UNIVERSIDADE FEDERAL DO CEARÁ

CENTRO DE TECNOLOGIA

DEPARTAMENTO DE ENGENHARIA DE TELEINFORMÁTICA

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE TELEINFORMÁTICA

MESTRADO ACADÊMICO EM ENGENHARIA DE TELEINFORMÁTICA

ALEXANDRE ALMEIDA DA SILVA

REFLECT3D: UM ALGORITMO DE ROTEAMENTO ADAPTATIVO E TOLERANTE A FALHAS PARA NOCS 3D PARCIALMENTE CONECTADAS

FORTALEZA

2022

ALEXANDRE ALMEIDA DA SILVA

REFLECT3D: UM ALGORITMO DE ROTEAMENTO ADAPTATIVO E TOLERANTE A FALHAS PARA NOCS 3D PARCIALMENTE CONECTADAS

Dissertação apresentada ao Curso de Mestrado Acadêmico em Engenharia de Teleinformática do Programa de Pós-Graduação em Engenharia de Teleinformática do Centro de Tecnologia da Universidade Federal do Ceará, como requisito parcial à obtenção do título de mestre em Engenharia de Teleinformática. Área de Concentração: Sinais e Sistemas

Orientador: Prof. Dr. Jarbas Nunes Aryel Silveira

FORTALEZA

2022

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

S578r Silva, Alexandre Almeida da.
REFLECT3D: Um Algoritmo de Roteamento Adaptativo e Tolerante a Falhas para NoCs 3D Parcialmente Conectadas / Alexandre Almeida da Silva. – 2023.
61 f. : il. color.

Dissertação (mestrado) – Universidade Federal do Ceará, Centro de Tecnologia, Programa de Pós-Graduação em Engenharia de Teleinformática, Fortaleza, 2023.
Orientação: Prof. Dr. Jarbas Nunes Aryel Silveira.

1. Algoritmo de roteamento tolerante a falhas. 2. NoC 3D. 3. NoC parcialmente e verticalmente conectado. 4. Through-silicon vias. 5. MPSoCs. I. Título.

CDD 621.38

ALEXANDRE ALMEIDA DA SILVA

REFLECT3D: UM ALGORITMO DE ROTEAMENTO ADAPTATIVO E TOLERANTE A FALHAS PARA NOCS 3D PARCIALMENTE CONECTADAS

Dissertação apresentada ao Curso de Mestrado Acadêmico em Engenharia de Teleinformática do Programa de Pós-Graduação em Engenharia de Teleinformática do Centro de Tecnologia da Universidade Federal do Ceará, como requisito parcial à obtenção do título de mestre em Engenharia de Teleinformática. Área de Concentração: Sinais e Sistemas

Aprovada em: 28/07/2022

BANCA EXAMINADORA

Prof. Dr. Jarbas Nunes Aryel Silveira (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Dr. Cesar Albenes Zeferino
Universidade do Vale do Itajaí (UNIVALI)

Prof. Dr. Giovani Cordeiro Barroso
Universidade Federal do Ceará (UFC)

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

Dedico este trabalho à minha família e esposa, pessoas que fizeram de tudo para que eu chegasse onde cheguei.

AGRADECIMENTOS

Agradeço primeiramente a Deus, que iluminou meu caminho durante essa jornada, me dando saúde e força para superar as dificuldades.

À minha esposa, Bruna Alcântara Almeida, pelo amor, paciência, dedicação e companheirismo em todos os momentos.

À minha família, por sua capacidade de acreditar e investir em mim. Meus irmãos, semrpe presentes e em constante apoio. Pai, sua dedicação, exemplo, conselhos e amor foi o que deu, em alguns momentos, a esperança para seguir.

Ao meu orientador, Prof. Jarbas, pelo acompanhamento e estreitamento da relação professor-aluno e exemplo de profissional bem como pelo apoio, incentivo, sugestões e comentários durante a supervisão dos meus estudos.

Ao meu co-orientador, Prof. Alexandre Coelho, pelo apoio, incentivo, sugestões e tempo dedicado para me ajudar durante meus estudos.

Aos meus amigos da Universidade Federal do Ceará pela amizade e pelos momentos de descontração e estudo.

E a todos que direta ou indiretamente fizeram parte da minha formação, o meu muito obrigado.

“Não julgue cada dia pela colheita que você obtém, mas pelas sementes que você planta.”

(William Arthur Ward)

RESUMO

Os benefícios combinados dos esquemas Tridimensional (3D) e as Redes em Chip (Network-on-Chip - NoC) permitem projetar um sistema de alto desempenho em uma área limitada do chip. As principais vantagens das NoCs 3D são as reduções médias do comprimento do fio e do atraso, resultando em uma menor dissipação de energia e um maior desempenho. No entanto, as NoCs 3D sofrem com alguns problemas de confiabilidade, como a variabilidade de fabricação dos circuitos integrados 3Ds. O baixo rendimento da conexão vertical prejudica o projeto de pilhas de matrizes 3D com muitos *Through Silicon Via* (TSV). As NoCs conectadas verticalmente e parcialmente têm ganhado relevância para reduzir o número de links de TSV. Este trabalho apresenta o Reflect3d, um algoritmo de roteamento adaptável, escalável e altamente resiliente que visa as NoCs 3D parcialmente e verticalmente conectados para rotear pacotes sob tais condições. O Reflect3d separa canais virtuais em quatro redes virtuais refletidas para procurar conexões verticais de forma dinâmica e progressiva, garantindo a entrega de pacotes, desde que um TSV saudável conectando todas as camadas esteja disponível em qualquer lugar da rede. Os resultados experimentais mostram que o Reflect3d pode garantir melhor resultado em termos de latência, alcançando caminhos mais válidos e melhor custo-benefício quando comparado com outros trabalhos relacionados. Além disso, a síntese de hardware realizada usando uma biblioteca de tecnologia comercial de 28 nm mostra poucas sobrecargas de área e energia em comparação com outros algoritmos de roteamento de última geração.

Palavras-chave: Algoritmo de roteamento tolerante a falhas, NoC 3D, NoC parcialmente e verticalmente conectado, TSV, Through-silicon vias

ABSTRACT

The combined benefits of the Three-Dimensional (3D) and Network-on-Chip (NoC) schemes enable designing a high-performance system in a limited chip area. The 3D-NoC major advantages are the average reductions of wire length and delay, resulting in lower power dissipation and higher performance. However, 3D-NoCs suffer from some reliability issues, such as the 3D-IC manufacturing variability. The low yield of vertical connection harms the design of 3D die stacks with many Through Silicon Via (TSV). Vertically-Partially-Connected NoCs have been gaining relevance to reduce the number of TSV links. This work introduces Reflect3d, an adaptive, scalable, and highly resilient routing algorithm targeting partially and vertically connected 3D-NoCs to route packets under such conditions. Reflect3d separates virtual channels in four reflected virtual networks to search for vertical connections dynamically and progressively, guaranteeing packet delivery as long as one healthy TSV connecting all layers is available anywhere in the network. The experimental results show that Reflect3d can ensure high performance in terms of latency, achieving more valid paths and better cost-effectiveness when compared to related work. Furthermore, the hardware synthesis performed using a commercial 28nm technology library shows few area and power overheads compared to other state-of-the-art routing algorithms.

Keywords: Fault-tolerant routing algorithm. 3D NoC. Vertically-partially-connected NoC. Through-silicon vias.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

NoCs	Network-on-Chip
PEs	Elementos de Processamento
SoCs	System-on-Chip
TSV	Through-Silicon-Via

SUMÁRIO

1 INTRODUÇÃO

O forte avanço da microeletrônica possibilitou o crescimento de sistemas completos em uma única pastilha de silício, chamados de System-on-Chip (SoCs). Diversas aplicações com SoCs são comumente utilizados em produtos eletrônicos de consumo diário e fazem parte da era de arquiteturas multi-core devido ao número crescente de funções que um dispositivo pode suportar. Um System-on-Chip (SoC) pode conter vários Elementos de Processamento (PEs) interconectados por meio de barramentos de comunicação (CHEN *et al.*, 2017).

No entanto, conforme o número de PE aumenta, a interconexão baseada em barramento torna-se um gargalo para o projeto de SoCs de alto desempenho (DALLY; TOWLES, 2001). Neste sentido, as redes de interconexões em chip, Network-on-Chip (NoCs) surgiram como uma solução alternativa aos barramentos de comunicação tradicionais (CHEN *et al.*, 2016). Basicamente, uma NoC é constituída por roteadores, links e elementos de processamento. A partir daí, é configurada uma rede entre os roteadores que utiliza conceitos de redes de computadores e sistemas distribuídos para as conexões intrachip dos PEs. As NoCs permitem uma maior escalabilidade e um desempenho necessário para diversos SoCs com alto grau de paralelismo (RIJPKEMA *et al.*, 2003).

A Rede no Chip bidimensional, também denominada *Network-on-Chip* de duas dimensões (NoC 2D), surgiu como uma infraestrutura promissora para comunicações no chip, superando essa limitação dos barramentos ao fornecer alta escalabilidade, alta largura de banda e menor consumo de energia (RIJPKEMA *et al.*, 2003). No entanto, o atraso, o congestionamento e o consumo de energia aumentam significativamente à medida que novos elementos são adicionados às arquiteturas da NoC 2D devido às restrições do layout impostas pelo circuito integrado bidimensional (FEERO; PANDE, 2009a). Para transpor essas restrições de layout bidimensional, os designers adotaram um modelo de cirtuito integrado capaz de conectar as camadas dos dispositivos ativos por meio de interconexão vertical de alta velocidade. A esse tipo de arquitetura deram o nome de Circuito Integrado Tridimensional (3D-IC) (DAVIS *et al.*, 2005). Esta abordagem permite usar uma NoC 3D como uma infraestrutura de comunicação que reduz os comprimentos de interconexão entre PEs, melhorando o desempenho geral do 3D-IC. Isso acontece porque o 3D-IC viabiliza o empilhamento de várias camadas de silício permitindo topologias de NoC tridimensionais de baixa latência (NoC 3Ds)(FEERO; PANDE, 2009b) (PAVLIDIS; FRIEDMAN, 2007).

Entretanto, junto com as vantagens trazidas pelas NoCs também apareceram as

limitações, pois com essa nova abordagem que reduz a área e consumo de energia do circuito integrado também o torna suscetível a falhas causadas por interferências, por desgastes devido ao envelhecimento do circuito ou falhas no processo de fabricação (JERRAYA; WOLF, 2004). Nesse contexto, para que não haja falha na comunicação é preciso ter mecanismos de reconfiguração e recuperação na presença de falhas (SILVEIRA *et al.*, 2016). Vários tipos de erros são discutidos e apresentados em (GUPTA *et al.*, 2015) (CHO *et al.*, 2012). No contexto dessa dissertação estamos preocupados em especial as falhas e defeitos que ocorrem nas conexões verticais conforme discutidas em (EGHBAL *et al.*, 2015)

Das tecnologias de interconexão vertical, o Through-Silicon-Via (TSV) tem sido aceito como uma das tecnologias de interconexão vertical mais viáveis para 3D-IC, permitindo uma comunicação inter-camadas rápida e eficiente (BURNS *et al.*, 2001). Devido à complexidade do processo de fabricação, os TSVs apresentam cerca de 1% das falhas mesmo em sua fase de construção (PATTI, 2006) (LOI *et al.*, 2008). O desalinhamento do TSV por ações mecânicas é outro fenômeno construtivo que danifica o acoplamento das camadas, levando a falhas de comunicação vertical. Além disso, a probabilidade de falhas de TSV, como eletromigração, aumenta drasticamente com o aumento da temperatura entre as camadas da NoC (EGHBAL *et al.*, 2015). Consequentemente, os TSVs são estruturas que podem apresentar defeitos tanto no nível de fabricação quanto no nível de execução. Além disso, os TSVs demandam uma área considerável em silício, aumentando assim o custo de fabricação dos circuitos 3D.

Os designers de IC 3D atenuam esses problemas explorando as NoCs 3D com menos roteadores conectando camadas por meio de TSVs. Este modelo de NoC implica a construção de uma estrutura da NoC heterogênea, onde a maioria dos roteadores tem portas para conexão entre camadas e alguns roteadores têm portas adicionais que empregam os TSVs para conexão entre camadas. Esta estrutura heterogênea define uma topologia em NoC 3D irregular, geralmente referenciada como NoC 3D parcialmente conectada verticalmente.

Este trabalho apresenta o Reflect3d, um algoritmo de roteamento eficiente, adaptável e altamente resiliente que visa uma NoC 3D parcialmente conectada verticalmente. O Reflect3d usa dois canais virtuais (VCs) em cada direção para evitar deadlocks, garantindo a entrega do pacote, desde que um TSV saudável conectando todas as camadas esteja disponível, independentemente de sua posição no plano. Além disso, Reflect3d pode rotear pacotes de forma adaptativa dentro de cada camada para evitar áreas congestionadas e alcançar mais caminhos válidos com menos latência do que outros trabalhos.

1.1 Objetivos

Objetivo Geral

O objetivo geral desta dissertação é apresentar o algoritmo de roteamento Reflect3d e a motivação do seu desenvolvimento, assim como avaliar seu desempenho atavés da comparação com outros trabalhos da literatura.

Desta forma, apresentamos nesta dissertação os detalhes que envolvem os processos de comunicação no chip atual, suas principais problemáticas, como a comunicação pode melhorar e como isso pode ser feito. É explicado a importância do algoritmo de roteamento dentro das comunicações *in chip* e também quais são os principais algoritmos atuais que envolvem esse tipo de comunicação.

Assim, propusemos um trabalho que aplica o algoritmo de roteamento através de simulações e comparações que comprovam a eficácia do algoritmo desenvolvido em termos de latência, área, potência, bem como, são apresentados resultados de confiabilidade para falhas dentro da rede de comunicação no chip.

Objetivos Específicos

Além do objetivo geral, outros objetivos específicos devem ser alcançados:

- avaliar o desempenho em termos de área e potência com outros trabalhos relacionados fazendo o comparativo de simualação, a partir de diferentes técnicas encontradas na literatura;
- comparar os resultados e analisar como essa contribuição pode ser benéfica para o uso em sistemas MPSoCs.

1.2 Contribuições

Os resultados desta dissertação foram reunidos no seguinte artigo:

- Alexandre Almeida da Silva, Leonel Maia e Silva Junior, Alexandre Coelho, Jarbas Silveira e Cesar Marcon. "Reflect3d: An Adaptive and Fault-Tolerant Routing Algorithm for Vertically-Partially-Connected 3D-NoC". SBCCI, 2021.

1.3 Organização da Dissertação

Esta dissertação está organizada de forma que no Capítulo 2 está descrita a fundamentação teórica sobre os temas abordados no trabalho. Os trabalhos relacionados são discutidos no Capítulo 3. No Capítulo 4 é apresentada a metodologia do trabalho, descrevendo e explicando os experimentos realizados na dissertação. Os resultados e discussões são abordados no Capítulo 5. Por fim, no Capítulo 6 são sintetizadas as conclusões, contribuições, bem como as perspectivas de trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, os conceitos para a compreensão desta dissertação são apresentados, abordando definições na área de MPSoCs, NoCs bidimensionais (NoCs 2D) e NoCs tridimensionais (NoCs 3D).

2.1 Sistemas Intrachip e MPSoCs

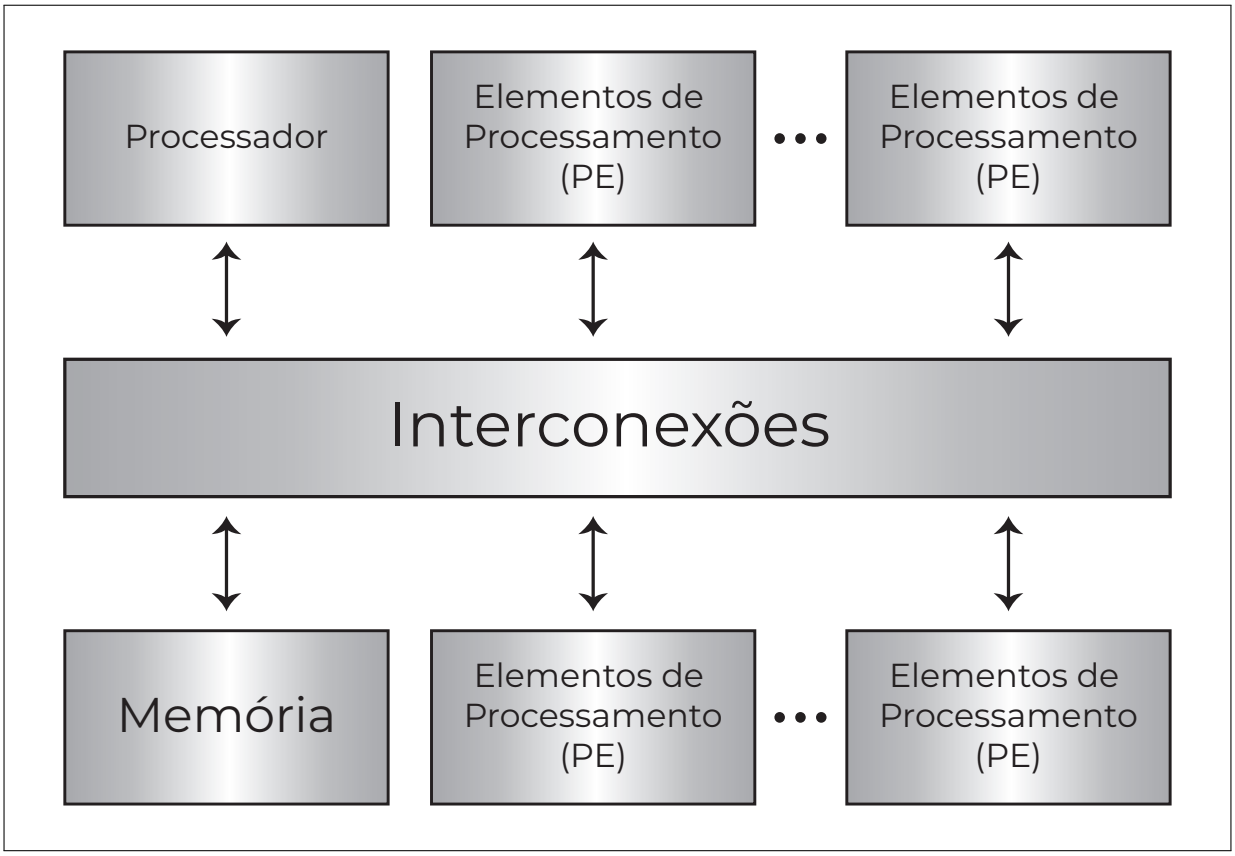
Ao longo das últimas décadas, o desenvolvimento contínuo e os avanços tecnológicos na fabricação de transistores resultaram em uma miniaturização de circuitos integrados (CIs), permitindo com isso que projetos de sistemas complexos possam ser desenvolvidos em um único chip, o qual chamamos de Sistema em um Chip (SoC, do inglês, System-on-Chip). Normalmente, um SoC é um conjunto de processadores, memória e interconexões com um ou mais processadores de uso geral, lógica digital, circuitos analógicos e bancos de memória (PASRICHA; DUTT, 2008).

Para o uso de um grande número de transistores que podem ser fabricados no mesmo CI, o uso de blocos de propriedade intelectual (IP) tornou-se uma abordagem padrão para o projeto desses circuitos. Um circuito integrado consiste em um conjunto de núcleos que incluem múltiplos processadores heterogêneos interligados a um ou mais elementos de memória, possivelmente com uma matriz lógica reconfigurável (AKBARI *et al.*, 2012). O diagrama descrito na Figura 1 ilustra a arquitetura geral de um SoC.

Um SoC contém vários elementos de processamento (PE, *Processor Element*), que são organizados e interligados na forma de um barramento. É importante notar que a estrutura de comunicação mais comum usada para conectar blocos IP a circuitos integrados é um barramento, mas essa estrutura não é muito útil para sistemas computacionais como MPSoCs (sistema multiprocessador em um chip, *multiprocessor system on chip*) em um único CI, visto que a comunicação por barramento tem seus elementos conectados através de um canal compartilhado e com isso temos uma transmissão por vez, o que pode ser um grande problema quando falamos em comunicações de alto desempenho (FEERO; PANDE, 2009a).

Dessa forma, à medida que o número de PE cresce dentro do chip, a interconexão baseada nessa topologia torna-se um gargalo para futuros projetos SoC de alto desempenho. Com a busca constante por melhorias de desempenho e escalabilidade e a necessidade de ter uma infraestrutura de comunicação escalável e flexível, as redes intrachip (NoCs, do inglês, Network-

Figura 1 – Modelo básico de um sistema SoC



Fonte: elaborado pelo autor (2022).

on-Chip) surgem como a tecnologia de comunicação intrachip por ter uma grande escalabilidade, flexibilidade e paralelismo em relação às soluções por barramentos. (VAHDATPANAH *et al.*, 2019).

2.2 NoCs 2D

Para superar as limitações de comunicação por barramentos, as redes intrachip (Networks-on-Chip) surgiram como uma infraestrutura promissora para comunicação on-chip.

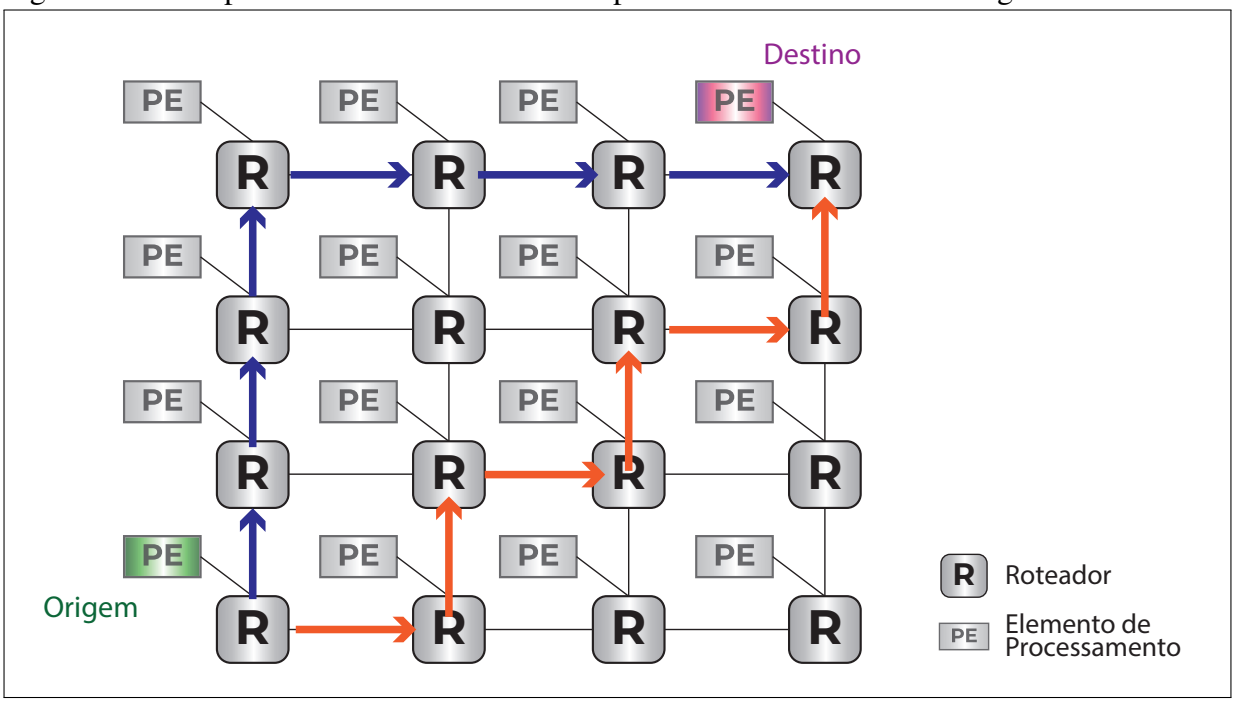
Pode-se pensar em Redes-em-chip (NoCs) como uma rede de computadores na qual os computadores são substituídos por elementos de processamento no chip e a informação viaja para o chip em formato de pacotes (CHARIF *et al.*, 2017).

A figura 2 ilustra o descrito. Há uma NoC bidimensional ou NoC 2D onde cada elemento do processador (PE) é conectado a um roteador por uma interface de rede local. E os roteadores são conectados por links. Estes links permitem que um grande número de elementos de processamento se comuniquem uns com os outros, propagando os pacotes por meio dos roteadores. Significa que as transmissões em uma NoC 2D podem ser feitas quase ao mesmo

tempo. Esta é a razão pela qual Networks-on-chip é o meio de comunicação preferido para Systems-on-Chip.

Além disso, outra vantagem de usar NoC 2D é que ela pode fornecer um número maior de caminhos diferentes entre a origem e o destino, como é visto na Figura 2. Há pelo menos dois caminhos possíveis entre a origem e o destino. Usar o caminho laranja e, também o caminho em azul, são duas alternativas possíveis. Esta possibilidade de selecionar vários caminhos pode aumentar a confiabilidade de uma NoC 2D.

Figura 2 – Exemplo de uma NoC 2D com dois possíveis caminhos entre a origem e o destino.

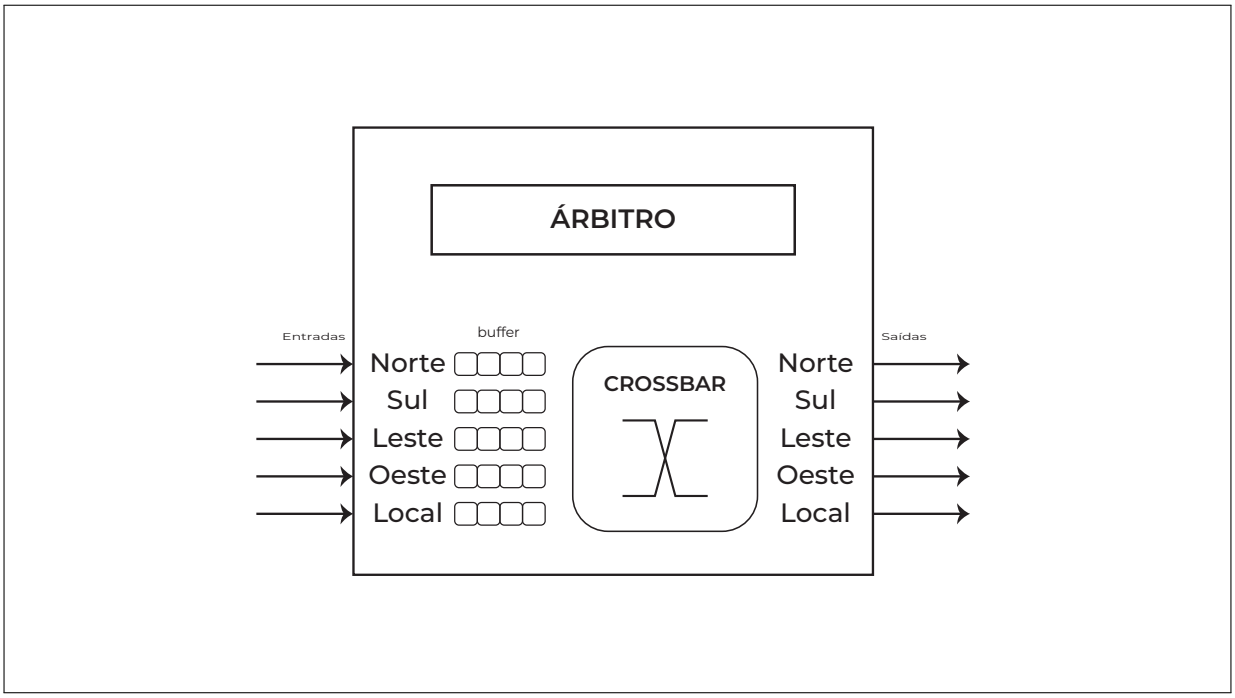


Fonte: elaborado pelo autor (2022).

Para melhor entender todo o funcionamento de uma NoC é importante explorar alguns de seus elementos. O mais importante deles é o roteador, cuja descrição podemos observar na Figura 3. O roteador é conectado a outros roteadores por meio de conexões ou links interligando todos os elementos de uma NoC. Esses links levam pacotes por meio de uma porta entre o roteador e uma porta de conexão até o elemento de processamento correspondente.

No roteador os canais são selecionados através de uma porta de entrada e uma porta de saída, em que os pacotes são encaminhados até sua rota, essa tarefa é executada por um elemento do roteador chamado árbitro; Outra função no roteador é direcionar os pacotes de entrada até a porta de saída correspondente além de armazenar temporariamente os dados enquanto eles estão sendo movidos, para executar essas tarefas o roteador conta com mais dois elementos, o Crossbar e o Buffer respectivamente.

Figura 3 – Roteador



Fonte: elaborado pelo autor (2022).

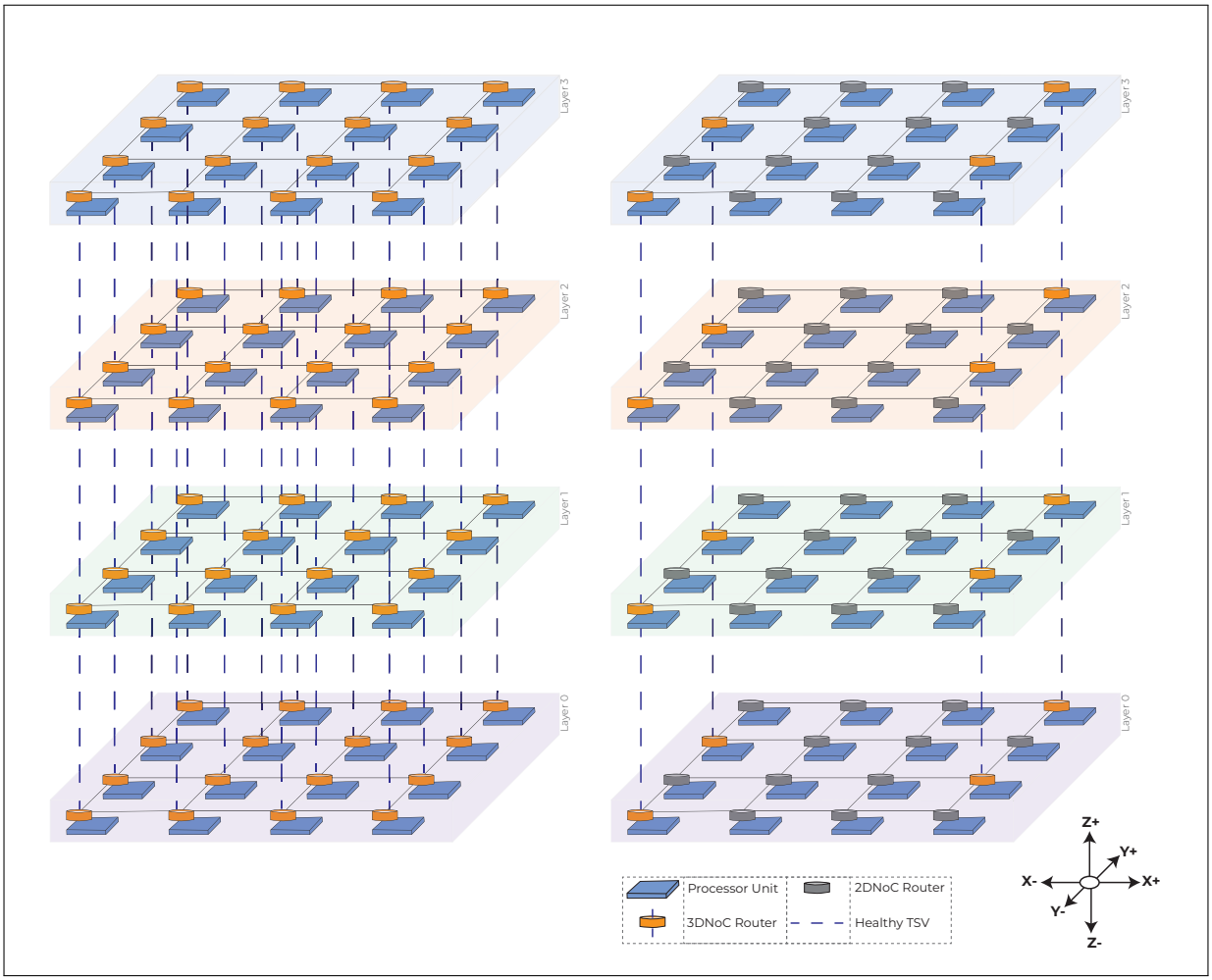
Embora as NoC 2D tragam consigo muitos benefícios para a comunicação on-chip, há um problema relevante, conforme o tamanho de uma rede 2D aumenta, o atraso de transmissão entre roteadores distantes aumenta significativamente. sendo assim, ao invés de crescer a NoC em duas dimensões, a ideia é se mover em direção ao conceito de circuitos integrados 3D onde as camadas são empilhadas verticalmente, uma em cima da outra, utilizando TSVs, sigla para *through-silicon via*, para fazer as conexões verticais entre as camadas planares.

2.3 NoCs 3D

Controle de fluxo, dimensão da rede, arquitetura de comutação, dentre outras, são características que definem uma NoC 3D. Todas essas características impactam diretamente no consumo de energia, na latência e no desempenho (EBRAHIMI *et al.*, 2013). As topologias e os modelos de arquitetura sobre NoCs são bastante variados, contudo, para esta dissertação, apenas os modelos pertinentes a ela serão abordados.

Para melhor entendimento das NoCs 3D, tomaremos um conceito já abordado anteriormente, o atraso de transmissão entre roteadores distantes aumenta significativamente quando o tamanho de uma NoC 2D aumenta. Isso resulta não só em um menor desempenho, como também em um maior consumo de energia. Assim, para evitar tais características, a ideia é avançar para o conceito de circuitos integrados 3D, onde as camadas são empilhadas

Figura 4 – NoC 3D (a)Totalmente Conectada (b)Parcialmente Conectada



Fonte: elaborado pelo autor (2022).

verticalmente, em vez de crescer em duas dimensões.

Neste caso, uma NoC 3D pode ser vista como muitas camadas de NoCs 2D que são empilhadas verticalmente e conectadas por meio de conexões verticais. Isso significa que além das conexões planares, existem também as conexões verticais que permitem as comunicações para cima (up) e para baixo (down) intra-circuito. A Figura 4 (a) mostra um exemplo de uma rede tridimensional totalmente conectada em um chip onde é possível ver que todos os roteadores estão conectados verticalmente.

A partir da figura 4 observa-se que em uma NoC 3D totalmente conectada todos os roteadores podem enviar e receber mensagens tanto nas direções planares, quanto nas direções *Up* e *Down*. Usando essa topologia NoC 3D podemos obter um alto desempenho, pois os pacotes podem percorrer menores caminhos para chegar em seu destino (CHARIF *et al.*, 2018). Além disso, em caso de falhas em um roteador, é possível selecionar qualquer outro roteador para enviar pacotes verticalmente para sua camada de destino. Porém, as NoCs 3D totalmente

conectadas apresentam várias desvantagens, como a necessidade de usar mais canais virtuais e possuírem mais conexão do que os NoCs 2D. Além disso, o custo de fabricação do TSV é alto e a confiabilidade do circuito diminui à medida que o número de TSV aumenta, pois isso nos leva a um número maior de pontos de falhas (DANG *et al.*, 2020).

2.3.1 NoC 3Ds Parcialmente e Verticalmente Conectadas

As conexões verticais têm um alto custo de fabricação, dessa forma a solução parcialmente e verticalmente conectada tem sido destaque como alternativa ao uso da tecnologia 3D IC (DUBOIS *et al.*, 2013). A Figura 4 (b) mostra uma NoC 3D construída usando apenas algumas conexões verticais em sua arquitetura. Embora o uso das NoCs 3D parcialmente conectados diminua o custo de fabricação dos circuitos integrados 3D, o número reduzido de conexões verticais traz novos desafios para as arquiteturas NoC 3Ds. Em particular, o algoritmo de roteamento e o controle de fluxo das NoCs tendem a crescer em complexidade, uma vez que o número de caminhos válidos é diretamente afetado pelo número de conexões verticais. Além disso, a tarefa de obter um caminho para um pacote pode ser ainda mais complicada quando ocorrem falhas permanentes e/ou transitórias dentro de um roteador, o que pode, em alguns casos, invalidar toda a operação da NoC (BORKAR, 2007).

As arquiteturas de malha NoC 3D parcialmente e verticalmente conectadas, incluem em seus roteadores, além das cinco portas planares (Norte, Sul, Leste, Oeste e Local), uma porta para cima (*Up* ou uma porta para baixo *Down* ou uma porta nas duas direções para as camadas intermediárias. Isso significa que cada camada consiste em uma mistura de roteadores 2D clássicos, incluindo apenas 5 portas e roteadores 3D com 5, 6 ou 7 portas, como podemos ver na Figura 4 (b). As portas *Up* e *Down* do roteador são conectadas verticalmente usando *Through-Silicon Via* (TSV). Além disso, cada roteador é identificado por suas coordenadas (X, Y, Z), em que X identifica a coluna, Y a linha e Z a camada.

2.3.2 Propriedade de Comutação (Chaveamento)

A técnica de comutação ou chaveamento (do inglês, *switching*), determina como os recursos da NoC (roteadores, buffers, links, etc.) são alocados para uma rota de mensagem determinada pelo protocolo de roteamento entre a origem e os nós de destino, ou seja, define quando e como um canal de entrada de um roteador é conectado a um canal de saída para permitir a transferência de mensagem pela rede (BEECHU *et al.*, 2017). Das várias formas de

comutação/chaveamento, (TATAS *et al.*, 2014) considera duas como principais, chaveamento por circuito e chaveamento por pacote.

2.3.2.1 Comutação/Chaveamento por Circuitos

Nesse tipo de abordagem, antes da transmissão de dados acontecer, um caminho (circuito) físico entre o nó de origem e o nó de destino é reservado e é constituido de uma série de canais e roteadores. Esse circuito é mantido durante toda a mensagem, até que ela seja entregue por completa no receptor. Devido a isso é fornecido uma total largura de banda ao canal desse circuito o que resulta em baixa latência . Apesar dessa vantagem, esse modo de chaveamento não facilita caso seja necessário expandir a NoC, dificultando a escalabilidade, pois os canais ficam ocupados durante toda a transmissão de dados, mesmo que não haja envio de dados, visto que o caminho é reservado antes do envio da mensagem. (PASRICHA; DUTT, 2008).

2.3.2.2 Comutação/Chaveamento por Pacotes

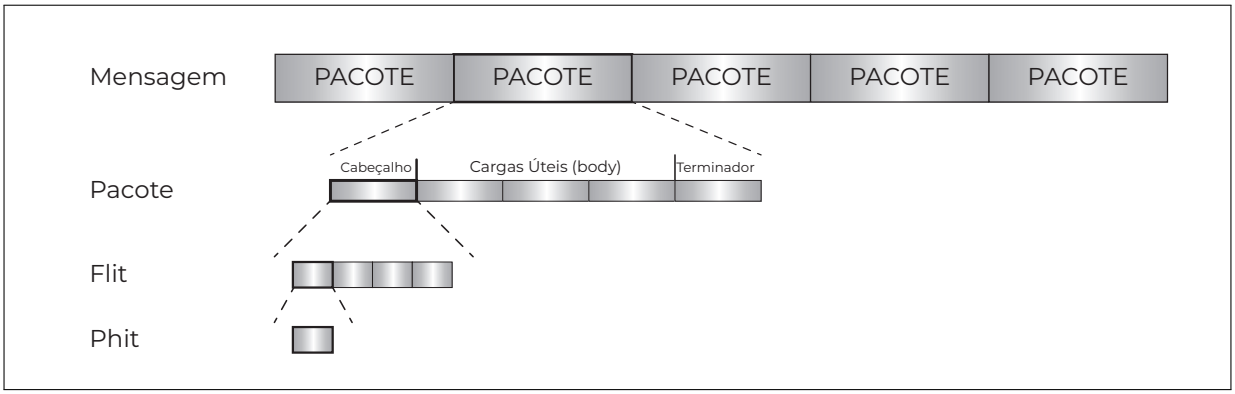
Já nesse modo de chaveamento, ao contrário da comutação por circuito, os pacotes percorrem seu caminho de forma independente, por diferentes roteadores e com diferentes atrasos, até chegar ao seu destino. O tempo de espera para o início do envio dos pacotes é zero e isso é seguido por um atraso variável. Três tecninas de comutação por pacote são bem conhecidas. *Store and Foward* (SAF), nesta técnica um pacote é enviado para o roteador seguinte apenas quando ele ja foi recebido pelo roteador anterior, nesse caso, o tamnho do *buffer* tem que armazenar o pacote inteiro. *Virtual Cut Through* (VTC), em que o primeiro *flit* é enviado de um pacote assim que o espaço necessário para um pacote se torna disponível no próximo roteador. Por último o *Wormhole* em que o espaço necessário no *buffer* receptor é de apenas um *flit* para o encaminhamento de dados, ao invés do pacote inteiro, nesse modo de chaveamento os pacotes são distribuidos entre os vários roteadores. (PASRICHA; DUTT, 2008).

Nesta dissertação é adotada uma técnica de comutação por pacotes *Wormhole* pois a partir dela é possível obter melhor escalabilidade para NoCs e também por ser comumente utilizada na maioria das implementações típicas de NoCs. Em uma NoC baseada em comutação de pacotes, o link entre os roteadores é alocado iterativamente até que o nó de destino seja alcançado. Os dados são transmitidos por mensagens em forma de pacotes decompostos em *flits* (do inglês *flow control unit*). É nos *flits* onde se opera o controle de fluxo. Eles são formados por *phit* (do inglês *physical unit*. (PASRICHA; DUTT, 2008). A estrutura básica de uma mensagem

pode ser observada na figura 5. Os flits são a menor unidade de dados enviados entre diferentes nós na rede e são de três tipos:

- Cabeçalho: Cada pacote contém um único cabeçalho que carrega informações sobre o destino do pacote.
- Carga Útil: Cada pacote pode ter várias cargas úteis (*body*) que transportam os dados reais a serem enviados entre os diferentes nós da rede. Os flits do *body* estão contidos entre o flit do cabeçalho e do terminador.
- Terminador: Cada pacote contém um único terminador que significa o fim do pacote.

Figura 5 – Estrutura da Mensagem



Fonte: elaborado pelo autor (2022).

Além das informações citadas acima, cada um dos flits também carrega informações de qual tipo de flit, do canal virtual (VC) no qual o pacote está sendo roteado e sua sequência no pacote. Outro ponto a considerar é que o cabeçalho é usado para reservar uma porta de saída do roteador e a reserva é mantida até que o terminador seja transmitido.

2.3.3 Controle de Fluxo

O controle de fluxo determina não só como os recursos são compartilhados na rede, como também determina quando os *buffers* e os canais de um roteador são alocados. Segundo (DALLY; TOWLES, 2001), ter uma boa política de controle de fluxo permite um efetivo compartilhamento dos recursos, isso reduz a congestão nos canais da rede e melhora a latência.

Das várias técnicas de controle de fluxo utilizadas em NoCs podemos citar o controle de fluxo baseado em créditos, que é uma técnica onde o emissor mantém um contador com a quantidade de espaços livres no *buffer* do receptor. Se o *buffer* estiver cheio, nenhum *flit* pode ser enviado até que o receptor esteja disponível (DALLY; TOWLES, 2004). Outra técnica é o

handshake, onde um nodo emissor informa a necessidade de enviar dados por meio de um sinal de validação e o receptor confirma a disponibilidade de espaço em *buffer*.

Para esta dissertação a técnica utilizada é a baseada em canais virtuais onde o *buffer* é associado ao canal físico, permitindo que mais pacotes usem o canal (PASRICHA; DUTT, 2008). Embora a técnica de comutação de pacotes apresente uma boa solução para implementação em hardware, alguns problemas devem ser abordados no projeto das NoCs. Em particular, duas questões/propriedades devem ser evitadas para qualquer arquitetura NoC utilizável:

- Deadlock é uma situação em que os pacotes estão envolvidos em uma dependência cíclica que não pode ser resolvida. O deadlock faz com que os pacotes não progridam em direção ao nó de destino. A liberdade de deadlock pode ser fornecida pela característica do protocolo de roteamento ou usando uma lógica de alto custo de detecção e resolução de deadlock.
- Livelock é uma situação em que os pacotes caminham pela rede sem nunca chegar ao destino. Os protocolos de roteamento mínimos e determinísticos são livres de livelock.

2.3.4 Algoritmo de Roteamento

O ponto chave desta dissertação está no algoritmo de roteamento, onde é definido um caminho para ser percorrido pela mensagem, do roteador de origem até o roteador de destino. Além de tentar manter o tamanho das rotas o mais curto possível, o algoritmo de roteamento também deve balancear as cargas do canal de comunicação, com isso, reduzindo a latência e a quantidade de roteadores percorridos (DALLY; TOWLES, 2004).

Os algoritmos de roteamento podem ser classificados como determinísticos ou adaptativos. No primeiro o pacote vai da origem ao destino por um caminho fixo. No segundo a escolha do caminho leva em conta o estado atual da rede, fazendo-se uma escolha dinâmica. Geralmente o algoritmo de roteamento é quem garante que o pacote percorra os caminhos sem gerar *deadlock*.

2.3.5 Canais Virtuais

Nas NoCs com chaveamento *Wormhole*, como abordado por (PASRICHA; DUTT, 2008), o espaço necessário no *buffer* receptor é de apenas um *flit* para o encaminhamento de dados, ao invés do pacote inteiro, isso faz com que ocorra um bloqueio do canal quando um pacote não pode prosseguir para o próximo *buffer*. Assim os canais e o *buffers* que fazem parte

desse caminho possam ficar ocupados. O canal virtual tem por objetivo diminuir o tamanho real dos *buffers* da NoC e replicar a quantidade *buffers* ligados a cada canal de entrada. Assim, um canal físico possui mais de um caminho virtual por meio da comutação entre os *buffers* desse canal. Quando algum desses caminhos estiver bloqueante, os outros ainda permitirão que dados transitem por aquele canal (DALLY; TOWLES, 2004).

Para melhor entendimento dos estágios abordados nessa dissertação, nesse tipo de arquitetura, o roteador consiste em sete portas de entrada, sete portas de saída, dois canais virtuais por cada porta e uma lógica de controle distribuída em *pipeline* de quatro estágios. Segundo (COELHO *et al.*, 2018) a lógica de controle do roteador é composta pela unidade de Computação de Roteamento (RC), a unidade de Alocação de Canal Virtual (VA) e a unidade de Alocação do Comutador (SA). Uma barra central conecta as portas de entrada e saída do roteador. Essas etapas podem ser melhor detalhadas abaixo:

- RC: É o primeiro estágio no pipeline e está ativo na chegada do cabeçalho do flit no roteador. É nessa etapa que se determina a porta de saída do roteador atual onde o cabeçalho do flit irá sair.
- VA: Nessa segunda etapa são alocados os Canais Virtuais e o pacote é direcioando para o seu destino.
- SA: Esssa etapa é responsável por determinar qual entrada do Canal Virtual de uma porta de entrada consegue transmitir um flit através da barra transversal no próximo ciclo.
- Crossbar: Nessa etapa final é conectada as portas de entrada e saída, facilitando assim o caminho do flit de um VC de uma porta de entrada para uma porta de saída.

3 TRABALHOS RELACIONADOS

Neste capítulo é apresentada uma revisão da literatura dos principais temas deste trabalho:

3.0.1 NoC 3D Totalmente Conectadas

Vários trabalhos como (DANG *et al.*, 2020; EBRAHIMI *et al.*, 2013; EBRAHIMI *et al.*, 2013; SALAMAT *et al.*, 2016b; COELHO *et al.*, 2019; AKBARI *et al.*, 2012) apresentam propostas de algoritmos de roteamento tolerantes a falhas aplicados as NoCs 3D totalmente conectadas. Por exemplo, Akbari et al. (DANG *et al.*, 2020) propõem o Afra, que inclui um algoritmo de roteamento adaptativo para tolerar falhas de links verticais sem comprometer significativamente o desempenho da comunicação. Dang et al. (AKBARI *et al.*, 2012) propõem um mecanismo para testar TSVs defeituosos sem interromper ou degradar o desempenho do sistema; no entanto, o mecanismo requer muito tempo de execução e os pacotes de dados podem permanecer nesses caminhos defeituosos durante o procedimento de teste. Embora esses algoritmos tolerem vários links verticais defeituosos, eles assumem que todos os roteadores da NoC 3D devem ter TSVs conectando as camadas.

3.0.2 NoC 3D Parcialmente Conectadas

Enquanto a maioria dos trabalhos são baseados em NoCs 3D totalmente conectadas, estamos interessados naqueles que consideram a NoC 3D parcialmente e verticalmete conectada, como o LBDR3D proposto por Niazmand et al. (NIAZMAND *et al.*, 2016). O LBDR3D é um framework que suporta uma variedade de algoritmos de roteamento parcialmente adaptativos baseados no modelo de turnos que podem ser reconfigurados para tolerar falhas em links horizontais e verticais. O LBDR3D usa bits configuráveis em cada roteador para apontar para o link vertical mais próximo selecionado offline com base na distância de Manhattan. O algoritmo de roteamento Rout3D proposto por Charif et al. (CHARIF *et al.*, 2017) opera de forma semelhante ao LBDR3D, usando bits configuráveis por roteador e empregando reconfiguração offline para selecionar o TSV mais próximo e saudável, também chamado de elevador, com base na distância de Manhattan. O Rout3D mostra excelentes resultados usando um total de 8 (oito) Canais Virtuais (VCs). Também foi proposto um algoritmo de localização de elevadores válidos, visto que o algoritmo proposto é validado nesses casos, chamado de *Elevator Compass*,

que localiza o elevador mais próximo através de uma combinação de 4 bits para cada elevador disponível na rede. Esses bits podem ser reconfigurados toda vez que o estado da rede muda devido a falhas. Além disso, esse método não é específico ao Rout3D e pode ser combinado com qualquer outro algoritmo de roteamento. No entanto, como nenhuma solução online é apresentada para tolerar falhas de tempo de execução, os pacotes são descartados durante a fase de reconfiguração.

3.0.3 Outros trabalhos Importantes em NoC

Nessa subseção são apresentados alguns dos principais trabalhos que serviram como temas bases para elaboração desta dissetação.

Salma Hesham (HESHAM *et al.*, 2017) discute as implicações das aplicações em tempo real (*Real-Time-Applications*, RTAs) no design de NoC. As contribuições existentes que cobrem aspectos garantidos de suporte de qualidade de serviço (QoS), adaptabilidade e eficiência de energia em NoCs em tempo real (RT-NoCs) são revisados. O artigo examina ainda as metodologias de avaliação e medidas de desempenho experimental para RT-NoCs. Além disso, este artigo fornece uma visão dos pontos de pesquisa abertos nesse campo.

Yuxiang Fu (FU *et al.*, 2019) faz uma análise do gargalo de desempenho das NoCs 3D parcialmente conectadas. Um esquema de atribuição dinâmica de elevador com reconhecimento de congestionamento, chamado de *Congestion-aware Dynamic elevator Assignment* (CDA). O critério de atribuição é escolhido como a soma ponderada do atraso do roteador e o quadrado da utilização do buffer ao longo do caminho da origem até o final do elevador atribuído na camada de destino. Entre os caminhos disponíveis, é selecionado o caminho com o critério de atribuição mínimo e o elevador ao longo do caminho é atribuído ao pacote.

Ronak Salamat (SALAMAT *et al.*, 2018) fala da importância dos algoritmos de roteamento adaptável nas arquiteturas em NoC, especialmente no balanceamento de carga para trafegos não uniformes, propor um algoritmo de roteamento adaptável para NoC 3D parcialmente conectada que forneça melhor desempenho em comparação com os algoritmos de roteamento publicados.

3.0.4 Trabalhos para Base Comparativas a essa Dissertação

Os trabalhos apresentados nessa subseção mostram os algoritmos principais que foram simulados e comparados nesta dissertação.

Salamat et al. (SALAMAT *et al.*, 2016a) projetaram o algoritmo CoBRA para tolerar falhas de TSV em tempo de execução. O algoritmo requer um TSV saudável na coluna mais a leste ou oeste, gerando restrições que limitam a localização e seleção do elevador. O CoBRA roteia os pacotes para o leste até encontrar um TSV saudável; se nenhum TSV for encontrado, o roteamento será reconfigurado para entregar os pacotes para o oeste. CoBRA usa dois VCs ao longo da dimensão Y para evitar impasse. Os principais problemas do algoritmo são:

- congestionamento de pacotes, no caso de um grande fluxo de dados, já que todos os seus pacotes são direcionados primeiramente para leste, e
- descarte de pacotes NoC durante a fase de reconfiguração.

Charif et al. (CHARIF *et al.*, 2018) projetaram o algoritmo de roteamento First-Last que garante a entrega de pacotes se houver pelo menos uma coluna de TSVs disponível na rede. A abordagem adotada utiliza VCs para comunicação entre roteadores e pode ser reconfigurada offline, o que não está presente nas abordagens mencionadas anteriormente. Entretanto, o algoritmo First-Last possui algumas restrições quanto ao posicionamento do TSV na NoC, não podendo corrigir falhas de tempo de execução, gerando perda de pacotes durante a fase de reconfiguração.

Dubois et al. (DUBOIS *et al.*, 2013) propuseram o algoritmo de roteamento determinístico Elevator-First, cuja implementação insere o endereço TSV alvo em cada pacote. Embora não tenha restrições quanto ao posicionamento do TSV e suporte qualquer topologia NoC, os endereços dos elevadores são definidos em tempo de projeto. Assim, falhas de TSV em tempo de execução podem implicar no descarte de pacotes. A abordagem Elevator-First implementa um total de 10 (dez) VCs, dois para cada direção planar e um para cima/baixo.

Tabela 1 – Síntese dos principais trabalhos usados para comparação nessa dissertação

Algoritmo	Abordagem
Elevator-First	Algoritmo de roteamento determinístico, cuja implementação insere o endereço TSV alvo em cada pacote.
CoBRA	Tolera falhas de TSV em tempo de execução. O algoritmo requer um TSV saudável na coluna mais a leste ou oeste para localizar o elevador
First-Last	Algoritmo de roteamento que garante a entrega de pacotes se houver pelo menos uma coluna de TSVs disponível na rede

Fonte: o autor.

Esta dissertação, em contraste com os trabalhos descritos acima, propõe um algoritmo de roteamento adaptativo para as NoCs 3D capaz de tolerar falhas de TSV detectadas na

fabricação e em tempo de execução sem impor regras específicas para a seleção dos elevadores NoC. Dentre os trabalhos relacionados, além da tabela 1, alguns outros deles são detalhados no Apêndice A.

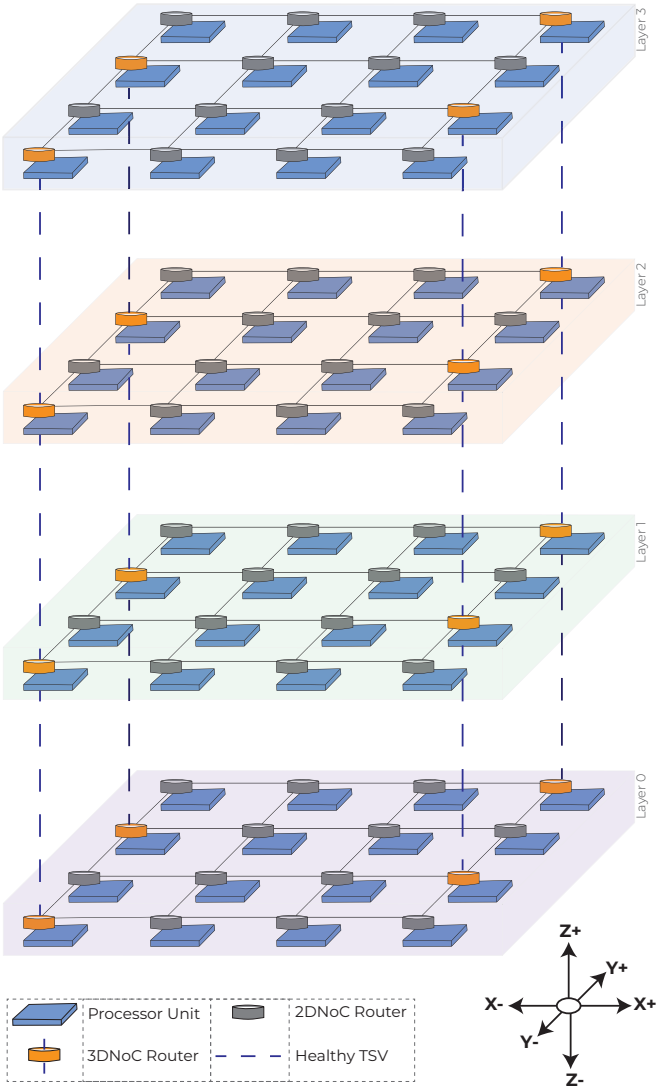
4 METODOLOGIA

Neste Capítulo, a metodologia desta dissertação é apresentada. Esta dissertação apresenta uma NoC 3D com quatro camadas planares interligadas por TSVs, que são links verticais também chamados de elevadores - terminologia definida por (DUBOIS *et al.*, 2013). A Figura 6 ilustra a arquitetura NoC 3D utilizada nos experimentos desta dissertação, contendo 64 (sessenta e quatro) PEs distribuídos em quatro camadas, cada camada possuindo 16 (dezesseis) PEs interligados por redes *mesh* 4×4. Apenas quatro roteadores são conectados verticalmente pelos elevadores para ilustrar uma NoC parcialmente e verticalmente conectada, reduzindo os custos de área. Esta topologia resulta em uma NoC 3D com roteadores 2D contendo apenas cinco portas de entrada/saída (Local, Leste: **X+**, Oeste: **X-**, Norte: **Y+**, Sul: **Y-**) e roteadores 3D que, além das portas intracamadas, incluem duas portas intercamadas (Para cima: **Z+**, Para baixo: **Z-**). Além disso, os roteadores localizados nos limites (norte, sul, leste, oeste, cima e baixo) da NoC 3D possuem um número menor de portas devido à topologia de malha não definir conexões nas extremidades. Para definir uma terminologia padrão, chamamos as portas Norte, Sul, Leste, Oeste, Cima e Baixo como **Y+**, **Y-**, **X+**, **X-**, **Z+** e **Z-**, respectivamente. Ressaltamos que o algoritmo Reflect3D, usado nesta dissertação foi desenvolvido para rotear pacotes independente do número e posicionamento dos TSVs; as escolhas de TSVs descritas na Figura 6 são aleatórias, servindo apenas como exemplo de arquitetura de destino para compreensão do algoritmo Reflect3d aqui proposto.

4.1 Reflect3d - O Algoritmo de Roteamento Proposto

Reflec3d é um algoritmo de roteamento distribuído baseado na abordagem de (CHARIF *et al.*, 2018) e (SALAMAT *et al.*, 2016b), que consiste em definir um conjunto de redes virtuais (do inglês, *Virtual Network* - VNs) para garantir conectividade e evitar deadlock. Uma Rede Virtual (VN) é uma rede lógica projetada na rede física que usa canais virtuais fixos que podem ser selecionados pelo algoritmo de roteamento. O algoritmo Refletc3d define quatro VNs (ou seja, *VN0*, *VN1*, *VN2*, *VN3*), correspondendo a canais físicos (portas de saída) que o pacote pode usar para progredir em direção ao seu destino *Address*. O Refletc3d emprega dois VCs, implementados por buffers dedicados, em cada direção, pois dois VNs compartilham canais físicos. Um VC é compartilhado pelo par (*VN0*, *VN1*), e o outro é compartilhado pelo par (*VN2*, *VN3*).

Figura 6 – Ilustração de uma 3D-NoC parcialmente e verticalmente conectada.



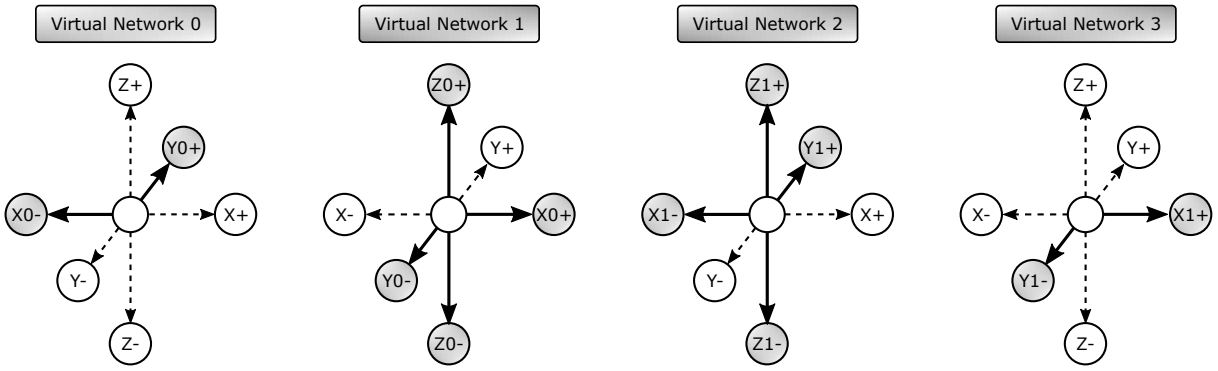
Fonte: elaborado pelo autor (2021).

A Figura 7 ilustra que *VN0* e *VN3* roteiam pacotes apenas no plano, com *VN0* usando os canais físicos **X-** e **Y+**, enquanto *VN3* usa os canais físicos espelhados **X+** e **Y-**. *VN1* e *VN2* operam em três dimensões, ambos acessando os canais físicos **Z+** e **Z-**, com *VN1* direcionando no plano através do **X+** , **Y-** canais físicos, enquanto *VN2* usa os canais físicos espelhados **X-**, **Y+**.

Refletc3d implementa dois algoritmos aninhados; o nível mais externo (Algoritmo 1 - algoritmo de roteamento intercamada) executa o roteamento para a camada de destino, e o nível mais interno (Algoritmo 2 - algoritmo da camada de roteamento intracamadas) implementa roteamento entre camadas. Observe que o Refletc3d é executado de forma distribuída em cada roteador sempre que recebe um pacote.

O algoritmo de roteamento pode ser descrito como segue. O pacote é direcionado

Figura 7 – Canais físicos usados pelas quatro redes virtuais empregadas no algoritmo Reflect3d.



Fonte: elaborado pelo autor (2021).

Algorithm 1 Inter-layer routing algorithm

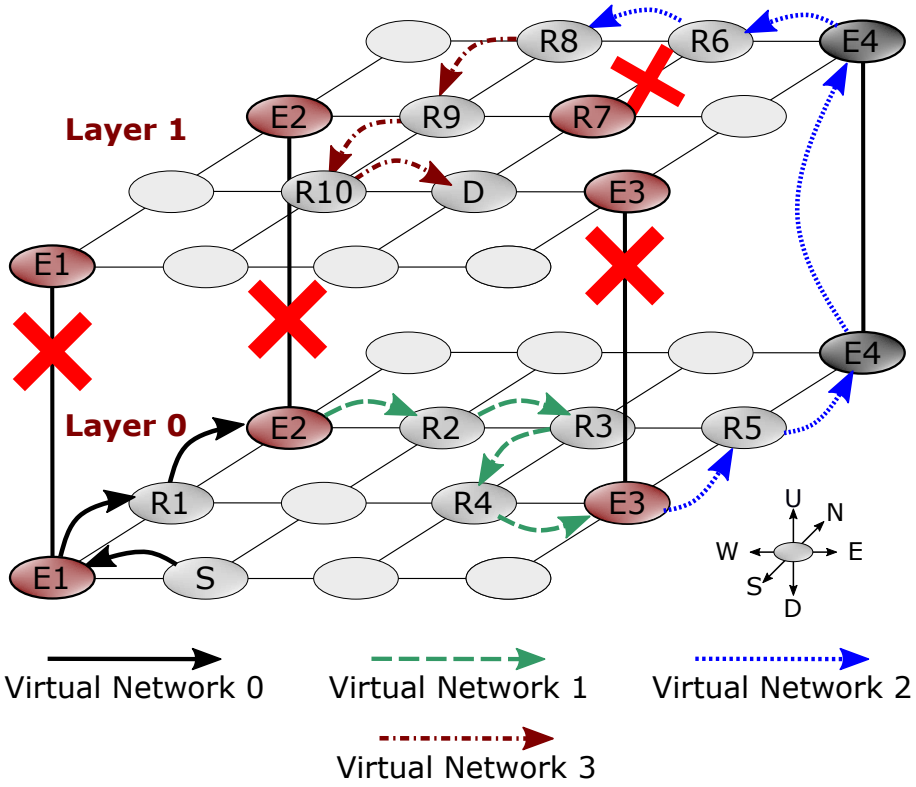
Input:
1: $Dest$: Destination router
2: VN_in: Current virtual network
Output:
3: R: Set of possible output channels
4: VN_out : Output virtual network
5: **if** $Dest.Up$ or $Dest.Down$ **then**
6: **if** $VN\_in = 0$ **then**
7: **if** $Dest.Up$ **then**
8: $Elevator \leftarrow Elevator\_Up$
9: **else**
10: $Elevator \leftarrow Elevator\_Down$
11: **end if**
12: **else**
13: $(R, VN\_out) \leftarrow Algorithm\ 2(Dest, VN\_in)$
14: **end if**
15: **end if**
=0

ao roteador de destino em cada roteador, sempre seguindo uma ordem crescente de roteamento entre os VNs; ou seja, o único fluxo válido é dado por ($VN0 \rightarrow VN1 \rightarrow VN2 \rightarrow VN3$), não sendo necessário passar por todos os VNs . O roteamento sempre começa com *VN0*, e se necessário, a mudança de camadas só ocorre durante os roteamentos *VN1* ou *VN2*.

O algoritmo recebe *Dest* e VN_in como entradas descrevendo o endereço de destino do pacote e o número VN. *Dest* é obtido comparando a posição atual do roteador com o endereço de destino colocado no cabeçalho do pacote. O algoritmo gera um conjunto de portas de saída e o número do próximo VN, que pode ser o mesmo número de VN ou maior quando necessário alterar o VN. Se o endereço de destino estiver na mesma camada do roteador atual, o roteamento é executado dentro do plano aplicando Algoritmo 2 (veja Algoritmo 1 - linha 13). Caso contrário, Algorithm 1 calcula o endereço do elevador apropriado em relação à camada de destino e o VN

atual (*VN1* ou *VN2*). É possível mover para cima e para baixo em *VN1* e *VN2*, então não há necessidade de alterar o VN nessas duas regiões. Além disso, o roteamento pode ser realizado de forma adaptativa, selecionando a rota menos congestionada e mais opções de rotas devido à simetria da rede e à existência de VCs.

Figura 8 – Exemplo de um cenário com links verticais e horizontais falhos



Fonte: elaborado pelo autor (2022).

4.2 Reflect3d em Ação

Esta seção exemplifica a operação do Reflect3d para um pacote trafegando na rede com um cenário de falha, destacando aspectos importantes da arquitetura alvo sobre os mecanismos de detecção de falhas e avaliação de tráfego.

Essa função de custo tem escopo local, pois analisa apenas a presença de falhas, que é uma restrição de operação, e o tráfego de pacotes, que é um requisito para mitigar o congestionamento, nos links do roteador local. A detecção de falha de link é feita com um pacote de controle especial e um comando de loopback. Este pacote de controle, que é executado de acordo com uma periodicidade definida na camada de aplicação, contém uma sequência padrão para testar todos os bits de enlace. Como o padrão enviado não corresponde ao recebido, todo o link é considerado defeituoso, independentemente de quais e quantos bits falharam e se a falha

ocorreu no canal de entrada ou saída do link. Os roteadores têm um bit de status para cada link, que informa a presença de falha. O algoritmo de roteamento analisa esse bit para verificar a viabilidade do uso do link.

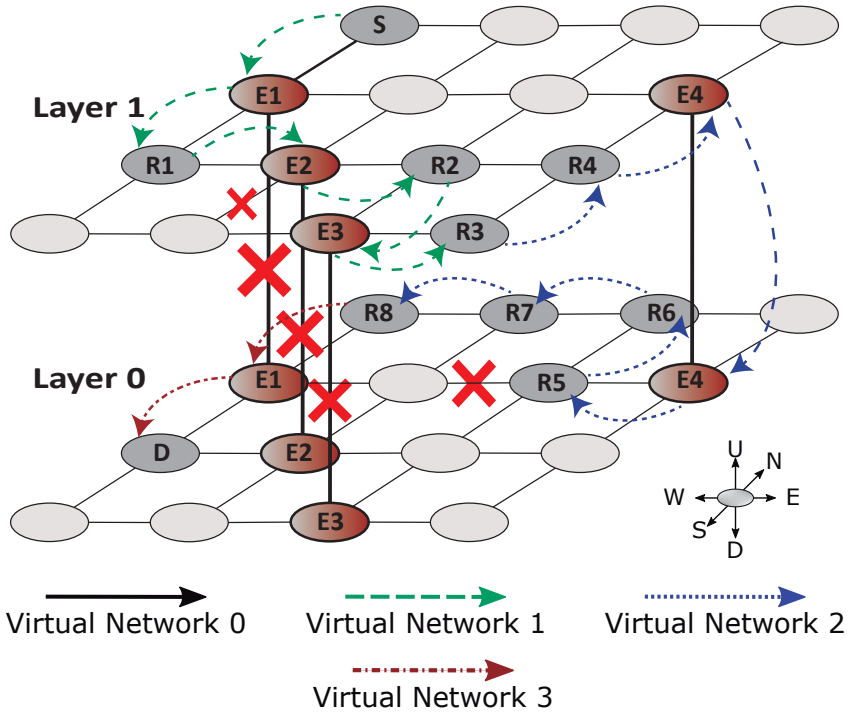
A análise do congestionamento é feita através de contadores temporizados localizados em cada porta do roteador, permitindo calcular o fluxo de pacotes por intervalos de tempo. O algoritmo de roteamento adaptativo recebe as informações de fluxo para escolher aquele com menor tráfego se tiver dois links habilitados. Observe que uma função de custo com escopo local pode levar a rotas com atrasos maiores do que aquelas obtidas com status da NoC global. No entanto, conhecer o status global da NoC implica em um mecanismo de comunicação mais complexo para distribuir todos os dados de tráfego e links defeituosos através da NoC.

A Figura 8 apresenta duas camadas (*Layer*0 e *Layer*1) da NoC 3D utilizada neste trabalho (Figura 6) para um cenário com links defeituosos verticais e horizontais. A Figura 8 também descreve um roteamento de pacotes que começa do roteador de origem *S*, localizado na *Layer*0, até o roteador de destino *D*, localizado na *Layer*1, usando os VNs para executar o roteamento na presença de falhas.

Inicialmente, o pacote é roteado para o elevador $E1$ usando os caminhos definidos na *VN0*; neste caso, o pacote vai na direção **X-**. No entanto, $E1$ está com defeito, forçando o roteador a direcionar para $R1$ através do caminho **Y+**; por sua vez, $R1$ redireciona para o próximo elevador ($E2$), que também está com defeito, forçando o pacote a mudar de direção para buscar $E3$ pelos caminhos habilitados para *VN1*. Aqui, o pacote segue o caminho **X+** até atingir $R3$, onde pode escolher os caminhos **X+** e **Y-**; para ilustrar o funcionamento do algoritmo adaptativo, assumimos que o tráfego hipotético entre $R3$ e $R5$ é superior ao entre $R3$ e $R4$, forçando a escolha da direção **Y-**. O pacote usa a direção **X+** para ir de $R4$ a $E3$, mantendo assim *VN1*. O roteador conectado a $E3$ sabe que está com defeito, precisando rotear o pacote para $E4$, que é o único elevador que não foi avaliado; este novo roteamento deve ser feito na direção **Y+** de *VN2*. Se $E4$ for um elevador saudável, o pacote pode alcançar a camada de destino através do caminho **Z+**. Dentro da camada de destino, o pacote continua em *VN2* em direção a $R7$ através do caminho **X-**. O link entre os roteadores $R6$ e $R7$ está com defeito, forçando uma nova decisão de roteamento. A falha é superada enviando o pacote para $R8$; para direcionar para $R9$ e $R10$, é necessário mudar para *VN3*, permitindo usar a direção negativa **Y-**. Finalmente, $R10$ roteia o pacote na direção positiva **X+** para chegar ao seu destino em *D*.

A Figura 9 também apresenta duas camadas (*Layer*0 e *Layer*1) da NoC 3D utilizada

Figura 9 – Exemplo 2 de um cenário com links verticais e horizontais falhos com origem na camada superior



Fonte: elaborado pelo autor (2022).

neste trabalho (Figura 6) para um cenário com links defeituosos verticais e horizontais. A Figura 9 descreve um roteamento de pacotes que começa do roteador de origem *S*, localizado na *Layer*1, até o roteador de destino *D*, localizado na *Layer*0, dessa forma validamos o algoritmo para o roteamento dos pacotes da camada de cima para baixo e também observamos um caso em que o pacote de origem se encontra nos roteadores do canto.

Inicialmente, o pacote é roteado para o elevador $E1$ usando os caminhos definidos na *VN1*, não é possível usar a *VN0* já que o roteador de origem se encontra no canto e nessa condição já tem que trocar de VN; dessa forma, o pacote vai na direção **Y-** até chegar em $E1$. No entanto, $E1$ está com defeito, forçando o roteador a direcionar para $R1$ através do caminho **Y-** e em seguida **X+**; por sua vez, o elevador $E2$ também está com defeito, permanecendo na **VN1**, redireciona para $R2$ em busca do elevador $E3$, que também está com defeito, forçando o pacote a mudar de direção para buscar $E4$, o pacote vai na direção **X+** para chegar em $R3$, mas para chegar em $R4$, os pacotes são direcionados pelos caminhos habilitados para *VN2*. Aqui, o pacote segue o caminho **Y+** até atingir $E4$, sendo $E4$ um elevador saudável, o pacote pode alcançar a camada de destino através do caminho **Z-** onde pode escolher os caminhos **X-** e **Y+**. O pacote usa a direção **X-** para ir de $R6$ a $R8$, mantendo assim *VN2*. A partir de $R8$ é necessário mudar

para *VN3*, permitindo usar a direção negativa **Y-**. Finalmente, o pacote chegar ao seu destino em *D*.

4.3 Liberdade de Deadlock and Livelock

Uma situação de *deadlock* exige ciclos em pelo menos uma Rede Virtual (EBRAHIMI; DANESHTALAB, 2017). O Reflect3d propõe dois canais virtuais em cada direção, divididos em quatro VNs projetados para evitar ciclos e, assim, evitar deadlocks. Além disso, o algoritmo de roteamento realiza uma abordagem crescente para a seleção de VN; quando um pacote está em *VN1*, ele não pode retornar a *VN0*; este pacote só pode mudar para *VN2* e *VN3*. Consequentemente, o Reflect3d evita pacotes de roteamento de formação de ciclo em forma de espiral. O algoritmo Reflect3d também é livre de livelock, pois qualquer VN individual é percorrido em ordem crescente, evitando loops.

Algorithm 2 Intra-layer routing algorithm

Input:

Dest: Destination router
VN_in: Current virtual network

Output:

R: Set of possible output channels
VN_out : Output virtual network

```
VN_out ← VN_in
if Dest.North and Dest.West then
    R ← {X−, Y+}
else if Dest.West then
    R ← {X−}
else if Dest.North then
    R ← {Y+}
else
    VN_out ← 1
    if Dest.South and Dest.East then
        R ← {X+, Y−}
    else if Dest.East then
        R ← {X+}
    else if Dest.South then
        R ← {Y−}
    else
        VN_out ← 2
        if Dest.North and Dest.West then
            R ← {X−, Y+}
        else if Dest.West then
            R ← {X−}
        else if Dest.North then
            R ← {Y+}
        else
            VN_out ← 3
            if Dest.South and Dest.East then
                R ← {X+, Y−}
            else if Dest.East then
                R ← {X+}
            else if Dest.South then
                R ← {Y−}
            end if
        end if
    end if
end if
=0
```

5 RESULTADOS E DISCUSSÕES

Neste Capítulo, os resultados encontrados pela aplicação da metodologia descrita no Capítulo 4 são apresentados e discutidos. Este capítulo compara o algoritmo desenvolvido nessa dissertação (Reflect3d) com três outros algoritmos que foram descritos nos trabalhos relacionados, esses algoritmos também implementam roteamento tolerante a falhas para uma NoC 3D parcialmente conectada na mesma arquitetura NoC, eles são CoBRA (SALAMAT *et al.*, 2016a), Elevator-First (DUBOIS *et al.*, 2013), e First-Last (CHARIF *et al.*, 2018). Os quatro algoritmos foram implementados na linguagem de descrição de hardware SystemVerilog e simulados utilizando o simulador ModelSim da Mentor Graphics.

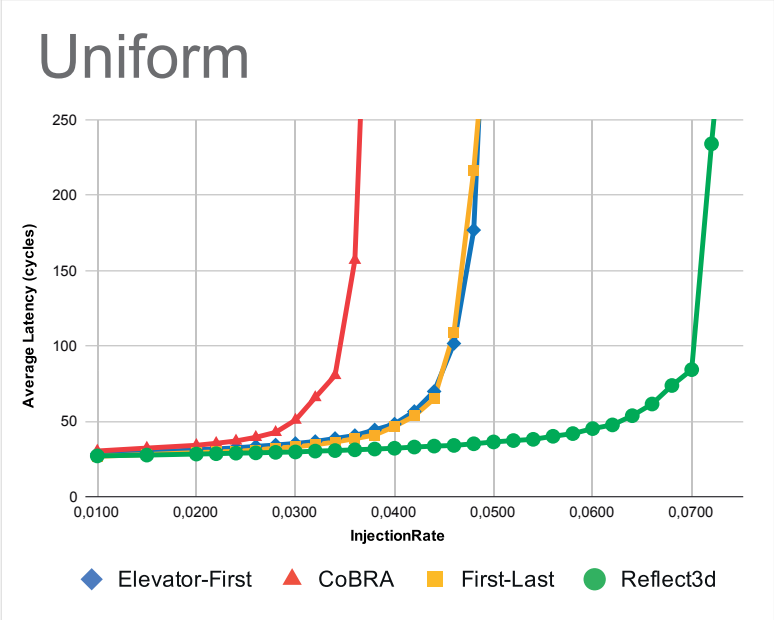
5.1 Análise de simulação

A avaliação da latência considera uma malha 4x4x4 da NoC 3D descrita na Figura 6, contendo 64 PEs e quatro conexões verticais. As figuras 10, 11 e 12 exibem a comparação de latência média, em simulações de 100.000 ciclos, os roteadores da NoC 3D são baseados em FIFOs com 4 flits e um tamanho de pacote fixado em 5 flits, quando vários candidatos estão disponíveis, o algoritmo de roteamento seleciona a porta de saída menos congestionada, com base em uma métrica de congestionamento local. Para cada simulação, os pacotes sintéticos de 5 flits são injetados e aguardam até que 100.000 deles sejam recebidos. Todos os dados do três modos de tráfego são apresentados no Apêncice B. Os algoritmos dos trabalhos Elevator-First, CoBRA, First-Last e Reflect3d usam esse mesmo padrão para simulação. Para avaliação de desempenho, são considerados 3 tipos de modos de tráfego sintéticos:

- **Uniform Random:** Cada origem envia pacotes para um destino aleatório seguindo uma distribuição uniforme.
- **Bit complement:** O nó (X, Y, Z) envia pacotes para o nó (X', Y', Z') em que X' é o complemento de X.
- **Shuffle:** Seja N o número de nós e id o número de identificação do nó, então um nó de id $<$ N/2 envia pacotes para o nó 2 × id. Um nó de id $\geq$ N/2 envia pacotes para o nó (2 × id + 1) mod N.

As Figuras 10, 11 e 12 mostram que o Reflect3d resulta em menor latência média para todos os padrões de tráfego. Este melhor desempenho é devido ao número de VCs presentes no algoritmo Reflect3d e de como essas VCs são organizadas em VNs.

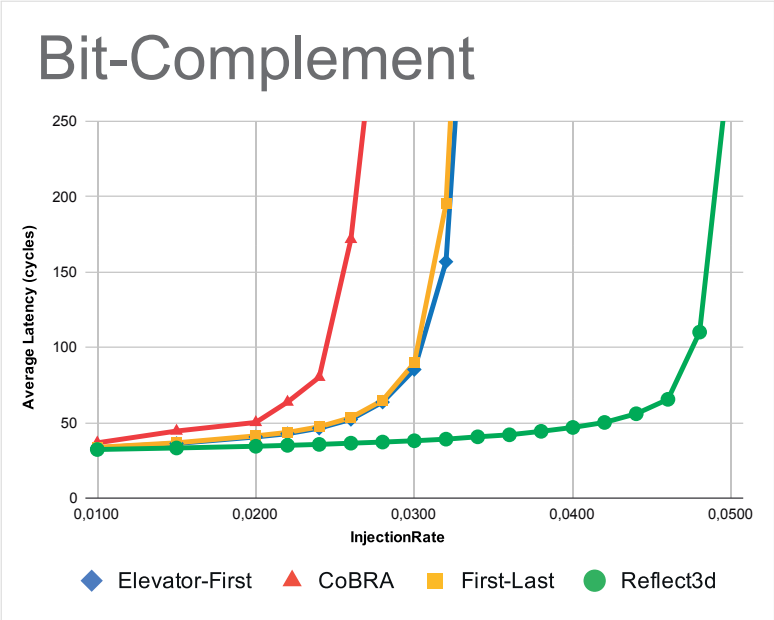
Figura 10 – Latência média de pacote para uma NoC 3D 4x4x4. Modo de tráfego: Uniform Random



Fonte: elaborado pelo autor (2022).

O Reflect3d usa 12 VCs no total; dois canais a mais do que o Elevator-First e quatro canais a mais do que os algoritmos CoBRA e First-Last. No entanto, adicionar VCs não é suficiente para reduzir a latência do pacote; a organização dos VCs é outro fator determinante no desempenho temporal com tolerância a falhas em NoC 3D. Nesse sentido, o Reflect3d emprega VNs de forma refletida que permite novas rotas de pacotes devido ao aumento de novos caminhos

Figura 11 – Latência média de pacote para uma NoC 3D 4x4x4. Modo de tráfego: Bit-Complement



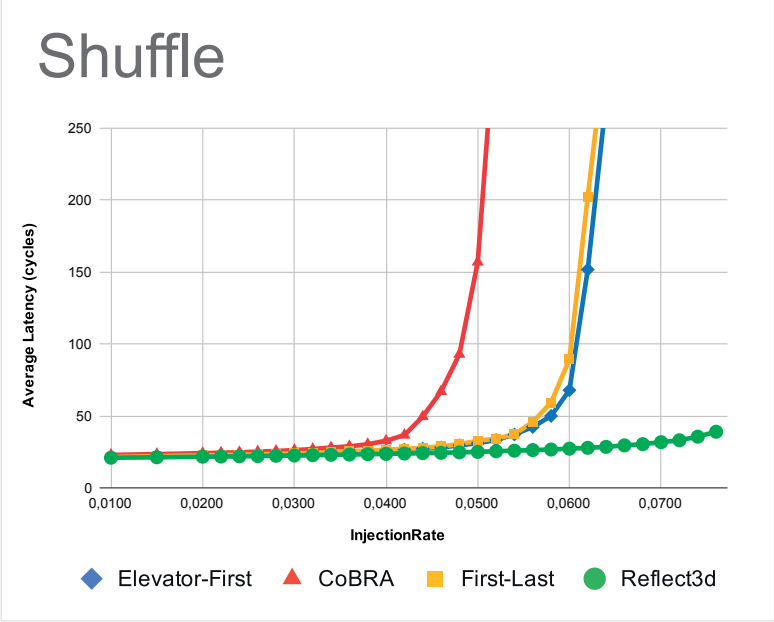
Fonte: elaborado pelo autor (2022).

válido. O roteador pode não só escolher novas rotas devido ao congestionamento durante a viagem dos pacotes, como também verificar as falhas nos links verticais e horizontais, tudo isso possibilita agregar um maior suporte à tolerância a falhas, além de obter menor latência quando comparado aos outros algoritmos.

O Reflect3d tem latências médias mais baixas, mesmo quando comparado aos algoritmos First-Last e Elevator-First, que direcionam os pacotes para os TSVs mais próximos. Essa redução de latência global ocorre porque o Reflect3d realiza uma pesquisa aleatória TSV, o que ajuda a distribuir melhor os pacotes na rede. Por outro lado, os algoritmos First-Last e Elevator-First geram hotspots (ou seja, pontos de congestionamento de rede) ao selecionar os elevadores mais próximos, implicando em uma maior média de latência de pacotes.

Já o Reflect3d procura de forma aleatória e distribuída por um elevador de rede, diminuindo a concentração de pacotes em uma direção NoC específica. Assim, o Reflect3d reduz a latência média dos pacotes em relação ao CoBRA que sempre direciona todos os pacotes para leste primeiro para encontrar um elevador e, caso contrário, redireciona para oeste, gerando tráfego excessivo que congestiona o eixo X.

Figura 12 – Latência média de pacote para uma NoC 3D 4x4x4. Modo de tráfego: Shuffle



Fonte: elaborado pelo autor (2022).

5.2 Análise de Sínteses de Hardware

Adicionar tolerância a falhas e desempenho ao mesmo tempo a um algoritmo de roteamento requer um custo de hardware que deve ser avaliado e comparado. Para isso, utilizamos a ferramenta *Genus Synthesis Solution* da Cadence com uma biblioteca CMOS de 28nm.

As Tabelas 2 e 3 resumem as estimativas de consumo de área e dissipação de energia resultantes para os quatro algoritmos de roteamento. A dissipação de potência foi extraída com base na corrente de fuga e na corrente dinâmica.

Para melhor validação dos resultados obtidos, foi escolhido avaliar os roteadores em diferentes formas devido suas distribuições entre as camadas, assim um roteador 2D de 5 portas, um roteador 3D de 6 portas com uma única conexão vertical e um roteador 3D de 7 portas com links verticais para cima e para baixo foram analisados. Todos os roteadores foram sintetizados com funcionalidade completa, incluindo buffers de porta de entrada, controle VC, alocador de switch, barra cruzada e o algoritmo de roteamento.

Os resultados mostram que o Reflect3d consome em média 17,93%, 15,85% e 6,49% a mais de área em relação aos algoritmos CoBRA, First-Last e Elevator-First, respectivamente. Além disso, o Reflect3d dissipa uma média de 24,26%, 21,39% e 8,94% mais potência do que os algoritmos CoBRA, First-Last e Elevator-First, respectivamente. Este aumento no consumo de área e dissipação de energia é principalmente devido ao maior número de VC e à complexidade adicional do algoritmo de roteamento. No entanto, para arquiteturas 3D em que a estrutura de comunicação tem um custo proporcional reduzido em relação aos elementos de processamento, o trade-off entre a redução na latência média de comunicação e os custos de síntese favorece a escolha do algoritmo de roteamento Reflect3d, proposto nessa dissertação.

Tabela 2 – Resultados de Sínteses de Hardware: Área

Type	**Área** (μm^2)			
(# Ports)	Elevator-First	CoBRA	First-Last	Reflect3d
5-Port	17371	14677	15220	17784
6-Port	21735	18957	19395	23308
7-Port	25870	23274	23727	29074

Fonte: o autor.

Tabela 3 – Resultados de Sínteses de Hardware: Potência

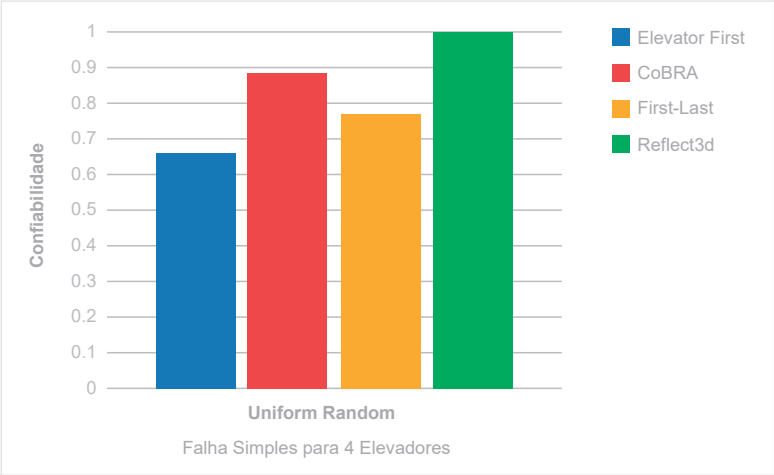
Type (# Ports)	**Potência** (mW) Elevator-First	CoBRA	First-Last	Reflect3d
5-Port	7.23	5.94	6.01	7.53
6-Port	8.58	7.13	7.50	9.45
7-Port	10.04	8.50	8.97	11.98

Fonte: o autor.

5.3 Análise de Confiabilidade

Finalmente, para a comparação de confiabilidade do algoritmo, cada simulação injeta um número fixo de flits (10.000/núcleo), isso permite que sejam feitas comparações para as comunicações entre origem e destino que foram completadas. Em outras palavras, a medida da confiabilidade definida nesta dissertação é a porcentagem de flits entregues com sucesso a camada de destino.

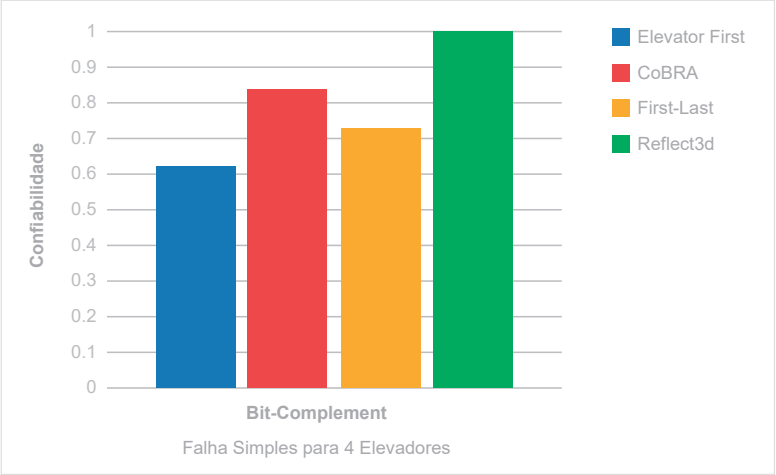
Figura 13 – Confiabilidade para Falhas Simples com 4 TSVs. Modo de tráfego: Uniform Random



Fonte: elaborado pelo autor (2022).

As figuras 13, 14 e 15 mostram a comparação de confiabilidade normalizada para Elevator-First, CoBRA, First-Last e Reflect3d sob o efeito de falhas simples. Podemos observar que para essas falhas que a confiabilidade do Reflect3d se sobressai. O algoritmo CoBRA, apesar de apresentar um bom desempenho, perde pacotes quando ocorrem falhas no elevador mais a leste do roteador. Isso ocorre porque alguns pacotes são descartados na fase de reconfiguração até que o CoBRA retorne à sua condição estável. Já o Reflect3d pode enviar todos os seus pacotes para um elevador saudável, enquanto o CoBRA precisa de pelo menos um elevador colocado na coluna mais a leste ou mais a oeste para evitar a perda de pacotes. Comparando os algoritmos

Figura 14 – Confiabilidade para Falhas Simples com 4 TSVs. Modo de tráfego: Bit-Complement



Fonte: elaborado pelo autor (2022).

First-Last e Elevator-First com o Reflect3d, percebe-se que o Relfect3d também mostra melhor confiabilidade. O Elevator-First mostra o pior cenário para falhas porque não pode se adaptar a falhas em tempo de execução. Além disso, a Elevator-First não possui nenhum mecanismo off-line para reconfigurar a seleção do elevador após o processo de fabricação.

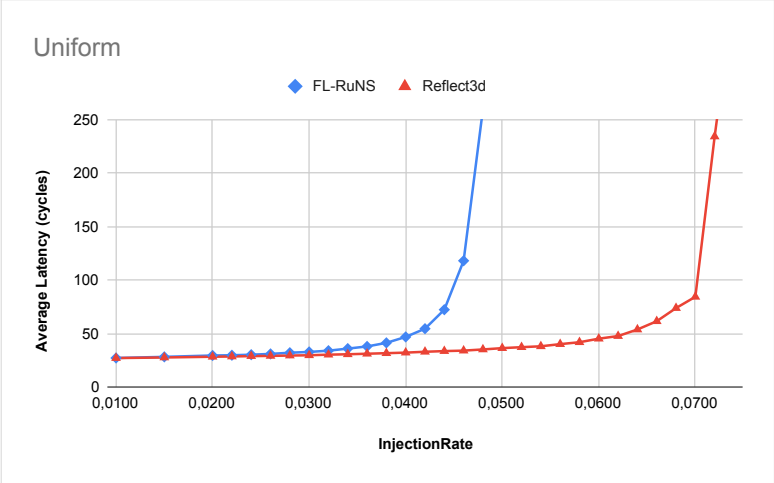
5.4 Comparação de Latência com o Algoritmo FL-Runs

O algoritmo FL-RuNS (COELHO *et al.*, 2019) é um dos algoritmos sobre NoCs 3D mais recentes da literatura. Esse algoritmo usar bem mais recursos de melhorias, estabilidade, escalabilidade e confiabilidade que o algoritmo base desta dissertação, porém para efeitos comparativos e para mostrar o desempenho do algoritmo Reflect3d foi feita uma simulação em termos de latência.

Figura 15 – Confiabilidade para Falhas Simples com 4 TSVs. Modo de tráfego: Shuffle

Fonte: elaborado pelo autor (2022).

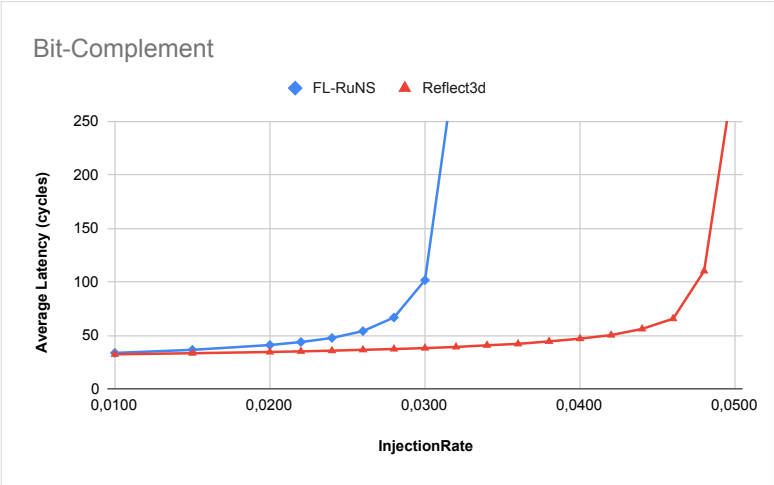
Figura 16 – Latência média de pacote para uma NoC 3D 4x4x4. Modo de tráfego: Uniform Random



Fonte: elaborado pelo autor (2022).

O algoritmo Relfect3d implementado neste trabalho foi criado estendendo a biblioteca sobre NoC 2D Netmaker de código aberto (MULLINS, 2009) adicionando suporte para uma rede 3D. Esta biblioteca também foi usada para implementar o FL-RuNS, o Elevator-First, o First-Last e o CoBRA. O FL-RuNS é um aprimoramento do First-Last. Esse algoritmo apresenta um esquema de roteamento tolerante a falhas para obter 100% de entrega de pacotes sob um conjunto irrestrito de tempo de execução e falhas permanentes no link vertical, é capaz de reconfigurar dinamicamente e progressivamente toda a rede sem perda de pacotes. Sua principal problemática aborda sobre falhas de TSVs na fase de frabricação e na fase de operação.

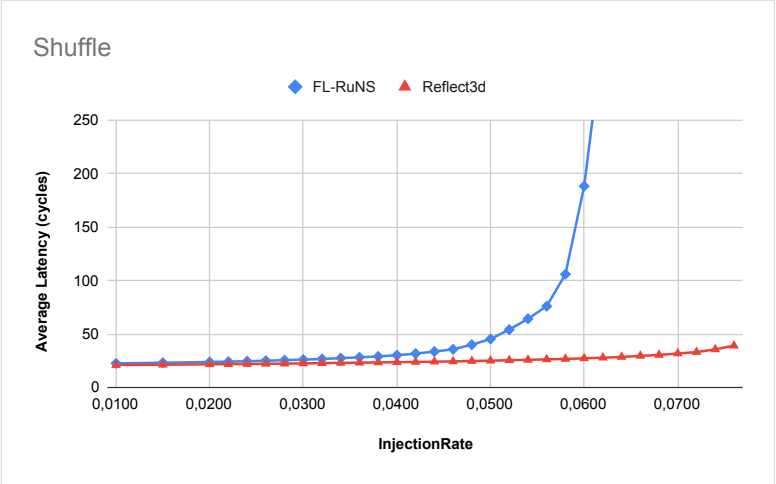
Figura 17 – Latência média de pacote para uma NoC 3D 4x4x4. Modo de tráfego: Bit-Complement



Fonte: elaborado pelo autor (2022).

Porém o Reflect3d ainda apresenta melhor latência média dos pacotes pois usa um

Figura 18 – Latência média de pacote para uma NoC 3D 4x4x4. Modo de tráfego: Shuffle



Fonte: elaborado pelo autor (2022).

número maior de canais virtuais. E como já comentado anteriormente, adicionar VCs não necessariamente é o suficiente para reduzir a latência do pacote. A organização destes canais virtuais em um conjunto de rotas de fuga para escapar de áreas congestionadas ou defeituosas, pode sim reduzir a latência geral da NoC. Nesse sentido, o Reflect3d emprega VNs de forma refletida que permite novas rotas de pacotes devido ao aumento de mais caminhos válido.

6 CONCLUSÕES, CONTRIBUIÇÕES E TRABALHOS FUTUROS

Os benefícios do uso de redes 3D para sistemas integrados em chip é de grande importância para o desenvolvimento de trabalhos futuros com MPSoCs. Chegamos a um gargalo na comunicação desses sistemas e por isso a relevevância nos estudos para aprimoramento da comunicação no chip tem um valor considerável.

Este trabalho propôs um algoritmo de roteamento tolerânte a falhas e validou esse algoritmo através de testes comparativos com outros trabalhos da literatura. Foram realizados testes em modos diferentes de tráfego pela NoC, proporcionado um comparativo mais efiente em termos de latência, área e potência.

Dessa forma o Reflect3d é um algoritmo adaptável tolerante a falhas e projetado para rotear pacotes em NoCs 3D parcialmente e verticalmente conectados. O uso do Reflect3d em trabalhos futuros pode ser de grande valor para estruturas em NoC de alto desempenho e seus testes em um cenário de roteamento de pacotes apresentaram bons resultados quando comparados a outros trabalhos da literatura.

Os resultados experimentais demonstram que o Reflect3d garante a entrega do pacote, desde que haja pelo menos um TSV saudável conectando todas as camadas da NoC 3D. Além disso, a simetria de particionamento dos canais virtuais, juntamente com a sequência de uso da rede virtual, permite que o Reflect3d tolere falhas nos links horizontais. Em outras palavras, o Reflect3d pode fornecer roteamento livre de deadlock e livelock mesmo com a presença de falhas nas conexões verticais e horizontais.

Os resultados da síntese mostram que o Reflect3D possui maior consumo de área e dissipação de energia do que algoritmos similares de última geração. Esse custo adicional de hardware se deve à organização do VC e ao algoritmo de roteamento que oferece adaptabilidade ao tráfego e falhas nos links horizontais. No entanto, os resultados de latência de pacotes para três tráfegos sintéticos revelam que os custos de síntese são compensados, mostrando que o Reflect3d é uma alternativa de roteamento eficaz para sistemas com um número limitado de TSVs, alcançando baixa latência média de pacotes e fornecendo roteamento de pacotes confiável.

Para trabalhos futuros é interessente continuar validando e comparando os algoritmos com a injeção de falhas na NoC e o comportamento dos mesmos em relação aos trabalhos relacionados avaliando a reabilitação da NoC em cenários críticos de falhas.

REFERÊNCIAS

AKBARI, S.; SHAFIEE, A.; FATHY, M.; BERANGI, R. Afra: A low cost high performance reliable routing for 3d mesh nocs. In: **Proceedings of the Design, Automation Test in Europe Conference Exhibition**. [S.l.: s.n.], 2012. p. 332–337.

BEECHU, N. K. R.; Moodabettu Harishchandra, V.; Yernad Balachandra, N. K. High-performance and energy-efficient fault-tolerance core mapping in noc. **Sustainable Computing: Informatics and Systems**, v. 16, p. 1–10, 2017. ISSN 2210-5379. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2210537916302360>.

BORKAR, S. Thousand core chips: A technology perspective. In: **Proceedings of the 44th Annual Design Automation Conference**. New York, NY, USA: Association for Computing Machinery, 2007. (DAC '07), p. 746–749. ISBN 9781595936271. Disponível em: <https://doi.org/10.1145/1278480.1278667>.

BURNS, J.; MCILRATH, L.; KEAST, C.; LEWIS, C.; LOOMIS, A.; WARNER, K.; WYATT, P. Three-dimensional integrated circuits for low-power, high-bandwidth systems on a chip. In: **Proceedings of the IEEE International Solid-State Circuits Conference**. [S.l.: s.n.], 2001. p. 268–269. ISSN 0193-6530.

CHARIF, A.; COELHO, A.; EBRAHIMI, M.; BAGHERZADEH, N.; ZERGAINOH, N. First-last: A cost-effective adaptive routing solution for tsv-based three-dimensional networks-on-chip. **IEEE Transactions on Computers**, v. 67, n. 10, p. 1430–1444, Oct. 2018.

CHARIF, A.; ZERGAINOH, N.; COELHO, A.; NICOLAIDIS, M. Rout3d: A lightweight adaptive routing algorithm for tolerating faulty vertical links in 3d-nocs. In: **Proceedings of the IEEE European Test Symposium**. [S.l.: s.n.], 2017. p. 1–6.

CHEN, L.; ZHU, D.; PEDRAM, M.; PINKSTON, T. M. Simulation of noc power-gating: Requirements, optimizations, and the agate simulator. **Journal of Parallel and Distributed Computing**, v. 95, p. 69–78, 2016. ISSN 0743-7315. Special Issue on Energy Efficient Multi-Core and Many-Core Systems, Part I. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0743731516000356>.

CHEN, Y.-J.; CHANG, W.-W.; LIU, C.-Y.; WU, C.-E.; CHEN, B.-Y.; TSAI, M.-Y. Processors allocation for mpsocs with single isa heterogeneous multi-core architecture. **IEEE Access**, v. 5, p. 4028–4036, Apr. 2017.

CHO, H.; LEEM, L.; MITRA, S. Ersa: Error resilient system architecture for probabilistic applications. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems**, v. 31, n. 4, p. 546–558, 2012.

COELHO, A.; CHARIF, A.; ZERGAINOH, N.; VELAZCO, R. Fl-runs: A high-performance and runtime reconfigurable fault-tolerant routing scheme for partially connected three-dimensional networks on chip. **IEEE Transactions on Nanotechnology**, v. 18, p. 806–818, Jul. 2019.

COELHO, A.; CHARIF, A.; ZERGAINOH, N.-E.; FRAIRE, J.; VELAZCO, R. A soft-error resilient route computation unit for 3d networks-on-chips. In: **2018 Design, Automation Test in Europe Conference Exhibition (DATE)**. [S.l.: s.n.], 2018. p. 1357–1362.

DALLY, W.; TOWLES, B. Route packets, not wires: on-chip interconnection networks. In: **Proceedings of the Design Automation Conference**. [S.l.: s.n.], 2001. p. 684–689. ISBN 1-58113-297-2. ISSN 0738-100X.

DALLY, W. J.; TOWLES, B. P. **Principles and Practices of Interconnection Networks**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2004. ISBN 9780080497808.

DANG, K.; AHMED, A.; ABDALLAH, A.; TRAN, X.-T. Tsv-oct: A scalable online multiple-tsv defects localization for real-time 3-d-ic systems. **IEEE Transactions on Very Large Scale Integration (VLSI) Systems**, v. 28, n. 3, p. 672–685, Mar. 2020.

DAVIS, W.; WILSON, J.; MICK, S.; XU, J.; HUA, H.; MINEO, C.; SULE, A.; STEER, M.; FRANZON, P. Demystifying 3d ics: the pros and cons of going vertical. **IEEE Design Test of Computers**, v. 22, n. 6, p. 498–510, Nov. 2005. ISSN 0740-7475.

DUBOIS, F.; SHEIBANYRAD, A.; PéTROT, F.; BAHMANI, M. Elevator-first: A deadlock-free distributed routing algorithm for vertically partially connected 3d-nocs. **IEEE Transactions on Computers**, v. 62, n. 3, p. 609–615, Mar. 2013.

EBRAHIMI, M.; CHANG, X.; DANESHTALAB, M.; PLOSILA, J.; LILJEBERG, P.; TENHUNEN, H. Dyxyz: Fully adaptive routing algorithm for 3d nocs. In: **Proceedings of the Euromicro International Conference on Parallel, Distributed, and Network-Based Processing**. [S.l.: s.n.], 2013. p. 499–503.

EBRAHIMI, M.; DANESHTALAB, M. Ebda: A new theory on design and verification of deadlock-free interconnection networks. In: **Proceedings of the Annual International Symposium on Computer Architecture**. [S.l.: s.n.], 2017. p. 703–715.

EBRAHIMI, M.; DANESHTALAB, M.; LILJEBERG, P.; TENHUNEN, H. Fault-tolerant method with distributed monitoring and management technique for 3d stacked meshes. In: **Proceedings of the International Symposium on Computer Architecture Digital Systems**. [S.l.: s.n.], 2013. p. 93–98.

EGHBAL, A.; YAGHINI, P.; BAGHERZADEH, N.; KHAYAMBASHI, M. Analytical fault tolerance assessment and metrics for tsv-based 3d network-on-chip. **IEEE Transactions on Computers**, v. 64, n. 12, p. 3591–3604, Dec. 2015. ISSN 0018-9340.

FEERO, B.; PANDE, P. Networks-on-chip in a three-dimensional environment: A performance evaluation. **IEEE Transactions on Computers**, v. 58, n. 1, p. 32–45, Jan. 2009. ISSN 0018-9340.

FEERO, B.; PANDE, P. Networks-on-chip in a three-dimensional environment: A performance evaluation. **IEEE Transactions on Computers**, v. 58, n. 1, p. 32–45, 2009.

FU, Y.; CHEN, Q.; HE, G.; CHEN, K.; LUT, Z.; ZHANG, C.; LI, L. Congestion-aware dynamic elevator assignment for partially connected 3d-nocs. In: . [S.l.: s.n.], 2019. p. 1–5.

GUPTA, S.; GALA, N.; MADHUSUDAN, G.; KAMAKOTI, V. Shakti-f: A fault tolerant microprocessor architecture. In: **2015 IEEE 24th Asian Test Symposium (ATS)**. [S.l.: s.n.], 2015. p. 163–168.

HESHAM, S.; RETTKOWSKI, J.; GOEHRINGER, D.; GHANY, M. A. A. E. Survey on real-time networks-on-chip. **IEEE Transactions on Parallel and Distributed Systems**, v. 28, n. 5, p. 1500–1517, 2017.

JERRAYA, A.; WOLF, W. **Multiprocessor Systems-on-Chips**. 1st. ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2004. ISBN 012385251X.

LOI, I.; MITRA, S.; LEE, T.; FUJITA, S.; BENINI, L. A low-overhead fault tolerance scheme for tsv-based 3d network on chip links. In: **Proceedings of the IEEE/ACM International Conference on Computer-Aided Design**. [S.l.: s.n.], 2008. p. 598–602. ISSN 1092-3152.

MULLINS, R. Netmaker. 2009.

NIAZMAND, B.; AZAD, S.; FLICH, J.; RAIK, J.; JERVAN, G.; HOLLSTEIN, T. Logic-based implementation of fault-tolerant routing in 3d network-on-chips. In: **Proceedings of the IEEE/ACM International Symposium on Networks-on-Chip**. [S.l.: s.n.], 2016. p. 1–8.

PASRICHA, S.; DUTT, N. **On-Chip Communication Architectures: System on Chip Interconnect**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2008. ISBN 012373892X.

PATTI, R. Three-dimensional integrated circuits and the future of system-on-chip designs. **Proceedings of the IEEE**, v. 94, n. 6, p. 1214–1224, Jun. 2006. ISSN 0018-9219.

PAVLIDIS, V. F.; FRIEDMAN, E. G. 3-d topologies for networks-on-chip. **IEEE Transactions on Very Large Scale Integration (VLSI) Systems**, v. 15, n. 10, p. 1081–1090, 2007.

RIJPKEMA, E.; GOOSSENS, K.; RADULESCU, A.; DIELISSEN, J.; MEERBERGEN, J. van; WIELAGE, P.; WATERLANDER, E. Trade offs in the design of a router with both guaranteed and best-effort services for networks on chip. In: **Proceedings of the Design, Automation and Test in Europe Conference and Exhibition**. [S.l.: s.n.], 2003. p. 350–355. ISSN 1530-1591.

SALAMAT, R.; EBRAHIMI, M.; BAGHERZADEH, N.; VERBEEK, F. Cobra: Low cost compensation of tsv failures in 3d-noc. In: **Proceedings of the IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems**. [S.l.: s.n.], 2016. p. 115–120.

SALAMAT, R.; KHAYAMBASHI, M.; EBRAHIMI, M.; BAGHERZADEH, N. A resilient routing algorithm with formal reliability analysis for partially connected 3d-nocs. **IEEE Transactions on Computers**, v. 65, n. 11, p. 3265–3279, Nov. 2016.

SALAMAT, R.; KHAYAMBASHI, M.; EBRAHIMI, M.; BAGHERZADEH, N. Lead: An adaptive 3d-noc routing algorithm with queuing-theory based analytical verification. **IEEE Transactions on Computers**, PP, p. 1–1, 02 2018.

SILVEIRA, J.; MARCON, C.; CORTEZ, P.; BARROSO, G.; FERREIRA, J.; MOTA, R. Scenario preprocessing approach for the reconfiguration of fault-tolerant noc-based mpsocs. **Microprocessors and Microsystems**, v. 40, p. 137–153, 2016. ISSN 0141-9331. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0141933115001180>.

TATAS, K.; SIOZIOS, K.; SOUDRIS, D.; JANTSCH, A. **Designing 2D and 3D Network-on-Chip Architectures**. [S.l.: s.n.], 2014. ISBN 978-1-4614-4273-8.

VAHDATPANAH, F.; ELAHI, M.; KASHI, S.; TAHERI, E.; PATOOGHY, A. 3dep: A efficient routing algorithm to evenly distribute traffic over 3d network-on-chips. In: **2019 27th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP)**. [S.l.: s.n.], 2019. p. 237–241.

APÊNDICE A – PRINCIPAIS TRABALHOS RELACIONADOS A ESTA DISSERTAÇÃO

Neste apêndice apresentaremos os principais trabalhos que se relacionam a esta dissertação. Abaixo as tabelas esboçam o nome do trabalho, seus autores, a principal proposta apresentada pelo trabalho bem como suas principais problemáticas.

Trabalho Relacionado	Altores	Proposta Apresentada	Principal Problemática
Rout3D: A Lightweight Adaptive Routing Algorithm for Tolerating Faulty Vertical Links in 3D-NoCs	Amir Charif, Nacer-Eddine Zergainoh, Alexandre Coelho, Michael Nicolaidis	Introduz um algorítimo de roteamento eficiente e altamente resiliente, visando as NoCs 3D parcialmente e verticalmente conectados, algorítimo chamado de “Rout3D” (Routed), esse algorítimo garante a entraga dos pacotes sob condições extremas com um níumero limitado de links verticais || Também foi proposto um algorítimo de localização de elevadores válidos, visto que o algorítimo proposto é validado nesses casos, chamado de “Elevator Compass”, que localiza o elevador mais próximo através de uma combinação de 4 bits para cada elevador disponível na rede. Esses bits podem ser reconfigurados toda vez que o estado da rede muda devido a falhas. Além disso, esse método não é específico ao Rout3D e pode ser combinado com qualquer outro algoritmo de roteamento.	Alto custo para Vários TSVs, Nessecidade de redução desses TSVs, falhas de fabricação e operação.
First-Last: A Cost-Effective Adaptive Routing Solution for TSV-Based Three-Dimensional Networks-on-Chip	Amir Charif, Alexandre Coelho, Masoumeh Ebrahimi, Nader Bagherzadeh, Nacer-Eddine Zergainoh	Versão estendida e aprimmorada do Rout3D. De forma resumida acrescenta: Identificação formal da condição a ser atendida pelo posicionamento do TSV para garantir a conectividade total; usa um VC adicional ao longo da dimensão Z, não apenas para melhorar o desempenho, mas também para aumentar a resiliência; atribuição de elevadores foi aprimorada para permitir mais adaptabilidade ao tempo de execução; resultados de síntese de hardware e comparação com soluções de roteamento mais recentes.	Alto custo para Vários TSVs, Nessecidade de redução desses TSVs, falhas de fabricação e operação.
LEAD: An Adaptive 3D-NoC Routing Algorithm with Queuing--Theory Based Analytical Verification	Ronak Salamat, Misagh Khayambashi, Masoumeh Ebrahimi, Nader Bagherzadeh	Importância dos algoritmos de roteamento adaptável nessas arquiteturas, especialmente no balanceamento de carga para trafegos não uniformes, propor um algoritmo de roteamento adaptável para 3D-NoC parcialmente conectado que forneça melhor desempenho em comparação com os algoritmos de roteamento publicados anteriormente	Ausência de TSVs em determinados pontos resultando em mais carga nos TSVs atuais; desempenho da rede e o envelhecimento do TSV.

Trabalho Relacionado	Altores	Proposta Apresentada	Principal Problemática
A Dynamic Sufficient Condition of Deadlock-Freedom for High-Performance Fault-Tolerant Routing in Networks-on-Chips	Amir Charif, Alexandre Coelho, Nacer-Eddine Zergainoh, Michael Nicolaidis	Introduz uma nova condição para deadlock-freedom que melhora bastante as restrições impostas pelos métodos clássicos de prevenção de impasse baseados em VC. Com base nessa condição, é apresentada uma estrutura de design de algoritmo de roteamento intuitivo e independente da topologia, que compreende o conjunto mínimo de regras que os algoritmos de roteamento devem cumprir para satisfazer tal condição de deadlock-freedom e é apresentada e usada para construir algoritmos eficientes para NoCs 2D e 3D.	Particionamento estático excessivamente restritivo de VCs (Canais Virtuais)
FL-RuNS: A High-Performance and Runtime Reconfigurable Fault-Tolerant Routing Scheme for Partially Connected Three-Dimensional Networks on Chip	Alexandre Coelho, Amir Charif, Nacer-Eddine Zergainoh, Raoul Velazco	Aprimoramento do First-Last. É apresentado o FL-RuNS, um esquema de roteamento tolerante a falhas para obter 100% de entrega de pacotes sob um conjunto irrestrito de tempo de execução e falhas permanentes no link vertical, é capaz de reconfigurar dinamicamente e progressivamente toda a rede sem perda de pacotes.	falhas de TSVs na fase de frabricação e na fase de operação
EbDa: A new theory on design and verification of deadlock-free interconnection networks	Masoumeh Ebrahimi, Masoud Daneshtalab	É mostrado um roteiro para projetar algoritmos de roteamento sem conflito em uma rede de comutação wormhole. São apresentados três teoremas (sessão 3) que removem todas as dependências cíclicas no gráfico de dependência de canal e sugerem diretamente opções de roteamento sem deadlock. Esse conjuto de teoremas é chamado de EbDa.	Encontrar um "gráfico" acíclico, ou seja, sem deadlock

Trabalho Relacionado	Altores	Proposta Apresentada	Principal Problemática
Survey on Real-Time Networks-on-Chip	Salma Hesham, Jens Rettkowski, Diana Goehringer, and Mohamed A. Abd El Ghany	Discute as implicações dos RTAs no design de NoC. As contribuições existentes que cobrem aspectos garantidos de suporte de qualidade de serviço (QoS), adaptabilidade e eficiência de energia em NoCs em tempo real (RT-NoCs) são revisados. O artigo examina ainda as metodologias de avaliação e medidas de desempenho experimental para RT-NoCs. Além disso, este artigo fornece uma visão dos pontos de pesquisa abertos nesse campo	Real-time applications (RTAs)
Congestion-Aware Dynamic Elevator Assignment for Partially Connected 3D-NoCs	Yuxiang Fu, Qinyu Chen, Guoqiang He, Kai Chen, Zhonghai Lu, Chuan Zhang, Li Li	Análise do gargalo de desempenho das NoCs 3D parcialmente conectados. Um esquema de atribuição dinâmica de elevador com reconhecimento de congestionamento, chamado de Congestion-aware Dynamic elevator Assignment (CDA). O critério de atribuição é escolhido como a soma ponderada do atraso do roteador e o quadrado da utilização do buffer ao longo do caminho da origem até o final do elevador atribuído na camada de destino. Entre os caminhos disponíveis, selecionamos o caminho com o critério de atribuição mínimo e o elevador ao longo do caminho é atribuído ao pacote.	Alto tráfego em um elevador
A Resilient Routing Algorithm with Formal Reliability Analysis for Partially Connected 3D-NoCs	Ronak Salamat, Misagh Khayambashi, Masoumeh Ebrahimi, Nader Bagherzadeh	Algoritmo de roteamento eficiente para 3D-NoCs parcialmente conectados, chamado East-Then-West (ETW). Esse algoritmo é confiável desde que haja pelo menos um TSV na coluna mais a leste, enquanto o desempenho pode ser aprimorado aumentando o número de TSVs. O algoritmo ETW é extremamente leve. Ou seja, requer apenas um canal virtual ao longo da dimensão Y. Esse algoritmo fornece adaptabilidade para entregar pacotes, de preferência usando os caminhos mais curtos	Ausência de TSVs em determinados pontos resultando em mais carga nos TSVs atuais; desempenho da rede e o envelhecimento do TSV.

APÊNDICE B – DADOS DE LATÊNCIA NOS TRÊS MODOS DE TRÁFEGO

Tabela 4 – Uniform Randon

InjectionRate	Elevator-First	CoBRA	First-Last	Reflect3d
0,0100	28,53	30,29	26,75	27,02
0,0150	29,57	32,22	27,95	27,57
0,0200	31,04	34,07	29,23	28,26
0,0220	31,7	35,39	29,67	28,50
0,0240	32,36	36,95	30,07	28,86
0,0260	33,35	39,37	30,9	29,05
0,0280	34,27	42,67	32,24	29,39
0,0300	35,27	50,8	33,07	29,64
0,0320	36,59	65,79	34,44	30,21
0,0340	38,6	80,46	36,25	30,57
0,0360	40,55	156,92	38,48	31,08
0,0380	44,3	493,59	40,9	31,59
0,0400	48,16	-	46,51	32,11
0,0420	56,52	-	53,55	32,91
0,0440	69,82	-	64,97	33,54
0,0460	101,64	-	109	33,98
0,0480	176,91	-	216,72	35,08
0,0500	464,58	-	367,9	36,30
0,0520	-	-	-	37,21
0,0540	-	-	-	38,03
0,0560	-	-	-	40,05
0,0580	-	-	-	41,84
0,0600	-	-	-	45,15
0,0620	-	-	-	47,60
0,0640	-	-	-	53,73
0,0660	-	-	-	61,47
0,0680	-	-	-	73,76
0,0700	-	-	-	84,29
0,0720	-	-	-	234,17
0,0740	-	-	-	369,61
0,0760	-	-	-	-

Fonte: o autor.

Tabela 5 – Bit Complement

InjectionRate	Elevator-First	CoBRA	First-Last	Reflect3d
0,0100	33,41	36,66	33,68	32,21
0,0150	36,19	44,48	36,67	33,23
0,0200	40,56	50,2	41,28	34,37
0,0220	42,7	63,7	43,57	34,98
0,0240	46,37	80,18	47,28	35,61
0,0260	52,21	171,7	53,36	36,42
0,0280	63,67	357,75	64,88	37,16
0,0300	85,32	-	89,66	38,02
0,0320	156,86	-	195,19	39,09
0,0340	476,54	-	590,44	40,64
0,0360	-	-	-	41,98
0,0380	-	-	-	44,28
0,0400	-	-	-	46,86
0,0420	-	-	-	50,18
0,0440	-	-	-	55,95
0,0460	-	-	-	65,52
0,0480	-	-	-	110,14
0,0500	-	-	-	300,76
0,0520	-	-	-	-
0,0540	-	-	-	-
0,0560	-	-	-	-
0,0580	-	-	-	-
0,0600	-	-	-	-
0,0620	-	-	-	-
0,0640	-	-	-	-
0,0660	-	-	-	-
0,0680	-	-	-	-
0,0700	-	-	-	-
0,0720	-	-	-	-
0,0740	-	-	-	-
0,0760	-	-	-	-

Fonte: o autor.

Tabela 6 – Shuffle

InjectionRate	Elevator-First	CoBRA	First-Last	Reflect3d
0,0100	21,39	22,87	21,46	20,92
0,0150	21,89	23,41	22,01	21,24
0,0200	22,41	24,09	22,56	21,62
0,0220	22,58	24,39	22,74	21,76
0,0240	22,8	24,68	22,96	21,92
0,0260	23,03	25,15	23,24	22,08
0,0280	23,39	25,6	23,56	22,28
0,0300	23,7	26,21	23,89	22,48
0,0320	24,01	26,92	24,22	22,69
0,0340	24,51	27,98	24,74	22,95
0,0360	25	28,85	25,35	23,13
0,0380	25,52	30,29	25,95	23,40
0,0400	25,97	32,72	26,53	23,59
0,0420	26,47	36,65	27,06	23,83
0,0440	27,22	49,52	27,82	24,10
0,0460	28,06	67	28,86	24,33
0,0480	29,54	92,88	30,37	24,69
0,0500	31,05	157	32,66	24,99
0,0520	33,26	326,34	34,18	25,42
0,0540	37,12	-	37,22	25,82
0,0560	42,27	-	45,82	26,24
0,0580	49,99	-	59,32	26,64
0,0600	67,89	-	89,3	27,20
0,0620	151,81	-	202,47	27,81
0,0640	270,02	-	310,01	28,48
0,0660	-	-	-	29,51
0,0680	-	-	-	30,36
0,0700	-	-	-	31,73
0,0720	-	-	-	33,07
0,0740	-	-	-	35,57
0,0760	-	-	-	39,01

Fonte: o autor.

Tabela 7 – Comparativo com FL-Runs - Modo de Tráfego: Uniform Randon

InjectionRate	FL-Runs	Reflect3d
0,0100	27,09	27,02
0,0150	28,16	27,57
0,0200	29,44	28,26
0,0220	29,67	28,50
0,0240	30,18	28,86
0,0260	30,91	29,05
0,0280	32,02	29,39
0,0300	32,76	29,64
0,0320	33,92	30,21
0,0340	35,87	30,57
0,0360	37,89	31,08
0,0380	41,24	31,59
0,0400	46,78	32,11
0,0420	54,48	32,91
0,0440	72,24	33,54
0,0460	117,99	33,98
0,0480	259,7	35,08
0,0500	-	36,30
0,0520	-	37,21
0,0540	-	38,03
0,0560	-	40,05
0,0580	-	41,84
0,0600	-	45,15
0,0620	-	47,60
0,0640	-	53,73
0,0660	-	61,47
0,0680	-	73,76
0,0700	-	84,29
0,0720	-	234,17
0,0740	-	369,61
0,0760	-	

Fonte: o autor.

Tabela 8 – Comparativo com FL-Runs - Modo de Tráfego: Bit-Complement

InjectionRate	FL-Runs	Reflect3d
0,0100	33,51	32,21
0,0150	36,38	33,23
0,0200	40,89	34,37
0,0220	43,68	34,98
0,0240	47,5	35,61
0,0260	53,89	36,42
0,0280	66,64	37,16
0,0300	101,57	38,02
0,0320	308,79	39,09
0,0340	-	40,64
0,0360	-	41,98
0,0380	-	44,28
0,0400	-	46,86
0,0420	-	50,18
0,0440	-	55,95
0,0460	-	65,52
0,0480	-	110,14
0,0500	-	300,76
0,0520	-	-
0,0540	-	-
0,0560	-	-
0,0580	-	-
0,0600	-	-
0,0620	-	-
0,0640	-	-
0,0660	-	-
0,0680	-	-
0,0700	-	-
0,0720	-	-
0,0740	-	-
0,0760	-	-

Fonte: o autor.

Tabela 9 – Comparativo com FL-Runs - Modo de Tráfego: Shuffle

InjectionRate	FL-Runs	Reflect3d
0,0100	22,5	20,92
0,0150	23,08	21,24
0,0200	23,8	21,62
0,0220	24,14	21,76
0,0240	24,5	21,92
0,0260	24,95	22,08
0,0280	25,48	22,28
0,0300	26,02	22,48
0,0320	26,58	22,69
0,0340	27,4	22,95
0,0360	28,12	23,13
0,0380	29,01	23,40
0,0400	30,13	23,59
0,0420	31,62	23,83
0,0440	33,55	24,10
0,0460	35,63	24,33
0,0480	39,93	24,69
0,0500	45,28	24,99
0,0520	54,03	25,42
0,0540	64,18	25,82
0,0560	75,93	26,24
0,0580	105,83	26,64
0,0600	188,18	27,20
0,0620	327,93	27,81
0,0640	-	28,48
0,0660	-	29,51
0,0680	-	30,36
0,0700	-	31,73
0,0720	-	33,07
0,0740	-	35,57
0,0760	-	39,01

Fonte: o autor.