UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

Desenvolvimento e avaliação de redes-em-chip hierárquicas e reconfiguráveis para MPSoCs

por

CEZAR RODOLFO WEDIG REINBRECHT

Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação

Prof. Dr. Altamiro Amadeu Susin
Orientador

Porto Alegre, Julho de 2012

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Wedig Reinbrecht, Cezar Rodolfo
Desenvolvimento e avaliação de redes-em-chip hierárquicas e reconfiguráveis para MPSoCs / Cezar Rodolfo Wedig Reinbrecht. -- 2012.
119 f.
Orientador: Altamiro Amadeu Susin.
Dissertação (Mestrado) -- Universidade Federal do Rio Grande do Sul, Instituto de Informática, Programa de Pós-Graduação em Computação, Porto Alegre, BR-RS, 2012.
1. Redes-em-chip. 2. Reconfiguração. 3. Hierarquia. 4. Sistemas-em-chip. I. Amadeu Susin, Altamiro, orient. II. Título.

Agradecimentos

Gostaria de agradecer a todos que de uma forma ou de outra contribuíram para realização deste trabalho de pesquisa e escrita desta dissertação para minha formação no Programa de Pós-Graduação em Computação da Universidade Federal do Rio Grande do Sul (UFRGS).

Ao meu orientador, Prof. Altamiro Amadeu Susin, pela sua dedicação, amizade, orientação e pela sua visão de futuro que inspira muitos de seus orientandos. Assim como César Zeferino, também presto uma homenagem através de uma arquitetura de rede-em-chip, a rede HASIN, uma arquitetura para o futuro sistema computacional.

Ao professor Luigi Carro, que inspirou meus ideais como arquiteto de hardware, sempre me instigando a desenvolver novas ideias, não se limitando com os conceitos pré-existentes. Aprendi muito e espero ainda aprender mais.

À professora Fernanda Kastensmidt pelo apoio e ensino, onde através das aulas e projetos, obtive experiências diferenciadas em uma área estratégica, que complementaram meu conhecimento.

Ao professor Márcio Kreutz, da Universidade Federal do Rio Grande do Norte (UFRN) pelo apoio, dedicação e profissionalismo, que foram fundamentais para que todo o trabalho pudesse ser realizado. Através de suas ferramentas e participações no trabalho, foi possível atingir excelentes resultados e conclusões. Obrigado.

Aos demais professores que me orientaram e ensinaram durante o curso na pós-graduação: Sérgio Bampi, Flávio Wagner e Philippe Navaux, Erika Cota e Marcelo Lubaszewski.

À colega Débora Matos, que esteve presente em todo o desenvolvimento deste trabalho de pesquisa, onde gastamos muitos esforços para amadurecer nossos conceitos e realizamos inúmeras propostas arquiteturais. Após muitos esforços e dedicação, começamos a colher os frutos do excelente trabalho, através de artigos, revistas, trabalhos em colaboração internacional, etc. Agora, estamos em busca de mais desafios, onde a partir desse trabalho de pesquisa avançaremos em nossos doutorados evoluindo o conceito desenvolvido para que no futuro se torne um padrão de interconexão industrial em chips.

Agradeço também aos bolsistas de iniciação científica que contribuíram muito para o desenvolvimento do trabalho, e cito aqui Jonathan Martinelli, Rafaela Wetternick e Tiago Motta.

Aos colegas Paulo César, Cláudio, Jucemar, Anelise, Cristiano, Gabriel, Antonio Felipe, Bruno Visotto e Eduarda por compartilharem momentos de tensão pelas aulas e de descontração também.

Aos amigos que sentiram minha ausência, mas mesmo assim continuaram sempre me apoiando: Jonas, Gerson (esse não sentiu tanto minha ausência, hehe), Raupp, Abella, Igor, Digão, Júlia e Wilkens, Jumir, Fábio, Frederico, Emérson, Yumi, Maico, Leandro, Mônica, Arthur, Andrea, Renata, Posselt, Bianchi e Veríssimo.

Ao NSCAD, empresa que trabalho, por permitir que meus horários fossem flexíveis, tornando mais prático e eficiente a dualidade de trabalhar e cursar o mestrado. Através do trabalho no NSCAD obtive as bases para realizar um trabalho acadêmico com perspectivas industriais, o que considero muito importante para o cenário de pesquisa brasileiro em semicondutores. Agradeço também ao Prof. Eric Fabris por proporcionar ao NSCAD e CT1 um ambiente profissional e de excelência em formação de projetistas de circuitos integrados.

À minha mãe e irmã, Lisete e Stéfani, que são responsáveis pela pessoa que sou e pelos valores e princípios que trago comigo. Sempre me apoiaram e me ajudaram no que precisei para seguir em frente e buscar meus sonhos. Aos meus familiares, que mesmo de longe me apoiaram, agradecendo especialmente minha dinda Vânia, que lutou muito e se preocupou para que eu realizasse essa conquista. Muito obrigado a vocês!

À minha lindinha (namorada), Paula Rodrigues, que sempre esteve ao meu lado, atuando como minha inspiração. Agradeço por suportar os momentos de ausência, de angústia e de stress. Juntos nós conseguimos vencer mais uma batalha, e daqui para frente vamos enfrentar muitas outras lado a lado. Obrigado por ser a minha metade.

A todos, minha sincera gratidão!

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

ASIC	Aplication Specific Integrated Circuit
BCS	Buffered Circuit Switching
BIST	Built-in Self Test
BOP	Begin of-packet
DOR	Dimension-ordered routing
EP	Elemento de Processamento
EOP	End-of-packet
FIFO	First-In, First-Out
FLIT	Flow Control Unit
FSM	Finite State Machine
GALS	Globalmente Assíncrono Localmente Síncrono
HiCIT	Hierarchical Crossbar-based Interconnection Topology
HASIN	Hierarchical Adaptive Switching Interconnection Network
IP	Intellectual Property
ITRS	International Technology Roadmap for Semiconductors
MINoC	Multiple Interconnections Networks-on-Chip
MPEG	Motion Picture Experts Group
MPSoC	Multiprocessor System-on-Chip
NCS	Network Communication Sub-system
NoC	Networks-on-Chip
PS	Packet Switching
RAPeC	Router Architecture Packet & Circuit
RASoC	Router Architecture for System-on-Chip
SAF	Store-and-forward
SISD	Single Instruction Single Data
SoC	System-on-Chip
SoCIN	System-on-Chip Interconnection Network

SwIX	Switching Interconnection Xbar
TDMA	Time Division Multiple Access
TVOPD	Triple Video Object Plane Decoder
UCS	Unbuffered Circuit Switching
UFRGS	Universidade Federal do Rio Grande do Sul
VCT	Virtua Cut-through
VC	Virtual Channel
VHDL	VHSIC Hardware Description Language
VOPD	Video Object Plane Decoder
WH	Wormhole
XBAR	Crossbar

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

Com o advento dos processos submicrônicos, a capacidade de integração de transistores numa mesma pastilha de silício atingiu níveis que possibilitaram a construção dos sistemas com múltiplos processadores num chip (MPSoCs, do inglês **M**ulti**P**rocessor **S**ystem-**o**n-**C**hip). Essa possibilidade de integração permite inserir dezenas de **E**lementos de **P**rocessamento (EPs) nos circuitos integrados atuais, e já se projeta centenas de EPs para os sistemas da próxima década (ITRS, 2011). Nesse cenário, um dos principais desafios se refere ao serviço de interconexão dos EPs, que deve apresentar um desempenho de comunicação necessário para as aplicações em execução sem comprometer as limitações de consumo de área e energia do circuito. Nos primeiros sistemas multiprocessados, com poucos nodos, arquiteturas baseadas em barramento foram suficientes para cumprir esses requisitos. Porém, o número de elementos nos sistemas recentes aumentou rapidamente, tornando as redes-em-chip a solução mais apropriada, por aliar escalabilidade e reuso na mesma estrutura. Contudo, diante da previsão de que essa tendência de aumento se manterá retorna a discussão se as redes-em-chip atuais continuarão adequadas para os futuros sistemas. De fato, o custo das redes-em-chip convencionais pode se tornar proibitivo para as escalas dos circuitos em um futuro próximo. Novas propostas têm sido apresentadas na literatura científica onde se podem destacar duas principais estratégias de projeto às redes de interconexão: reconfiguração arquitetural e organização hierárquica da topologia. A reconfiguração arquitetural permite obter uma grande eficiência, independente do tipo de aplicação em execução, pois uma das alternativas é projetar o circuito para que ele se auto adapte conforme os requisitos de desempenho para cada aplicação. Por outro lado, arquiteturas organizadas em topologias hierárquicas são desenvolvidas para uma estrutura computacional definida em tempo de projeto, sendo mais eficazes para uma classe de aplicações. O presente trabalho explora a sinergia da combinação das potencialidades das duas soluções e propõe uma nova estrutura que oferece melhor desempenho para uma classe maior de aplicações apropriada para os futuros sistemas. Como resultado foi implementada uma arquitetura adaptativa chamada MINoC (**M**ultiple **I**nterconnections **N**etworks-**o**n-**C**hip), uma arquitetura organizada em hierarquia, chamada HiCIT (**Hi**erarchical **C**rossbar-based **I**nterconnection **T**opology) e uma simbiose de ambas culminando na arquitetura hierárquica adaptativa HASIN (**H**ierarchical **A**daptive **S**witching **I**nterconnection **N**etwork). São apresentados resultados que mostram a eficiência desses conceitos validando a proposta hierárquica adaptativa.

Palavras-Chave: MPSoC; redes-em-chip; reconfiguração; hierarquia; soluções de interconexão; arquitetura adaptativa; sistemas-em-chip.

Development and Evaluation of Hierarchical and Reconfigurable Networks-on-Chip for MPSoCs

ABSTRACT

With the advent of submicron processes, the number of transistors integrated on a single chip has reached levels that allowed the design of **M**ulti**p**rocessor **S**ystems-**o**n-**C**hip (MPSoCs). This capability allows the integration of several **p**rocessing **e**lements (PEs) in integrated circuits designed nowadays. In the next decade it is expected that hundreds of PEs will be integrated on a single chip. In this scenario, a key challenge is the interconnection network between PEs, which must provide the communication service required to run applications without compromising the limitations of area and energy consumption. In the first multiprocessor systems, with few nodes, bus-based approaches have been sufficient to meet these requirements. However, current systems increased quickly the number of elements, making the **N**etworks-**o**n-**C**hip (NoCs) the most appropriate solution, because it handles scalability and reusability in the same structure. Nevertheless, ITRS roadmap predicts that this increase will continue (ITRS, 2011), which resumes the discussion if present NoC architectures will be the most adequate for future systems, since its costs could be prohibitive. Therefore, new proposals have been presented in the literature with two main design strategies: architectural reconfiguration and hierarchical organization of the topology. With the architectural reconfiguration it is possible to obtain an application independent high efficiency structure, because the circuit is designed to adapt itself to satisfy performance requirements. On the other hand, architectural organizations in hierarchical topologies are defined at design time to have the most appropriate features for a class of applications, being very effective. The current work identified the synergy of both approaches and proposes a new symbiotic structure suitable for a broader class of applications. As a result, it was implemented an adaptive architecture called MINoC (**M**ultiple **I**nterconexions **N**etworks-**o**n-**c**hip), an architecture organized in hierarchy called HiCIT (**Hi**erarchical **C**rossbar-based **I**nterconnection **T**opology) and a mix of both ending up with the hierarchical adaptive architecture HASIN (**H**ierarchical **I**nterconnection **N**etwork **A**daptive **S**witching). Results show the efficiency of these concepts validating the proposed hierarchical adaptive architecture.

Keywords: MPSoCs; Network-on-chip; interconnections; adaptive architecture; system-on-chip.

1 INTRODUÇÃO

O avanço dos processos submicrônicos e o escalamento tecnológico permitiram uma alta capacidade de integração de transistores em uma mesma pastilha de silício. Com o desenvolvimento dessa tecnologia, os engenheiros tiveram a possibilidade de projetar, em um único circuito integrado, sistemas computacionais cada vez mais complexos, com o propósito de atender aos requisitos do mercado eletrônico. Contudo, o aumento da complexidade nas arquiteturas, para cumprir as exigências das novas aplicações, acarreta custos para o seu desenvolvimento (tempo de projeto), custos operacionais e custos de fabricação (maior dissipação de potência e maior área), que precisam ser ponderados durante a etapa de ideação do produto.

Em um resumo cronológico, as arquiteturas de processadores partiram do princípio do modelo SISD da taxonomia de Flynn (*Single Instruction Single Data*), onde um único fluxo de instrução atua sobre um único conjunto de dados. Essa definição remete a um sistema composto por um elemento de processamento, uma memória, um elemento de interconexão e uma interface com o mundo externo (TANNENBAUM, 2001). Com a evolução tecnológica, todos esses elementos puderam ser integrados em um mesmo circuito, resultando nos sistemas em chip ou SoCs (*Systems-on-chip*) (MORI, 1993). A crescente demanda por desempenho resultou na aplicação de paralelismo em nível de instrução (*Instruction Level Paralelism*) (PATT, 85), agregando estruturas de maior complexidade e custo operacional aos processadores. Essa estratégia foi explorada até se chegar a uma limitação de potência para esses projetos, conhecida na literatura como *Power Wall*. Nesse contexto, partiu-se para o paralelismo em nível de thread (*Thread Level Paralelism*) e de tarefa. Assim, surgiram os modelos computacionais paralelos e distribuídos, compostos por diversos elementos de processamento, que distribuem entre si a computação. Na sequência, a possibilidade de integrar esses inúmeros elementos de processamento no mesmo chip culminou no surgimento dos sistemas multiprocessados em chip (MPSoCS) (JERRAYA, 2005).

Atualmente, é possível integrar em um circuito dezenas de elementos de processamento, elementos de memória, gerência de recursos, sensoriamento, entre outros. Esses elementos podem ser denominados núcleos ou propriedade intelectual (do inglês *Intelectual Property* ou IP). Os núcleos são componentes já pré-projetados e pré-verificados, que possuem uma funcionalidade específica, a qual pode ser utilizada para formar sistemas homogêneos (núcleos de processamento com mesma arquitetura) ou sistemas heterogêneos (núcleos de processamento com diferentes arquiteturas). A singularidade dos núcleos permite reutilizá-los no projeto de diferentes sistemas diminuindo o tempo de desenvolvimento e amortizando custos, tornando sua utilização

uma metodologia determinante para as empresas atenderem as pressões do mercado. No mesmo contexto, é necessária uma estrutura de interconexão intrachip para interligar e comunicar esses núcleos, que da mesma maneira, possua também alta reusabilidade e escalabilidade.

O meio de intercomunicação tem relação direta com a eficiência energética de qualquer sistema computacional (GRECU, 2005). Projetar um meio eficiente para interligar os elementos é um grande desafio atual, pois os ambientes encontrados nos MPSoCS já possuem dezenas de núcleos, e dependendo da aplicação, podem apresentar severas restrições de projeto. Esse requisito funcional é mais crítico em sistemas embarcados móveis, que possuem limitações relativas ao consumo de energia e à área. Portanto, uma solução de interconexão deve prover qualidade de serviço aliado a uma baixa dissipação de potência enquanto independente da aplicação em software; e isso a torna um dos principais desafios de projeto nos sistemas MPSoCs.

Os barramentos já foram alvo de inúmeras pesquisas; e várias soluções e melhorias foram propostas para o aumento de desempenho dos mesmos (LAHTINEN, 2003), (RYU, 2001),(AMBA, 1999) e (LEE, 2004). Porém, quando se faz o uso de um barramento, à medida que mais módulos são conectados a ele, o desempenho do sistema como um todo é degradado (REGO, 2006), já que a largura de banda do barramento é dividida por todos os dispositivos conectados a ele (GUERRIER, 2000). Além disso, cada módulo extra conectado ao barramento aumenta a capacitância do mesmo, fazendo com que o sistema dissipe mais potência. Devido a estes fatores, um barramento comporta apenas alguns módulos para que não ocorra uma grande queda no desempenho global do sistema (ZEFERINO, 2002).

Uma solução aos barramentos proposta pela academia foram as redes-em-chip (do inglês, Network-on-Chip ou NoC) (BENINI, 2002), onde os problemas encontrados em barramentos não acontecem porque as conexões são do tipo ponto-a-ponto, e, dessa forma, o desempenho é independente do número de módulos conectados à rede, podendo ocorrer de maneira paralela. Sendo assim, uma rede de interconexão no lugar de um barramento se mostrou interessante para os primeiros MPSoCs que possuíam dezenas de elementos de processamento (BENINI, 2001) (DALLY, 2001); já que foi possível obter uma comunicação escalável, paralela e com qualidade de serviço.

Durante a última década, as redes-em-chip foram adotadas como a solução mais apropriada aos sistemas multiprocessados, resultando em diversos produtos comerciais como o Am2045 (RALFHILL, 2006), 80-Tile (INTEL, 2007) e Tile64 (TILERA, 2007). No entanto, a International Technology Roadmap for Semiconductor em 2011 (ITRS, 2011) apontou a tendência de que nos próximos quinze anos os sistemas serão compostos por mais de seiscentos elementos de processamento de dados, ou seja, centenas de elementos com hardware simplificado. Nesse caso, o custo de cada roteador de uma rede-em-chip convencional passa a ser considerável e tende a limitar o sistema, seja em desempenho ou em potência. Portanto, é necessário buscar as melhores estratégias arquiteturais para comportar sistemas com centenas de núcleos. Pesquisas recentes vêm apresentando soluções em duas categorias de arquiteturas de redes-em-chip: as reconfiguráveis e as hierárquicas.

As arquiteturas de redes reconfiguráveis abordam o princípio de monitorar o sistema, e em tempo de execução alterar suas características para utilizar seus recursos de maneira mais eficiente. Sua reconfiguração pode trabalhar em nível de alocação de tarefas, de *buffers* ou mesmo enlaces, dentre outros. Em outra linha de pesquisa, as

redes hierárquicas permitem utilizar diferentes arquiteturas de interconexão para ajustar o custo ao desempenho requerido. Essa solução visa explorar a localidade das comunicações, estruturando em grupos os elementos com afinidade de comunicação, os quais possuem mecanismos de alto desempenho, e deixando os mecanismos com desempenho reduzido para comunicação entre esses grupos. Dessa forma, é possível diminuir os custos de potência e área sem comprometer o desempenho. A principal diferença entre as duas abordagens é que a hierárquica é projetada para uma classe de aplicações, enquanto que a reconfigurável busca ser eficiente com qualquer aplicação, sendo genérica.

Esse trabalho realiza um estudo sobre essas duas metodologias de projeto - reconfigurável e hierárquica - buscando identificar algumas características estratégicas que demonstrem as suas potencialidades e limitações e, a partir disso, criar uma sinergia de técnicas. Como resultado foi implementada uma arquitetura reconfigurável, uma arquitetura hierárquica e uma arquitetura da composição de ambas. As arquiteturas implementadas tiveram como base uma rede-em-chip convencional parametrizável, mais especificamente a SoCIN (ZEFERINO, 2002), com o diferencial de possuir um mecanismo adicional de *pipeline* nos roteadores. Para as análises e avaliações das propostas são utilizadas duas aplicações da literatura para sistemas multiprocessados, a TVOPD (MURALI, 2009) e a NCS (TINO, 2010). Além disso, os custos das propostas são avaliados para tecnologia ASIC com células padrão de nodo de 65 nanômetros. Por fim, o trabalho emprega as arquiteturas desenvolvidas como solução de interconexão para cada benchmark, realizando simulações em nível de sistema, bem como suas sínteses ASIC, obtendo resultados comparativos entre cada solução e a rede SoCIN.

1.1 Objetivos

O objetivo da presente dissertação é dominar as tecnologias de rede-em-chip reconfigurável e rede-em-chip hierárquica.

Dentre os objetivos específicos enumera-se:

- Identificar e estudar características importantes para o desenvolvimento de novas arquiteturas de interconexão
- Implementar uma proposta de arquitetura de rede reconfigurável
- Implementar uma proposta de arquitetura de rede hierárquica
- Implementar uma proposta de arquitetura que englobe reconfiguração e organização hierárquica
- Avaliar todas as estratégias estudadas e desenvolvidas e comparar com uma rede-em-chip convencional.

1.2 Organização do Documento

Esse documento está dividido em oito seções, onde a primeira apresenta a introdução e objetivos. A segunda realiza uma revisão conceitual de elementos básicos necessários à compreensão desse trabalho de dissertação. Na terceira seção apresentam-se os aspectos relativos às redes-em-chip reconfiguráveis, que compreende desde a revisão bibliográfica, estudos analíticos de suas características e uma proposta arquitetural. A quarta seção expõe a área de redes-em-chip hierárquicas, incluindo também uma revisão da literatura sobre trabalhos propostos, análises estratégicas sobre alternativas de interconexão e agrupamento, bem como uma proposta arquitetural. Seguindo no estudo dessas duas linhas de pesquisa pode-se observar na quinta seção a maior contribuição desse trabalho que é o estudo e o resultado da união das duas estratégias, buscando investigar quais as implicações de unir hierarquia e reconfiguração. Na seção seis são apresentados todos os experimentos realizados para cada arquitetura proposta e as análises de custos através de sínteses para ASIC. Por fim, a seção sete conclui esse trabalho de dissertação apresentando as conclusões e futuros trabalhos.

2 CONCEITOS BÁSICOS

Nesta seção são apresentados conceitos básicos para um melhor entendimento deste trabalho de dissertação. Dentre os principais tópicos abordados podemos citar os meios de interconexão e as redes-em-chip.

2.1 Interconexões Intrachip

Interconexões intrachip podem ser definidas como estruturas arquiteturais responsáveis por interligar os blocos internos de um projeto de chip. Essas arquiteturas podem assumir desde as formas mais simples de conexão como um fio ligando diretamente todos os nodos, chamado de ponto-a-ponto, a estruturas extremamente complexas que imitam as redes de computadores atuais. Variáveis de projeto, como o número de elementos, o desempenho requerido e as restrições de custo (área e potência) auxiliam na definição de qual arquitetura de interconexão é a mais apropriada. Nessa subseção, será brevemente descrito as arquiteturas de ponto-a-ponto, barramentos, chaves *crossbar* e redes-em-chip, demonstrado na Figura 2.1.

Um dos primeiros conceitos utilizados para interligar os nodos internos de um circuito integrado foi a conexão ponto-a-ponto totalmente conectada. Essa arquitetura conecta todos com todos, possuindo um protocolo específico para cada conexão. Além disso, o ponto-a-ponto apresenta o melhor desempenho, pois permite comunicação dedicada para cada elemento. Porém, essa característica acarreta em maior tempo de projeto, pois é necessário emparelhar os diferentes protocolos de comunicação existentes entre os nodos e a implementação física gera muitos fios, sendo um problema para questões de roteamento. Além disso, sua arquitetura não pode ser reutilizada em projetos futuros, visto que é muito engessada a uma única implementação, prejudicando o conceito de *time-to-market*, muito presente nos projetos de chips atuais.

Logo, essas conexões evoluíram para estruturas conhecidas como barramentos. Nessa arquitetura é utilizado apenas um enlace físico, onde todos se conectam por ele, e utilizam o mesmo protocolo de comunicação. Portanto, são compostos apenas por fios e lógica para controle de fluxo de dados, possuindo pouco custo de computação. Porém, apesar de seu baixo custo de área, seu desempenho degrada à medida que aumentam o número de elementos, pois uma única saída para comunicação se torna um gargalo (ZEFERINO, 2002). Essa limitação levou a proposta de barramentos hierárquicos, onde eram criados níveis diferentes de barramentos, cada nível com suas características

específicas, aumentando um pouco a área, mas garantindo maior desempenho (ZEFERINO, 2003). No entanto, com o constante aumento no número de elementos dentro do chip, seu desempenho também se tornou limitado.

Partindo da lógica dos barramentos, foi proposta a chave *crossbar*, que possui uma matriz de barramentos. Sua principal vantagem é permitir paralelismo a um sistema baseado em barramentos. Porém, sua desvantagem está no seu custo, que aumenta exponencialmente a cada novo nodo, devido a matriz de barramentos (BUHYAN, 1989). Além disso, *crossbares* grandes geram problemas de roteamento.

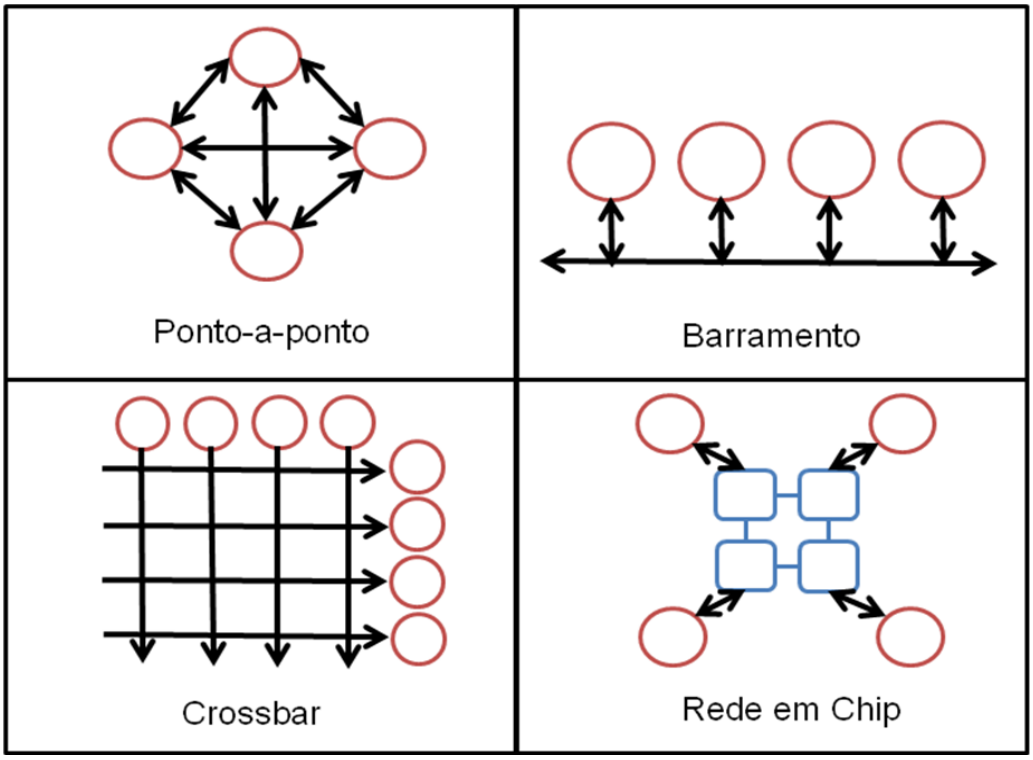


Figura 2.1 - Estruturas de Comunicação Intrachip

A possibilidade de integrar milhões de transistores proporcionou à literatura optar por evoluir para uma estrutura complexa, mas com estrutura fixa e totalmente escalável, as redes-em-chip (do inglês *Networks-on-Chip* ou NoC). Uma NoC é uma estrutura composta por um conjunto de roteadores e enlaces (do inglês *links*). Os roteadores são interconectados entre si através dos *links* e são responsáveis por verificarem um possível caminho para a transferência de dados. Normalmente estes dados são divididos em pacotes e enviados pela rede. Os *links* são responsáveis pela conexão entre os roteadores. Cada roteador pode possuir um núcleo acoplado, sendo que um núcleo pode ser um processador, uma memória, um cluster composto por processadores e/ou memória ou ainda blocos dedicados (MOLLER, 2006), também chamados de IPs.

Na tabela 2.1, é apresentada uma comparação feita por (FREITAS, 2009) entre as três arquiteturas mais utilizadas comercialmente, apontando seus prós e contras.

Tabela 2.1 – Comparação entre arquiteturas de interconexão.

Tipo de Interconexão		Prós (+) e Contras (-)	
Barramento	Fio	O aumento do fio aumenta a resistência degradando o desempenho.	-
Chave *Crossbar*		O aumento do fio aumenta a resistência degradando o desempenho.	-
Network-on-Chip		Os fios são ponto-a-ponto entre roteadores e o desempenho não degrada em função do aumento de nós.	+
Barramento	Árbitro	O árbitro é um gargalo à medida que o número de nós aumenta.	-
Chave *Crossbar*		O árbitro pode ser centralizado ou descentralizado e não é o fator principal para degradação do desempenho em função do aumento dos nós.	+-
Network-on-Chip		As decisões de roteamento são distribuídas e não representam um gargalo.	+
Barramento	Largura de banda	A largura de banda é limitada e compartilhada por todos os nós.	-
Chave *Crossbar*		Cada interconexão é independente e a largura de banda de comunicação por conexão não é afetada pelas demais.	+
Network-on-Chip		A largura de banda não é afetada pelo aumento da rede.	+
Barramento	Latência	Latência é afetada pelo fio.	+
Chave *Crossbar*		Latência é afetada pelo fio.	+
Network-on-Chip		Latência é afetada pelas contenções em roteadores	-
Barramento	Compatibilidade	Em sua maioria são compatíveis com qualquer IP (*Intelectual Property*) incluindo os softwares.	+
Chave *Crossbar*		Em sua maioria são compatíveis com qualquer IP (*Intelectual Property*) incluindo os softwares.	+
Network-on-Chip		São necessários adaptadores (*wrappers*) entre os IPs e os softwares precisam de sincronização em sistemas *multi-core*.	-
Barramento	Complexidade	Conceitos simples e bem compreendidos.	+
Chave *Crossbar*		Conceitos simples e bem compreendidos.	+
Network-on-Chip		Projetistas precisam de uma reeducação em função dos novos conceitos.	-

Na seção que segue, serão comentadas as principais características das redes-em-chip e as vantagens e desvantagens em seu uso.

2.2 Redes-em-chip

A arquitetura de redes-em-chip busca incorporar conceitos de redes de computadores para dentro do chip, a fim de prover qualidade de serviço na comunicação tendo como principais características, segundo (MATOS, 2010):

- Paralelismo (BJERREGAARD,2006): devido à multiplicidade de caminhos possíveis em uma NoC;
- Estruturação e o gerenciamento de conexões em tecnologias submicrônicas (BENINI, 2002), (DALLY,2001): utilização de fios mais curtos, ponto-a-ponto entre os roteadores, com menor capacitância parasita;

- Compartilhamento de fios (BOLOTIN,2004), (DALLY,2001): possibilitando sua utilização de maneira mais eficiente;
- Confiabilidade (VELLANKI,2004): possibilitando o uso de várias técnicas que permitem aumentar a confiabilidade do sistema;
- Escalabilidade (GUERRIER,2000), (BJERREGAARD,2006): permitindo adicionar mais módulos a rede sem comprometer o desempenho do sistema;
- Reusabilidade (BENINI, 2002): possibilitando aproveitar a mesma estrutura de comunicação e os núcleos já existentes em aplicações distintas;
- Decisões de roteamento distribuídas (BENINI, 2002), (GUERRIER,2000): permitindo um maior balanceamento na utilização dos recursos da rede.

Se analisarmos as futuras aplicações previstas pelo ITRS 2011 (ITRS, 2011), que utilizarão centenas ou milhares de núcleos, podemos observar que existem algumas características chaves presentes nas redes-em-chip para esse cenário. O paralelismo de comunicação e a escalabilidade permitirão que mesmo em sistemas com muitos elementos, o desempenho não sofrerá grandes perdas. Outro fator essencial é relativo ao tempo do projeto, onde a reusabilidade das estruturas das redes-em-chip permitirão que o *time-to-market* continue sendo atingido. No entanto, nesse tipo de arquitetura, ainda temos algumas desvantagens como (GUERRIER, 2000):

- Considerável consumo de área de silício e potência em relação à estruturas mais simplificadas, como barramentos ou pequenos crossbares, o que se deve em função do uso de roteadores e interfaces que não são necessários em outras possibilidades de conexões;
- Necessidade do uso de interfaces de rede: os núcleos conectados à rede necessitam de adaptadores de protocolo para a comunicação;
- Latência (tempo para transmissão de pacotes): causada por contenções na rede e em função dos roteadores;
- Complexidade de projeto.

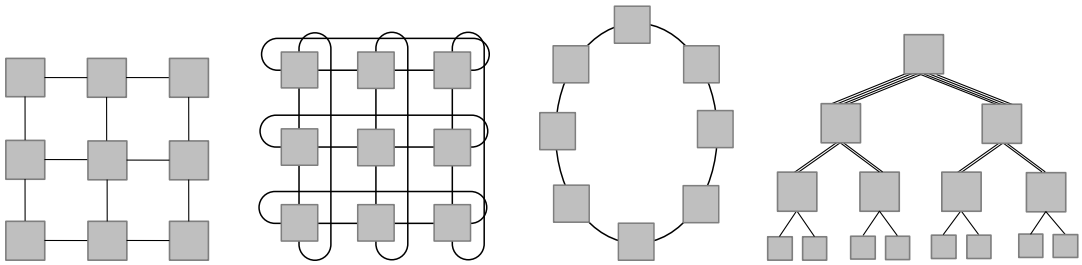
Assim, observando essas limitações, pode-se presumir que o custo das redes-em-chip, relativo à área e potência, tenderá para uma porcentagem importante no projeto. Em vista desse problema é que surge o trabalho investigativo apresentado nessa dissertação, que visa diminuir os custos sem prejudicar o desempenho.

2.2.1 Topologia

Uma rede-em-chip deve prover conectividade entre um conjunto de nodos. Essa conectividade é realizada indiretamente, ou seja, para se comunicar com outros nodos é necessário passar por um ou mais elementos intermediários, chamados roteadores. Esses estão conectados fisicamente com outros roteadores em uma determinada disposição. A forma em que estão arranjados é definida como a topologia da rede de comunicação. A topologia é uma característica que está diretamente relacionada ao desempenho e ao custo da rede (DAS, 2009). As aplicações possuem diferentes comportamentos de comunicação, onde alguns nodos podem apresentar maiores necessidades de comunicação do que outros, tornando a sua localização um fator importante para

comunicar em menor tempo. Em relação ao custo, caso seja definido uma topologia onde existam muitos enlaces, pode-se aumentar o custo dos roteadores, que tenham mais canais e diferentes métodos de roteamento, acarretando em mais potência e área.

Os tipos de topologias mais tradicionais são a malha (do inglês *mesh*), toróide (do inglês *torus*), anel (do inglês *ring*) e árvore (do inglês *tree*). A topologia malha é a mais utilizada pela literatura, e é composta por uma estrutura homogênea onde cada roteador é conectado a quatro roteadores adjacentes (chamados de norte, sul, leste e oeste), com exceção dos cantos da rede, possuindo largura de dados fixa. A topologia toróide é muito semelhante a malha, tendo como diferença a conexão das extremidades com a extremidade oposta, possibilitando mais opções de caminhos. A topologia em anel é a conexão sequencial entre os elementos, podendo obter um roteador de menor custo (de apenas dois canais), mas, no entanto, apresenta escalabilidade limitada já que o desempenho decresce à medida que mais nodos são adicionados à rede. Na topologia em árvore, o modelo mais utilizado é a *fat-tree*, onde a largura dos canais diminui a medida que se aproxima das folhas, já que é onde há maior concentração de tráfego na rede (PASRICHA, 2008). A figura 2.2 apresenta algumas das topologias de NoC acima mencionadas.



(a) (b) (c) (d)

Figura 2.2 - Exemplos de topologias: malha (a), toróide (b), anel (c) e árvore (d).

2.2.2 Deadlock, Livelock e Starvation

Para assegurar a correta funcionalidade da rede em termos de entrega de mensagens, uma rede deve evitar *deadlock*, *livelock* e *starvation* (DUA 97)(PAT 96). *Deadlock* pode ser definido como uma dependência cíclica entre dois ou mais nodos requisitando acesso a um conjunto de recursos, de forma que nenhum possa obter progresso algum, independente da sequencia de eventos que ocorra. *Livelock* refere-se a mensagens circulando na rede sem fazer nenhum progresso em direção ao seu destino. Este é um problema que normalmente atinge algoritmos de roteamento adaptativos não mínimos (Seção 2.2.3). Isto pode ser evitado, restringindo o número de desvios que a mensagem pode efetuar. *Starvation* ocorre quando uma mensagem requisita um recurso, sendo bloqueada por um tempo indeterminado porque o recurso está alocado sempre a outra mensagem.

2.2.3 Roteamento

Em qualquer comunicação presente na rede-em-chip, temos o nodo que quer enviar a mensagem (origem) e o nodo que irá receber (destino). Ambos devem possuir um endereço único na rede, para assim, ser definido qual o caminho que deve ser percorrido da origem até o destino. O processo que determina como enviar as mensagens pela rede é chamado de roteamento. Diferentes estratégias podem ser empregadas para definir o algoritmo de roteamento (MELLO 2006) (PASRICHA, 2008) (ZEFERINO, 2003):

- Número de destinos: O algoritmo pode permitir que uma mensagem tenha um único destino (unicast) ou múltiplos destinos (multicast).
- Decisão de roteamento: A decisão do caminho a ser percorrido pode ser centralizada, ou seja, definido na origem, ou pode ser decidido distribuidamente, onde cada roteador examina qual a próxima rota para aquela mensagem.
- Adaptabilidade: A estratégia pode ser determinística, ou seja, já é sabido qual caminho se percorre entre as origens e destinos, ou a estratégia pode seguir uma tendência adaptativa. Nesse caso, são utilizadas informações da rede, como condições do trafego para decidir qual rota é a mais favorável.
- Minimalidade: Nesses algoritmos, tem-se como meta utilizar sempre o menor caminho possível, independente se existe congestão em determinados caminhos, facilitando o controle visto que sempre se aproxima do destino. As soluções não-minímas permitem outras opções de caminhos, mas podem apresentar o problema de *livelock.*

O algoritmo de roteamento deve considerar também a topologia, pois as rotas dependem da organização da rede. Inúmeros algoritmos de roteamento têm sido propostos nos últimos anos, embora os algoritmos mais utilizados ainda sejam os de ordenamento por dimensão (DOR – *dimension-ordered routing*) devido a sua simplicidade. Estes algoritmos de roteamento são classificados como unicast, distribuído, determinísticos e mínimos; e são frequentemente utilizados em redes de topologia malha e toróide. Um exemplo de algoritmo determinístico muito presente na literatura é o X-Y (BOBDA, 2005), observado na Figura 2.3. Esse algoritmo primeiramente envia pacotes na direção de X e somente quando não precisa percorrer mais nessa direção é que passa a percorrer na direção Y, até atingir o seu destino. No entanto, essa estratégia não permite obter alternativas de rotas para as mensagens, tendendo a criar maiores contenções na rede.

Outra abordagem muito utilizada na literatura são os algoritmos de roteamento adaptativos, como visto em (BOBDA, 2005), (MING, 2006), (HAIBO, 2007), exemplificados na Figura 2.3 (modificada de (MATOS, 2010)). A definição da rota por este tipo de algoritmo é decidida conforme a situação de tráfego nos canais ou conforme alguma outra condição estabelecida, como por exemplo, a detecção de *links* falhos na rede. Dessa forma, os algoritmos de roteamento adaptativos podem optar por um caminho diferente para diminuir a latência e aumentar a vazão da rede (MING, 2006), ou para tolerar falhas que tenham sido detectadas em algum canal ou enlace da NoC (HAIBO, 2007). No entanto, quando um algoritmo totalmente adaptativo é escolhido para o roteamento de mensagens na rede, situações de *deadlock* podem ser um problema. Alguns trabalhos propõem soluções para contornar estas situações, e este é

um assunto que há algum tempo tem sido discutido na literatura (DUATO, 1993), (DUATO, 1995).

Outro desafio com relação ao uso dos roteamentos totalmente adaptativos é preservar o ordenamento dos flits no receptor, já que cada flit pode percorrer uma rota diferente até o destinatário. Mecanismos de reordenamento já foram propostos como solução para este problema (KIM, 2006).

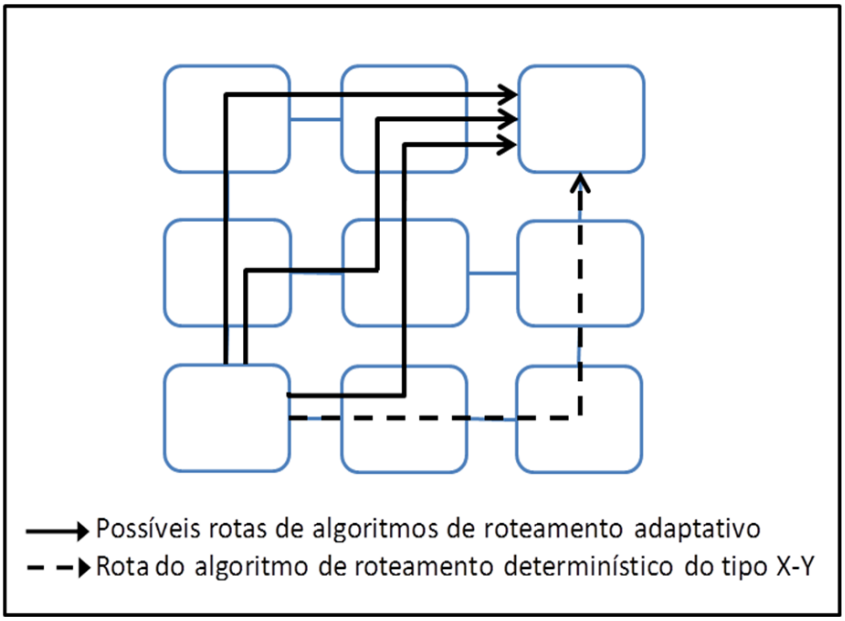


Figura 2.3 - Exemplos de algoritmos de roteamento e suas rotas para enviar uma mensagem do nodo A ao nodo B.

2.2.4 Modos de Chaveamento

A transmissão de dados pela rede é realizada através de chaveamentos ocorridos internamente em cada roteador de um determinado caminho. Os métodos de chaveamento mais utilizados e encontrados na literatura são o chaveamento por circuito e o chaveamento por pacotes. O chaveamento por circuito utiliza a analogia de um circuito fechado para a transferência dos dados, ou seja, necessita que todas as chaves estejam configuradas desde a origem até o destino. Por outro lado, temos o chaveamento por pacotes, onde os dados são abstraídos para elementos de menor granularidade, que são enviados aos poucos pelos roteadores da rede, e assim que vão sendo transmitidos os chaveamentos vão sendo configurados.

2.2.4.1 Chaveamento por Circuito

Neste tipo de chaveamento um caminho dado pelo roteamento entre a origem e o destino é reservado antes das mensagens começarem a ser enviadas, cria-se um circuito fechado. Pode-se dizer que é estabelecido um caminho físico direto entre fonte e

destinatário para a transferência das mensagens que deve ser mantido até o término da comunicação, evitando contenção para aquela mensagem. Porém, deve-se mencionar que um caminho não pode ser reservado se já existir algum caminho em uso, sendo necessário esperar. Os circuitos fechados são liberados durante o avanço do identificador de término da mensagem até o destino final (CHANG, 2006).

2.2.4.2 Chaveamento por Pacotes

Para a melhor compreensão desse modo de chaveamento, é preciso compreender as diferentes granularidades dos dados nessa abordagem. Uma mensagem é dividida em pacotes e cada pacote é particionado em unidades menores, denominadas de *flits* (*flow control unit*). Cada *flit* é formado por um ou mais *phits*, sendo que cada *phit* tem a largura do canal físico de dados. Conforme (PASRICHA, 2008), a figura 2.4 apresenta a estrutura de uma mensagem para facilitar o entendimento destas definições.

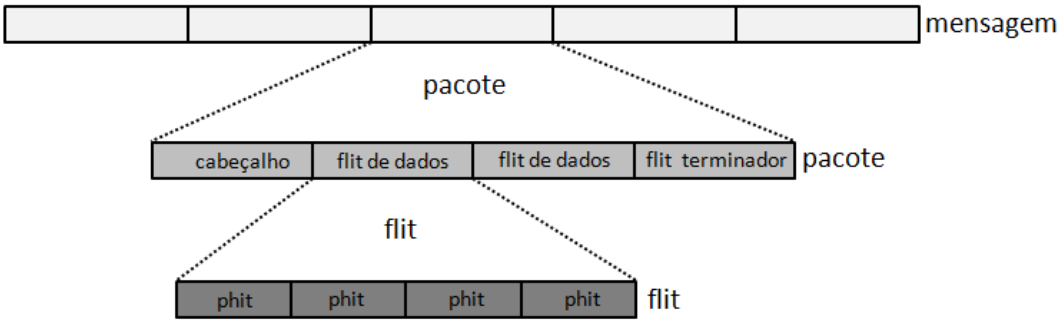


Figura 2.4 - Composição de uma mensagem por pacotes, flit e phits.

O chaveamento por pacotes é realizado transmitindo sem a necessidade de reservar o caminho entre origem e destino. Os pacotes são chaveados para as saídas respectivas a medida que passam pelos roteadores. Caso um roteador esteja ocupado, existe a necessidade de armazenar os elementos do pacote em buffers, para esperar a saída desejada estar liberada. Portanto, nessa estratégia, as mensagens podem conter atrasos por fins de congestionamento na rede. Em relação às estratégias de lidar com as contenções no caminho, podem-se citar três alternativas:

- Armazena e repassa (*store-and-foward* - SAF): este método é utilizado quando as mensagens enviadas entre os nodos da rede são curtas e frequentes. Nesse caso, os pacotes possuem um cabeçalho com as informações de roteamento. Cada pacote reserva o caminho necessário até o seu destinatário e os pacotes são repassados através de um protocolo entre os buffers presentes nos canais.

- Transpasse Virtual (*virtual cut-through* - VCT): este método apresenta o mesmo conceito de mensagens por pacotes, no entanto, neste caso, o pacote não precisa ser inteiramente armazenado. Quando o canal de saída desejado

estiver disponível, o restante do pacote é repassado diretamente para o canal, sem a necessidade de ser armazenado nos *buffers*.

- *Wormhole*: nesse tipo de chaveamento, o envio dos dados pela rede é feito por unidades de *flits*. Neste método, todos os flits seguem o caminho reservado pelo cabeçalho e somente após o envio do flit de término é que o caminho reservado é liberado. A vantagem desta técnica é que os canais de entrada não precisam armazenar um pacote inteiro e com isso, profundidades menores de *buffers* na entrada dos canais podem ser definidas. *Buffers* de maior profundidade inserem um aumento considerável de área e potência na rede. Com essa técnica, é possível definir uma profundidade de *buffer* que justifique seu uso em termos de custo e desempenho. Esse método apresenta como desvantagem a não possibilidade de intercalar *flits* de diferentes pacotes, já que somente após o envio de todo o pacote é que o caminho alocado é liberado para outras mensagens.

Algumas arquiteturas baseadas em chaveamento por pacotes utilizam uma técnica de multiplexação dos canais para criar múltiplos canais virtuais nos roteadores, podendo trabalhar com diferentes pacotes e diferentes destinos, de forma paralela (SAR, 2000). Essa técnica é chamada de Canais Virtuais (virtual channels – VC) e permite o envio de flits de diferentes pacotes intercalados em um mesmo enlace. Dessa forma, soluciona-se o problema de bloqueio de um canal por um determinado pacote na rede.

2.2.5 Controle de Fluxo

Controle de fluxo é a linguagem utilizada para sincronizar uma transmissão e recepção de informação (DUA 2002). Isso é feito através de um protocolo bem definido que permite ao transmissor informar que deseja enviar e ao receptor informar que esta disponível para receber. Assim, evitamos perdas de dados pela rede, por fatores como contenção na rede e buffers lotados ou duplicação de dados, por pegar repetidas vezes algo que não era para ser enviado.

Segundo (DALLY 2004), podemos dividir o controle de fluxo em três tipos:

- Credit-based: Nesse protocolo o roteador que deseja enviar verifica se o roteador vizinho possui créditos. Se sim, transmite até que os créditos acabem ou a mensagem esteja finalizada. Caso contrário, é aguardado. Geralmente os créditos são implementados utilizando o número de espaços disponíveis no buffer.
- On/Off: O fluxo On/Off é baseado em um único sinal indicando se pode ou não enviar para o próximo roteador. Geralmente o sinal é colocado em Off (não pode enviar) quando o buffer fica lotado e o sinal é colocado em On (pode enviar) quando o buffer está quase vazio.
- Ack/Nack: A técnica Ack/Nack não tem muita relação direta com o buffer, ou seja, o dado é enviado independente de saber se o buffer está disponível ou não. Após o envio é enviada uma mensagem informando se o dado foi armazenado corretamente (ack) ou se o dado foi descartado (negação de ack/Nack) necessitando uma retransmissão.

O controle de fluxo utilizado nesse trabalho é baseado no sistema Ack/Nack com a adição do sinal de valid, que serve para informar o envio de um dado. Esse protocolo é amplamente conhecido como *handshake*. A figura 2.5 ilustra como funciona o protocolo de *handshake*.

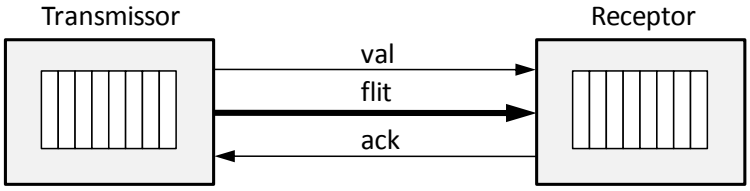


Figura 2.5 - Controle de fluxo baseado em Ack/Nack - *Handshake*.

2.2.6 Rede SoCIN

A rede-em-chip utilizada como base para as propostas arquiteturais, bem como arquitetura referência para comparações, é a rede SoCIN (*System-on-Chip Interconnection Network*) (ZEFERINO, 2003). A SoCIN possui topologia malha, algoritmo de roteamenteo X-Y, chaveamento por pacotes e controle de fluxo por *handshake*. Além disso, seu sistema de memorização é dado por FIFOs nas portas de entrada.

A rede SoCIN utiliza chaveamento por pacotes do tipo *wormhole.* Um flit nessa rede corresponde à largura do canal físico da rede. Cada flit possui tamanho de *n+2* bits, sendo esses bits utilizados para indicar se a informação do flit é um cabeçalho (*header*), uma carga útil (*payload*) ou um flit de término (*trailer*). O resto dos bits correspondente a *n* são os dados úteis do pacote.

Como o pacote é dividido em flits, a informação de roteamento consta no cabeçalho da mensagem, que tem como função alocar o caminho pelo qual os demais flits seguirão. Neste tipo de chaveamento, não é possível intercalar flits de pacotes diferentes pelo mesmo canal lógico, sendo assim, um pacote com todos os seus flits deve atravessar o canal antes de liberá-lo para outro pacote.

O esquema de arbitragem, que decide quem tem prioridade sob uma determinada saída, utilizado na SoCIN é dinâmico e distribuído e, para tanto, faz uso do algoritmo *Round-Robin*, que garante também que nenhum pacote fique bloqueado por menor prioridade permanentemente na rede (*starvation).*

A técnica de controle de fluxo utilizada na rede SoCIN é baseada no protocolo de *handshake*, em que o emissor informa ao receptor a intenção de enviar um dado, pelo sinal de dado válido, e este responde se o *buffer* consumiu ou descartou o flit, pelo sinal de reconhecimento. Esse controle de fluxo é feito através dos sinais de *valid* e *ack.*

O roteador utilizado na SoCIN é chamado de RASoC e possui 5 portas bidirecionais chamadas de L(*Local*), N(*North*), E(*East*), S(*South*) e W(*Weast*). Cada porta possui dois canais unidirecionais denominados canal de entrada e canal de saída. O roteador RASoC utiliza um *crossbar* para cada saída para conectar às entradas. Este roteador possui 3 dimensões configuráveis: largura dos canais de comunicação, profundidade dos buffers e largura da informação de roteamento no cabeçalho do pacote.

A figura 2.6 (MATOS, 2010) apresenta uma rede SoCIN 3x3 com a identificação de algumas partes que a constituem, como buffer nos canais de entrada, os crossbares, os enlaces (links) e os núcleos.

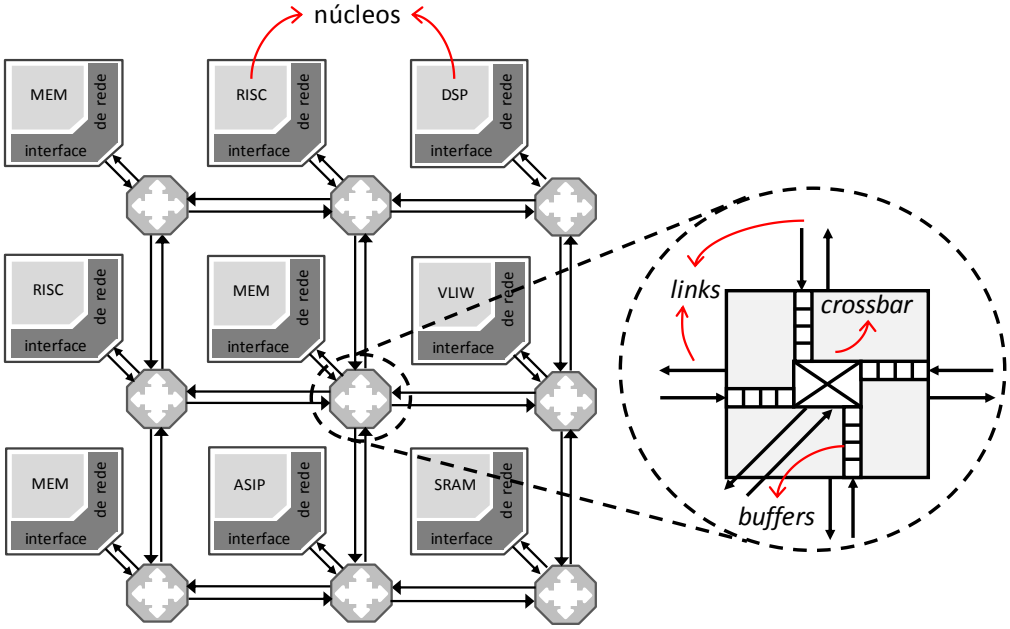


Figura 2.6 - Exemplo de uma SoCIN 3x3 e as partes que a constituem.

2.2.6.1 Modificação Arquitetural no RaSoC

Para esse trabalho de dissertação, realizou-se uma pequena modificação na arquitetura original do roteador RaSoC da rede SoCIN. Observando as redes-em-chip encontradas no estado da arte - importantes métricas de comparação – constatou-se que para as análises de custo e desempenho é necessária um arquitetura com estruturas de *pipeline*. A modificação permite que a rede SoCIN obtenha um maior desempenho, tornando as análises mais condizentes com novas propostas. Todas as arquiteturas elaboradas utilizam a SoCIN modificada (com *pipeline*) como princípio. Da mesma forma, todos os experimentos consideram a nova versão como NoC convencional para fins de comparação.

O RaSoC com *pipeline* possui cinco etapas, demonstradas na Figura 2.7, divididos estratégicamente para proporcionar o maior desempenho dessa estrutura. A primeira é chamada de *Input Protocol* (IP), responsável por separar o protocolo de fluxo de dados handshake. A segunda parte compreende o armazenamento dos dados de entrada, definido como Bufferização (BUF). O terceiro estágio é definido como *Input Control* (IC), onde ocorre a avaliação do cabeçalho do pacote para definir qual a porta de saída deve ser requisitada. A quarta etapa é chamada de *Switch Allocator* (SA), que é a computação de alocação do chaveamento, onde o árbitro decide se a requisição será atendida ou não. Por fim, há o *Switch Selector* (SS), que se refere à conexão da entrada com a saída pela chave crossbar da porta de saída.

(a) (b)

Figura 2.7 – Pipeline no roteador RaSoC. (a) RaSoC original. (b) RaSoC modificado.

Apesar do fato dessa modificação aumentar a latência em dois ciclos a mais, por necessitar cinco ciclos de relógio para um flit passar pelo roteador ao invés de três, a frequência de operação passou de 400 MHz para 1,5 GHz para uma tecnologia de 65nm em células padrão, um aumento de quase quatro vezes mais.

2.3 Considerações

Nessa seção, foram apresentados conceitos básicos de interconexões intrachip. Além disso, diferentes características das redes-em-chip foram explicadas, a fim de permitir um melhor entendimento do escopo de atuação e estudo do presente trabalho de dissertação. Apesar de o foco principal estar nas redes-em-chip, se vê necessário ter a compreensão de que existem outras possibilidades para interconectar núcleos de sistemas complexos. Essa variedade de opções é importante para definir qualquer estratégia de projeto, bem como incorporar às novas ideias antigos conceitos.

3 REDES-EM-CHIP RECONFIGURÁVEIS

Uma área de grande importância devido ao contexto atual dos MPSoCs são as redes-em-chip reconfiguráveis. A estratégia da reconfiguração significa agregar inteligência em uma dada arquitetura para utilizar seus recursos de maneira mais eficiente, ou seja, conforme o estado atual do sistema. Dessa forma, a rede pode se adaptar a fim de obter vantagens sobre determinados aspectos, como por exemplo, maior desempenho, baixo consumo de energia, possibilidade de tolerar falhas, dentre outros.

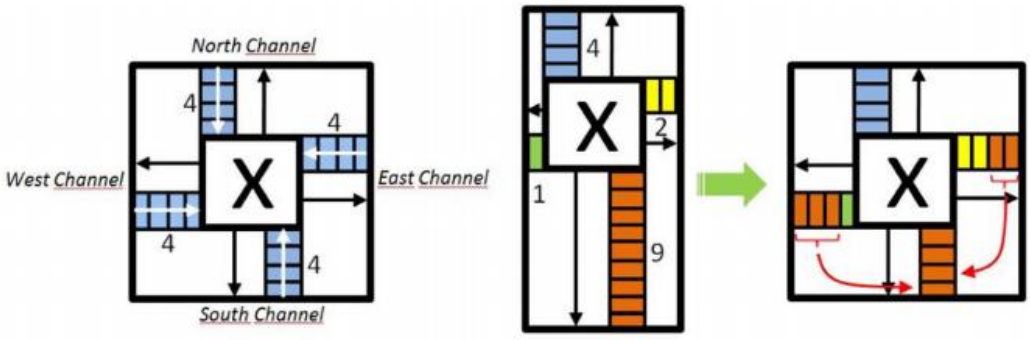
3.1 Trabalhos Relacionados

No contexto de redes-em-chip reconfiguráveis, existem diferentes possibilidades de reconfiguração da arquitetura para satisfazer uma determinada necessidade ou melhorar alguma característica do sistema. Nessa subseção, são apresentadas diferentes abordagens encontradas na literatura que utilizam as estratégias de reconfiguração de NoC conforme determinadas condições do sistema. A primeira abordagem que será apresentada refere-se à reconfiguração dos elementos de armazenamento, os *buffers*. Os *buffers* são responsáveis pelo armazenamento dos pacotes que trafegam pela rede nos canais de cada roteador. Outra estratégia de adaptabilidade em NoCs é possibilitar a alteração das conexões entre os nodos conforme o tráfego entre eles, criando um caminho dedicado ponto-a-ponto. E por último serão apresentadas ideias que modificam logicamente o modo que os elementos estão associados. Assim, é gerada uma nova topologia na rede, orientada pela afinidade de comunicação dos núcleos da aplicação.

Relativo à aplicação de reconfiguração dos sistemas de armazenamento, pode-se encontrar a proposta de (MATOS, 2009). Nesse trabalho, um controle distribuído modifica as conexões entre os *buffers* e as portas de entrada conforme os requisitos de comunicação da rede. Isso significa emprestar elementos de armazenamento às portas vizinhas conforme a necessidade de vazão dos dados. Apesar de implicar em mais área no roteador, esse sistema possui uma maior eficiência na utilização dos elementos de memorização. Sendo assim, é possível utilizar *buffers* menores sem perder desempenho, diminuindo o custo final do roteador, principalmente pelo fato que o principal custo de potência vem desses elementos (MATOS, 2009).

O princípio dessa proposta está no fato de que, tratando-se de aplicações com núcleos heterogêneos, as portas do roteador não vão requerer a mesma taxa de

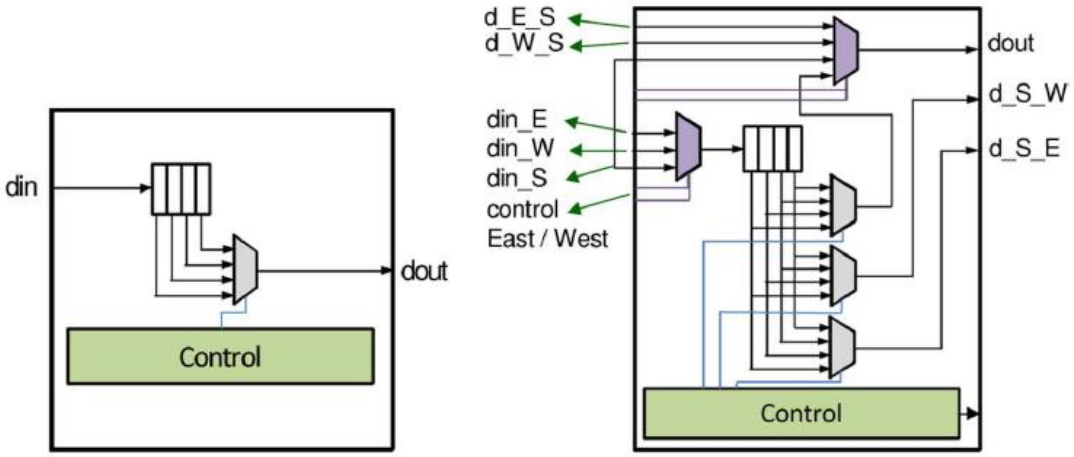
comunicação para todos os núcleos. Sendo assim, aqueles canais que apresentam taxas de comunicação mais altas podem ser beneficiados com esta estratégia de empréstimo dos elementos de armazenamento (*slots*) do *buffer* entre os canais. Na Figura 3.1, pode-se observar diferentes representações de uma reconfiguração da estrutura de armazenamento. Nessa demonstração, a Figura 3.1.a mostra como o roteador foi fabricado, ou seja, possui o mesmo número de elementos de armazenamento para cada porta de entrada; na figura 3.1.b é representada uma distribuição desses dada às necessidades de uma aplicação, ou seja, onde a porta sul apresenta maior demanda; e, por fim, a figura 3.1.c nos mostra de onde vieram os elementos extras para complementar o *buffer* da porta sul.



(a) (b) (c)

Figura 3.1 - (a) Organização Física do Roteador. (b) Representação da distribuição dos elementos de armazenamento. (c) Reconfiguração dos *Buffers*

A proposta foi baseada na arquitetura da rede-em-chip SoCIN (ZEFERINO, 2003) e esse mecanismo de empréstimos é realizado por uma série de multiplexadores e um controle em cada *buffer* de entrada, com exceção da porta local. Os multiplexadores permitem alterar o rumo do fluxo de dados, conforme mostra a Figura 3.2.



(a) (b)

Figura 3.2 - (a) Arquitetura do Buffer de entrada original. (b) Arquitetura do Buffer de entrada reconfigurável.

Os controladores presentes nessa arquitetura comunicam-se, enviando sinais de requisição ou de disponibilidade de alocação de elementos em seus *slots* de memória. Assim, quando existe disponibilidade em uma porta e necessidade em outra, o empréstimo pode ser realizado, ajustando os habilitadores dos multiplexadores de forma a alterar o fluxo de dados para àqueles elementos.

Em (MATOS, 2009), são apresentados experimentos com quatro *benchmarks*: MPEG4, VOPD (BERTOZZI, 2005), *Multi-window Display* (SRINIVASAN 2006) e Xbox (ANDREWS, 2006). Os resultados demonstraram que o roteador proposto atinge o mesmo desempenho do roteador original, reduzindo aproximadamente 25% da potência dissipada para o pior caso, e 52% para o melhor caso analisado. Além disso, comparado ao roteador convencional, essa proposta pôde alcançar o mesmo desempenho com um *buffer* 64% menor. Portanto, a possibilidade de reconfigurar os elementos de armazenamento conforme as demandas de comunicação das aplicações permite obter uma alta eficiência de utilização desses recursos, possibilitando o desenvolvimento de roteadores menores.

Outra área de pesquisa investiga a configuração de caminhos dedicados entre dois ou mais nodos na rede, criando um ponto-a-ponto lógico. Essa abordagem, também chamada de *bypass* (lógica de passagem direta), permite acelerar as comunicações na rede através da passagem direta das mensagens pelos roteadores, ou seja, evitando o hardware de controle do roteador e demais protocolos. A grande maioria das propostas que aborda essa estratégia utiliza a técnica de chaveamento por circuito para realizar as configurações, conforme encontrado nas arquiteturas HCS (JERGER 2008) e VIP (MODARRESSI 2010). Porém, (OGRAS 2005) realiza as configurações através da fabricação física das conexões especializadas.

Em (JERGER 2008) é apresentada uma arquitetura de roteador que permite trabalhar com chaveamento por circuito e chaveamento por pacotes, chamada de HCS (Hybrid Circuit Switching). Nessa proposta, o autor utiliza duas redes malhas separadas, onde uma comunica apenas informações de controle para configuração do sistema e a outra os dados úteis dos pacotes. Os roteadores da rede encarregados de trafegar os dados possuem uma arquitetura diferenciada, com elementos específicos para trabalhar com cada chaveamento (circuito e pacote), ou seja, a lógica e os canais são duplicados, o que acarreta em um alto custo de área adicional. Consequentemente, o custo de área reflete na dissipação de potência, que também apresentará resultados elevados, visto que não há reuso de recursos. O esquema dessa arquitetura pode ser observado na Figura 3.3.

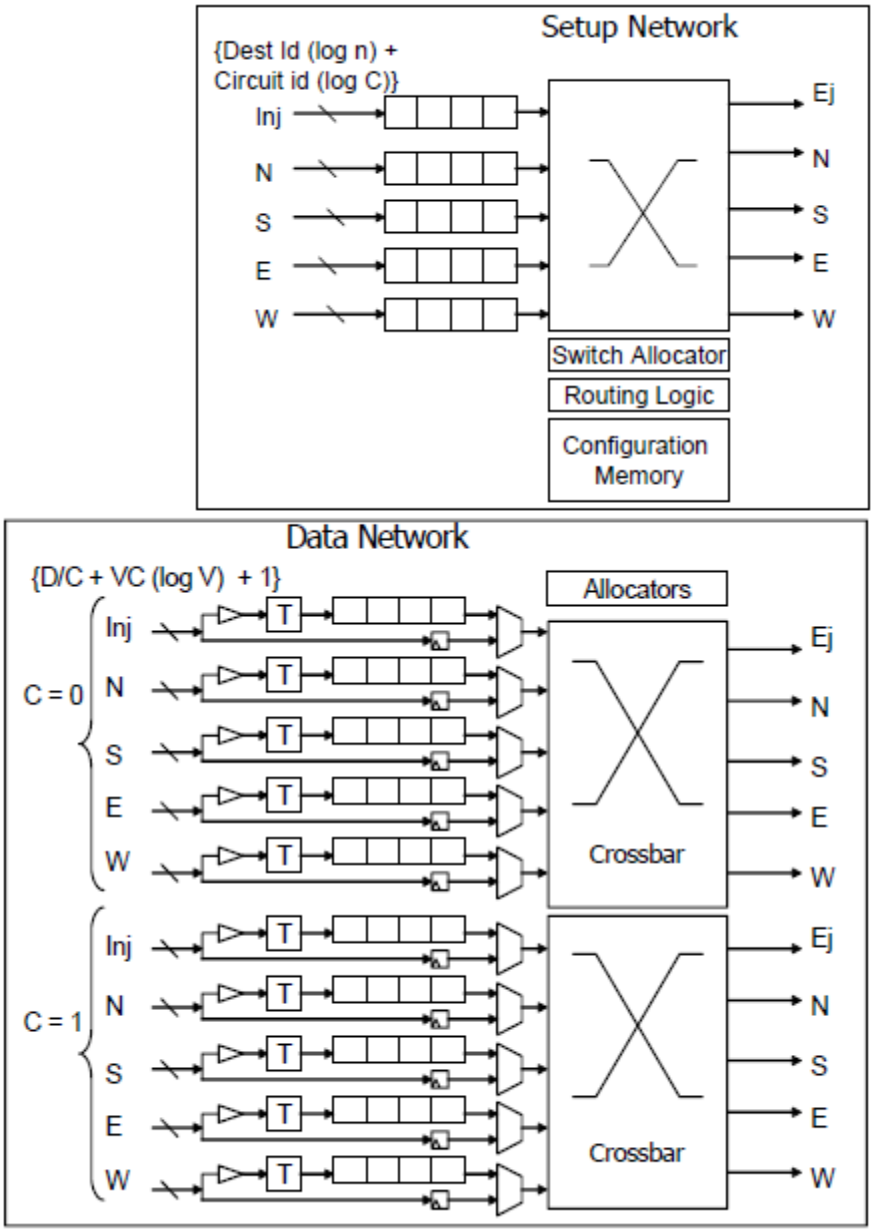


Figura 3.3 – Arquitetura dos Roteadores HCS.

A estratégia de utilizar uma rede especializada na configuração da rede de dados visa reduzir o tempo gasto na configuração inicial para utilizar o chaveamento por circuito. Essa rede de controle utiliza um pacote de configuração que cria o circuito pelo caminho à medida que esse pacote trafega até o destino, não necessitando qualquer mensagem de confirmação. Quando existe um conflito durante a configuração, devido a um circuito já existente, o antigo circuito é desativado retornando ao chaveamento por pacotes e o novo circuito prevalece. Ao desativar o antigo circuito, uma mensagem de aviso é enviada à sua origem, para que possa ser alterado o método de envio dos dados.

Esse trabalho utiliza também algoritmos para lidar com sistemas multiprocessados com hierarquias de cache. Os experimentos foram obtidos utilizando um simulador de MPSoCs com os seguintes benchmarks: TPC-H, TPC-W, SPECweb99 e SPECjbb2000

e diferentes Splash2. Infelizmente, os autores não apresentam nenhum resultado de síntese para permitir melhores análises e comparações, embora seus experimentos demonstrem uma redução de aproximadamente 23% na latência e aumento do desempenho geral em 7% quando comparado com um roteador convencional melhorado chaveado por pacotes, que possui latência de um ciclo por etapa para baixas cargas.

Em (MODARRESSI 2010), o autor propôs a arquitetura VIP (virtual point-to-point), onde apresenta uma estrutura capaz de passar direto pelos roteadores, assim como visto em (JERGER 2008). A diferença está na utilização da técnica de canais virtuais, onde um mesmo canal físico é multiplexado no tempo para permitir a passagem de múltiplos pacotes. Dessa forma, esse autor utiliza um desses canais virtuais para utilizar na lógica de passagem direta, enquanto os outros sempre utilizam chaveamento por pacotes. Assim, a consistência da rede é mantida, visto que pelo menos um canal estará livre para a utilização de chaveamento por pacote. No entanto, essa estratégia cria muita duplicação de hardware e, tendo em vista que não há nenhuma forma de compartilhamento de recursos de armazenamento, possui alto custo adicional de área e potência. Além disso, possui um registrador a mais para cada porta de entrada para armazenar temporariamente flits que chegam ao canal virtual. A arquitetura do roteador proposto pode ser observada na Figura 3.4.

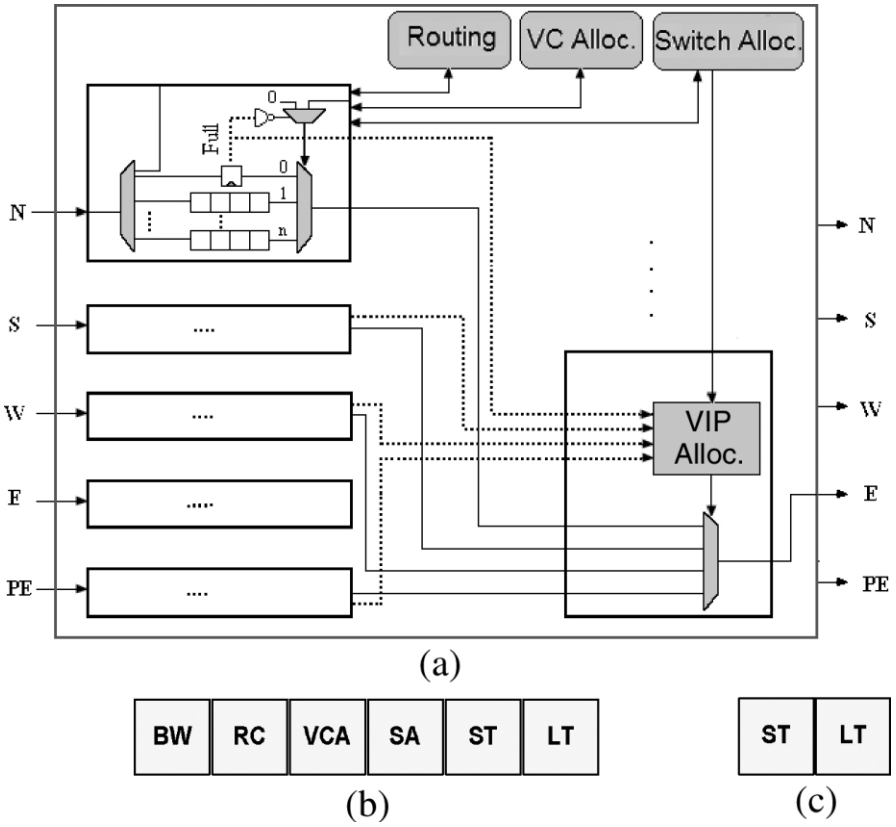


Figura 3.4 - Arquitetura VIP

Nesse trabalho o autor menciona poder controlar sua reconfiguração estaticamente, em tempo de pré-execução, e dinamicamente, durante a execução. Porém, para a implementação dinâmica salienta os custos necessários como monitores de trafego e um

controle para tomar a decisão de configuração, e define os parâmetros que julga necessário para realizar essas decisões.

Experimentos foram feitos com benchmarks para MPSoCs e demonstraram melhora de desempenho perante redes-em-chip convencionais. Também foi explorada uma proposta de reconfiguração dinâmica com uma rede de configuração de baixo custo, onde foi suposto que em MPSoCs existe a tendência da comunicação ocorrer sempre entre os mesmos nodos, permitindo ampla utilização da técnica VIP em uma mesma aplicação.

Em (MODARRESSI 2009) é proposta uma rede híbrida que permite chaveamento por pacotes e por circuito de forma similar ao proposto por (JERGER 2008), possuindo uma rede de controle (Snet) e duas sub-redes de dados: a Pnet e a Cnet. A Pnet é utilizada para aplicar o tradicional chaveamento por pacotes, enquanto que a Cnet é reservada para o chaveamento por circuito ou lógica de passagem direta. A rede de configuração, chamada Snet, é utilizada para estabelecer o circuito e essa solução utiliza o algoritmo de multiplexação por divisão espacial (SDM – Spatial-division Multiplexing) para dividir os caminhos. Consequentemente, essa técnica define pequenos grupos de caminhos e dividem eles entre os nodos. O trabalho comenta que essa solução apresenta um custo adicional de área mínimo (10%), mas os resultados globais de área, ou seja, do sistema completo, não são apresentados. Esse pequeno percentual no *overhead* de área é justificado pelo fato de que o roteador base desse trabalho é bastante complexo, possuindo mais que o dobro de área se comparado com um roteador convencional (YOON, 2010), o que torna qualquer adição de hardware insignificante.

Em (OGRAS, 2005) é apresentada uma solução alternativa para criar caminhos dedicados. Nesse trabalho, caminhos são fisicamente colocados durante o processo de projeto do sistema, na etapa de síntese. Portanto, nesse caso, os caminhos são criados fisicamente, sendo necessário, em tempo de projeto, saber a aplicação que será executada no sistema. Apesar de ser uma estratégia que traz ganhos, pois evita custos de configuração e de controle extra de sincronização é extremamente dependente da aplicação.

A linha de pesquisa em NoCs reconfiguráveis visa não somente modificar alguns caminhos para acelerar mensagens separadamente, mas sim modificar todas as conexões que são realizadas para criar logicamente outra topologia na rede, que seja mais eficiente para a aplicação alvo. Nesse contexto, temos os trabalhos desenvolvidos por (STENSGAARD, 2008) e (MODARRESSI, 2011).

Em (STENSGAARD, 2008) é proposta a arquitetura ReNoC, que permite reconfiguração de topologia. Basicamente, chaves existentes ao redor dos roteadores permitem que as conexões entre eles se modifiquem, alterando o modo como esses roteadores estão conectados, ou seja, modificando logicamente a topologia original. Essa característica possibilita que elementos com maiores taxas de comunicação entre si tenham conexões diretas ou com menos roteadores intermediários, diminuindo a latência final do pacote, e consequentemente, obtendo melhor desempenho.

ReNoC é baseado em roteadores com chaves ao seu redor para possibilitar a utilização de chaveamento por circuito em uma rede com chaveamento por pacotes. A técnica de chaveamento por circuito permite criar uma conexão direta entre quaisquer dois roteadores, sob o custo de restringir os *links* dessa comunicação apenas entre eles. Porém, a estratégia é configurar o melhor mapeamento topológico antes de executar

qualquer aplicação, baseado nas características de comunicação da mesma. Na figura 3.5 pode ser observado que a partir de uma topologia fisicamente fabricada, pode-se logicamente ter diferentes possibilidades de conexões.

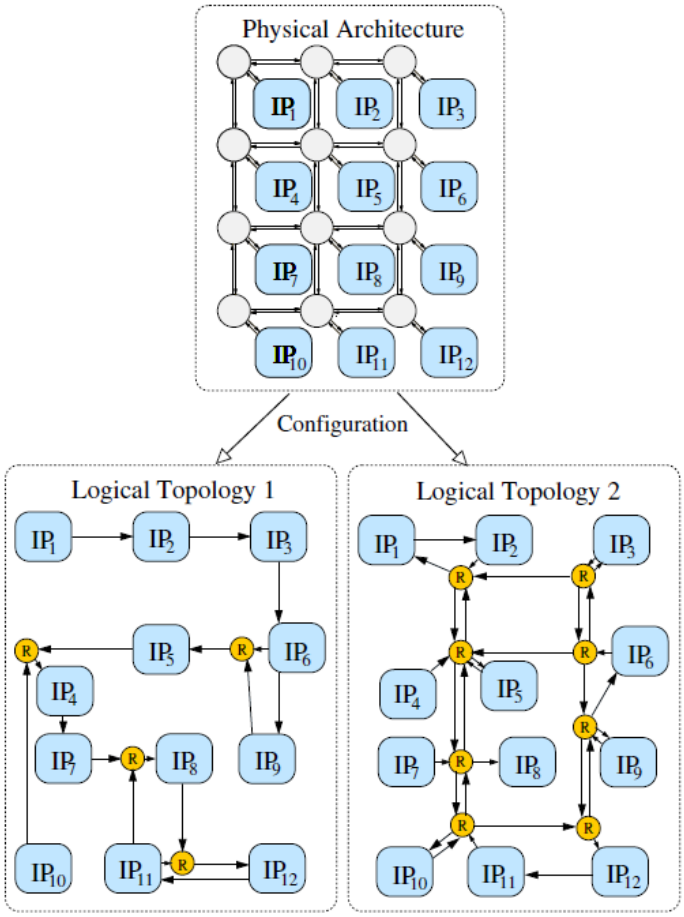


Figura 3.5 - Configuração Topológica da proposta ReNoC.

A arquitetura do ReNoC é composta por um roteador e uma “chave de topologia”. O autor alega que sua proposta é a arquitetura da “chave de topologia”, que pode ser integrada em qualquer arquitetura de roteador já existente, desde que utilize os mesmos protocolos de fluxo de dados e tamanhos dos fios. A “chave de topologia”, definida em (STEENSGAARD, 2008), é composta por um multiplexador para cada porta de saída, que possui como entrada todas as outras portas de entrada e a respectiva porta de saída do roteador, conforme pode ser observado na Figura 3.6.

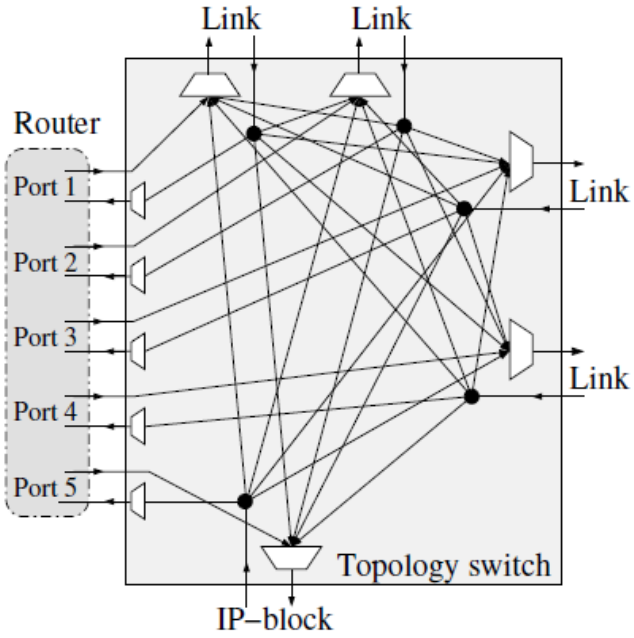


Figura 3.6 - Arquitetura da “Chave de Topologia”

Para os experimentos, os autores desse trabalho também utilizam o benchmark intitulado de Video Object Decoder (VOPD) (BERTOZZI, 2005). Como análise dessa proposta, foram verificados os resultados de uma rede convencional utilizando topologia malha, da ReNoC utilizando topologia malha e da ReNoC utilizando uma topologia configurada para a aplicação em questão. Após o mapeamento da ReNoC para a aplicação VOPD, foi constatado que são necessários apenas 25% dos roteadores quando comparado ao número de roteadores utilizados em uma topologia malha. Pensando nisso, os autores assumiram que os roteadores não necessários seriam desligados durante a execução dessa aplicação, não consumindo potência. Portanto, mesmo essa arquitetura possuindo 10% a mais de área que uma rede-em-chip convencional, devido ao custo das “chaves de topologia”, ela permitiu uma redução na dissipação de potência total do sistema em 56%.

Portanto, essa proposta permite que uma arquitetura com topologia fisicamente implementada possa adquirir diferentes topologias lógicas transparentes às aplicações, apenas sob um custo de tempo de configuração inicial. Isso permite que diferentes topologias possam ser empregadas, mesmo as hierárquicas, tornando o hardware mais eficiente e obtendo melhores resultados de energia. Além disso, é possível implementar em tempo de projeto arquiteturas com um menor número de portas, diminuindo ainda mais os custos de área, sem perder a flexibilidade das conexões, permitindo que sejam configuráveis à taxa de comunicação requerida

A mesma estratégia da arquitetura ReNoC é empregada por (MODARRESSI, 2011) em uma estrutura que reconfigura sua topologia para cada aplicação. Seu objetivo está em reduzir o número de roteadores intermediários presentes nas comunicações com altos fluxos de dados. Dessa forma, pretende-se reduzir a latência geral da rede e obter melhor desempenho aliado a uma menor dissipação de potência.

Para isso, (MODARRESSI, 2011) implementou chaves reconfiguráveis, chamados de *switch box*, que são colocadas entre os roteadores de uma rede com topologia malha 2D, conforme mostra a figura 3.7. Assim, nenhum roteador está interligado diretamente com outro, mas sim, com chaves intermediárias. O autor comenta que um contraponto de sua proposta são os fios longos gerados, quando configurado um caminho muito longo na rede. Para resolver isso, é proposto que algumas chaves possuam *latches*¸ para que atuem como registradores no caso de um caminho que possa prejudicar a frequência de operação do sistema. No entanto, essa inserção acarreta em um aumento de custo, podendo se tornar proibitivo devido ao grande número de chaves necessárias. Outra consideração é a inclusão de um gerenciador de potência para desligar elementos não utilizados durante a execução de uma aplicação, reduzindo a potência estática.

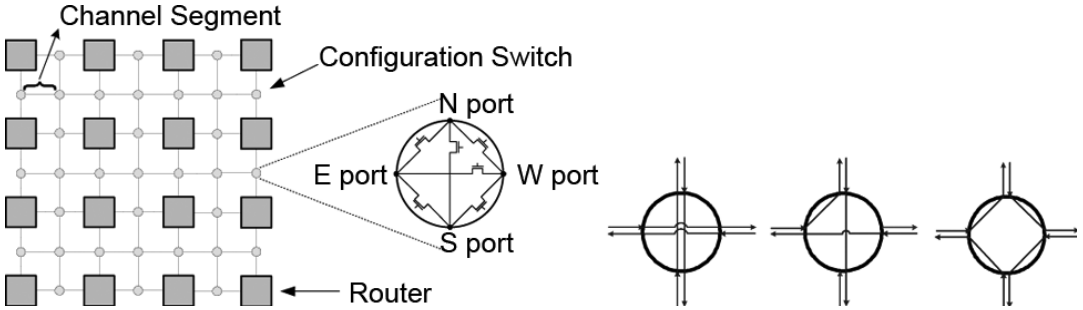


Figura 3.7 – Arquitetura de (MODARRESSI, 2011). (a) Rede em malha 2D com roteadores e chaves *switch-box*. (b) Possíveis conexões do elemento *switch-box.*

Experimentos foram realizados com benchmarks sintéticos e com as aplicações *Multi-window Display* (MWD) (SRINIVASAN 2006), *Video Object Plane Decoder* (VOPD) (BERTOZZI, 2005) e *Multi-media System* (MMS) (HU, 2005). Para a aplicação VOPD, tida como a mais custosa em termos de comunicação e elementos de processamento, é obtida uma melhoria de 7% em latência e de 10% em potência e um custo em acréscimo de área de 12% quando comparado com a arquitetura ReNoC de (STENSGAARD, 2008).

A tabela 3.1 apresenta um resumo dos trabalhos propostos encontrados na literatura relativas à reconfiguração em redes-em-chip.

Tabela 3.1 – Resumo das propostas da literatura de redes reconfiguráveis.

Autor	**Técnica de Reconfiguração**	**Principal Benefício**	**Principal Custo**
(MATOS, 2009)	Empréstimo de Buffers	Potência	Área do controlador
(JERGER 2008)	*Bypass*	Latência	Área da rede extra
(MODARRESSI 2009)	*Bypass*	Latência e Vazão	Área da rede extra
(MODARRESSI 2010)	*Bypass*	Latência	Área dos canais virtuais
(OGRAS 2005)	*Bypass*	Latência, Área e Potência	Perda de reusabilidade
(STEENGAARD 2008)	Reconfiguração Topológica	Latência	Fios longos gerados
(MODARRESSI 2011)	Reconfiguração Topológica	Latência e Vazão	Área das chaves.

A revisão da literatura atual revela que muitas propostas estão sendo desenvolvidas, com o objetivo de acelerar algumas comunicações, permitindo diminuir a latência geral do sistema e melhorar a eficiência energética. Arquiteturas como as (MODARRESSI, 2010), (MODARRESSI, 2011) e (JERGER, 2008) consideram fatores problemáticos como geração de fio longo e adição de circuitos de memorização (maior dissipação de potência). A maioria das propostas não possuem mecanismos automáticos de definir a configuração, sendo necessário um gerente do sistema reconfigurável. Isso prejudica uma análise real dos custos de implementação desses sistemas.

3.2 Estudos Estratégicos

Em relação às propostas de redes-em-chips reconfiguráveis existentes, observou-se uma grande potencialidade na proposta de reconfiguração topológica. Isso se justifica pela premissa de que se a topologia é eficiente (ajustada ao padrão de comunicação), não há tanta necessidade de reconfigurar outras partes da arquitetura, como *buffer* e algoritmos de roteamento, por exemplo, bem como pode ainda evitar a utilização de hardware extra, como nos casos de canais virtuais. Dessa forma, é possível obter maiores ganhos de energia em relação às outras estratégias. Além disso, essa abordagem permite que se configurem topologias hierárquicas, o que está diretamente relacionado com a outra área de pesquisa dessa dissertação que também diz respeito em obter uma melhor topologia para as aplicações em termos de desempenho e energia. Essa compatibilidade entre essas duas linhas de pesquisa, atrelada à necessidade dos projetos *many-cores*, aponta a uma favorável integração dos mecanismos.

Além disso, para realizar a reconfiguração topológica vamos abordar a utilização da criação de múltiplos caminhos dedicados através da utilização de chaveamento por circuito, como encontrado na maioria das propostas acadêmicas (como HCS e VIP), ao invés de chaves reconfiguráveis como na ReNoC. A abordagem escolhida é justificada, pois é previsto um acréscimo de área e potência menor, pois muitos elementos já existentes nos roteadores podem ser reaproveitados, diferentemente da abordagem de acrescentar inúmeras chaves extras. Assim, espera-se obter um impacto de custo de área e potência mínimas. Entretanto, ao entrar no tema de reconfiguração da topologia utilizando chaveamento por circuito, é necessário analisar duas características importantes para o seu projeto arquitetural: o tamanho dos fios gerados e as limitações do método de chaveamento por circuito.

Ao interligar dois pontos quaisquer através de chaves, cria-se o problema do fio longo. Isso implica numa latência relevante se considerarmos percorrer alguns núcleos em uma alta frequência (conforme pode ser observado na subseção 3.2.1), o que pode prejudicar o desempenho de vazão ou a frequência do sistema. Em outro ponto, em relação ao método de chaveamento, é importante entender como configurar de maneira eficiente um chaveamento por circuito e quando essa estratégia é interessante para a aplicação.

Nas próximas subseções são demonstrados estudos em relação ao custo do fio em diferentes nodos tecnológicos, e os custos e estratégias para os métodos de chaveamentos.

3.2.1 Conexões

O tema dos fios nos circuitos integrados não era considerado um desafio para tecnologias mais antigas (180 nm, 350 nm, 600 nm), pois não apresentavam valores consideráveis de latência quando comparados à lógica. Porém, o escalamento tecnológico e o avanço na complexidade das arquiteturas mudaram esse cenário. Esse também foi um dos principais motivos do advento das redes-em-chip, pois o modelo clássico de topologia malha prevê fios curtos nas interconexões entre os roteadores (DALLY, 2001).

Em relação às arquiteturas, observa-se uma tendência gradativa de aumento no número de elementos dentro do circuito integrado, criando estruturas com interconexões maiores e mais complexas para satisfazer todos os requisitos de comunicação. Essas estruturas podem, muitas vezes, em prol de um maior desempenho, criar conexões entre pontos muito distantes, gerando o problema do fio longo. Muitos trabalhos na literatura já evidenciam os problemas de degradação de desempenho ocasionados decorrente dos fios longos, como em (MODARRESSI, 2010), (JERGER, 2008) e (MODARRESSI, 2009).

Além disso, podemos analisar as questões físicas do fio, visto que este não está escalando linearmente para as atuais tecnologias devido ao fato de que o comprimento não reduz na mesma proporção da sua secção transversal (HO, 2001). Isso implica em um aumento na capacitância adjacente resultando em uma redução de latência em escala menor que a dos transistores. Assim, visto que a lógica está mais rápida e o fio não tanto, a latência acumulada de elementos baseados em fios passa a ser relevante para o desempenho dos sistemas. Em resumo, baseado em um modelo de RC para fio (resistivo capacitivo), a latência e potência aumentam proporcionalmente ao comprimento do fio e isso infere que essa medida é uma limitação a qualquer sistema baseado em chaveamento por circuito ou qualquer lógica que exija interconexões mais longas.

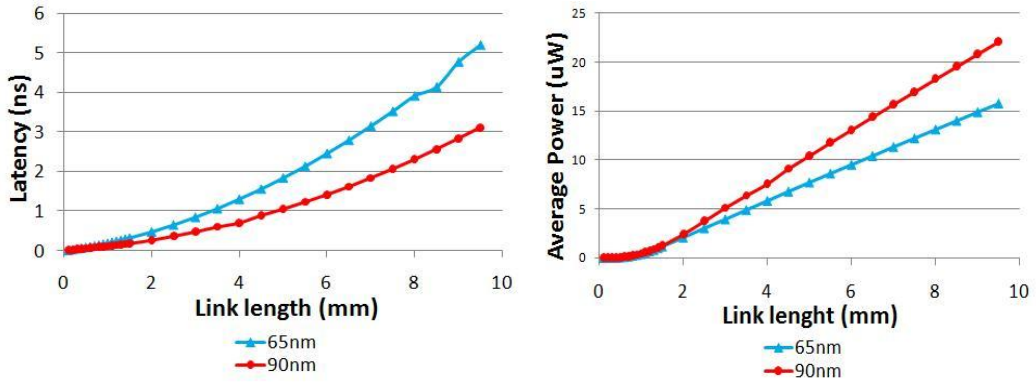
Tendo em vista essa realidade, fez-se necessário realizar um estudo analítico, onde se utilizou a ferramenta Spice para um processo na tecnologia de 65 nanômetros. Verificou-se a relação do comprimento de fio com a latência através de emulações com um modelo RLC-π (SAKURAI, 1983), que descreve o comportamento elétrico do fio. Com as informações tecnológicas obtidas através de (PTM, 2012), obteve-se a relação entre a frequência de relógio e a limitação de comprimento de fio considerando repetidores, conforme demonstrados na tabela 3.2. Os repetidores são elementos introduzidos nos circuitos para restaurar o sinal em conexões, tipicamente chamados de buffers elétricos. Utilizaram-se um e dois milímetros como espaçamento entre os repetidores para obter os resultados, não considerando o fator ideal para cada tecnologia por ser um experimento simplificado. Assim, pode-se definir qual o tamanho máximo que o fio pode ter para propagar sem atraso em determinada frequência de operação (MATOS, 2011).

A tabela 3.2 demonstra que para altas frequências existe uma limitação importante, visto que 3 a 4 mm pode significar a largura de um ou poucos núcleos e, portanto, poucos *hops* na rede já poderiam limitar as diferentes técnicas observadas nas propostas atuais.

Tabela 3.2 - Máximo Comprimento do Fio em determinadas Frequências para 65 nm

Frequência (MHz)	**Max. comp. Fio (mm) 1 repetidor por mm**	**Max. comp. Fio (mm) 1 repetidor por 2 mm**
200	15,3	16,0
300	10,5	11,1
500	6,5	7,3
600	5,8	6,5
700	5,2	5,9
800	4,7	5,3
900	4,3	4,8
1000	3,9	4,5
1500	2,9	3,7

O segundo estudo demonstra que se verificarmos a latência e a dissipação de potência de acordo com o aumento no tamanho do fio, pode-se observar que ambos crescem rapidamente, conforme se observa na Figura 3.8. Esse estudo utilizou processos tecnológicos de 65nm e 90nm para o mesmo modelo RLC-π em simulação Spice.



(a) (b)

Figura 3.8 - (a) Latency and (b) Average power consumption considering the wire length for 65nm and 90nm.

Os resultados acima apresentaram uma resistência de aproximadamente 450 Ω para tecnologia de 65nm e 240 Ω para tecnologia de 90 nm e uma capacitância de 170fF para 65nm e 200 fF para 90nm, o que explica a dissipação de potência encontrada na figura.

Portanto, esse estudo conclui que o projeto dos fios acarreta custos relevantes para os sistemas atuais e futuros, principalmente com a atual escala dos núcleos, cujos tamanhos superam alguns milímetros (ARM, 2010). Sendo assim, deve-se ter cautela

em qualquer projeto que envolva conexões ponto-a-ponto ou com circuito fechado, pois pode gerar fios que gerem limitações ao desempenho do sistema.

3.2.2 Método de Chaveamento

Os métodos de chaveamento considerados nesse trabalho, cujo foco é redes-em-chip, são chaveamento por circuito e chaveamento por pacotes. No entanto, pensando na possibilidade da ocorrência de fios longos, conforme apresentado na subseção anterior, além dessas duas possibilidades de chaveamento foi introduzido o conceito de chaveamento por circuito com barreira temporal ou memorização. Com isso, é possível manter as frequências de operação, visto que mesmo nas comunicações entre roteadores distantes a presença do elemento de memorização permite que a frequência de execução da lógica não seja prejudicado. Sendo assim, foram analisados os desempenhos considerando três formas de chaveamento possíveis: passagem direta (UCS – *unbuffered circuit switching*), passagem direta memorizada (BCS - *Buffered Circuit Switching*) e passagem por pacotes (PS – *Packet Switching*).

Primeiramente, para fins de analise, elaboraram-se equações que descrevem a latência dos três métodos de chaveamento, considerando as implementações que serão apresentadas nesse trabalho. Abaixo são ilustradas essas equações para PS (L_{PS}), BCS (L_{BCS}) e UCS (L_{UCS}),

1.) $$L_{PS} = (PERIOD \times HOPS \times (PIPE_{PS})) + L_W + T_C$$

2.) $$L_{BCS} = (PERIOD \times HOPS \times PIPE_{BCS}) + L_W + T_S$$

3.) $$L_{UCS} = (3xL_{MUXES} + L_{AND} + L_W) \times HOPS + T_S$$

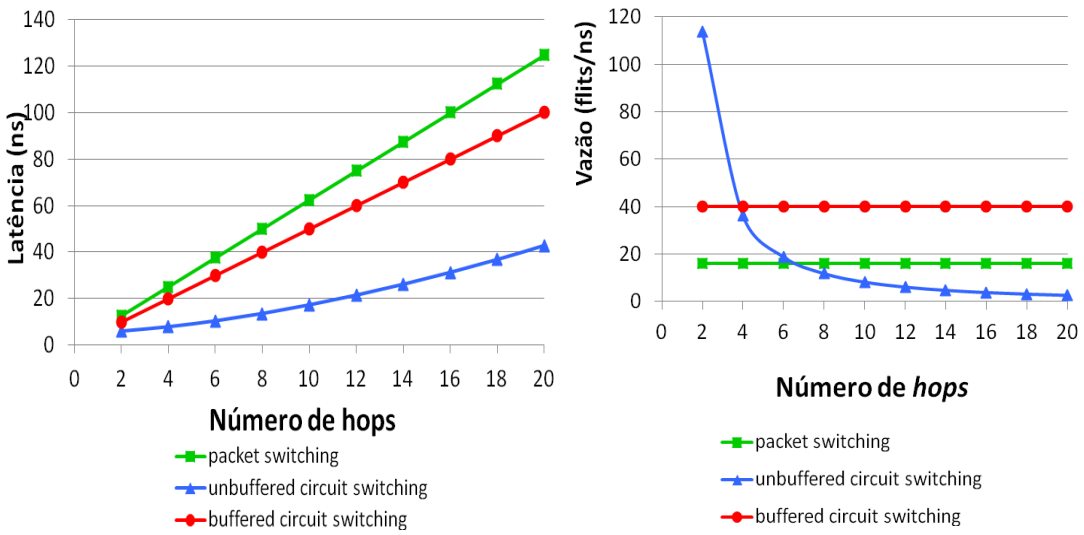
onde:

- PERIOD – Período da frequência de operação;
- HOPS – quantidade de caminhos intermediários entre origem e destino;
- PIPE – quantidade de ciclos presentes no *pipeline*;
- T_C – Tempo de espera por contingência na rede;
- T_S – Tempo de *setup;* configuração inicial do chaveamento por circuito;
- L_W – Latência do fio;
- L_{MUX} – Latência de um multiplexador;
- L_{AND} – Latência de uma porta AND;

A informação de período é assumida nessas análises como 1,25 nanosegundos, refletindo uma frequência de operação de 800 MHz. Em relação à variável HOPS temos

um valor específico para cada mensagem. No aspecto de ciclos de *pipeline* é importante salientar que o número de ciclos por roteador para o chaveamento por pacotes é de cinco ciclos (PIPEPS), enquanto que o número de ciclos para o chaveamento direto memorizado é de dois ciclos (PIPEBCS). A condição de tempo de espera por motivos de contingência na rede é definida pelas condições da rede durante a operação do sistema, não sendo tratado nessa análise. O tempo de inicialização do chaveamento por circuito representa dois ciclos a mais por roteador, um gasto para mandar a mensagem de configuração e outro para receber a informação de circuito fechado. Os tempos de latência dos fios e elementos lógicos foram calculados utilizando a ferramenta SPICE, onde para o fio, foi utilizado como comprimento entre os roteadores o tamanho de um ARM1176 (ARM, 2010), processador amplamente utilizado em sistemas embarcados, conforme ilustrado na Figura 3.10. É importante mencionar que o cálculo de latência no chaveamento direto (UCS) é baseado na implementação realizada nesse trabalho, que possui para cada roteador a latência agregada de três multiplexadores e uma porta AND.

Para analisarmos a influência dos diferentes modos de chaveamento foram elaborados dois gráficos a partir das equações, variando o número de HOPS e obtendo as diferentes latências. Esses resultados são demonstrados na Figura 3.9. Como esperado, as propostas com menor latência por roteador apresentam menor latência no geral, como apresentado na Figura 3.9.a. No entanto, se verificarmos a vazão (taxa de dados), na Figura 3.9.b. o cenário muda, pois nas propostas que apresentam armazenamento dos dados trafegados, como eles são "*pipelinezados*", a taxa de envio dos dados se mantém constante, ao contrário da proposta que não possui nenhuma lógica de armazenamento, pois nesse caso, à medida que o número de *hops* aumenta na rede, a vazão diminui. Portanto, a definição para um ou outro método é decidida em função do número de *hops* que uma mensagem precisa trafegar na rede, ou seja, da distância entre os roteadores da rede que precisam se comunicar, e da taxa de comunicação requerida entre os nodos.



(a) (b)

Figura 3.9 - Resultados de (a) Latência (b) e Vazão considerando o número de caminhos entre roteadores (*hops*).

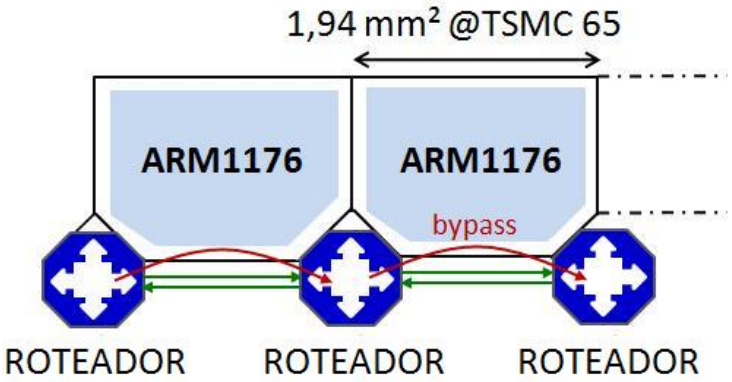


Figura 3.10 – Tamanho dos fios entre roteadores considerados para os experimentos de fios longos.

Com os experimentos anteriores, conclui-se que quando se deseja transmitir uma mensagem rápida e curta, como por exemplo, uma mensagem de interrupção ou alarme, o modo de passagem sem memorização (UCS) é o ideal, pois é o que apresenta menor latência. No entanto, quando o objetivo é transmitir mensagens do tipo *stream* de dados (elevadas taxas), o modo BCS é o mais indicado, pois é o que apresenta maior vazão. E quando há necessidade de compartilhar muitos caminhos entre as mensagens, a estratégia por pacotes acaba prevalecendo. Esse estudo demonstra que um controle que seleciona qual modo de operação deve ser utilizado em um dado momento precisa levar em consideração a natureza da mensagem, bem como o estado do sistema e os requisitos da aplicação.

Portanto, para se atingir os melhores índices de desempenho, a melhor estratégia é um arranjo dessas soluções apresentadas empregadas no momento certo. Quando existir uma alta taxa de comunicação a uma pequena distância deve-se empregar UCS. Se existir uma distância maior, onde o comprimento do fio gerado pelo chaveamento por circuito implica em perda de desempenho, pode-se utilizar a solução BCS. E consequentemente, quando existir muitos compartilhamentos dos caminhos na rede pelas mensagens das aplicações, a estratégia de fechar circuito acaba ficando limitada, sendo necessário o emprego de PS. Porém, uma arquitetura de controle que realize todas essas considerações de forma efetiva possui uma alta complexidade, sendo uma solução com maiores custos de área e potência. Para evitar este custo agregado, o mecanismo de controle desenvolvido nesse trabalho foi simplificado, e por isso não garante a melhor solução sempre. Na próxima subseção é apresentada a arquitetura proposta que possui essas estratégias como base para controlar dinamicamente o sistema através de um controle distribuído.

3.3 Arquitetura Proposta – MINoC

A arquitetura MINoC (*Multiple Interconnections Networks-on-Chip*) foi desenvolvida para lidar com qualquer tipo de comportamento de comunicação, procurando fornecer alto desempenho para diversos perfis de aplicação. Isso é feito utilizando duas ideias bem difundidas na literatura, caminhos dedicados através de

chaveamento por circuito e reconfiguração topológica através de mudanças nas conexões entre nodos. Dessa forma, essa proposta permite que o chaveamento seja alterado dinamicamente através de chaveamento por circuito alterando a maneira que os nodos estão conectados para cada nova mensagem na rede, resultando em diferentes topologias durante a execução de uma determinada aplicação. O roteador dessa arquitetura, chamado de RAPeC (*Router Architecture Packet & Circuit*), configura-se automaticamente através de uma unidade de controle específica. Além disso, a rede MINoC leva em consideração as limitações relativas aos fios longos gerados e observações estratégicas na decisão de chaveamento, (conforme abordado nas subseções anteriores) para garantir o correto funcionamento na frequência de operação do sistema. E, esse processo é feito transparente à aplicação com o custo de apenas dois ciclos por *hop* para sua configuração.

3.3.1 Comportamento Funcional

O comportamento funcional de todo o sistema é objetivo e metódico, principalmente para permitir um controle com pouco custo de hardware e rápido tempo de resposta. Esse controle é distribuído, onde cada roteador toma sua própria decisão se deve utilizar a técnica de passagem direta (*bypass*). Para a tomada de decisão existem duas premissas:

- A porta de saída desejada não deve estar ocupada, ou seja, sendo utilizada por outro pacote. Por exemplo, caso uma comunicação esteja utilizando a porta de saída sul e um novo pacote de outra porta também deseja sair pelo o sul. Assim, deve esperar a primeira comunicação liberar essa saída.
- Roteador vizinho permita comportar um fluxo de dados contínuo. Por exemplo, os dados enviados não sofrem paradas no fluxo de dados durante a transmissão.

Com essas premissas confirmadas é possível definir que um determinado pacote poderá trafegar por um caminho dedicado, criando um *bypass* local. Sendo assim, múltiplos *bypass* locais executando em paralelo na rede acarretam em uma mudança das conexões ao mesmo tempo, alterando logicamente a topologia. Além disso, visto que essa configuração é realizada a cada troca de mensagem durante a execução da aplicação, é possível ter bom desempenho mesmo que ela apresente comportamentos inesperados de comunicação, como por exemplo, interrupções.

Na figura 3.11, são demonstradas três situações que exemplificam como a rede se configura e cria uma topologia diferente. Primeiramente, para fins didáticos, apresenta-se um caso com situações sem disputa pelos mesmos caminhos, ocorrendo chaveamento por circuito em todos os casos.

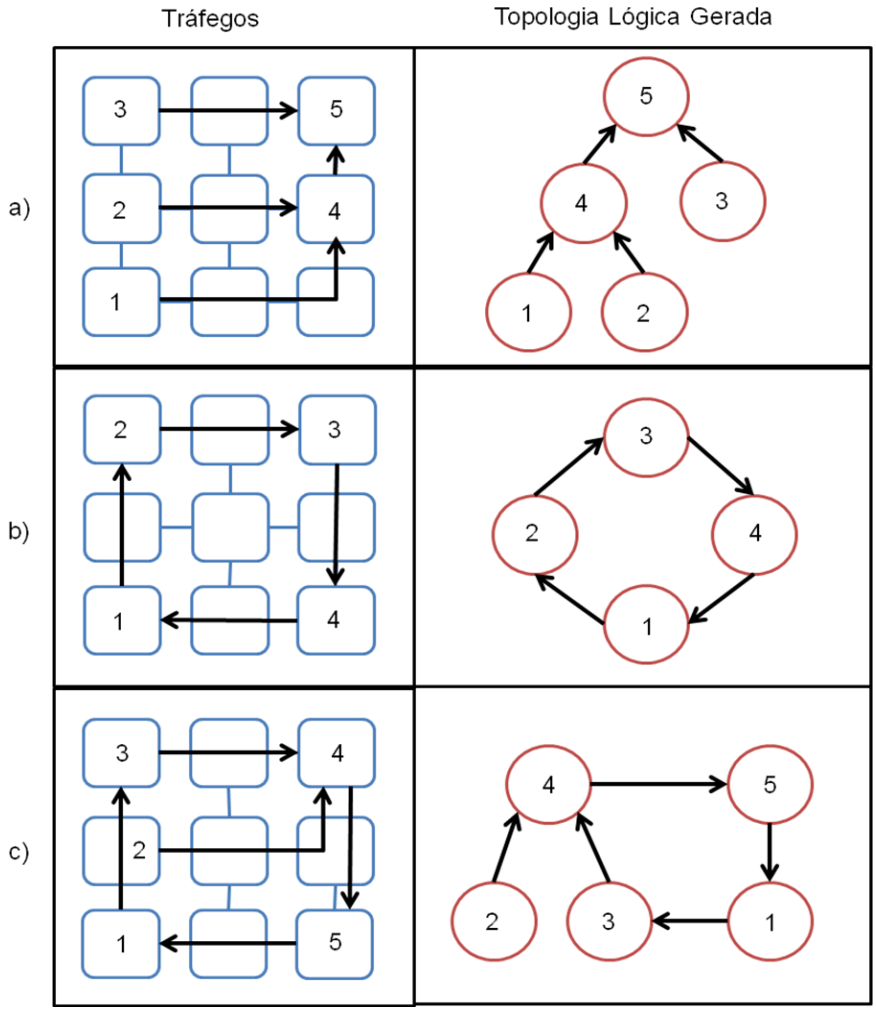


Figura 3.11 – Exemplificação do mecanismo de reconfiguração topológica. (a) Tráfego configura uma estrutura topológica de árvore. (b) Tráfego configura uma estrutura topológica de anel. (c) Tráfego configura uma estrutura mista entre árvore e anel.

Observa-se na Figura 3.11, que comunicações sem conflito de recursos permitem ajustar uma topologia à aplicação. No entanto, existe a possibilidade de ocorrer disputa por uma porta de saída em algum roteador. Nesse caso, o primeiro pacote a solicitar o caminho fecha o circuito enquanto que o segundo entra no modo de chaveamento por pacotes e aguarda o fim da primeira comunicação. Após esse tempo de espera, a partir desse ponto pode ser criado um novo caminho dedicado (circuito), caso exista a disponibilidade nos roteadores que seguem. É importante salientar que durante esse tempo de espera, o chaveamento por circuito que ocorre nos roteadores anteriores não é desativado. A desativação apenas ocorre quando a comunicação em *stream* não consegue ser mantida, ou seja, quando o *buffer* fica cheio. Nessa ocasião, um sinal de sincronismo informa ao roteador antecessor que deve utilizar o modo por pacotes PS, e esse mecanismo se propaga aos poucos até chegar ao nodo origem. O esperado é que não ocorra uma desativação completa de um circuito, pois é necessário que lote todos os *buffers* de todos os roteadores que antecedem uma contingência. Todos esses comportamentos relativos a definições dos modos de chaveamento em um cenário com contingência na rede são demonstrados na figura 3.12.

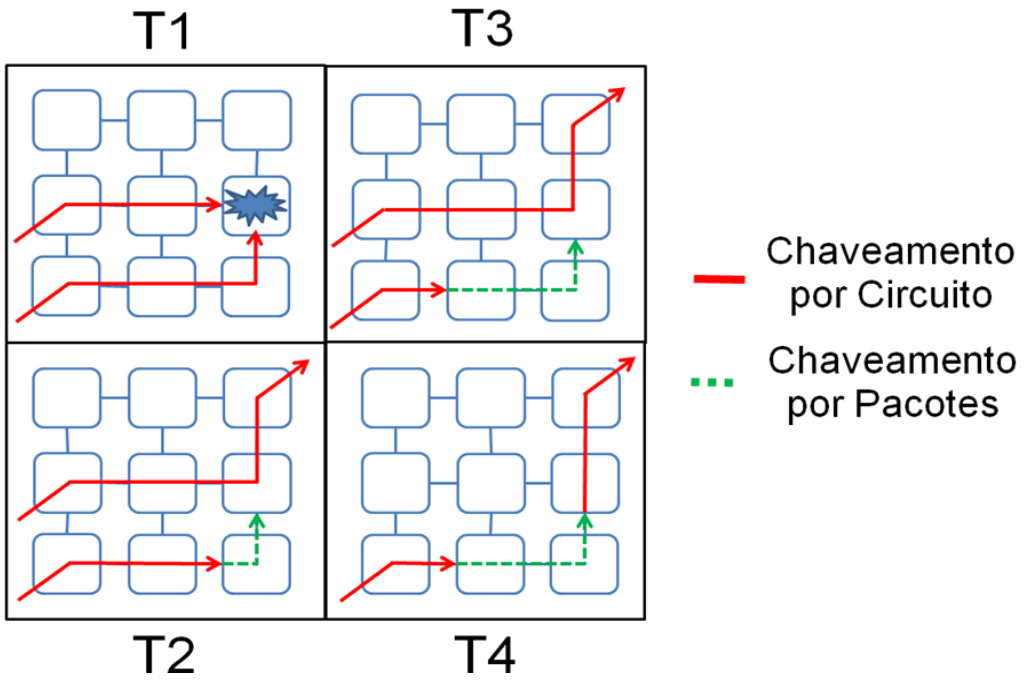


Figura 3.12 - Comportamento de Reconfigurabilidade de Chaveamento (vermelho refere-se a transmissão por chaveamento por circuito e verde refere-se a transmissão por chaveamento de pacotes)

A figura 3.12 apresenta o comportamento da reconfigurabilidade de chaveamento presente na arquitetura MINoC. Nessa imagem, dividiu-se em quatro tempos de atuação, sendo o primeiro a ocorrência do conflito. Após, no tempo dois, há a definição de quem permanece chaveado por circuito e quem passa a operar por pacotes. O terceiro mostra a propagação do modo PS por motivos de lotação de *buffer*. Finalmente, no tempo quatro, ocorre a liberação do caminho, onde se configura um novo circuito, a partir do ponto de contenção.

Para entender melhor a funcionalidade dos modos de chaveamento presentes no roteador RAPeC são apresentadas na figura 3.13 as três situações de chaveamento possíveis. Esse roteador utiliza uma lógica especulativa, onde sempre tenta utilizar chaveamento por circuito como primeira situação, escolhendo entre UCS ou BCS, dependendo da identificação de fio longo. Quando não é possível essa configuração, é utilizado o chaveamento por pacotes (PS), que permite esperar a liberação da saída através da utilização dos *buffers* de entrada para armazenar as partes do pacote. A explicação aprofundada sobre os métodos e métricas de controle de chaveamento é apresentada na subseção 3.3.3.3 relativa à arquitetura do Controlador do Modo Operacional.

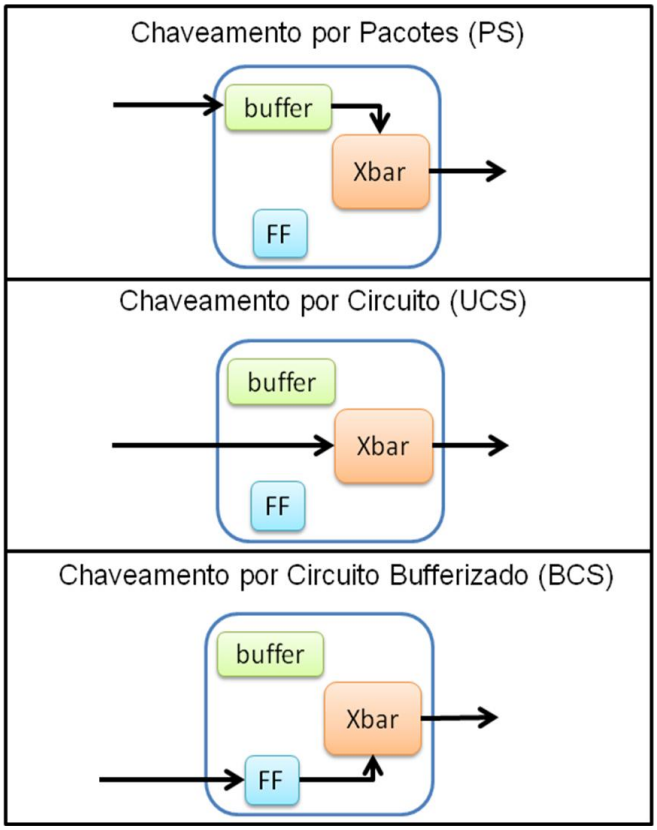


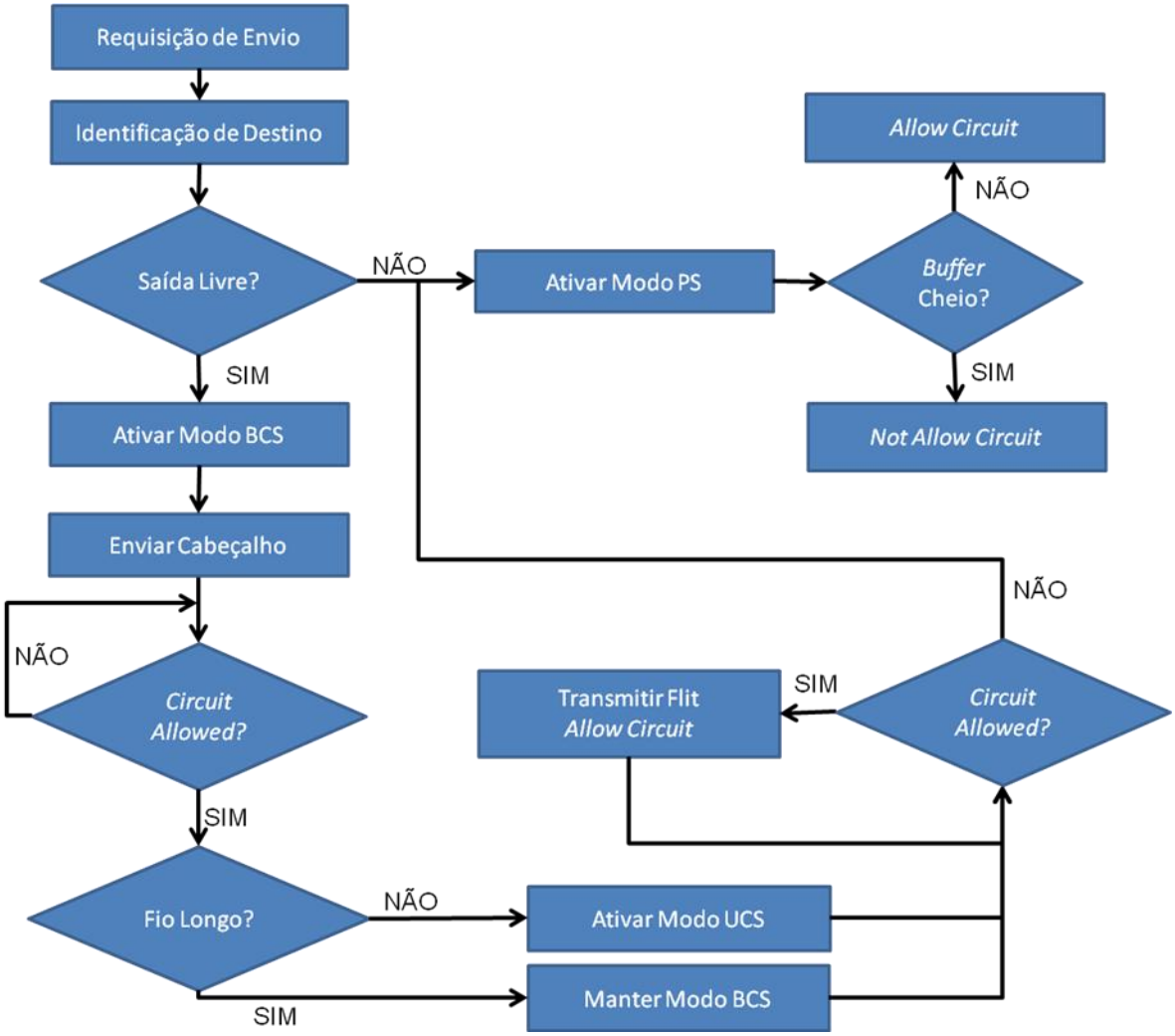
Figura 3.13 – Modos de chaveamento existentes no roteador RAPeC. A sigla FF remete a *Flip-flop* enquanto Xbar a *Crossbar*.

3.3.2 Protocolo

O protocolo existente na arquitetura da rede MINoC segue um fluxo diferente das redes encontradas no estado da arte, principalmente pelo fato que a arquitetura decide automaticamente o método de chaveamento. Para isso, a cada inicio de comunicação, os roteadores sempre especulam que usarão chaveamento por circuito, assumindo essa forma até que algum sinal confirme a impossibilidade disso. Caso isso ocorra, o roteador utiliza o chaveamento por pacotes e aproveita um dado salvo no buffer para manter a coerência dos dados.

Os modos de chaveamento por circuito apresentam a característica de tornar um determinado caminho exclusivo àquela comunicação, o que pode causar problemas de contenção de outras transmissões caso muitos dados sejam transmitidos. Para evitar essa questão, os dados devem trafegar de acordo com a mesma organização por pacotes realizados no chaveamento por pacotes, com limitações de tamanho máximo possível por mensagem.

Em relação ao protocolo do sistema de configuração da arquitetura MINoC, é necessário realizar uma série de passos, descritas no fluxograma da Figura 3.14.



3.14 - Fluxograma do Sistema de Configuração da Arquitetura MINoC

Esse fluxograma apresenta uma série de passos e verificações que devem ser realizadas durante o processo. O protocolo desse sistema inicia com algum nodo na rede querendo comunicar uma mensagem, e isso é realizado através de um sinal de controle de fluxo definido como *valid*. Após essa requisição, o roteador leva um ciclo de relógio para identificar o destino desse pacote e para verificar se esse destino está livre (*crossbar* livre). Caso o destino não esteja livre, o modo de chaveamento por pacotes (modo PS) é ativado e os dados passam a ser armazenados no *buffer*, e a informação de que o circuito está liberado (*allow circuit*) é repassado para o elemento antecessor que interpreta esse sinal como *circuit allowed*. Por outro lado, se estiver livre, esse primeiro flit, representando o cabeçalho do pacote, é enviado ao próximo roteador e o modo de transmissão definido nessa etapa de configuração é o chaveamento por circuito com memorização (modo BCS). Esse processo é realizado até o sinal *circuit allowed* ser ativado pelo próximo roteador, ou seja, o destino do pacote em questão. O *circuit allowed* é ativado apenas quando o modo PS está ativado e o *buffer* se encontra cheio ou quando o pacote encontrou seu destino final, pois os núcleos sempre enviam esse sinal ativado. A partir dessa autorização, o roteador verifica uma informação presente no cabeçalho (referente ao tamanho do fio) para definir se mantém seu modo em BCS ou se deve passar para o modo sem memorização (modo UCS). Assim, é ativado esse

circuito (*circuit allow*) repassando essa informação aos elementos anteriores, que a interpretam como *circuit allowed* ativado. Por fim, a comunicação é iniciada e os dados trafegam por *stream*, tendo como condição de parada uma possível desativação do sinal *circuit allowed.* Nesse caso, o modo PS é ativado e o *buffer* passa a ser utilizado. Caso o *buffer* fique cheio o sinal de circuito deve ser desativado (*Not Allow Circuit*) para os nodos que o antecedem.

3.3.3 Arquitetura

Em relação à arquitetura MINoC é importante salientar que tem como base o roteador RaSoC da arquitetura SoCIN, que possui cinco canais de entrada e cinco canais de saídas independentes. Essa estrutura foi alterada para comportar chaveamento por circuito, tanto em sua forma direta como bufferizada (UCS e BCS). Além disso, foi adicionado a cada porta de entrada de cada roteador um controle que realiza as decisões de qual chaveamento deve ser utilizado para determinada mensagem (Controlador do Modo de Operação). O diagrama de blocos exemplificando a arquitetura desse novo roteador, chamado RAPeC, pode ser observado na Figura 3.15. Essa imagem está apresentando apenas uma porta, sendo necessário replicar por cinco (número total de portas) esse esquema para representar o roteador completo.

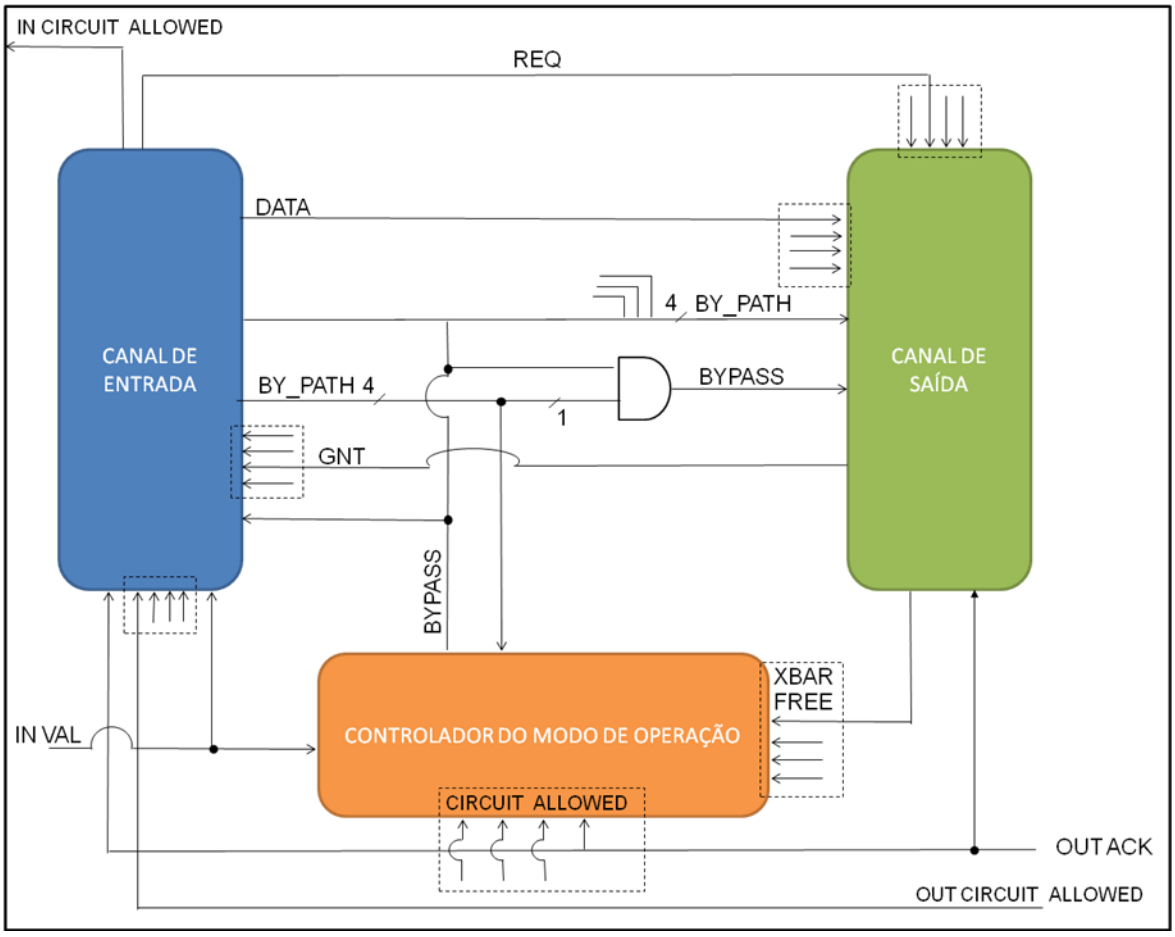


Figura 3.15 – Diagramas de blocos do roteador RAPeC

Nas próximas subseções será abordado cada um desses blocos: Canal de Entrada, Canal de Saída e Controlador do Modo de Operação.

3.3.3.1 Canal de Entrada

A arquitetura do canal de entrada difere do convencional no sistema de *buffer* e controle. A primeira mudança é justificada pela necessidade de obter rapidamente um dado do *buffer* quando ocorrer mudanças dinâmicas de chaveamento. A segunda mudança se refere em identificar no cabeçalho a estrutura indicadora do tamanho do fio do circuito fechado até então. Com essa informação é possível decidir se é necessário realizar um circuito com memorização (BCS) quando em modo de chaveamento por circuito. Outras pequenas mudanças se referem aos multiplexadores que foram adicionados para permitirem modificar o controle de fluxo dos dados, como por exemplo, quando for definido o modo UCS em que o dado precisa evitar o armazenamento (dcon). Além disso, alguns outros controles foram inseridos a fim de propagar para os outros roteadores o habilitador de circuito (*circuit_allowed*). Os elementos alterados para essa arquitetura são explicitados na Figura 3.16.

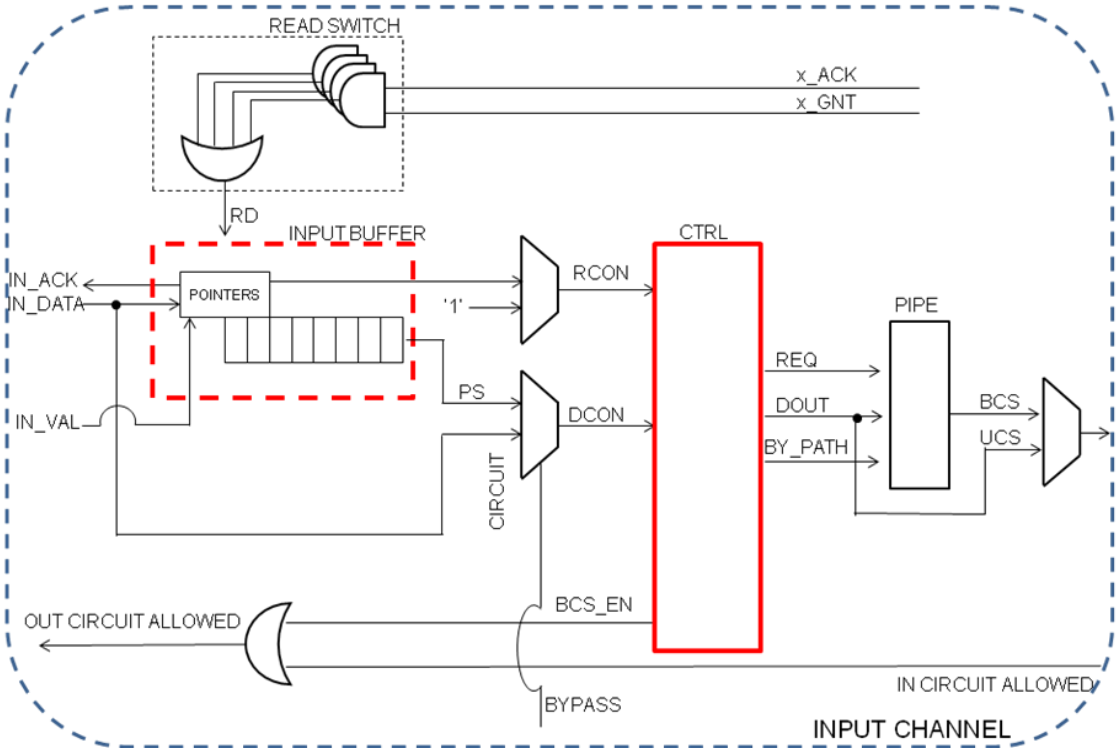


Figura 3.16 – Arquitetura do Canal de Entrada

O sistema de buffer deve utilizar uma lógica onde sempre é escrito o dado na mesma posição, assim sempre podemos acessar o último dado salvo que entrou no roteador. Essa informação é necessária caso o circuito tenha que ser desligado, necessitando obter o último dado salvo para não perder informação ao habilitar o modo por pacotes. Esse

novo *buffer* implementado sempre escreve na primeira posição, ou seja, o ponteiro de escrita fica fixo enquanto que os dados sofrem operações de deslocamento na memória. O ponteiro de leitura atua normalmente. Além disso, o *buffer* possui um desabilitador de enfileiramento, que impede que ocorram operações de deslocamento dos dados caso o roteador esteja em chaveamento por circuito. Isso impede que seja gravado um dado inválido no sistema de buffer. Abaixo, a Figura 3.17, exemplifica esse funcionamento.

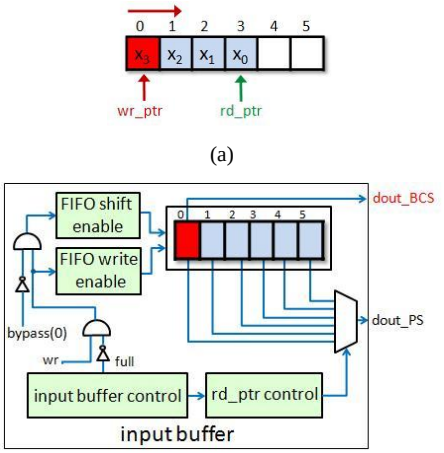


Figura 3.17 – (a) Representação da FIFO e (b) arquitetura da FIFO e controle utilizada na MINoC, chamada de *input buffer*.

O sistema de controle do canal de entrada, além de suas funções básicas, que é definir a porta de saída para um determinado pacote, possui um algoritmo para tomar a decisão de habilitar ou não a memorização dos dados trafegando pelo circuito fechado (modo BCS). Esse controle verifica o tamanho do fio do circuito fechado, o que possibilita verificar quando os *flits* precisam ser armazenados. O valor acumulado do tamanho de um fio em um determinado circuito fechado está armazenado em um espaço reservado do cabeçalho de cada pacote, conforme apresentado na figura 3.18. Assim, somando essa informação com a informação de comprimento do presente roteador até o próximo destino (valor conhecido pelo controlador) é verificado se é necessário ativar o mecanismo de memorização para esse pacote. Se não necessitar o emprego do modo BCS, apenas é atualizado o valor do espaço reservado do cabeçalho com o resultado dessa soma de comprimentos.

Para definir qual método de chaveamento por circuito utilizar, a MINoC considera a informação real dos tamanhos dos fios, permitindo criar circuitos longos sem prejudicar a frequência do sistema. A informação real do tamanho dos fios entre cada roteador é extraída do *floorplanning* do projeto. Os valores retirados do *floorplan*, relativos ao tamanho das interconexões, são convertidos manualmente em valores ponderados, e são passados como parâmetros para cada roteador. Assim, os controles utilizam essa informação para colocarem barreiras temporais quando necessário, evitando um circuito combinacional com atraso maior que o período do relógio.

A informação presente no controlador sobre o custo do tamanho do fio do roteador vizinho até ele é definido em tempo de projeto durante síntese física e inserido em forma de parâmetro fixo no hardware. O valor não se refere ao tamanho exato do fio, mas sim uma normalização matemática para simplificar o hardware. Por exemplo, se for

definido que serão utilizados dois bits para esse controle, verifica-se o tamanho máximo permitido de fio para não obter violação de temporização de projeto, e esse será o valor máximo, ou seja, relativo ao número três (em binário 11). Os tamanhos de todos os fios entre roteadores extraídos da síntese física são convertidos em porcentagens desse fator máximo, e nesse caso teríamos três faixas de custo possíveis, visto que a opção de custo zero não pode ser utilizada. Sendo assim, valores até 33% do máximo tem custo 1, de 33% a 66% custo 2 e acima disso custo 3. Como podem ser observados, poucos bits para essa representação pode prejudicar o funcionamento do sistema, no entanto reduzem a quantidade de informações que precisam ser transmitidas no cabeçalho das mensagens.

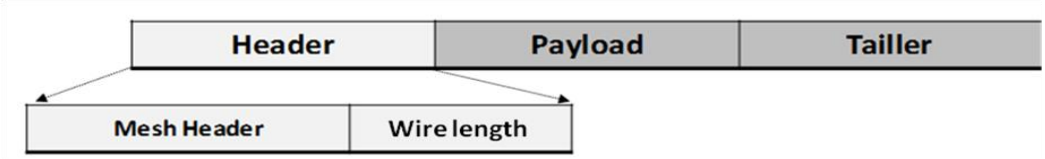


Figura 3.18 – Pacote MINoC – Cabeçalho com informação do tamanho do fio

3.3.3.2 Canal de Saída

A arquitetura do canal de saída não sofreu muitas modificações. Apenas foram colocados multiplexadores para modificar o fluxo de dados e também para desabilitar o protocolo do *crossbar* durante os modos UCS e BCS. No caso de chaveamento por circuito, os multiplexadores colocam os dados diretamente na porta de saída.

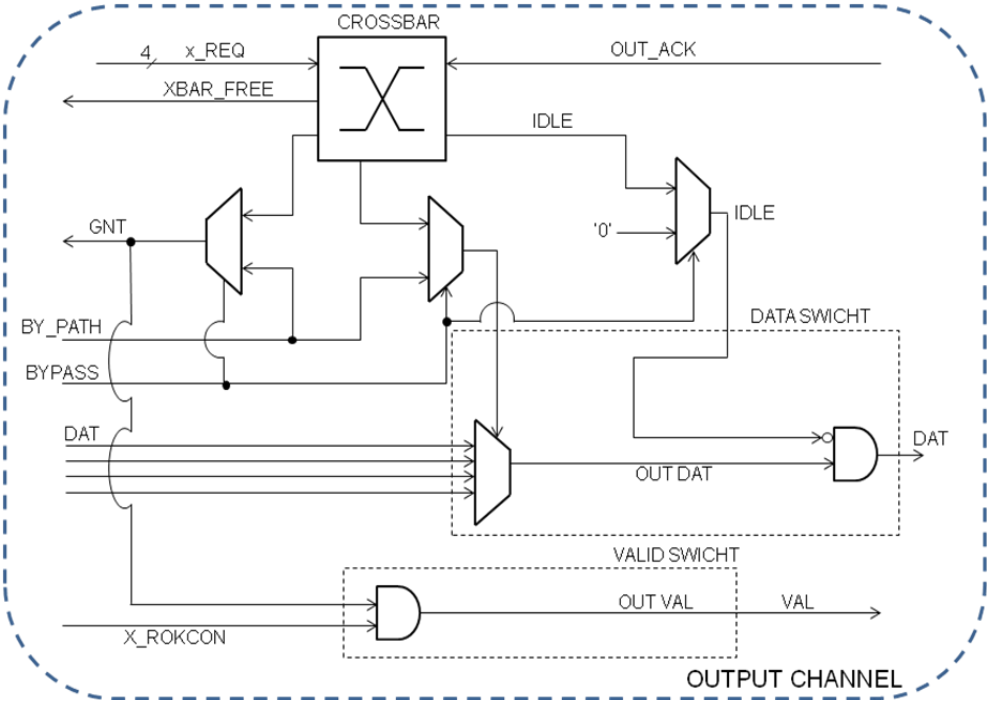


Figura 3.19 – Canal de Saída

Como pode ser observado na figura 3.19, temos dois sinais importantes: *bypass* e *by_path*. O *bypass* é utilizado para isolar os sinais de controle do crossbar, desabilitando ele para o resto do roteador durante os modos UCS e BCS. Além disso, essse sinal ativa o protocolo de handshake de saída, ativando o sinal *out_val* e selecionando a saída de dados para o fluxo de circuito. O sinal *by_path* apenas define qual das entradas que possui o fluxo de dados por circuito.

Além disso, é importante mencionar que os sinais do *crossbar* para o roteador são desabilitados, mas o inverso não. Isso permite que o crossbar identifique uma requisição de outra porta, e já fica preparado para atender esse chamado assim que a mensagem por circuito for concluída. Esse mecanismo foi elaborado para evitar o problema de *starvation* que poderia ser causado pela alta prioridade do sistema por circuito. Dessa forma, evita-se o problema de atender apenas as requisições por circuito da mesma porta de entrada, visto que o crossbar trabalhando em paralelo permite atender futuramente outras requisições.

3.3.3.3 Controlador do Modo de Operação

O Controlador do Modo de Operação é responsável por definir as trocas dinâmicas de chaveamento em cada porta de entrada de cada roteador RAPeC. Ele controla o principal sinal de definição de chaveamento, chamado *bypass*, que utiliza os seguintes parâmetros para sua definição:

- Se existe dados para transmitir (*in_val* ativo na porta de entrada);

- Se o crossbar da saída de destino se encontra livre no inicio da comunicação (*xbar_free* ativo);
- Se o roteador de destino pode receber os dados em modo BCS ou UCS (*circuit_allowed* ativo);

É importante mencionar que esse controlador utiliza uma lógica de especulação para todo inicio de comunicação, assumindo que todos os pacotes no início podem ser chaveados por circuito. Caso a especulação cometa um erro e não possa ser assumido o modo BCS ou UCS ou até mesmo se esses modos tiverem de ser desligados devido a um congestionamento na rede, o elemento de *buffer* da porta de entrada possui um mecanismo de armazenamento de emergência, assim, recuperando o dado, caso contrário, o mesmo teria sido perdido.

Há também nesse bloco um sinal auxiliar, *bypath*, para informar qual a porta de saída que porta de entrada pretende se comunicar. Assim, podem-se selecionar os sinais de controle das portas de saída certas, como *xbar_free* e *circuit_allowed*. Portanto, se as condições citadas acima são satisfeitas, os modos de chaveamento por circuito são habilitados, apesar de existirem algumas outras considerações importantes que serão mencionadas a seguir.

A primeira consideração é relacionada ao fato do roteador precisar de um ciclo para definir qual é a porta de saída para uma determinada mensagem, gerando um atraso na decisão do controlador, o que justifica a necessidade de uma decisão especulativa. Os elementos responsáveis pela especulação são os definidos como Sync&Hold do circuit_allowed e o Sync do xbar_free (demonstrados na figura 3.20). O primeiro mantém o circuit_allowed ativo no primeiro ciclo. O segundo objetiva passar apenas o último dado válido relativo ao sinal xbar_free, congelando essa informação após fechar o circuito.

Outra consideração se refere a desabilitação dos modos de chaveamento por circuito. Essa informação deve ter um atraso para a porta de saída, para não ocorrer perda de informação. Isso é resolvido com o bloco Sync&Hold para o sinal in_val.

Também é importante mencionar que cada roteador é independente, sendo possível obter diferentes estratégias de chaveamento na mesma mensagem. Um pacote vai ser puramente chaveado por pacote apenas se a contenção da rede se propagar até o roteador inicial.

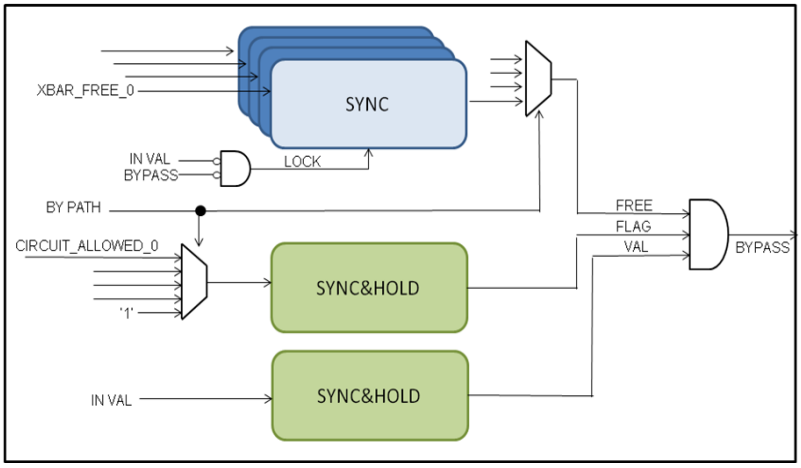


Figura 3.20 – Arquitetura do *Operation Mode Controller*

3.4 Considerações

Nessa seção foram apresentados os conceitos de reconfiguração, bem como algumas propostas encontradas na literatura que utilizam essa estratégia para prover desempenho. Além disso, a literatura apresentou quais seriam os maiores benefícios e fraquezas das estratégias mais empregadas, como fios longos e chaveamento por circuito. Essas características foram estudadas e as conclusões das análises permitiram a proposta de uma nova arquitetura reconfigurável. A proposta MINoC permite um alto grau de reconfiguração buscando o emprego de múltiplas conexões dedicadas para alterar a configuração lógica da topologia dinamicamente para cada mensagem.

4 REDE HIERÁRQUICA

O conceito de que o desempenho e a dissipação de potência de um sistema são totalmente dependentes da topologia de rede definida e como os núcleos estão interligados (DAS, 2009) motivou o emprego da estratégia de projetar redes hierárquicas. Esse método consiste em utilizar uma estrutura de comunicação com diferentes elementos arquiteturais organizados em uma hierarquia. Seu fundamento se baseia no fato de que existe uma localidade nas comunicações, ou seja, os elementos têm afinidade de comunicação predeterminada, e isso permite arranjá-los em grupos. Essa área de estudo tem provado não apenas que reduz o número de caminhos percorridos na rede pelos pacotes quando comparadas com as redes convencionais, mas também provê uma largura de banda adequada, baixa potência e qualidade de serviço na comunicação (CHOU, 2010) (BOURDUAS, 2007). Ao modificar as disposições dos elementos utilizando-se hierarquias, almeja-se obter o mínimo de custo de hardware aliado a um máximo desempenho, mas para isso é necessário que o projeto esteja alinhado com o comportamento de comunicação da aplicação alvo. Portanto, essa estratégia pode obter ganhos tanto em custos quanto em desempenho sob o preço de ser uma solução especifica para cada aplicação.

4.1 Trabalhos Relacionados

Existe uma grande variedade de redes hierárquicas propostas pela literatura utilizando diferentes combinações de estruturas de interconexão. A maioria das contribuições utiliza uma rede-em-chip no nível global da hierarquia, diferenciando-se apenas pelo elemento nos níveis locais, que pode ser desde barramentos a redes-em-chip em diferentes topologias, sendo que a mais utilizada ainda é a topologia malha.

A primeira proposta analisada é chamada de GigaNoC e foi proposta por (PUTTMANN, 2007). Nesta solução, a hierarquia foi dividida em três níveis, chamados de *PE level*, *Cluster level* e *SoC level* (figura 4.1). No nível mais inferior (*PE level*), tem-se um sistema com processador, composto pelo processador N-Core e uma memória cache. O próximo nível (*Cluster level*) apresenta um barramento *Wishbone* para interligar quatro desses sistemas do nível *PE*. Por fim, no nível *SoC level* tem-se roteadores chamados de *Switch Box* para interligar todos os clusters do nível inferior.

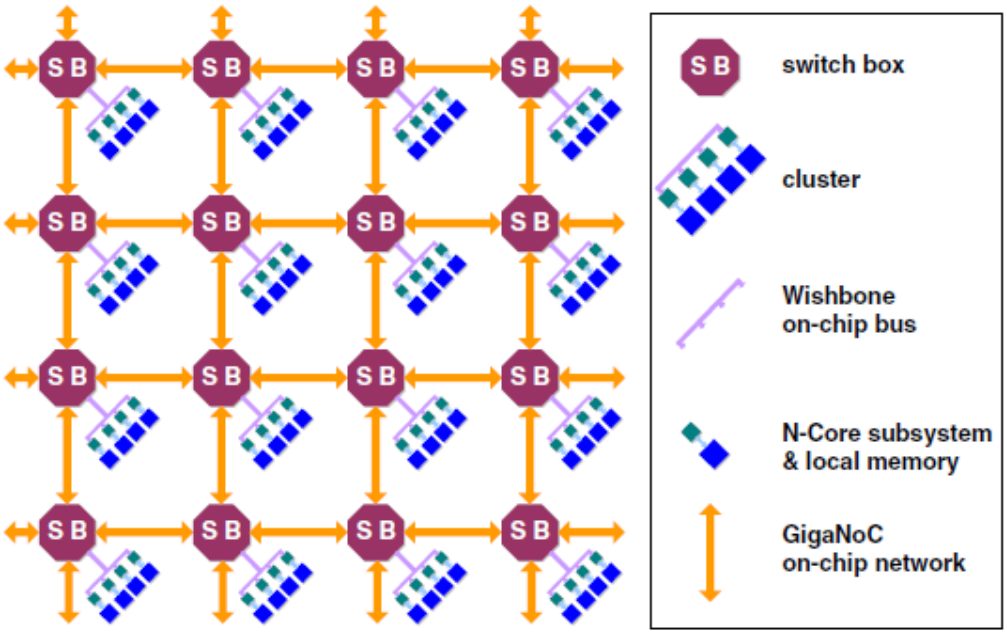


Figura 4.1 – Arquitetura GigaNoC

A rede-em-chip presente na GigaNoC utiliza chaveamento por pacotes do tipo *wormhole*, apresentando em seus protocolos parâmetros para a correta transmissão pela rede. Esses parâmetros são encontrados nos pacotes, onde o cabeçalho possui o identificador do destino dentro de um cluster e o tipo de mensagem carregada pelo pacote. Os tipos de mensagens compreendem quatro formatos: dados, comandos, instruções e programas. Sua utilização serve para configurar a rede ou para informar ao destino algumas informações de transmissão. Exemplificando, toda mensagem a ser enviada deve, primeiramente, enviar um *flit* de comando para informar quantos *flits* de dados serão enviados para o destino. Em seguida, os *flits* de dados são encaminhados. O tipo de mensagem por instrução pode ser utilizado para alterar as configurações da rede-em-chip, como algoritmo de roteamento. Logo, o tipo de mensagem no formato de programa é utilizado para inicializar as memórias dos sistemas processados (PE_level), enviando os dados diretamente para as memórias locais dos processadores. Os autores não apresentam nenhuma comparação para identificar os reais ganhos e perdas da proposta, apresentando apenas que, para uma tecnologia *standard cell* 90 nanômetros, a área do sistema resulta em 5,3 milímetros quadrados e a potência por volta de 200 miliwatts. Assim, baseado nesses resultados e nas características usuais de redes hierárquicas, conclui-se que o maior ganho dessa proposta se encontra na redução de área e potência. Porém, a GigaNoC apresenta grande complexidade para configurar seu sistema devido ao seu protocolo, o que pode acarretar em altos custos de latência, devido ao seu tempo de *setup*, ou pode dificultar a obtenção do máximo desempenho dessa arquitetura.

Outra abordagem que utiliza barramentos aliados a redes utilizando topologia malha é encontrada em (DAS, 2009). Nesse trabalho são comparadas três possibilidades topológicas, uma rede em malha, uma rede *butterfly* e uma rede hierárquica com barramentos. As redes malha e *butterfly* possuem roteadores que se conectam a quatro núcleos por vez, e, da mesma maneira, os barramentos da rede hierárquica, conectada a

cada roteador, também possuem quatro elementos de processamento acoplados, como pode ser observado na figura 4.2.

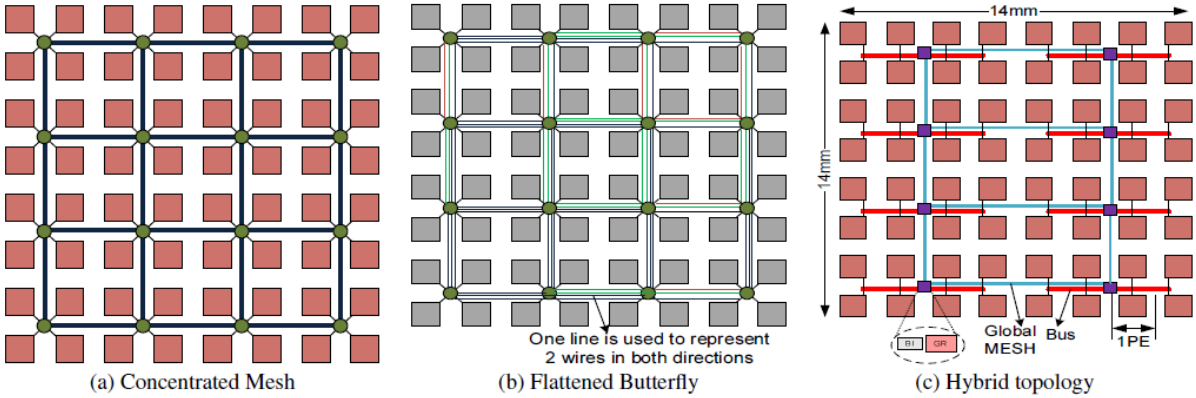


Figura 4.2 – Topologias comparadas. (a) Malha. (b) Butterfly (c) Hierárquica – Malha e barramento

Como resultado, foram realizadas diferentes análises de latência, vazão, potência e energia. Como métrica final para definir a melhor proposta, foi utilizada a relação de desempenho por energia. Nesse caso, a solução hierárquica foi a que apresentou melhor resultado em energia, obtendo um consumo 33% inferior à malha e 47% inferior à *butterfly*. Porém, esta solução possui limitações relativas à vazão de dados, e dessa forma, foi proposto no mesmo trabalho um esquema para compartilhamento de canais chamado *Xshare*. Esse compartilhamento permite que pacotes com poucos bits, como pacotes de controle, por exemplo, possam utilizar com maior eficiência os canais. Para isso, a arquitetura foi modificada de modo a dividir cada canal virtual por dois, deixando cada metade totalmente independente, ou seja, o *crossbar* permite rotear cada parte do mesmo canal virtual para portas diferentes. Portanto, pode-se ajustar a utilização do canal à quantidade de bits a ser enviada, aumentando a eficiência da arquitetura, embora penalize área devido à inserção de lógica no *crossbar* e de multiplexadores nas portas de entrada, conforme apresentado na figura 4.3. Sendo assim, foi possível atingir melhores resultados de latência e vazão com um aprimoramento de 14% e 35%, respectivamente, mas ainda permanecendo a limitação desse sistema na vazão dos dados.

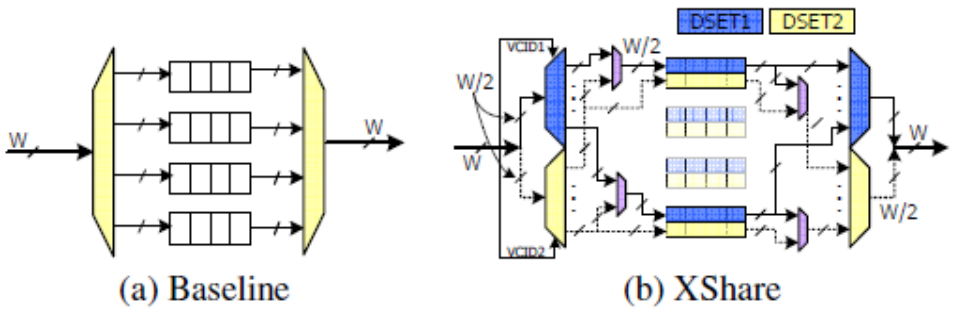


Figura 4.3 – (a) Arquitetura convencional de canal virtual (*baseline*) e (b) Xshare

Em (FREITAS, 2008), é apresentado o MCNoC, uma rede-em-chip com topologia malha no topo da hierarquia aliada a uma estrutura no segundo nível com *crossbar* reconfigurável para cada agrupamento. Essa estrutura visa não apenas utilizar o *crossbar* para comunicar elementos de processamento, mas também hierarquias de cache independentes em cada grupo, conforme pode ser observado na figura 4.4.

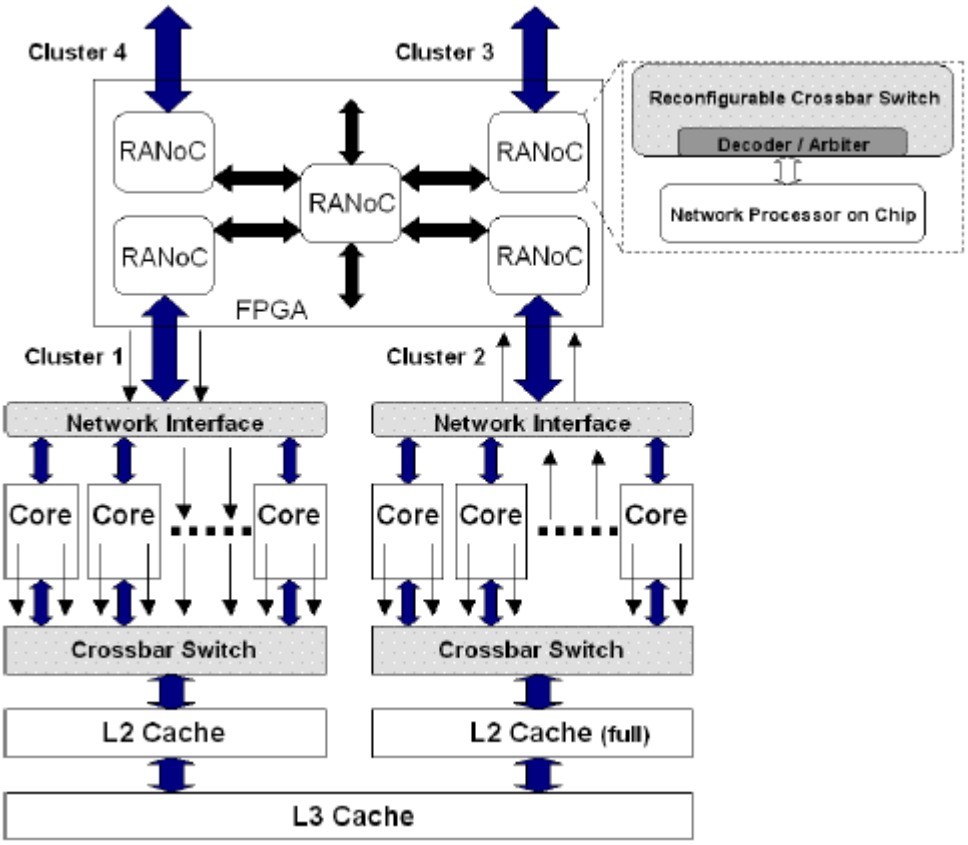


Figura 4.4 – Arquitetura MCNoC

O principal ponto da proposta está no *crossbar* reconfigurável que permite qualquer variedade de conexões simultâneas entre os nodos, permitindo uma reconfiguração topológica dentro de cada *cluster*. Exemplificando, na figura 4.5 pode-se observar que o nodo 2 está interligado ao mesmo tempo com os elementos 1 e 3. Porém, para implementar esse *crossbar* reconfigurável é necessário criar fisicamente todas as conexões possíveis entre todos os elementos presentes no cluster, o que acarreta em um custo de área e potência muito elevados. Além disso, o grande número de fios gerados acarretará problemas de roteamento para grupos com muitos elementos. Essas considerações são embasadas pelos resultados apresentados pelo autor onde a sua área é equivalente a de uma abordagem sem hierarquia. O principal ganho está na vazão dos dados, pois não ocorre qualquer contenção dentro dos agrupamentos na rede. Essa proposta buscou trazer conceitos de adaptabilidade a uma rede hierárquica, porém não considerou as penalidades de se usufruir de uma arquitetura baseada em *crossbares*. Na próxima seção é realizado um estudo sobre esse tipo de arquitetura, a fim de definir as suas reais limitações. Portanto, essa arquitetura hierárquica apresenta bons resultados de

desempenho, porém não é uma alternativa escalável para projetos futuros que requisitarão mais de centenas de núcleos.

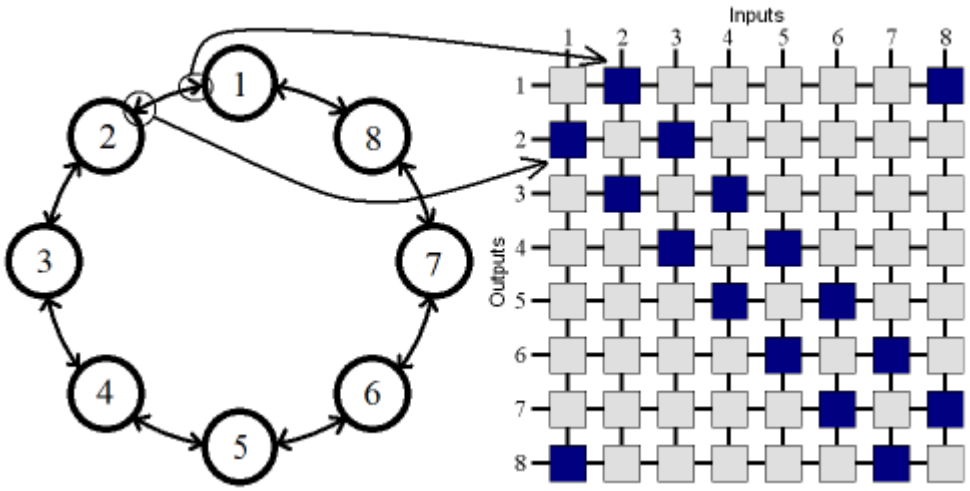


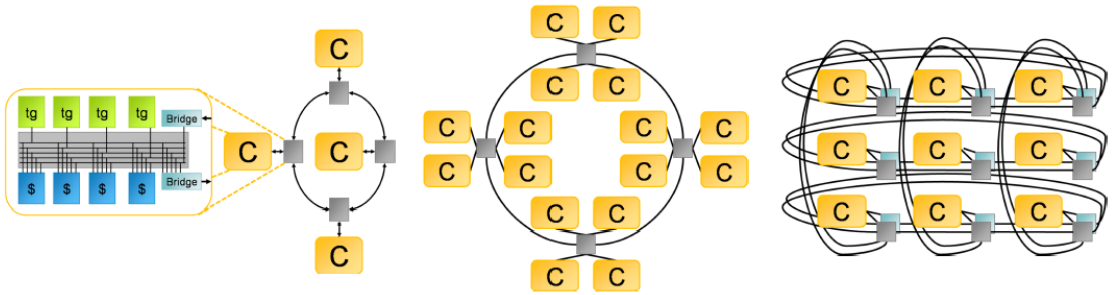
Figura 4.5 - Mapeamento de comunicação no *Crossbar* Reconfigurável

Seguindo a linha evolutiva das arquiteturas de intercomunicação, após o emprego de barramentos e chaves *crossbares*, surgem propostas que utilizam redes-em-chip nos agrupamentos dos níveis inferiores das hierarquias. Porém, essas propostas visam, principalmente, trabalhar com diferentes topologias perante as hierarquias, onde algumas propõem métodos e algoritmos para obter um maior ganho de eficiência energética.

O trabalho apresentado em (GUERRE, 2010) realiza uma comparação entre oito possibilidades de arranjos topológicos, onde três delas são redes hierárquicas. As topologias estudadas foram divididas em não hierárquicas (cinco propostas) e hierárquicas (três propostas), conforme segue:

- Não Hierárquicas:
  - Anel, Malha e Toróide: Topologias convencionais, amplamente encontradas na literatura. Utilizam o mesmo roteador, com exceção do reduzido número de portas para a abordagem em anel.
  - Multi-estágio: Nessa topologia os roteadores são organizados em estágios e cada estágio apresenta diferentes meios de conexão. Nesse trabalho foi utilizada uma abordagem *butterfly*.
  - Multi-barramento: Nessa topologia existe mais de um barramento, aumentando o paralelismo em relação à abordagem por barramento.
- Hierárquicas (figura 4.6):
  - Multi-anel: Nessa proposta, o autor apresenta no topo da hierárquica uma rede-em-chip em anel, e na hierarquia inferior, a topologia Multi-barramento.
  - Multi-cross: Nessa proposta, o autor também apresenta no topo da hierárquica uma rede-em-chip em anel, mas cada roteador possui quatro clusters de topologia Multi-barramento.

- Multi-toróide: Nessa proposta, o autor apresenta no topo da hierárquica uma rede-em-chip toróide, e na hierarquia inferior, a topologia Multi-barramento.



(a) (b) (c)

Figura 4.6 – Topologias Hierárquicas Propostas. (a) Multi-anel. (b) Multi-cross. (c) Multi-toróide

Durante as análises, a primeira constatação dos autores foi que as soluções hierárquicas apresentam maior eficiência, principalmente em área, do que as abordagens não hierárquicas. Além disso, eles observaram que a rede Multi-cross apresenta melhores resultados em relação às demais topologias estudadas. No entanto, a topologia Multi-cross não apresenta os mesmos ganhos quando a comunicação e o número de elementos aumentam, pois essa proposta utiliza a topologia em anel no topo da hierarquia, que é uma topologia que não possui escalabilidade e, dessa forma, essa torna-se um gargalo para o sistema.

Continuando com as tendências de integrar diferentes topologias de redes-em-chip de forma hierarquica, o autor (HOLLSTEIN, 2006) propôs a topologia HiNoC, onde ele divide a hierarquia em dois níveis apenas. O nível do topo é dado por uma rede-em-chip com topologia malha enquanto o nível base é dado por uma rede em topologia árvore gorda (fat-tree), onde a granularidade da árvore pode ser parametrizada de acordo com os requisitos de projeto (figura 4.7). Além disso, sua arquitetura permite que sejam realizados dois tipos de transmissão de dados. Um método é definido como transmissão por chaveamento de pacotes (no nível mais alto da hierarquia) e o outro é definido como método baseado em qualidade de serviço ou baseado em chaveamento por circuito (no nível mais baixo da hierarquia). Basicamente, seus roteadores possuem estruturas de canais virtuais que suportam alguns canais exclusivamente para um circuito fechado, tendo como configuração um pacote de *setup* na rede. O autor ainda comenta que sua topologia é aplicável à metodologia GALS (Globalmente Assíncrono Localmente Síncrono), onde o nível topo seria assíncrono e a base seria síncrona, e aplicável a testabilidade, integrando a HiNoC a um ambiente auto-depurável (Built-In Self Test – BIST).

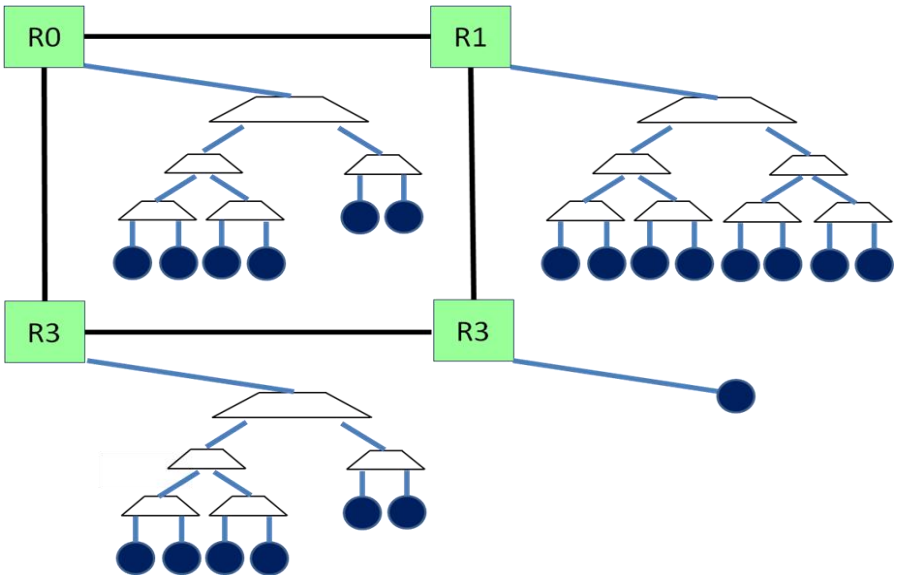


Figura 4.7 – Arquitetura HiNoC

Como conclusão dessa proposta, é apresentado que a organização hierárquica é mais eficiente para agrupamentos menores. Isso é justificado pelo custo da configuração da comunicação QoS, dada por chaveamento por circuito, que limita o desempenho desse sistema.

Em (ZHENG, 2010), a metodologia hierárquica engloba mecanismos de reconfiguração, também definida pelo autor como uma rede híbrida reconfigurável. Essa arquitetura, chamada de HCR-NoC, implementa a lógica inversa ao analisado nas propostas anteriores, pois utiliza no nível global um barramento configurável e realiza os agrupamentos com as redes-em-chip em topologia malha, conforme demonstra a figura 4.8.

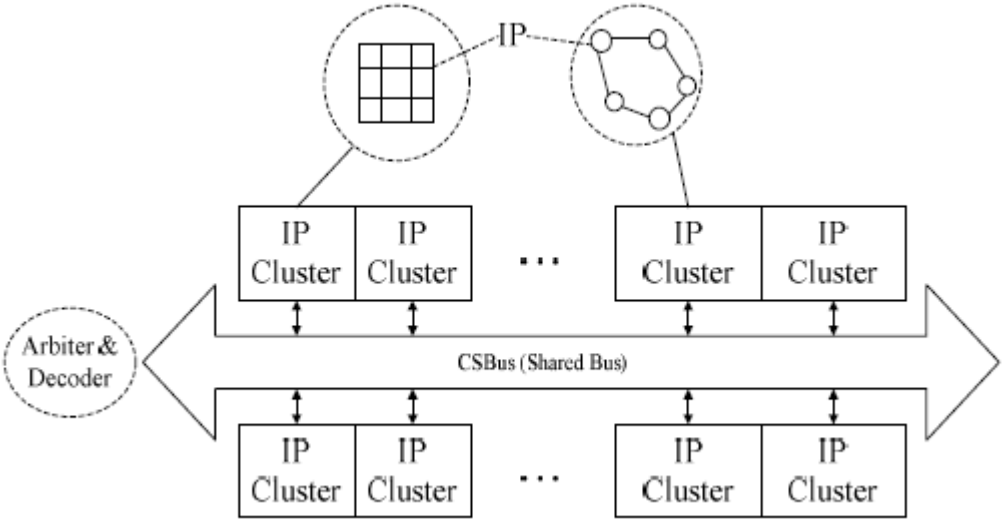
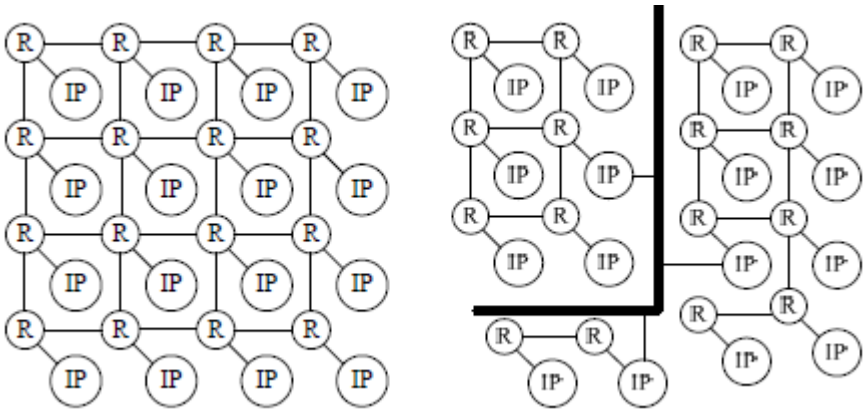


Figura 4.8 – Arquitetura HCR-NoC

Esse sistema utiliza software e hardware para monitorar e gerenciar o sistema a fim de realizar as configurações necessárias para cada aplicação. O barramento, chamado CSBus, interliga todos os clusters, configurando as conexões entre eles através da técnica TDMA (Time Division Multiple Access) que permite modificar as conexões lógicas em diferentes tempos de execução. Essa técnica divide o tempo em sub-canais, separando as comunicações por sub-canais e permitindo múltiplas comunicações ao longo do tempo.

A implementação em hardware dessa proposta utiliza roteadores com chaves ao seu redor. Essas chaves são interligadas através do barramento compartilhado CSBus, e através delas é possível conectar diferentes núcleos. O contraste da configuração física e lógica durante uma aplicação pode ser observada na figura 4.9.



(a) (b)

Figura 4.9 – Implementação física (a) e lógica (b) da topologia em HCR-NoC

Nesse trabalho, foram realizados experimentos com seis diferentes benchmarks (VPROC, MPEG4, VOPD, MWD, PIP e IMP) e os resultados apontaram uma redução energética no fator de 1,93 em relação a uma rede-em-chip chamada Opti-Mesh (WANG, 2002). A área entre as duas ficou equivalente, mas constatou-se que à medida que se aumenta o número de núcleos no sistema, a proposta HCR-NoC apresenta uma relação de custo de área menor que a Opti-Mesh. A proposta não deixa claro sobre as limitações do barramento compartilhado perante o aumento de IPs no sistema, sendo esse um fator que pode limitar o desempenho do sistema.

A seguir, pode-se observar uma tabela resumindo as principais características de cada proposta no estado da arte.

Tabela 4.1 – Resumo das propostas da literatura de redes hierárquicas

Autor	**Estratégia Hierárquica**	**Principal Benefício**	**Principal Limitação**
(PUTTMANN, 2007)	Global: NoC em Malha Local: Barramento	Área Potência	Complexidade de Configuração
(DAS, 2009)	Global: NoC em Malha Local: Barramento	Energia	Limitação de Vazão
(FREITAS, 2008)	Global: NoC em Malha Local: *Crossbar*	Vazão	Área e Potência. Roteabilidade
(GUERRE, 2010)	Global: NoC em Anel Local: Multi-barramento	Área Potência	Limitação de Escalabilidade
(HOLLSTEIN, 2006)	Global: NoC em Malha Local: NoC em *Fat-tree*	Qos Testabilidade	Tempo de *setup*
(ZHENG, 2010)	Global: Barramento Local: NoC em malha	Energia	Escalabilidade do nível global

Nessa subseção foram apresentadas diferentes propostas de arquiteturas organizadas em hierarquias. Cada conceito demonstrou ter estratégias específicas para obter algum ganho, mas sempre objetivando um sistema voltado para uma classe específica de aplicação, com exceção de (Freitas, 2008). Os principais ganhos alcançados se referem aos custos, onde foram empregadas arquiteturas de menor desempenho para equilibrar o custo final, e assim, obter menor área, potência ou energia. Uma proposta que se mostrou interessante foi a utilização de *crossbares* nos níveis locais, pois para granularidades pequenas permite obter desempenho e custos de área e potência menores que roteadores. Além disso, embora exista uma limitação referente a escalabilidade, em que seu custo aumenta exponencialmente a cada novo nodo, é possível encontrar um fator limitante que permita sua utilização. Esse estudo, sobre a análise de custo da utilização de chaves *crossbares*, é realizado na subseção 4.2.

Além disso, pôde-se observar que a reconfiguração começa a estar presente nessa área de pesquisa, permitindo ajustar melhor a arquitetura às necessidades da aplicação em tempo real ou a comportar outras aplicações. Essa estratégia será analisada na seção 5, e refere-se à proposta arquitetural final dessa dissertação.

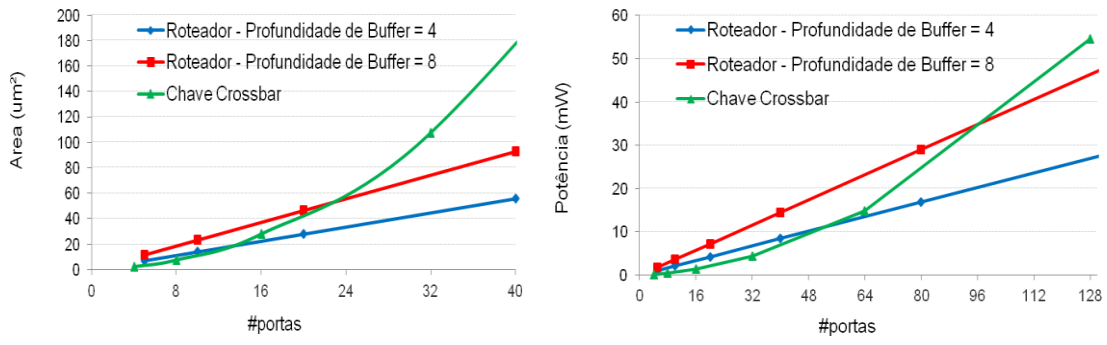
4.2 Estudos Estratégicos

Observando os trabalhos apresentados na literatura, pode-se constatar que é possível obter-se muitas alternativas de redes-em-chip hierárquicas, geralmente justificadas por um nicho de aplicações específicas. Dificilmente conseguimos identificar alguma proposta que apresente uma solução mais genérica em termos de comportamento das aplicações. Assim, pode-se perceber que a que mais se aproxima desse conceito é a proposta por (Freitas, 2008) que utiliza *crossbares* reconfiguráveis. Além disso, o autor

busca a utilização de reconfiguração na arquitetura, o que entra de acordo com os objetivos desse trabalho de dissertação, que busca a integração dessas áreas. Porém sua proposta apresenta um custo muito alto de área e pouca escalabilidade para o sistema. Portanto, é realizado nessa subseção um estudo aprofundado sobre a natureza e limitações das interconexões chaves *crossbares*, a fim de se elaborar uma estratégia eficiente para sua utilização em redes hierárquicas.

4.2.1 Chave *Crossbar*

A utilização de chaves *crossbar* como interconexão representa, numa primeira instância, menor dissipação de potência e maior desempenho, ou seja, maior eficiência energética. Essa constatação se deve pelo fato de possuir uma conexão direta entre dois pontos através de chaves, sem utilização de memorização ou muita lógica para controle, permitindo assim menor latência e dissipação de potência quando comparados com um complexo roteador de uma rede com chaveamentos por pacotes. Porém, à medida que mais conexões são requeridas, mais chaves devem ser colocadas, assim como mais árbitros e fios. Esse custo de lógica combinacional acarreta em uma perda de desempenho e em uma maior dissipação de potência, visto que a cada nova porta inserida, o número de conexões aumenta de forma quadrática, pois essa nova porta deve conectar com as chaves de todas as outras saídas. Baseado nesse fator foi realizado um estudo, para identificar se para futuras aplicações, a chave *crossbar* seria considerada uma limitação ou uma solução em potencial. Para isso, foi realizado um experimento sintetizando chaves *crossbares* e roteadores de redes do tipo malha (com buffer de profundidade 4 e 8 posições), a fim de realizar-se uma análise referente ao custo de área, dissipação de potência (figura 4.10), frequência máxima de operação e densidade de potência (figura 4.11). A variável utilizada foi o número de portas, representando a quantidade máxima de elementos que podem ser conectados. O tamanho do canal de dados utilizado foi de 16 bits sob uma frequência de operação de 100MHz para dados de potência.

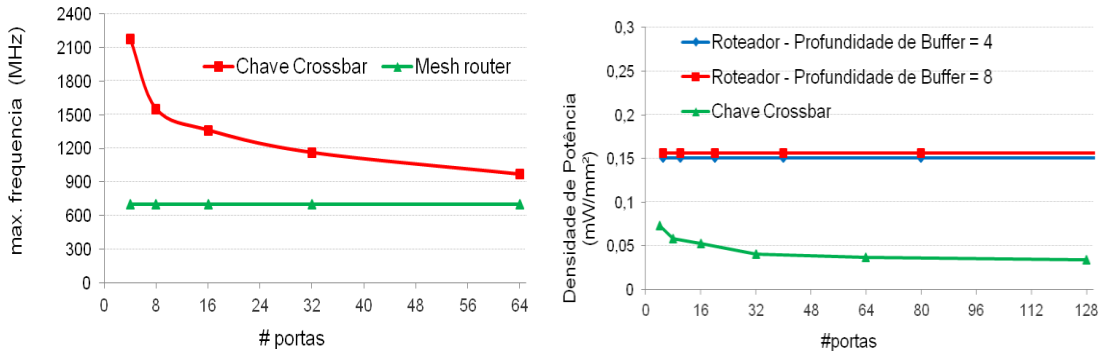


(a) (b)

Figura 4.10 – Gráfico de (a) área, (b) potência do roteador convencional e da chave *crossbar* sobre diferentes portas para conexão.

Observando os resultados obtidos na figura 4.10 e tendo em vista o cenário de simulação utilizado pode-se concluir que para esse caso identifica-se uma vantagem das

chaves *crossbares* até o valor de 16 portas para conexão. Portanto, dependendo das variáveis envolvidas no sistema, essa limitação de granularidade pode variar, sendo um fator importante para entender a escalabilidade dessa estrutura.



(a) (b)

Figura 4.11 – Gráfico de (a) máxima frequência de operação e (b) densidade de potência do roteador convencional e da chave *crossbar* sobre diferentes portas para conexão.

Por outro lado, a figura 4.11 aponta mais vantagens nessa arquitetura, onde ela supera os roteadores independente do número de portas. É interessante ressaltar que para uma mesma configuração de conexões com núcleos, é possível obter menor área, potência e ainda atingir maiores frequências de operação, possibilitando aumentar o desempenho. Esse fator indica grandes possibilidades de explorar a eficiência energética de um sistema de interconexão baseado nessa arquitetura. Além disso, o gráfico de densidade de potência (dado pela relação potência por área) indica que o aproveitamento da potência é mais bem utilizado nessas soluções. Pensando em implementações futuras, como por exemplo, as arquiteturas tri-dimensionais, soluções que apresentem menor dissipação de potência por micrometro quadrado tendem a serem abordagens mais relevantes devido ao desafio de dissipar baixa potência nessas novas tecnologias (TOPOL, 2006).

Outro estudo importante é o entendimento da relação do custo interno dessa estrutura, relativa aos fios perante a lógica. Assim, através de simulações Spice, para tecnologia de 65 nanômetros, foi modelada uma arquitetura simples de *crossbar* sem árbitro, composta apenas por multiplexadores e fios. Verificou-se o resultado de área dos multiplexadores e dos fios para interligá-los em relação à quantidade de portas. Pode-se observar na figura 4.12 que o fio também passará a limitar um sistema baseado em *crossbar*, onde para o caso estudado, as granularidades acima de 16 portas apresentam o seu desempenho limitado pelo fio.

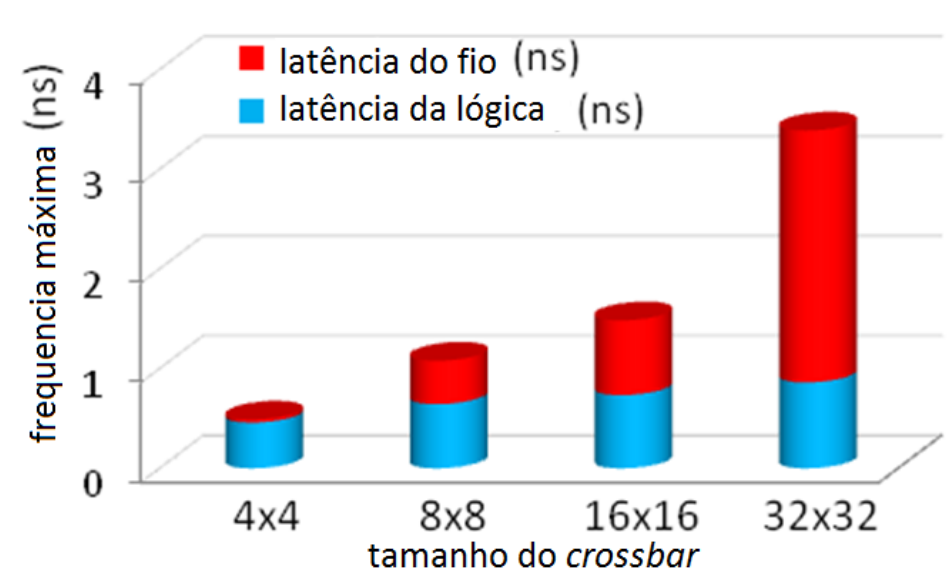


Figura 4.12 – Estimativa de latência de fio e de lógica no *crossbar* para diferentes granularidades.

Concluindo essa subseção analítica, através de dois experimentos encontraram-se fundamentos que demonstram as limitações e potencialidades da arquitetura chave *crossbar* e, assim, possibilitando desenvolver estratégias considerando essas características. O primeiro estudo revelou que em comparação com roteadores convencionais que utilizam chaveamento por pacotes, utilizar mais de 16 portas pode tornar essa solução mais custosa, embora independente desse custo a densidade de potência e frequência máxima de operação ainda se apresentem como fatores favoráveis para um grande número de conexões. O segundo estudo apontou que o fio realmente se tornou a principal limitação dessa arquitetura nas tecnologias mais recentes, pois para 16 portas já apresenta um custo de metade da latência combinacional em tecnologia de 65 nm.

4.3 Arquitetura Implementada – HiCIT

A principal característica a ser explorada por uma arquitetura organizada em hierarquia é a localidade de comunicação presente nas aplicações, ou seja, a afinidade de relação existente entre os núcleos em uma aplicação. Assim, o objetivo é fazer com que os núcleos que se comuniquem com mais frequência estejam mapeados nos mesmos grupos e esses grupos sejam interligados por um nível global de comunicação. A estratégia elaborada nesse trabalho objetiva utilizar uma arquitetura de alto desempenho nos agrupamentos e de médio desempenho para interligá-los. O alto desempenho é garantido através do emprego de chaves *crossbares*, que possui além de desempenho, baixo custo de área e potência para um número reduzido de portas (em torno de 16 portas conforme o estudo apresentado na subseção 4.2). O principal problema dessa solução, a escalabilidade, é ajustado através da replicação de uma estrutura regular de interconexão em uma rede-em-chip. Essa estrutura é composta por um crossbar e um roteador de uma rede-em-chip chaveada por pacotes. Dessa forma, utilizam-se sempre

clusters com um número reduzido de elementos, e quando necessário um grupo maior, os elementos extras são distribuídos em clusters vizinhos, conforme representado na figura 4.13.

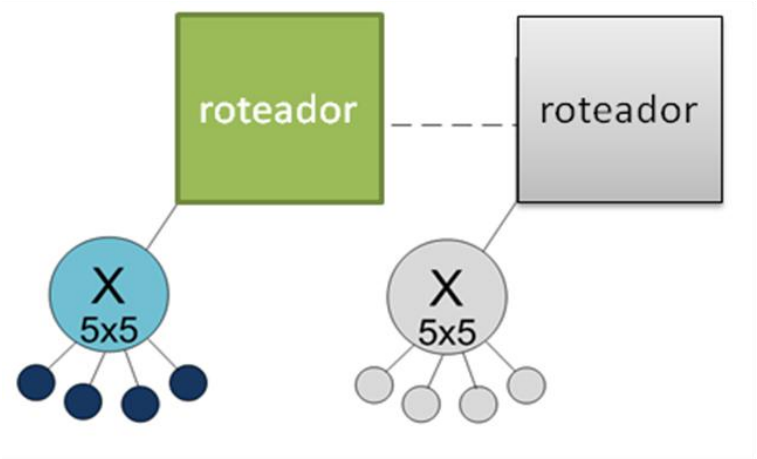


Figura 4.13 - Escalabilidade de *Cluster Crossbar*

Portanto, essa estratégia resultou na arquitetura *Hierarchical Crossbar-based Interconnection Topology* (HiCIT). Nessa proposta, existem dois níveis de hierarquia, onde no topo há uma rede-em-chip chaveada por pacotes com topologia malha, a rede SoCIN, e no nível abaixo (de grupos), *crossbares* parametrizáveis- a arquitetura SwIX (do inglês *Switching Interconnection Xbar*). Essa arquitetura é parametrizável, e dessa forma, é possível a utilização de diferentes granularidades nos agrupamentos, buscando a melhor relação de custo e desempenho para o sistema. Portanto, cada aplicação terá um mapeamento específico, com diferentes granularidades de seus *crossbares*, sendo uma estrutura hierárquica irregular. A figura 4.14 exemplifica a arquitetura HiCIT em uma aplicação que possui 38 núcleos.

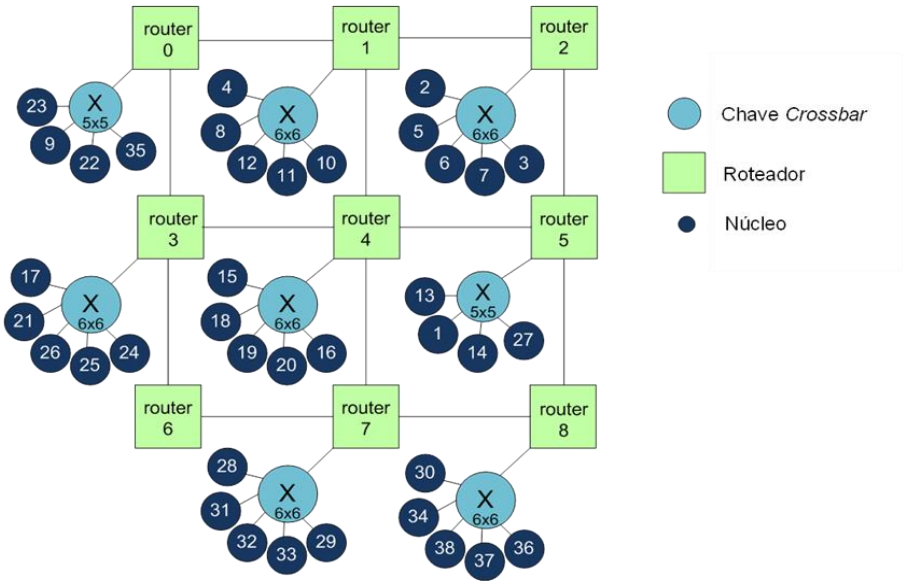
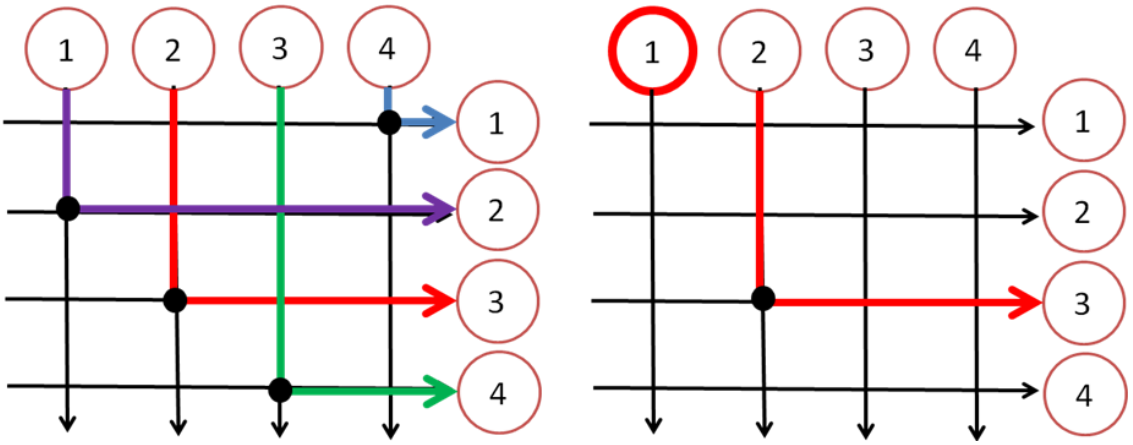


Figura 4.14 – HiCIT com 38 núcleos

O nível global desse sistema é composto pela rede SoCIN, que possui roteadores com cinco portas cada, quatro para conectar com outros roteadores e uma para interligar ao seu grupo, definido pela arquitetura SwIX. Além disso, considerando a estratégia dessa proposta, comunicações entre os grupos ocorrem em uma taxa menor que nos grupos, e por isso, não é justificável utilizar roteadores muito complexos. Portanto, nossa arquitetura permite investir em roteadores simplificados cumprindo os requisitos mínimos de comunicação das aplicações, mas com custos reduzidos de área e potência.

4.3.1 Comportamento Funcional

Para apresentar o comportamento funcional presente no HiCIT, é necessário começar entendendo como funciona a interconexão nos grupos. Como descrito na subseção anterior, os elementos dentro dos grupos são interligados através da arquitetura SwIX, que é uma chave *crossbar* de granularidade parametrizável. Assim, é possível existir múltiplas comunicações paralelas desde que não ocorra conflito de destino, ou seja, sem que mais de um elemento queira enviar dados para um mesmo destino. Quando isso ocorre, o árbitro organiza-os em uma fila, onde um deve esperar o outro comunicar. Esse comportamento está exemplificado através de uma representação de *crossbar* como uma matriz de fios com chaves nos seus cruzamentos, conforme figura 4.15. Uma vez criada essa conexão ela apenas se desfaz quando o núcleo de origem enviar toda a mensagem.



(a) (b)

Figura 4.15 – Em (a) todos se comunicam sem conflito, permitindo alto paralelismo (todos com todos). Em (b), 1 e 2 desejam enviar para a porta do núcleo 3, representando um conflito. Árbitro definiu que 2 comunica e 1 aguarda.

As comunicações que ocorrem dentro da arquitetura SwIX, chamadas de comunicações locais, utilizam apenas uma requisição de envio, seguido por uma confirmação de transmissão dada pelo árbitro da porta desejada. Uma vez que essa confirmação foi feita, os elementos de origem e destino são interligados, se comunicando através do controle de fluxo de dados. Na arquitetura HiCIT, os

problemas de *starvation* são evitados através do emprego de um árbitro com escalonamento *Round Robin*, que atende um pacote por vez de cada requisição.

Considerando a arquitetura HiCIT, comunicações entre elementos de diferentes grupos podem ocorrer. Nesse caso, o comportamento é diferenciado na comunicação, que é transparente para os elementos envolvidos. Para isso, existe um elemento importante que mascara toda a rede-em-chip entre os grupos chamada de *Bridge*. A *Bridge* é responsável por fazer as traduções de protocolos entre os níveis da hierarquia, realizando controle de fluxo e permitindo uma comunicação correta do início ao fim. Para os elementos de um grupo, a *Bridge* representa apenas mais um núcleo conectado ao SwIX, mas que pode assumir a forma de qualquer nodo presente em outros grupos devido a sua transparência de comunicação. Essa estrutura e conexão são mais bem observadas na figura 4.16. Detalhes mais aprofundados dessa arquitetura são apresentados na próxima subseção.

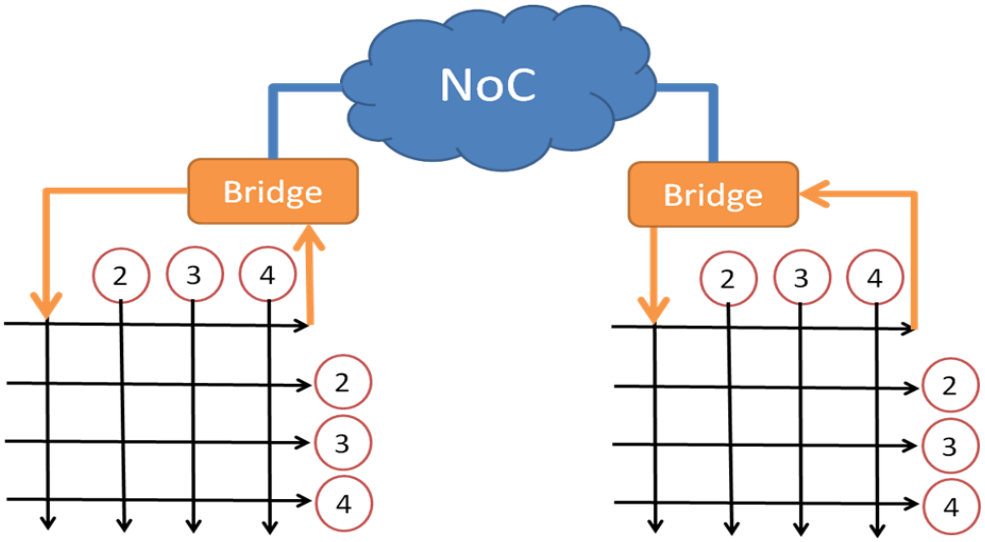


Figura 4.16 – Transparência da rede-em-chip através da arquitetura Bridge.

Em relação à rede-em-chip, as comunicações globais ocorrem conforme a seção 2.2, utilizando chaveamento por pacotes com topologia malha, mecanismo *wormhole* e roteamento XY.

4.3.2 Protocolo

Na proposta HiCIT tem-se bem definidos dois protocolos básicos: o protocolo local e o protocolo global. O local engloba sinais de requisição e confirmação da comunicação, além do controle de fluxo realizado entre a origem e o destino. O protocolo global segue a configuração de dividir pacotes em partes menores, chamados *flits*, e enviá-los pela rede até o seu destino. A parte comum de ambos está no fato de sempre haver um pacote para definir o destino da comunicação. A utilização desses pacotes permite fácil integração entre as diferentes arquiteturas.

Para comunicações intra-cluster o pacote, além da finalidade de identificar o destino no grupo, serve para limitar o tempo de utilização da porta de um determinado elemento, visto que cada comunicação é cessada a cada pacote transmitido. Quando a comunicação é inter-cluster , o pacote passa a ser crucial para definir o grupo destino e, após, o elemento destino dentro desse grupo. Portanto, a integração dos protocolos é realizada através do encapsulamento das informações em um pacote seguindo as definições da rede-em-chip do topo, com a inclusão de um cabeçalho local para o grupo e de um espaço no pacote global para o elemento desejado no *cluster* destino. Assim, o cabeçalho dos pacotes é composto pelo cabeçalho local (informação intra *cluster*), cabeçalho global e cabeçalho local do destino, chamado nesse trabalho de *crossbar header* (informação *inter-cluster*). Na figura 4.17 pode-se observar a composição desse pacote.

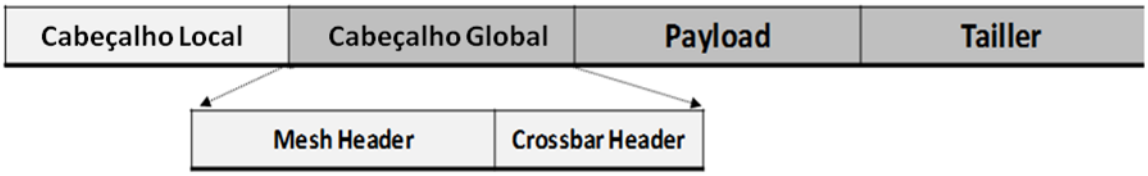


Figura 4.17 – Pacote de dados da arquitetura HiCIT.

No cabeçalho local, tem-se a informação do núcleo destino que é definido como zero sempre que a comunicação for inter-cluster. Nos demais campos, tem-se a configuração referente aos pacotes que trafegam na rede-em-chip, nível topo. Dentro dessa informação, identifica-se o cabeçalho local de destino (*crossbar header*), que será utilizado quando a mensagem encontrar o seu grupo destino.

Em um primeiro momento, o núcleo monta o pacote com a carga útil, adicionando o cabeçalho local com a informação da porta de destino de seu mesmo grupo. Logo, é ativado o sinal de requisição de envio. O SwIX identifica o destino e o respectivo árbitro concede permissão ou não para fechar a comunicação. Caso o destino esteja ocupado em outra transmissão, essa requisição entra em uma fila de espera. No entanto, se a saída estiver livre, a comunicação é fechada e esse sinal de requisição é convertido em um sinal de aviso de envio ao elemento destino, chamado de *valid*. O destino, quando pronto para receber, envia um sinal *ack*. Por fim, com a comunicação fechada e sincronizada, o pacote é transmitido em rajada.

O protocolo de comunicação local segue o fluxo da figura 4.18:

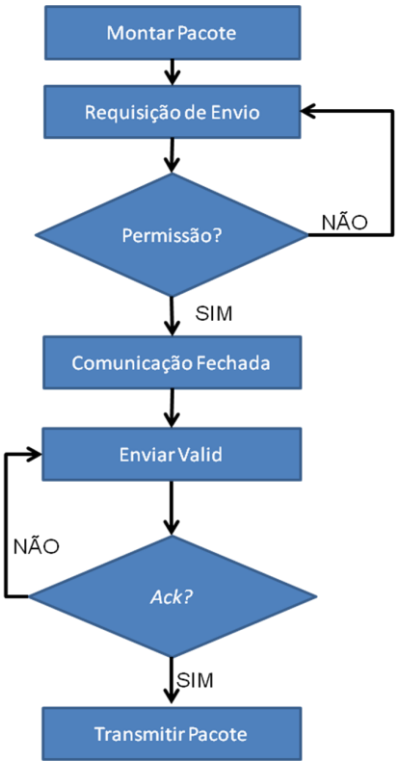


Figura 4.18 – Fluxo do protocolo de comunicação intra *cluster*.

Quando for necessária uma comunicação entre grupos, o protocolo local é realizado duas vezes, além do protocolo global, conforme o fluxo mostrado na figura 4.19:

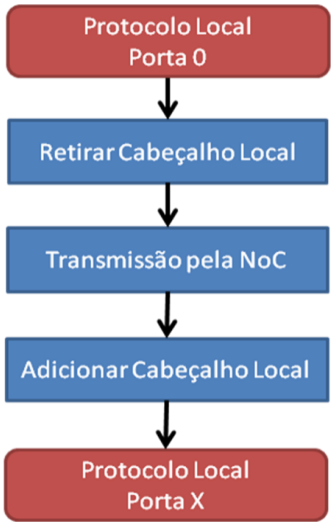


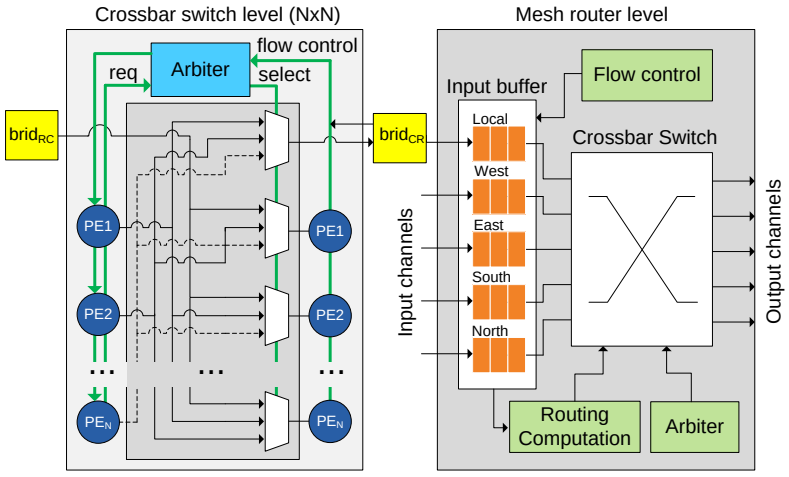
Figura 4.19 – Fluxo do protocolo de comunicação inter *cluster*.

Assim, executa-se o protocolo local, informando ao SwIX que deseja-se a porta zero, reservada para as comunicações para fora do grupo. Uma informação de dado a ser transmitido chega a *Bridge* que retira a informação do cabeçalho local, repassando tal informação para a porta local do roteador. Essa mensagem trafega pela rede-em-chip até

chegar ao seu destino. Na porta local do roteador de destino, outro elemento *Bridge* retira a informação presente no *crossbar header*, criando um novo cabeçalho local, e assim realizando um novo protocolo local onde o o SwIX recebe uma requisição para a porta onde está o núcleo desejado.

4.3.3 Arquitetura

A arquitetura HiCIT é composta por uma chave *crossbar* parametrizável, chamada de SwIX, uma rede-em-chip SoCIN com roteadores RaSoC com *pipeline* (conforme explicado na seção 2.3) e um tradutor de protocolos para interligar os níveis hierárquicos, chamado de Bridge. A representação de um grupo (SwIX, Bridge e RaSoC) está apresentada na figura 4.20. Visto que a arquitetura RaSoC é a mesma explicada na seção 2, não há maiores detalhes que precisem ser descritos nessa seção.



4.20 – Arquitetura HiCIT. (a) SwIX, (b) Bridge e (c) RaSoC.

4.3.3.1 SwIX – *Switching Interconnection Xbar*

A arquitetura SwIX tem por objetivo permitir a comunicação entre os elementos que desejam se comunicar, fechando suas conexões através de chaves multiplexadoras. Porém, esse elemento apenas permite que uma saída seja utilizada por vez, sendo necessária a intervenção do árbitro para resolver conflitos. A arbitragem utiliza o algoritmo Round Robin, que permite evitar *starvation* e garantir um comportamento homogêneo em relação às prioridades. Portanto, essa arquitetura é composta apenas por um conjunto de fios e multiplexadores e uma lógica de controle que implementa a arbitragem e controle de protocolo de comunicação para cada porta de saída, conforme é apresentado na figura 4.21.

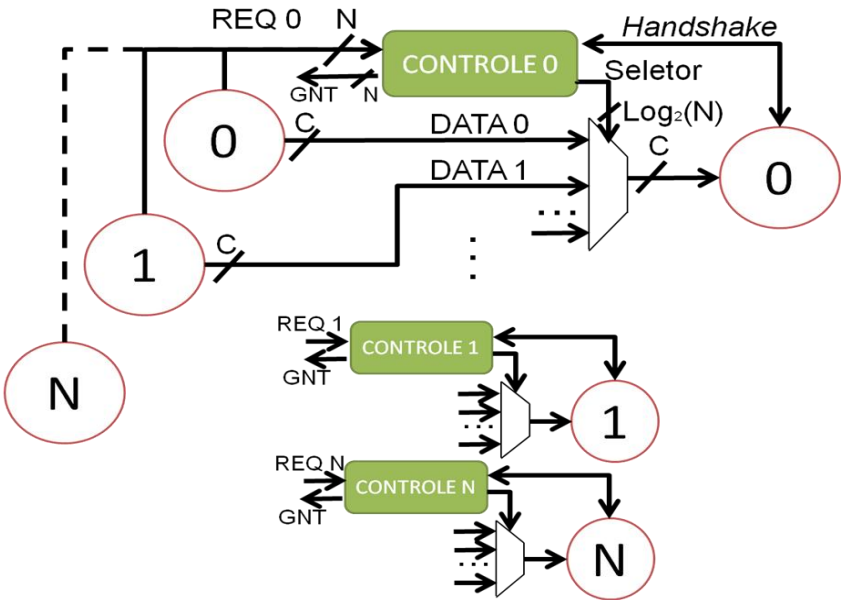


Figura 4.21 – Arquitetura SwIX, onde N é o número de elementos, C é o tamanho do canal, ou seja, a quantidade de bits transmitidos, e quando não houver identificação significa que é utilizado apenas 1 bit.

A figura 4.21 demonstra o caminho de dados, onde para todos os nodos envolvidos existem dois sinais de controle (REQ e GNT) e um barramento de dados (DATA). O sinal de requisição (REQ) existe para cada porta de saída, e as requisições de todos os núcleos são concatenadas formando um barramento de N elementos (N é a granularidade do *crossbar*). O mesmo ocorre com o sinal de confirmação de comunicação (GNT). Assim, a lógica de controle define o sinal "Seletor" fechando o caminho entre um determinado elemento de origem e outro de destino, e realiza o fluxo de controle de dados *handshake* com o destino, atuando como intermediário na comunicação.

A lógica de controle, responsável pela arbitragem e controle do fluxo de dados, realiza suas atividades através de uma máquina de estados finita, com quatro estados, onde são necessários dois ciclos para definir o chaveamento dos multiplexadores. O diagrama de estados da figura 4.22 demonstra o funcionamento desse mecanismo de controle.

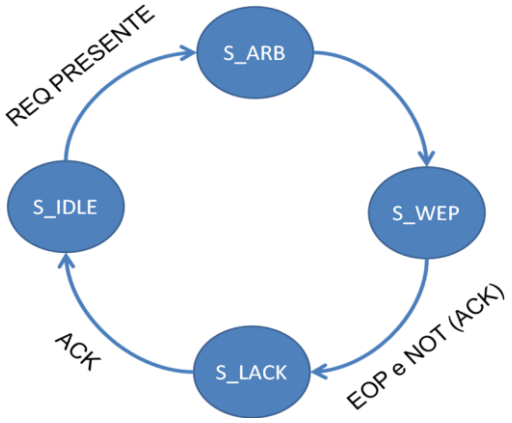


Figura 4.22 – Diagrama de estados da máquina de controle presente no SwIX

Essa máquina de estados inicia com o estado S_IDLE, aguardando alguma requisição de comunicação para essa porta de saída. Assim, quando existe uma ou mais requisições, ocorre uma transição de estado para S_ARB, onde se consulta em uma tabela de prioridades para definir qual requisição deve ser atendida. Após essa consulta que leva um ciclo, passa-se para o estado S_WEP, que modifica os multiplexadores através do sinal "Seletor" e fecha a comunicação entre origem e destino. Além disso, é enviado ao destino um sinal de *valid*, referente à intenção de enviar dados através do protocolo *handshake*. O sinal de retorno desse protocolo, *ack*, é repassado ao elemento origem da comunicação através do sinal "GNT". Dessa forma, a comunicação ocorre livremente até que chegue o *flit* identificado como EOP (*End of Packet* - fim de pacote), que faz a máquina mudar de estado. Nesse caso, também é verificado o sinal de *ack* vindo do destino que informa se se o último dado foi consumido. Se sim, o próximo estado é o S_IDLE, repetindo todos os passos anteriores. Porém, caso não ocorra o consumo imediato dessa informação, tem-se o estado S_LACK, que aguarda o último *ack* do destino, e após isso retorna-se ao modo S_IDLE.

Portanto, essa implementação do algoritmo *Round Robin* necessita de dois ciclos de relógio para verificar e definir qual porta tem a permissão para utilizar a saída respectiva, independente do tamanho do crossbar. No entanto, o tempo de um ciclo de relógio depende da granularidade do *crossbar*, onde a máxima frequência de operação pode ficar limitada por esse fator. A arquitetura SwIX é totalmente parametrizável, permitindo alterar rapidamente o número de portas e o tamanho dos canais em tempo de projeto.

4.3.3.2 Bridge

A arquitetura Bridge possibilita uma comunicação transparente entre os níveis hierárquicos da HiCIT, manipulando as informações nos pacotes para que tanto na rede-em-chip SoCIN quanto no *crossbar* SwIX a comunicação continue correta. Para isso, existem duas sub-arquiteturas, a Bridge-CR e a Bridge-RC. A Bridge-CR se refere a comunicação *cluster* para roteador, onde a principal tarefa é retirar o cabeçalho local e repassar o resto do pacote. Em contraponto, a Bridge-RC realiza a comunicação roteador para *cluster*, onde a informação de destino local é retirada de dentro do

cabeçalho global do pacote recebido, em uma área reservada (conforme visto na subseção 4.3.2). Portanto, essa arquitetura realiza uma manipulação das conexões para eliminar informação, conforme arquitetura Bridge-CR na figura 4.23, ou adicionar informação, conforme demonstra a estrutura arquitetural da Bridge-RC na figura 4.24.

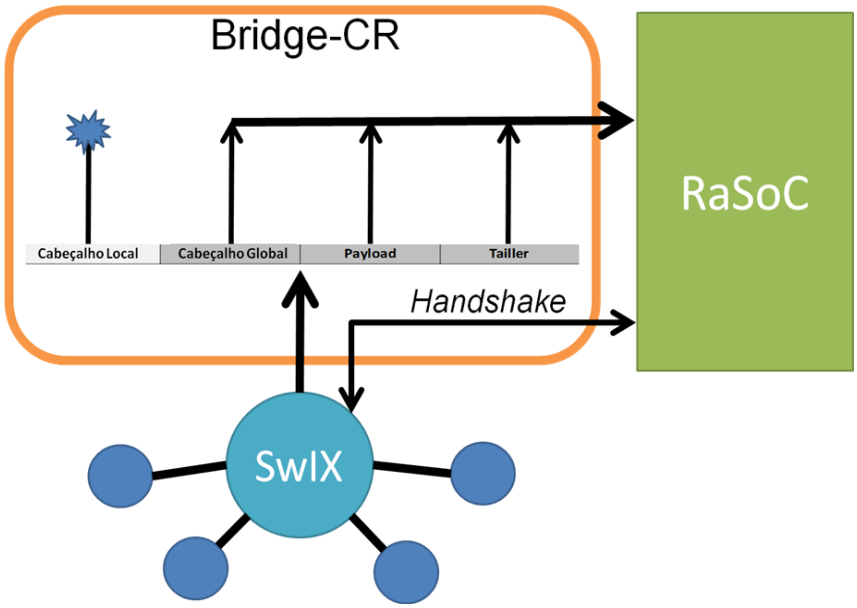


Figura 4.23 – Arquitetura Bridge-CR

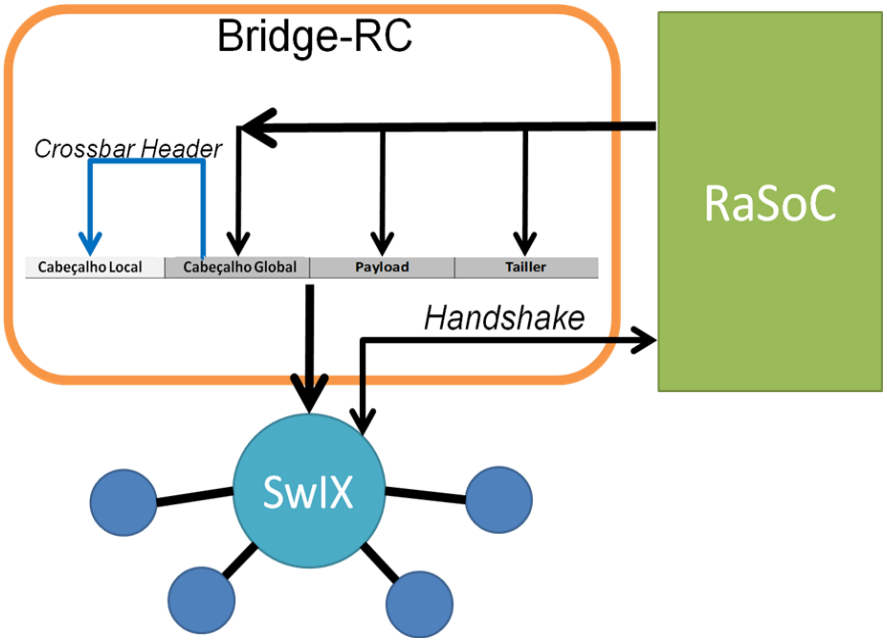


Figura 4.24 – Arquitetura Bridge-RC

A partir da figura 4.23, observa-se que quando um pacote sai de um grupo para a rede malha, o pacote é enviado pela porta reservada do SwIX destinada para o nível

topo (porta zero) e o Bridge apenas repassa a mensagem para a porta local do roteador retirando a informação de controle de comunicação do grupo (cabeçalho local).

No entanto, na figura 4.24, é possível identificar que quando um pacote vem do topo para um grupo, a *Bridge* deve identificar dentro do cabeçalho global, em um espaço reservado, qual o elemento daquele grupo é o núcleo de destino. Assim, ele adiciona essa informação à mensagem criando um novo cabeçalho local. Além disso, ambas *Bridges* repassam a informação de todo controle de fluxo entre os níveis evitando perda de pacotes.

Conforme apresentado nas figuras 4.23 e 4.24, a implementação das Bridges é simplificada, pois ela apenas realiza um reordenamento dos dados nos fios. No entanto, a Bridge-CR poderia ter uma função similar a uma interface de rede, podendo vir a ter um buffer para melhorar a eficiência de vazão de dados. O conceito das arquiteturas dos Bridges foi elaborado almejando inserir um mecanismo de sincronização para domínios com diferentes frequências de relógio, permitindo uma abordagem GALS (Globalmente Assíncrono Localmente Síncrono).

4.4 Considerações

Nessa seção foram apresentadas as arquiteturas de interconexão hierárquicas. Pode-se identificar que essa estratégia busca balancear o custo de diferentes arquiteturas com o desempenho desejado para uma determinada aplicação, sendo uma proposta de arquitetura totalmente dependente à aplicação. É possível encontrar diferentes propostas na literatura referente a estruturas hierárquicas, onde a meta sempre se encontra na redução de área, potência ou energia. Nesse trabalho identificou-se um grande potencial na utilização de chaves *crossbares* para ser utilizado na abordagem hierárquica, visto seu grande desempenho e baixo custo para poucas conexões. Para determinar o limite e melhorar o entendimento das limitações de qualquer projeto baseado nessa arquitetura, uma análise de custos deve ser realizada, para identificar, sob um determinado cenário, qual seria a limitação adequada. Em nossas análises, obtivemos como número máximo 16 portas. Portanto, essa seção demonstrou a elaboração da arquitetura HiCIT, que apresenta uma rede-em-chip convencional no topo da hierarquia e uma chave *crossbar* parametrizável para formar os níveis mais baixos, de grupos.

5 PROPOSTA DE ARQUITETURA HIERÁRQUICA ADAPTÁVEL - HASIN

Nessa seção, é apresentada a principal contribuição desse trabalho, a *Hierarchical Adaptable System Interconnection Network* - HASIN. Essa arquitetura utiliza o conceito hierárquico e reconfigurável na mesma estrutura, e tem por objetivo oferecer arquiteturas específicas mais flexíveis. Essa proposta foi elaborada a partir da integração direta das duas propostas dessa dissertação. Portanto, HASIN é a composição de MINoC com HiCIT, promovendo uma arquitetura hierárquica auto-adaptável, onde possui *crossbars* nos níveis locais e reconfiguração topológica no nível global.

HASIN possui características para lidar com comportamentos de comunicação específicos, por estar organizado em uma hierarquia, mas permite reconfiguração no nível topo, adaptando possíveis aumentos de tráfegos entre clusters, considerados atípicos. Dessa forma, três hipóteses justificam a utilização dessa arquitetura. A primeira está nas aplicações que realmente possuem um comportamento de comunicação diferenciado, sem padrões fixos. A segunda estaria em sistemas com alocação e migração dinâmica de tarefas, onde o recurso de reconfiguração ampliaria as possibilidades para as decisões feitas pelo controlador responsável das tarefas. A terceira se refere à limitação do hardware dado o gradativo aumento na complexidade das aplicações, ou seja, a evolução natural de uma aplicação não necessitaria um novo sistema, visto que a característica adaptativa garantiria desempenho mesmo com novos comportamentos na comunicação. Por exemplo, atualizações de sistemas operacionais de *smartphones*, que podem exigir um maior processamento gráfico ou outras tarefas dedicadas.

5.1 Comportamento Funcional

Nessa arquitetura, há dois comportamentos funcionais distintos, que juntos criam um novo conceito comportamental. Basicamente, tem-se o comportamento dado pela chave *crossbar* existente nos agrupamentos (*cluster*) da hierarquia (descrito na subseção 3.3.1) e pelo roteador reconfigurável encontrado no topo dela (descrito na subseção 4.3.1). Porém, ao se obter uma conexão ponto-a-ponto dentro do crossbar aliado a uma conexão ponto-a-ponto através da técnica de passagem direta, gerada na rede-em-chip, cria-se um caminho dedicado entre diferentes hierarquias, culminando em novas configurações topológicas.

Exemplificando, em uma situação sem contenção de comunicação, pode-se reconfigurar inúmeras topologias nesse sistema, independente da hierarquia, ajustando o hardware às necessidades da aplicação. Porém, o fato de existir apenas um canal de acesso à rede do topo limita essas possibilidades. Outras formas de interligar o topo da hierarquia aos *clusters* não são abordadas nesse trabalho, pois não está no escopo estudar diferentes possibilidades de integração e necessitaria reavaliar questões de *deadlock*, *livelock* e *starvation*. Na figura 5.1 tem-se um exemplo de possível reconfiguração topológica, levando em conta os *crossbares* e os chaveamentos por circuito. Nesse exemplo, como não há disputa por mesmas portas de saída, o grafo do comportamento de comunicação da aplicação se mostra igual à topologia gerada.

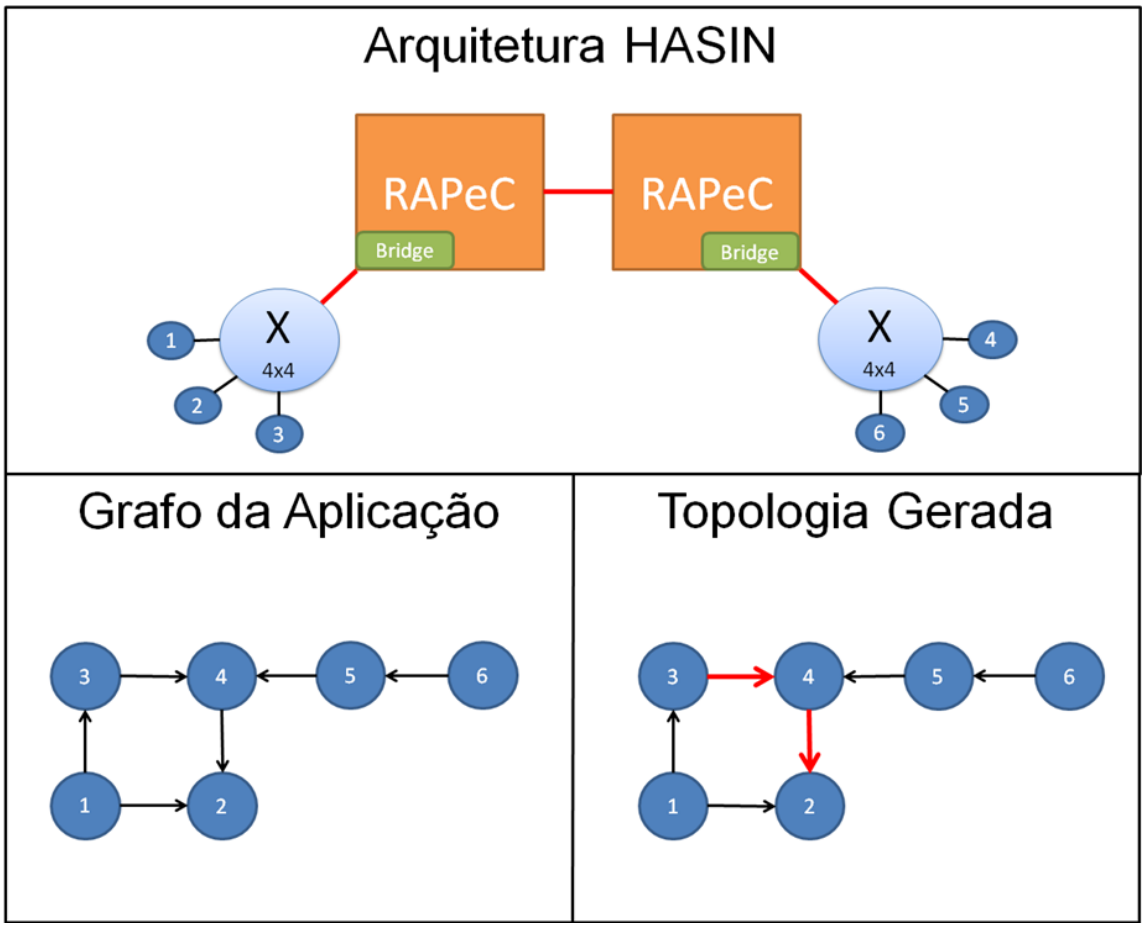


Figura 5.1 – Arquitetura HASIN e Topologia lógica para comunicação sem contenção

No exemplo acima se pode observar que há uma ênfase em vermelho para as conexões entre o nodo 3 e 4 e também entre os nodos 4 e 2, que representam os enlaces que passam pelo topo da hierarquia - comunicação entre *clusters*. Essa ênfase demonstra que o custo de comunicação nesses casos é maior do que nas outras conexões. De qualquer maneira, no cenário sem contenção, essas comunicações estão utilizando um caminho dedicado, buscando um mínimo de latência.

Quando existir contenção local, no *cluster*, o núcleo deve aguardar a autorização do árbitro. E, quando ocorrer contenção global, no topo, os dados passam a trafegar por meio de chaveamento por pacotes, sendo armazenados nos buffers dos roteadores. Em

ambos os casos, tem-se a latência prejudicada, possuindo um desempenho equivalente a rede HiCIT. Assim, no pior caso, essa solução tem desempenho equivalente à proposta hierárquica. Observando a topologia logicamente criada, considerando contenção em toda a rede, não é possível configurar caminhos dedicados, resultando em uma estrutura lógica equivalente à estrutura fisicamente fabricada, a malha.

5.2 Protocolo

O protocolo existente na arquitetura HASIN compreende ao mesmo tempo as considerações existentes na MINoC e na HiCIT. Todos os protocolos existentes no MINoC, relativos à configuração dos chaveamentos, podendo ser UCS, BCS e OS, são realizados. Da mesma forma, os protocolos de requisição e arbitragem das chaves crossbar SwIX são os mesmos, inclusive o protocolo utilizando nas *Bridges*. Porém, ambos os protocolos utilizam informações de controle incorporadas nos cabeçalhos dos pacotes. Portanto, a consideração a ser realizada nessa subseção é como organizar a informação de ambas as estratégias no mesmo pacote.

O pacote na HASIN possui dois cabeçalhos: o local e o global. Assim, como no HiCIT, o cabeçalho local possui o identificador da porta de destino dentro do mesmo grupo, para o SwIX local. E, caso seja enviado para uma porta de saída do grupo, as *Bridges* atuarão retirando esse cabeçalho, deixando apenas a informação global. Ao trafegar pela rede global, agora reconfigurável, a informação relativa ao tamanho do fio deve estar armazenada em uma área reservada. Além disso, logo após essa informação, tem-se a definição de qual porta do grupo de destino o pacote deve ser enviado. Todas estas informações presentes no cabeçalho do pacote estão representadas na figura 5.2.

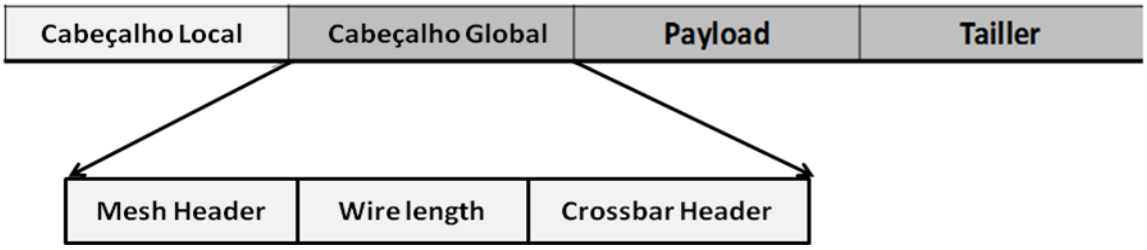


Figura 5.2 – Pacote na Rede HASIN

5.3 Arquitetura

No contexto da arquitetura fisicamente implementada pode-se novamente assumir que tem-se a união direta das arquiteturas MINoC e HiCIT, apresentadas nas subseções 3.3.3 e 4.3.3 respectivamente. Portanto, a rede HASIN é composta por roteadores RAPeC no topo, com hierarquias formadas pelos crossbares parametrizáveis SwIX. Além disso, a arquitetura *Bridge* está presente em todas as divisões hierárquicas.

5.4 Considerações

Nessa seção foi apresentada a proposta da arquitetura hierárquica e adaptável HASIN. Essa rede de interconexão utiliza como base uma estruturação hierárquica, onde os grupos são estruturados com a arquitetura baseada em *crossbar* chamada SwIX, e roteadores no nível topo com características de reconfiguração. Essa solução busca utilizar ao máximo o contexto de chaveamento por circuito, visto que mesmo quando os pacotes trafegam entre as hierarquias, a técnica de conexão direta (*bypass*) é empregada, melhorando resultados de latência. Porém, como penalidade pela inserção de reconfiguração, o custo dos roteadores presentes na rede topo é maior que numa solução puramente hierárquica.

.

6 AVALIAÇÃO DAS ARQUITETURAS

A etapa de avaliação compreende criar um ambiente de experimentação para identificar o desempenho e custo de hardware das arquiteturas propostas perante a solução convencional, a rede-em-chip. Assim, espera-se comprovar a eficiência e as penalidades de cada estratégia, já consideradas quando elaboradas. Além disso, nessa etapa, realiza-se uma análise qualitativa da relação entre os diferentes conceitos abordados, clarificando qual a relação numérica entre os benefícios e custos de cada solução.

Para a elaboração desse ambiente de avaliação, é necessário considerar múltiplos fatores, sendo uma etapa de grande planejamento para obtenção de resultados confiáveis sem favorecer alguma proposta. O primeiro fator a considerar é definir as aplicações ou *benchmarks* que realmente apresentem um cenário que se aproxime ao previsto pelo ITRS 2011 (ITRS, 2011), onde há muitos elementos com diferentes taxas de comunicação. A segunda consideração se refere ao mapeamento dessas aplicações para cada arquitetura, sendo necessária uma métrica fixa para realizar essa tarefa para todas as arquiteturas. A terceira envolve simulação que deve ser realizada em níveis altos de abstração, devido à alta complexidade e grande número de elementos envolvidos. Finalmente, as estratégias de síntese e comparação dos resultados, obtendo as informações necessárias para as conclusões finais e definição do rumo dos trabalhos futuros.

Nas próximas subseções são apresentados os *benchmarks* utilizados (provenientes da literatura), as ferramentas de software essenciais para experimentação utilizada e as metodologias, os resultados obtidos, as análises e considerações finais.

6.1 *Benchmarks*

Para a escolha das aplicações, é necessário que elas satisfaçam dois requisitos para estarem de acordo com a metodologia de experimentação empregada, sendo eles:

- Comportamento de comunicação desbalanceado (diferentes taxas entre os elementos) contendo múltiplos núcleos heterogêneos;
- Informação de área física dos elementos envolvidos;

O comportamento de comunicação, definido por um grafo de aplicação, informa o número de elementos, as taxas de comunicação entre eles e como os nodos estão conectados, identificando as afinidades de comunicação. Por outro lado, a área física identifica a heterogeneidade da aplicação, pois quando consideradas questões de *floorplanning*, tais parâmetros implicam em diferentes custos de conexões e fios no sistema. Sendo assim, utilizando essas considerações, identificam-se duas aplicações amplamente citadas na literatura, que apresentam as informações dos comportamentos de comunicação e suas implementações. A primeira é referente a um decodificador de vídeo, cujas funções básicas foram triplicadas para criar um cenário mais desafiador para projetos atuais, chamada de TVOPD ou *Triple Video Object Plane Decoder* (MURALLI, 2009). A segunda aplicação apresenta um conjunto reduzido do comportamento de comunicação que ocorre nas redes de computadores utilizando protocolos TCP/IP, chamada de NCS ou *Network Communication Subsystem* (TINO, 2010).

6.1.1 TVOPD

TVOPD é a abreviação para *Triple Video Object Plane Decoder* (MURALI, 2009), que utiliza 38 núcleos no total. Essa aplicação é definida como processamento de multimídia e apresenta um alto grau de heterogeneidade, possuindo três decodificadores independentes trabalhando em paralelo. Cada decodificador possui 12 núcleos de processamento, dispostos em forma de *pipeline*. Existem também duas memórias nesse sistema, que são compartilhadas por todos os decodificadores, servindo de buffers de entrada e saída.

O grafo de comunicação é apresentado na figura 6.1, onde os números em cada vértice representam as taxas de comunicação entre os núcleos, em MB/s.

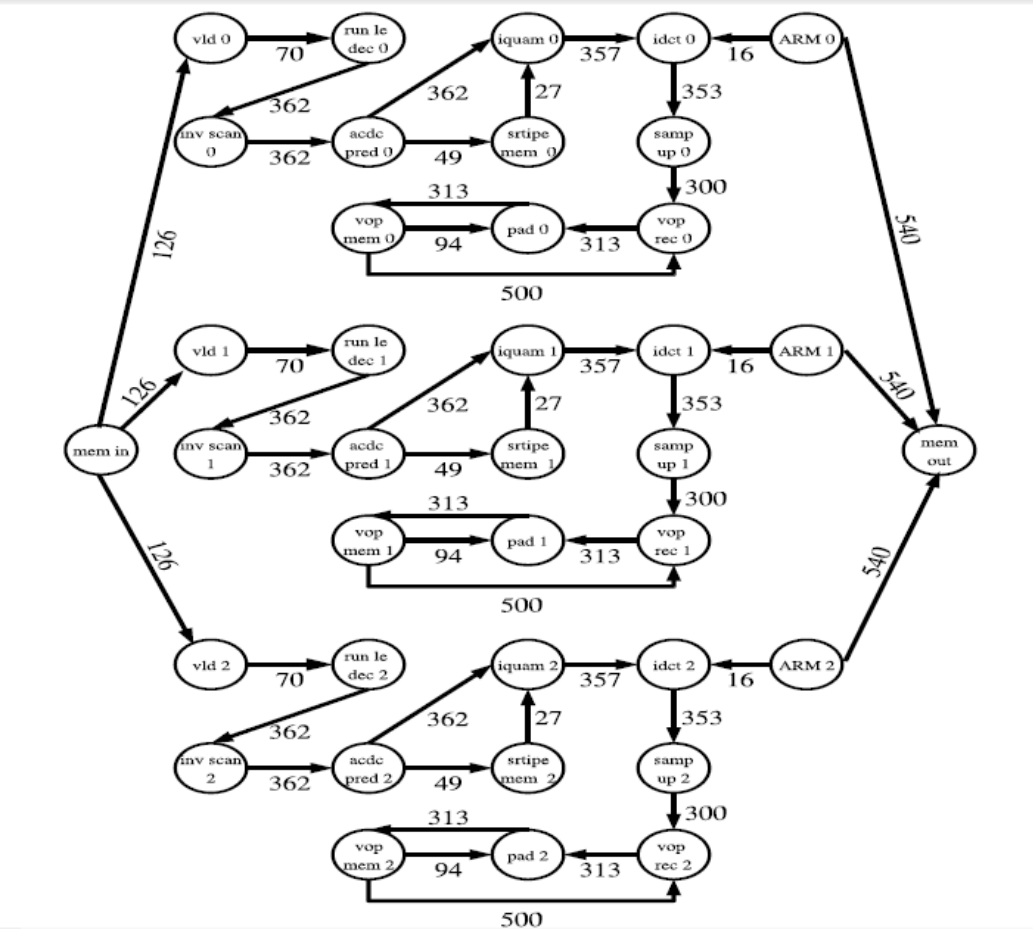


Figura 6.1 – Grafo da aplicação TVOPD

Além do grafo da aplicação é necessário identificar as informações físicas de cada elemento presente nessa aplicação. Dessa forma, em (ATIENZA, 2008) é apresentada a informação de *floorplanning* da aplicação VOPD (ver figura 6.2), que pode ser utilizada para nossa aplicação. A justificativa é que o VOPD apresenta todos os elementos presentes em TVOPD, possibilitando utilizar essa informação para a aplicação alvo.

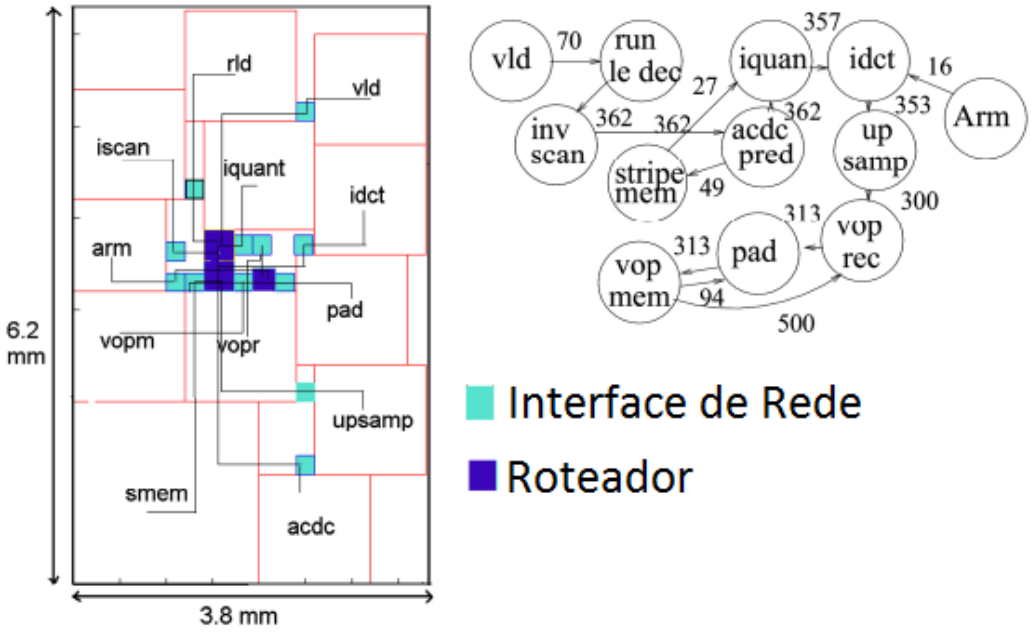


Figura 6.2 – *Floorplanning* da aplicação VOPD

As informações físicas apresentadas na figura 6.2, apresentado por (ATIENZA, 2008), utilizaram a tecnologia de 130 nanômetros, sendo necessário realizar um calculo de escala (*technologic scaling*) para obtermos uma estimativa das respectivas áreas para 65 nanômetros, tecnologia alvo desse trabalho. Os resultados desses cálculos estão apresentados na tabela 6.1.

Tabela 6.1 – Conversão de área de 130 para 65 nanômetros

Núcleos	130 nm			65 nm
	comprimento (mm)	largura (mm)	área (mm²)	Área (mm²)
RLD	1,21	1,19	1,44	0,61
ISCAN	1,21	1,18	1,44	0,61
ARM	0,98	0,98	0,96	0,41
VLD	1,18	1,18	1,41	0,59
IDCT	1,18	1,18	1,41	0,59
PAD	1,18	1,19	1,42	0,60
UPSAMP	1,18	1,19	1,42	0,60
ACDC	1,17	1,18	1,39	0,59
SMEM	1,98	1,98	3,93	1,66
VOPM	1,18	1,19	1,42	0,60
VOPR	1,19	1,19	1,43	0,60
IQUANT	1,19	1,17	1,40	0,59

6.1.2 NCS

A aplicação NCS, do inglês *Network Communication Subsystem*, é apresentada em (PARISCHA, 2004). O autor apresenta NCS como um subsistema do tratamento de comunicação para sistemas em chip que realizam rápido processamento de pacotes de dados e transferência direta de pacotes, em sua maioria da pilha TCP/IP. Nesse *benchmark*, 15 núcleos se comunicam entre si com diferentes taxas de comunicação, conforme mostrado no grafo da figura 6.3.

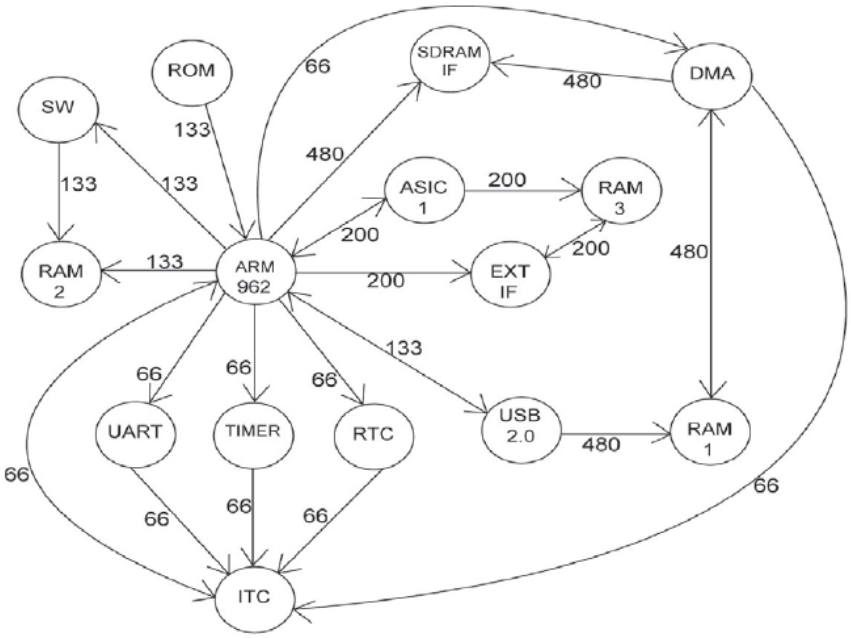
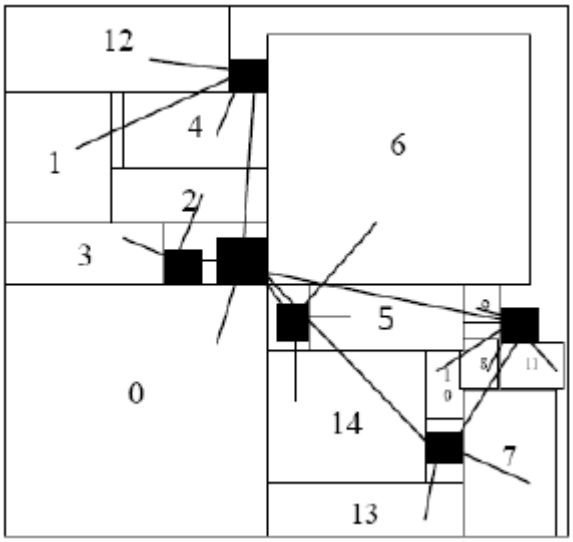


Figura 6.3 – Grafo da aplicação NCS

Da mesma forma que para a outra aplicação, é necessário conhecer a área física dos elementos envolvidos para o nodo tecnológico de 65 nanômetros. Assim, em (TINO, 2010) é desenvolvido um *floorplanning* com essa aplicação, figura 6.4, apresentando as características físicas já para 65 nanômetros. Dessa forma, não foi necessário realizar conversões, obtendo-se assim as informações descritas na tabela 6.2.



ID	Núcleo
0	ARM
1	ROM
2	SWITCH
3	RAM2
4	USB2.0
5	EXT-IF
6	ASIC1
7	SDRAM-IF
8	TIMER
9	UART
10	RTC
11	ITC
12	RAM1
13	DMA
14	RAM3

Figura 6.4 – *Floorplanning* da aplicação NCS.

Tabela 6.2 – Informações físicas dos núcleos presentes na aplicação NCS

Núcleos	Comprimento (mm)	Altura (mm)	Área (mm²)
ARM	1,20	1,16	1,39
ROM	0,50	0,60	0,30
SW	0,71	0,24	0,17
RAM2	0,73	0,28	0,21
USB2	0,66	0,35	0,23
EXT-IF	0,70	0,34	0,23
ASIC1	0,85	1,15	0,99
SDRAM-IF	0,43	0,68	0,29
TIMER	0,16	0,30	0,05
UART	0,33	0,21	0,07
RTC	1,03	0,40	0,41
ITC	0,91	0,25	0,23
RAM1	0,91	0,59	0,54
DMA	0,90	0,27	0,24
RAM3	0,72	0,60	0,43

6.2 Ambiente de Experimentação

Essa subseção descreve todas as ferramentas utilizadas no presente trabalho, fundamentais para realizar os experimentos. Esse ambiente envolve diferentes *softwares*, sendo utilizados aplicativos comerciais, de código aberto (*open-source*) e acadêmico. Para facilitar o entendimento, as áreas de atuação estão classificadas em três

ambientes de trabalho, sendo elas simulação, síntese lógica com informação de parasitas e suporte, dado por duas ferramentas específicas, conforme descrito abaixo.

6.2.1 Simulação

Para simulação funcional dos códigos *vhdl*, utilizou-se a ferramenta comercial *Modelsim*. Essa ferramenta permite simular e verificar o comportamento funcional de arquiteturas desenvolvidas. O propósito principal dessa ferramenta é de verificação funcional dos elementos de hardware desenvolvidos. Para simulações de sistemas completos utilizou-se uma ferramenta de simulação que utiliza um alto nível de abstração (explicada na subseção 6.2.3), para reduzir os custos de tempo de computação e complexidade na obtenção e análise de resultados.

6.2.2 Síntese Lógica com Informação de Parasitas

O ambiente que compreende a tarefa de síntese lógica com informação de capacitâncias parasitas foi realizado utilizando-se ferramentas comerciais da empresa *Cadence Design Systems*. Assim, o *software RTL Compiler* é utilizado para realizar as sínteses lógicas dos sistemas desenvolvidos, obtendo assim resultados de área, potência e frequência máxima de operação em tecnologia de células padrão em 65 nanômetros.

Além disso, para realizar análises precisas com os fios, é necessário realizar uma síntese física simplificada para extrair as informações dos parasitas (capacitâncias e resistências) dos fios gerados. Para isso, utilizou-se a ferramente *First Encounter*, também da empresa *Cadence*, com nove camadas de metal nos roteamentos, para geração do arquivo tipo *SPEF*. Esse arquivo é realimentado na síntese lógica para recálculo dos custos.

6.2.3 Suporte - Floorplanning-Aware Design Space Exploration Framework

A primeira ferramenta de suporte específico é intitulada de *Floorplanning-Aware Design Space Exploration Framework* e sua função é realizar mapeamento automático de aplicações para arquiteturas de redes-em-chip convencionais e redes hierárquicas baseadas em NoC e *crossbares* (MATOS, 2011). Essa ferramenta tem uma importância fundamental para esse trabalho, não apenas em relação à tarefa de mapeamento ótimo com um menor custo de tempo, mas, principalmente, por permitir definir uma métrica padrão para a tradução das aplicações às arquiteturas. Dessa forma, tem-se um mecanismo modelo que realiza as configurações arquiteturais, sempre observando as mesmas características da aplicação: área e taxas de comunicação.

Para utilizar essa ferramenta, é necessário fornecer informações da aplicação, física dos elementos e interconexão, ou seja, arquivos que descrevam o grafo da aplicação, incluindo as taxas, o tipo de arquitetura de interconexão utilizada, a área dos núcleos e a quantidade de recursos disponíveis ou desejados. Assim, a ferramenta utiliza essas informações para definir possíveis *floorplans*, para estimar custos de fios no sistema e

junto com as informações de desempenho, encontrar o mapeamento que melhor satisfaz na relação de área x potência x latência. Para definir qualquer mapeamento é realizado um fluxo de operações iterativo na ferramenta, que utiliza um algoritmo genético para a escolha das mesmas (MATOS, 2011). O fluxo de iterações pode ser observado na figura 6.5.

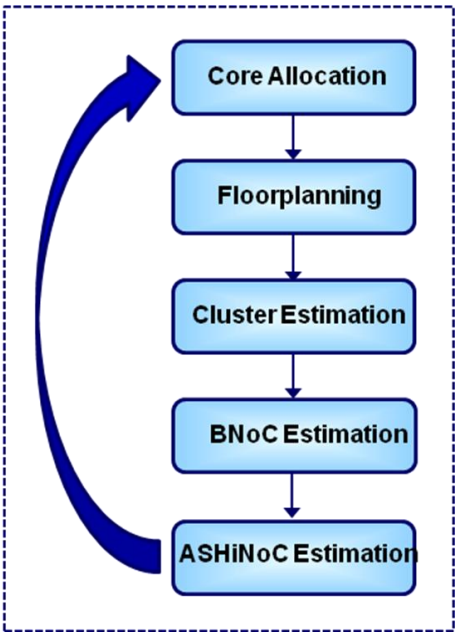


Figura 6.5 - Fluxo do Framework de mapeamento

A primeira etapa no fluxo se chama *Core Allocation*, e nessa etapa ocorre o mapeamento da aplicação em grupos de afinidade, identificando os núcleos que devem ficar próximos. Para essa tarefa, o algoritmo genético NSGA-II é aplicado para resolver o problema de existir um espaço de projeto muito vasto. Após, tem-se a etapa *Floorplanning*, onde é realizada uma estimativa dos *floorplannings* considerando todas as estruturas de hardware presentes, sendo geradas diferentes possibilidades de arranjo físico das mesmas. Nessa etapa a ferramenta utiliza um aplicativo incorporado, chamado *Hotfloorplan*. Seguindo o fluxo da ferramenta, tem-se o *Cluster Estimation*, que para arquiteturas hierárquicas é realizada uma estimativa do custo de área, potência e latência para *crossbares*, dado os *floorplannings* gerados na etapa anterior. É importante ressaltar, que nessa etapa são consideradas as implicações dos fios. Para arquiteturas não hierárquicas, essa etapa é ignorada, passando direto para *BNoC Estimation*, que realiza a mesma estimativa, mas considerando a arquitetura dos roteadores. Ao final, é realizada uma estimativa geral das arquiteturas e calculada a função de custo final, dado pela multiplicação direta dos valores estimados de potência, área e latência, e assim, essas informações realimentam o processo novamente para que o algoritmo genético possa aperfeiçoar os mapeamentos.

6.2.4 Suporte - *Simulation Framework*

Para realizar simulações de grande complexidade envolvendo inúmeros elementos se fez necessário utilizar um software de simulação comportamental de redes-em-chip, desenvolvido por (Kreutz, 2005). Essa aplicação permite configurar a arquitetura, as taxas de comunicação entre os elementos, e de maneira prática obter resultados de latência e vazão para cada comunicação.

A maior justificativa de utilizar essa ferramenta está no fato da dificuldade de analisar formas de onda para obter dados de latência e vazão. Para configurar a ferramenta, as informações em nível de abstração RTL (*Register Transfer Level*) de cada arquitetura são extraídas através de simulações, e traduzidas para a linguagem de programação Java, utilizada. Essas informações contem o comportamento, bem como a temporização em ciclos de relógio para cada ação realizada em cada arquitetura. Dessa forma, é possível obter um simulador comportamental com alta fidelidade da arquitetura, permitindo configurar para executar uma aplicação com dezenas de elementos. Além disso, por ser um *software*, o acesso aos dados e seu processamento durante a execução da simulação é facilitado, permitindo rápido acesso aos resultados.

6.3 Metodologia de Experimentação

A metodologia para a experimentação de cada arquitetura realizada está disposta em quatro etapas de procedimentos. Elas são:

- Simulação unitária
- Mapeamento
- Simulação de sistema
- Síntese

Primeiramente, a simulação unitária tem por objetivo verificar a funcionalidade de cada elemento arquitetural presente nas propostas desenvolvidas. Isso garante a funcionalidade correta e aplicabilidade no sistema. Além disso, são extraídas todas as informações comportamentais e temporais necessárias para o *software* de simulação de sistema. A seguir, ocorre a etapa de definição da topologia e o mapeamento, realizado através do *Floorplanning-Aware Design Space Exploration Framework*. Esse *software* define como serão formado os sistemas para experimentação, relacionando as arquiteturas com as aplicações. Logo, tem-se a simulação de sistema, que é a simulação da aplicação na arquitetura completa realizada pelo *Simulation Framework*. Nessa etapa são obtidos os dados de latência e vazão. Finalmente, ocorre a etapa de síntese lógica, onde o sistema inteiro é sintetizado utilizando abstrações dos núcleos, sendo analisadas apenas as arquiteturas das interconexões. Nessa etapa também são extraídas informações dos fios para calcular todos os custos arquiteturais. Nas seguintes subseções essas etapas são aprofundadas, considerando cada aplicação e arquitetura.

6.3.1 Simulação Unitária

Nessa etapa, cada elemento arquitetural mínimo (arquitetura unitária) é verificado separadamente. Definem-se como elemento arquitetural mínimo as partes que são replicadas nos sistemas de interconexão, como os roteadores, as Bridges e os crossbares. Para essas simulações, utiliza-se um tráfego de dados sintético, apenas com o intuito de extrair o comportamento funcional de cada arquitetura e identificar possíveis erros. As verificações funcionais realizadas estão definidas na tabela 6.3.

Tabela 6.3 – Relação de Verificações Funcionais Unitárias

Verificação Funcional	**Arquitetura Unitária**
Teste 1	RaSoC com *Pipeline*
Teste 2	RAPeC
Teste 3	RaSoC com *Pipeline* + Bridge + SwIX
Teste 4	RaPeC com *Pipeline* + Bridge + SwIX

Em relação à tabela 6.3, é apresentado quatro testes fundamentais. O teste 1 foi realizado para verificar o correto funcionamento da inserção de *pipelines* na estrutura do roteador RaSoC da rede SoCIN. Assim, enviaram-se pacotes sintéticos por uma rede malha 2x3, comprovando os cinco ciclos de latência em cada roteador. O teste 2 apresenta o roteador reconfigurável RAPeC, onde é verificado o funcionamento correto nas três formas de chaveamento possíveis, UCS, BCS e PS. No teste 3, é importante verificar o correto funcionamento do árbitro presente no SwIX, bem como as informações sendo transmitidas entre hierarquias. Enfim, o teste 4 apresenta o mesmo teste que ocorre no teste 3, mas com o diferencial que a comunicação entre hierarquias permite três métodos de chaveamento.

6.3.2 Mapeamento

Nessa etapa, utiliza-se o *Floorplanning-Aware Design Space Exploration Framework*, aliado aos *benchmarks* em questão para realizar o mapeamento nas arquiteturas em experimentação, SoCIN com *pipeline*, MINoC, HiCIT e HASIN. Dividiu-se essas arquiteturas em dois conceitos de topologia, as baseadas em malha e as baseadas em hierarquia. Assim, obtiveram-se quatro mapeamentos, malha para aplicação NCS, malha para aplicação TVOPD, hierárquica para aplicação NCS e hierárquica para aplicação TVOPD. Os mapeamentos para a aplicação NCS estão apresentados na figura 6.6, enquanto que os mapeamentos para a aplicação TVOPD estão na figura 6.7.

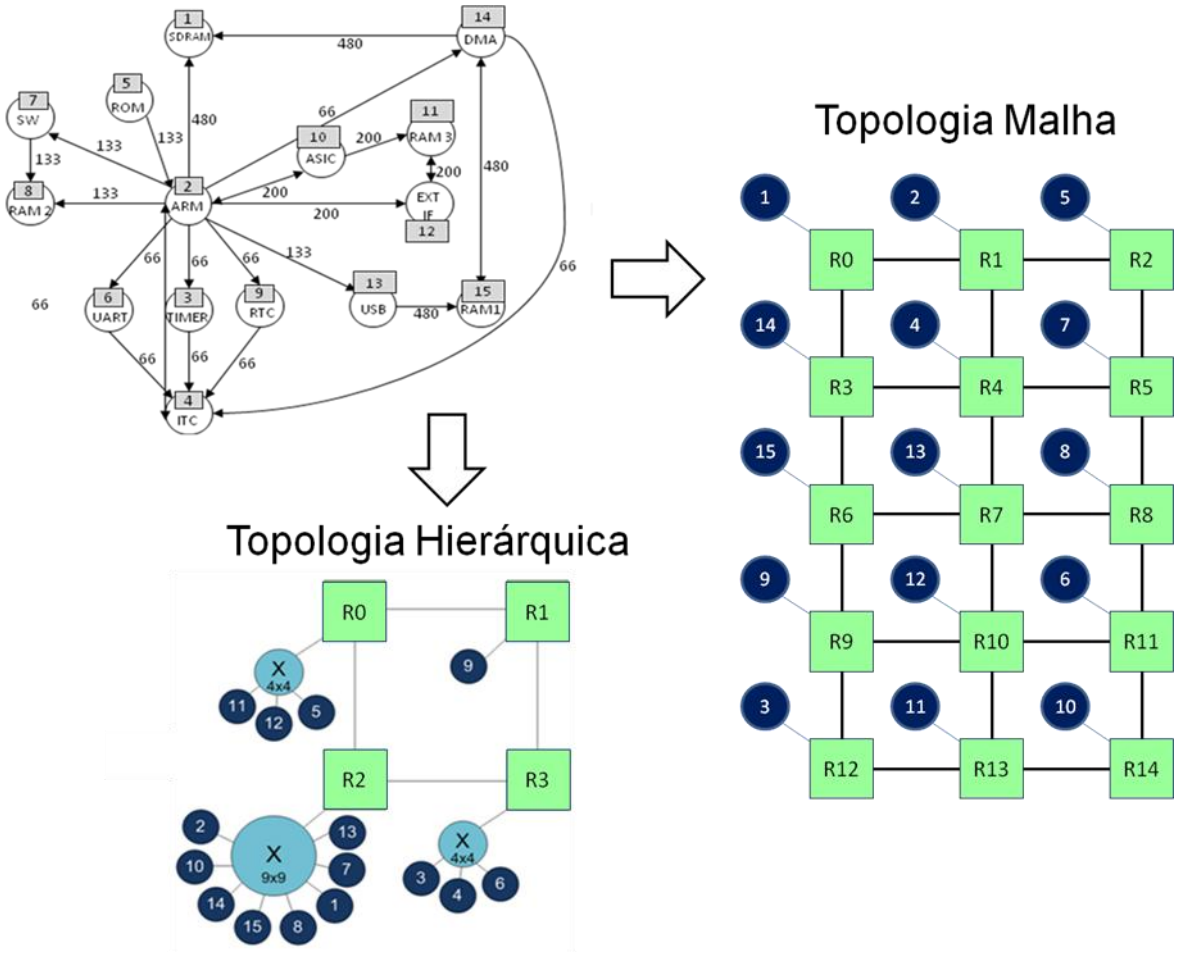


Figura 6.6 - Mapeamento da aplicação NCS para topologia malha (SoCIN e MINoC) e para topologia hierárquica (HiCIT e HASIN).

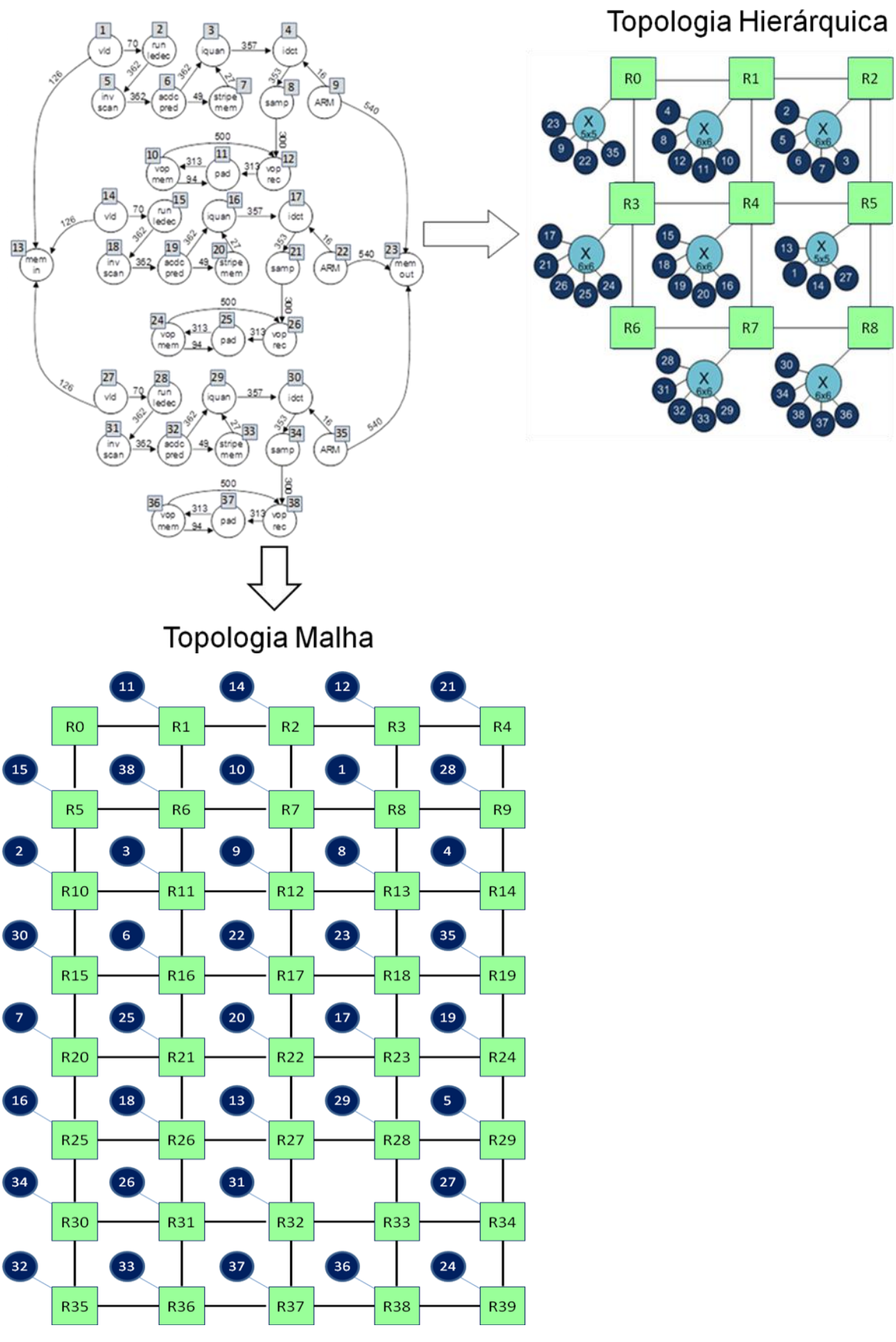


Figura 6.7 - Mapeamento da aplicação TVOPD para topologia malha (SoCIN e MINoC) e para topologia hierárquica (HiCIT e HASIN).

6.3.3 Simulação Sistema

Com a informação detalhada do comportamento funcional e temporal das arquiteturas unitárias (obtidas na subseção 6.3.1), foi possível configurar o *Simulation Framework* e realizar a simulação dos sistemas. Assim, o próximo passo era obter a quantidade de flits que deveriam ser enviadas em cada iteração para cumprir as taxas definidas nos benchmarks. Após esses cálculos, obtiveram-se os resultados de latência e vazão para os diferentes cenários, apresentados na subseção 6.4. Além disso, foram realizados testes aumentando a taxa de injeção de dados na rede pelos elementos de processamento, até chegarem à saturação do meio de interconexão.

6.3.4 Síntese Lógica e Extração dos Elementos Parasitas

Por fim, para obter as informações de custo do hardware de cada arquitetura é realizada a etapa de síntese lógica, contendo o sistema completo, para as duas aplicações alvo de experimentação. Para isso, são desenvolvidos dois novos arquivos *vhdl*, para descrever o topo do sistema no *benchmark*, ou seja, incluindo os núcleos de processamento e seguindo os mapeamentos da subseção 6.3.2. Como não há descrição em hardware dos núcleos é informado para a ferramenta que os núcleos são caixas pretas (*blackbox*) apenas para que seja possível considerar nos resultados a área referente a cada núcleo, o que impactará nos resultados finais. Além disso, são passadas as métricas de frequência operacional, para obter dados de potência, que nesse trabalho fixamos em um Gigahertz.

No entanto, a síntese lógica não considera informações físicas dos fios da arquitetura para realizar análises de área e potência. Para isso, foi realizada uma síntese física simplificada, que consistiu de um floorplanning automático, *placement* e roteamento. Após o roteamento, extraíram-se as características parasitas (capacitâncias e resistências) de cada fio na arquitetura. Com essa informação, introduziu-se novamente na ferramenta de síntese lógica, para gerar os relatórios de área e potência com a informação dos fios. A figura 6.8 apresenta a etapa após o roteamento realizado para a arquitetura HiCIT.

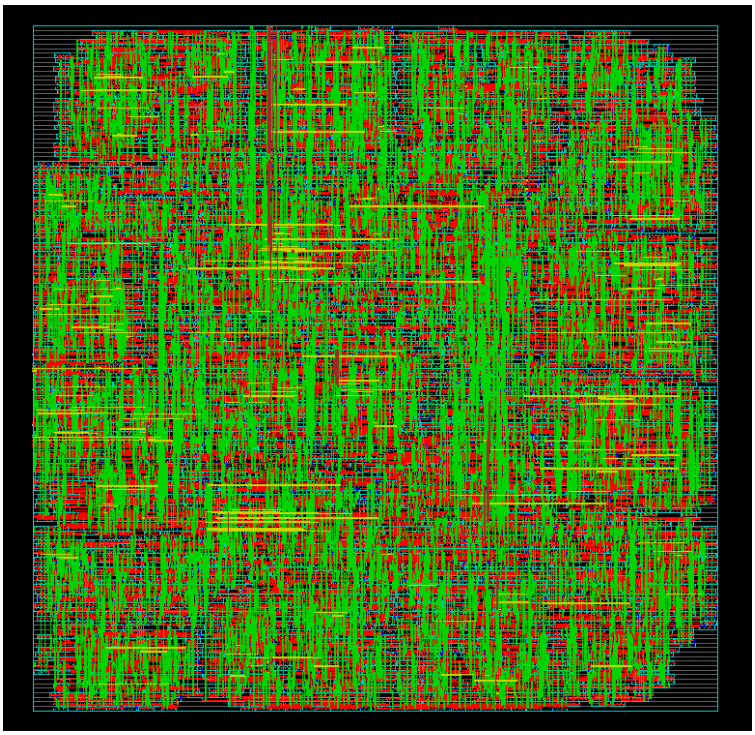


Figura 6.8 - Síntese física com roteamento em nove camadas de metais para o projeto HiCIT na aplicação NCS.

6.4 Resultados de Latência

Nessa subseção, apresentam-se os resultados de latência obtidos da simulação de sistema. Os resultados foram normalizados considerando os resultados da rede SoCIN, sendo apresentada a porcentagem de redução das arquiteturas propostas em relação à rede-em-chip convencional. A figura 6.9 demonstra a aplicação NCS, enquanto que a figura 6.10 representa a aplicação TVOPD.

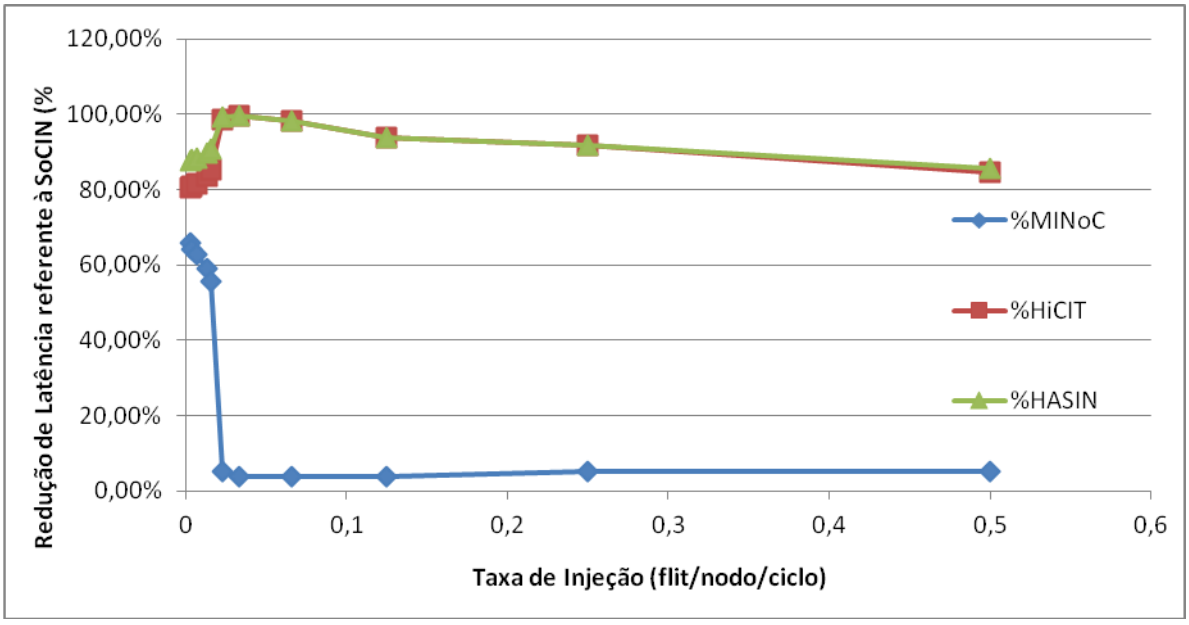


Figura 6.9 – Resultados de Latência para aplicação NCS.

Os dados da figura 6.9 permitem concluir que a solução reconfigurável permite um melhor desempenho que a rede-em-chip convencional, pois sempre que possível, realiza chaveamento por circuito, enviando dados em rajadas. No entanto as soluções hierárquicas apresentam um desempenho ainda maior e o mantém mesmo sob aumento da taxa de injeção na rede. Além disso, nos primeiros aumentos da taxa de injeção, o maior volume de dados pôde ser absorvido pela rede hierárquica sem complicações, enquanto que as outras duas soluções (SoCIN e MINoC) enfrentaram uma maior contingência, apresentando um pico de desempenho para HiCIT e HASIN no gráfico. O agrupamento considerando a afinidade de comunicação permite evitar os custos de latência da rede-em-chip, visto que a maior parte da comunicação se concentra dentro dos grupos. Pode-se notar também que a reconfiguração resultou em um pequeno acréscimo de desempenho à arquitetura HASIN, pois as poucas comunicações que necessitaram passar pela rede-em-chip puderam ser realizadas por chaveamento por circuito. A relação da diminuição de latência média, em porcentagem, em relação à solução convencional (rede-em-chip), está apresentada na tabela 6.4.

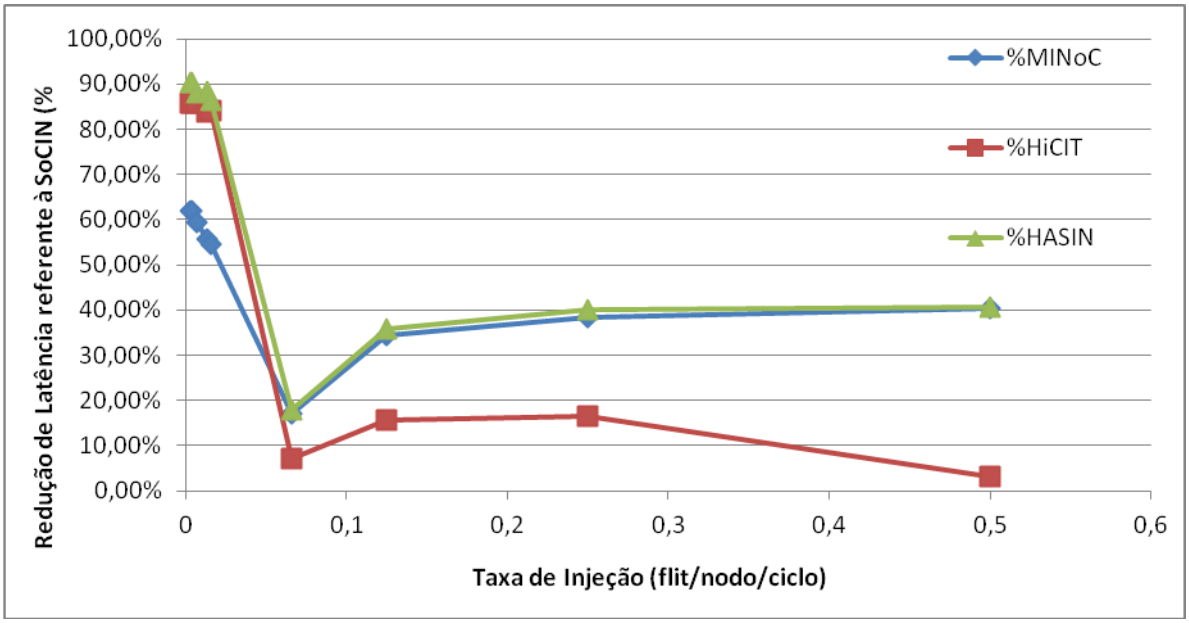


Figura 6.10 – Resultados de Latência para aplicação TVOPD.

Um comportamento semelhante da aplicação NCS pode ser observado na aplicação TVOPD na figura 6.10. Porém nesse caso, com o aumento inicial da taxa de injeção, obteve-se um alto tráfego entre grupos, onde a solução hierárquica obteve maiores prejuízos de latência, embora a estratégia reconfigurável pode se sobressair, mantendo a rede HASIN e MINoC com maiores ganhos. A relação da diminuição de latência média, em porcentagem, em relação à solução convencional (rede-em-chip), está apresentada na tabela 6.4.

Tabela 6.4 – Relação da redução na latência para as aplicações NCS e TVOPD.

Aplicação	**SoCIN**	**MINoC**		**HiCIT**		**HASIN**	
	Latência Média (ns)	Latência Média (ns)	Redução	Latência Média (ns)	Redução	Latência Média (ns)	Redução
NCS	2078,50	1974,90	**4,98%**	113,30	**94,55%**	109,20	**94,75%**
TVOPD	3009,36	1900,73	**36,84%**	2697,00	**10,38%**	1864,45	**38,04%**

Portanto, através da análise da tabela 6.4, pode-se confirmar o constatado através dos gráficos, onde observa-se a maior redução na latência para a aplicação NCS das arquiteturas hierárquicas e a maior redução para a aplicação TVOPD para as arquiteturas reconfiguráveis. Uma importante observação é o fato de que a rede HASIN se aproveitou de suas duas características (reconfiguração e hierarquia) para obter sempre o melhor desempenho dentre as propostas, demonstrando que consegue usufruir ambas as partes eficientemente.

6.5 Resultados de Vazão

Nessa subseção, apresentam-se os resultados de vazão, ou *throughput*, obtidos da simulação de sistema. Nesse caso, também foram normalizados os dados, de modo a obter os resultados comparados com a rede-em-chip SoCIN. A figura 6.11 demonstra a aplicação NCS, enquanto que a figura 6.12 representa a aplicação TVOPD.

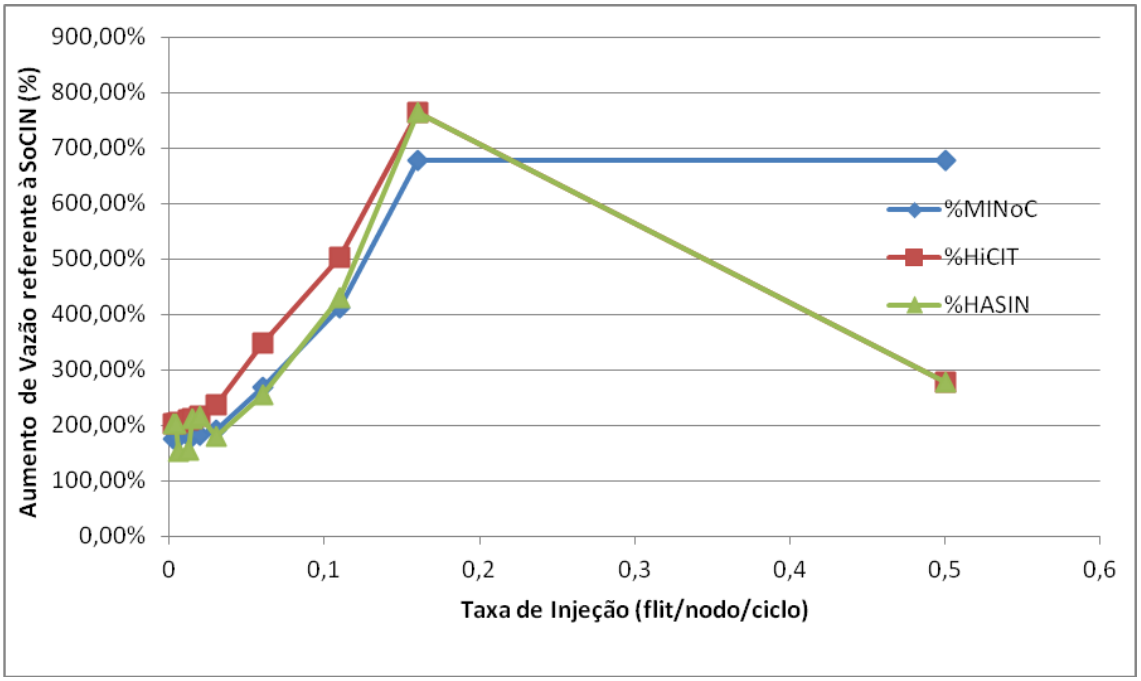


Figura 6.11 – Resultados de Vazão para aplicação NCS.

De acordo com a figura 6.11, as soluções propostas apresentam maior vazão de dados. Isso se justifica, pois todas exploram o conceito de chaveamento por circuito, onde ocorrem rajadas de comunicações, permitindo alta vazão, se não ocorrer muitas contigências na rede. Para taxas de injeção baixas, pode-se observar que a conexão direta promovida pela arquitetura SwIX resultou em uma vazão maior para as soluções hierárquicas, onde praticamente não houve diferença na utilização de reconfiguração. Como os elementos com as maiores taxas de comunicação foram agrupados juntos, essas comunicações não ocorreram em grande quantidade pela rede topo, não causando maiores diferenciações entre HiCIT e HASIN. No entanto, para taxas de injeção mais altas, obtém-se uma vazão melhor com a proposta reconfigurável, que se ajusta adequadamente com seus *buffers*, através do emprego de chaveamento por circuito com memorização ou do chaveamento por pacotes. A relação de vazão entre a solução convencional SoCIN e as arquituras propostas pode ser observado na tabela 6.5.

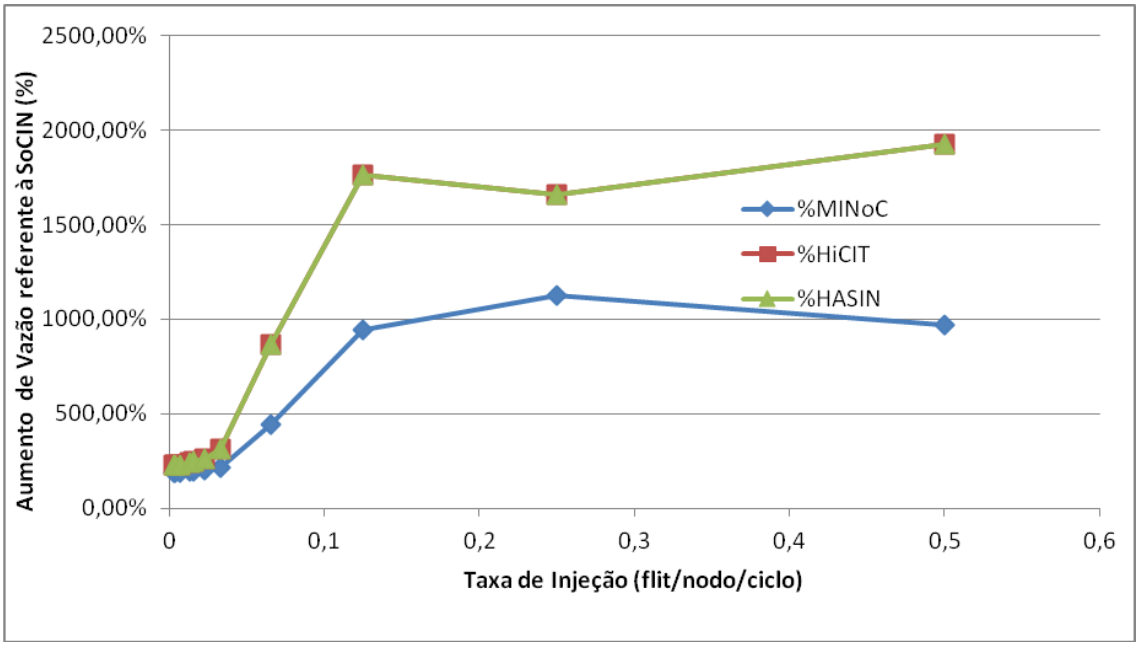


Figura - 6.12 – Resultados de Vazão para aplicação TVOPD.

Na aplicação da figura 6.12, TVOPD, é identificado claramente um aumento grande na contingência da rede, onde as soluções hierárquicas apresentam uma queda na vazão mais acentuada que na solução convencional. Isso se justifica principalmente pelo fato das soluções por circuito não utilizarem *buffers*, necessitando parar toda a comunicação durante contenções na rede. A relação de vazão entre a solução convencional SoCIN e as arquituras propostas pode ser observado na tabela 6.5.

Tabela 6.5 - Relação do aumento na vazão para as aplicações NCS e TVOPD.

Aplicação	**SoCIN**	**MINoC**		**HiCIT**		**HASIN**	
	Vazão Média	Vazão Média	Aumento	Vazão Média	Aumento	Vazão Média	Aumento
NCS	0,20	0,67	**2,31x**	0,72	**2,57x**	0,67	**2,28x**
TVOPD	0,19	0,68	**2,49x**	0,87	**3,48x**	0,87	**3,48x**

A tabela 6.5 apresenta informações de vazão para um sistema sem muita contingência na rede. Essa falta de conflitos na comunicação se deve ao mapeamento realizado, que procurou deixar os elementos com afinidade mais próximos. No entanto, para um cenário com muito congestionamento ou gargalo de comunicação, as arquiteturas que possuem estruturas de memorização terão vantagens em relação a vazão do sistema.

6.6 Resultados de Custos – Área e Potência

Após o trabalho de síntese lógica e extração dos parasitas para cálculo de influência dos fios, obtiveram-se os resultados de área e potência para todas as arquiteturas analisadas considerando as aplicações NCS e TVOPD. Primeiramente, têm-se os resultados para NCS, figura 6.13 e 6.14. Logo, apresentam-se os resultados para TVOPD, figura 6.15 e 6.16.

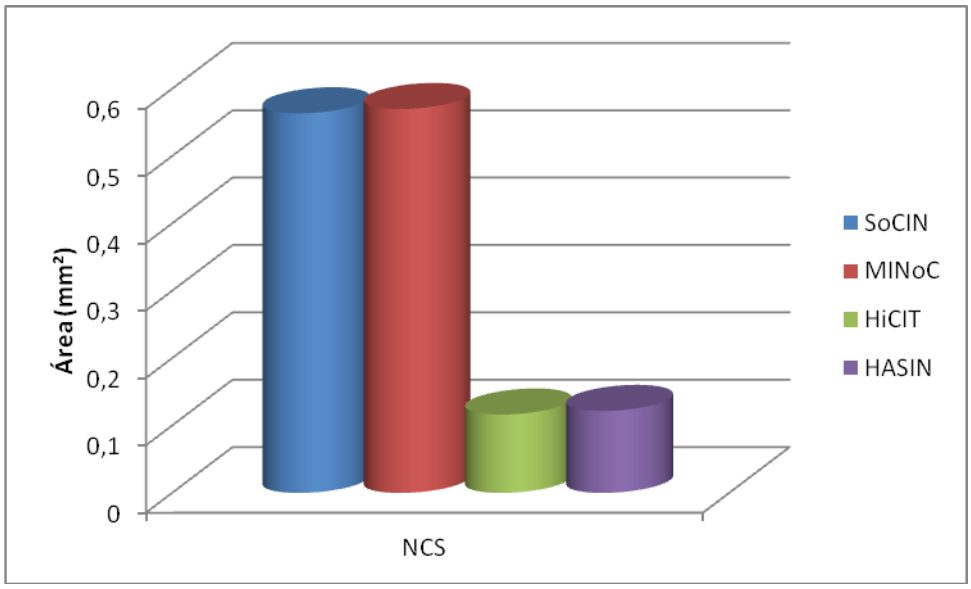


Figura 6.13 - Resultados e comparação de área para aplicação NCS.

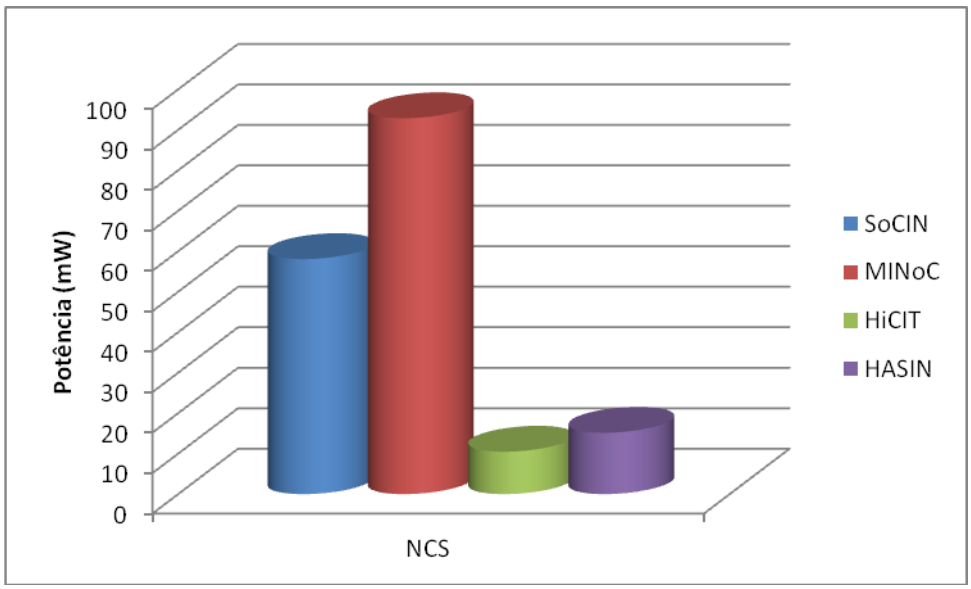


Figura 6.14 – Resultados e comparação de potência para aplicação NCS

Os resultados relativos à aplicação NCS apresentam que a solução reconfigurável tem um alto custo adicional de potência comparado à rede convencional, enquanto que as soluções hierárquicas possuem um custo muito menor que ela, tanto em potência como em área.

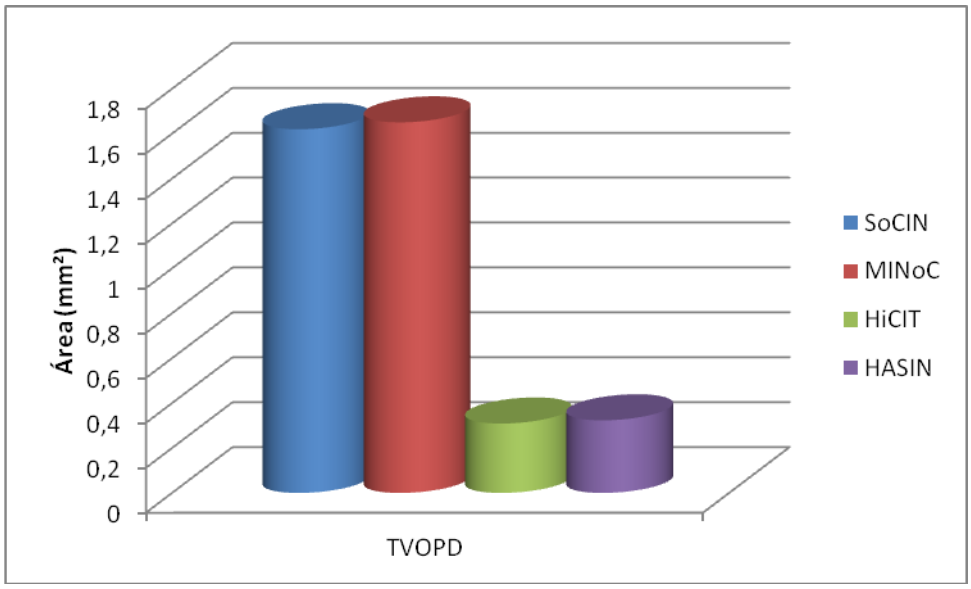


Figura 6.15 - Resultados e comparação de área para aplicação TVOPD

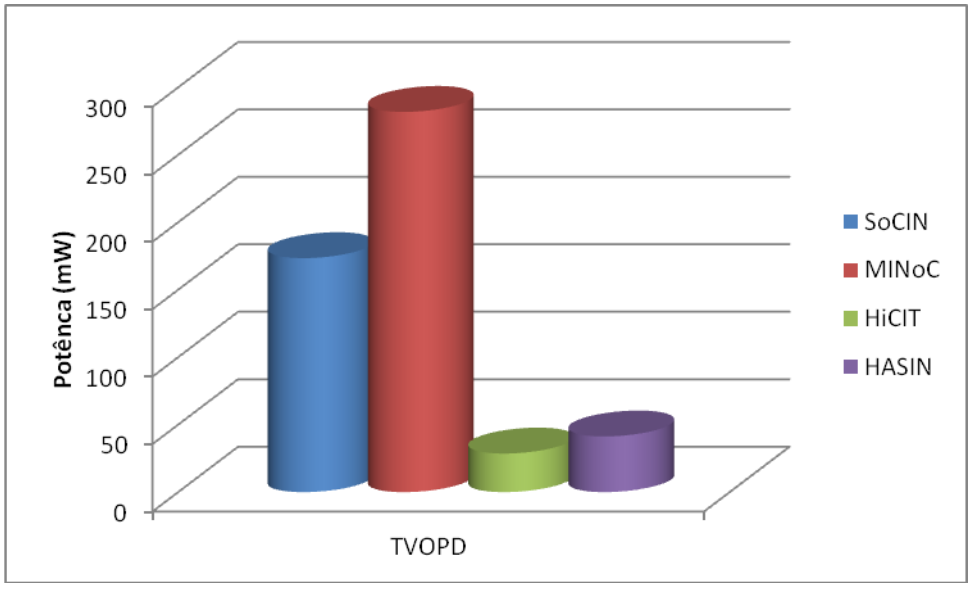


Figura 6.16 - Resultados e comparação de potência para aplicação TVOPD

O mesmo comportamento de custo pôde ser observado para a aplicação TVOPD. Todos os resultados estão relacionados em questão de proporção dos custos de área e potência nas tabelas 6.6 e 6.7 respectivamente.

Tabela 6.6 – Relação dos custos de área comparados à rede SoCIN

Aplicação	SoCIN	MINoC		HiCIT		HASIN	
	Área	Área	Aumento	Área	Redução	Área	Redução
NCS	0,56	0,57	**1,13%**	0,116326	**79,36%**	0,1217	**78,41%**
TVOPD	1,62	1,64	**1,12%**	0,309016	**80,92%**	0,323319	**80,04%**

Tabela 6.7 - Relação dos custos de potência comparados à rede SoCIN

Aplicação	SoCIN	MINoC		HiCIT		HASIN	
	Potência	Potência	Aumento	Potência	Redução	Potência	Redução
NCS	58,08	92,87	**59,91%**	10,48787	**81,94%**	15,16772	**73,88%**
TVOPD	173,44	259,59	**49,67%**	28,60753	**83,51%**	41,27872	**76,20%**

Os dados de área da tabela 6.6 confirmam que para adicionar inteligência ao circuito, ou seja, um mecanismo de reconfigurabilidade, implica em um aumento de área no sistema. No entanto, esse custo global ficou em menos de 2% comparado à rede convencional, tornando essa uma solução de pouca penalidade em questão de área. Por outro lado, a potência aumentou em quase 60%. Esse custo se justifica, pois os caminhos que definem os circuitos são puramente combinacionais e interligam fisicamente todos os elementos no sistema. Essa lógica realiza chaveamentos a todo o momento mesmo quando não utilizado devido a sua natureza combinacional.

As arquiteturas hierárquicas, como esperado, apresentaram baixo custo de área e potência, praticamente com 80% de redução. É interessante ressaltar que a solução hierárquica adaptativa tem um pequeno acréscimo de custo quando comparado à solução puramente hierárquica, mas mesmo assim, esses valores se apresentam na ordem de 70% menos de área e dissipação de potência, sendo também uma solução econômica.

6.7 Considerações

Nessa seção foram apresentados o ambiente, as metodologias e as ferramentas para experimentação das arquiteturas propostas. Seus resultados demonstraram que todas as propostas desenvolvidas apresentaram melhor desempenho comparadas à rede-em-chip convencional SoCIN. As soluções hierárquicas atingiram praticamente 90% menos de latência e um aumento de até 3x na vazão média. Em relação aos custos de área e potência, também se pode observar um baixo custo das soluções hierárquicas, onde há uma redução de até 80% de área e potência. Esses resultados realmente comprovam a eficiência da solução hierárquica, e demonstram que é possível aprimorá-la através da estratégia de reconfiguração, sob um baixo incremento de custo acrescentado. A solução reconfigurável MINoC apresentou melhorias de desempenho, mas sob um custo alto de dissipação de potência, em torno de 60%. No entanto, a solução que empregou reconfiguração e hierarquização possibilitou obter um custo de dissipação reduzido.

7 CONCLUSÕES E TRABALHOS FUTUROS

O presente trabalho de dissertação apresentou conceitos e propostas relacionadas a futuros sistemas multi processados. Inicialmente, a tendência de aumento na demanda das aplicações e a necessidade de executar múltiplas aplicações em paralelo, demandam cada vez mais a utilização de sistemas multiprocessados com maior poder de computação. Esses sistemas necessitam de inúmeros elementos de processamento heterogêneos interligados por uma rede de interconexão para obter uma maior eficiência, sob os pontos de vista de área e potência. Para sistemas atuais, a rede-em-chip é a solução de interconexão adequada, embora estudos demonstrem que para o futuro aumento no número de núcleos, o custo de uma rede convencional poderá limitar esses sistemas e projetos. Por isso, nesse trabalho foram apresentadas duas linhas de pesquisa que se mostram como soluções a esse impasse, reconfiguração e hierarquização.

Reconfiguração de arquiteturas permite criar um sistema que se adapte às requisições de desempenho das aplicações em tempo de execução. O contraponto é o custo desse controle e adaptabilidade. Por outro lado, a hierarquização de arquiteturas permite atender os requisitos mínimos das aplicações sob um custo muito menor de hardware. No entanto, a solução é eficiente somente se a aplicação se adapta à topologia projetada. Contudo, ambas permitem alto desempenho para futuras aplicações, diferenciadas pelo foco de classe de aplicação, mais específicas ou mais genéricas. Assim, a presente dissertação realizou um estudo em cada área, onde foram desenvolvidas propostas para investigar e entender seus fundamentos e aplicações (MINoC e HiCIT). O resultado desses estudos foi a proposta de uma arquitetura que visa a sinergia de ambas as técnicas (HASIN).

Resultados sobre todas as propostas arquiteturais revelaram que tanto a solução hierárquica quanto a reconfigurável aumentam o desempenho do sistema comparado à solução convencional de rede-em-chip. Mas, a solução hierárquica demonstrou um custo muito menor de hardware (área e potência), na ordem de 80% para ambas, enquanto a rede reconfigurável aumentou o custo de área em 1,13% e potência em 60%. Em sequência, a solução que emprega adaptabilidade à hierarquia resulta em um pouco mais de desempenho, sob um pouco mais de custos quando comparada à puramente hierárquica. Porém, identifica-se grande potencial para essa nova arquitetura, visto que ainda se manterá eficiente mesmo com variações no padrão de comunicação ou atualizações das aplicações envolvidas.

Portanto, este trabalho de dissertação conclui que a metodologia de projeto de redes-em-chip hierárquicas adaptativas é uma linha mais apropriada para as questões

mencionadas pela academia como desafios para sistemas futuros. Sua adaptabilidade às novas aplicações e sua organização em hierarquia permite aumentar desempenho com custo baixo.

7.1 Trabalhos Futuros

.

No que refere aos trabalhos futuros, o desenvolvimento da arquitetura HASIN resultou em diferentes necessidades e possibilidades de pesquisa. Resultante da integração direta de reconfiguração e hierarquização, a rede HASIN apresenta um método simples de adaptabilidade organizado em hierarquia. Futuras propostas podem incorporar métodos mais elaborados para uma integração mais eficiente.

Em curto prazo, pretende-se realizar as sínteses físicas dessa rede para diferentes aplicações e identificar os tamanhos exatos dos fios entre cada roteador e inserir essa informação na rede, diminuindo ainda mais os custos de latência presentes nessa arquitetura, pois utilizaria o chaveamento com armazenamento eficientemente, por utilizar maior precisão na decisão. Outro experimento pretende comprovar que o custo da adaptabilidade da HASIN é justificável, demonstrando que mesmo em diferentes classes de aplicações, para o qual não foi projetado, o desempenho mínimo é garantido.

Em longo prazo, pretende-se pesquisar sobre as diferentes características explorando as topologias e arranjos hierárquicos, mecanismos de reconfiguração e monitoramento e meios híbridos para a interconexão. A possibilidade de utilizar diferentes tecnologias, como 3D ou óptica, permite projetar um cenário coerente de arquiteturas para futuras gerações de sistemas integrados multi processados.

REFERÊNCIAS

AMBA Specification Rev2.0, ARM Ltd, 1999.

ANDREWS, J; BAKER, N. Xbox 360 System Architecture. IN: IEEE MICRO, vol 26 n2, p.25-37, Mar./Apr 2006.

ATIENZA, D.; ANGIOLINI, F.; MURALI, S.; PULLINI, A.; BENINI, L.; DE MICHELI, G. Network-on-Chip design and synthesis outlook. IN: **Integration, the VLSI Journal**, vol 41, n3, p.340-359, May 2008.

BENINI, L.; DE MICHELI, G. Networks on Chips: a New SoC Paradigm. IN: **IEEE Computer Magazine**, p. 70-78, 2002.

BENINI, L.; DE MICHELI, G. Powering Networks on Chips. IN: 14TH INTERNATIONAL SYMPOSIUM ON SYSTEMS SYNTHESIS. Proceedings of the 14th International Symposium on Systems Synthesis, Montreal, p. 33-38, 2001.

BERTOZZI, D.; JALABERT, A.; SRINIVASA, M.; TAMHANKAR, R.; STERGIOU, S.; BENINI, L.; DE MICHELI, G. NoC syntesis flow for customized domain specific multiprocessor systems-on-chip. IN: IEEE TRANSACTIONS ON PARALLEL DISTRIBUTED SYSTEMS, vol. 16 n2, p. 113–129, Feb. 2005.

BHUYAN, L. N. Performance of multiprocessor interconnections. IN: **Computer Magazine**, vol.22 n2, p.25-37, Feb. 1989.

BJERREGAARD, T.; MAHADEVAN, S. A Survey of Research and Practice of NoC. IN: ACM COMPUTING SURVEYS, USA, 2006.

BOBDA, C. et al. DyNoC: A Dynamic Infrastructure for Communication in Dynamically Reconfigurable Devices. IN: INTERNATIONAL CONFERENCE ON FIELD PROGRAMMABLE LOGIC AND APPLICATIONS, p. 153-158, 2005.

BOLOTIN, E. et al. QNoC: QoS Architecture and Design Process for Network on Chip. IN: **Journal of Systems Architecture**, v.50, n.2, p. 1-24, 2004.

CHANG, K.C, SHEN, J.S., CHEN, F.T. Evaluation and Design Trade-offs Between Circuit-Switched and Packet-Switched NOCs for Application-Specific SOCs. IN: DESIGN AUTOMATION CONFERENCE – DAC, p. 143-148, 2006.

DALLY, W.; TOWLES, B. Route Packets, Not Wires: On-Chip Interconnection Networks. IN: DESIGN AUTOMATION CONFERENCE – DAC, p. 684-689, 2001.

DALLY, W. J.; TOWLES, B. Principles and Practices of Interconnection Networks. Morgan Kauffmann Publishers, 2004, 550p.

DAS, R.; EACHEMPATI, S.; MISHRA, A.; NARAYANAN, V.; DAS, C. Design and Evaluation of a Hierarchical On-Chip Interconnect for Next-Generation CMPs. IN HPCA, pp. 175–186, 2009.

DUATO, J.; YALAMANCHILI, S.; NI, L. Interconnection Networks. IN: **IEEE Computer Society Press**, 1997, 515 p.

DUATO, J. A New Theory of Deadlock-Free Adaptive Routing in Wormhole Networks. IN: IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, p. 1320-1331, 1993.

DUATO, J., A Necessary and Sufficient Condition for Deadlock-Free Adaptive Routing in Wormhole Networks. IN: IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, p. 1055-1067, 1995.

DUATO, J.; YALAMANCHILI, S.; NI, L. Interconnection Networks. Elsevier Science, 2002, 600 p.

FLYNN, M. J. Some Computer Organizations and Their Effectiveness. IN: IEEE TRANSACTIONS ON COMPUTERS, v.C-21, n.9, p.948-960, Set. 1972.

FREITAS, H.C.; NAVAUX, P. A High-Throughput Multi-cluster NoC Architecture. IN: 11TH IEEE INTERNATIONAL CONFERENCE ON COMPUTATIONAL SCIENCE AND ENGINEERING, p. 56-63, July 2008.

FREITAS, H.; ALVES, M.; NAVAUX, P. NoC e NUCA: Conceitos e Tendências para Arquiteturas de Processadores Maný-Core. IN: 9ª ESCOLA REGIONAL DE ALTO DESEMPENHO ARQUITETURAS MULTICORE. Disponível em: <http://bibliotecadigital.sbc.org.br/download.php?paper=2436>. Acesso em 26 out. 2012.

GRECU, C.; JONES, M.; IVANOV, A.; SALEH, R. Performance evaluation and design trade-offs for networks-on-chip interconnect architectures. IN: IEEE TRANSACTIONS ON COMPUTERS, vol 54 n8, p.1025-1040, Aug, 2005.

GUERRE, A.; VENTROUX, N.; DAVID, R.; MERIGOT, A. Hierarchical Network-on-Chip for Embedded Many-Core Architectures. IN 4TH ACM/IEEE INTERNATIONAL SYMPOSIUM ON NETWORKS-ON-CHIP (NOCS), p.189-196, May 2010.

GUERRIER, P.; GREINER, A. A Generic Architecture for On-Chip Packet-Switched Interconnections. IN: DESIGN AUTOMATION AND TEST IN EUROPE - DATE, p. 250–256, 2000.

HAIBO Z., PANDE .P, GRECU C. Performance Evaluation of Adaptive Routing Algorithms for achieving Fault Tolerance in NoC Fabrics. IN: IEEE INTERNATIONAL CONFERENCE ON APPLICATION-SPECIF SYSTEMS, ARCHITECTURES AND PROCESSORS, ASAP, p. 42-47, 2007.

HOLLSTEIN, T.; LUDEWIG, R.; ZIMMER, H.; MAGER, C.; HOHENSTERN. S.; GLESNER, M. Hinoc: A Hierarchical Generic Approach for on-Chip Communication, Testing and Debugging of SoCs. IN: VLSI-SOC: FROM SYSTEMS TO CHIPS. IFIP International Federation for Information Processing, Vol. 200/2006, p. 39-54, 2006.

ITRS – International Technology Roadmap for Semiconductors. Disponível em: <www.itrs.net>. Acesso em 26 out. 2011.

JERGER, N.; PEH, L.; LIPASTI, M. Circuit-Switched Coherence. IN: ACM/IEEE INTERNATIONAL SYMPOSIUM ON NETWORKS-ON-CHIP (NOCS), pp. 193-202, 2008.

JERRAYA, A.; TIMA; TENHUNEN, H.; WOLF, W. Guest Editors Introduction: Multiprocessor Systems-On-Chip". IN: **IEEE COMPUTER MAGAZINE**, vol 38 n7 p.36-40, July, 2005.

KIM, D. et al. NIUGAP: Low Latency Network Interface Architecture with Gray Code for Networks-on-Chip. IN: IEEE INTERNATIONAL SYMPOSIUM ON CIRCUITS AND SYSTEMS - ISCAS, p. 3901-3905, 2006.

KREUTZ, M.; Método para a otimização de plataformas arquiteturais para sistemas multiprocessados heterogêneos. Tese (Doutorado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre, 2005.

LEE, S.; LEE, C.; LEE, H. A New Multi-Channel On-Chip-Bus Architecture for System-on-Chips. IN: IEEE INTERNATIONAL SOC CONFERENCE, Proceedings of IEEE International SOC Conference, p.305–308, 2004.

MATOS, D.; CONCATTO, C.; KOLOGESKI, A.; KREUTZ, M.; CARRO, L.; KASTENSMIDT, F.; SUSIN, A. Adaptive Router Architecture Based on Traffic Behavior Observability. IN: INTERNATIONAL WORKSHOP ON NETWORKS-ON-CHIP ARCHITECTURES WORKSHOP - NoCARC, 2009b.

MATOS, D.; Interfaces Parametrizáveis para Aplicações Interconectadas por Rede-em-Chip. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre, 2010.

MATOS, D.; PALERMO, G.; ZACCARIA, V.; REINBRECHT, C.; SUSIN, A.; SILVANO, C.; CARRO, L. Floorplanning-Aware Design Space Exploration for Application-Specific Hierarchical Networks-on-Chip. IN: INTERNATIONAL WORKSHOP ON NETWORKS-ON-CHIP ARCHITECTURES WORKSHOP - NoCARC, 2011.

MELLO, A. Qualidade de Serviço em Redes Intra-Chip - implementação e avaliação sobre a rede HERMES. Dissertação (Mestrado em Ciência da Computação) – Faculdade de Informática, PUCRS, Porto Alegre.

MING L., QING-AN Z. e WEN-BEN J. DyXY - A Proximity Congestion-Aware Deadlock-Free Dynamic Routing Method for Network on Chip. IN: DESIGN AUTOMATION CONFERENCE - DAC, p. 849-852, 2006.

MODARRESSI, M.; TAVAKKORL, A.; SARBAZI-AZAD, H. Virtual Point-to-Point Connections for NoCs. IN: IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS (TCAD), vol. 29, no. 6, pp. 855-868, 2010.

MODARRESSI, M.; AZAD, H.; ARJOMAND, M. A Hybrid Packet-Circuit Switched On-Chip Network Based on SDM. IN: DESIGN AUTOMATION AND TEST IN EUROPE (DATE), pp. 566 – 569, 2009.

MODARRESSI, M.; TAVAKKOL, A.; SARBAZI-AZAD, H. Application-Aware Topology Reconfiguration for On-Chip Networks. IN: IEEE TRANSACTIONS VERY LARGE SCALE INTEGRATION SYSTEMS (VLSI), vol. 19, issue 11, p. 2010 – 2022, 2011.

MÖLLER, L. et al. Infrastructure for Dynamic Reconfigurable Systems: Choices and Trade-offs. IN: SYMPOSIUM ON INTEGRATED CIRCUITS AND SYSTEMS DESIGN - SBCCI, MG, Brasil, p. 44-49, 2006.

MORI, K.; TOSHIBA CORP.; YAMADA, H.; TAKIZAWA, S. System on Chip Age. IN: INTERNATIONAL SYMPOSIUM ON VLSI TECHNOLOGY, SYSTEMS AND APPLICATIONS, Proceedings of Technical Papers of 1993 International Symposium on VLSI Technology, Systems, and Applications, pp; K15 - K20. 1993.

MURALI, S. et al. Synthesis of networks on chips for 3d systems on chips. IN: ASP-DAC, pp. 242-247, 2009.

OGRAS, U.; MARCULESCU, R. Application-Specific Network-on-Chip Architecture Customization via Long-Range Link Insertion. IN: IEEE/ACM INTERNATIONAL CONFERENCE ON COMPUTER-AIDED DESIGN, p. 246-253, 2005.

PASRICHA, S.; DUTT, N.; On-Chip Communication Architectures: System on Chip Interconnect. San Francisco: Editora Morgan Kaufmann, p. 439 -466, 2008.

PATT, Y. N.; HWU, W. M.; SHEBANOW, M. HPS, a new microarchitecture: rationale and introduction. IN: **ACM SIGMICRO Newsletter**, vol 16 n4, p.103-108, Dec. 1985.

PATTERSON, D.; HENNESSY, J. L. Computer Architecture: A Quantitative Approach. Kauffmann Publishers, 1996, 760 p.

PREDICTIVE TECHNOLOGY MODEL (PTM), Disponível em: <http://ptm.asu.edu/>. Acesso em 10 jul.2012.

PUTTMANN, C.; NIEMANN, J. C.; PORRMANN, M.; RUCKERT, U. GigaNoC - A Hierarchical Network-on-Chip for Scalable Chip-Multiprocessors. IN: 10th EUROMICRO CONFERENCE ON DIGITAL SYSTEM DESIGN ARCHITECTURES, METHODS AND TOOLS, p. 495-502, Aug. 2007.

RALFHILL, T. R. Ambric's New Parallel Processor. Microprocessor Report, October 2006.

REGO, R. Projeto e Implementação de uma Plataforma MP-SoC usando SystemC. Dissertação (Mestrado em Sistemas e Computação), UFRN, Natal, 144 pag., 2006.

RYU, K.; SHIN, E.; MOONEY, V. A Comparison of Five Different Multiprocessor SoC Bus Architectures. IN: EUROMICRO SYMPOSIUM DIGITAL SYSTEMS DESIGN, Proceedings of Euromicro Symposium Digital Systems Design, p. 202-209, 2001.

SARBAZI-AZAD, H.; MACKENZIE, L. M.; OULD-KHAOUA, M. The Effect of the Number of Virtual Channels on the Performance of Wormhole-Routed Mesh Interconnection Networks. IN: 16TH ANNUAL UK PERFORMANCE ENGINEERING WORKSHOP (UKPEW), p.95-102, 2000.

SAKURAI T. Approximation of wiring delay in mosfet lsi. IN: **IEEE Journal of Solid-State Circuits**, volume 18, pages 418-426, 1983.

SRINIVASAN, K; CHATHA, K. S. A low complexity heuristic for design of custom network-on-chip architectures. IN: DESIGN AUTOMATION AND TEST IN EUROPE CONFERENCE, Proceedings of Design Automation and Test in Europe Conference, vol. 1, p. 1–6, 2006.

STENSGAARD, M.; SPARSO, J. ReNoC: A Network-on-Chip Architecture with Reconfigurable Topology. IN: ACM/IEEE INTERNATIONAL SYMPOSIUM ON NETWORKS-ON-CHIP (NOCS), p. 55 – 64, 2008.

TANENBAUM, A. S. Organização Estruturada de Computadores. Rio de Janeiro, RJ: Livros Técnicos e Científicos Editora, 2001.

Tilera Corporation. "TILE64™ Processor". Disponível em: http://www.tilera.com/pdf/ProBrief_Tile64_Web.pdf. Acesso em Agosto 2007.

TINO, A.; KHAN, G. "Power and Performance Tabu Search Based Multicore Network-on-Chip Design". IN: INTERNATIONAL CONFERENCE ON PARALLEL PROCESSING WORKSHOP, pp. 74-81, 2010.

TOPOL, A. W. et al., Three-Dimensional Integrated Circuits. IBM J. Research and Development, Vol. 50, No. 4/5, July-Sept. 2006, pp. 491-506.

VANGAL, S.; HOWARD, J.; RUHL, G.; DIGHE, S.; WILSON, H.; TSCHANZ, J.; FINAN, D.; IYER, P.; SINGH, A.; JACOB, T.; JAIN, S.; VENKATARAMAN, S.; HOSKOTE, Y.; BORKAR, N. An 80-Tile 1.28TFLOPS Network-on-Chip in 65nm CMOS. IN: IEEE INTERNATIONAL SOLID-STATE CIRCUITS CONFERENCE (ISSCC), Fevereiro 2007. pp. 5-7.

VELLANKI P. et al. Quality-of-Service and Error Control Techniques for Network-on-Chip Architectures. IN: GREAT LAKES SYMPOSIUM ON VLSI. Boston, p. 45-50, 2004.

YOON, Y.; CONVER, N.; PETRACCA, M.; CARLONI, L. Virtual Channels vs. Multiple Physical Networks: a Comparative Analysis. IN: ACM IEEE DESIGN AUTOMATION CONFERENCE (DAC), p. 162- 165, 2010.

WANG, H.; ZHU, X.; PEH, L.; MAKIL, S. Orion: A Power-Performance Simulator for Interconnection Networks. IN: 35th ANNUAL IEEE/ACM INTERNATIONAL SYMPOSIUM ON MICROARCHITECTURES (MICRO-35), p. 294-305, Nov. 2002.

ZEFERINO, C. et al. A Study on Communication Issues for Systems-on-Chip. IN: 15th SYMPOSIUM ON INTEGRATED CIRCUITS AND SYSTEMS DESIGN, p. 121 – 126, 2002.

ZEFERINO, C. Redes-em-Chip: Arquiteturas e Modelos para Avaliação e Área e Desempenho. 2003. Tese (Doutorado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.

ZHENG, L.; JUEPING, C.; MING D.; LEI, Y.; ZAN, L. Hybrid Communication Reconfigurable Network on Chip for MPSoC. IN: 24th IEEE INTERNATIONAL CONFERENCE ON ADVANCED INFORMATION NETWORKING AND APPLICATIONS (AINA) 2010, p. 356-361, 2010.