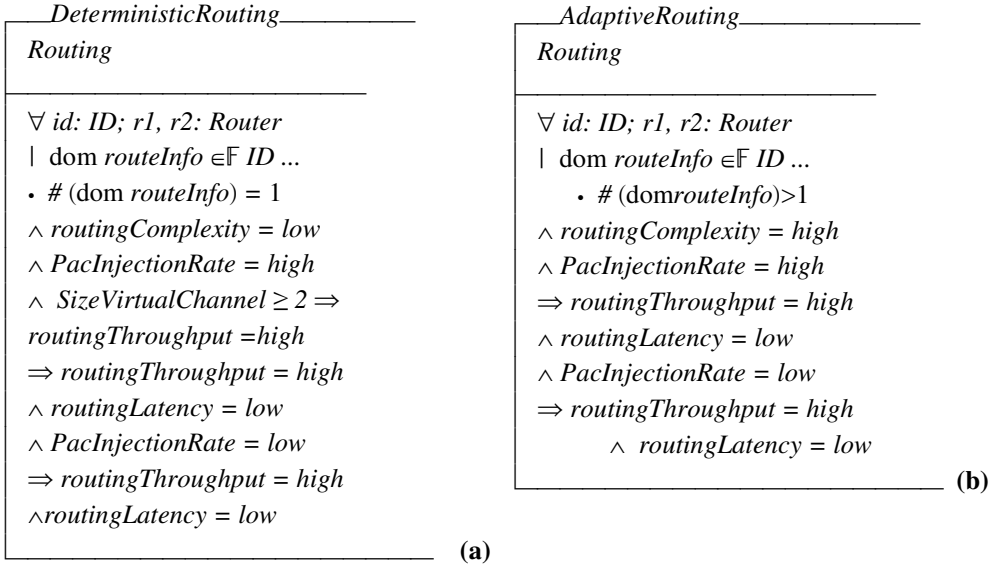


Figura 4.11. (a) Esquema *DeterministRouting*; (b) Esquema *AdaptiveRouting*.

Por meio do esquema *AdaptiveRouting*, percebe-se que apesar da alta complexidade envolvida na implementação do roteamento adaptativo (*routingComplexity = high*), em situações de alta taxa de injeção de dados na rede, a latência pode permanecer baixa e a vazão alta (*PacInjectionRate = high* ⇒ *routingThroughput = high* ∧ *routingLatency = low*), visto que são permitidas mais de uma alternativa de rota (*#* (dom *routeInfo*)>1).

Diferentemente das especificações apresentadas em (Ramos, 2007), que apenas define restrições para distinguir apenas características de adaptatividade, os esquemas *DeterministicRouting* e *AdaptiveRouting* são estendidos neste trabalho de forma a descrever formalmente algoritmos de roteamento específicos, que podem ser aplicados em redes com uma das topologias apresentadas anteriormente.

O esquema *DeterministicRouting* é estendido para a especificação dos algoritmos determinísticos *XY* e *YX*, desenvolvidos para rotear mensagens em redes com topologia *Mesh 2D*. O algoritmo *XY*, especificado no esquema *XYRouting* (Figura 4.12 (a)), roteia pacotes primeiramente na direção horizontal, quando o pacote alcança a coluna do nó destino, este é encaminhado na direção vertical até o receptor.

O esquema *XYRouting* considera que um pacote é enviado de um roteador fonte (*sourceRouter*) a um roteador alvo (*targetRouter*) e para isso atravessa roteadores intermediários representados pelas variáveis *actualRouter* e *nextRouter*. Para a decisão de quais *links* farão parte das rotas, é considerado que o nó destino encontra-se localizado na posição (*xt, yt*) ((*xt, yt*) = *routerplacexy (targetRouter)*) e o nó atual, na posição (*x,y*) ((*x, y*) = *routerplacexy (actualRouter)*). Para exemplificação das regras utilizadas pelo algoritmo, a Figura 4.12 (b) ilustra graficamente a situação em que $x \leqslant xt \wedge y < yt$, logo, a localização do próximo roteador será $(x, y + 1)$ $(x \leqslant xt \wedge y < yt \Rightarrow (x, y + 1) = routerplacexy\ (rnext))$.

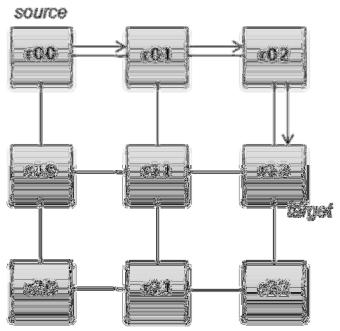
O esquema *XYRouting* considera que um pacote é enviado de um roteador fonte (*sourceRouter*) a um roteador alvo (*targetRouter*) e para isso atravessa roteadores intermediários representados pelas variáveis *actualRouter* e *nextRouter*. Para a decisão de quais *links* farão parte das rotas, é considerado que o nó destino encontra-se localizado na posição (*xt, yt*) ((*xt, yt*) = *routerplacexy (targetRouter)*) e o nó atual, na posição (*x,y*) ((*x, y*) = *routerplacexy (actualRouter)*). Para exemplificação das regras utilizadas pelo algoritmo, a Figura 4.12 (b) ilustra graficamente a situação em que $x \leqslant xt \wedge y < yt$, logo, a localização do próximo roteador será $(x, y + 1)$ $(x \leqslant xt \wedge y < yt \Rightarrow (x, y + 1) = routerplacexy\ (rnext))$.

XYRouting

RegularMesh2D
DeterministicRouting ...

$\forall$ *id: ID; xt, yt, x, y:* $\mathbb{N}$
| ... $(xt, yt) = routerplacexy\ targetRouter$
$\wedge (x, y) = routerplacexy\ actualRouter$
• $(x \leqslant xt \wedge y < yt \Rightarrow (x, y + 1) = routerplacexy\ rnext)$
$\vee (x \geqslant xt \wedge y > yt \Rightarrow (x, y - 1) = routerplacexy\ rnext)$
$\vee (x < xt \wedge y = yt \Rightarrow (x + 1, y) = routerplacexy\ rnext)$
$\vee (x > xt \wedge y = yt \Rightarrow (x - 1, y) = routerplacexy\ rnext)$

(a)



(b)

Figura 4.12. (a) Esquema *XYRouting*; (b) Exemplo gráfico de roteamento XY

Para descrever os algoritmos parcialmente adaptativos que compõem o *Turn Model* (ver Capítulo 2), a saber, *West-First*, *Negative-First* e *North-Last*, o esquema *AdaptiveRouting* foi estendido. A Figura 4.13(a) apresenta o esquema *WestFirstRouting* que especifica o algoritmo *West First*. Considerando que o roteador alvo (*targetRouter*) está localizado na coordenada (*xt, yt*) e o roteador atual (*actualRouter*) possui coordenada (*x, y*), se$yt < y$, então os pacotes são roteados deterministicamente na direção oeste. Porém, se $yt > y$, os pacotes devem ser roteados adaptativamente para leste, norte ou sul, dependendo da disponibilidade dos *links* e roteadores.

WestFirstRouting

AdaptiveRouting
RegularMesh2D ...

$\forall$ *id: ID; x, y, xt, yt:* $\mathbb{N}$
| ... (*actualRouter,nextRouter*)$\in$ *links*
$\wedge$(*xt, yt*) = *routerplacexy targetRouter*
$\wedge$(*x, y*) = *routerplacexy actualRouter*
$\wedge$(*actualRouter,nextRouter*)$\in$dom *linkStatus*
$\wedge$*nextRouter*$\in$dom *routerStatus*
• ($yt < y \Rightarrow$(*x, y* - 1) = *routerplacexy nextRouter*) ...
$\vee(yt > y \wedge xt < x$
$\wedge$*routerStatusnextRouter = busy*
$\vee$ *linkStatus* (*actualRouter,nextRouter*) = *busy*
$\Rightarrow$(*x, y + 1*) = *routerplacexy nextRouter*)
$\vee$(*x, y* + 1) = *routerplacexy nextRouter*)

Figura 4.13. Esquema *WestFirstRouting*;

Os algoritmos descritos anteriormente não são implementáveis em Redes em Chip com topologias diferentes da *Mesh* ou da *Torus*, por isso, o modelo de QoS disponibiliza também a especificação de algoritmos aplicáveis em redes com topologia em Anel ou em Árvore.

A Figura 4.14(a) apresenta, resumidamente, o algoritmo de roteamento *Turn Around* para a topologia em árvore, especificado pelo esquema *TurnAroundRouting*. Nesta estratégia, para o envio de um pacote de um roteador para outro, primeiramente, ele é encaminhado para o menor nível em comum acima e depois encaminhado até o nó destino.

TurnAroundRouting

FatTree
DeterministicRouting
Forward: Router $\rightarrowtail\!\!\!\!\!\rightarrow$ *Router* $\rightarrowtail\!\!\!\!\!\rightarrow$ *Router…*

$\forall$ *id: ID; ractual, rnext: Router; l, n, lt, nt:* $\mathbb{N}$
| … (*lt, nt*) = *routerplacexy target*
• (... (*ractual, rnext*)$\in$*routeElements id* $\wedge$ *lt* $\leqslant$ *l*
$\wedge$ *l* mod2 = 0 $\Rightarrow$(*l* - 1, *n* div2) = *routerplacexy rnext*)...

Figura 4.14. (a) Esquema *TurnAroundRouting*

Para Redes em Chip com a disposição dos nós em anel, o modelo de QoS proposto especifica a estratégia de roteamento *straightforward*, ou seja, o encaminhamento de mensagens na direção horária ou anti-horária, dependendo do caminho mais curto. Para isso o esquema *StraightforwardRouting* (Figura 4.15) estende o esquema *Minimal*, que traz as regras para determinar o caminho mais curto.

StraightforwardRouting

Ring
Minimal …

$\forall$ *id: ID; l, lt:* $\mathbb{N}$
| *... lt = routerplacex targetRouter*
$\wedge$ *l = routerplacex actualRouter*
• (*... lt*> *l* $\Rightarrow$ *l* + 1 = *routerplacex nextRouter*)
$\vee$(*... lt* > *l* $\Rightarrow$ *l* - 1 = *routerplacex nextRouter*)

Figura 4.15. Esquema *StraightforwardRouting*

Controle de Fluxo

A escolha da técnica de controle de fluxo deve fundamentar-se na necessidade de minimizar os requisitos de memorização e a latência na Rede em Chip. O modelo de QoS especifica três tipos de controle de fluxo: *HandShake*, já apresentado em (Ramos, 2007); Baseado em Crédito e Canal Virtual.

O mecanismo representado graficamente no Capítulo 2 é especificado pelo esquema *CreditBasedFlowControl* (Figura 4.16), com o aprimoramento proposto em (Radulescu *et al.*, 2004). Os autores propõem que a atualização do contador aconteça apenas quando não haja recurso disponível, a fim de diminuir o *overhead* causado pela transmissão de sinais de controle.

CreditBasedFlowControl

PacketSwitching
targetBuffer: Buffer $\rightarrow\!\!\!\!\!\!\cdot$ *Packet*
sourceBuffer: Buffer $\rightarrow\!\!\!\!\!\!\cdot$ *Packet*
bufferRouter: Buffer $\rightarrow\!\!\!\!\!\!\cdot$ *Router*
fcontrolLatency: level
credit: $\mathbb{N}$

$\forall$ *ractual, rnext: Router; bfout, bfin: Buffer;*
pck1, pck2: Packet
| *...*(*bfout, pck1*)$\in$*sourceBuffer*
$\wedge$(*bfin, pck2*)$\in$ *targetBuffer*
• (*credit* = 0$\wedge$ *sourceBuffer* $\neq\emptyset\Rightarrow$
packetStatus pck1 = waiting)
$\vee$(*credit* = 0 $\wedge$(*sourceBuffer* = $\emptyset\vee$ *sourceBuffer* $\neq\emptyset$)
$\wedge$ *packetStatus pck2 = forwarded* $\Rightarrow$ *credit = credit* + 1)

Figura 4.16. Esquema *CreditBasedFlowControl*

O esquema *CreditBasedFlowControl* considera que o chaveamento utilizado é do tipo por Pacote (*PacketSwitching*). Neste esquema, os *buffers* ocupado por pacotes ou *flits* no roteador emissor é representado pela função *sourceBuffer,* enquanto os *buffers* no roteador receptor são representados pela função*targetBuffer*. A associação entre *buffers* e roteador é feita por meio da função *bufferRouter*. Considerando que existe um pacote a ser enviado de um roteador fonte ((*bfout*, *pck1*) $\in$ *sourceBuffer*) para um roteador destino e que neste segundo existe também um pacote armazenado ((*bfin*, *pck2*) $\in$ *targetBuffer*). Além disso, o contador no nó fonte é representado pela variável *credit*. O predicado do esquema *CreditBasedFlowControl* prevê apenas duas situações no controle do fluxo:

(i) *credit* = 0$\wedge$ *sourceBuffer* $\neq\emptyset$.Quando o contador de crédito está nulo e existe pacote no nó fonte a ser enviado, então, este pacote deve aguardar o incremento do contador(*packetStatus pck1 = waiting*).

(ii) *credit* = 0 $\wedge$ (*sourceBuffer* = $\emptyset$ $\vee$ *sourceBuffer* $\neq\emptyset$). Independentemente de existir algum pacote no nó transmissor, quando algum pacote no nó receptor é encaminhado (*packetStatus pck2 = forwarded*), então o contador de crédito no transmissor é incrementado (*credit = credit* + 1).

No controle de fluxo *Handshake* (Duato, 1997), a transmissão de dados acontece após uma negociação entre roteadores origem e destino. Esta negociação pode ser representada graficamente como na Figura 4.17 que mostra o seguinte: Quando um roteador destino, representado por B, está pronto para receber um *flit*, ele atribui valor *low* ao sinal de controle (Figura 4.17(a)). Quando um roteador origem, representado por A, está pronto para enviar, ele eleva o sinal para *high* e transmite o *flit* através do canal (Figura 4.17(b)). Enquanto o *flit* estiver sendo recebido por B, um sinal de controle *Request*/*Acknowledge* (R/A) é mantido com valor *high* (Figura 4.17 (c)). Depois que o *flit i* é removido do *buffer* de B (Figura 4.17 (d)), o ciclo se repete para a transmissão do próximo *flit* (*i* + 1) até que todo pacote seja recebido. Este mecanismo pode ser representado como no esquema *Handshake* (Ramos, 2007).

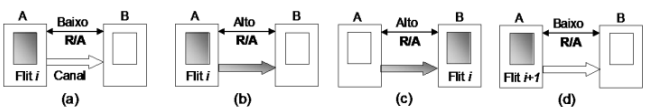
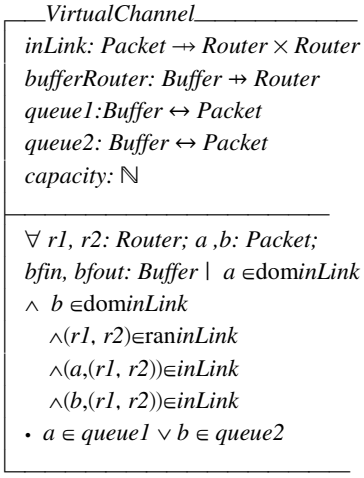
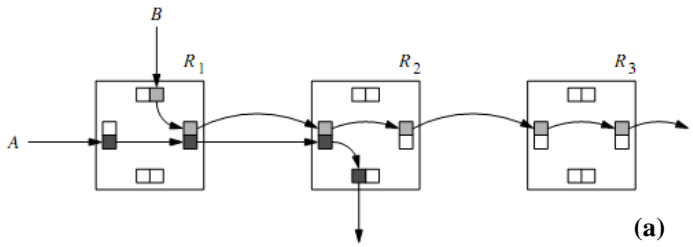


Figura 4.17. Controle de Fluxo Handshake

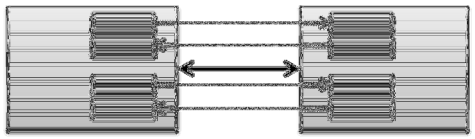
As especificações dos mecanismos de controle de fluxo apresentadas até agora consideram que um único *buffer* está associado a um canal físico. A associação de múltiplos *buffers* a cada *link* por meio da implementação dos chamados Canais Virtuais é descrita pelo esquema *VirtualChannel* (Figura 4.18). O canal virtual foi originalmente concebido com o objetivo de resolver problemas de *deadlocks* nas redes de interconexão (Dally, 1992). Porém, os canais virtuais têm sido utilizados hoje com a principal finalidade de prover tratamento diferenciado entre fluxos de dados e otimizar o uso dos recursos da rede. A Figura 4.18(c) mostra duas mensagens atravessando o canal físico entre os roteadores R1 e R2. Sem canal virtual, o encaminhamento da mensagem B só poderá ocorrer após todo o envio da mensagem A. No entanto, para cada canal físico existem dois canais virtuais multiplexados. Por meio desta multiplexação, os envios de ambas mensagens, compostas por *flits*, progredirão.



(b)



(a)



(c)

Figura 4.18. (a) Canais Virtuais; (b) Esquema *VirtualChannel*; (c) Uso de canais virtuais

Arbitragem

O mecanismo de arbitragem soluciona problemas de fluxos que concorrem por um mesmo recurso, mostrando-se fundamental na garantia do atendimento de requisitos de QoS para determinada mensagem. As especificações formais propostas em (Ramos, 2007) descrevem todas as abordagens constantes em (Hwang, 1993), que estabelece quatro estratégias para resolver a colisão entre dois pacotes: Armazenamento Temporário, Bloqueio, Descarte e Desvio. Dentre estas estratégias, o modelo de QoS condirá o Armazenamento Temporário e o Bloqueio.

Para aprimorar as especificações destas duas estratégias propostas em (Ramos, 2007), o modelo de QoS adiciona restrições de prioridade para que o árbitro selecione com mais rigor qual pacote deve ser bloqueado e armazenado temporariamente. A

adição de novas restrições deu origem às especificações dos mecanismos de Arbitragem por Prioridade e por Envelhecimento de Pacote.

A Figura 4.19(a) mostra o esquema *PriorityArbitration*, que considera a situação onde dois pacotes aguardam em um mesmo *Buffer* numa porta de saída de um roteador (*outBuffer: Packet ⇸ Buffer*) o encaminhamento para uma porta de entrada (*inBuffer: Packet ⇸ Buffer*) em um próximo roteador. Cada pacote concorrente possui uma prioridade (*packPriority: Packet ⇸ℕ*) que varia de acordo com a classe de serviço a qual pertence esse pacote (*classPriority: ServiceClass →ℕ*). O predicado do esquema define que, em um momento de contenção, a transmissão do pacote com menor prioridade deve ser interrompida (*pr1 > pr2 • preemptedPack = p2*).

O esquema *PriorityArbitration* considera a disciplina de prioridade preemptiva pré-vista pela Teoria das Filas e da Simulação (Prado, 1999). Neste caso, ao pacote com a mais alta prioridade é permitido entrar em transmissão independentemente de outro cliente com menor prioridade estar sendo servido, de forma que, o pacote com menor prioridade é interrompido e tem sua transmissão reiniciada posteriormente. Quando reiniciada a transmissão, o pacote volta a ser transmitido do ponto onde parou.

Para tratar a arbitragem de pacotes de uma mesma classe de serviço, ou seja, com a mesma prioridade, o modelo de QoS proposto possui o esquema *AgingArbitration* (Figura 4.19 (b)). Este esquema especifica a arbitragem por prioridade baseado em envelhecimento (Corrêa *et al*., 2007; Berejuck e Zeferino, 2009).

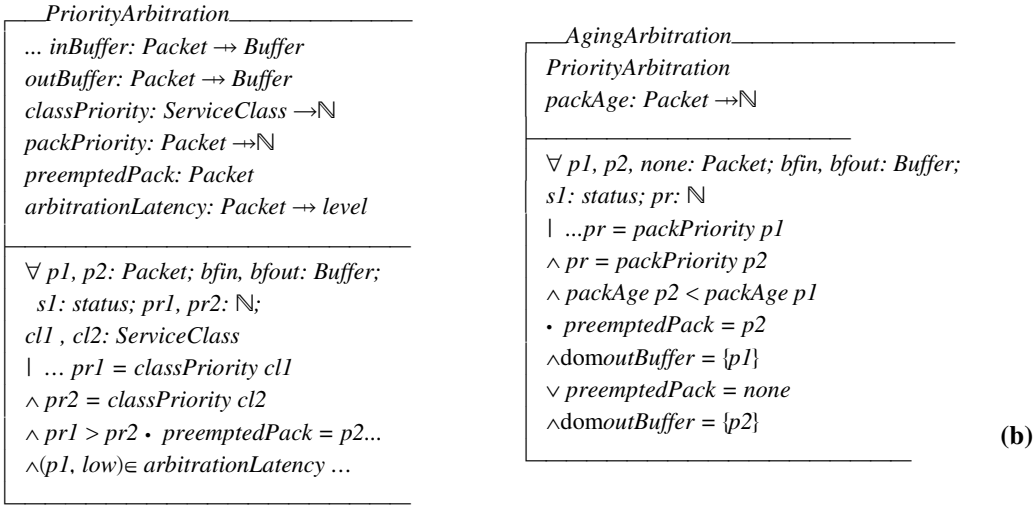


Figura 4.19. (a) Esquema *PriorityArbitration*; (b) Esquema *AgingArbitration*

A Figura 4.20 traz o esquema *RoundRobinArbitration* que especifica que, quando é concedido o acesso a um recurso a um *flit* de determinada classe, no ciclo

posterior, outro *flit* desta mesma classe terá a menor prioridade, como em (Zeferino, e Susin, 2003).

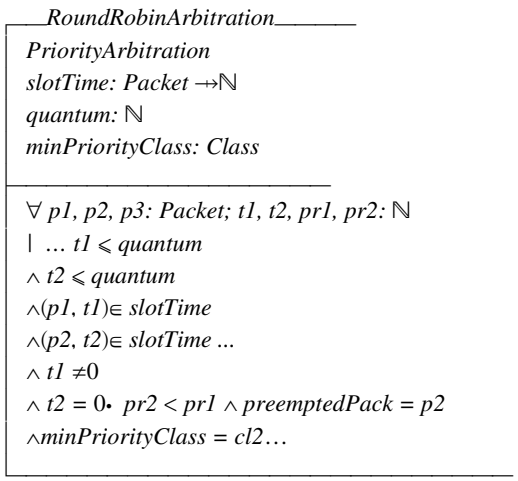


Figura 4.20. Esquema *RoundRobin*

Chaveamento

A estratégia utilizada por cada tipo de chaveamento para conectar *buffers* nas portas de entra e saída de roteadores e encaminhar pacotes internamente determina o tempo gasto para a conclusão deste processo.

Na técnica de chaveamento por circuito, um caminho físico é reservado do nó origem até o nó destino para a transmissão de uma mensagem. Esta reserva é realizada a partir da injeção do *flit* cabeçalho na rede, que além de informações de roteamento contém dados para controle. Quando este primeiro *flit* alcança o destino, toda a rota percorrida por ele está reservada para o encaminhamento dos demais. Esta estratégia de chaveamento é especificada pelo esquema mostrado na Figura 4.21.

Visto que o chaveamento por circuito objetiva alocar recursos ao longo de uma rota, a Figura 4.21 mostra que o esquema *CircuitSwitching* depende do esquema *Routes*, que especifica regras para a formação de rotas na rede. O esquema *CircuitSwitching* especifica restrições relacionadas à quantidade de *buffers* utilizados, complexidade de implementação, vazão e latência.

O chaveamento por circuito possui baixa complexidade de implementação, por isso em seu predicado o esquema *CircuitSwitching* atribui o valor *low* à variável *switchingComplexity*. Além disso, este esquema define que, por estabelecer um caminho dedicado, o chaveamento por circuito não exige memorização (*bufferSize = none*) e

que os *links* que compõem o caminho reservado estão associados ao valor *busy* de *status* (*linkStatus* (*r1*, *r2*) = *busy* ∧ *path* = *dedicated*).

CircuitSwitching

Routes
path: pathType
bufferSize: Size
switchingThroughput: level
switchingComplexity: level
switchingLatency: level

---

switchingComplexity = low
bufferSize = none
∀ *id: ID; r1, r2: Router*
| *id* ∈dom *routeElements*
∧(*r1*, *r2*)∈ *routeElements id*
∧(*r1*, *r2*)∈dom *linkStatus*
• *linkStatus* (*r1*, *r2*) = *busy* ∧ *path* = *dedicated*

Figura 4.21. Esquema *CircuitSwitching*

O elemento *switchingLatency* não está restrito a um valor, devido à possibilidade de variação da latência, uma vez que a latência no chaveamento de circuito envolve o tempo de estabelecimento do circuito e o tempo da transferência da mensagem. Se na fase de estabelecimento do circuito existir algum *link* desejado com o *status* ocupado (*busy*), então haverá uma espera pela liberação desse canal, acarretando em uma latência média ou baixa (*medium* ou *high*), sendo esta uma das principais desvantagens no uso do chaveamento por circuito.

Ainda que durante a fase de alocação de recurso não aconteça bloqueio, o processo de requisição de recurso, confirmação e alocação representa um aumento significativo na latência. Este processo também interfere negativamente na vazão, pois o tempo em que os *links* ficam reservados para transmissão é maior do que o tempo em que eles são efetivamente utilizados.

Uma alternativa ao chaveamento por circuito é a adição de *buffers* nas entradas e saídas dos roteadores, de forma que os dados a serem transmitidos sejam armazenados enquanto o recurso requerido não está disponível. Cada pacote que compõe uma mensagem pode ser roteado individualmente do nó origem até o destino e nisto consiste o chaveamento por pacote, especificado pelo esquema *PacketSwitching* (Figura 4.22 (a)). Este esquema apresenta funções que associam *status* aos pacotes (*Packet* ⇸ *status*) ou aos *flits* (*Flit* ⇸ *status*) que compõem uma mensagem e define que os links que pertencem à rota a ser seguida por um fluxo não estão ocupados durante toda a

transmissão, visto que o caminho não é dedicado (*linkStatus*(*r1*, *r2*) = *busy* ∨*linkStatus* (*r1*, *r2*) = *free*) ∧ *path* = *noDedicated*).

A especificação da técnica de chaveamento SAF (ver Capítulo 2) estende o esquema *PacketSwitching*, como mostra o esquema *StoreAndForward*. Esta técnica exige grande quantidade de recursos de armazenamento (*bufferSize* = *large*) e a espera do armazenamento de toda a mensagem em cada nó que compõe uma rota representa aumento na latência (*switchingLatency* = *high*).

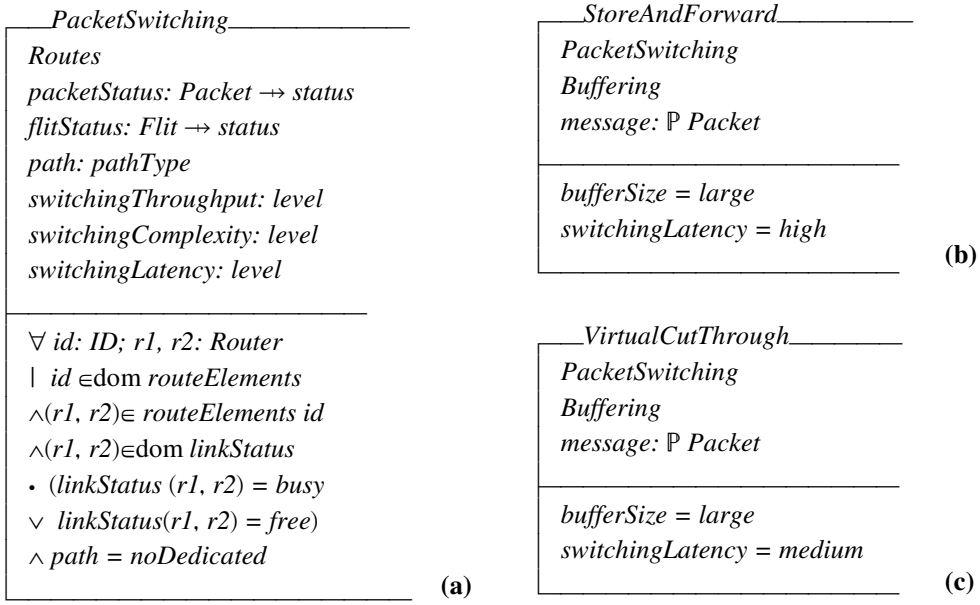


Figura 4.22. (a) Esquema *PacketSwitching*; (b) Esquema *StoreAndForward*; (c) Esquema *VirtualCutThrough*

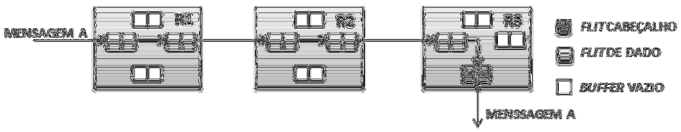
Com o objetivo de oferecer baixa latência de comunicação e reduzir os requisitos de *buffers*, o chaveamento *Wormhole* (Figura 4.23 (a)) tem sido usado em quase todas as novas gerações de redes de interconexão. Este chaveamento opera como o *Virtual-Cut-Through*, porém cada pacote é serializado em uma sequencia de *flits*, onde o *flit* cabeçalho orienta os demais *flits* ao longo da rota. Consequentemente, os tamanhos dos *buffers* são substancialmente menores em relação aos utilizados pelo VCT (*bufferSize* = *small*).

Como mostra a Figura 4.23(b), os demais *flits* seguem o rastro do *flit* cabeçalho. Se o cabeçalho encontrar algum *link* ocupado, todos os restantes terão que aguardar o encaminhamento. Esta característica do chaveamento *Wormhole* requer que este mecanismo seja combinado com estratégias que evitem *deadlock*.

Wormhole

PacketSwitching...
message: seq *Flit*

bufferSize = small
∀ *r1, r2: Router; id: ID*
| ...(*r1*, *r2*)∈*routeElements id*
∧(*r1*, *r2*)∈dom *linkStatus*
• ((*r1*, *r2*), *busy*)∈ *linkStatus*
⇒ *switchingLatency = high* ∨
((*r1*, *r2*), *busy*)∉*linkStatus*
⇒ *switchingLatency = low*



(a) **(b)**

Figura 4.23. (a) Chaveamento por Pacote *Wormhole*; (b) Representação gráfica do chaveamento *Wormhole*

Memorização

A estratégia de memorização é um fator preponderante no projeto de Redes em Chip, visto que um dos elementos que mais influencia no consumo de área e de energia é o armazenamento temporário ou *buffer* (Guerrier e Greiner, 2000).

O tipo de aplicação tem relação direta com a profundidade mínima de um *buffer*, pois, dependendo do tamanho dos pacotes que essa aplicação gera, um tamanho maior dos *buffers* pode permitir uma menor contenção na rede. Alguns autores (Bolotin, 2004; Bjerregaard, 2006) mostram que esse aumento na profundidade dos *buffers* não é uma solução para evitar contenção, pois o benefício no desempenho é inferior ao aumento na área e no consumo de energia. Entretanto, segundo (Bjerregaard, 2006), um aumento na profundidade dos *buffers* pode ser útil para absorver tráfego em rajadas. Segundo (Felicijan e Furber, 2004), o gerenciamento da memorização temporária deve tratar efetivamente de questões de QoS, tais como: (*i*) prover espaço de memorização suficiente para acomodar qualquer excesso de tráfego nas entradas e saídas dos roteadores; (*ii*) e assegurar que não ocorrerão situações de bloqueio de cabeça de linha dos pacotes.

Para apoiar o gerenciamento da memorização e tratar questões como tamanho de *buffers* e esquemas de distribuição de memórias nos roteadores o modelo de QoS proposto adiciona o esquema *Buffering* às especificações apresentadas em (Ramos, 2007), como mostra a Figura 4.24(a). Este esquema especifica as restrições para as estratégias de memorização centralizada (# (ran*routerBuffer*) = 1) e distribuída (# (ran *routerBuffer*)>1) e ainda considera o uso de *buffers* de tamanhos diferenciados. O

esquema *Buffering* é estendido de forma a abranger a estrutura *First In- First Out*, como mostra o esquema *FIFOBuffering* (Figura 4.24(b)).

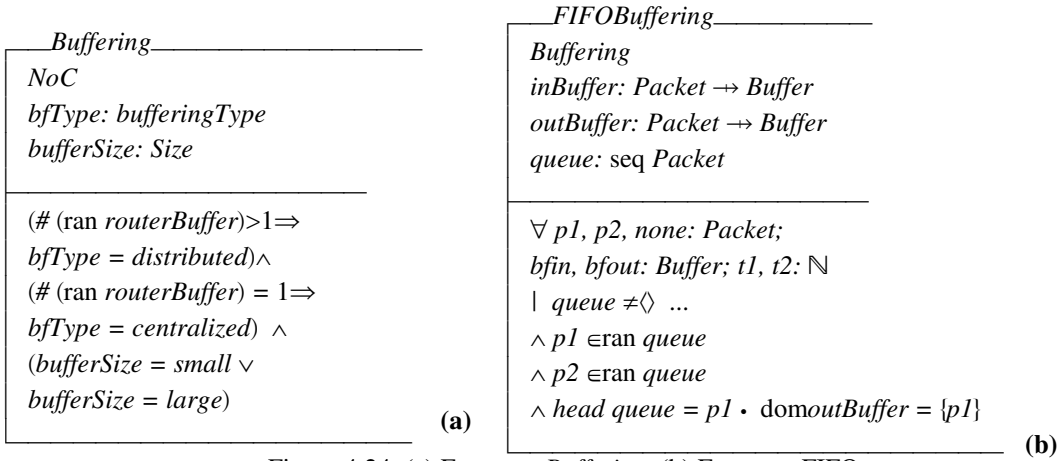


Figura 4.24. (a) Esquema *Buffering*; (b) Esquema FIFO

4.1.2 Especificação das Métricas de Desempenho

A comunicação numa rede de interconexão como a Rede em Chip é caracterizada por diferentes compromissos que levam em consideração características como a latência, a vazão e a energia dissipada (Pande *et al*., 2005). O modelo de QoS fornece meios para a análise qualitativa de diferentes configurações de Redes em Chip a partir da formalização destas métricas: Latência e Vazão.

O esquema *Latency* (Figura 4.25 (a)) é constituído pela função *routerLatency* que define o tempo gasto pelo roteador para rotear um pacote, pela função *linkLatency* que define o tempo de propagação do pacote através do *link* entre dois roteadores. O tempo de envio de um pacote armazenado no *buffer* de entrada para o *buffer* de saída é representado pela função *bufferLatency*. A variável *routeLatency* representa a soma de todos os atrasos existentes na comunicação entre roteadores adjacentes. A função *limitLatencyLevel* associa um valor ao nível de atraso oferecido pela rota.

Enquanto que a métrica de vazão é tratada pelo esquema mostrado na Figura 4.25(b), *Throughput*, que associa a cada *link* de uma rota o valor de vazão oferecida (*linkThroughput*).A vazão máxima ou mínimaoferecida pela rede é expressa por meio da função *limitThroughputLevel*. Este valor é posteriormente comparado ao nível de vazão requerido pela aplicação.

Como o valor total de vazão depende da rota escolhida pelo algoritmo de roteamento, o esquema *Routing* apresenta a relação *routingThroughput*, que permite associar a cada tipo de algoritmo um valor de vazão.

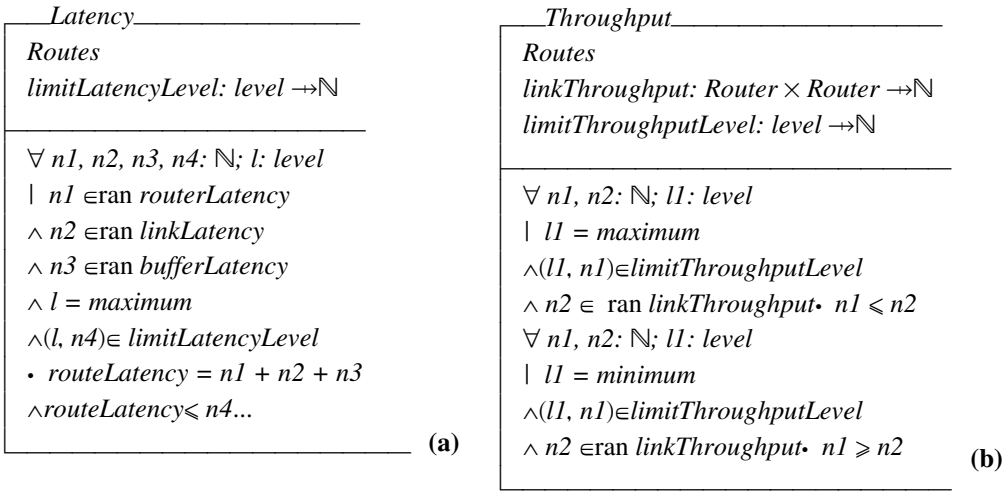


Figura 4.25. (a) Esquema *Latency*; (b) Esquema *Throughput*

5. CENÁRIOS, PROVA E ANIMAÇÃO

Alta vazão e baixa latência são características desejáveis numa Rede em Chip. O modelo de Que permite especificar diferentes configurações de Redes em Chip para auxiliar o projetista na análise qualitativa das características relacionadas à latência e vazão. As configurações apresentadas neste capítulo compõem cenários que ilustram as etapas de prova e animação necessárias para validar as especificações desenvolvidas.

As etapas necessárias para a construção dos cenários, desde a escolha da disposição dos nós até a análise de desempenho da rede, foram animadas por meio da ferramenta Possum (Hazel, 1997).

Cada um dos cenários desenvolvidos objetiva atender requisitos de latência e de vazão exigidos por diferentes aplicações. Estes requisitos são modelados como um conjunto enumerado e inseridos na modelagem da aplicação por meio da relação *requirementsLevel* no esquema *Application* (Figura 5.1).

Application

$processes: \mathbb{P}\ Process$
$communicationProcess: \mathbb{P}(Process \times Process)$
$label: Process \times Process \rightarrow\!\!\!\!\!\!+ \mathbb{N}$
$requirementsLevel: \mathbb{P}(requirements \times level)$

$processes \in \mathbb{F}\ Process$
$\mathrm{dom}\ label = communicationProcess$
$\forall\ p1, p2: Process \mid (p1, p2) \in communicationProcess$
$\bullet\ p1 \in processes \wedge p2 \in processes \wedge p1 \neq p2$

Figura 5.1. Esquema *Application*

Este esquema considera a definição 1, apresentada no capítulo 3, que descreve uma aplicação como um grafo. Sendo assim, neste esquema são descritos: o conjunto de processos, representados pelos vértices do grafo ($processes: \mathbb{P}\ Process$); a comunicação entre os processos, representados pelas arestas dos grafos ($communicationProcess: \mathbb{P}(Process \times Process)$); os rótulos das arestas ($label: Process \times Process \rightarrow\!\!\!\!\!\!+ \mathbb{N}$);e o requisito da aplicação seguido do seu nível de exigência ($requirementsLevel: \mathbb{P}(requirements \times level)$). O fator determinante na escolha dos mecanismos básicos da rede serão os requisitos da aplicação.

A seguir é apresenta o processo de prova das especificações formais por meio de cenários que combinam diferentes topologias e mecanismos de comunicação a fim de atender requisitos de QoS relacionados à latência e vazão.

Cenário1

O esquema *NoCMesh_DeterministRoutingVCAttended* (Figura 5.2(a)) ilustra uma configuração de Rede em Chip desenvolvida a partir dos requisitos de uma aplicação que exige baixo nível de latência.Enquanto que o esquema da Figura 5.2(b), *NoCMesh_DeterministRoutingVCnonAttended,* especifica o cenário em que os requisitos de latência e vazão da mesma aplicação não são atendidos. Desta forma, é possível demonstrar em quais circunstâncias determinadas propriedades de um sistema são verdadeiras ou falsas.

Para o atendimento dos requisitos, o esquema da Figura 5.2(a) especifica a configuração proposta em (Bolotin *et al*., 2004), integrando a topologia *Mesh 2D*(*RegularMesh2D*) com os seguintes mecanismos: Roteamento Determinístico XY (*XYRouting*); Chaveamento por Pacote *Wormhole* (*Wormhole*); Controle de Fluxo Baseado em Crédito com Canais Virtuais (*CreditBasedFlowControl*e *VirtualChannel*); Arbitragem *Round Robin* com Prioridade Baseada em Classes de Serviço (*PriorityArbitration*); e Memorização FIFO em cada porta do roteador (*FIFOBuffering*).

A latência da comunicação depende fortemente da técnica de chaveamento utilizada (Ni e McKinley, 1993). Nas Redes em Chip, o mecanismo de chaveamento por pacote *Wormhole* é amplamente utilizado por oferecer alta vazão e baixa latência com requisitos mínimos de memorização, como mostra o esquema na Figura 5.2.

Porém, o uso desta estratégia deixa a rede suscetível ao problema de *deadlock* (Seiculescu *et al.*, 2010). Devido a esta característica, o *Wormhole* deve ser integrado com mecanismos que garantam a ausência de *deadlock*, como o algoritmo *XY*, o Controle de Fluxo Baseado em Crédito e o uso de Canais Virtuais.

O algoritmo de roteamento *XY*, além de evitar *deadlocks*, possui baixa complexidade de implementação, como especifica o esquema da Figura 5.2 e, consequentemente, baixo custo (Bolotin *et al*., 2004). Enquanto que o controle de Fluxo Baseado em Crédito também previne a perda de *flits* (Dobkin, 2007; Bolotin et al. , 2004; Bjerregaard e Sparso, 2004).

O uso de canais virtuais evita contenções por associar cada canal físico a mais de um *buffer*. Além disso, a inserção de canais virtuais em roteadores permite implementar políticas para tratar fluxos de dados de forma diferenciada, suportando QoS (Melo *et al*., 2005).

No cenário especificado pelo esquema *NoCMesh_DeterministRoutingVCAttended*, o tratamento diferenciado de dados é feito por meio do mecanismo de Arbitragem por Prioridade baseado em Classes de Serviço, onde a cada classe está associado um canal virtual como em (Bolotin *et al*., 2004). As filas de *buffers* que compõem o canal virtual estão organizadas de acordo com a estratégia *First In- First Out* ou FIFO.

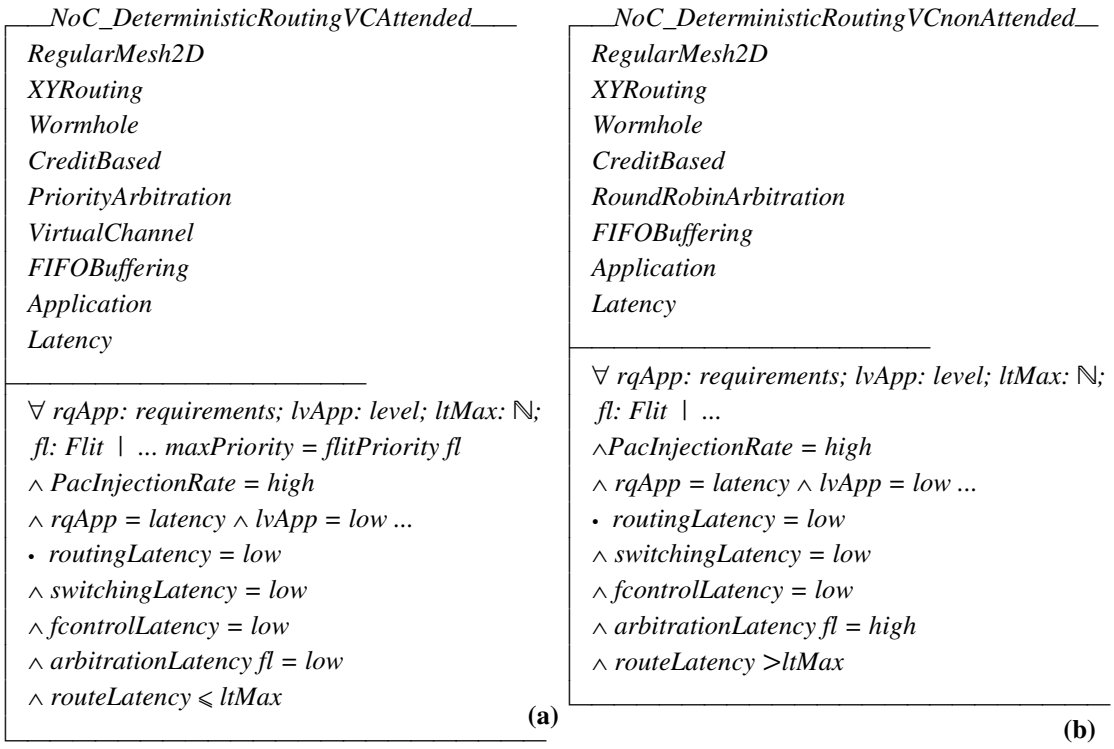


Figura 5.2. (a) Esquema *NoC_DeterministicRoutingVCAttended*(b) Esquema *NoC_DeterministicRoutingVCnonAttended*

O esquema da Figura 5.2(a) considera: *flits* pertencentes a uma mensagem com prioridade máxima (*maxPriority = flitPriority (fl)*); a rede com alta taxa de injeção de dados (*PacInjectionRate = high*); a mensagem é gerada por uma aplicação (*Application*) de tempo real para decodificação de vídeo, requerendo um baixo nível de Latência (*Latency; rqApp = latency ∧ lvApp = low*) e que interrompe o fluxo de menor prioridade adindo de uma aplicação que realiza multiplicação de matrizes. O predicado do esquema conclui que nestas circunstâncias este fluxo consegue uma latência dentro do limite requerido (*routeLatency⩽ ltMax*), desde que todos os mecanismos de comunicação ofereçam baixa latência.

O esquema da Figura 5.2(b) não considera o uso de canais virtuais e arbitragem por prioridade. Desta forma, não é garantido que o fluxo de dados gerado pela aplicação flua pela rede com a latência dentro do limite requerido, como conclui o predicado do

esquema (*routeLatency*≥ *ltMax*), desde que todos os mecanismos de comunicação ofereçam baixa latência.

O formalismo Z permite demonstrar que a configuração do cenário 1, Figura 5.2(a), é suficiente para atender ao requisito de latência da aplicação. Esta afirmação pode ser provada a partir do seguinte teorema:

RegularMesh2D, XYRouting, Wormhole, CreditBased, PriorityArbitration, VirtualChannel, FIFOBuffering⊢ *routeLatency*⩽ *ltMax* (1)

Para auxiliar a prova, uma proposição pode ser adicionada afirmando que se os níveis de latência oferecidos pelos mecanismos de roteamento, chaveamento e arbitragem são baixos, então é correto assumir que a latência total alcançada pela rota é baixa.

RegularMesh2D, XYRouting, Wormhole, CreditBased, PriorityArbitration, VirtualChannel, FIFOBuffering(routingLatency = low ∧ *switchingLatency = low* ∧*flowcontrolLatency = low* ∧ *arbitrationLatency = low* ∧*bufferingLatency = low* ⇒*routeLatencyLevel = low)*⊢ *routeLatency*⩽ *ltMax* (2)

A expansão dos esquemas *XYRouting, Wormhole* e*PriorityArbitration* gera as expressões rotuladas de 3 a 5, respectivamente. Na expressão (3), é mostrado que para garantir que uma Rede em Chip com um algoritmo de roteamento determinístico, como o *XY*, alcance um baixo nível de latência é preciso o uso de no mínimo dois canais virtuais, segundo experimento apresentado em (Duato, 1997) esta é a quantidade mínina de canais virtuais suficiente para prover *compromisso* entre latência e vazão.

(Routing | *routingComplexity = low* ∧ *SizeVirtualChannel = 2* ⇒ *routingThroughput =high* ∧ *routingLatency = low)* (3)

A expressão (4) mostra a expansão do esquema *Wormhole*, onde pacotes são divididos em *flits*, fazendo com que os requisitos de armazenamento sejam substancialmente reduzidos. Como este mecanismo permite o encaminhamento dos *flits* tão logo existam recursos disponíveise, a latência também é reduzida.

(PacketSwitching; Buffering;
switchingLatency: level | *bufferSize = small; switchingLatency = low)* (4)

Como mencionado, a arbitragem por prioridade garante a menor latência para o pacote de maior prioridade no momento de contenção da rede. Considerando que o esquema *NoC_DeterministicRoutingVC* (Figura 6.2) descreve o encaminhamento de *flits* (*fl*) com prioridade máxima (*maxPriority = flitPriority* (*fl*)), pela expansão do esquema *PriorityArbitration* (5), percebe-se que o fluxo de maior prioridade terá transmissão com menor latência.

$$(arbitrationLatency: Packet \rightarrow level \mid pr1 = packPriority\ p1 \wedge pr2 = packPriority\ p2 \wedge pr1 > pr2 \Rightarrow preemptedPack = p2 \wedge arbitrationLatency = low) \quad (5)$$

Considerando as afirmações apresentadas, pode-se afirmar que são verdadeiras as expressões (6) e (7) abaixo:

$$routingLatency = low \wedge switchingLatency = low \wedge arbitrationLatency = low \quad (6)$$

$$routeLatency \leqslant ltMax \quad (7)$$

Cenário 2

No cenário 2, o esquema *NoC_PartialAdaptiveRoutingAttended* (Figura 5.3 (a)) ilustra a configuração de Rede em Chip proposta em (Haibo Zhu*et al*., 2007), desenvolvida a partir dos requisitos de uma aplicação que exige alto nível de vazão. O esquema *NoC_PartialAdaptiveRoutingnonAttended* especifica a configuração que não atendem ao requisito da aplicação relacionado à vazão.

Para o atendimento do requisito, o esquema *NoC_PartialAdaptiveRoutingAttended* integra a topologia *Mesh 2D* (*RegularMesh2D*) com os seguintes mecanismos: Roteamento Parcialmente Adaptativo *Negative-First* (*NegativeFirstRouting*); Chaveamento por Pacote *Wormhole* (*Wormhole*); Controle de Fluxo Baseado em Crédito(*CreditBasedFlowControl*); Arbitragem *Round Robin* (*RoundRobin*); e Memorização FIFO em cada porta do roteador (*FIFOBuffering*).

Diferentemente do cenário anterior, o cenário 2 não utiliza a arbitragem por prioridade como forma de viabilizar o tratamento diferenciado aos fluxos de dados. Para alcançar alta vazão, o principal mecanismo apresentado no esquema da Figura 5.3 (a) é o roteamento parcialmente adaptativo. Embora as metodologias adaptativas de roteamento possam adicionar *overheads* em termos de energia dissipada, estas possuem a habilidade de estabelecer rotas alternativas na situação de contenção e de falhas (Haibo Zhu *et al.*, 2007).

Segundo Haibo Zhu *et al.* (2007), em uma Rede em Chip com roteamento parcialmente adaptativo, na medida em que se aumenta a injeção de dados, a desempenho é mantido e consegue superar o desempenho de uma rede com algoritmo determinístico. Logo, o esquema *NoC_PartialAdaptiveRoutingnonnonAttended*, Figura 5.3 (b), representa a configuração que não oferece o desempenho requerido pela aplicação, pois o esquema considera o uso de um algoritmo determístico. O esquema da Figura 5.3 (a) considera: a rede com alta taxa de injeção de dados (*PacInjectionRate = high*); e uma aplicação (*Application*) que requer um alto nível de Vazão (*rqApp = throughput* $\wedge$ *lvApp = high* $\wedge$(*rqApp*, *lvApp*)$\in$ *requirementsLevel*). O predicado do esquema associa o atendimento bem sucedido de vazão às altas vazões proporcionadas pelos mecanismos de roteamento e chaveamento (*routingThroughput = low* $\wedge$ *switchingThroughput = low* $\wedge$ *routeThroughput* $\geq$ *tpMin*).

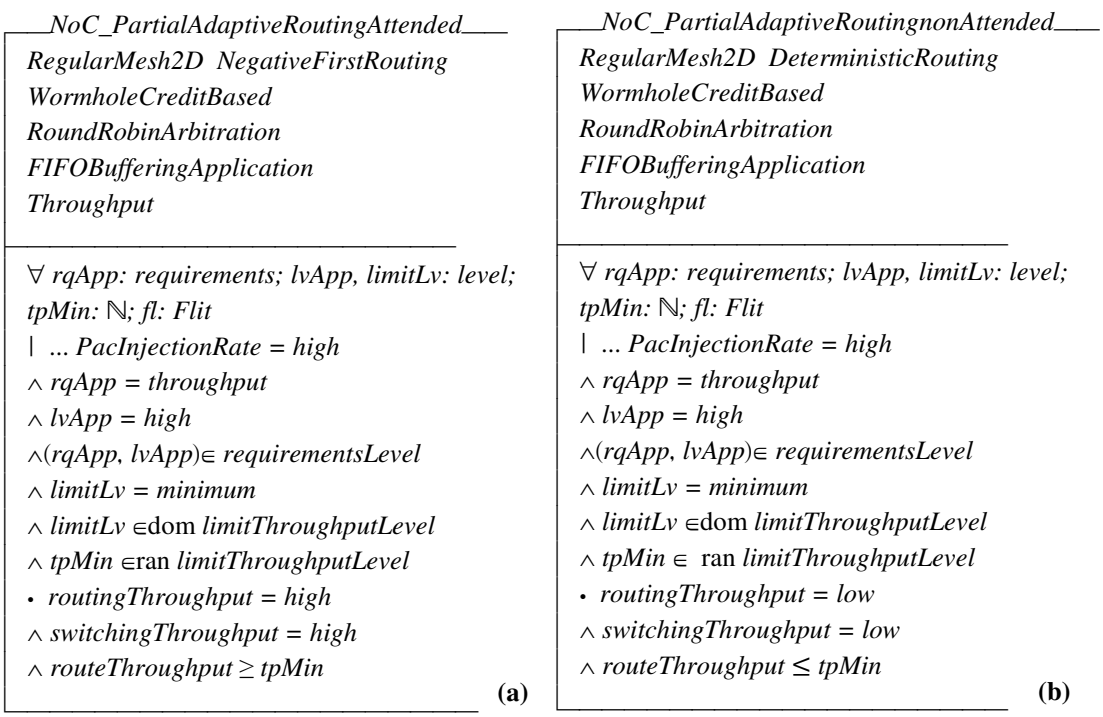


Figura 5.3. (a) Esquema *NoC_PartialAdaptiveRoutingAttended*; (b) Esquema *NoC_PartialAdaptiveRoutingnonAttended*

A ênfase ao mecanismo de chaveamento, e não apenas ao roteamento, deve-se à dependência entre vazão e tamanho da mensagem, determinado pelo tipo de chaveamento aplicado (Duato *et al.*, 1997). A vazão é considerada sustentável quando o número de mensagens a serem armazenadas nos nós não excede o limite máximo estabelecido (Ni *et al.*, 1997), como descreve a Figura 5.3 (a) (*routeThroughput*$\geq$ *tpMin*).

Em situação de contenção, o chaveamento por pacote *Wormhole* pode

ocasionar o bloqueio de *flits* em múltiplos roteadores e, consequentemente, diminuir a vazão na rota. Nesta situação, o algoritmo de roteamento adaptativo oferece alternativas de fuga às regiões de contenção, evitando a redução da vazão.Sendo assim, é possível combinar estes mecanismos para alcançar a vazão requerida, como demonstra o teorema expresso em (8).

$$NegativeFirstRouting,\ Wormhole \vdash routeThroughput \geq tpMin \tag{8}$$

Se for verdadeira a afirmação de que os níveis de vazão oferecidos pelos mecanismos de roteamento e chaveamento não sãobaixos, então a vazão total na rota estará dentro do limite requerido, como mostra a preposição adicionada na expressão (9).

$$NegativeFirstRouting,\ Wormhole\ (routingThroughput = high \wedge switchingThroughput = high) \vdash routeLatency \leqslant ltMax \tag{9}$$

Como mencionado anteriormente, para que uma rede chaveada por pacote do tipo *Wormhole* alcance altos níveis de vazão é preciso evitar rotas com trechos ocupados, para que não haja *flits*bloqueados ao longo do caminho.Isso é mostrado na expressão (10), onde o esquema *Wormhole* é expandido.

$$PacketSwitching \mid ((r1, r2), busy) \notin linkStatus \Rightarrow switchingLatency = low \wedge switchingThroughput = high \tag{10}$$

Ao expandir o esquema *NegativeFirstRouting* (expressão (11)), é possível observar que o algoritmo, depois do primeiro encaminhamento, escolhe canais livres da rede para compor a rota.

$$AdaptiveRouting \mid ((r1, r2), busy) \notin linkStatus \Rightarrow switchingLatency = low \wedge switchingThroughput = high \tag{11}$$

Considerando as afirmações apresentadas, pode-se afirmar que são verdadeiras as expressões (12) e (13) abaixo:

$$routingLatency = low \wedge switchingLatency = low \tag{12}$$

$$routeLatency \leqslant ltMax \tag{13}$$

Cenário 3

No cenário 3, o esquema *NoCFatTree_DeterministicRoutingAttended* (Figura 5.4 (a)), ilustra uma configuração de Rede em Chip desenvolvida a partir dos requisitos de uma aplicação que exige baixo nível de latência. Esta configuração de Rede em Chip

é utilizada no trabalho apresentado em (Gomez *et al.*, 2007). A Figura 5.4(b) apresenta o esquema *NoCFatTree_DeterministicRoutingnonAttended* que demonstra uma configuração que não atende aos requisitos exigidos pela aplicação.

Para atender aos requisitos da aplicação, o esquema *NoCFatTree_DeterministicRoutingAttended* integra a topologia *Fat-Tree* (*FatTree*) com os seguintes mecanismos: Roteamento Determinístico *Turn Around* (*TurnAroundRouting*); Chaveamento por Pacote *Virtual Cut-Through* (*VirtualCutThrough*); Controle de Fluxo Baseado em Crédito (*CreditBasedFlowControl*); e Memorização FIFO em cada porta do roteador (*FIFOBuffering*).

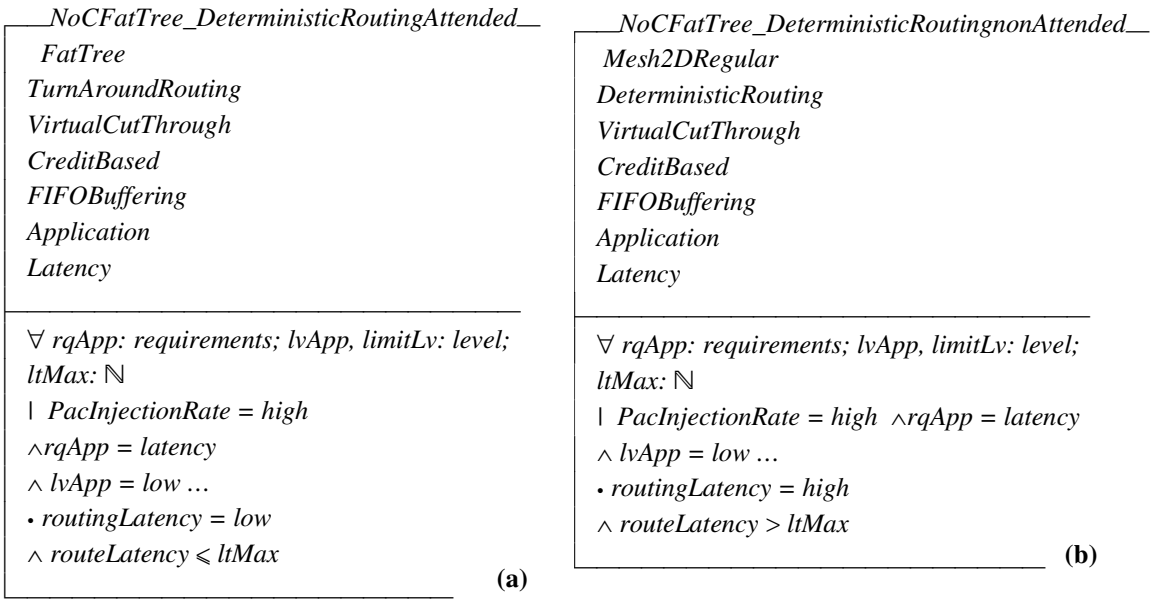


Figura 5.4. (a) Esquema *NoCFatTree_DeterministicRouting*; (b) Esquema *NoCFatTree_DeterministicRoutingnonAttended*

Segundo experimentos apresentados em Gomez *et al*. (2007), numa rede de interconexão com topologia *Fat-Tree*,o roteamento determinístico apresenta desempenho superior ao algoritmo adaptativo, provendo menor latência e mantendo o mesmo nível de vazão. Desta forma, o esquema *NoCFatTree_DeterministicRoutingnonAttended* (Figura 5.4 (b)), que especifica uma configuração que não combina a topologia *FatTree* com um algoritmo determinístico, não garante o atendimento do requisito de latência exigido pela aplicação.

A afirmação de que, numa rede de interconexão com topologia *Fat-Tree*,o roteamento determinístico apresenta desempenho superior ao algoritmo adaptativo pode ser provada por meio do teorema expresso em (14). À semelhança de teoremas

expressos anteriormente, o teorema (14) pode ser demonstrado pela adição da proposição ((*routingLatency* = *high* ∧*switchingThroughput* = *high*)) e pela expansão do esquema *TurnAroundRouting*.

$$TurnAroundRouting \vdash routeLatency \leqslant ltMax \quad (14)$$

Para a validação das especificações, além das provas, as seguintes operações passaram pelo processo de animação: adição de roteadores, *links* e propriedades destes elementos; e transições de estados que representam as mudanças ocorridas durante a execução dos mecanismos de comunicação. A etapa de animação contempla todas as topologias e mecanismos de comunicação que compõem o modelo de QoS proposto. Para ilustrar as operações envolvidas na animação, esta subseção considera, a topologia e os mecanismos apresentados no Cenário 1.

A exemplificação é iniciada pela operação de adição de roteadores à Rede em Chip e de propriedades que independem da topologia a ser utilizada, como mostra o esquema*AddRouterProperties* na Figura 5.5 (a). Por meio deste esquema, altera-se o estado da Rede em Chip (Δ*NoC*), adicionando roteadores ao conjunto *routers* (*routers'* = *routers* ∪ {*r?*}), bem como associando valores de latência, energia e *status* a estes elementos. A Figura 5.5 (b) ilustra o estado final dos conjuntos *routers*, *routerLatency* e *routerEnergy*, após a inclusão de todos os nós representados graficamente na Figura 5.5 (c).

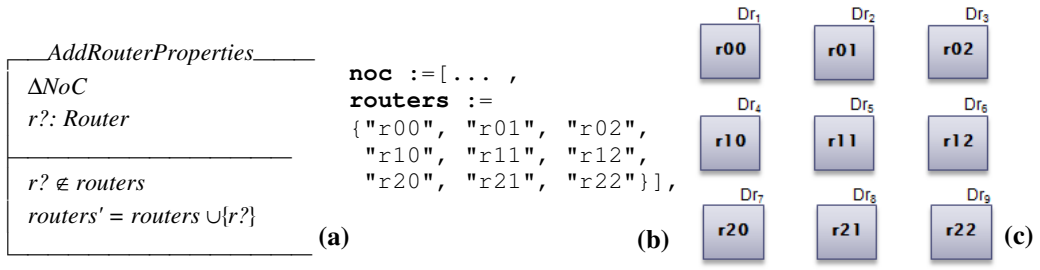


Figura 5.5. (a) Operação *AddRouterProperties*; (b) Animação Possum do estado final da redeapós operação; (c). Representação gráfica de roteadores e suas propriedades

A representação do local de cada um dos roteadores depende da topologia selecionada, por isso o esquema *AddRouterPlace2D* (Figura 5.5 (a))altera o estado da topologia (Δ*Topology*), adicionando ao conjunto *routerplacexy* associações entre roteadores e coordenadas que definem suas localizações(*routerplacexy'* = *routerplacexy* ∪{(*r?*,(*x?*, *y?*))}).

A representação por meio de coordenada, mostrada na Figura 5.6 (a), é válida para quaisquer topologias que considerem duas dimensões, como *Mesh 2D* e *Fat-Tree*.

Na Figura 5.6 (b), que mostra parte da animação realizada no Possum, é possível observar que cada roteador anteriormente adicionado em *routers* está associado a um valor de ordenada e a um valor de abscissa.

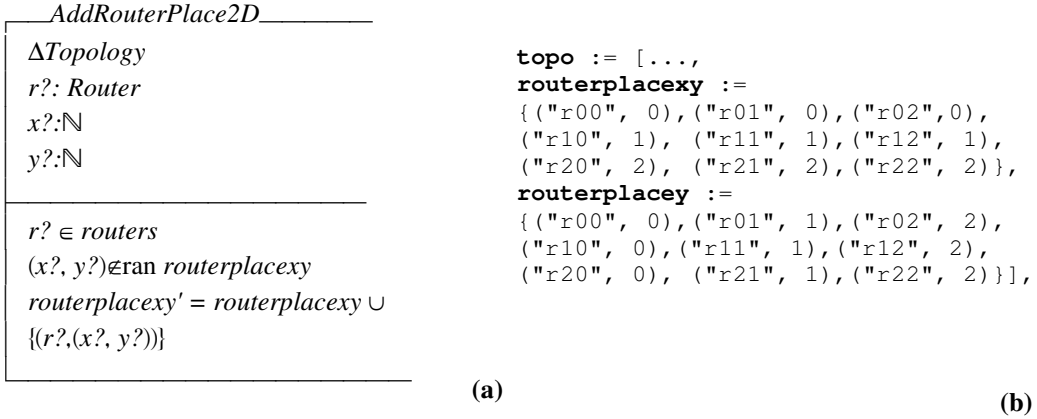


Figura 5.6. (a) Operação AddRouterPlace2D; (b) Animação Possum do estado final da topologia após operação

A partir das associações de coordenadas aos roteadores é possível determinar quais nós são adjacentes e podem formar *links*, como no esquema *AddRegularMesh2D*. Na Figura 5.7 (a), os *links* são formados de acordo com as restrições do esquema *RegularMesh2D* (*RegularMesh2D*). Com a formação dos *links* é possível adicionar as propriedades dos canais, tais como latência, largura de banda e *status.* Trecho do resultado da animação da operação *AddRegularMesh2D* é mostrado na Figura 5.7 (b).

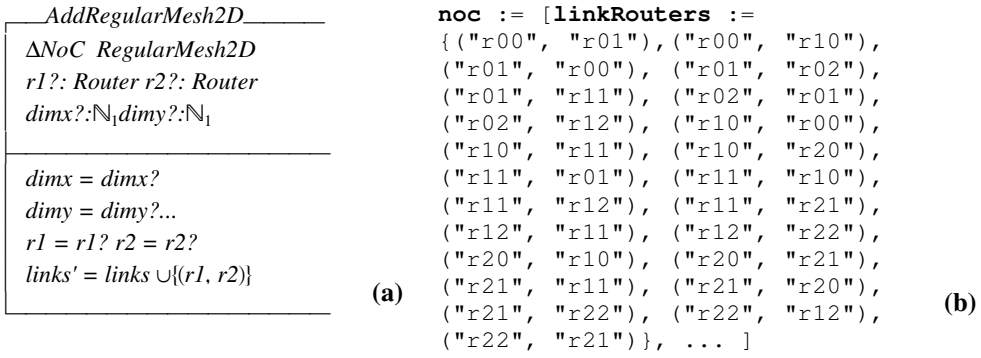


Figura 5.7. (a) Operação AddRegularMesh2D; (b) Animação Possum do estado final da rede após a operação

Para exemplificar a animação de mecanismos de comunicação, nesta subseção são apresentados trechos da animação Possum que mostram as transições ocorridas na Arbitragem por Prioridade e no Controle de Fluxo Baseado em Crédito utilizados no Cenário 1.

Na Figura 5.8 (a), o conjunto *classPriority* (esquema *PriorityArbitration*) contém as classes *RealTime* e *NonRealTime* associadas às suas respectivas prioridades. No momento em que dois *flits* (`fl1` e `fl2`) disputam pelo mesmo *link*, o árbitro seleciona

o *flit* de maior prioridade (`fl1`) para ser encaminhado ao longo de um *link* até um *buffer* de entrada num roteador alvo (`inBuffer:={("fl1","bfin")}`). O*flit* de menor prioridade (`fl2`) é selecionado para armazenamento temporário (`preemptedFlit:= "fl2"`).AFigura 5.8 (b) mostra o estado dos conjuntos relacionados ao controle de fluxo antes e depoisdo encaminhamento do *flit* de maior prioridade. O trecho da animação mostra que existe recurso suficiente para armazenar um *flit* no roteador alvo (`credit:= 1`). Após o encaminhamento do primeiro *flit*, o contador *credit* é decrementado (`credit:= 0`) e o conjunto *targetBuffer*, antes vazio, passa a conter um novo elemento (`targetBuffer:={("bfin", "fl1")}`).

```
arbitration:=
[classPriority:={("NonRealTime", 1),
("RealTime", 2)}, flitPriority:={("fl1",2),
("fl2", 1)}, inBuffer:={("fl1","bfin")},
maxpriority:= 2, outBuffer:={("fl2","bfout")},
preemptedFlit:= "fl2"
```

(a)

```
flowcontrol':=
[bufferRouter:={("bfin","r01"),
("bfout","r00")},credit:= 0,
sourceBuffer:={("bfout", "fl2")},
targetBuffer:={("bfin", "fl1")}],
```

(b)

Figura 5.8. (a) Animação Possum da Arbitragem por Prioridade; (b) Animação Possum do Controle de Fluxo

Cenário 4

Este cenário trata da aplicação que processa áudiode acordo com os modelos digitais NXPs (*Next generation Philips*)para rádios automotivos apresentados em (Van den Berg e Bhullar, 2004; Bhullar *et al*., 2004). O objetivo desse estudo de caso, conforme documentado em (Moonen *et al*., 2011), é mapear os processos da aplicação em uma Rede em Chip, de modo a possibilitar a comparação do desempenho da Rede em Chip com arquiteturas tradicionais baseadas em barramento e com outras configurações de Redes em Chip.

A aplicação de áudio processa fluxos de dados sob restrições de Tempo Real, podendo ser representada por um grafo que consiste em tarefas que se comunicam por conexões ponto-a-ponto. Em um cenário real, tais processos podem ser identificados pela função ou componente genérico, por exemplo, FIR (*Finite Impulse Response*), CRD (*Coordinate Rotation Digital*), etc., conforme visualizado na Figura 5.9 (a). No entanto, para a modelagem da aplicação é suficiente definir os processos e os requisitos da aplicação. Assim sendo, os processos foram genericamente identificados no grafo (v1, v2,..., vn), como mostra a Figura 5.9 (b).

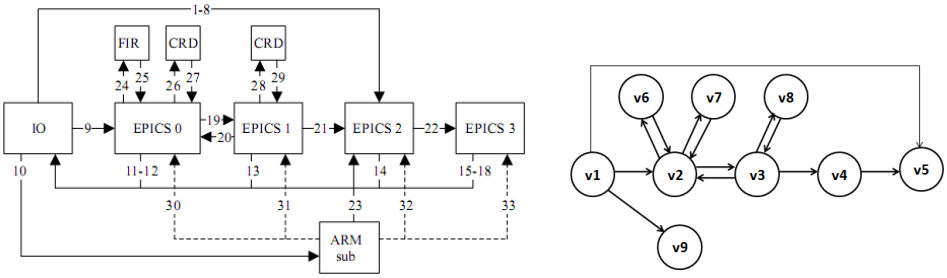


Figura 5.9. (a) Processos da aplicação de áudio; (b) Grafo da aplicação

A modelagem do grafo da Figura 5.9(b) é realizada a partir do esquema *Application* que especifica o conjunto de processos da aplicação (**processes**:={"v1",...,"v9"}),a comunicação entre os processos (**communicationprocess**:={("v1","v2"),("v2","v3"),...}.), os rótulos das arestas (**label**:={(("v1","v2"),9),(("v2","v3"),19),...)}) e o requisito da aplicação seguido do seu nível de exigência (**requimentsLevel**:={(latency,minimum)}). A Figura 5.10 ilustra o trecho do resultado da animação desta aplicação no Possum.

```
application:=[processes:={"v1","v2","v3","v4","v5","v6","v7","v8","v9"},
communicationprocess:={("v1","v2"),("v2","v3"),...},
label:={(("v1","v2"),9), ("v2","v3"),19),...)},
requimentsLevel:={(latency,minimum)}]
```

Figura 5.10. Animação Possum da aplicação

Enquanto o esquema *Application* define o grafo da Figura 5.9(b), o esquema *Latency* associa um valor ao nível mínimo, ou máximo, exigido para a latência necessária entre os processos. Este valor é associado por meio da função *limitLatencyLevel* e dos invariantes constantes na parte do predicado do esquema *Latency*.

```
latency:= [limitLatencyLevel:={(latency, 3)
```

Figura 5.11. Animação da Função *limitLatencyLevel*

Diante das várias possibilidades de topologias para o atendimento dos requisitos de QoS da aplicação, para este estudo de caso foi assumido o uso da topologia *RegularMesh2D* com dimensão 3x3. Esta topologia provê *compromisso* entre a minimização da quantidade de roteadores e interfaces e minimização de contenções na rede (Moonen *et al*., 2011).

Neste estudo de caso o roteamento é realizado por meio do algoritmo determinístico XY. Os demais mecanismos utilizados foram: controle de fluxo Baseado em Crédito; arbitragem por TDMA; e chaveamento por circuito. O esquema *NoC_TDMA* na Figura 5.12 apresenta a configuração utilizada no estudo de caso, que atende aos requisitos de latência requeridos pela aplicação.

Nesta configuração, a baixa latência pode ser alcançada por meio da arbitragem TDMA, que garante latência mínima para o fluxo da classe GT (*Guaranteed Throughput*). A técnica TDMA organiza a comunicação de forma a evitar disputas entre os fluxos, mantendo o compromisso entre a quantidade de *buffers* nos roteadores, latência e vazão na rede (Coenen *et al.*, 2006).

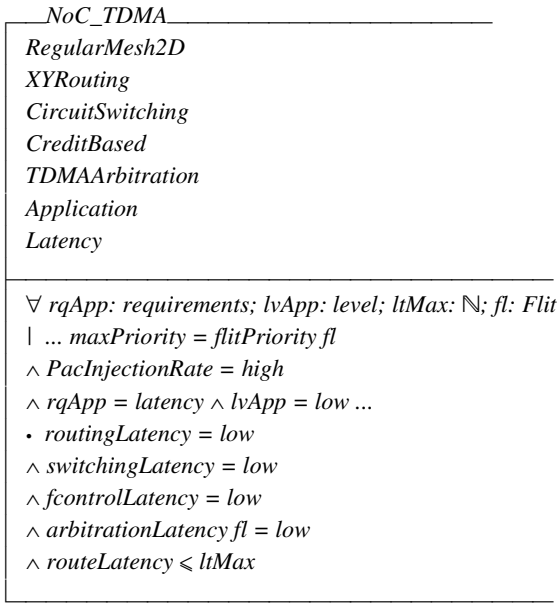


Figura 5.12. Esquema *NoC_TDMA*

6. RESULTADOS E CONCLUSÕES

Esta dissertação apresentou um modelo formal de QoS gerado a partir da extensão da especificação formal que compõe a metodologia CADZ, a fim de auxiliar o projeto de Redes em Chip que atendem requisitos de QoS.O modelo proposto foi desenvolvido de acordo com o formalismo Z, que permite construir de maneira clara e livre de ambiguidades especificações para projetos de Redes em Chip, bem como, para análise qualitativa das propriedades vitais de comunicação com QoS.

O uso de linguagens de alto nível no projeto de sistemas complexos, como as Redes em Chip, diminui o tempo de lançamento do produto no mercado, estas linguagens aumentam a produtividade. Este aumento envolve a combinação de quatro aspectos chave contemplados pela notação Z: abstração; reuso; automação e exploração.

A elaboração da extensão de CADZ considerou as seguintes premissas:

- Atender requisitos de QoS representa um aspecto crítico no projeto de Redes em Chip, visto que estes requisitos estão cada vez mais rigorosos;
- A escolha de topologias e mecanismos de comunicação suportados pela estrutura de interconexão é fundamental para prover requisitos de QoS;
- A análise da eficiência de topologias e mecanismos de comunicação de uma Rede em Chip do projeto deve acontecer nas fases iniciais do desenvolvimento;
- As métricas de desempenho latência e vazão fornecem uma análise sólida do comportamento de uma configuração de Rede em Chip;
- O reuso representa uma forma efetiva para acelerar o desenvolvimento do projeto;
- A especificação formal é um importante meio de reter o conhecimento adquirido favorecendo o reuso;
- As etapas de animação e prova de uma especificação formal são importantes para sua validação.

6.1 Resumo das Contribuições

Embora as contribuições efetivas de um modelo só possam ser validadas após seu uso em situações reais, o que não seria possível neste caso, a validação com estudo de caso descrito no Capítulo 5, permite inferir os benefícios do modelo.

A primeira contribuição obtida neste trabalho é a identificação dos compromissos entre topologias e mecanismos de comunicação a serem considerados

quando se deseja atender requisitos de QoS. Dado o aumento significativo dos requisitos exigidos por aplicações, em especial as de Tempo Real que não toleram perdas de prazos, a forma como a combinação entre topologias e mecanismos se dá pode ser um fator decisivo para a obtenção do desempenho esperado. Sistematizar a preocupação com esse fato durante todo o desenvolvimento do projeto é uma forma de trabalhar para o sucesso.

Outra contribuição, legada da metodologia CADZ, é a identificação de linguagens formais como um elemento chave na retenção de conhecimento de projetos e base para o reuso. Ao utilizar este tipo de linguagem durante as fases iniciais, é possível antecipar a avaliação de propriedades do projeto, e consequente identificação em tempo hábil de potenciais problemas, que possam vir a comprometer o sistema.

A integração das especificações anteriormente oferecidas pela CADZ ao modelo de QoS proposto por este trabalho só foi possível pelo potencial de reuso presente na notação Z. Além disso, o modelo de QoS acrescenta ao conjunto de especificações CADZ não só descrições de algoritmos específicos de roteamento, chaveamento, controle de fluxo, arbitragem e memorização, mas também de cenários formados pela combinação destas descrições.

Uma importante contribuição deste trabalho é a adição da etapa de prova no desenvolvimento das especificações. O processo de prova, além de auxiliar a entender de forma mais completa os requisitos aos quais o sistema se propõe a atender, permite identificar erros e inconsistências presentes nas especificações.

6.2 Perspectivas Futuras

A partir desta pesquisa está previsto o aprimoramento da etapa de prova apresentada no capítulo 5 como trabalho futuro. É possível continuar a exploração do espaço de projeto, gerando novos cenários de forma a identificar mais configurações de Redes em Chip que atendam a diferentes níveis de latência e vazão. Além disso, a especificação pode ser continuada a fim de contemplar diferentes métricas de desempenho como largura de banda e consumo de energia.

A especificação pode ainda ser ampliada para que contemple outros aspectos da Teoria das Filas e da Simulação e não apenas a disciplina preemptiva por prioridade especificada pelo esquema *PriorityArbitration*. Para isso é necessário considerar a discretização do tempo, tempos de chegada dos pacotes, a suspensão de serviço, a

simulação baseada em eventos, entre outros. Neste caso, existe a possibilidade de integrar a especificação gerada com descrições baseadas em formalismos mais adequados para descrever concorrência e temporização como LOTOS, CSPZ ou outros.

Embora o modelo de QoS possa ser aplicado de forma não automatizada, a criação de um software que simule um ambiente real de projeto para Redes em Chip com QoS é sem dúvida um importante trabalho a ser desenvolvido.

As especificações formais dos elementos, a identificação de requisitos de QoS, bem como as configurações que atendem a estes requisitos serão de extrema importância no desenvolvimento deste ambiente. Além disso, este ambiente deverá auxiliar na etapa de refinamentos sucessivos até níveis mais baixos de abstração.

A etapa de refinamento possibilitará a integração do modelo de QoS gerado com a etapa de síntese, podendo o modelo guiar a geração de código em linguagem de programação ou o desenvolvimento de ASICs em linguagem de descrição de hardware, como VHDL.

REFERÊNCIAS

Adriahantenaina, A.; Charlery, H.; Greiner, A.; Mortiez, L.; Zeferino, C. A. (2003), "SPIN: A Scalable, Packet Switched, On-Chip Micro-Network". In *Proceedings of the conference on Design, Automation and Test in Europe: Designers' Forum - Volume 2* (DATE '03), Vol. 2. IEEE Computer Society, Washington, DC, USA.

Agarwal A., Iskander C. D., Shankar, R. (2009), "Survey of Network on Chip (NoC) Architectures & Contributions". Journal of Engineering Computing and Architecture, Volume 3, Issue 1.

Alameldeen,A. R.; Wood, D. A. (2004)."Adaptive cache compression for high-performance processors". In ISCA '04: Proceedings of the 31st Annual International Symposium on Computer Architecture, page 212.

Bafumba-Lokilo, D., Savaria, Y., David, J. P. (2008), "Generic crossbar network on chip for FPGA MPSoCs".*Joint 6th International IEEE Northeast Workshop on Circuits and Systems and TAISA Conference*, 269-272.

Benini, L.; De Micheli, G. (2001)"Powering networks on chips: energy-efficient and reliable interconnect design for SoCs". In: 14th International Symposium on System Synthesis (ISSS'01), pp. 33-38.

Benini, L.; De Micheli, G. (2004), "Networks on chips: a new SoC paradigm". Computer, 35(1), pp. 70-78.

Berejuck, M. D., Zeferino, C. A. (2009), "Adding Mechanisms for QoS to a Network-on-Chip", In: 22nd Symposium on Integrated Circuits and Systems – SBCCI, pp. 153-158.

Bhullar, H.S.; van den Berg, R.; Josten,J.; Zegers, F. (2004), "Serving digital radio and audioprocessing requirements with sea-of-dsps for automotive applications the Philips way," inIEEE ISPC GSPx.

Bjerregaard, T.; Mahadevan, S. (2006), "A survey of research and practices of Network-on-chip". ACM Computer Surveys, New York, v. 38, n. 1, p. 1-51.

Bjerregaard,T.; Sparso, J. (2004), "Virtual channel designs for guaranteeing bandwidth in asynchronous network-on-chip", Norchip Conference, 269-272.

Bolotin E. (2004) ,"QNoC: QoS architecture and design process for network on chip", Journal of Systems Architecture, 50(2-3), pp. 105-128.

Bolotin, E., Cidon, I., Ginosar, R., Kolodny, A. (2007) ,"Routing table minimization for irregular mesh NoCs". In *Proceedings of the conference on Design, automation and test in Europe* (DATE '07). EDA Consortium, San Jose, CA, USA, 942-947.

Bononi, L.; Concer, N. (2006), "Simulation and analysis of network on chip architectures: ring, spidergon and 2D mesh," *Design, Automation and Test in Europe. DATE '06.Proceedings*.

Bononi, L.; Concer, N.; Grammatikakis, M.; Coppola, M.; Locatelli, R. (2007), "NoC Topologies Exploration based on Mapping and Simulation Models," *Digital System Design Architectures, Methods and Tools.DSD 2007. 10th Euromicro Conference on* , vol., no., pp.543-546, 29-31.

Butterfield, A., Freitas, L., Woodcock, J. (2009),"Mechanising a formal model of flash memory". Sci. Comput. Program. 74, 219-237.

Carro, L., Wagner, F. (2003), “Sistemas Computacionais Embarcados”. JAI - XXII Jornada de Atualização em Informática.

Cavalcanti, A.L.C.; Woodcock, J.C.P. (1998), “ZRC - A Refinement Calculus for Z”, *Formal Aspects of Computing*, 10(3), p.267-289.

Chiu, G. (2000), "The odd-even turn model for adaptive routing".*Parallel and Distributed Systems, IEEE Transactions on* , vol.11, no.7, pp.729-738, Jul 2000.

Coenen, M.; Murali, S.; Radulescu, A.; Goossens, K.; De Micheli, G. (2006). "A buffer-sizing algorithm for networks on chip using TDMA and credit-based end-to-end flow control," *Hardware/Software Codesign and System Synthesis, 2006. CODES+ISSS '06. Proceedings of the 4th International Conference* , vol., no., pp.130-135.

Cogito (1999), “SUM The Reference Manual”, Software Verification Research Centre, School of Information Technology, The University of Queensland.

Coombes, A.; McDermid, J. (1993), "Specifying temporal requirements for distributed real-time systems in Z," *Software Engineering Journal* , vol.8, no.5, pp.273-283.

Corrêa, E. F.; Silva, L.A.P.; Wagner, F. R.; Carro, L. (2007), “Fitting the Router Characteristics in NoCs to Meet QoS Requirements”, In Proceedings of 20thSBCCI, ACM Press,pp. 105-110.

Coussy, P.; Gajski, D.D.; Meredith, M.; Takach, A. (2009), "An Introduction to High-Level Synthesis," *Design & Test of Computers, IEEE* , vol.26, no.4, pp.8-17.

Dally, W. J.; Towles, B. (2004), “Principles and Practices of Interconnection Networks”. Morgan Kaufmann Publishers, 550 p.

Dally, W.; Towles, B. (2001), “Route Packets, Not Wires: On Chip Interconnection Networks”, Proc. of Design Automation Conference, pp. 684-689.

Dally, W.J. (1992),“Virtual-channel flow control”. IEEE Trans. Parallel Distrib. Syst. 3 (2), 194–204.

Dally, W.J., Seitz, C.L (1987), “Deadlock Free Message Routing in Multiprocessor Interconnection Networks”. IEEE Transactions on Computers, Vol C-3G, No 5, p. 547-553.

Das, R.; Mishra, A.K.; Nicopoulos, C.; Dongkook Park; Narayanan, V.; Iyer, R.; Yousif, M.S.; Das, C.R. (2008), "Performance and power optimization through data compression in Network-on-Chip architectures," *High Performance Computer Architecture, 2008.HPCA 2008. IEEE 14th International Symposium on* , vol., no., pp.215-225.

Dobkin, R. R. (2007), “Credit-based Communication in NoC”. In Course: Introduction to Networks on Chips: VLSI aspects, Technion.

Dobkin, R. R.; Ginosar, R.; Kolodny, A. (2009).“QNoC asynchronous router”.*Integr.VLSI J.* 42, 103-115.

Dobkin, R.; Vishnyakov, V.; Friedman, E.; Ginosar, R. (2005), "An Asynchronous Router for Multiple Service Levels Networks on Chip," Proc. ASYNC, pp. 44-53.

Dodge, C. J.; Undrill, P. E.; Allen, A. R.; Ross, P.G.B. (1996), “Application of Z in digital hardware design”.Computers and Digital Techniques, IEE Proceedings, vol.143, no.1, pp.79-86.

Duato, J., Yalamanchili, S., Ni, L. (1997), “Interconnection Networks: An Engineering Approach”. Morgan Kaufmann Publishers.

Escale, D.K.F.R., Ramos, K.D.N., Ribeiro, C.M.F.A. (2011),“ Supporting NoC projects with a Formal QoS-aware Model”. In Proceedings of The 14th Brazilian Symposium on Formal Methods.

Feliciian, F.; Furber, S.B. (2004), "An asynchronous on-chip network router with quality-of-service (QoS) support".*SOC Conference, 2004.Proceedings. IEEE International* , vol., no., pp. 274- 277, 12-15.

Felicijan, T. (2004),“Quality-of-Service (QoS) for Asynchronous On-Chip Networks”. Ph.D. thesis,Department of Computer Science, The University of Manchester, Manchester,UK.

Freitas, L., Woodcock, J. (2007),“Mechanising Mondex with Z/Eves”.*Form.Asp.Comput.* 20, 1, 117-139.

Gangwal, O. P., Radulescu,A., Goossens, K., Pestana, S. G., Rijpkema, E. (2005), “Building Predictable Systems on Chip: An Analysis of Guaranteed Communication in the Æthereal Network on Chip”. In P. van der Stok, editor, Philips Research Book Series, chapter 1.

Gebremichael, B., Vaandrager, F.W., Zhang, M., Goossens, K., Rijpkema, E. e Radulescu, A. (2005), “Deadlock Prevention in the Æthereal Protocol”.In D.Borrione e W. Paul, editors.Proceedings 13th IFIP Advanced Research Working Conference on Correct Hardware Design and Verification Methods, pp.345 – 348.

Glass, C. J.; Ni, L.M. (1992),"The Turn Model for Adaptive Routing," *Computer Architecture Proceedings., The 19th Annual International Symposium on* , vol., no., pp.278-287.

Gomez, C.; Gilabert, F.; Gomez, M.E.; Lopez, P.; Duato, J. (2007), "Deterministic versus Adaptive Routing in Fat-Trees," *Parallel and Distributed Processing Symposium, 2007. IPDPS 2007. IEEE International* , vol., no., pp.1-8, 26-30.

Goossens, K. (2004), “Formal Methods for Networks on Chips”, Fifth (ISCA’04), pp. 188-197.

Goossens, K., Dielissen, J., Grangwal, O. P., Pestana, S. G. (2005a), “A Design Flow for Application-Specific Networks on Chip with Guaranteed Performance to Accelerate SoC Design and Verification”, Proc. Design, Automation and Test in Europe Conference and Exhibition.

Goossens, K., Dielissen, J., Radulescu, A. (2005b), “Æthereal network on chip: concepts, architectures, and implementations”. IEEE Design & Test of Computers.

Goossens, K.; Hansson, A. (2010), "The aethereal network on chip after ten years: Goals, evolution, lessons, and future," *Design Automation Conference (DAC), 2010 47th ACM/IEEE* , vol., no., pp.306-311.

Grot B., Hestness J., Keckler, S. W., Mutlu O., (2011),“Kilo-NOC: a heterogeneous network-on-chip architecture for scalability and service guarantees”. In *Proceeding of the 38th annual international symposium on Computer architecture* (ISCA '11).ACM, New York, NY, USA, 401-412.v.

Guerrier, P., Greiner A. (2000), “A generic architecture for on-chip packet switched interconnections”. In DATE´2000.International Conference on Application of Concurrency to System Design, pp188-189.

Gupta, R.; Zorian, Y.(1997), “Introducing Core-Based System Design”. IEEE Design & Tes Computers, 14(4), pp. 15-25.

Haibo Zhu; Pande, P.P.; Grecu, C. (2007), "Performance Evaluation of Adaptive Routing Algorithms for achieving Fault Tolerance in NoC Fabrics," *Application - specific Systems, Architectures and Processors, 2007.ASAP. IEEE International Conf. on* , vol., no., pp.42-47, 9-11.

Hazel, D., Strooper, P., Traynor, O. (1997), “Possum: An Animator for the SUM Specification Language”. In Proc. AsiaPacific Software Engineering Conference and Int. Computer Science Conference, pp. 42-51.

Helmy, A.; Pierre, L.; Jantsch, A. (2010), "Theorem proving techniques for the formal verification of NoC communications with non-minimal adaptive routing," *Design and Diagnostics of Electronic Circuits and Systems (DDECS), 2010 IEEE 13th International Symposium on* , vol., no., pp.221-224, 14-16.

Hu, R.; Marculescu, R. (2005), "Energy-and performance-aware mapping for regular NoC architectures," IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems, vol. 24, pp. 551-562.

Huaxi Gu; Jiang Xu; Wei Zhang (2009), "A low-power fat tree-based optical Network-On-Chip for multiprocessor system-on-chip," *Design, Automation & Test in Europe Conference & Exhibition, 2009. DATE '09.*, vol., no., pp.3-8.

Hwang, K. (1993), “Advanced Computer Architecture: parallelism, scalability, programmability”, McGraw-Hill Series in Computer Science.

ISO/IEC 13568:2002 - Information technology (2002), “Z formal specification notation –Syntax”, type system and semantics. Disponível em: http://www.iso.org.

Jantsch, A., Tenhunen, H. (2003), Network on chip. Kluwer Academic Publishers.

Johnson, C.W. (1995), "Using Z to support the design of interactive safety-critical systems," *Software Engineering Journal* ,vol.10, no.2, pp.49-60.

Karim, F.; Nguyen, A. e Dey, S. (2002), “An Interconnect Architecture for Networking Systems on Chips”, IEEE Micro 22(5), 36–45.

Kauffman, M., Manolios, P., Moore, J. S., (2000), “Computer-Aided Reasoning: An Approach”. Kluwer Academic Publishers.

Kogel, T., Leupers, R., Heinrich, M. (2006), “Integrated System-Level Modeling of Network-On-Chip Enabled Multi-Processor Platforms”. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Kreutz, M. E. (2005), “Método para otimização de plataformas arquiteturais para sistemas multiprocessados heterogênes”, Tese de Doutorado, Programa de Pós-Graduação em Computação, UFRGS.

Kuhma, W. (1989) “Stress inducing properties of system response times”, Ergon, 1989,32, (3), pp. 271-280.

Leiserson, C. E. (1985) “Fat-trees: universal networks for hardware-efficient supercomputing. *IEEE Trans. Comput.* 34, 892-901.

Li B., Zhao L., Iyer R., Peh L., Leddige M., Espig M., E. L. Seung, Newell D. (2011). “CoQoS: Coordinating QoS-aware shared resources in NoC-based SoCs”. *J. Parallel Distrib. Comput.*71, 5.

Marculescu, R.; Ogras, U.Y.; Li-Shiuan Peh; Jerger, N.E.; Hoskote, Y.; (2009) "Outstanding Research Problems in NoC Design: System, Microarchitecture, and Circuit Perspectives," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on* , vol.28, no.1, pp.3-21.

Martin, G.; Chang, H. (2001), "System on Chip Design". In: 9th International Symposium on Integrated Circuits, Devices & Systems (ISIC'01), Tutorial 2, pp. 12-17.

Mello, A.; Tedesco, L.; Calazans, N.; Moraes, F.; (2005), "Virtual Channels in Networks on Chip: Implementation and Evaluation on Hermes NoC," *Integrated Circuits and Systems Design, 18th Symposium on* , vol., no., pp.178-183, 4-7.

Mirza-Aghatabar, M., Koohi, S., Hessabi, S., Pedram, M. (2007), "An Empirical Investigation of Mesh and Torus NoC Topologies Under Different Routing Algorithms and Traffic Models," dsd, pp.19-26, 10th Euromicro Conference on Digital System Design Architectures, Methods and Tools.

Moonen, A.;Bartels, C.;Bekooij, M.;Berg, R. V. D.;Bhullar, H.;Goossens, K.G.W.;Groeneveld, P.;Huiskens,J.;Meerbergen, J.V. (2011), "Comparison of an Aethereal Network on Chip and traditional interconnects - Two case studies", VLSI-SoC: Research Trends in VLSI and Systems on Chip, number 249 in *IFIP International Federation for Information Processing*, Springer.

Moraes, F.; Calazans, N.; Mello, A.; Möller, L.; Ost, L. (2004), "Hermes: an Infrastructure for Low Area Overhead Packet-switching Networks on Chip". Integration the VLSI Journal, 38(1), pp. 69-93.

Murali, S.; De Micheli, G.(2004), "Bandwidth-Constrained Mapping of Cores onto NoC Architectures". In *Proceedings of the conference on Design, automation and test in Europe - Volume 2* (DATE '04), Vol. 2.IEEE Computer Society, Washington, DC, USA.

Murali, S.; De Micheli, G.(2004),"SUNMAP: a tool for automatic topology selection and generation for NoCs". In *Proceedings of the 41st annual Design Automation Conference* (DAC '04). ACM, New York, NY, USA, 914-919.

NASA (1995), "Formal Methods Specification and Verification Guidebook for Software and Computer Systems".Volume I - Planning and Technology Insertion.

Ni, L.M.; Yadong Gui; Moore, S. (1997), "Performance evaluation of switch-based wormhole networks," *Parallel and Distributed Systems, IEEE Transactions on* , vol.8, no.5, pp.462-474.

Ni, L.M; McKinley, P.K. (1993) "A Survey of Wormhole Routing Techniques in Direct Networks," Computer, vol. 26, no. 2, pp. 62-76.

Ogras, U. Y.; Hu, J.; Marculescu, R. (2005) "Key research problems in NoC design: a holistic perspective". In *Proceedings of the 3rd IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis* (CODES+ISSS '05).ACM, New York, NY, USA, 69-74.

Palaniveloo, V. A.; Sowmya, A. (2011), "Application of Formal Methods for System-Level Verification of Network on Chip," VLSI, IEEE Computer Society Annual Symposium on, pp. 162-169, 2011 IEEE Computer Society Annual Symposium on VLSI.

Pande, P. P.; Grecu, C.; Jones, M; Ivanov, A.; Saleh, R. (2005) "Performance Evaluation and Design Trade-Offs for Network-on-Chip Interconnect Architectures," IEEE Transactions on Computers, pp. 1025-1040.

Pastrnak, M.; De With, P.H.N.; Van Meerbergen, J. (2006), "Qos concept for scalable MPEG-4 video object decoding on multimedia (NoC) chips," *Consumer Electronics, IEEE Transactions on* , vol.52, no.4, pp.1418-1426.

Paterson, L.; Davie, B. (2003), "Computer Networks: A System Approach". 3ª Edição, Morgan Kauffmann Publishers, 2003, 813 p.

Pontes, J.; Moreira, M.; Moraes, F.; Calazans, N. (2010a) "HERMES-A - An Asynchronous NoC Router with Distributed Routing." In: International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS'10), Grenoble.

Pontes, J.J.H.; Moreira, M.T.; Moraes, F.G.; Calazans, N.L.V. (2010b), "Hermes-AA: A 65nm asynchronous NoC router with adaptive routing," *SOC Conference (SOCC), 2010 IEEE International* , vol., no., pp.493-498.

Potter, B.; Till, D.; Sinclair,J.:(1996), "*An Introduction to Formal Specification and Z",*(2nd ed.). Prentice Hall PTR, Upper Saddle River, NJ, USA.

Prado, D. S. (1999); Teoria das Filas e da Simulação. Belo Horizonte: Editora de Desenvolvimento Gerencial, 124 p. Série Pesquisa Operacional, v. 2.

Radulescu, A.; Dielissen, J.; Goossens, K.; Rijpkema, E.; Wielage, P. (2004). "An Efficient On-Chip Network Interface Offering Guaranteed Services, Shared-Memory Abstraction, and Flexible Network Configuration". In *Proceedings of the conference on Design, automation and test in Europe - Volume 2* (DATE '04), Vol. 2.IEEE Computer Society, Washington, DC, USA.

Ramos, K. D. N. (2007), "CADZ: Uma Metodologia de Projeto baseada em Z para Redes-em-Chip", Tese de Doutorado, Programa de Pós-Graduação em Engenharia Elétrica, UFRN.

Rigo, S.; Azevedo, R.; Santos, L.(2011), "Electronic System Level Design An Open-Source Approach", 1st Edition, IX, 146 p. 134 illus.

Rocha, D.K.F., Ramos, K.D.N., Ribeiro, C.M.F.A. (2010),"Especificação em LOTOS de requisitos de QoS para Redes-em-Chip". In Proceedings of The 16th IBERCHIP Workshop.

Rose, A.V.V.; Seshasayanan, R.; Oviya, G. (2011), "FPGA implementation of low latency routing algorithm for 3D Network on Chip," *Recent Trends in Information Technology (ICRTIT), 2011 International Conference on* , vol., no., pp.385-388, 3-5.

Saaltink, M. (1997a), "The Z/EVES User's guide".Technical report, ORA Canadá, Ontário, Canadá.TR-97-5493-06.

Saaltink, M. (1997b), "The Z/EVES system". In J. P. Bowen, M. G. Hinchey, and D. Till, editors, ZUM'97: Z Formal Specification Notation, vol. 1212 of Lecture Notes in Computer Science., pp. 72–85.

Salaun, G.; Serwe, W.; Thonnart, Y.; Vivet, P. (2007), "Formal Verification of CHP Specifications with CADP Illustration on an Asynchronous Network-on-Chip," *Asynchronous Circuits and Systems.ASYNC 2007. 13th IEEE International Symposium on* , vol., no., pp.73-82.

Schmaltz, J.; Borrione, D. (2004a), “A Functional Approach to the Formal Specificationof Networks on Chip”.In. Proc. of Formal Methods in Computer-Aided Design. A. J. Hu and A. K. Martin (eds), LNCS 3312, Springer-Verlag, pp52-66.

Schmaltz, J.; Borrione, D. (2004b), “A Functional Specification and Validation Model for Networks on Chip the ACL2 Logic”. In Proceedings of the 5th International Workshop on the ACL2 Theorem Prover and its Applications.

Schmaltz, J.; Borrione, D. (2005), “A Generic Network on Chip Model”.Proceedings TPHOLs, Springer LNCS 3603.

Schonwald, T.; Zimmermann, J.; Bringmann, O.; Rosenstiel, W. (2007), "Fully Adaptive Fault-Tolerant Routing Algorithm for Network-on-Chip Architectures," *Digital System Design Architectures, Methods and Tools, 2007.DSD 2007.10th Euromicro Conference.*

Seiculescu, C.; Murali, S.; Benini, L.; De Micheli, G.; (2010), "A method to remove deadlocks in Networks-on-Chips with Wormhole flow control," *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2010* , vol., no., pp.1625-1628, 8-12.

Spivey, J. M. (1992), “The Z Notation: A Reference Manual”, Prentice Hall International, 2nd edition.

Tamir, Y., Frazier, G. (1992), “Dinamically-Allocated Multi-Queue Buffers for VLSI Communication Switches”. IEEE Trans. On Computers, v. 41, n.6, pp. 725-737.

Tanenbaum, A. S. (1996),“Computer Networks”. Prentice Hall.

Tsiopoulos, L., Waldén, M. (2006), “Formal Development of NoC Systems in B”.TUCS Technical Report No. 751.

Van den Berg, R.; Bhullar, H.S. (2004), “Next generation Philips Digital Car Radios, based on asea-of-DSP concept,” in IEEE ISPC GSPx.

Waldén, M., Sere, K. (1998), “Reasoning about action systems using the B Method”.In Formal Methods in System Design, Vol. 13, No 1, pp. 5 - 35.Kluwer Academic Publishers.

Wang, Y. *et al*. (2008),“Dynamic TDM virtual circuit implementation for NoC”. In APCCAS.

Wezeman, C. D.(1995),“Using Z for network modelling: An industrial experience report”. Computer Standards & Interfaces, Volume 17, Issues 5-6, 30 September, Pages 631-638.

Woodcock, J., Davies, J. (1996), “Using Z: Specification, Refinement and Proof”, Prentice-Hall International Series in Computer Science.

Zafar, N.A. (2009), "Formal Specification and Validation of Railway Network Components using Z Notation" IET, Software Vol:3(4) pp:312-320.

Zeferino, C. A., Susin, A., (2003), “SoCIN: a parametric and scalable network-on-chip”, in: 16th Symposium on Integrated Circuits and Systems Design (SBCCI’03), pp. 169–174.

Zimmermann H. (1980), “OSI Reference Model - The ISO Model of Architecture forOpen Systems Interconnection,” IEEE Transactions on Communications, vol. 28,no. 4, pp. 425-432.