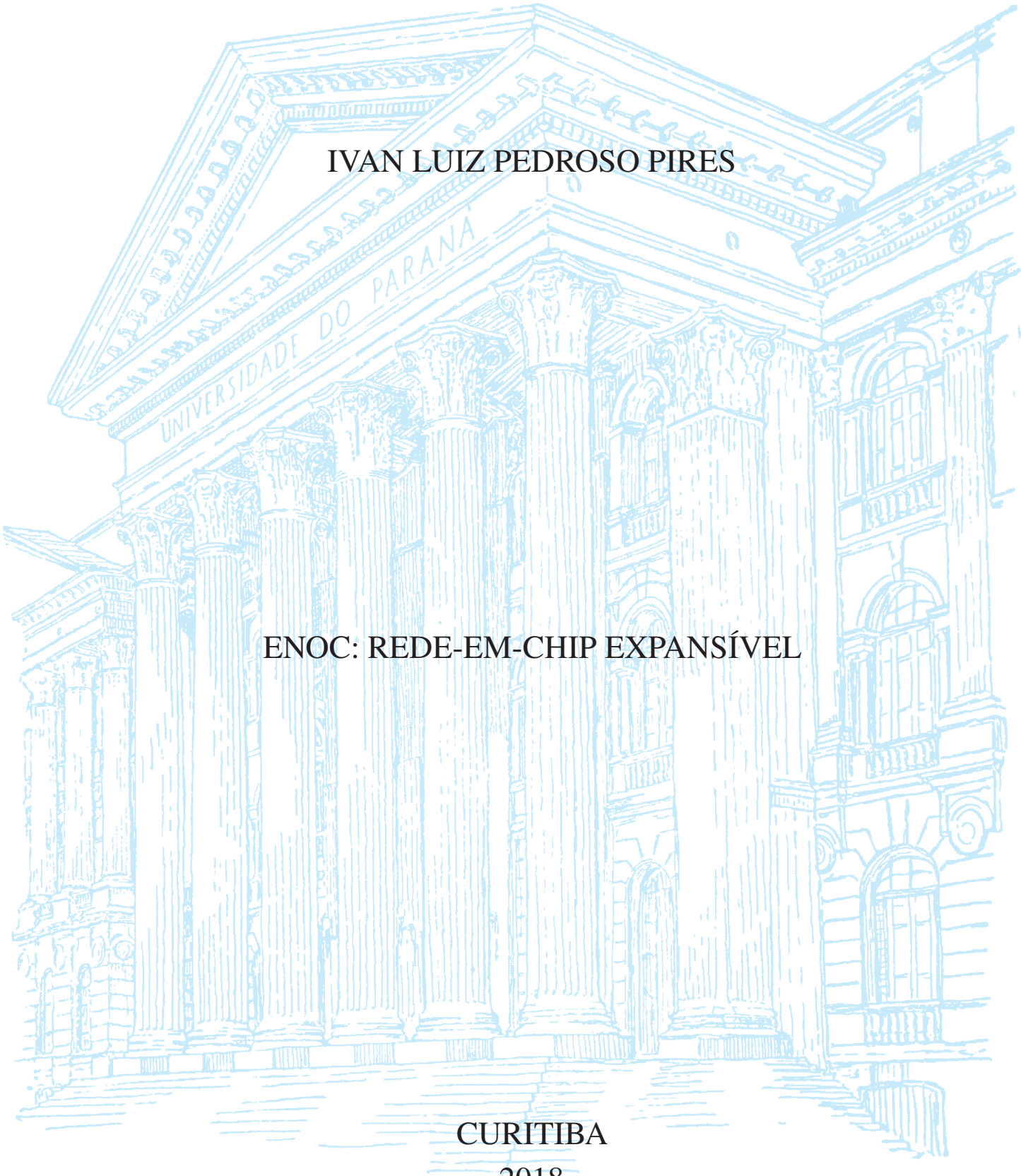
UNIVERSIDADE FEDERAL DO PARANÁ

IVAN LUIZ PEDROSO PIRES



ENOC: REDE-EM-CHIP EXPANSÍVEL

CURITIBA
2018

IVAN LUIZ PEDROSO PIRES

ENOC: REDE-EM-CHIP EXPANSÍVEL

Tese apresentada ao curso de Pós-Graduação em Informática, setor de Ciências Exatas, da Universidade Federal do Paraná, como requisito parcial à obtenção do título de Doutor em Informática..

Orientador: Prof. Dr. Luiz Carlos Pessoa Albini.

Coorientador: Prof. Dr. Marco Antonio Zanata Alves.

CURITIBA
2018

Catalogação na Fonte: Sistema de Bibliotecas, UFPR
Biblioteca de Ciência e Tecnologia

P667e Pires, Ivan Luiz Pedroso
ENoC: Rede-em-Chip Expansível / Ivan Luiz Pedroso Pires. – Curitiba, 2018.

Tese - Universidade Federal do Paraná, Setor de Ciências Exatas, Programa de Pós-Graduação em Informática, 2018.

Orientador: Luiz Carlos Pessoa Albini. Coorientador: Marco Antonio Zanata Alves.

1. Sistemas programáveis em chip. 2. Redes-em-chip. 3. criptografia. I. Universidade Federal do Paraná. II. Centeno, Jorge Antonio Silva. III. Alves, Marco Antonio Zanata. VI. Título.

CDD: 005.8

Bibliotecária: Vanusa Maciel CRB- 9/1928

MINISTÉRIO DA EDUCAÇÃO
SETOR SETOR DE CIENCIAS EXATAS
UNIVERSIDADE FEDERAL DO PARANÁ
PRÓ-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO INFORMÁTICA

TERMO DE APROVAÇÃO

Os membros da Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em INFORMÁTICA da Universidade Federal do Paraná foram convocados para realizar a arguição da tese de Doutorado de **IVAN LUIZ PEDROSO PIRES** intitulada: **ENoC: Rede-em-Chip Expansivel**, após terem inquirido o aluno e realizado a avaliação do trabalho, são de parecer pela sua APROVAÇÃO no rito de defesa.

A outorga do título de doutor está sujeita à homologação pelo colegiado, ao atendimento de todas as indicações e correções solicitadas pela banca e ao pleno atendimento das demandas regimentais do Programa de Pós-Graduação.

Curitiba, 10 de Agosto de 2018.

LUIZ CARLOS PESSOA ALBINI
Presidente da Banca Examinadora

GUSTAVO ALBERTO GIMENEZ LUGO
Avaliador Externo

BRUNO MÜLLER JUNIOR
Avaliador Externo

EDUARDO TODT
Avaliador Interno

PHILIPPE OLIVIER ALEXANDRE NAVAUX
Avaliador Externo

MARCO ANTONIO ZANATA ALVES
Coorientador

MINISTÉRIO DA EDUCAÇÃO - UNIVERSIDADE FEDERAL DO PARANÁ - Setor de Ciências Exatas - Coordenação do Programa de Pós-Graduação em Informática

A minha família.

Agradecimentos

Aos meus mentores Luiz Carlos Pessoa Albini e Marco Antonio Zanata Alves. Muito além de meus orientadores, eu os considero amigos e mestres, que me ensinaram a trilhar os caminhos sinuosos da pesquisa científica. Muito aprendi, muito além da ciência. São verdadeiros faróis que me nortearam quando mais precisei e em meio a vastidão escura e incerta da busca da tese lá estavam para me conduzir, meu muito obrigado!

Agradeço também aos professores Gustavo Lugo, Bruno Muller, Eduardo Todt e Philippe Navaux por aceitarem a ser a banca da defesa da minha Tese e contribuir com sugestões, ideias e novas visões.

Ao meu querido Pai, Jorge Luiz de Souza Pires e minha querida mãe Margarethe Pedroso Lino Pires por nunca terem medido esforços para me oferecer a melhor educação possível, concedendo oportunidades que nunca tiveram e muitas vezes abrindo a mão de suas merecidas vontades em prol do sucesso dos filhos. Reconheço cada esforço, trabalho e renúncia que tiveram por mim, muito obrigado. Farei o mesmo com meus filhos.

A minha amável família de Cáceres, Colíder e Curitiba que estão sempre de braços abertos prontos para me amparar a cada dificuldade e celebrar cada vitória. Vocês são meu Lar!

A minha querida esposa Elaine Alves da Rocha Pires, luz da minha vida! Agradeço por ser minha grande parceira nessa pesquisa, seja na escrita, na codificação, nas ideias, nos testes, nas angústias, alegrias, enfim, grande parte da minha Tese devo a você. Te amo!

Agradeço também aos meus novos amigos que conquistei durante essa passagem por Curitiba e que levarei para sempre em minhas lembranças. Aos amigos que estão e estiveram no grupo Curitibanos/MT por todos os minutos de alegria, risadas, tristezas, frustação, comemoração, passeios, ensaios, dúvidas, discussão, reflexões, enfim muito momentos de amizades que passamos juntos. Vocês foram demais e marcaram momentos inesquecíveis pra mim. Aos amigos do grupo Refeições/UFPR do Hangout tornar as horas do café e os almoços mais agradáveis e divertidos.

Aos amigos do Laboratório de Sistemas Embarcados, por serem sempre solícitos em me ajudar sejam com palavras de apoio, boas conversas, apoio com prévias de apresentação e até na programação. Faço um agradecimento especial ao Emmanuell que, embora tenha convivido pouco tempo, me ajudou eficazmente com trechos de códigos, com o cluster e dicas fundamentais para a defesa da tese.

E aos meus amigos e colegas da unemat de Colíder e Sinop, que sempre me deram todo o suporte necessário sem exitar em qualquer momento, para que pudesse cursar e concluir o Doutorado sem preocupações com a carreira docente, mesmo quando fatos marcantes aconteciam em nossa instituição.

Ao Programa de Pós-graduação em Informática por propiciar esta evolução em minha vida profissional, enquanto houver programas de qualidade como este, ainda resta esperança para a pesquisa no Brasil. Agradeço também á Capes por suportar a pesquisa que realizei por meio de bolsa de pesquisa.

A Universidade do Estado de Mato Grosso que me permitiu afastar parcialmente das atividades docente para dedicar integralmente ao curso de Doutorado. Estes três anos concedidos foram essenciais e vitais para que pudesse concluir esta etapa em minha formação. Retribuirei todo este investimento com ensino e pesquisa de qualidade para o nosso estado de Mato Grosso.

E por fim, agradeço a Deus pelo dom da vida.

RESUMO

Os sistemas multiprocessados integrados em chip têm emergido como uma importante tendência para projetos de sistemas em chip. Estes sistemas são formados por vários elementos de processamento conectados originalmente por um barramento compartilhado. Este barramento possui restrições à crescente integração de mais elementos de processamento em um único chip, pois não permite a comunicação paralela e à medida que os elementos aumentam o barramento apresenta menor desempenho na comunicação devido a capacidade fixa. A rede em chip, do inglês *Network-on-Chip* (NoC), é uma alternativa ao barramento que permite a comunicação paralela e escalável entre os diferentes elementos de processamento de um chip. Tradicionalmente, a NoC é composta por interligações metálicas entre os roteadores e cada roteador é ligado a um elemento de processamento, a comunicação acontece por encaminhamento de pacotes seguindo um determinado algoritmo de roteamento. Esta comunicação pode ser estendida de ligações metálicas para ligações sem fio principalmente para mitigar a latência resultante dos diversos saltos necessários para comunicar elementos de processamento de um chip, em especial dos mais distantes, uma vez que na comunicação sem fio o pacote é transmitido com apenas um salto. Entretanto, há sobrecustos em utilizar esta tecnologia, e por isto várias pesquisas abordam a interligação de apenas regiões do chip, e não todos os elementos. Mesmo com a evolução das formas de comunicação em um chip, a capacidade de um sistema em chip estava limitada aos seus elementos inseridos em momento de fabricação. Esta tese apresenta a ENoC, uma rede em chip expansível capaz de interligar sistemas em chip distintos reconfigurando-se para oferecer uma visão única de sistema com processamento paralelo distribuído por passagem de mensagem. A arquitetura e a comunicação na ENoC são apresentadas juntamente com uma discussão sobre o uso de sistema operacional e organização da memória. A avaliação é realizada por meio de simulações e análise de desempenho. A segurança da comunicação entre os chips é discutida e sistemas de criptografias são avaliados para manter a confidencialidade da informação. Com os resultados dos experimentos concluímos que a ENoC é uma abordagem adequada para a expandir os recursos entre chips e que cada sistema de criptografia possui vantagens e desvantagens próprias para proteger a comunicação sem fio entre as ENoCs, e a escolha de qual criptossistema é uma decisão de projeto.

Palavras-chave: sistema em chip, rede em chip, criptografia.

ABSTRACT

Multiprocessor Systems-on-Chip has emerged as an important trend for System-on-Chip designs. These systems consists in several processing elements interconnected, originally, by a shared bus. This bus has restrictions to the increasing integration of many processing elements in a single chip, due to does not allow the parallel communication and as the elements increase the bus presents fewer communication performance because its capacity is fixed. The Network-on-Chip (NoC) is an alternative to the bus that allows parallel and scalable communication among all processing elements on chip. Traditionally, the NoC is made up of metallic wired interconnecting the routers and each router is connected to a processing element, the communication is performed by packets routing following a routing algorithm. This communication may be extended from metal wired links to wireless links, mainly to mitigate the latency from several needed hops to communicate processing elements, in special, the more distant ones, once in wireless communication the packet is transmitted by a single hop. However, there are additional costs in using this technology, and for this reason several researches focus on interconnecting only chip regions, not all elements. Even with the evolution of communication on NoC, the capacity of a system-on-chip was limited to its elements at manufacture time. This thesis presents the ENoC, an Expansible Network-on-Chip capable of interconnecting distinct reconfigurable SoCs to provide a single system view with parallel processing distributed by message passing. The architecture and communication of ENoC are presented within a discussion of operational system and memory organization. The evaluation is performed by simulation and performance analysis. The security of inter-chip communication is discussed and cryptography systems are evaluated to offer a confidentiality of the information. With the results, we conclude that the ENoC is a suitable approach to expand the resources between chips and that each encryption system has its own advantages and disadvantages in order to protect the wireless inter-chip communication, in such way, the choice of which criptosystem is a design decision.

Keywords: system-on-chip, network-on-chip, cryptography.

Lista de Figuras

2.1 Representação de *a*) uma NoC com 16 recursos em malha bidimensional e *b*) de um roteador para NoC (Kumar et al. (2002)). 20
2.2 Topologias para NoC (Chen et al. (2012)). 22

3.1 Expansão com a ENoC.. 26
3.2 Roteador ENoC.. 27
3.3 Pacote ENoC. 28
3.4 Estrutura de endereço de origem e destino.. 28
3.5 *Handshake* inicial na ENoC. 30

5.1 Formato do traço de entrada para um PE específico. 39
5.2 Exemplo de entrada de traço. 39
5.3 Exemplo de traço para obtenção do tempo de processamento.. 40

6.1 Ilustração da instrumentação realizada.. 43
6.2 Sobrecarga dos sistemas de interconexão e da ENoC em função do crescimento da carga útil em Bytes. 47
6.3 Proporção do tamanho das mensagens das aplicações NPB. 48
6.4 Causas da sobrecarga para relação sistema de interconexão e aplicação NPB.. . . 49
6.5 Ilustração dos sistemas avaliados.. 50
6.6 Trecho de arquivo de traço da aplicação BT. 51
6.7 Tempo de execução sem tempo de processamento - parte 1. 52
6.8 Tempo de execução sem tempo de processamento - parte 2. 53
6.9 Tempo de execução com variação do número de PEs divididos em 04 sistemas executando a carga de trabalho LU.. 54
6.10 Tempo de execução com variação do número de PEs divididos em 04 sistemas executando a carga de trabalho SP. 55
6.11 Tempo de execução da ENoC com variação do número de sistemas executando 32 tarefas. 55
6.12 Tempo de execução da ENoC com variação do número de sistemas executando 320 tarefas. 56
6.13 Tempo de execução com tempo de processamento - parte 1.. 57
6.14 Tempo de execução com tempo de processamento - parte 2.. 57

7.1 Ataque à confidencialidade na ENoC. 60

7.2 *Handshake* inicial para comunicação inter-chip com ECC. 62
7.3 Comparação do tempo de execução para cifrar. 65
7.4 Comparação do tempo de execução para decifrar. 66
7.5 Impacto da segurança nas aplicações NAS-NPB. 66

A.1 Exemplo de cenário para comunicação baseada em MPI. 82
A.2 Primitivas MPI. 83

Lista de Tabelas

3.1 Sistema operacional e gerenciamento de memória para a ENoC. 30

4.1 Comparação com trabalhos relacionados 34

5.1 Comparação dos simuladores para NoC. 36

6.1 Primitivas MPI usadas nas aplicações NPB classe *A*. 45
6.2 Parâmetro das redes. 46
6.3 Sobrecarga das aplicações NPB. 48
6.4 Resumo dos parâmetros de simulação. 50

7.1 Tamanho total dos parâmetros de diferentes curvas. 62
7.2 Montante das mensagens NPB pelo tamanho. 67

B.1 Comparação dos tamanhos das chaves em bits entre os criptossistemas AES, ECC e RSA.. 86
B.2 Etapas da cifra e decifra com AES.. 88
B.3 Etapas da ECC. 91
B.4 Etapas da cifragem com ECIES. 92
B.5 Etapas da decifragem com ECIES. 92

Lista de Acrônimos

3PIP	Third-Party Intellectual Property
AES	Advanced Encryption System
BT	Block Tri-diagonal
CBC	Cipher Block Chaining
CDMA	Code Divison Multiple Access
CG	Conjugate Gradient
CRC	Cyclic Redundancy Check
ECC	Elliptic Curve Cryptography
ECDH	Elliptic Curve Diffie-Hellman
ECDSA	Elliptic Curve Digital Signature Algorithm
ECIES	Elliptic Curve Integrated Encryption Scheme
EDR	Enhanced Data Rate
ECMQV	Elliptic Curve Menezes-Qu-Vanstone
ENoC	Expansible Network-on-Chip
Eth	Ethernet
FCS	Frame check sequence
Flit	Flow-control Unit
FT	Fast Fourier Transform
GHASH	Galois Hash
GPU	Graphical Process Unit
HPC	High Performance Computing
HT	Hardware Trojan
IB	InfiniBand
IoT	Internet of Things
IS	Integer Sort
IP	Intellectual Property
KDF	Key Derivate Function
LU	Lower-Upper Symmetric Gauss-Seidel
MAC	Message Authentication Code
MANET	Mobile Ad-hoc Network
MCNoC	Multi-Cluster Network-on-Chip
MG	Multigrid
MIMO	Multiple-Input and Multiple-Output

MIRACL	Multiprecision Integer and Rational Arithmetic Cryptographic Library
MPE	Multi Processing Environment
MPI	Message Passing Interface
MPICH	MPI Chameleon
MPSoC	Multiprocessor System-on-Chip
NAS	Numerical Aerodynamic Simulation
NoC	Network-on-Chip
NIST	National Institute of Standards and Technology
NPB	NAS Parallel Benchmark
NUMA	Non-Uniform Memory Access
OFDM	Orthogonal Frequency-Division Multiplexing
OFDMA	Orthogonal Frequency-Division Multiple Access
OOK	On-Off Keying
QAM	Quadrature Amplitude Modulation
PAL	Protocol Adaptation Layer
PE	Processing Element
PKG	Private Key Generator
PUF	Physical Unclonable Functions
QoS	Quality-of-Service
RDMA	Remote Direct Memory Access
RSA	Rivest Shamir and Adleman
SGI	Silicon Graphics International
SHA	Secure Hash Algorithm
SMP	Symmetric Multi-Processor
SSOR	Symmetric Successive OverRelaxation
SO	Sistema Operacional
SP	Scalar Pentadiagonal
SPIN	Scalable Programmable Interconnection Network
SoC	System-on-Chip
TCP	Transfer Control Protocol
UML	Unified Modeling Language
WiGig	Wireless Gigabit
WiHD	Wireless HD
WiNoC	Wireless Network-on-Chip
WI	Wireless Interface
WI-CDMA	Wireless Interconnection with Code Division Multiple Access
WI-Token	Wireless Interconnection Token-based
YAML	Yet Another UML front end

Sumário

1 Introdução 15
2 Rede em Chip 19
2.1 Arquitetura e comunicação 19
2.1.1 Elementos da rede em chip 19
2.1.2 Algoritmos de roteamento 21
2.1.3 Protocolo de comunicação 21
2.2 Topologias. 22
2.3 Redes em Chip sem fio 23
2.4 Considerações sobre o capítulo 24
3 ENoC: The Expansible Network-on-Chip 25
3.1 Arquitetura 25
3.2 Camada Física e Enlace de dados. 26
3.3 Camada de rede 27
3.4 Camada de transporte 29
3.5 Reconfiguração de expansão 29
3.6 Sistema operacional e organização da memória 30
3.7 Considerações sobre o capítulo 31
4 Trabalhos Relacionados 32
4.1 Considerações sobre o capítulo 34
5 Ambiente de Simulação 35
5.1 Simulador Noxim 36
5.2 Modificações no simulador 37
5.3 Suporte à entrada de traços de aplicação 38
5.4 Suporte a comunicação MPI 39
5.5 Suporte ao tempo de processamento 40
5.6 Consideração do capítulo 40
6 Avaliação de desempenho da ENoC 42
6.1 Carga de trabalho 42
6.1.1 Geração da carga de trabalho 42
6.1.2 Aplicações NPB. 44
6.2 Sistemas de interconexão 45
6.2.1 Ethernet 45

6.2.2 WiGig . . . 46
6.2.3 Infiniband . . . 46
6.2.4 Wi-CDMA e Wi-Token . . . 46
6.2.5 Comparação entre os sistemas de interconexão . . . 46
6.3 Correlação de sobrecarga dos *Benchmarks* . . . 47
6.4 Métricas e Parâmetros . . . 49
6.4.1 Parâmetros . . . 50
6.4.2 Métricas . . . 51
6.5 Avaliações de desempenho . . . 52
6.5.1 Avaliação da comunicação inter-chip . . . 52
6.5.2 Avaliação da escalabilidade dos PEs . . . 54
6.5.3 Avaliação da escalabilidade dos sistemas . . . 55
6.5.4 Avaliação considerando o tempo de processamento . . . 56
6.6 Considerações do capítulo . . . 57
7 Confidencialidade na ENoC . . . 59
7.1 Modelo de ataque . . . 59
7.2 Protegendo a ENoC com criptografia . . . 60
7.2.1 AES . . . 60
7.2.2 ECC . . . 61
7.2.3 ECIES . . . 63
7.2.4 Acordo de chaves . . . 64
7.3 Avaliação da Segurança na ENoC . . . 64
7.4 Soluções relacionadas . . . 66
7.5 Considerações sobre o capítulo . . . 67
8 Conclusões . . . 69
Referências . . . 71
Apêndice A: Comunicação MPI . . . 82
Apêndice B: Sistemas de criptografia . . . 85
B.1 Introdução . . . 85
B.2 AES . . . 87
B.2.1 Implementação na rede em chip . . . 87
B.3 Criptografia de Curvas Elípticas . . . 88
B.3.1 Descrição do sistema . . . 89
B.3.2 Implementação na rede em chip . . . 90
B.4 Elliptic Curve Integrated Encryption Scheme . . . 91
B.4.1 Descrição do sistema . . . 91
B.4.2 Implementação na rede em chip . . . 92

B.5 Considerações 93

1 Introdução

Chips são circuitos integrados presentes em equipamentos, sistemas e aplicações eletrônicas embarcadas. Em um único chip é possível integrar centenas de milhares de transistores e centenas de componentes passivos, estes chips podem integrar um sistema completo, os chamados de Sistemas em Chip (*System-on-Chip* (SoC)) (Reis et al. (2007)). Estes sistemas podem possuir múltiplas unidades de processamento interagindo entre si, nesse caso, eles são chamados de Sistemas em Chip com Múltiplos Processadores (*Multiprocessor System-on-Chip* (MPSoC)), neste texto trataremos todos como SoC.

A evolução das tecnologias de fabricação de chips possibilitou aos SoCs evoluírem para sistemas completos de blocos de processadores, coprocessadores, dispositivos com fins específicos, etc. Estes blocos são tradicionalmente chamados de bloco de propriedade intelectual (*Intellectual Property* (IP)) ou Elementos de Processamento (*Processing Element* (PE)), estes PEs podem ser agregados ao SoC por meio de reuso permitindo amortizar os custos do projeto. Eles são disponibilizados por empresas responsáveis pelo projeto ou por terceiros, na forma de modelos sintetizáveis em linguagens de descrição de *hardware* (Zeferino (2003)).

Os PEs são interligados dentro do SoC para interagir e interoperar entre si. A comunicação ocorre tradicionalmente por meio de um barramento compartilhado, reutilizável e de baixo custo, porém sua largura de banda é fixa. Esta desvantagem do barramento inviabiliza a escalabilidade por elemento do sistema pois quanto maior o número de PEs menor será a largura de banda disponível. A Rede em Chip (*Network-on-Chip* (NoC)) (Benini e Micheli (2002)) é uma alternativa balanceada e escalável para comunicação intra-chip ao tradicional barramento.

Uma NoC consiste de um conjunto PEs e roteadores. Cada PE está ligado a um roteador por meio de uma interface de rede. Os roteadores são interligados entre si formando uma rede chaveada que pode ter diferentes tipos de topologia. Adicionalmente uma NoC pode também fazer o uso de um *Hub*, principalmente para a comunicação sem fio. A comunicação nas NoCs com ou sem fio é realizada por meio de pacotes, divididos em pequenas unidades de informações chamadas de *Flow-control Unit*s (Flits). Os *Flits* são transmitidos da origem para o destino pelos roteadores, *hubs* e interfaces de rede, sob ligação metálica ou sem fio (Deb et al. (2012)). As NoCs possuem largura de banda escalável, paralelismo em comunicação, maior frequência de operação em relação ao barramento, embora apresentam maior latência de comunicação e sobrecusto de silício (Zeferino (2003), Dally e Towles (2001)).

As NoCs tradicionais são estruturas estáticas formadas sob uma topologia, como malha de 2 ou 3 dimensões. A comunicação acontece por meio de ligações metálicas entre roteadores e PEs. O conceito original de ligações metálicas foi estendido para considerar as ligações sem fio, como apresentado por Floyd et al. (2002), Chang et al. (2008) e Deb et al. (2012), com o objetivo principal de reduzir o diâmetro da rede intra-chip.

A comunicação entre vários PEs por meio da NoC permite utilizar o paradigma de passagem de mensagens na computação de aplicações diversas. A *Message Passing Interface* (MPI) (Gropp et al. (2017)) é a biblioteca mais popular de passagem de mensagem entre processos para computação paralela. O paradigma de passagem de mensagem é amplamente utilizado em

máquinas paralelas, e a MPI têm se tornado a biblioteca mais popular para este paradigma, seja para computadores de propósito geral ou processadores embarcados.

O suporte a comunicação MPI a nível da NoC já foi demonstrado no processador experimental *Intel* 48 *Pentium class IA* – 32 núcleos (Howard et al. (2010)). Este suporte permite que núcleos de processadores interconectados executem aplicações de alto desempenho (*High Performance Computing* (HPC)) baseado em MPI (Freitas et al. (2010)). Entretanto, esta abordagem é limitada pelos componentes interconectados na NoC, uma vez que o número destes componentes é fixo e não há a possibilidade de ampliá-lo.

Além disso, vários dispositivos são constituídos por unidades de processamento passíveis de utilizar uma NoC para interligação interna. Estes dispositivos podem ser parte de um *cluster*, computador pessoal, dispositivos inteligentes em ambientes corporativos, pessoais, domésticos, de comunicação e entretenimento como *smartphones*, *smart TVs*, *tablets*, eletrodomésticos, *smart "things"*, etc. De fato, com o crescimento dos dispositivos inteligentes para *Internet of Things* (IoT) várias abordagem são apresentadas na literatura com uso de SoCs para diferentes aplicações (Chien et al. (2015); Li et al. (2016); Halim et al. (2017); Conti et al. (2017)). Em geral, os dispositivos da IoT comunicam-se por meio de uma rede de dados para obter ou enviar informações. A NoC pode estar presente nesses SoCs, onde cada sistema age individualmente com seus PEs locais para diversos propósitos. Neste sentido esta tese levanta a questão de pesquisa: *É possível que estes diferentes sistemas possam interconectar-se por meio de suas NoCs para expandir seus recursos com visão única de um sistema?* No entanto, há a limitação das NoCs tradicionais não permitirem a inclusão ou remoção de novos elementos no sistema para oferecer recursos adicionais, após o tempo de fabricação.

Freitas et al. (2008) apresentam a *Multi-Cluster Network-on-Chip* (MCNoC), uma NoC multi-*cluster* reconfigurável. Nessa abordagem há vários *clusters* de núcleos de processamento interligados por meio de uma NoC para para processamento de alto desempenho. No entanto o objetivo dessa abordagem restringe-se a um chip, ou seja, toda a abordagem multi-*cluster* e seu poder de processamento está limitada ao seu único chip, não há expansibilidade dos seus recursos por interligação com outros sistemas MCNoC.

Mangano e Urzi (2015) propõem uma NoC em *cluster* contendo a NoC principal conectada a uma ou mais sub-NoCs. Embora esta abordagem auxilie na melhora do desempenho do sistema por anexar novos recursos ao SoC (ou *cluster* de SoC), há a necessidade de um *backbone* físico no qual todas as sub-NoCs devem ser conectadas. Com a adoção de novos sistemas na NoC, têm-se dois tipos de comunicação, a que acontece entre os PEs de um mesmo SoC ou de SoCs diferentes, chamadas nesta tese de comunicação intra-chip e inter-chip, respectivamente. Shamim et al. (2015, 2017) estenderam este conceito para comunicar SoCs por meio de ligações sem fio, formando um sistema multi-chip. No entanto todos os elementos devem ser conhecidos previamente, ou seja, não é expansível e nem reconfigurável.

As comunicações inter-chip na NoC, sejam por ligações metálicas ou sem fio, devem garantir a confidencialidade da informação. Os trabalhos Adamov et al. (2009), Ancajas et al. (2014a), Frey e Yu (2015), Boraten et al. (2016) e Rajesh et al. (2018) discutem o problema da confidencialidade em NoC e apresentam soluções para a NoC tradicional (comunicação intra-chip), mas nenhuma discussão considera os cenários de comunicação inter-chips.

A conjectura fundamental desta tese é a possibilidade de superar a limitação levantada pela questão de pesquisa com um novo conceito de expansibilidade com as NoCs. Esta tese de doutorado propõe a *Expansible Network-on-Chip* (ENoC), uma NoC expansível. A ENoC permite a conexão de múltiplas NoCs por meio de ligações sem fio com suporte a comunicações MPI. Ela é projetada para expandir os recursos de um sistema interligando-o a vários outros sistemas com comunicação inter-chip de maneira *on-the-fly*. O termo *on-the-fly* é entendido

nesta tese como a capacidade de realizar a expansão sem a necessidade de adaptações ou acordos prévios, em tempo de execução e com sistemas desconhecidos. Esta tese apresenta também uma discussão sobre o Sistema Operacional (SO) para ENoC com diferentes organizações da memória e modelos de programação para cada cenário de configuração, discute também o uso de criptografia para garantir a confidencialidade da informação na comunicação entre os sistemas.

As principais contribuições deste trabalho são:

- *Arquitetura de NoC expansível*: A arquitetura ENoC usa um ambiente integrado de ligações metálicas e sem fio para interconectar automaticamente múltiplos sistemas em uma única visão lógica, comunicando seus elementos intra ou inter-chip;
- *Protocolo de baixa sobrecarga*: A ENoC usa um protocolo de comunicação tão simples quanto possível para executar a comunicação intra e inter-chip, simplificando a pilha de protocolo e reduzindo sua sobrecarga;
- *SO, organização da memória e modelo de programação*: São discutidas várias abordagens a respeito do comportamento do SO ao conectar múltiplos sistemas ENoC. Essa discussão baseia-se principalmente no uso de memórias privadas ou compartilhadas em relação aos SO únicos, múltiplos ou estendidos;
- *Garantia da confidencialidade inter-chip:* Diferentes formas de garantir a confidencialidade na ENoC são propostas com uso de criptossistemas, deixando a decisão para o projetista do sistema, guiado por suas próprias necessidades;
- *Análise de desempenho:* este trabalho apresenta uma implementação real do custo da comunicação ENoC em relação a outras tecnologias de rede, e o impacto de cada criptossistema para garantir a confidencialidade da comunicação inter-chip.

Além dessas contribuições principais, essa tese apresenta como contribuição secundária a extensão do simulador Noxim. Esta extensão suporta a comunicação MPI a partir de traços de aplicações paralelas externas ao simulador e a característica de expansibilidade da ENoC. Além disso, uma das principais contribuições desta extensão é permitir que as simulações aproximarem da realidade ao considerar o tempo de processamento de cada primitiva MPI.

O restante desta tese está organizado da seguinte forma:

- O Capítulo 2 introduz os conceitos que fundamentam a NoC, com foco nos elementos, algoritmos e protocolos que formam a arquitetura e a comunicação, além disso são apresentadas as topologias e como o uso da hierarquia é utilizada nas NoCs;
- O Capítulo 3 apresenta em detalhes da arquitetura ENoC. O foco deste capítulo está em mostrar esta arquitetura, seu protocolo de baixa sobrecarga e uma discussão sobre como pode ser o uso do SO e da organização da memória;
- O Capítulo 4 contém os trabalhos relacionados e considerações entre eles e a ENoC proposta;
- O Capítulo 5 apresenta uma extensão realizada no simulador Noxim para suportar a arquitetura ENoC proposta, traços externos de aplicações reais e a comunicação MPI;
- O Capítulo 6 apresenta a carga de trabalho, parâmetros, métricas e tecnologias de interconexão utilizadas como base de comparação para realizar a avaliação de desempenho da ENoC. Esta avaliação é realizada com dados obtidos a partir de uma suíte de aplicações de problemas reais;

- O Capítulo 7 apresenta as vulnerabilidades na comunicação inter-chip da ENoC com foco na confidencialidade da informação. Diferentes soluções baseadas em criptografia foram analisadas como contramedidas, cada uma possui vantagens e desvantagens e a escolha é uma decisão do projetista. As soluções são testadas e os custos de desempenho são apresentados.
- O Capítulo 8 apresenta as considerações finais e trabalhos futuros.

2 Rede em Chip

A comunicação convencional de uma SoC é realizada por meio de um barramento de transmissão compartilhado, dessa forma apenas um PE pode utilizar esse barramento por vez. Para os PEs utilizarem esse barramento de forma simultânea é necessário um mecanismo de arbitragem e isto resulta em perda de desempenho de comunicação, além disso, o barramento compartilhado possui escalabilidade seriamente limitada (Benini e Micheli (2002)). A NoC é uma alternativa escalável na comunicação do sistema em chip que permite comunicações paralelas simultaneamente. A rede chaveada para a comunicação entre os componentes foi proposta inicialmente em Tewksbury et al. (1992), embora a primeira solução experimental descrita na literatura foi a *Scalable Programmable Interconnection Network* (SPIN) em Guerrier e Greiner (1999). O termo NoC foi proposto por Hemani et al. (2000) e posteriormente adotado pela comunidade científica após a publicação do artigo de Benini e Micheli (2002).

No contexto das redes em chip, a arquitetura especifica a organização física e a topologia de interconexão, e o protocolo especifica como usar os recursos da rede numa operação do sistema. Essa arquitetura está relacionada a redes de interconexões de alto desempenho para computação paralela com múltiplos processadores, na qual cada processador é um chip individual, pois os nós estão fisicamente próximos uns aos outros e têm uma ligação altamente confiável.

2.1 Arquitetura e comunicação

Uma NoC consiste de um conjunto de recursos interligados por roteadores formando uma rede chaveada, como ilustrado na parte *a*) da Figura 2.1, esta Figura foi traduzida de Kumar et al. (2002). A NoC possui dois principais objetivos: que os recursos de *hardware* sejam desenvolvidos de forma independente, ou seja, blocos *stand-alone*, e interligá-los como elementos da rede; e ser uma plataforma de rede escalável, configurável e flexível o suficiente para ser adaptada às necessidades de diferentes tarefas, mantendo a generalidade dos métodos e práticas de desenvolvimento das aplicações.

2.1.1 Elementos da rede em chip

Os recursos do sistema em chip também são conhecidos como núcleos ou elementos de processamento, aqui serão tratados como PE. O PE pode ser um processador de propósito específico, genérico ou uma *Graphical Process Unit* (GPU). Cada PE possui uma interface de rede conectada a um roteador, ilustrado como *IR* na Figura 2.1. A junção do PE e seu roteador forma um *tile*.

Os roteadores encaminham e armazenam temporariamente as mensagens entre os PEs, por meio de suas ligações metálicas e memória temporária (*buffer*). O roteador pode ou não gerenciar as entradas e saídas das mensagens em *buffer* por meio de filas para tratar possíveis

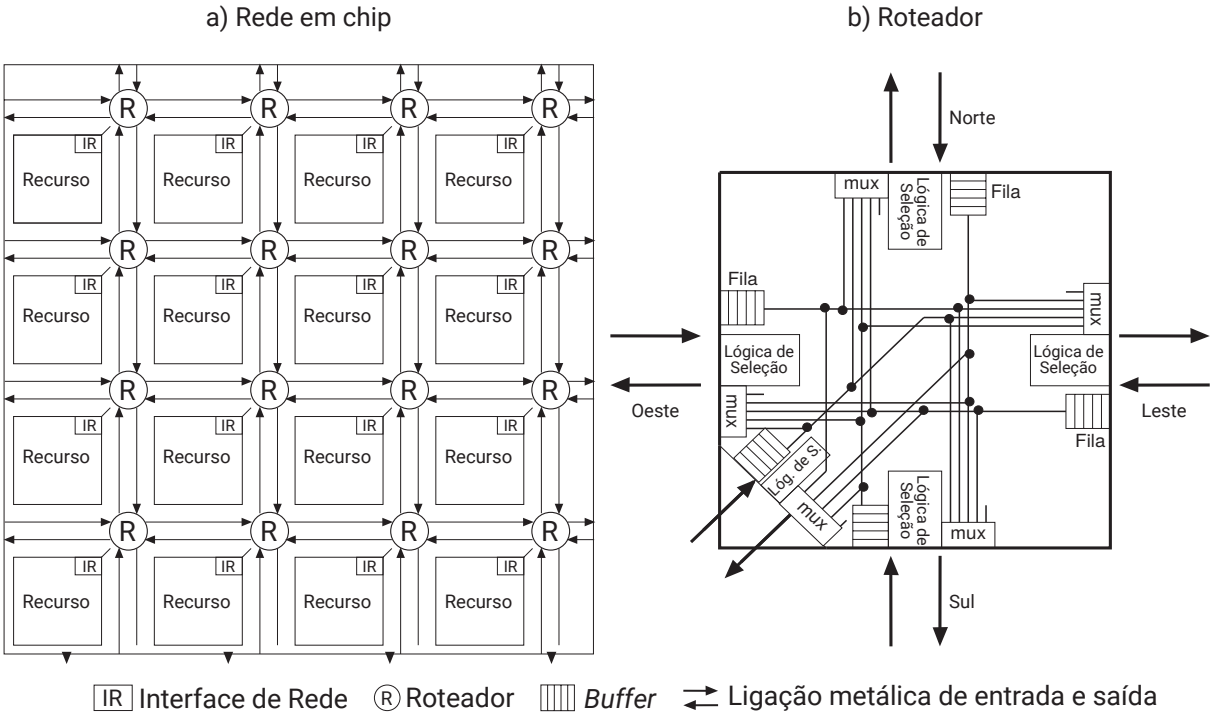


Figura 2.1: Representação de *a*) uma NoC com 16 recursos em malha bidimensional e *b*) de um roteador para NoC (Kumar et al. (2002)).

congestionamentos. Além disso, o roteador pode contar com um multiplexador e seleção lógica para tratar as informações de entrada e saída. Cada roteador possui 5 portas: Norte, Sul, Leste e Oeste para conectar-se aos seus vizinhos, e uma porta local para conectar-se ao PE, como ilustrado na parte *b*) da Figura 2.1.

A informação é enviada por pacotes. O pacote pode ser implementado com diferentes características e formatos pois depende diretamente do protocolo de rede que está sendo usado. Este pacote é fragmentado em pequenas unidades de informação chamadas de *Flits*. Eles são transmitidos na rede em chip por meio de saltos entre os roteadores, trafegando a informação do PE de origem ao PE de destino.

Essa comunicação acontece da seguinte forma:

1. O remetente solicita o envio de uma mensagem a um destinatário;
2. A mensagem é enviada como carga útil do pacote. O pacote é formado pela carga útil, cabeçalho e cauda. Dependendo do tamanho da mensagem e da configuração do protocolo implementado, a mensagem pode ser dividida em mais de um pacote;
3. Cada pacote é dividido em *Flits* pelo PE. O tamanho de cada *Flit* é previamente atribuído no projeto da NoC;
4. O roteador do PE remetente envia os *Flits* para o roteador vizinho;
5. O roteador vizinho recebe os *Flits* e os repassa de acordo com o algoritmo de roteamento utilizado;
6. Este processo se repete até a mensagem alcançar o seu destinatário;

7. O roteador do PE destinatário recebe os *Flits* e repassa para o seu PE;

8. O PE destinatário recebe os *Flits* que compõem o pacote e recupera a mensagem original enviada como carga útil.

2.1.2 Algoritmos de roteamento

Os algoritmos de roteamento para NoC são usados para indicar ao sistema de que forma os *Flits* serão encaminhados entre os roteadores, ou mesmo para evitar *deadlock* como mostrado em Catania et al. (2006). Cada projeto de NoC pode utilizar um algoritmo que seja melhor para as suas necessidades, considerando a topologia utilizada. Como por exemplo, os algoritmos Odd-Even (Chiu (2000)), XY (Bolotin et al. (2004b)) e DyAD (Hu e Marculescu (2004)) podem ser utilizados em projetos tradicionais de NoC baseada em malha bidimensional.

O projetista possui também a alternativa de especificar os possíveis caminhos dos pacotes na NoC em uma tabela de roteamento. Discussões sobre os algoritmos de roteamento para NoC podem ser encontradas em Rantala et al. (2006); Ascia et al. (2008); Valencia et al. (2017).

2.1.3 Protocolo de comunicação

Os protocolos de comunicação na NoC especificam a forma da comunicação entre os remetente, destinatário e roteadores ao longo do caminho, quando for o caso. De forma análoga às redes locais, os protocolos para NoC podem possuir as camadas de enlace, rede, transporte e aplicação (Benini e Micheli (2002)).

- *Física*: consiste no meio físico utilizado para transmissão da informação. Nas redes em chip são utilizados ligações metálicas, por ondas de rádio e fotônica.

- *Enlace*: também chamada de ligação de dados, tem o objetivo de aumentar a confiabilidade até um nível mínimo requerido na rede em chip;

- *Rede*: é responsável pelo controle de entrega fim-a-fim e considera vários canais de comunicação. Existem muitas pesquisas e propostas que abordam a camada de rede para redes em chip, como Hu e Marculescu (2004), Ye et al. (2004), Bertozzi e Benini (2004), Goossens et al. (2005), Li et al. (2006), Zhang et al. (2008), Moscibroda e Mutlu (2009), Ebrahimi et al. (2012), Gabis et al. (2018) envolvendo o roteamento, circuitos, pacotes e chaveamento;

- *Transporte*: decompõe as mensagens em pacotes na origem e as remonta no destinatário, é responsável também por controlar o fluxo e a negociação da transmissão, como o atraso ou perdas de mensagens;

- *Aplicação*: considerada também como camada de *software*, esta camada divide-se em programas de sistema e de aplicação. Os programas de sistemas provêm uma abstração da plataforma de *hardware*, de tal forma que se pode operar (ajustar, ativar, desativar, etc) vários níveis de serviços e dessa forma controlam o consumo energético e de desempenho. Os programas de aplicação objetivam preservar a portabilidade das aplicações sob diferentes plataformas e provêm alguma inteligência para melhorar a natureza do sistema distribuído.

2.2 Topologias

A topologia mais utilizada nas redes em chip é a malha bidimensional, formada pela conexão dos roteadores aos seus vizinhos (Figura 2.2 *a*). A principal vantagem dessa topologia está na simplicidade do projeto e a existência de algoritmos de roteamento bem conhecidos e aceitos. A principal desvantagem está no gasto energético e alta latência, ao considerar um grande número de PEs, em especial na comunicação dos mais distantes

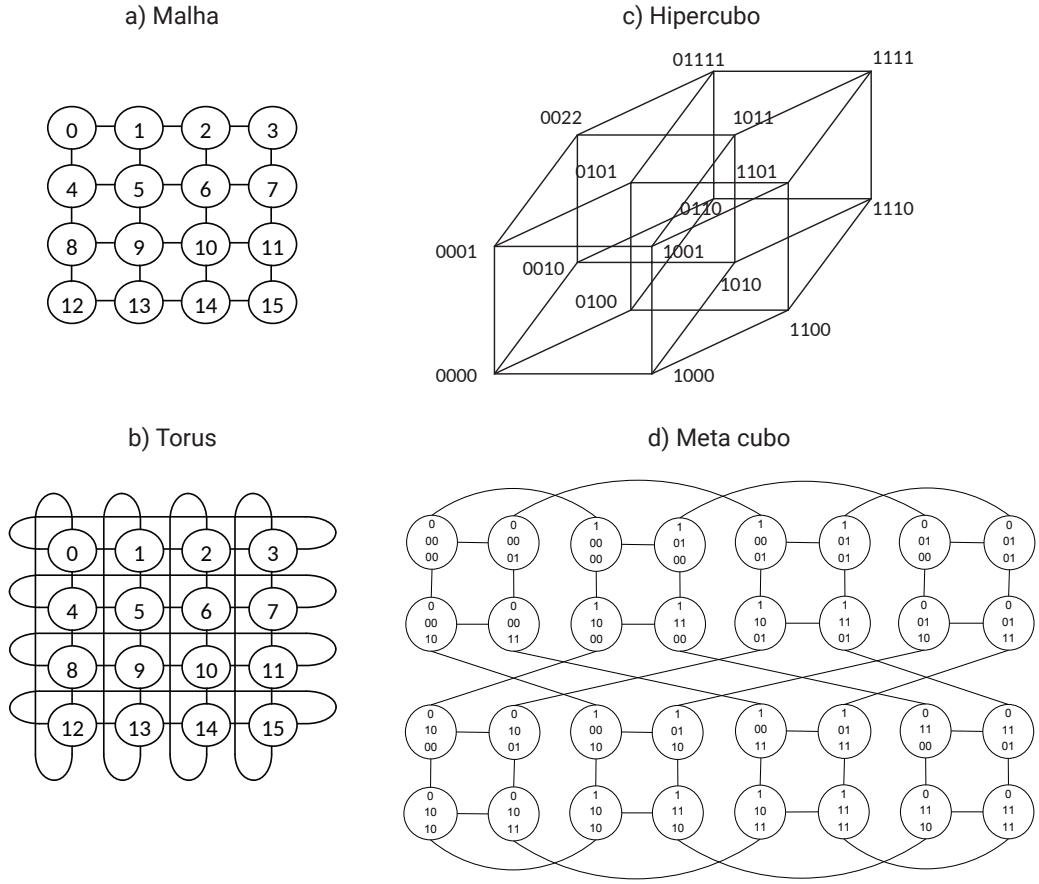


Figura 2.2: Topologias para NoC (Chen et al. (2012)).

Com o objetivo de mitigar a desvantagem da malha bidimensional, outras topologias foram propostas como a NoC de 3 dimensões (Ben Abdallah (2013)), a *Torus* (Figura 2.2 *b*) (Dally (1986); Ye et al. (2012)), Hipercubo (Figura 2.2 *c*) (Dally e Towles (2003)) e Meta cubo (Figura 2.2 *d*) (Li et al. (2010)). Chen et al. (2012) mostram resultados de uma comparação e análise de desempenho das topologias NoC que podem ser levados em consideração na escolha da topologia para uma determinada aplicação. Parte das ilustrações das topologias representadas na Figura 2.2 foram adaptadas de Chen et al. (2012).

Outras topologias para NoC também são encontradas na literatura (Bjerregaard e Mahadevan (2006)), como a árvore binária, anel, e topologias irregulares. As topologias irregulares são derivadas de alterações das conectividades de estruturas regulares, como por exemplo, removendo ligações de uma malha ou misturando topologias diferentes. Freitas et al. (2008) apresenta também uma discussão sobre a importância de adaptar a topologia da NoC à aplicação que será utilizada.

A alta latência na comunicação entre PEs distantes pode ser enfrentada também com o uso da técnica de *Wormhole* (Ni e McKinley (1993)) em topologias NoC, como utilizado em

Bolotin et al. (2004a), esta técnica consiste em conectar fisicamente com uma ligação metálica os pontos distantes que comunicam-se frequentemente. É possível também mitigar esta distância sem a necessidade de ligações metálicas com o uso de ligações de rede sem fio (Deb et al. (2012)), de forma estática ou reconfigurável.

2.3 Redes em Chip sem fio

A ideia de utilizar antenas integradas em chip para interconectar seus elementos por meio de ligações sem fio foi introduzida em Kim et al. (2000), e a distribuição de sinais de relógio (*clock signals*) utilizando a comunicação sem fio foi apresentada em Floyd et al. (2002). A comunicação em uma NoC sem fio ocorre de forma similar à tradicional comunicação por ligações metálicas, porém permite reduzir a um salto a comunicação entre os PEs no sistema, o que impacta diretamente na comunicação dos mais distantes.

Uma típica NoC sem fio, como proposta em Pande et al. (2009) e Deb et al. (2012), é formada por um *hub* conectado a roteadores por ligações metálicas e com outros *hubs* por ligações sem fio, enquanto que em Wang et al. (2011) utilizam alguns roteadores sem fio ao invés de um *hub*. Estas abordagens utilizam as ligações metálicas para comunicar PEs próximos e ligações sem fio para comunicar PEs distantes, separados logicamente ou não em sub NoCs. Vale observar que há impacto de área utilizada no sistema ao utilizar um dispositivo sem fio, de tal forma que Deb et al. (2012) utiliza apenas alguns *hubs* para comunicação sem fio entre áreas do chip ao invés de inserir essa funcionalidade em todos os roteadores, como em Wang et al. (2011), pois este impacto pode tornar o projeto inviável.

Um componente importante no projeto de uma NoC sem fio é a antena. Em Lin et al. (2007), os autores demonstram o desempenho de antenas de silício em chip; Ganguly et al. (2011) discutem a viabilidade em usar nanotubos de carbono para implementar as antenas com ganho em emissão e absorção do sinal, esta antena pode operar em altas frequências alcançando taxas de transferência de dados superiores a antenas de silício (Huang et al. (2008), Carloni et al. (2009)). Embora o uso tradicional na NoC seja restrito a interconectar seus componentes internos, Laha et al. (2015) apresentam a viabilidade em considerar o uso de antenas para também conectar *chips* a uma distância de até 100 cm.

Embora o uso da comunicação sem fio possua as vantagens de diminuir o número de saltos na comunicação entre PEs distantes e comunicar diferentes *chips*, as desvantagens devem ser consideradas. Um projeto NoC sem fio deve possuir um dispositivo como o *Hub* sem fio e isto causa custo de área, consumo extra de energia e possível gargalo na comunicação, de acordo com a implementação deste *hub*, embora esses custos podem ser mitigados quando há muitas conexões entre PEs distantes.

Como a comunicação sem fio acontece por meio de difusão originalmente sem qualquer proteção, qualquer entidade dentro do raio de alcance do sinal pode receber a transmissão e obter as informações trafegadas, a NoC sem fio também está suscetível (Ganguly et al. (2017); Pereñíguez García e Abellán (2017)). A confidencialidade dos dados nessa transmissão pode ser obtida por sistemas de criptografia. No entanto questões relacionadas ao tipo de sistema de criptografia devem ser discutidas bem como seu impacto no desempenho da comunicação na NoC.

2.4 Considerações sobre o capítulo

Este capítulo apresentou os conceitos fundamentais das NoCs e uma breve descrição do seu cenário atual. Como principal vantagem, é possível concluir que as NoCs possuem escalabilidade e paralelismo na comunicação, e maior frequência de operação em relação ao barramento. No entanto, apresentam maior latência de comunicação, gasto de energia e sobrecusto de silício (Dally e Towles (2001), Zeferino (2003)). A versatilidade e escalabilidade tornam a NoC atrativa para vários cenários, como a computação de alto desempenho e interligação entre núcleo de processadores. Estas vantagens sobrepujam seu ônus, e a NoC está cada vez mais presente nos sistemas em chip.

3 ENoC: The Expansible Network-on-Chip

Tradicionalmente, as NoCs são estruturas estáticas que não permitem a inclusão e remoção de elementos interligados por ligações metálicas ou sem fio. As abordagens de interconexão baseadas em rede sem fio foram propostas com o foco em reduzir o diâmetro da rede intra-chip, e não na expansibilidade dos recursos e processamento do SoC, ou seja, é limitada pelos componentes internos interconectados pela NoC.

Mangano e Urzi (2015) propõem a criação de sub-NoCs por ligações físicas, Shamim et al. (2015) e Shamim et al. (2017) apresentam abordagens de comunicação entre diferentes chips. Estas soluções não visam a característica de expandir (ou mesmo contrair) os recursos e poder de processamento de forma *on-the-fly*, independente de conhecimento prévio dos elementos ou suas particularidades. Em outras palavras, Mangano e Urzi (2015) necessitam de conexão física e de acordos sobre a hierarquia entre "mestres e escravos", e Shamim et al. (2015, 2017) limitam-se na necessidade dos elementos intra ou inter-chip serem conhecidos previamente, ou seja, não são dinâmicos e nem reconfiguráveis.

Este trabalho propõe a ENoC, uma arquitetura para rede em chip capaz de expandir recursos de um SoC interligando dois ou mais sistemas, como ilustrado na Figura 3.1. A comunicação intra-chip ocorre por meio de ligações metálicas enquanto que a comunicação inter-chip ocorre por meio de ligações sem fio, com um protocolo tão simples quanto possível para mitigar a sobrecarga de comunicação. O uso da ligação sem fio facilita a inclusão e remoção de novos dispositivos e provê uma interface de comunicação com baixa sobrecarga ao utilizar o protocolo customizado, além disso, a comunicação entre os diferentes sistemas é realizada com apenas 1 "pulo" e baixa latência, como demonstrado em DiTomaso et al. (2015).

A arquitetura da ENoC utiliza os conceitos da NoC sem fio para interconectar automaticamente múltiplos dispositivos do sistema em uma única visão lógica. Esta proposta também implementa um simples protocolo com baixa sobrecarga para comunicação intra e inter-chip com objetivo de simplificar a pilha de protocolo e reduzir sua sobrecarga.

3.1 Arquitetura

A arquitetura com fio ENoC consiste em *N tiles*, no qual cada *tile* é composto por um PE e um roteador ligados por interface de rede. O PE pode ser qualquer elemento de processamento, seja de propósito específico ou genérico, banco de memória ou dispositivos que realizam operações computacionais oferecendo ou requisitando recursos ao sistema. Os roteadores são responsáveis por entregar pacotes corretamente por meio de uma malha interligada fisicamente. A topologia padrão utilizada na comunicação intra-chip da ENoC é a malha bidimensional, escolhida por ser a mais simples, do ponto de vista de *layout* e interconexões entre os PEs e roteadores, independente do tamanho da rede. Além disso, o roteamento nesta

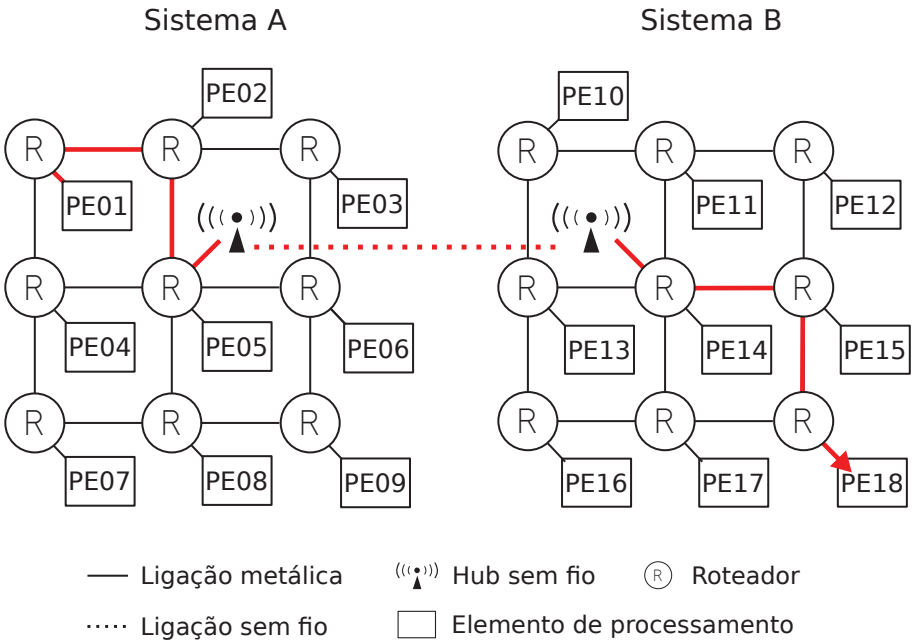


Figura 3.1: Expansão com a ENoC.

topologia pode ser simples ao ponto de resultar em roteadores pequenos, alta capacidade e ciclos curtos de *clock*, como apresentado em Kumar et al. (2002).

A interface sem fio da arquitetura consiste em um ou mais *hubs* sem fio para cada sistema diferente, conectando e comunicando com outros sistemas. A Figura 3.1 ilustra 2 sistemas ENoCs interagindo, cada um representado por uma malha $2D$ de tamanho 3×3. Neste exemplo, o PE 01 do sistema A envia uma mensagem para o PE 18 do sistema B, a comunicação acontece com apenas 4 pulos sob ligações metálicas e 1 pulo sob ligação sem fio.

Os roteadores na ENoC possuem 5 portas: uma local para conectar ao PE e as portas Norte, Leste, Sul e Oeste para ligar com seus vizinhos. Cada porta utiliza *buffers* para armazenar as mensagens de entrada e saída. Cada roteador usa uma tabela de roteamento local para comunicação intra-chip. A tabela de roteamento local deve conter rotas para que todos os PEs comuniquem entre si e alcancem o *hub* sem fio do próprio sistema.

Em todo sistema, há ao menos um roteador que possui uma porta adicional para conectar a um *hub* sem fio, como mostrado na Figura 3.2. A ENoC deve ter ao menos 1 *hub* sem fio, mas a decisão da quantidade e sua disposição é uma decisão de implementação de projeto, levando em conta o *trade-off* entre o objetivo e os recursos disponíveis.

3.2 Camada Física e Enlace de dados

A comunicação dentro do sistema é *full-duplex* através de roteadores interligados por ligações metálicas, habilitando transmissão paralela em todos os roteadores. A comunicação para fora do sistema é feita através do *hub* sem fio. A ligação sem fio ENoC é baseada em *Wireless HD* (WiHD) (Wireless HD Consortium (2010)) trabalhando na frequência de 60 Ghz com antenas omnidirecionais, com o padrão *Orthogonal Frequency-Division Multiplexing* (OFDM) para multiplexação e 64 *Quadrature Amplitude Modulation* (QAM) para modulação.

O OFDM com 64 QAM é utilizado com *network code rate* de $13/16$ e sensibilidade de -47 dBm alcançando 6.6 Gb/s por canal (como demonstrado em Matyjas et al. (2015)), permitindo até 6 canais usando *Multiple-Input and Multiple-Output* (MIMO) (Zelst e Schenk

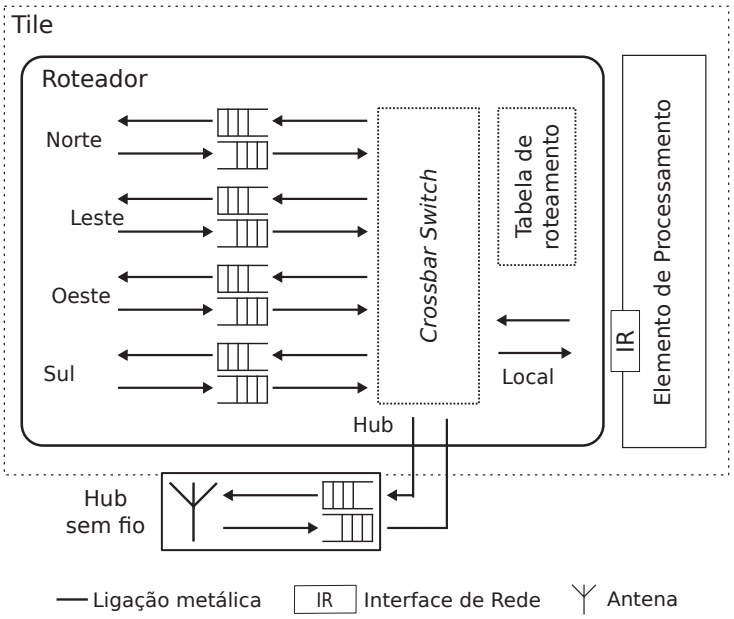


Figura 3.2: Roteador ENoC.

(2004)) obtendo até 28 Gb/s. A multiplexação com OFDM e a modulação com 64 QAM foram escolhidos pois provêm melhor desempenho geral em canais seletivos de alta frequência, realizam a codificação de dados, intercalação na captura de domínios de frequência e frequências diversas mais eficientemente (Daniels et al. (2010)).

Além disso, a característica de ortogonalidade provê a sinalização poderosa e eficiente para um grande número de usuários em um mesmo canal. Cada frequência é modulada com dados binários para prover um número de portadoras paralelas, na qual cada uma contém uma porção de dados de usuário (Rappaport (2002)). A conexão de acesso múltiplo e prevenção de colisão é alcançado com *Orthogonal Frequency-Division Multiple Access* (OFDMA) (Wireless HD Consortium (2010), Hansen (2011)). OFDMA (Rohling et al. (1999)) é uma versão multi usuário do OFDM o qual provê operação de multiplexação de fluxo de dados de usuário dentro da portadora do *link* sem fio.

O dado dentro da ENoC é transmitido em forma de pacotes divididos em *flits*. O *flit* possui o tamanho de 32 bits e são chaveados entre os roteadores e *hubs* para alcançar seu destino, seja na comunicação intra ou inter-chip. A divisão e remontagem dos pacotes em flits é realizada pelos PEs de origem e destino, respectivamente. O *hub* sem fio é o dispositivo responsável pela ligação de dados inter-chip na ENoC. Ele é responsável por sempre perceber o canal sem fio para identificar novos sistemas ENoCs, receber/transmitir os pacotes entre os sistemas distintos e reduzir a latência na comunicação entre os PEs distantes no sistema local.

Em intervalos de tempo, o *hub* envia um *beacon* por difusão para que possa ser detectado. Este *beacon* é uma pequena informação utilizada na ENoC para que o *hub* possa identificar a si e consequentemente o sistema a qual pertence. Quando uma nova ENoC aparece nas proximidades, o *hub* sem fio detecta-a e ambos os *hubs* trocam informações que identificam os sistemas e consequentemente os seus recursos.

3.3 Camada de rede

A camada de rede é responsável pelo pacote, sua segmentação em *flits* e remontagem, e pelo roteamento do pacote por meio de saltos entre os roteadores. Na ENoC, na camada de rede

são implementados o pacote e a técnica de roteamento que permitem a comunicação com novos sistemas.

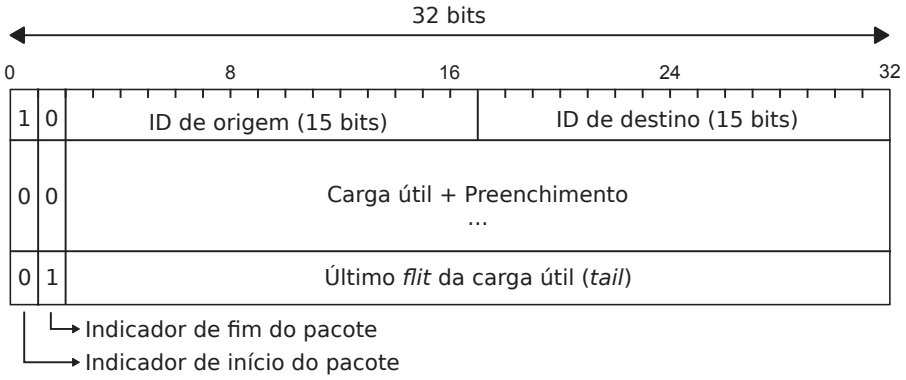


Figura 3.3: Pacote ENoC.

O pacote ENoC é projetado para ser tão simples quanto possível, como mostrado na figura 3.3, com cabeçalho, carga útil (*payload*) e cauda. O pacote possui dois bits iniciais que identificam suas partes. O primeiro bit indica a primeira parte do pacote (cabeçalho) enquanto que o segundo bit indica o final do pacote (cauda ou *tail*). Vale ressaltar que o pacote é dividido em *flits* de 32 bits, assim o cabeçalho e cauda são enviados como primeiro e último flit, respectivamente. Em outras palavras, os bits indicam:

- 10: o cabeçalho;
- 00: a carga útil;
- 01: o último *flit* da carga útil e cauda do pacote.

O cabeçalho possui 15 bits para indicar o endereço de origem e 15 bits para indicar o endereço de destino. A Figura 3.4 mostra a estrutura desses endereços. Os bits de identificação são divididos em: 7 bits para identificar o PE no sistema local, e 8 bits para identificar o *hub* sem fio, dessa forma a ENoC suporta até 128 PEs por SoC e no máximo 256 sistemas, alcançando um total de 32768 PEs.

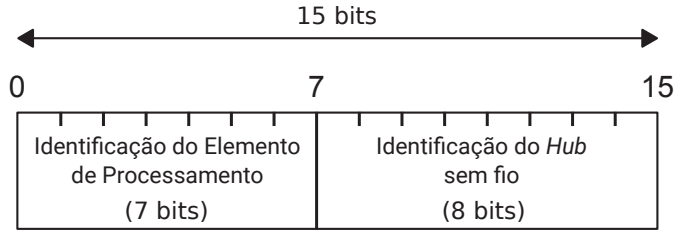


Figura 3.4: Estrutura de endereço de origem e destino.

Esta combinação permite uma fácil identificação de cada PE na ENoC, porém diferentes formas podem ser facilmente implementadas. A carga útil contém entre 1 e 1500 bytes, porém é conveniente considerar o tamanho mínimo de 4 bytes para que a carga útil mínima possa ser enviada em um *flit* de 32 bits, padrão na ENoC. Este intervalo foi escolhido por ser tão pequeno quanto possível para evitar problemas de preenchimento e grande o suficiente para evitar colisão.

Em resumo, o cabeçalho da ENoC possui o tamanho fixo de 4 bytes, a carga útil varia entre 4 e 1500 bytes, e a cauda não adiciona valores aos tamanhos mínimo e máximo pois utiliza os últimos bits da carga útil.

O roteamento na ENoC é realizado por meio de uma tabela de roteamento. Todos os PEs do sistema devem conhecer esta tabela. Ela contém rotas que permitem cada PE comunicar com qualquer outro PE no sistema local. Quando a comunicação acontece com um PE que não é conhecido localmente (ou seja, com sistemas externos), a rota é direcionada ao *hub* sem fio. O *hub* sem fio atua como um *gateway* do sistema local para os sistemas externos, e vice-versa, realizando a comunicação entre os diferentes sistemas.

Ao expandir, as tabelas locais de roteamento não precisam ser reconfiguradas, uma vez que os pacotes destinados ao sistemas externos são enviados por difusão entre os *hubs*, e o sistema destinatário identifica-se pelo protocolo utilizado. Vale ressaltar que os últimos 8 bits da estrutura de endereço é destinado especificamente para identificar o *hub* sem fio a qual pertence o PE, permitindo a comunicação inter-chip tão simples quanto possível.

3.4 Camada de transporte

A camada de transporte é responsável por decompor as mensagens em pacotes na origem e remontar a mensagem original enviada por meio de um ou mais pacotes. Esta camada também é responsável pelo controle de fluxo e entrega confiável. Na ENoC a comunicação fim-a-fim é implementada nos PEs. O controle de fluxo pode ser implementado utilizando o método *send-and-wait*, no qual o remetente aguarda uma mensagem de confirmação (*ACK*) antes de enviar a próxima mensagem, ou mesmo não utilizar *ACKs* de confirmação e o destinatário solicitar a retransmissão quando necessário.

A verificação da entrega confiável pode ser implementada com retransmissão de pacotes, utilizando técnicas como *piggybacking* ou *go-back-N*, como utilizados em Radulescu et al. (2004), Dall'Osso et al. (2012), Ramakrishna et al. (2016), Rambo et al. (2016). Na ENoC não há a limitação no uso de uma tecnologia de retransmissão específica, e o projetista possui a liberdade de implementar a forma de transmissão confiável mais adequada ao seu projeto. Além disso, a camada de transporte também é responsável pelo *handshake* entre os sistemas no momento de expansão.

3.5 Reconfiguração de expansão

Cada *hub* da ENoC envia em tempos regulares um *beacon* para que possa ser identificado por outra ENoC. O *beacon* pode ser implementado pelo envio de um pacote de tamanho mínimo com objetivo apenas de informar a presença do sistema. Cada *hub* também monitora o canal sem fio para perceber o *beacon* de um novo sistema. Quando uma ENoC é detectada por outra, o *handshake* para estabelecerem uma sessão de comunicação é iniciado pela ENoC detectante. Neste momento há a troca de um conjunto de informações entre os sistemas para identificarem a si mesmo e aos seus recursos. A Figura 3.5 ilustra este *handshake*.

Este *Handshake* é realizado apenas uma vez, no momento de expansão. Os dados de identificação do *Hub* e dos PEs recebidos são armazenadas na memória do *Hub*. A identificação inicial contém 8 bits de identificação do *hub* sem fio e 7 bits para cada *PE* do sistema. Vale ressaltar que, de acordo com a implementação do protocolo ENoC é possível que cada sistema possua no máximo 128 PEs e 256 sistemas simultâneos, assim o conjunto de informação de identificação para cada sistema tem um valor variável entre $\sim$ 2 bytes (15 bits) a 120 bytes.

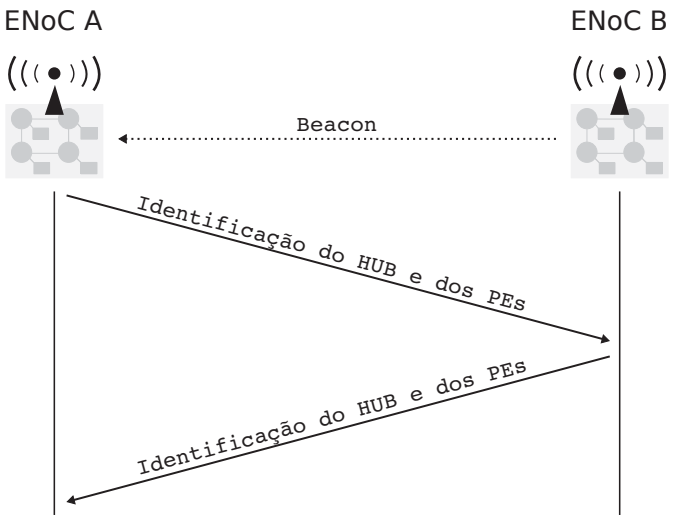


Figura 3.5: *Handshake* inicial na ENoC.

Dessa forma, cada *hub* deve ser capaz de armazenar e gerenciar dinamicamente 30, 6 KB de informações dos sistemas durante a expansibilidade.

Após se reconfigurarem, os novos dados de identificação são associados ao *beacon* percebido para que esta troca de informação acontece apenas uma vez, com isso um sistema de identificação única deve ser considerado para cada sistema ENoC. Esta identificação pode basear-se em *Physical Unclonable Functions* (PUF) (Blaise et al. (2004)) utilizando as características únicas de cada chip, ou mesmo por um sistema padronizado globalmente como o endereço de *hardware* (IEEE (2017)). Os dados de identificação devem permanecer nas memórias dos *Hubs* sem fio durante toda a comunicação, após estes dados podem ser removidos após um tempo arbitrário de inatividade ou da ausência do *beacon* que originou a comunicação.

3.6 Sistema operacional e organização da memória

Para que a arquitetura da ENoC seja viável para computação de alto desempenho, é necessário que haja uma forma de que o sistema operacional interaja entre as aplicações e com uma arquitetura distribuída. Essa viabilidade já foi provada pelo Sistema Operacional Amoeba (Mullender et al. (1990)). Existem diferentes possibilidades de configuração em sistemas ENoCs relacionadas ao SO e a organização da memória. A memória pode ser privada ou compartilhada, e o SO pode ser único ou múltiplos. Essas possibilidades são entendidas a partir de dois pontos de discussão. O primeiro ponto é o aspecto do SO ao conectar sistemas distintos. Esses sistemas podem ter um único SO se houver alto nível de acoplamento de dispositivos, eles podem ter múltiplos SOs se forem isolados ou sistemas estendidos ou uma abordagem de extensão de um SO para todos sistemas. O segundo ponto é a visão da memória pelo SO, ela pode ser compartilhada ou privada. A combinação dessas possibilidades está sumarizada na Tabela 3.1.

Tabela 3.1: Sistema operacional e gerenciamento de memória para a ENoC.

	Memória privada	Memória compartilhada
SO único		•
SOs múltiplos	•	
SO estendido	•	•

A abordagem de *SO único com memória privada* não é conveniente pois toda a memória deve ser acessível para um único SO. A combinação de *SO único com memória compartilhada* provê uma visão única do sistema, como o Locus OS (Walker et al. (1983)) sob um sistema *Non-Uniform Memory Access* (NUMA). Se na ENoC os sistemas possuírem próprios SOs antes do acoplamento, eles devem definir um SO mestre e eliminar todos os escravos após o acoplamento. Tal reconfiguração pode ser altamente custosa.

A configuração de múltiplos SOs com memória privada é o cenário padrão para a ENoC. Cada dispositivo pode ter seu próprio SO e espaço privado de memória. Nesta configuração, um sistema não é capaz de acessar diretamente o espaço de memória do outro, requerendo assim a passagem de mensagem entre processos para utilizar os recursos dos demais sistemas após a expansão.

Quando dois ou mais sistemas ENoC são conectados há a expansão dos recursos e o sistema é visto como único para o processamento de uma aplicação. O processamento pode ser realizado de forma paralela por diferentes PEs de diferentes chips. Vale ressaltar que na ENoC não há o conceito de hierarquia, ou seja, não há um sistema principal e os demais são subordinados a ele.

Múltiplos SOs com memória compartilhada é uma combinação inviável para a ENoC. O controle, principalmente da memória, requerido para fundir múltiplos SOs causaria impacto no desempenho do sistema.

A configuração de *SO estendido com memória privada* permite a criação de *cluster* de máquinas, formada por um mestre e múltiplos escravos comportando-se como uma única máquina. Isto requer uma extensão do SO com adição de uma camada entre si mesmo e as aplicações a fim de gerenciar todos os recursos. Consequentemente, essa camada terá impacto no desempenho do sistema.

E por fim, o uso de *SO estendido com memória compartilhada* permite uma abordagem similar ao SO MOSIX (Barak e La'adan (1998)) no sentido de prover a visão *Symmetric Multi-Processor* (SMP) para o usuário. Este SO estendido procura os recursos de forma transparente e automática, e é capaz de migrar os processos entre os núcleos de processamento na tentativa de melhorar o desempenho final do sistema.

3.7 Considerações sobre o capítulo

Este capítulo apresentou a arquitetura ENoC, a primeira NoC expansível *on-the-fly*. Em resumo, a ENoC pode identificar e conectar a novos sistemas sem nenhuma modificação ou adaptação física, possibilitando múltiplos sistemas trabalharem como uma única visão lógica. Para isso, é utilizado um protocolo com baixa sobrecarga sem a pilha de protocolos convencionais dos sistemas de comunicação. Ela é baseada em comunicação sem fio de 60 GHz usando o esquema de modulação digital OFDM com 64 QAM.

No entanto, a comunicação sem fio usada nesta expansão entre sistemas é insegura. O capítulo 7 explora os cenários de insegurança que podem acontecer durante a comunicação inter-chip da ENoC e apresenta propostas de segurança baseadas em criptografia para proteger essa comunicação e tornar viável o uso da ENoC em ambientes inseguros.

4 Trabalhos Relacionados

Vários pesquisadores (Benini e Micheli (2002), Zeferino e Susin (2003), Krichene et al. (2016)) avaliaram o uso da NoC para interconectar múltiplos núcleos de processamento com sistemas de memória cache privada ou compartilhada visando prover um alto desempenho e baixa contenção de comunicação entre os recursos no chip. O uso da NoC foi experimentado com sucesso para o uso de passagem de mensagem com o processador de 48 núcleos da Intel (Howard et al. (2010)), ele possui uma NoC $2D$ em malha 6×4 com coerência de memória compartilhada mantida por meio de biblioteca de aplicações paralelas, tais como *Open Multi-Processing* (OMP) ou MPI.

A NoC com ligações metálicas foi estendida para usar sinais sem fio (Floyd et al. (2002) Chang et al. (2008) Deb et al. (2012)) para comunicar dois elementos dentro do chip, chamada de *Wireless Network-on-Chip* (WiNoC). Basicamente, a WiNoC foi proposta principalmente para reduzir o número de pulos durante a comunicação intra-chip. Estes trabalhos foram estendidos por DiTomaso et al. (2015) para realizar comunicação com ligações metálicas e sem fio dependendo da distância entre o remetente e destinatário, mitigando a latência em comunicação com múltiplos saltos.

Estes trabalhos corroboram com a ideia da ENoC em utilizar a comunicação sem fio para interconectar vários PEs de diferentes sistemas, interoperando com passagem de mensagem para executar computação paralela. Com a ENoC é possível que o sistema seja expandido e visto como único sistema, assim como o *Silicon Graphics International* (SGI) *Rackable Systems* (Saini et al. (2013)). O SGI é capaz de conectar múltiplos sistemas em um único ponto de visão por um SO, utilizando a tecnologia de interconexão *InfiniBand* como *backbone* para conectar com fio múltiplos sistemas.

Freitas et al. (2008) propõe um chip *multi-cluster* interligados por uma NoC para processamento distribuído de alto desempenho. A interligação é realizada por roteadores reconfiguráveis a fim de suportar uma topologia adaptável a aplicação. Dependendo da topologia utilizada, pode haver a presença de roteadores centrais e roteadores de borda onde cada roteador de borda é ligado a um cluster ao invés de um PE. Para que isso seja possível, os autores propõem um roteador reconfigurável para essa NoC. A ideia de processamento distribuído entre diferentes núcleos de um SoC aproxima a MCNoC da ENoC, porém os objetivos são diferentes. A MCNoC restringe-se a um chip para suportar clusters de alto desempenho, e não há expansibilidade entre sistemas como a ENoC.

A inovação abordada pela ENoC é a expansibilidade, pois ela leva a ideia da comunicação sem fio na NoC para além, conectando múltiplos *chips* de maneira *on-the-fly*, usando NoC com fio dentro do chip e comunicação sem fio para fora do chip de forma que a expansão ou contração do sistema ocorra simplesmente por aproximação. Esta ideia de interligar diferentes NoCs também foi abordada por Mangano e Urzi (2015), porém os autores abordam uma estrutura hierárquica por meio de ligações físicas, com a necessidade de uma NoC central que permite sub NoCs serem anexadas a ela.

A ENoC estende esta ideia de conectar NoCs de Mangano e Urzi (2015), porém com abordagem diferente. A expansão de recursos ocorre de forma não hierárquica, ou seja, um SoC pode conectar-se à outro sem a necessidade de atribuição de níveis de acesso, eles apenas compartilham os recursos. As vantagens e desvantagens de uma NoC hierárquica são discutidas em Rezaei et al. (2016), de fato, a ENoC pode ser estendida para suportar a hierarquia, se for a necessidade do projeto, mas esta característica não está no escopo desta pesquisa. Além disso, as ligações inter-chips acontecem por meio de ligações sem fio, retirando a necessidade de modificações físicas ou adaptações.

Os trabalhos encontrados na literatura que mais se relacionam com o objetivo e metodologia da ENoC proposta são Laha et al. (2015), Shamim et al. (2015) e Shamim et al. (2017). Laha et al. (2015) explora interconexões *on-chip* e *off-chip* por canal sem fio para comunicar blocos de processamento com blocos de memória em chips diferentes. Estes chips possuem elementos que se conhecem previamente, porém são separados fisicamente em chips distintos, que interoperam entre si para implementar um sistema HPC de alto desempenho. De forma bem similar, Shamim et al. (2015) e Shamim et al. (2017) propõem um *framework* de interconexão intra e inter-chip de maneira (*seamless*), porém considera duas topologias, malha $2D$ e mundo pequeno. Estes dois últimos trabalhos são dos mesmos autores, o segundo trabalho é uma evolução do primeiro, no qual a diferença está na forma de controlar o acesso ao meio, o primeiro é baseada em *Code Divison Multiple Access* (CDMA) e o segundo é baseado em *token*.

Os autores de Laha et al. (2015); Shamim et al. (2017) propõem o uso de uma *Wireless Interface* (WI) associada com determinados PEs do sistema. Cada NoC é dividida em sub redes lógicas e cada sub rede possui um PE com uma WI para comunicar com outras sub redes por comunicação sem fio. Estes autores sugerem o uso de uma antena miniaturizada e embarcada no chip operando 60 GHz, essa antena é fabricada por metal com formato em zig-zag, como proposto em Lin et al. (2007). A modulação do sinal é realizada de mesma forma que em Laha et al. (2015), com uso da modulação *On-Off Keying* (OOK) não coerente (Yu et al. (2015)), a escolha desta modulação é embasada na simples implementação e baixo custo energético.

O roteamento do sistema utiliza um mecanismo *seamless* de comunicação intra e inter-chip. Este mecanismo significa que todos os PEs do sistema possuem um endereço único, se conhecem e trocam informações entre si de forma direta, comunicando a sub rede do PE de origem com outra sub rede que contém o PE de destino, independente de qual *chip* está a sub rede. O protocolo de roteamento utilizado é o baseado em tabela com menor caminho, a partir de caminhos pré-computados determinados por algoritmo de menor caminho.

As ligações sem fio utilizam chaveamento *Wormhole* modificado para permitir o controle de acesso ao meio baseado em token, como proposto em Deb et al. (2012). Este controle de acesso é garantido pela posse de um *token*, de tal forma que somente a WI que possui o *token* pode transmitir pelo canal sem fio. Em resultados experimentais, os autores apontaram a taxa de transferência de 16 Gbps neste canal sem fio com modulação *OOK* e controle de fluxo com *Token*.

Para habilitar a passagem do token entre as WIs de forma justa, as WIs são numeradas sequencialmente em um *token-ring* virtual. O *token* circula autonomamente entre as WIs como um *flit* sem fio, similar ao escalonamento *round-robin*. Cada WI mantém a posse do token por um *quantum* variável, no qual o *quantum* é o mesmo do ciclo de relógio (*clock cycle*) do sistema.

Os autores simularam essa proposta modelando flits com 32 bits, comunicação sem fio com 4 canais virtuais, *buffers* ligados as WIs com suporte a 64 flits, pacote de tamanho fixo (máximo e mínimo) de 256 bytes. Note que o tamanho do *buffer* é suficiente para armazenar um pacote inteiro. A taxa de transferência de dados inter-chip, como já mencionada, é de 16 Gb/s. Na abordagem anterior desse trabalho, essa taxa de transferência é de 6 Gb/s por utilizar CDMA

ao invés de OOK com uso de *tokens* para controlar o fluxo da comunicação e evitar colisões. A Tabela 4.1 sumariza as principais diferenças entre a ENoC e os trabalhos relacionados.

Tabela 4.1: Comparação com trabalhos relacionados

	Shamim 2015	Shamim 2017	ENoC
Taxa de Transferência	6	16	25
Acesso ao meio	OOK	OOK + Token	OFDM
Topologia	Malha 2D + Mundo Pequeno	Malha 2D + Mundo Pequeno	Malha 2D
Pacote de comunicação	Pacote fixo de 256 bytes	Pacote fixo de 256 bytes	Pacote com tamanho variável para o protocolo de baixa sobrecarga
Sistema reconfigurável	Não	Não	Sim
Restrições	Todos os componentes devem se conhecer previamente	Todos os componentes devem se conhecer previamente	Concebido para expandir recursos com sistemas diferentes

Diferentemente desses trabalhos relacionados, a ENoC é baseada em OFDM e obtém até 28 Gb/s de taxa de transferência de dados. Além disso, a metodologia apresentada por Shamim et al. (2017) é inviável para a ENoC, pois os sistemas devem se conhecer previamente, em específico, todas as sub redes lógicas de todos os chips devem se conhecerem previamente para realizar a comunicação sem fio. Esta exigência inviabiliza uma expansão e contração do sistema de maneira *on-the-fly*. Diferentemente destes trabalhos relacionados, a ENoC é expansível e reconfigurável.

Há também a necessidade de várias WIs, um para cada grupo. Se a ENoC utilizasse essa abordagem haveria um maior custo de memória e tempo para reconfigurar todos os *hubs* sem fio de todas as sub redes, e ao considerar alguma abordagem de segurança, esta deveria ser aplicada em cada *hub* impactando no desempenho da comunicação inter-chip. Na prática cada sub rede trabalha com uma ENoC distinta, multiplicando a sobrecarga de interconexão.

4.1 Considerações sobre o capítulo

Este capítulo apresentou os trabalhos presentes na literatura que mais se aproximam da ENoC. Embora a abordagem de conexão inter-chip apresentada por Shamim et al. (2015, 2017) seja similar à ENoC, os propósitos são diferentes. A ENoC apresenta o conceito de expansibilidade de um sistema em chip por simples aproximação a outro não conhecidos previamente, além disso, o processamento pode ser distribuído entre os núcleos de todos os sistemas na expansão por meio de passagem de mensagem. A interligação entre sistemas em chip por meio de ligações sem fio é a principal semelhança entre estes trabalhos relacionados. O conceito da ENoC vai além de simples interligações sem fio.

5 Ambiente de Simulação

A simulação é uma maneira viável de projetistas NoC avaliarem novas arquiteturas e organização de interconexão, permitindo-os analisar o tráfego de comunicação, medir desempenho, gasto de energia e custo de área. O simulador é a ferramenta adequada para obter resultados de testes e analisar a viabilidade de novos conceitos e ideias sem prototipar um *hardware*. Este capítulo apresenta a ferramenta de simulação utilizada para avaliar a ENoC.

A escolha do simulador é uma etapa fundamental para a simulação, pois suas características devem cobrir a proposta e suportar tecnologias no estado da arte. Assim, buscou-se simuladores NoC que permitam implementar os conceitos de *clusters* NoC, comunicação sem fio e suporte a MPI.

O simulador deve prover formas de receber entradas de carga de trabalho externas a ele, isto pode ser feito com simulação dirigida a traço. O traço é um *log* de eventos e usualmente inclui o tempo, tipo, tamanho e outros parâmetros associados a cada evento, dependendo do tipo de simulação que está sendo realizada.

Simulações dirigidas a traço podem ser usadas por algoritmos de gerenciamento de recursos, prevenção de *deadlock*, escalonador de CPU, análise de *cache*, entre outros. Entre todas as vantagens, a simulação dirigida a traço permite uma carga de trabalho realística, entretanto ainda há desvantagem de ser um sistema complexo para considerar os aspectos realísticos dessa carga de trabalho, e considerar apenas a simulação resulta em um único ponto de validação (Jain (1991)).

Existem vários simuladores específicos para NoC como: Noxim (Catania et al. (2015)); Booksim (Jiang et al. (2013)); Naxim (Nakajima et al. (2013)); Hornet (Ren et al. (2012)); Topaz (Abad et al. (2012)); HNOCS (Ben-Itzhak et al. (2012)); NoC for OMNeT++ (Ben-Itzhak et al. (2011)); WNoC Simulator (Wang et al. (2011)); Darsim (Lis, Mieszko and Shim, Keun Sup and Cho, Myong Hyon and Ren, Pengju and Khan, Omer and Devadas, Srinivas (2010)); Netrace (Hestness, Joel and Grot, Boris and Keckler, Stephen W. (2010)); Garnet (Agarwal et al. (2009)); gpNoCsim (Hossain et al. (2007)); Nirgam (Jain et al. (2007)); NNSE (Lu et al. (2005)). Outros simuladores não específicos para NoC podem ser adaptados de alguma forma, os quais são: Gem5 (Binkert et al. (2011)); CACTI (Wilton e Jouppi (1996)); NS-3 (Riley e Henderson (2010)). Para a seleção do simulador, foram considerados somente os simuladores específicos para NoC com foco no desempenho e tempo de execução, e que foram publicados em artigos científicos. A Tabela 5.1 mostra um breve resumo desses simuladores e retratam suas características de manipular tráfego por entrada de traço, utilizar a comunicação sem fio, possuir o código fonte disponível e suportar a comunicação MPI. Os simuladores Naxim e NNSE não estão presentes na Tabela 5.1 pois não apresentam nenhuma das características comparadas.

É possível observar que nenhum simulador suporta todas as características elencadas nesta comparação. Além disso, o tempo de processamento não é considerado por nenhum dos simuladores. O simulador Noxim foi escolhido para ser estendido e contemplar todas as características necessárias para avaliar o desempenho da ENoC.

Tabela 5.1: Comparação dos simuladores para NoC.

Simulador	Dirigida a traço	Comunicação sem fio	Código fonte disponível	Suporte a MPI
Noxim'15		•	•	
Booksim'13	•		•	
Hornet'12	•		•	
Topaz'12	•		•	
HNOCS'12	•		•	
NoC OMNeT++'11	•		•	
WNoC Simulator'11		•		
Darsim'10	•		•	•
Netrace'10	•		•	
Garnet'09	•		•	
gpNoCsim'07			•	
Nirgam'07	•		•	

5.1 Simulador Noxim

Noxim é um simulador com precisão de ciclos, desenvolvido em C++ e integrado com a biblioteca System-C, voltado para arquiteturas NoC com e sem fio, capaz de estimar o desempenho e consumo de energia. Ele foi criado em 2005 e seu código-fonte foi disponibilizado em repositórios *online* para a comunidade científica, posteriormente os autores publicaram em Catania et al. (2015) e Catania et al. (2016). Embora as publicações sejam recentes, o Noxim foi bem aceito por pesquisadores na área de NoC para realizar experimentações de novas arquiteturas, inclusive foi estendido em outros trabalhos antes mesmo de suas publicações em meio científico.

Este simulador trabalha com dois conceitos principais: *tiles* e a infraestrutura de comunicação. Os *tiles* são os nós que computam ou armazenam e possuem roteadores ligados a ele; a infraestrutura de comunicação consiste de roteadores para interligar o *tile* com seus vizinhos por ligações metálicas e opcionalmente a um *hub* sem fio. O *hub* conecta com o *tile* por ligação metálica e com outros *hubs* por ligação sem fio. Assim, há três padrões de comunicação: *tile*-para-*tile*, *tile*-para-*hub* e *hub*-para-*hub*. A comunicação é realizada por meio de pacotes divididos em pequenas partes chamadas de *flits*.

O modelo do sistema NoC é definido em um arquivo de configuração de alto nível no formato *Yet Another UML front end* (YAML) (Sinha et al. (2000)). Este formato é uma extensão das notações *Unified Modeling Language* (UML) para modelar *hardware* e permitir que o usuário insira informações sobre o sistema modelado. Os parâmetros de configuração são organizados em três grupos, a configuração por ligações com fio, sem fio e os parâmetros da simulação.

A configuração com fio informa a dimensão da malha, os *buffers* internos, tabela de roteamento, algoritmos de roteamento e estratégia de seleção entre múltiplas direções de saída. A configuração sem fio configura o *hub* sem fio e seus *buffers*, os canais e a taxa de transmissão, política de controle de acesso e *bit error rate*. Os parâmetros de simulação configuram a frequência de operação (*clock period*), tempo de *reset*, tempo de *warm up*, uma *flag* para uso da rede sem fio, outra *flag* para o modo *debug*, modo de traço do simulador, tamanho do pacote, taxa de injeção de pacote, probabilidade de retransmissão e distribuição do tráfego.

Em Kurimoto et al. (2013), o Noxim foi usado juntamente com o QEMU para formar um co-simulador de *hardware/software* para NoC. Cada núcleo da CPU foi simulado por um PE e conectado por uma conexão TCP com o simulador. O Noxim interage com o QEMU como um PE formado por núcleos de *CPUs*.

O simulador Noxim foi estendido e melhorado por vários autores. Originalmente ele suportava somente topologia em malha, em Swaminathan et al. (2014), os autores apresentam uma melhoria ao Noxim para suportar outras topologias NoC. Eles implementam as topologias malha, torus e *twisted torus*. Esses autores também propõem diferentes algoritmos de roteamento para testar as melhorias.

Em Pires et al. (2017) o Noxim foi estendido para suportar a entrada de traços gerados a partir de suítes *benchmarks* externas ao simulador. Além disso, os autores implementam um sistema de controle de fluxo chamado *send-wait-send* para aproximar a simulação ao tempo de processamento dos PEs do sistema e corrigir vazamento de memória detectado ao utilizar grandes cargas de trabalho.

A seção 5.2 apresenta uma extensão ao Noxim para simular a ENoC. Esta extensão considera a entrada de traços externos baseados em mensagens, e permite ao simulador utilizar uma carga de trabalho realística e controlada em diferentes cenários da NoC, considera o tempo de processamento gerado pela carga de trabalho de cada PE, e suporte a processamento com passagem de mensagem. Com a extensão os resultados são mais realísticos e se aproximam ao tempo real de execução das aplicações testadas.

5.2 Modificações no simulador

As modificações realizadas foram no arquivo YAML de configuração e no código fonte do simulador Noxim. As modificações são:

- *traffic_distribution*: no Noxim original a distribuição de tráfego pode ser *random, transpose, hot spot, table based, bit reversal, shuffle and butterfly*. Nesta extensão, foi adicionado a opção *traffic trace* para suportar as entradas de traços externas;

- *traffic_trace_filename*: este parâmetro informa o nome do arquivo de traço (ex.: *trace.txt*). Cada PE no sistema deve ter seu arquivo de traço correspondente, mesmo se vazio, o nome do arquivo será concatenado com sua identificação. Por exemplo, em um sistema com 4 PEs deve existir os arquivos de traço 000_*trace.txt*, 001_*trace.txt*, 002_*trace.txt* e 003_*trace.txt*;

- *traffic_trace_flit_headtail_size*: este parâmetro informa o tamanho do cabeçalho e cauda somados do pacote para a tecnologia de rede modelada. Esta característica auxilia o simulador a estimar a taxa de transferência de dados considerando a sobrecarga de cada tecnologia de comunicação.

Para suportar os novos parâmetros e a geração do pacote descritos no arquivo de traço, múltiplos códigos fonte foram modificados. A seguir, apontamos cada arquivo de código-fonte e uma breve descrição das modificações.

- *ConfigurationManager.cpp*: o método *loadConfiguration()* foi alterado para carregar os novos parâmetros de configuração do tráfego de traço do arquivo YAML, e estes novos parâmetros são validados no método *checkConfiguration()*;

- *GlobalTrafficTrace.cpp*: novo arquivo de código adicionado ao simulador. Ele é responsável por receber a entrada de traço e inseri-lo no simulador;
- *Hub.cpp*: alterado para corrigir um vazamento de memória, presente na versão atual do Noxim. Os métodos *antennaToTileProcess()* e o *tileToAntennaProcess()* contêm falhas de alocação de memória o que leva ao travamento do sistema com enormes arquivos de traços;
- *Noc.cpp*: um controlador de entrada de traço foi adicionado no método *buildMesh()*, de tal forma que o simulador reconheça quando todos os traços de entrada estão completos para que possa parar a simulação;
- *ProcessingElement.cpp*: este arquivo recebeu a maioria das modificações. No método *rxProcess()* o controlador de tráfego de traço implementa o critério de parada, o qual permite o simulador parar após todos traços de todos PEs serem trafegados. O controlador de tráfego usa uma fila para controlar os flits do *buffer* e os transmitidos, e ele é usado pelos métodos *rxProcess()* e *txProcess()*. No método *canShot()*, os traços carregados para cada PE são executados, um a um, da seguinte forma:
  - (a) verifica no fluxo de controle se o flit pode ser enviado;
  - (b) se o fluxo de controle permitir o envio, um novo pacote será criado, formado pelos endereços de origem e destino, *timestamp*, carga útil (tamanho da mensagem mais o cabeçalho e cauda) e eventuais preenchimentos se o tamanho do pacote for menor que o tamanho mínimo.

Considerando que o arquivo de traço pode conter milhões de linhas, e cada linha contém informações a serem transmitidas por meio de pacotes, estes pacotes são gerados somente se o simulador puder enviá-los para seus destinatários, reduzindo a sobrecarga de memória do simulador. Além disso, uma limitação do Noxim foi removida para permitir executar simulações maiores do que 2147 milissegundos. Para isto, as variáveis estatísticas e de controle foram aumentadas dentro do Noxim.

5.3 Suporte à entrada de traços de aplicação

No sentido de ser o mais genérico possível, foi escolhido usar um traço baseado em formato de texto puro. O modo de entrada de traço, bem como o nome do arquivo, devem ser informados na configuração YAML. Este formato de arquivo é bem simples, cada linha deve conter o nome da primitiva MPI, o *timestamp* inicial e final, o PE de destino e o tamanho da mensagem em bytes. O tamanho da mensagem considera somente a carga útil sem qualquer encapsulamento do protocolo e sobrecarga MPI. Note que o traço não contém o remetente da comunicação, pois cada PE tem seu próprio arquivo. O traço pode ter múltiplas linhas para informar toda a carga de trabalho (múltiplas mensagens a enviar).

A Figura 5.1 ilustra este formato. A Figura 5.2 mostra um exemplo do formato de entrada, a primitiva MPI usada é o *MPI_Alltoall*, o tempo de início é 192605257 nanosegundos, o tempo final é 256628513 nanosegundos, o PE 15 é o destino e o tamanho a mensagem é 4 bytes.

O nome da primitiva MPI indica o tipo da primitiva na simulação. A maioria das primitivas MPI usadas nas aplicações *NAS Parallel Benchmark* (NPB) e a barreira de sincronização (*MPI_Barrier*) foram implementadas nesta extensão do Noxim. O tipo da primitiva e seus *timestamps* associados são essenciais para simular o tempo de processamento dos PEs.

```
<MPI_Primitive> <Start_Time> <End_Time> <Destination> <Message_Size>
```

Figura 5.1: Formato do traço de entrada para um PE específico.

```
MPI_Alltoall 192605257 256628513 15 4
```

Figura 5.2: Exemplo de entrada de traço.

5.4 Suporte a comunicação MPI

O suporte as primitivas MPI foram também adicionadas ao Noxim para suportar a característica da comunicação na ENoC. As primitivas suportadas são:

- MPI_Allgather
- MPI_Allgatherv
- MPI_Allreduce
- MPI_Alltoall
- MPI_Alltoallv
- MPI_Bcast
- MPI_Gather
- MPI_Gatherv
- MPI_Reduce
- MPI_Scatter
- MPI_Scatterv
- MPI_Send
- MPI_Isend

Estas primitivas apresentam comportamentos distintos em relação a sua comunicação, o que influenciam diretamente em sua instrumentação para geração de carga de trabalho. Além disso, elas podem ser coletivas, bloqueantes ou barreira de sincronização. O apêndice A apresenta uma breve introdução a comunicação MPI e uma descrição de suas primitivas e a subseção 6.1.1 apresenta os passos utilizados para instrumentá-las na geração da carga de trabalho.

As primitivas *MPI_Send* e *MPI_Isend* são bloqueantes. Elas indicam que o destinatário fica bloqueado até ele receber um *ACK* do destino. Essa implementação considera que em

cada PE as mensagens são mantidas com em um *buffer* de entrada até chegar a confirmação do recebimento de envio. O *MPI_Send* bloqueia o processo e toda comunicação do PE, enquanto que o *MPI_Isend* somente bloqueia a comunicação. As *ACKs* podem utilizar a técnica de *Piggyback* em outras mensagens ou através de links de controle dedicados. Nesta extensão o *ACK* é implementado de forma virtual, usando uma estrutura global no simulador.

A barreira MPI é usada para sincronizar a comunicação em um dado momento, ela foi incluída no traço para indicar o *timestamp* de cada PE. Ao detectar esta barreira no arquivo de traço, cada PE levanta uma *flag* local que indica a barreira e incrementa um contador global. O contador é compartilhado entre todas as *threads*, e indica o número de PEs bloqueados. Quando o contador indica que todos PEs estão bloqueados, todas as *flags* são reiniciadas e os PEs prosseguem com suas comunicações.

5.5 Suporte ao tempo de processamento

Outra grande vantagem nesta versão estendida do Noxim é a simulação do tempo de processamento dos PEs. O tempo de processamento de pacote é calculado pela diferença entre o *timestamp* final do pacote e o *timestamp* inicial do próximo pacote. No exemplo mostrado na Figura 5.3 a primitiva *MPI_Send* inicia em 2625852823 nanossegundos e termina em 2625853538 nanossegundos, enquanto que o *MPI_Reduce* inicia em 2704602479 e termina em 2704607248. Isto significa que entre 2625852823 e 2704602479 há um tempo de processamento de 78749656 nanossegundos.

```
MPI_Send          2625852823 2625853538 2 4
MPI_Reduce        2704602479 2704607248 0 4
```

Figura 5.3: Exemplo de traço para obtenção do tempo de processamento.

Ao agendar um pacote para transmissão, o simulador verifica se há mais pacotes a serem enviados. Então ele calcula o tempo de processamento como mencionado anteriormente. Durante o tempo de intervalo, o PE fica ocupado e nenhum novo agendamento de transmissão é permitido.

Entretanto, incluir o tempo de processamento na simulação pode implicar em alto tempo de simulação para grandes cargas de trabalho com alto tempo de processamento entre as mensagens. Para sobrepor esta questão uma nova característica ao Noxim foi incluída, chamada de *Step Forward*, ela consiste em rápidos avanços da simulação durante o tempo de processamento dos PEs reduzindo o custo de tempo total da simulação, porém preservando os ciclos necessários. Em cada passo da simulação duas condições são verificadas: *i)* PE está processando o tempo; *ii)* não há flits na fila de recepção. Se ambas condições forem satisfeitas o tempo é rapidamente avançado até o menor tempo de processamento. O rápido avanço de tempo é contabilizado em todos os cenários.

5.6 Consideração do capítulo

Este capítulo apresentou a ferramenta de simulação usada para avaliar o desempenho da ENoC. Foram adicionados novas caraterísticas no simulador Noxim que permitem utilizar cargas

de trabalho de aplicações reais, considerar as características das primitivas MPI, e calcular o tempo de processamento utilizado para computar a solução da carga de trabalho. Esta extensão foi a primeira a considerar o tempo de processamento e não somente o tempo de transmissão da rede.

Simulação é uma das principais ferramentas usadas para analisar novas propostas no campo de redes em chip. Entre todos os simuladores estudados, o Noxim destaca-se por suas características e precisão próxima a sistemas reais. Entretanto, traços de aplicações reais não são facilmente suportadas pelo método de tabela presente no Noxim, o qual requer que o utilizador informe os comportamentos gerais do sistema (taxa de injeção de tráfego) e não consideram os parâmetros de tamanhos total e mínimo dos pacotes, e nem o tamanho do cabeçalho e cauda.

Neste trabalho, nós melhoramos o simulador Noxim no sentido de considerar o tempo de processamento do PE e aceitar entradas externas formadas a partir de instrumentação de aplicações que utilizam MPI pra computar uma solução. Além disso, agora o Noxim considera características únicas de pacotes de cada sistema de interconexão. Vale observar que as alterações adicionaram novas características ao Noxim, dessa forma as simulações que não utilizam essas novas características não sofrem impacto em seus resultados. A validação e teste do Noxim estendido está apresentado no Capítulo 6, bem como descrição dos resultados obtidos com essa ferramenta.

6 Avaliação de desempenho da ENoC

Este capítulo apresenta a avaliação da ENoC baseada em simulações com carga de trabalho de aplicações reais. São apresentados a carga de trabalho e os sistemas de interconexão utilizados como base de comparação com a ENoC. As cargas de trabalho são baseadas em *benchmarks* criadas a partir da suíte *NPB* e suas aplicações paralelas. Os detalhes dos parâmetros e métricas da avaliações são apresentados em seguida, juntamente com os resultados e análises.

6.1 Carga de trabalho

A carga de trabalho é a parte mais crucial de uma avaliação de desempenho, pois é possível obter conclusões equivocadas se a carga de trabalho não for propriamente selecionada, invalidando a avaliação do sistema proposto (Jain (1991)). A carga de trabalho selecionada para avaliar a ENoC é formada por aplicações que utilizam a comunicação MPI entre os processos no sistema para computar soluções a determinados problemas. O Apêndice A apresenta uma introdução a comunicação MPI e suas primitivas.

Esta seleção de carga de trabalho é apropriada pois vai ao encontro da arquitetura proposta com comunicação entre os PEs de diferentes sistemas, ao expandir ou contrair seu poder de computação. Essa carga de trabalho aproxima a arquitetura proposta à realidade pois as diversas mensagens instrumentadas das aplicações NPB e trocadas entre os PEs são os passos necessários para resolver os problemas da suíte *Benchmark*. Além disso, o tempo de processamento necessário para computar cada operação também é considerado.

6.1.1 Geração da carga de trabalho

A carga de trabalho foi gerada a partir das primitivas MPI utilizadas para resolver os problemas das aplicações NPB. Para isso, foi utilizado um invólucro dentro da biblioteca *Multi Processing Environment* (MPE), a partir da instrumentação do seu arquivo de traços [1].

O MPE é uma biblioteca que explora as características do MPI e provê facilidades e rotinas uteis como *debugging* de desempenho e corretude, gráficos, etc., a versão utilizada nesta pesquisa foi a 2.4.7. Esta biblioteca é executada juntamente com o *MPI Chameleon* (MPICH). O MPICH é uma implementação portátil do padrão MPI, a versão utilizada foi a 3.2. Esta implementação provê as ferramentas necessárias para a compilação e execução dos programas MPI com separação entre gerenciamento de processos e a comunicação. O código fonte, binários e manual podem ser obtidos em MPICH (2018).

A instrumentação foi implementada para obter informações de comunicação de cada *thread*, ou seja, serão gerados novos arquivos de traços um para cada *thread*. Estes arquivos representam cada PE que compõe a ENoC na simulação, cujo próprio nome do arquivo representa o PE que origina a mensagem.

[1] mpe2-2.4.7/src/wrappers/src/trace_mpi_core.c

No método *MPI_Init*() foi iniciado um *buffer* para registrar todas as informações da carga de trabalho. Após, em cada método que implementa as primitivas MPI este buffer é preenchido com as informações instrumentadas, um *buffer* para cada informação. No método *MPI_Finalize*() evocado ao final da execução, todos os *buffers* são escritos em arquivo de texto plano. As comunicações coletivas são escritas utilizando laço de repetição, sejam elas um para muitos ou muitos para muitos. A Figura 6.1 apresenta uma ilustração para representar a instrumentação da primitiva *MPI_Send*, foram retiradas algumas variáveis e argumentos para melhor visualização.

```
int MPI_Send(){
  trace_output_begtime[trace_line] = PMPI_Wtime();
  returnVal = PMPI_Send(buf,count,datatype,dest,tag,comm );
  trace_output_endtime[trace_line] = PMPI_Wtime();
  trace_output_size[trace_line] = (count * sizeof(datatype));
  trace_output_dest[trace_line] = dest;
  strcpy(trace_output_buffer[trace_line], "MPI_Send");
  return returnVal;
}
```

Figura 6.1: Ilustração da instrumentação realizada.

As informações inseridas nos arquivos de traços gerados são:

- *Nome da primitiva MPI*: esta informação é necessária para indicar ao simulador como esta mensagem deve ser tratada;
- *Tempo inicial e final (timestamp)*: os *timestamps* foram obtidos com uso da função *PMPI_Wtime*() antes e depois de cada rotina, e da função *PMPI_Wtick*() para obter a resolução do tempo. A marcação de tempo inicial e final para cada mensagem permite ao simulador calcular o tempo de processamento de cada operação, isto aproxima a simulação à realidade;
- *Tamanho da mensagem*: a mensagem foi obtida em bytes através da multiplicação do tamanho (ou quantidade) de mensagens enviadas e seu tipo. Esta informação permite ao simulador calcular a quantidade de pacotes necessários para trafegá-la, ou mesmo preencher um pacote para alcançar o tamanho mínimo;
- *Destino*: traz um número identificador do destino, em um simulador esta informação indica a qual PE o pacote deve ser entregue.

A instrumentação MPI manipula as primitivas registrando tempo de início, fim, origem, destino e tamanho das mensagens. Somente a *thread* que invoca a primitiva MPI usa a instrumentação. Ao fim do registro das primitivas, a informação é salva no arquivo.

As primitivas devem ser manipuladas de acordo com suas características: todos-para-todos, todos-para-um ou um-para-todos. A comunicação todos-para-todos representa as seguintes primitivas: *Allgather, Allgatherv, Allreduce, Alltoall* e *Alltoallv*. A comunicação todos-para-um

representa as primitivas *MPI_Gather, MPI_Gatherv* e *MPI_Reduce*. Comunicação um-para-um representa as primitivas *MPI_Bcast, MPI_Scatter* e *MPI_Scatterv*.

A comunicação todos-para-um é implementada por meio de um laço durante a comunicação MPI que entrega mensagens para todos PEs. A comunicação todos-para-um não necessita de codificações adicionais. Todas as comunicações MPI são separadas por *thread*, a própria *thread* registra a transmissão e salva o *log* em um arquivo. A comunicação um-para-todos segue a mesma implementação usada em todos-para-todos, exceto que o laço é restrito ao PE remetente.

Nem todas primitivas existentes na biblioteca foram instrumentadas. Durante os testes e simulações optou-se em suportar preferencialmente as utilizadas pelas aplicações NPB e que impactam diretamente no desempenho do sistema. As primitivas MPI_Recv e MPI_Irecv não foram instrumentadas para geração da carga, e são tratadas no simulador com um *Ack* para informar ao remetente o seu correto recebimento por meio de um vetor global.

6.1.2 Aplicações NPB

Os traços da carga de trabalho são gerados a partir da execução de aplicações que computam soluções de problemas. Para esta avaliação de desempenho foram selecionadas aplicações paralelas de métodos numéricos para simulações de problemas de aerodinâmica e dinâmica de fluídos. Estas aplicações são disponibilizadas pela suíte NPB, trata-se de *benchmarks* criadas pela NASA para auxiliar as avaliações de desempenho de computação paralela.

A suíte NPB possui várias aplicações, como descritas em Bailey et al. (1992), Bailey et al. (1995) e VanderWijngaart e Haopiang (2003), cada aplicação cobre diversos padrões de comunicação e processamento. Dentre essas aplicações, as sete principais foram selecionadas para avaliar a ENoC, elas são:

- *Block Tri-diagonal (BT)*: nesta aplicação cada processador é responsável por múltiplos subconjuntos cartesianos de um conjunto inteiro de dados. Esta aplicação computa soluções para dinâmica de fluído, além de testar a comunicação entre conjuntos desassociados (*uncoupled systems*) no sistema;

- *Conjugate Gradient (CG)*: usa o método *Conjugate Gradient* para computar aproximações de menores autovalores (*eigenvalues*) de uma matriz esparsa não estruturada. Esta aplicação testa comunicações de longas distâncias e irregulares;

- *Fast Fourier Transform (FT)*: resolve a transformada rápida, discreta e tridimensional de *Fourier*. Esta aplicação também testa o desempenho na comunicação de longas distâncias;

- *Integer Sort (IS)*: realiza a ordenação de chaves inteiras usando um algoritmo de tempo linear baseado na computação do histograma das chaves. Esta aplicação testa a velocidade da computação e o desempenho da comunicação;

- *Lower-Upper Symmetric Gauss-Seidel (LU)*: usa o esquema numérico *Symmetric Successive OverRelaxation* (SSOR) (Axelsson (1972)) para uma solução *Lower-Upper* em sistemas de blocos triangulares, regulares e esparsos para resolver problemas de dinâmica de fluídos, assim como o BT;

- *Multigrid (MG)*: usa o método *V-Cycle Multi-Grid* para computar a solução da equação escalar tridimensional de *Poisson*. Esta aplicação testa comunicações de longas e curtas distâncias;

- *Scalar Pentadiagonal (SP)*: nesta aplicação são resolvidas equações de múltiplos sistemas independentes, não diagonais e pentadiagonal escalar para computar soluções a dinâmica de fluídos, assim como o BT e LU. E de forma similar ao BT, esta aplicação também testa a comunicação em multipartições (ou sub-blocos) do sistema durante a computação.

As aplicações do NPB possuem diferentes tamanhos dos problemas (*benchmark classes*), em ordem ascendente de tamanho elas são: S, W, A, B, C e D. Esta avaliação de desempenho usa o problema tamanho A, o qual é o mais usado em testes reais de máquinas, devido ao seu tamanho médio e tempo de execução razoável para ser avaliado em ambiente de simulação.

Tabela 6.1: Primitivas MPI usadas nas aplicações NPB classe A.

	Allreduce	Alltoall	Alltoallv	Barrier	Bcast	Isend	Reduce	Send	Total
BT	480	0	0	32	90	77280	15	0	77897
CG	0	0	0	16	15	0	15	45440	45486
FT	0	1920	0	16	75	0	90	0	2101
IS	2640	2640	2640	0	15	0	30	15	7980
LU	2400	0	0	32	150	0	0	759204	761786
MG	21120	0	0	96	105	0	15	11024	32360
SP	480	0	0	32	60	154080	15	0	154667

Cada aplicação utiliza um conjunto de primitivas MPI para computar a solução, a Tabela 6.1 mostra quais e quantas vezes cada primitiva é utilizada e o total de mensagens instrumentadas para cada aplicação NPB. É possível perceber que a LU contém a maior quantidade de mensagens e também o maior uso da primitiva bloqueante MPI_Send, enquanto a FT contém a menor carga de trabalho, já a aplicação IS faz mais uso de coletivas.

Embora a Tabela 6.1 apresente apenas a quantidade do uso das primitivas, esses dados norteiam um possível comportamento da simulação. No entanto, cada aplicação NPB tem um padrão de comunicação diferente, como podem ser observados em Diener et al. (2015) e Bailey et al. (1995).

6.2 Sistemas de interconexão

Além da ENoC, 5 sistemas de interconexão são considerados para avaliar a comunicação inter-chip, eles são: *Ethernet* (Eth), *Wireless Gigabit* (WiGig), *InfiniBand* (IB), *Wireless Interconnection with Code Division Multiple Access* (WI-CDMA) e *Wireless Interconnection Token-based* (WI-Token). Esses sistemas de comunicação foram selecionados por serem tecnologias aplicáveis ao cenário inter-chip, eles servem como base de comparação na comunicação inter-chip da ENoC para HPC.

6.2.1 Ethernet

A *Ethernet* (LAN/MAN Standards Committee (2016)) é uma tecnologia de rede cabeada amplamente utilizada em redes locais. Existem diferentes padrões para comunicação em redes com a Eth, nesta pesquisa consideramos o padrão 10 GE (10 *Gigabit Ethernet*). Este padrão difere dos anteriores pois opera somente sobre fibra e em modo *full-duplex* a uma taxa de 10 Gb/s. O parâmetro de *Medium Access Control* foi mantido inalterado, com o tamanho de quadro máximo de 1518 B e mínimo 64 B.

6.2.2 WiGig

O novo padrão *WiGig* (Hansen (2011)) é uma conexão sem fio que trabalha em 60 GHz com 4 canais transmitindo a uma taxa máxima de 8 Gb/s ao considerar que não há congestionamento, como apresentado em WiFi Alliance (2016). O tamanho mínimo do pacote é de 8 B e a carga útil do pacote tem tamanho variável, considerando a extensão *Protocol Adaptation Layer* (PAL) do barramento sem fio (veja WiFi Alliance (2014)). O tamanho máximo do pacote contém o tamanho máximo de carga útil (128 B) somado a sobrecarga do encapsulamento (até 16 B) e o cabeçalho é 4 B. Assim, o tamanho máximo para um pacote WiGig é 148 B.

6.2.3 Infiniband

A *IB* é uma arquitetura que suporta um grande número de aplicações na interconexão cabeada em *backplane*, como *hosts* e dispositivos de entrada e saída para *cluster*. Ela trabalha em diferentes modos cada um com determinada taxa de transferência de dados, dentre os quais o modo *Enhanced Data Rate* (EDR) possui a maior taxa de transmissão por via (*lane*), 25 Gbps (Infiniband (2016)). Nesta comparação, consideramos 2 vias de comunicação *full-duplex* com IB alcançando 50 Gb/s. Taxas de transferência próxima a 50 Gb/s com a IB já foi reportada em Gugnani et al. (2016); Alali et al. (2017). A carga útil mínima é 256 B e a máxima é 4096 B, e seu cabeçalho e cauda de pacote tem 126 B. Dessa forma, o tamanho mínimo e máximo do pacote IB é 382 B e 4222 B, respectivamente.

6.2.4 Wi-CDMA e Wi-Token

WI-CDMA e *WI-Token* são projetados para comunicação *seamless* híbrida entre conexão cabeada e interconexão sem fio para sistemas multi chip. Todavia, Shamim et al. (2015) propõem o uso de CDMA alcançando somente 6 Gb/s no enlace sem fio e Shamim et al. (2017) usa o método de evitar colisão baseado em *token* e alcança 16 Gb/s. Em ambos, os pacotes são fixos em 256 B, os tamanhos da carga útil, cabeçalho e cauda não são informados. Esta subseção apresenta apenas as informações sobre taxa de transmissão de dados e limites dos pacotes sobre esses trabalhos, estas informações são suficientes para realizar a comparação com a ENoC. O capítulo 4 apresenta maiores detalhes destas abordagens de interconexão.

6.2.5 Comparação entre os sistemas de interconexão

Tabela 6.2: Parâmetro das redes.

	Eth	**WiGig**	**IB**	**WI CDMA**	**WI Token**	**ENoC**
Cabeçalho + Cauda (B)	26	4	126	-	-	4
Carga útil mínima (B)	46	4	256	-	-	4
Carga útil máxima (B)	1500	144	4096	-	-	1500
Tamanho mínimo do pacote (B)	64	8	382	256	256	8
Tamanho máximo do pacote (B)	1518	148	4222	256	256	1504
Transferência de dados (Gb/s)	10	8	50	6	16	25

A Tabela 6.2 apresenta um resumo dos parâmetros de cada sistema de interconexão. Os valores relacionados a ENoC estão descritos no Capítulo 3. É possível perceber que a ENoC é a tecnologia de inter comunicação que possui maior flexibilidade em relação ao tamanho dos seus pacotes, variando de 8 a 1504 bytes. Isto é possível devido ao seu protocolo reduzido, tão simples quanto possível, com objetivo de mitigar sobrecarga de preenchimento nos pacotes de

comunicação, e seu cabeçalho + cauda é tão pequeno quanto possível para também não impactar na sobrecarga do sistema.

Os tamanhos máximo e mínimo para a carga útil tem relação direta com sobrecarga dos pacotes, a Figura 6.2 mostra a crescente sobrecarga em bytes para cada sistema de interconexão com o aumento do tamanho da carga útil. Pode-se perceber que a ENoC possui a menor sobrecarga e em contraponta IB possui o tamanho mínimo de carga útil mais elevado entre os sistemas de interconexão estudados (256 B) causando muito preenchimento desnecessário (*padding*). Além disso a IB possui o maior tamanho de cabeçalho + cauda, o que pode gerar alto custo de sobrecarga. Por outro lado o tamanho máximo para a carga útil do WiGig é o menor entre os demais o que causa excesso de pacotes para transmitir a informação, impactando na sobrecarga pois para cada novo pacote há adição do cabeçalho e cauda.

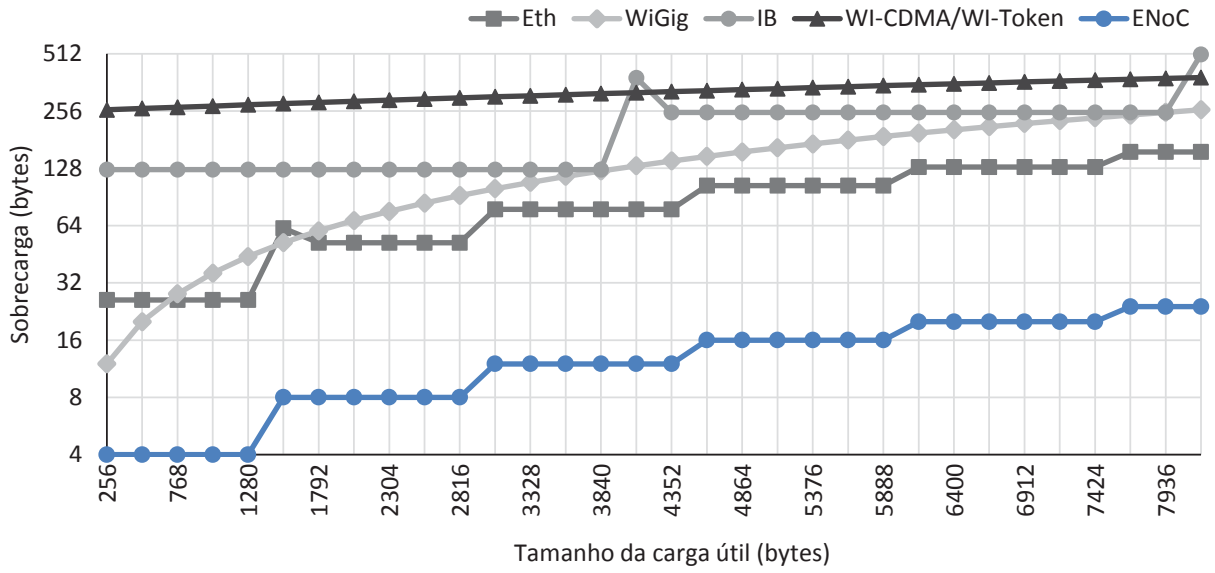


Figura 6.2: Sobrecarga dos sistemas de interconexão e da ENoC em função do crescimento da carga útil em Bytes.

Além do tamanho dos pacotes, nota-se que a IB possui a maior taxa de transferência de dados entre todos os sistemas, isto é devido a sua característica de utilizar várias vias de comunicação (do inglês, *lanes*) sendo indicada para comunicação de sistemas *Rackable*. Esta alta taxa de velocidade pode sobrepor eventuais sobrecargas de preenchimento de pacotes, dependendo da característica da carga a ser transmitida. A ENoC possui metade desta taxa de transferência de dados, porém seu desempenho pode ser próximo ou superior a IB dado seu simples protocolo de baixa sobrecarga e características da comunicação MPI.

6.3 Correlação de sobrecarga dos *Benchmarks*

Cada aplicação NPB possui uma certa quantidade de mensagens e cada mensagem tem um determinado tamanho, de acordo com a classe do problema utilizado. A coluna *Total* da Tabela 6.1 mostra esta quantidade para cada aplicação NPB com a classe *A*. A Figura 6.3 mostra o tamanho dessas mensagens.

A Tabela 6.3 está dividida em 4 intervalos: menor ou igual a 256 B, maior que 256 e menor igual a 1500 B, maior que 1500 e menor ou igual a 4096 B, e maior que 4096 B. Estes intervalos foram escolhidos considerando os limites de carga útil principalmente da IB e da ENoC. A maioria das mensagens, em todos as aplicações, possuem tamanhos maiores que 4096 B, ou

Figura 6.3: Proporção do tamanho das mensagens das aplicações NPB.

seja, causará sobrecarga na geração de vários pacotes para enviar uma mesma mensagem, e cada pacote adicionará bytes do cabeçalho e cauda. A aplicação LU possui uma pequena variação das demais, possuindo cerca de 32% da sua carga de informações com tamanho de 300 e 320 B.

Tabela 6.3: Sobrecarga das aplicações NPB.

Aplicação	Eth	WiGig	IB	WI-CDMA	WI-Token	ENoC
BT	1,79%	2,79%	3,39%	1,59%	1,59%	0,28%
CG	2,58%	2,84%	7,60%	1,64%	1,64%	0,33%
FT	1,75%	2,78%	3,10%	1,64%	1,64%	0,27%
IS	1,80%	2,78%	3,69%	1,59%	1,59%	0,27%
LU	3,89%	3,13%	15,11%	1,91%	1,91%	0,60%
MG	3,19%	2,87%	12,01%	1,69%	1,69%	0,37%
SP	1,85%	2,79%	3,47%	1,60%	1,60%	0,28%

Tanto a quantidade quanto o tamanho impactam na geração de sobrecarga de forma diferente para cada sistema de interconexão. Este impacto pode variar também de acordo com a aplicação, além dos parâmetros dos sistemas de interconexão, a Tabela 6.3 mostra a sobrecarga gerada na relação aplicações NPB $\times$ sistema de intercomunicação.

Nesta tabela, as colunas WI-CDMA e WI-Token são exatamente as mesmas pois possuem os mesmos parâmetros de mínimo e máximo, alterando apenas a taxa de transmissão. O cálculo da sobrecarga foi realizado pela razão entre a quantidade de bytes enviados e pela quantidade de bytes da carga útil. Esta sobrecarga pode ser gerado por dois fatores, preenchimento ou cabeçalho + cauda. Eles podem acontecer da seguinte forma:

- *Preenchimento*: acontece quando a informação a ser enviada possui o tamanho inferior ao tamanho mínimo da carga útil, por exemplo, ao enviar uma mensagem de 4 B utilizando a IB como sistema de comunicação, o pacote deve ser preenchido com mais 252 B para atingir o tamanho mínimo de um pacote IB. Além disso, se uma mensagem não couber em apenas um pacote, pacotes extras serão criados fracionando a mensagem em função do limite máximo de carga útil. Isto pode gerar sobra de pequenas quantidades de carga útil residual que devem ser preenchidas até atingir o tamanho mínimo do pacote;

- *Cabeçalho + cauda*: impactam na sobrecarga devido ao seu tamanho e ao número de pacotes extras gerados para enviar uma mesma informação. Em cada pacote gerado são adicionados bytes referente ao cabeçalho e cauda, contendo informações necessárias para cada sistema de interconexão funcionar corretamente.

  A adição do cabeçalho+cauda é o principal fator de sobrecarga em todas aplicações, ultrapassa 99% nas aplicações BT, FT, LU e SP para todos sistemas de interconexão, incluindo a ENoC. Nas outras aplicações o preenchimento causa impacto mais expressivo para Eth e IB, como mostrado na Figura 6.4. Na MG o preenchimento é responsável por 23, 81% para Eth e 45, 87% para IB de toda sobrecarga gerada; na CG esta proporção fica em 17, 25% para Eth e 35, 11% para IB; na IS este impacto é menos expressivo, com 1, 75% para Eth e 9, 81% para IB.

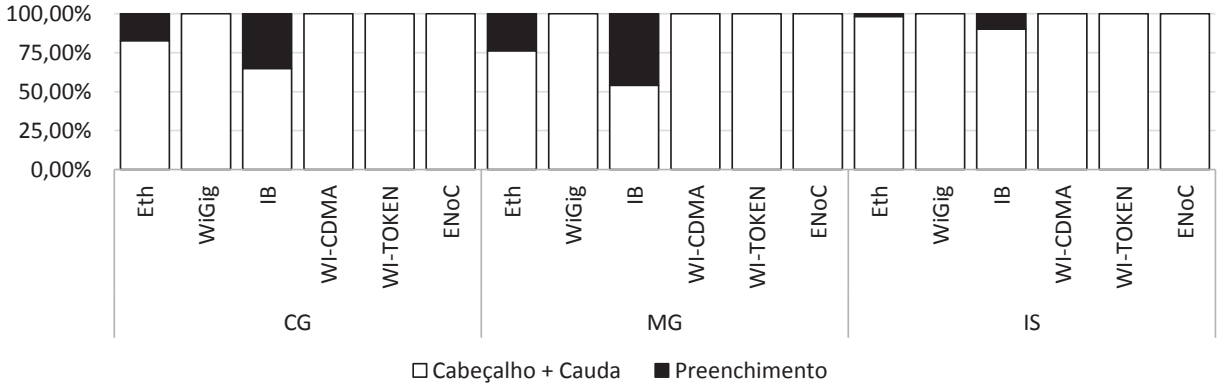


Figura 6.4: Causas da sobrecarga para relação sistema de interconexão e aplicação NPB.

A ENoC tem a menor sobrecarga em todas as aplicações NPB devido a seu protocolo simplificado, pequeno o suficiente para não haver preenchimentos desnecessários e grande o suficiente para não gerar colisões na transmissão sem fio. Além disso, mesmo a ENoC possuindo o tamanho máximo da carga útil menor que a Infiniband, o cabeçalho e cauda são pequenos para causar baixo impacto e permitir a menor sobrecarga final possível.

A IB possui a maior sobrecarga para qualquer aplicação devido aos seus parâmetros de carga útil mínimo e máximo, além do maior tamanho de cabeçalho e cauda, principal causa de sua sobrecarga. Nas aplicações CG, LU e MG esta sobrecarga fica mais evidente em relação aos demais sistemas de interconexão, em especial a ENoC. Esta sobrecarga pode afetar o desempenho das redes de alta velocidade para comunicação de uma NoC pois aumenta expressivamente o número de *flits* utilizados na computação de um problema.

6.4 Métricas e Parâmetros

Nesta avaliação consideramos a expansão de recursos de 4 sistemas. Cada sistema é formado por uma malha $2D\ 2 \times 2$ provendo uma visão única de um sistema em malha bidimensional 4×4. A palavra (*word*) possui o tamanho de 32 bits. Os diferentes sistemas de interconexão são testados para interligar os sistemas por meio de suas NoCs, além da própria ENoC. Vale ressaltar que estes sistemas de interconexão são a base de comparação para análise de desempenho. A Figura 6.5 ilustra este cenário.

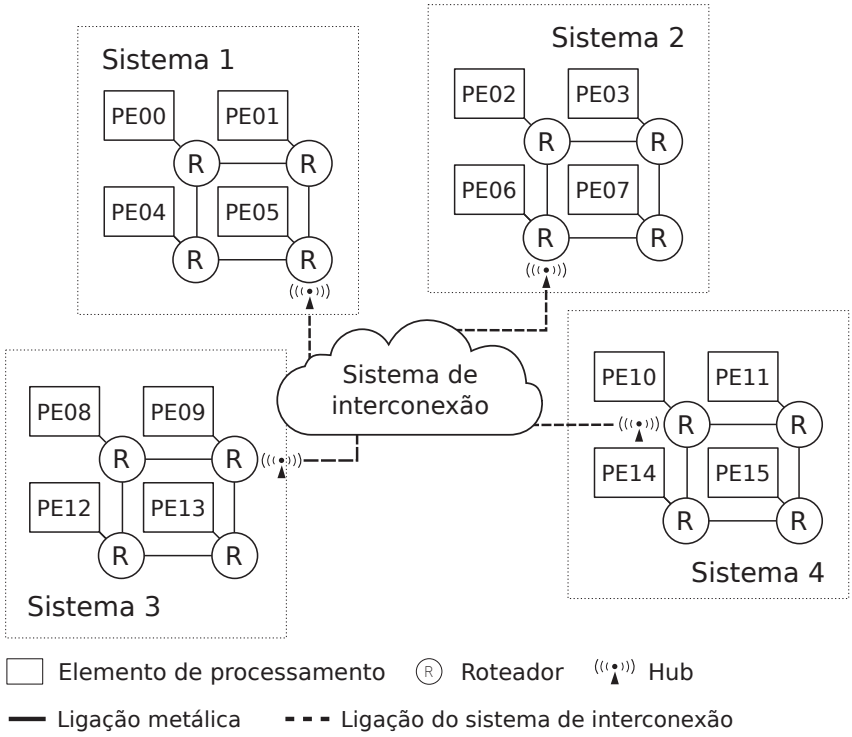


Figura 6.5: Ilustração dos sistemas avaliados.

6.4.1 Parâmetros

Os parâmetros são inseridos na simulação por meio de um arquivo de configuração YAML. A configuração dos parâmetros está organizada em: configuração básica da NoC; dos *hubs* sem fio; dos parâmetros da simulação; e parâmetros dos sistemas de interconexão. Para cada sistema de interconexão foi configurado um arquivo YAML contendo as três primeiras partes iguais em todos os arquivos, e os parâmetros da configuração dos sistemas de interconexão variam de acordo com os valores inerentes a cada sistema. A Tabela 6.4 apresenta os parâmetros dos sistemas de interconexão informados na simulação.

Tabela 6.4: Resumo dos parâmetros de simulação.

	Ether	*WiGig*	*IB*	*Wi-CDMA*	*Wi-Token*	*ENoC*
Data Rate (Gb/s):	10	8	50	6	16	25
Min. Packet Size (Flits):	18	2	96	64	64	2
Max. Packet Size (Flits):	382	37	1056	64	64	376
Flit Head+Tail Size (Flits):	26	4	126	4	4	4

O tamanho do *buffer* simulado é de 4224 bytes, ele é capaz de armazenar ao menos um pacote de tamanho máximo para todos os sistemas de interconexão. O roteamento é baseado em tabelas, e sua configuração é informada por meio de um arquivo. Esta tabela de roteamento é baseada no algoritmo de roteamento XY, porém ela separa a malha 4×4 em 4 sistemas 2×2, por meio de suas rotas. Cada sistema dividido possui rotas locais e não comunica diretamente com os demais sistemas. As comunicações remotas são redirecionadas para o roteador do sistema local que possui acesso ao *hub*.

A configuração dos *hubs* foi utilizada para informar qual roteador está ligado a qual *hub*. Como os sistemas estão divididos pelo algoritmo de roteamento, cada sistema 2×2 só comunicará com os demais sistemas por meio deste *hub*, dessa forma, ele trabalha como um *gateway* operando em um único canal, representando sistemas de interconexão com ou sem fio. Os PEs 05 do sistema 1, 06 do sistema 2, 09 do sistema 3 e 10 do sistema 4 estão ligados ao seus hubs, como ilustrado na Figura 6.5.

Os parâmetros de simulação são aplicados a todos os arquivos YAML. A frequência de operação simulada é de 1 GHz. Atribuir um tempo de simulação é indiferente pois a simulação só irá concluir quando toda a carga de trabalho for transmitida, considerando o tempo de processamento necessário para cada simulação. A carga utilizada será externa ao simulador, e informada por meio de arquivos de traço gerado a partir das aplicações NPB.

Todos os hubs comunicam em um mesmo canal a uma determinada taxa de transmissão referente a cada sistema de interconexão, este parâmetro é informado ao simulador em Gb/s. O tamanho mínimo e máximo são dados em *flits*, bem como o tamanho do cabeçalho e cauda de cada um. Os parâmetros de tamanho mínimo e máximo e do cabeçalho e cauda do WI-CDMA e WI-Token não foram informados pelos autores, e para uma justa simulação, foram considerados os mesmos parâmetros da ENoC.

6.4.2 Métricas

Com objetivo de medir o desempenho da ENoC tomando como base de comparação os demais sistemas de interconexão, a principal métrica utilizada foi o tempo de simulação. Este tempo é dado por ciclos (ou nanossegundos) necessários para simular cada aplicação NPB testada, ele é resultado principalmente de 2 fatores: o tempo necessário para os PEs trocarem mensagens e o tempo de processamento.

O tempo de trocas de mensagens é a quantidade de ciclos necessários para transmitir e receber o total das mensagens de cada aplicação simulada, como apresentado na coluna *total* da Tabela 6.1. Vale ressaltar que o simulador Noxim modificado considera as características das primitivas MPI, como por exemplo, na presença de envio bloqueante (MPI_Send) os ciclos continuam sendo executados porém nenhuma nova mensagem será enviada pelo remetente até o seu recebimento, ou mesmo quando há uma barreira de sincronização ($MPI\_Barrier$), onde todos PEs não enviam novas mensagens esperando a chegada das mensagens na barreira.

Cada informação a ser transmitida possui um *timestamp* inicial e final para a execução da primitiva. A diferença entre o tempo inicial atual e o tempo final da mensagem anterior indica que o PE estava processando uma tarefa. A Figura 6.6 apresenta um trecho da aplicação BT, nela mostra que entre o tempo final do $MPI\_Alltoallv$ (275217056) e o tempo inicial do MPI_Reduce (276373386) houve uma diferença de 1156330 ciclos, e este tempo foi utilizado para processar a solução. O mesmo ocorre entre o MPI_Reduce e MPI_Send.

```
MPI_Alltoallv   270370244   275217056   15   132104
MPI_Reduce      276373386   276396989   0    4
MPI_Send        285993814   285996913   13   4
```

Figura 6.6: Trecho de arquivo de traço da aplicação BT.

Além disso, a sobrecarga dos pacotes também foi considerada na simulação. Os parâmetros que definem esta sobrecarga são os tamanhos mínimo e máximo dos pacotes, além da quantidade dos pacotes gerados pelas aplicações NPB. A sobrecarga pode ser gerada pela necessidade de preenchimento de um pacote para alcançar o seu tamanho mínimo, quando a carga útil for muito pequena.

Se a informação a ser transmitida for muito grande ela será dividida em vários pacotes determinados pelo parâmetro de tamanho máximo, e cada pacote terá o cabeçalho e cauda adicionados ao seu tamanho. Isto também causa sobrecarga. A correlação das sobrecargas geradas pelos *benchmarks* foi explorada na Subseção 6.3.

6.5 Avaliações de desempenho

A avaliação de desempenho foi dividida em 4 partes: tempo da comunicação; crescimento de PEs por sistema; expansão de sistemas ENoC; e depempenho considerando o tempo de processamento. Para todas as avaliações foram consideradas as primitivas bloqueantes e a barreira sincronizante.

6.5.1 Avaliação da comunicação inter-chip

A primeira avaliação permite mensurar o desempenho do impacto de cada sistema de interconexão na expansão da NoC. As Figuras 6.7 e 6.8 mostram o tempo de execução necessário para cada rede de interconexão para transmitir todas as mensagens, sem o tempo de processamento. A ENoC possui o melhor desempenho em quase todas as aplicações, seguido por IB e WI-Token. A WI-CDMA possui o pior desempenho seguido por Eth e WiGig.

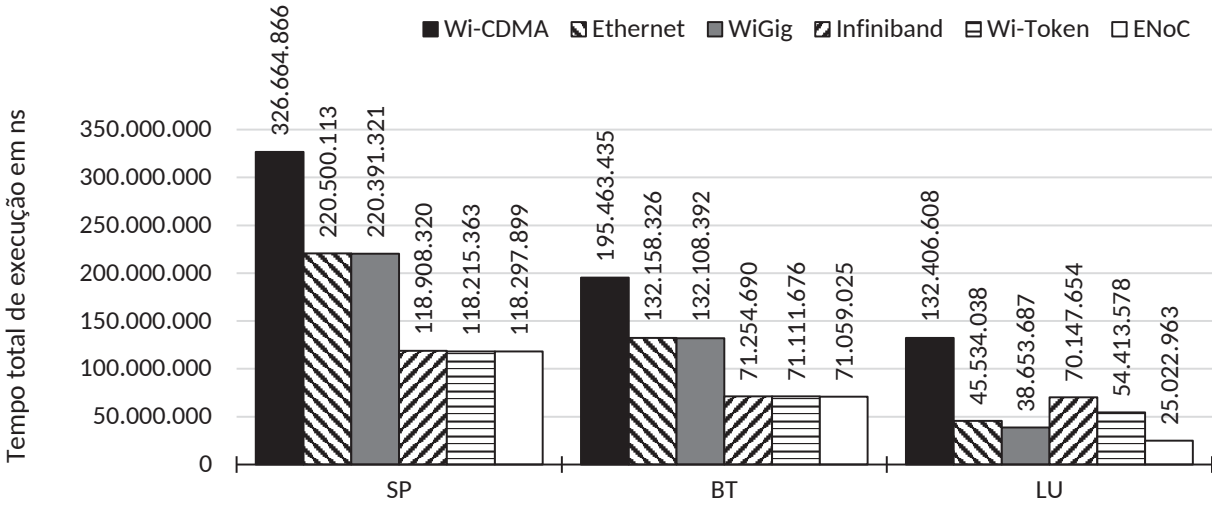


Figura 6.7: Tempo de execução sem tempo de processamento - parte 1.

O principal parâmetro que influencia nestes resultados é a taxa de transferência de dados. No entanto, a ENoC possui metade da taxa de transferência do que a IB e um desempenho igual ou superior para estas aplicações testadas. Isto acontece pois o tempo necessário para a barreira sincronizante e a confirmação de recebimento de uma primitiva bloqueante atenua as diferenças da taxa de transmissão de dados. Além disso, a sobrecarga gerada pelo protocolo IB resultou em

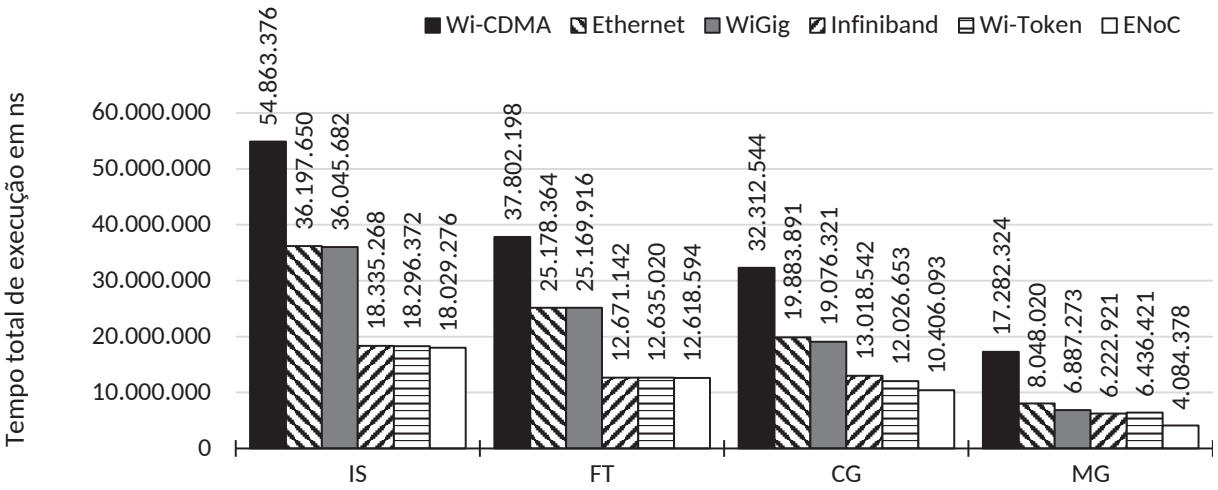


Figura 6.8: Tempo de execução sem tempo de processamento - parte 2.

flits demasiados afetando seu desempenho. O WI-Token aproximou-se da ENoC pois o parâmetro de tamanho de carga útil mínima não foi informado pelo autor e para uma justa comparação foi usado o parâmetro da ENoC.

A WI-Token supera muito sua versão anterior, a WI-CDMA, devido a diferença na taxa de transferência de dados, porém ambas perdem em desempenho para a ENoC, com uma pequena exceção na aplicação SP. A principal vantagem da ENoC está no uso do OFDM, devido ao paralelismo de subportadoras de taxas baixas, permitindo alta largura de banda com baixa taxa de colisão e interferências causadas por ruídos.

É possível correlacionar a sobrecarga dos pacotes e volume de mensagens com o resultado do desempenho. Na aplicação LU, os 32% de informação que estão entre 1501 a 4096 B são enviados cada um por apenas 1 pacote IB e deveria contribuir para o melhor desempenho, mas o volume de mensagens é muito expressivo (761786 de mensagens, cerca de 731 MB) o que gerou alta sobrecarga superando seu ganho com os 32% da carga. Esta é a aplicação de maior sobrecarga para todos os sistemas de interconexão.

A aplicação FT possui a menor sobrecarga de todas os testes. Esta foi a aplicação em que o IB mais se aproximou a ENoC, com diferença de apenas $0,4\%$. Somente na aplicação SP a ENoC não apresentou menor tempo de execução, perdendo para o WI-Token por uma diferença de $0,07\%$. A aplicação SP possui a maioria das mensagens do tipo MPI_Isend, que embora não seja bloqueante, aguarda o recebimento de um *Ack* para enviar a próxima mensagem. Este atraso atenua a vantagem da largura de banda e baixa sobrecarga da ENoC. Vale observar que a WI-Token possui a taxa de transferência de dados e sobrecarga mais próximas da ENoC. Neste caso, o IB não apresentou melhor desempenho, mesmo com maior taxa de transferência de dados, devido a sua alta sobrecarga (veja a Tabela 6.3).

Se o tempo das primitivas bloqueantes e de sincronização não fossem considerados, o IB teria o melhor desempenho, como apresentado em Pires et al. (2017). Estes resultados evidenciam que a sobrecarga da informação e a taxa de transmissão são fundamentais para o desempenho da comunicação de uma NoC expansível.

6.5.2 Avaliação da escalabilidade dos PEs

A segunda avaliação permite analisar o comportamento dos sistemas de interconexão com o crescimento do número de PEs por sistema. Nesta avaliação, o foco é a variação do número de PEs divididos em 4 sistemas bidimensionais para uma mesma carga de trabalho. As cargas de trabalho escolhidas foram LU e SP, por possuirem a maior carga dentre as aplicações selecionadas nesta avaliação de desempenho e com diferentes padrões de comunicação.

As cargas de trabalhos possuem o tamanho *A* de problema e exigem que a quantidade de PEs possam formar uma malha quadrada, por isso foram escolhidos os tamanhos 4×4, 6×6, 8×8 e 10×10. Assim, a variação do número de PEs está organizada da seguinte forma: 16 PEs em 04 sistemas 2×2 (04 PEs por sistema), 36 PEs em 04 sistemas 3×3 (09 PEs por sistema), 64 PEs em 04 sistemas 4×4 (16 PEs por sistema), e 100 PEs em 04 sistemas 5×5 (25 PEs por sistema).

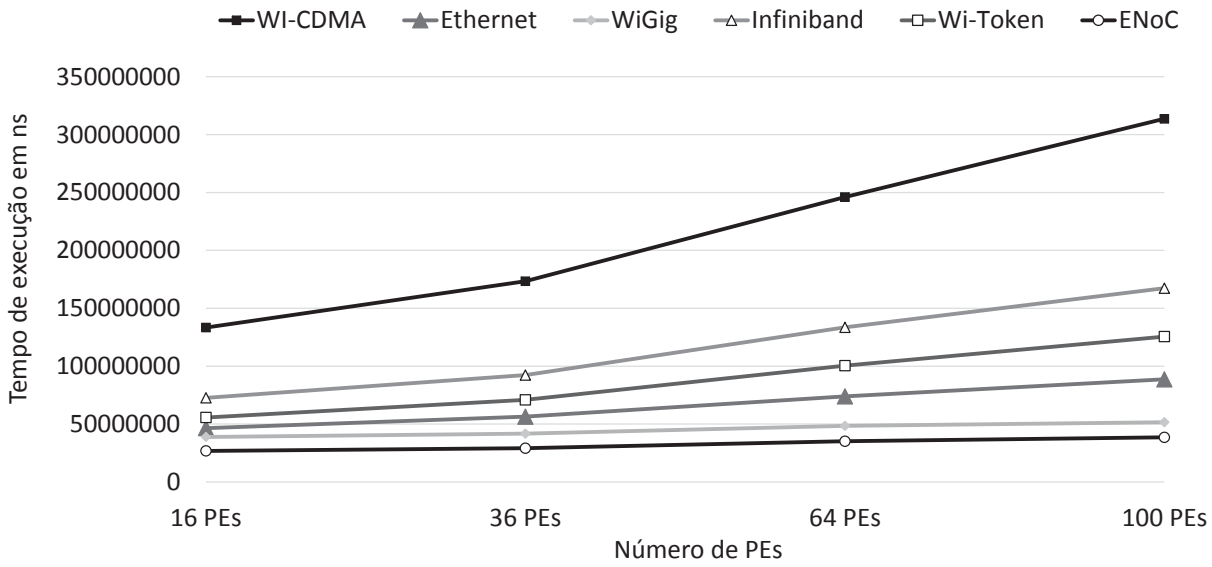


Figura 6.9: Tempo de execução com variação do número de PEs divididos em 04 sistemas executando a carga de trabalho LU.

A Figura 6.9 mostra que com o aumento de número de PEs a ENoC apresenta o melhor desempenho dentre todos os outros sitemas de interconexão. É possível correlacionar este resultado com o resulado apresentado do tempo de execução apresentado na Figura 6.7, no qual o WiGig apresentou o comportamento mais próximo a ENoC para a carga de trabalho da aplicação LU, e esta proporção segue com o crescimento linear do tempo de execução conforme mais PEs são adicionados nos sistemas.

Um comportamento similar também pode ser observado na Figura 6.10. Com a carga de trabalho SP percebe-se também o crescimento linear de todos sistemas de interconexão e também é possível perceber o desempenho da ENoC muito próximo aos sistemas de interconexão IB e WI-Token. Correlacionando com a Figura 6.7, esta proximidade pode ser percebida entres os três sistemas de interconexão, porém o teste de crescimento do número de PEs evidenciam o ponto no qual sobressai a diferença entre eles. Percebe-se também que acima de 64 PEs o ganho da ENoC torna-se mais evidente, assim como aumentando para 100 PEs. Com base nestes resultados, conclui-se que com maior número de PEs este ganho tende a aumentar.

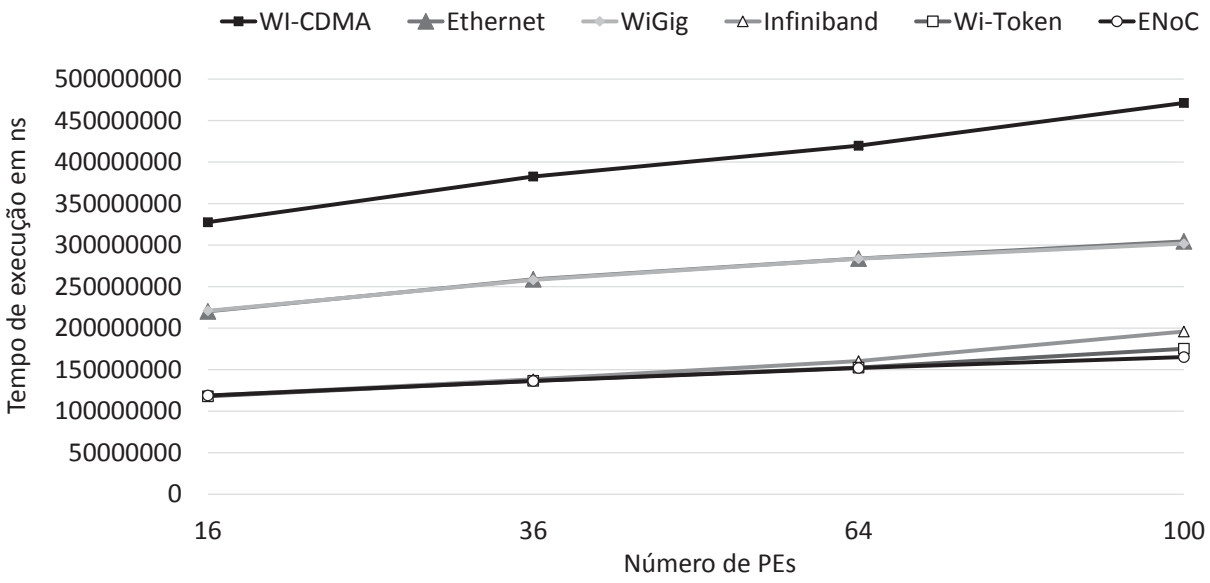


Figura 6.10: Tempo de execução com variação do número de PEs divididos em 04 sistemas executando a carga de trabalho SP.

6.5.3 Avaliação da escalabilidade dos sistemas

A terceira avaliação visa medir o ganho de tempo de execução com a ENoC considerando o crescimento do número de sistemas. Esta avaliação é feita em duas etapas com carga de trabalho criada a partir de partes da aplicação *CG*, compostas por 32 e 320 tarefas com comunicação entre todos os sistemas (quando houver). Estes valores foram selecionados a fim de permitir uma distribuição proporcional entre os PEs conforme o crescimento do número de sistemas. Este crescimento foi medido por 01, 02, 04 e 08 ENoCs, cada ENoC possui o tamanho fixo de 04 PEs locais.

Na primeira etapa as 32 tarefas foram distribuídas pelos PEs com diferentes números de ENoCs da seguinte forma: com 04 PEs em apenas 01 ENoC, cada PE processou 8 tarefas apenas com comunicação intra-chip, uma vez que há apenas 01 sistema; utilizando 02 ENoCs houve a comunicação inter-chip entre os sistemas e cada PE executou 4 tarefas; aumentando para 04 ENoCs houve a comunicação inter-chip com mais sistemas e cada PE executou apenas 2 tarefas; com 08 ENoCs, os 08 sistemas realizaram a comunicação inter-chip e cada PE executou apenas 01 tarefa.

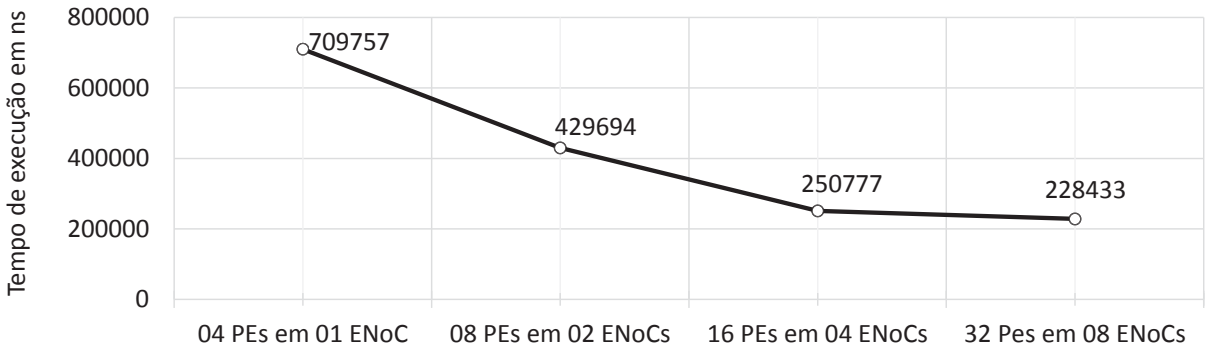


Figura 6.11: Tempo de execução da ENoC com variação do número de sistemas executando 32 tarefas.

Na segunda etapa, foram simulados os mesmos parâmetros da etapa anterior porém o número de tarefas foi aumentado em 10× com objetivo de observar o comportamento da ENoC ao aumentar consideravelmente o número de tarefas. A Figura 6.11 apresenta os resultados dos testes com 32 tarefas, enquanto a Figura 6.12 apresenta os resultados com 320 tarefas.

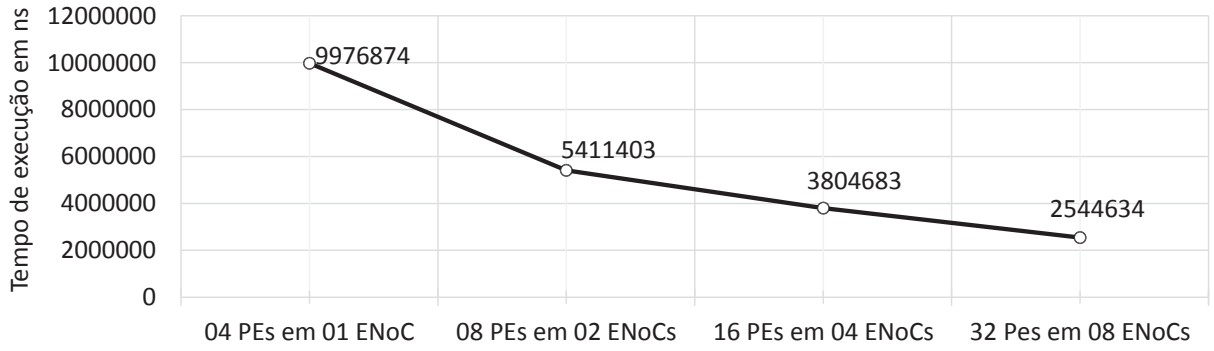


Figura 6.12: Tempo de execução da ENoC com variação do número de sistemas executando 320 tarefas.

Os resultados desta terceira avaliação mostram que quanto maior o número de ENoCs interconectadas maior é o desempenho na resolução de um mesmo problema. Estes testes consideraram a comunicação inter-chip, e mesmo com este custo adicional da comunicação a aplicação é processada com menor tempo de execução. No entanto este ganho está relacionado com a quantidade de tarefas atribuídas aos PEs de cada ENoC para realizar o processamento paralelo e distribuído.

Se considerar o caso de processamento das 32 tarefas com apenas 01 ENoC como base de comparação, há um ganho de 40% ao utilizar 02 ENoCs para processar a mesma carga de trabalho mesmo com o custo da comunicação. Dobrando para 04 ENoCs o ganho de desempenho é de 65%, uma diferença de 15% embora tenha dobrado o número de sistemas. Utilizando 08 ENoCs, com uma tarefa para cada PE, o ganho foi de 68% em relação a base de comparação, um aumento de 3% ao dobrar mais uma vez a quantidade de sistemas, isto acontece devido a pouca quantidade de tarefa para cada PE com maior custo de comunicação inter-chip, uma vez que há mais sistemas para comunicar-se.

No teste com 320 tarefas, e com as mesmas comparações anteriores, a porcentagem de ganho foi de aproximadamente 46%, 62% e 75% para 02, 04 e 08 ENoCs, respectivamente. O ganho de desempenho da ENoC neste teste foi mais evidente devido ao número maior de tarefa por cada sistema, pois o ganho do processamento paralelo atenuou o custo da comunicação entre os sistemas.

6.5.4 Avaliação considerando o tempo de processamento

A quarta avaliação tem o objetivo de medir o comportamento dos sistemas de interconexão de forma similar a primeira avaliação, porém considerando o tempo de processamento. No entanto esses testes apresentaram inconsistências nos resultados, pois toda a diferença de desempenho é atenuada pelos longos ciclos em espera necessários para computar as aplicações paralelamente. O tempo de processamento foi super estimado ao implementar o seu cálculo no simulador, o que resultou nesta incosistência dos dados obtidos não correspondendo com a realidade. As Figuras 6.13 e 6.14 mostram os resultados das mesmas aplicações e sistemas de interconexão porém considerando o tempo de processamento relativo a cada mensagem do traço NPB.

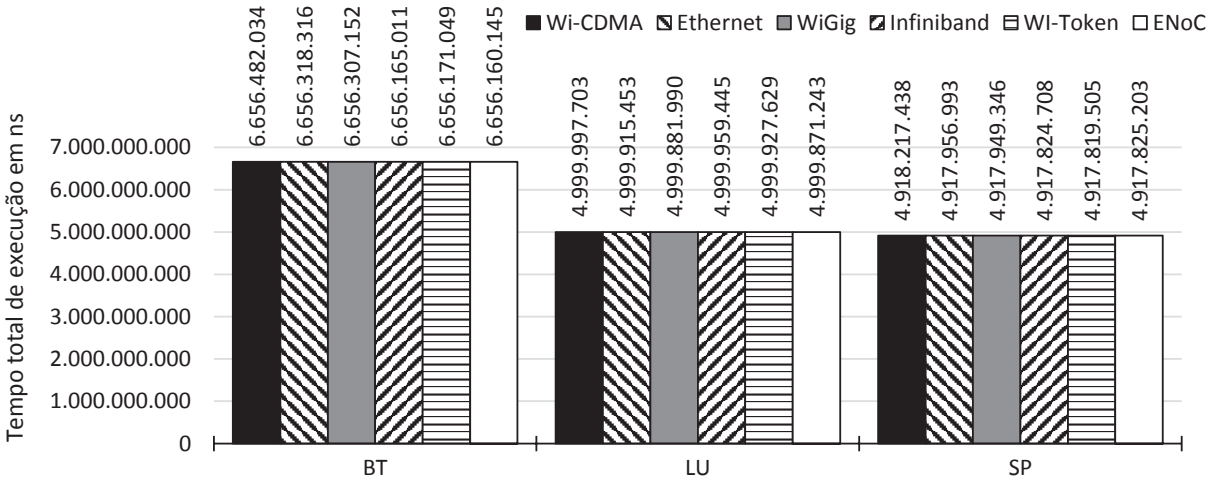


Figura 6.13: Tempo de execução com tempo de processamento - parte 1.

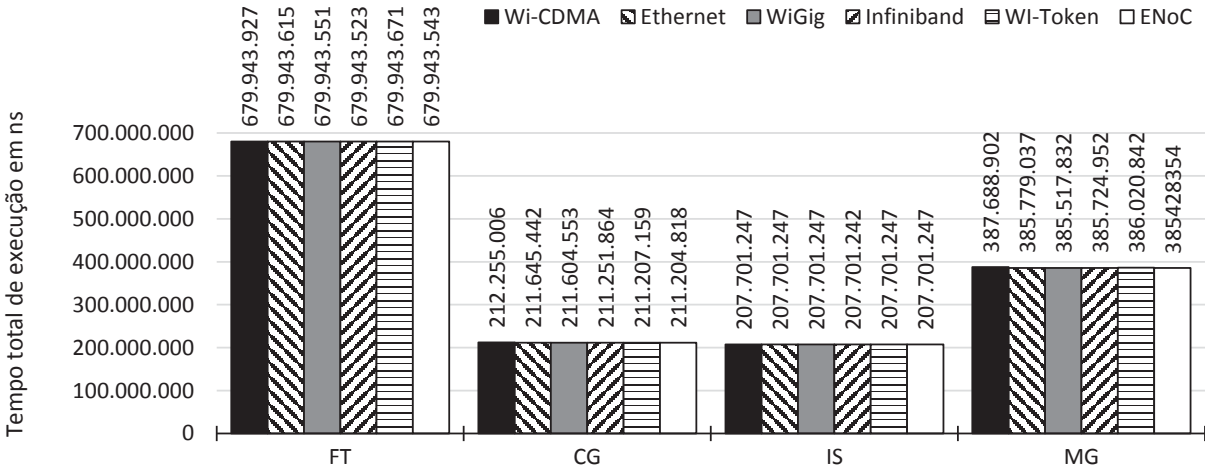


Figura 6.14: Tempo de execução com tempo de processamento - parte 2.

É possível perceber que além de atenuar as diferenças dos sistemas de interconexão, o comportamento de todo o sistema mudou. Na simulação sem o tempo de processamento, as aplicações SP, BT e LU demandam mais ciclos para sua execução, nesta mesma ordem. Mas, ao considerar o tempo de processamento esta ordem modifica para BT, LU e SP. De forma similar, a aplicação MG era a mais rápida para executar, mas ao considerar o tempo de processamento as aplicações CG e IS apresentam tempo menor. Este problema será tratado em trabalhos futuros.

6.6 Considerações do capítulo

A ENoC foi avaliada usando 7 aplicações NPB, comparando múltiplas infraestruturas de interconexão no estado da arte, como a Ethernet 10 Gb/s, Wireless Gigabit e Infiniband. Também foram comparados dois trabalhos relacionados, WI-CDMA e WI-Token. A ENoC mostrou-se com melhor desempenho em praticamente todos os testes, com baixa sobrecarga no envio das mensagens e taxa de transferência adequada e realística para implementação nas redes em chip.

Analisando todos os resultados, as especificações ENoC são facilmente alcançadas. Além de apresentar um desempenho melhor, a ENoC possui menor custo de infraestrutura, agregando expansibilidade ao apenas aproximar-se de outros sistemas ENoC. Os testes com a escalabilidade do sistema mostraram que a ENoC mantêm-se viável com o crescimento do

número de PEs por sistema e do crescimento do número de ENoCs interconectadas para processar uma mesma aplicação.

O nivelamento dos desempenhos gerado pelo tempo de processamento é resultado de uma inconsistência do cálculo de tempo de processamento, a ser enfrentado em trabalhos futuros. No entanto, os demais testes ressaltam o bom desempenho da ENoC e sua característica *on-the-fly* é seu diferencial e torna seu uso versátil para várias aplicações, principalmente por não necessitar de adaptações físicas, como a IB.

7 Confidencialidade na ENoC

A segurança na expansibilidade da ENoC deve ser garantida. Stallings (2002) classifica os ataques à transmissão de dados como ativos ou passivos. O ataque ativo altera o comportamento da comunicação e do sistema, como modificação do fluxo de dados, criação de um fluxo falso, alteração proibida do sistema e negação de serviço. O ataque passivo não modifica a comunicação ou o sistema, ele visa obter informações que estão sendo transmitidas sem que o ataque seja percebido pelo remetente ou destinatário, comprometendo a confidencialidade da informação. Este ataque é mais difícil de detectar do que o ativo, sua principal contramedida é a prevenção com uso de criptografia.

O ataque passivo pode ser realizado por meio de espionagem do tráfego de comunicação (Gebotys e Gebotys (2003)), também é chamado de extração da informação secreta em Evain e Diguet (2005), Diguet et al. (2007) e Kapoor et al. (2013) e na literatura não específica para *NoC* são encontrados os termos em inglês *sniffing*, *snooping* e *eavesdropping*. Este ataque pode ser praticado tanto por um dispositivo externo ao chip quanto por elementos comprometidos fabricados por terceiros e inseridos no sistema como uma "caixa preta" apenas para interagir com os demais elementos do chip. Esses elementos de terceiros são chamados de *Third-Party Intellectual Property*s (3PIPs).

A confidencialidade deve ser garantida tanto na comunicação intra quanto inter-chip. Existem muitos trabalhos, como Gebotys e Gebotys (2003), Sajeesh e Kapoor (2011), Kapoor et al. (2013), Ancajas et al. (2014a) e Sepúlveda et al. (2015) que oferecem soluções para a confidencialidade na comunicação intra-chip. Portanto, eles são considerados fora do escopo desta solução. A comunicação inter-chip acontece após a expansibilidade do sistema, um novo conceito apresentado pela ENoC, e não há na literatura abordagens que tratem a confidencialidade nessa comunicação para NoC.

7.1 Modelo de ataque

A garantia de confidencialidade na comunicação inter-chip é o foco desta proposta de segurança. Os cenários de ataque são todas as formas de se obter indevidamente uma informação trafegada por meio de ligações sem fio na comunicação inter-chip da ENoC. Os *hubs*, PEs ou roteadores 3PIPs podem também estar comprometidos e afetar a confidencialidade da informação em todo o sistema, este tipo de ataque é classificado na literatura como *Hardware Trojan* (HT). Muitas pesquisas visam identificá-los, isolá-los e proteger o restante do sistema deles, como Adamov et al. (2009), Yu e Frey (2013), Ancajas et al. (2014b), LeMay e Gunter (2015), Frey e Yu (2017), Salmani (2017), Rajesh et al. (2018), entre outros. Enfrentar o problema do HT foge ao escopo de garantia da confidencialidade na comunicação inter-chip e por este motivo não faz parte desta pesquisa, assim como os ataques ativos.

As comunicações sem fio são intrinsecamente vulneráveis à espionagem do tráfego, uma vez que a informação pode ser facilmente capturada por qualquer dispositivo dentro do raio de

comunicação do transmissor. A Figura 7.1 mostra um exemplo de extração da informação, nele há um sistema *C* ouvindo a comunicação entre dois sistemas ENoC expandidos, *A* e *B*. O atacante pode coletar todos os dados trafegados no meio de transmissão para posterior análise e obtenção de informação sigilosa, comprometendo a confidencialidade da informação na comunicação inter-chip com a ENoC.

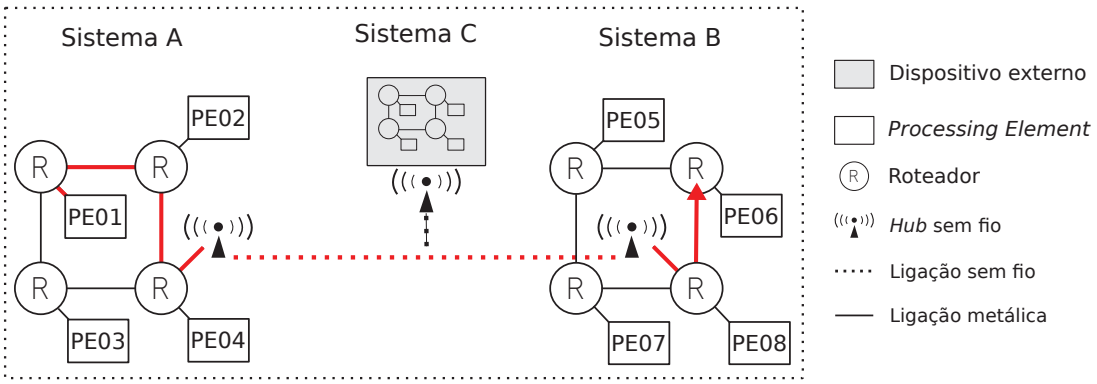


Figura 7.1: Ataque à confidencialidade na ENoC.

Este cenário de ataque pode ser realizado por qualquer dispositivo externo, e o sistema atacante pode ser uma SoC ou não. Para ser possível, basta o atacante apenas conhecer o protocolo de comunicação, frequência de operação e forma de modulação, com isso é possível analisar o tráfego em busca de uma informação sensível. Essa vulnerabilidade inviabiliza a comunicação inter-chip confiável, e dependendo do objetivo do projeto, o próprio uso da ENoC.

7.2 Protegendo a ENoC com criptografia

A confidencialidade na comunicação inter-chip da ENoC pode ser garantida com uso dos criptossistemas *Advanced Encryption System* (AES), *Elliptic Curve Cryptography* (ECC) e *Elliptic Curve Integrated Encryption Scheme* (ECIES). Cada criptossistema possui características próprias e a escolha da criptografia deve atender o objetivo e os requisitos do sistema a ser protegido, é uma decisão de projeto. Os detalhes sobre cada criptossistema e sua implementação são encontradas no Apêndice B.

7.2.1 AES

A escolha por utilizar o AES é possível se os *hubs* sem fio, ou todos os sistemas ENoC, forem produzidos por apenas um fabricante. Nessa abordagem todos os *hubs* devem compartilhar a mesma chave secreta, ela pode ser atribuída no momento de produção. A confidencialidade é garantida enquanto a chave compartilhada permanecer segura.

A premissa básica é que todos os sistemas compartilhem uma mesma chave secreta e que ela tenha o tamanho de 128 bits ou superior. Os PEs e roteadores locais não precisam saber a chave secreta, a criptografia é responsabilidade exclusiva do *hub* sem fio. A comunicação inter-chip com AES trabalha da seguinte forma:

1. O PE remetente cria o pacote e divide-o em *flits*;
2. Os *flits* são roteados na comunicação intra-chip até o *hub* sem fio local por meio de saltos entre os roteadores, de acordo com o algoritmo de roteamento utilizado;

3. O *hub* sem fio reconstrói o pacote, cifra-o com sua chave simétrica e envia-o pelo canal sem fio por difusão;

4. O *hub* sem fio do sistema destino recebe o pacote, decifra-o com a chave simétrica compartilhada, cria um pacote para encaminhar ao *PE* destinatário, divide-os em *flits* e transmite. Os demais *hubs* sem fio presentes no canal devem ignorar os pacotes não endereçados aos seus sistemas;

5. De forma similar ao envio, os flits são recebidos do *hub* sem fio pelo roteador, roteado por saltos, seguindo o algoritmo de roteamento, até alcançar seu destino;

6. O *PE* destinatário reconstrói o pacote e recupera a mensagem original.

A principal vantagem do AES é o desempenho em relação aos outros criptossistemas, pois é da natureza da cifra simétrica ter melhor desempenho do que cifras assimétricas. A principal desvantagem é sua premissa básica, o uso de uma única chave secreta para todo o sistema.

A ENoC não foi projetada para limitar-se a apenas um fabricante, embora seja possível. De fato, seu principal objetivo é integrar qualquer tipo de PE de qualquer fabricante. Neste caso, ECC e ECIES podem ser usados para assegurar a comunicação inter-chip.

7.2.2 ECC

A premissa básica para utilizar a cifra da comunicação inter-chip com a ECC é que os sistemas comunicantes conheçam a curva. Uma curva elíptica é formada por valores dos parâmetros que a compõe (ver Apêndice B.3), ela atua sob um corpo finito primo ($\mathbb{F}_p$) ou binário ($\mathbb{F}_{2^m}$). Existem uma infinidade de combinações para definir as curvas elípticas, porém há curvas testadas e recomendadas para criptografia, elas podem ser encontradas em CERTICOM (2000). As ENoCs podem estabelecer a curva utilizada por duas formas: com armazenamento prévio ou informado no *handshake* inicial de comunicação.

A Figura 7.2 ilustra a primeira forma. Ao perceber um *Beacon* de uma ENoC nas proximidades ocorre a expansão (ver subseção 3.5), neste momento a ENoC *A* pode enviar um pacote com o mínimo necessário para identificar a curva elíptica a ser utilizada na comunicação e sua chave pública, juntamente com a identificação do sistema. Todo sistema ENoC deve conter de fábrica todas as curvas suportadas em sua memória, desse forma, com a identificação de curva elíptica recebida, a ENoC *B* carrega os parâmetros das curvas elípticas selecionada e estabelece sua chave pública. A chave pública é gerada a partir dos parâmetros da curva e do ponto gerador (ver Apêndice B), ela pode ser gerada previamente e armazenada em sua memória ou gerar após receber a identificação da curva selecionada. Após, a ENoC *B* envia sua chave pública juntamente com a identificação do seu sistema. A partir deste momento as ENoCs *A* e *B* podem realizara comunicação inter-chip de forma cifrada mapeando as mensagens para pontos da curva elíptica e enviando-as pelo canal sem fio.

A segunda forma é bastante similar a primeira com a diferença de que não há a necessidade das ENoCs possuírem previamente em sua memória o rol de curvas suportadas. Os parâmetros das curvas elípticas utilizadas e o ponto gerador são enviados juntamente com a identificação do sistema no *handshake* inicial ao invés de enviar uma pequena informação da curva elíptica. O tamanho das informações da curva, considerando apenas seus parâmetros, não impacta significativamente na comunicação inicial, pois somando todos os parâmetros das curvas mais utilizadas caberiam em apenas 1 pacote ENoC, a Tabela 7.1 apresenta estes tamanhos em bytes.

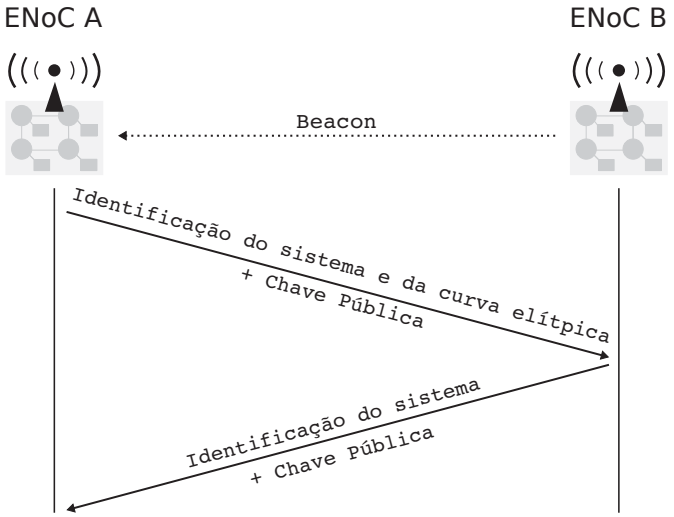


Figura 7.2: *Handshake* inicial para comunicação inter-chip com ECC.

Tabela 7.1: Tamanho total dos parâmetros de diferentes curvas.

Curva $\mathbb{F}_p$	**Tamanho (B)**	**Curva $\mathbb{F}_{2^m}$**	**Tamanho (B)**
secp192r1	122	sect193r1	102
secp224r1	142	sect233r1	122
secp256r1	162	sect283r1	146
secp384r1	242	sect409r1	210

Todos os *hubs* sem fio devem ser capazes de processar as operações de aritmética modular para a curva elíptica selecionada. Eles também devem implementar o método de mapeamento de mensagem para pontos da curva. A comunicação inter-chip com ECC funciona da seguinte forma:

1. O PE de origem cria o pacote, divide-o em flits e entrega-os a seu roteador;
2. Os *flits* são roteados pela malha com fio até alcançar o *hub* sem fio seguindo um algoritmo de roteamento;
3. O *hub* remonta a mensagem, cifra-a com sua chave privada e os parâmetros da curva elíptica já acordados, após ele transmite a informação em pacote pelo canal sem fio;
4. O *hub* sem fio do sistema de destino recebe o pacote, decifra-o com a chave pública recebida previamente, remonta os dados em um novo pacote, divide-o em *flits* e encaminha a seu roteador;
5. Os *flits* são roteados pela malha com fio até alcançar o PE destinatário, seguindo um algoritmo de roteamento;
6. O PE destinatário recebe o pacote e recupera a mensagem transmitida.

A principal vantagem da ECC é a ausência da chave secreta compartilhada, assim se um *hub* for comprometido, o resto dos sistemas continuam seguros e não há a necessidade de que todas as ENoCs sejam fabricadas por um único fabricante, diferentemente da abordagem com AES. Entretanto, a ECC pode apresentar pior desempenho ao comparar com os outros dois criptossistemas, como mostrado na subseção 7.3.

7.2.3 ECIES

O ECIES utiliza curvas elípticas para estabelecer chaves de sessão compartilhadas capazes de verificar a integridade e a confidencialidade da mensagem. De forma análoga a abordagem ECC, a curva elíptica também deve ser previamente acordada. Ela é utilizada para estabelecer as chaves efêmeras e consequentemente as chaves de sessão. As duas formas apresentadas para o acordo da definição da curva elíptica também são válidas para utilizar a ECIES.

Quando uma ENoC *A* percebe a ENoC *B*, cria uma chave pública e privada efêmeras sob determinada curva elíptica. Os sistemas ENoCs realizam um acordo de chaves para trocarem as chaves públicas. Cada sistema cria uma chave secreta compartilhada a partir da chave pública recebida e de sua chave privada. Esta chave secreta compartilhada é inserida como parâmetro para a *Key Derivate Function* (KDF) que retorna duas chaves simétricas concatenadas, uma para verificar a integridade e outra para cifrar e decifrar com um algoritmo de criptografia de chave simétrica. A integridade é realizada com uso do método *Message Authentication Code*, e a criptografia utiliza o padrão AES com chave de 128 bits. Os detalhes de cada passo, funções utilizadas e informações detalhadas podem ser acessadas no Apêndice B.4.

Após estabelecer a chave simétrica de sessão compartilhada, a ENoC *A* cifra seu conjunto de informações de identificação e envia ao sistema *B*. O Sistema *B* atualiza sua memória com as informações do sistema *A*. De mesma forma, o sistema *B* envia suas informações cifradas para o sistema *A*. A partir deste ponto a comunicação inter-chip pode ser trafegada com confidencialidade utilizando ECIES.

As premissas básicas para utilizar a ECIES são: os sistemas devem ser capazes de estabelecer a sessão e gerar suas chaves efêmeras e simétricas de sessão. Estes procedimentos devem ser realizados pelos *hubs* sem fio. Ele deve também ser capaz de realizar as operações de aritmética modular para a curva elíptica. Com as premissas atendidas, a comunicação inter-chip com ECIES acontece da seguinte forma:

1. O PE remetente cria o pacote se divide-os em *flits*;

2. Os *flits* são roteados na comunicação intra-chip até o *hub* sem fio local por meio de saltos entre os roteadores, de acordo com o algoritmo de roteamento utilizado;

3. O *hub* sem fio recupera a mensagem a ser enviada, e realiza os seguintes passos: a mensagem é cifrada com o algoritmo de chave simétrica e a chave de sessão de confidencialidade; a função *Medium Access Control* (MAC) gera um resumo da mensagem cifrada com a chave de sessão de integridade, este resumo é chamado de etiqueta (do inglês, *tag*), ele é utilizado pelo destinatário para verificar se a mensagem cifrada recebida é a mesma que a enviada;

4. O *hub* sem fio envia o criptograma contendo a mensagem cifrada, a etiqueta e a chave pública efêmera pelo canal sem fio. De fato, a sessão será válida enquanto as chaves efêmeras forem utilizadas e uma nova sessão pode ser estabelecida com a geração de novas chaves públicas e privadas e acordo de chaves;

5. O *hub* sem fio do sistema de destino recebe o pacote, verifica a integridade da mensagem com a função MAC, a mensagem cifrada e etiqueta recebidas, e a chave de sessão de integridade. Se a integridade for garantida, a mensagem cifrada é decifrada com uso do algoritmo de criptografia e a chave de sessão simétrica de confidencialidade. A mensagem original é recuperada. O *hub* sem fio remonta os dados em um novo pacote, divide-o em *flits* e encaminha a seu roteador;

6. Os *flits* são roteados pela malha com fio até alcançar o PE destinatário, seguindo um algoritmo de roteamento;

7. O PE destinatário recebe o pacote e recupera a mensagem transmitida.

A grande vantagem dessa abordagem é oferecer as características das chaves assimétricas com o desempenho de criptografia de chave simétrica. A desvantagem do ECIES é o tempo necessário para estabelecer as chaves de sessão compartilhadas, no entanto o seu desempenho pode superar esta questão. A relação será explorada na avaliação da segurança da ENoC.

7.2.4 Acordo de chaves

Os criptossistemas de chaves assimétricas podem ser alternativas viáveis para garantir a confidencialidade da informação sem a necessidade de uma chave secreta compartilhada por todos sistemas. Contudo, há sempre o problema de distribuição da chave pública de forma segura.

Cada *hub* pode espalhar sua chave pública no canal sem fio após receber um *beacon* ou mesmo informações de chaves públicas emitidas por outros sistemas. As chaves públicas recebidas podem ser armazenadas e gerenciadas dinamicamente pelo *hub* sem fio em uma memória reconfigurável. A chave pública deve ser associada aos endereços recebidos no *handshake* inicial, estes endereços são de identificação do *hub* e PEs de cada sistema ENoC (ver subseção 3.4). A principal vantagem dessa abordagem é sua simplicidade de implementação por não requerer dispositivos adicionais como uma unidade centralizadora, ou algoritmos que derivem essa chave dinamicamente causando impacto no desempenho. No entanto, esta abordagem é suscetível ao ataque do homem-do-meio (Chen et al. (2007); Conti et al. (2016)). Este ataque também é bem sucedido sob o protocolo de troca de chaves Diffie-Hellman (Diffie e Hellman (1976)), porém o acordo de chave baseado em curvas elípticas *Elliptic Curve Menezes-Qu-Vanstone* (ECMQV) ((Law et al. (2003))) oferece proteção contra ele.

A unidade central distribuidora e autenticadora das chaves públicas é uma alternativa para evitar o ataque de homem-do-meio. Neste cenário, todas as chaves públicas são armazenadas previamente e entregues por esta unidade, e ela também pode ser consultada para informar a veracidade da chave pública. Para considerar esta abordagem na ENoC deveria haver uma infraestrutura de comunicação formando uma rede local entre as ENoCs por meio de um ponto de acesso, este ponto de acesso agiria como a unidade central. Toda ENoC deveria registrar-se primeiramente neste ponto de acesso central a fim de informar e obter confiavelmente as chaves públicas. Esta abordagem é inviável e confronta com a proposta de expansibilidade na ENoC, pois as comunicações são dinâmicas, ponto-a-ponto e sem uma infraestrutura de conexão.

7.3 Avaliação da Segurança na ENoC

Esta seção mostra os parâmetros, métricas e ambiente de simulação, carga de trabalho e resultados do impacto da segurança na ENoC. A simulação consiste em 16 PEs divididos em quatro sistemas ENoC com malha de 2×2 cada.

Os parâmetros da comunicação na *ENoC* estão apresentados no Capítulo 3. As mesmas aplicações NPB da avaliação da ENoC foram consideradas para essa avaliação de impacto da segurança, os detalhes dessa carga de trabalho são encontradas no Capítulo 6.

O AES foi implementado usando *Cipher Block Chaining* (CBC) com chave de 128 bits. A ECC foi implementada usando a curva $secp256r1$ com chaves de 256 bits sob corpo finito primo, a semente utilizada para gerar a aleatoriedade foi a recomendada pela especificação da

curva. Foi também implementado o mapeamento dos pontos da curva elíptica a partir do método de Koblitz (Koblitz (1987)).

O ECIES usa a curva elíptica $secp256r1$ e 128 bits AES. A biblioteca *Multiprecision Integer and Rational Arithmetic Cryptographic Library* (MIRACL) (CertiVox (2014)) foi usada para implementar os três criptossistemas, testar e obter os resultados. Os resultados mostram o custo das operações de cifra e decifra para o AES e ECC. Para o ECIES foi considerado também o custo das gerações das chaves de sessão compartilhadas. O custo das trocas de chaves não foi considerado.

As Figuras 7.3, 7.4 e 7.5 mostram o custo da execução em milissegundos por bytes. A Figura 7.3 mostra o custo da cifra enquanto que a Figura 7.4 mostra o custo da decifra. Como esperado, o AES possui o melhor desempenho em todos os cenários. Para mensagens menores que 16 bytes, no qual é o tamanho do bloco AES, o preenchimento é realizado em conformidade.

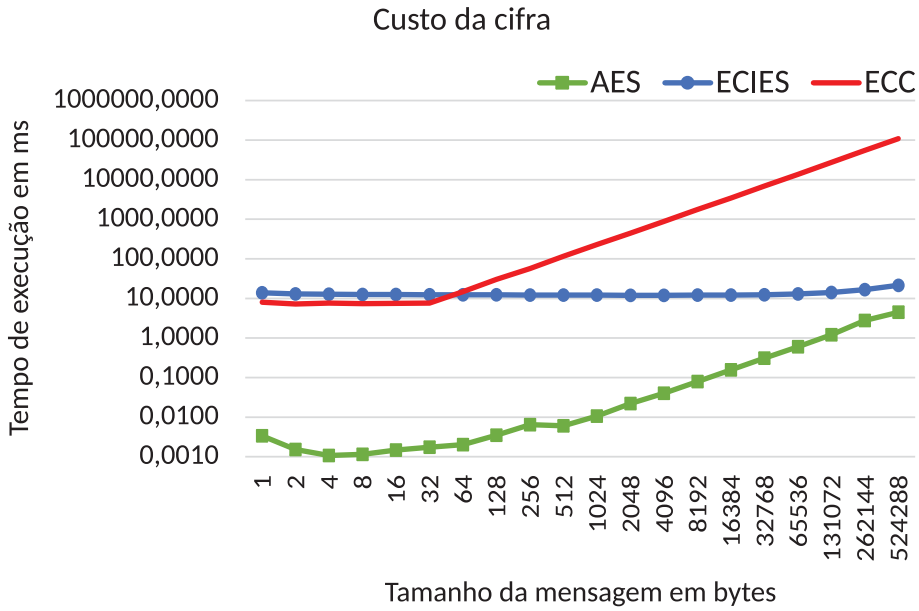


Figura 7.3: Comparação do tempo de execução para cifrar.

Comparando ECIES com AES, o ECIES tem o custo adicional do acordo de chaves no início de cada sessão de comunicação. O ECC tem um ligeiro ganho em relação ao ECIES para mensagens menores que a ordem da curva elíptica.

Para a curva escolhida ($secp256r1$) o seu limite é 32 bytes. Isto é devido ao acordo de chave de sessão do ECIES. Mensagens maiores que 32 bytes devem ser divididas em mensagens de 32 bytes e processadas uma a uma, resultando em um tempo de execução linear.

Os resultados apresentados mostram a sobrecarga de cada solução de segurança. O gerenciador do sistema deve balancear a força da segurança com a sobrecarga e escolher a melhor opção para o cenário considerado.

A Figura 7.5 compara a média do custo do AES e ECIES sob aplicações NPB. A base de comparação é a troca de mensagens inseguras em texto plano. O tempo de execução do AES e ECIES foi normalizado pela *baseline*. Para suportar a comparação, a Tabela 7.2 apresenta o número de mensagens para as aplicações agrupadas por tamanho.

É possível perceber que AES e ECIES tem aproximadamente a mesma sobrecarga para as aplicações FT, IS, e BT, enquanto eles tem uma alta sobrecarga diferente para MG, LU e CG. Esta diferença reflete o tamanho das mensagens de cada aplicação (veja a Tabela 7.2).

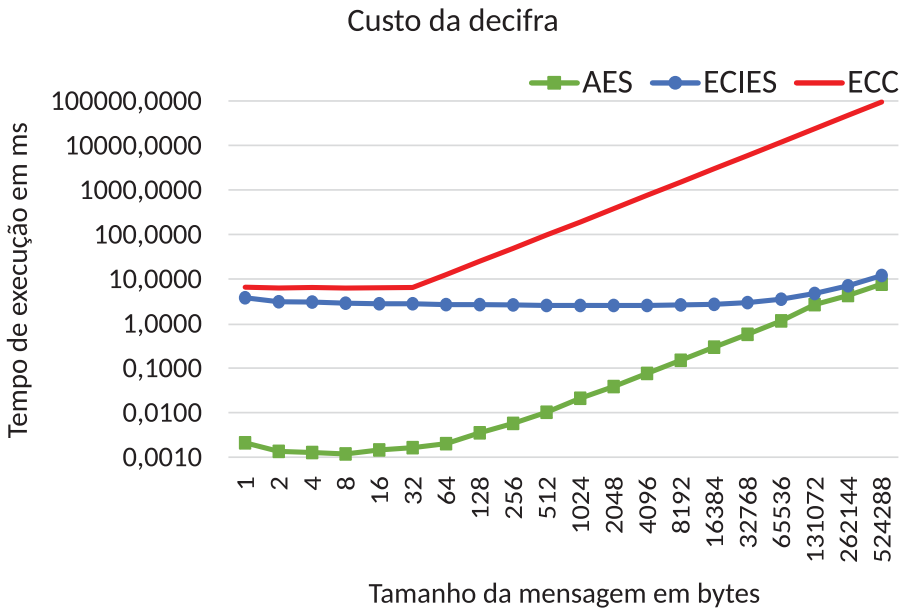


Figura 7.4: Comparação do tempo de execução para decifrar.

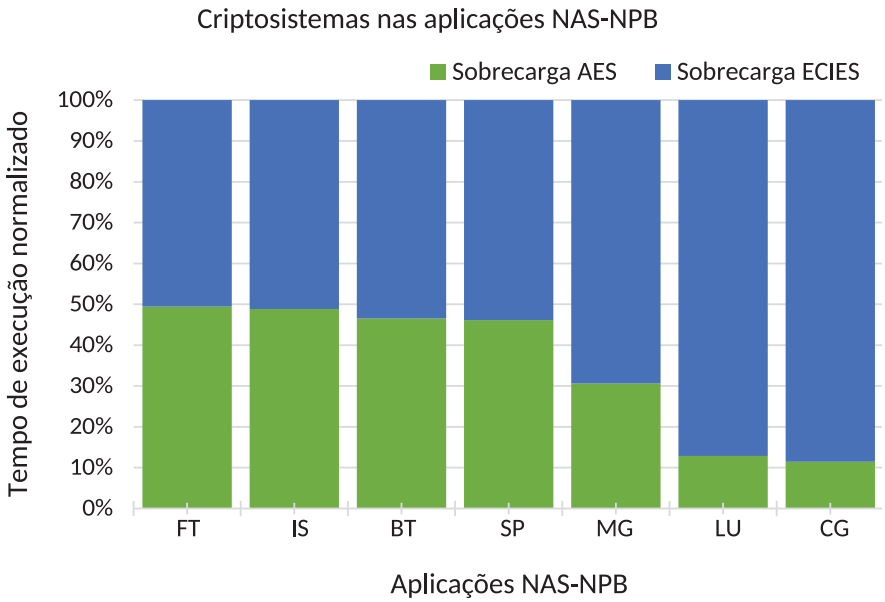


Figura 7.5: Impacto da segurança nas aplicações NAS-NPB.

7.4 Soluções relacionadas

As abordagens que enfrentam problema de confidencialidade na NoC são voltadas para a comunicação intra-chip. Gebotys e Gebotys (2003) apresentam um *framework* para NoCs. Ele trabalha como um envólucro de segurança para cada PE e um núcleo chamado de *key-keeper*. Ele usa uma chave simétrica armazenada em memória não volátil no envólucro de segurança para estabelecer a comunicação cifrada com o *key-keeper*. O impacto dessa segurança não é apresentado. Adaptar essa solução para a comunicação inter-chip da ENoC é inviável pois não há infraestrutura para ter um *key-keeper*, uma vez que a comunicação é ponto-a-ponto com sistemas que não se conhecem previamente.

Tabela 7.2: Montante das mensagens NPB pelo tamanho.

Aplicação	<=16B	>16B e <=130KB	>130KB
FT	1290	0	1920
IS	2925	2640	2640
BT	1455	77760	0
SP	975	154560	0
MG	23855	9984	0
LU	3111	114458	0
CG	27391	18304	0

Sajeesh e Kapoor (2011) e Kapoor et al. (2013) propõem *frameworks* para garantir a segurança na comunicação da NoC contra a extração da informação secreta. Eles usam o *Galois Hash* (GHASH) e 128 bits AES implementados na interface de rede, e ambos dividem a NoC em PEs seguros ou inseguros. A solução de Sajeesh e Kapoor (2011) é restrita para comunicação entre PEs não seguros. Ele foi testado considerando até 100 pacotes, e o tamanho do pacote não foi informado. Apesar dos autores concluírem que não há sobrecarga significante para cada transmissão de pacote, o padrão de comunicação é muito pequeno para considerar essa conclusão válida. O principal objetivo de Kapoor et al. (2013) é proteger um grupo seguro contra acessos não autorizados do grupo inseguro. Os PEs seguros podem comunicar entre si usando uma chave simétrica permanente, enquanto a comunicação entre os não seguros acontece em texto plano. A comunicação entre um seguro e não seguro é realizada usando chave de sessão. A divisão entre sistemas seguros e não seguros não é possível na ENoC, pois não há grupos na comunicação inter-chip.

Ancajas et al. (2014a) enfrentam o ataque de *snooping* na NoC, esse trabalho consiste em três camadas: *Data Scrambling*, para criar um rígida barreira para reduzir a eficiência deste ataque; Certificação de pacotes, para evitar transmissão de flits com certificação inválida; e *Node Obsfuscation*, para esconder os nós da comunicação pela introdução de ruído. Na ENoC a inserção de ruídos massivos pode afetar na comunicação sem fio. Sepúlveda et al. (2015) provêm o serviço de confidencialidade baseado em proteção de zonas disruptivas. Esse serviço usa trocas de chaves *Diffie-Hellman* para criar grupos secretos. A informação é codificada com uso do operador lógico *XOR* e a própria chave do grupo. A criação e manutenção de grupos em cenários dinâmicos como da ENoC podem resultar em complicadas e custosas operações adicionais.

7.5 Considerações sobre o capítulo

Muitos fabricantes são confiáveis e tem políticas eficientes e completas de segurança em sua linha de produção. Elementos de tais fabricantes podem ser considerados seguros para confidencialidade, não requerendo nenhuma medida de segurança adicional. Entretanto, o conceito de expansibilidade da ENoC traz a vulnerabilidade da comunicação inter-chip, e confidencialidade da comunicação deve ser garantida com medidas adicionais de segurança.

Este capítulo apresentou as propostas de confidencialidade para a ENoC. O foco da segurança é garantir que a comunicação inter-chip ocorra de forma que um terceiro sistema atacante não possa obtê-la. Este sistema não necessariamente é uma ENoC, bastando apenas conhecer a tecnologia de comunicação e isso amplia os dispositivos atacantes.

Esta é a primeira solução para garantir a confidencialidade na comunicação inter-chip. A segurança proposta é baseada em criptografia. Foram apresentadas três abordagens de segurança para ENoC, com AES, ECC e ECIES. O desempenho da proposta de segurança foi testado em cenário real. A escolha do criptossistema é uma decisão de projeto, e o projetista do sistema deve considerar as vantagens, desvantagens e desempenho das diferentes soluções para encontrar a opção correta.

Dessa forma, concluímos que AES, ECC e ECIES são opções viáveis para garantir a confidencialidade na comunicação inter-chip. Vale ressaltar que nenhum criptossistema é capaz de prover segurança contra um HT, pois o *hub*, roteador e PEs são considerados seguros, e o comportamento de insegurança com esses elementos comprometidos não foram enfrentados nesse trabalho. Como trabalho futuro, nós consideramos uma solução mesclada para garantir a confidencialidade nas comunicações intra e inter-chip. Por exemplo, é possível usar AES na comunicação intra-chip e ECIES para comunicação inter-chip; Também é possível utilizar a mesma criptografia assimétrica para proteção intra e inter-chip, para isso o *Hub* sem fio pode atuar como uma unidade centraliza que fornece de forma segura as chaves públicas locais e remotas passando a responsabilidade de cifrar e decifrar para cada PE e o *Hub* atuando apenas como um *Gateway* entre os sistemas, além de manter a integridade das chaves armazenadas; O *Rivest Shamir and Adleman* (RSA) também pode ser empregado de acordo com as características de implementação do projeto ENoC.

Além disso, é possível pesquisar outra forma de evitar o ataque de homem do meio no acordo de chaves sem a necessidade de uma unidade centralizadora. Uma alternativa que pode ser viável para ENoC é utilizar a criptografia baseada em identidade (Shamir (1985); Choie et al. (2005); Ryu et al. (2004)). Neste criptossistema cada sistema ENoC deve ter uma identidade, ela pode ser formada pelo conjunto de endereços do *hub* sem fio e dos PEs. A identidade é usada para derivar a chave pública, a chave privada é geralmente provida por uma entidade confiável, chamada de gerador de chaves privadas (*Private Key Generator* (PKG)). Como já informado, uma unidade central confiável é inviável para ENoC, uma vez que a comunicação acontece sem uma infraestrutura de rede por simples aproximação dos sistemas, não há uma unidade centralizadora.

Boneh e Franklin (2001) apresentam o uso distribuído da PKG em esquemas de criptografia, retirando a necessidade de centralização, e atribuindo a computação da chave privada a um subconjunto dos sistemas comunicantes. A partir desse trabalho, Silva e Albini (2013) apresentam uma solução chamada iFuso para *Mobile Ad-hoc Network*s (MANETs), uma solução descentralizada em que os nós gerenciam suas próprias chaves. Esta abordagem pode ser estendida como possível solução para a ENoC.

8 Conclusões

As NoCs permitem uma comunicação escalar e paralela para SoCs, interligando os PEs do sistema com desempenho superior ao tradicional barramento, o que as tornam atrativas para a computação de alto desempenho. O custo adicional de silício, maior latência e gasto energético são superados por suas vantagens e possibilidade de integração de cada vez mais PEs Zeferino (2003). Essa comunicação física pode ainda ser ampliada de ligações metálicas para também a rede sem fio, porém sua aplicação era voltada para diminuir os saltos entre PEs distantes. Embora existam tentativas de interligações de sistemas em chip, a possibilidade de expandir uma NoC interconectando vários sistemas de maneira *on-the-fly* com a visão de um sistema único foi introduzida pela ENoC.

A principal contribuição desta tese de doutorado é a criação da arquitetura ENoC, a primeira NoC expansível *on-the-fly*. A ENoC permite identificar e conectar novos sistemas sem nenhuma modificação ou adaptação física, e atribuir a eles uma visão lógica de único sistema expandindo ou mesmo contraindo seus recursos. Esta comunicação só é viável com o uso de um protocolo com baixa sobrecarga, sem a pilha de protocolos convencionais dos sistemas de comunicação, criado tão simples quanto possível. Ela é baseada em comunicação sem fio de 60 GHz usando o esquema de modulação digital OFDM com 64 QAM.

O protocolo na ENoC foi concebido tão simples quanto possível, com foco no desempenho do sistema. No entanto, algumas características ficaram ausentes no protocolo de comunicação, presente em sistemas de comunicação como a Ethernet e o Infiniband. Considerando estes dois sistemas de interconexão, a ENoC não possui identificador do tipo do pacote, verificação de integridade do pacote como *Frame check sequence* (FCS) ou *Cyclic Redundancy Check* (CRC), bits reservados para informações adicionais das aplicações, contagem de pulos entre os roteadores e número sequencial do pacote. A garantia da qualidade de serviço na ENoC foi atribuída à aplicação para poupar a pilha do protocolo; a verificação de integridade e retransmissão não foi implementada neste momento, uma vez que neste primeiro passo, a ENoC foi testada para trabalhar em um ambiente controlado como um ambiente de servidores de rack; reserva de informações para aplicações, contagem de pulos e sequencia de pacotes não são necessários na ENoC pois o objetivo é apenas distribuir o processamento com menor sobrecarga possível e os pacotes são enviados *flit* a *flit* sequencialmente entre origem e destino, além disso, o uso de barreiras sincronizantes auxiliam na chegada correta das informações para cada processo.

A comunicação sem fio entre sistemas é naturalmente insegura, pois ao considerar sistemas distintos comunicando por canais sem fio em ambiente externos ao SoCs, a informação trafegada pode ser extraída indevidamente. Em busca de prover a confidencialidade na comunicação inter-chip da ENoC, foram estudados os sistemas de criptografia AES, ECC e ECIES, também foram identificados os passos para sua implementação em uma NoC.

Com este estudo, foi possível concluir que o AES possui o melhor desempenho com adequado nível de segurança comparado ao ECC e ECIES, porém compartilha uma única chave secreta; A ECC remove a restrição de chave compartilhada do AES com o uso de chaves pública e privada porém com maior custo computacional. A ECIES combina a ECC com o AES removendo

a necessidade de única chave secreta compartilhada e mitiga o custo computacional de usar somente a ECC, porém ela usa uma chave de sessão que gera uma certa sobrecarga em cada sessão de comunicação. De fato, a escolha de qual abordagem de segurança a ser aplicada é uma decisão do projetista do sistema, levando em consideração suas necessidades.

Durante a realização desta pesquisa foram produzidas as publicações científicas:

1. A ENoC foi publicada na forma de artigo na revista científica *"Advances In Electrical and Computer Engineering"* (AECE). Nesta publicação a arquitetura ENoC foi apresentada à comunidade científica juntamente com uma avaliação do seu desempenho.

   *Ivan Luiz Pedroso Pires, Marco Antonio Zanata Alves, and Luiz Carlos Pessoa Albini. "**Expansible Network-on-Chip Architecture.**"Advances in Electrical and Computer Engineering (AECE), vol.18, no.2, pp.61-68, 2018, doi:10.4316/AECE.2018.02008*

2. Para validar o novo conceito de expansibilidade da ENoC nós estendemos o simulador Noxim para suportar a carga de trabalhos geradas em aplicações externas ao simulador e um controle de fluxo para aproximar o tempo de simulação ao tempo de execução necessária para realizar o processamento relativos as mensagens recebidas.

   *Ivan Luiz Pedroso Pires, Marco Antonio Zanata Alves, and Luiz Carlos Pessoa Albini. "**Trace-driven extension for Noxim simulator.**"VII Brazilian Symposium on Computing Systems Engineering (SBESC), IEEE, 2017, pp. 102-108. doi: 10.1109/SBESC.2017.20.*

3. A publicação do SBESC foi estendida e submetida ao periódico *Design Automation for Embedded Systems*. Nesta extensão foi adicionado o suporte às primitivas MPI e o tempo de processamento utilizado por cada primitiva ao simulador Noxim, o tempo de processamento é obtido na geração do arquivo de traço a ser informado como entrada para o simulador. As extensões do Noxim podem ser obtidas no endereço: http://gihub.com/ivanlppires/improved_noxim.

Os principais pontos para os próximos trabalhos que estenderão a ENoC são:

- Estudar e propor adaptações para um sistema operacional que utiliza e gerencia as características da ENoC, e que trabalhe como intermediador entre a arquitetura e os softwares aplicativos, de tal forma que a expansão e contração da ENoC seja invisível ao usuário;
- Analisar o impacto da ENoC em outros padrões de comunicação e relacionar o desempenho com o uso das primitivas MPI;
- Mesclar as soluções de criptografia para melhorar o desempenho no sistema, com uso da cifra simétrica para comunicação intra-chip e assimétrica para a inter-chip;
- Ampliar a segurança da ENoC para prover a disponibilidade, integridade e autenticidade;

Referências

Abad, P., Prieto, P., Menezo, L. G., Colaso, A., Puente, V. e Gregorio, J. A. (2012). TOPAZ: An Open-Source Interconnection Network Simulator for Chip Multiprocessors and Supercomputers. Em *IEEE/ACM Sixth International Symposium on Networks-on-Chip*, páginas 99–106.

Adamov, A., Saprykin, A., Melnik, D. e Lukashenko, O. (2009). The problem of Hardware Trojans detection in System-on-Chip. Em *The Experience of Designing and Application of CAD Systems in Microelectronics*, páginas 178–179.

Agarwal, N., Krishna, T., Peh, L. S. e Jha, N. K. (2009). GARNET: A detailed on-chip network model inside a full-system simulator. Em *International Symposium on Performance Analysis of Systems and Software*, páginas 33–42.

Alali, F., Mizero, F., Veeraraghavan, M. e Dennis, J. M. (2017). A measurement study of congestion in an InfiniBand network. Em *2017 Network Traffic Measurement and Analysis Conference (TMA)*, páginas 1–9.

Amiri, R. e Elkeelany, O. (2013). Concurrent reconfigurable architecture for mapping and encrypting a message in Elliptic Curve Cryptography. Em *Proceedings of IEEE Southeastcon*, páginas 1–6.

Ancajas, D. M., Chakraborty, K. e Roy, S. (2014a). Fort-NoCs: Mitigating the Threat of a Compromised NoC. Em *Proceedings of the 51st Annual Design Automation Conference*, páginas 1–6.

Ancajas, D. M., Chakraborty, K. e Roy, S. (2014b). Fort-NoCs: Mitigating the threat of a compromised NoC. Em *ACM/EDAC/IEEE Design Automation Conference (DAC)*, páginas 1–6.

Ascia, G., Catania, V., Palesi, M. e Patti, D. (2008). Implementation and Analysis of a New Selection Strategy for Adaptive Routing in Networks-on-Chip. *IEEE Transactions on Computers*, 57(6):809–820.

Axelsson, O. (1972). A generalized SSOR method. *BIT Numerical Mathematics*, 12(4):443–467.

Bailey, D., Harris, T., Saphir, W., van der Wijngaart, R., Woo, A., e Yarrow, M. (1995). The NAS parallel benchmarks 2.0. Relatório técnico, NASA.

Bailey, D. H., Dagum, L., Barszcz, E. e Simon, H. D. (1992). NAS parallel benchmark results. páginas 386–393.

Barak, A. e La'adan, O. (1998). The MOSIX multicomputer operating system for high performance cluster computing. *Future Generation Computer Systems*, 13(4):361–372.

Ben Abdallah, A. (2013). *3D Network-on-Chip*, páginas 89–125. Atlantis Press, Paris.

Ben-Itzhak, Y., Zahavi, E., Cidon, I. e Kolodny, A. (2011). NoCs Simulation Framework for OMNeT++. Em *International Symposium on Networks-on-Chip*, páginas 265–266.

Ben-Itzhak, Y., Zahavi, E., Cidon, I. e Kolodny, A. (2012). HNOCS: Modular open-source simulator for Heterogeneous NoCs. Em *International Conference on Embedded Computer Systems*, páginas 51–57.

Benini, L. e Micheli, G. D. (2002). Networks on chips: a new SoC paradigm. *Computer*, 35(1):70–78.

Bertozzi, D. e Benini, L. (2004). Xpipes: a network-on-chip architecture for gigascale systems-on-chip. *IEEE Circuits and Systems Magazine*, 4(2):18–31.

Binkert, N., Beckmann, B., Black, G., Reinhardt, S. K., Saidi, A., Basu, A., Hestness, J., Hower, D. R., Krishna, T., Sardashti, S., Sen, R., Sewell, K., Shoaib, M., Vaish, N., Hill, M. D. e Wood, D. A. (2011). The Gem5 Simulator. *SIGARCH Computer Architecture News*, 39(2):1–7.

Bjerregaard, T. e Mahadevan, S. (2006). A Survey of Research and Practices of Network-on-chip. *ACM Computing Surveys*, 38(1).

Blaise, G., Daihyun, L., Dwaine, C., van Dijk Marten e Srinivas, D. (2004). Identification and authentication of integrated circuits. *Concurrency and Computation: Practice and Experience*, 16(11):1077–1098.

Bolotin, E., Cidon, I., Ginosar, R. e Kolodny, A. (2004a). QNoC: QoS architecture and design process for network on chip. *Journal of Systems Architecture*, 50(2):105–128.

Bolotin, E., Morgenshtein, A., Cidon, I., Ginosar, R. e Kolodny, A. (2004b). Automatic hardware-efficient SoC integration by QoS network on chip. Em *IEEE International Conference on Electronics, Circuits and Systems*, páginas 479–482.

Boneh, D. e Franklin, M. (2001). Identity-Based Encryption from the Weil Pairing. Em Kilian, J., editor, *Advances in Cryptology*, páginas 213–229. Springer Berlin Heidelberg.

Boraten, T., DiTomaso, D. e Kodi, A. K. (2016). Secure model checkers for Network-on-Chip (NoC) architectures. Em *International Great Lakes Symposium on VLSI*, páginas 45–50.

Carloni, L. P., Pande, P. e Xie, Y. (2009). Networks-on-chip in Emerging Interconnect Paradigms: Advantages and Challenges. Em *Proceedings of the 2009 3rd ACM/IEEE International Symposium on Networks-on-Chip*, páginas 93–102. IEEE Computer Society.

Catania, V., Holsmark, R., Kumar, S. e Palesi, M. (2006). A methodology for design of application specific deadlock-free routing algorithms for NoC systems. Em *Proceedings of the 4th International Conference on Hardware/Software Codesign and System Synthesis*, páginas 142–147.

Catania, V., Mineo, A., Monteleone, S., Palesi, M. e Patti, D. (2015). Noxim: An Open, Extensible and Cycle-Accurate Network-on-Chip Simulator. Em *International Conference on Application-specific Systems, Architectures and Processors*, páginas 162–163.

Catania, V., Mineo, A., Monteleone, S., Palesi, M. e Patti, D. (2016). Cycle-Accurate Network on Chip Simulation with Noxim. *ACM Transaction on Modeling and Computer Simulation*, 27(1):1–25.

CERTICOM (2000). SEC 2: Recommended Elliptic Curve Domain Parameters. `http://www.secg.org/SEC2-Ver-1.0.pdf`. Acessado em 12/11/2017.

CertiVox (2014). Multiprecision Integer and Rational Arithmetic Cryptographic Library (MIRACL). `http://github.com/miracl`. Acessado em 20/02/2018.

Chang, M. F., Cong, J., Kaplan, A., Naik, M., Reinman, G., Socher, E. e Tam, S. W. (2008). CMP Network-on-Chip Overlaid with Multi-Band RF-Interconnect. *International Symposium on High Performance Computer Architecture*, páginas 191–202.

Chen, J., Gillard, P. e Li, C. (2012). Performance evaluation of three Network-on-Chip (NoC) architectures. Em *International Conference on Communications in China*, páginas 91–96.

Chen, Z., Guo, S., Zheng, K. e Yang, Y. (2007). *Modeling of Man-in-the-Middle Attack in the Wireless Networks*, páginas 2255–2258.

Chien, S. Y., Chan, W. K., Tseng, Y. H., Lee, C. H., Somayazulu, V. S. e Chen, Y. K. (2015). Distributed computing in IoT: System-on-a-chip for smart cameras as an example. Em *Asia and South Pacific Design Automation Conference*, páginas 130–135.

Chiu, G.-M. (2000). The odd-even turn model for adaptive routing. *IEEE Transactions on Parallel and Distributed Systems*, 11(7):729–738.

Choie, Y. J., Jeong, E. e Lee, E. (2005). Efficient identity-based authenticated key agreement protocol from pairings. *Applied Mathematics and Computation*, 162(1):179–188.

Conti, F., Schilling, R., Schiavone, P. D., Pullini, A., Rossi, D., Gürkaynak, F. K., Muehlberghuber, M., Gautschi, M., Loi, I., Haugou, G., Mangard, S. e Benini, L. (2017). An IoT Endpoint System-on-Chip for Secure and Energy-Efficient Near-Sensor Analytics. *IEEE Transactions on Circuits and Systems I*, 64(9):2481–2494.

Conti, M., Dragoni, N. e Lesyk, V. (2016). A Survey of Man In The Middle Attacks. *IEEE Communications Surveys Tutorials*, 18(3):2027–2051.

Daemen, J. e Rijmen, V. (2002). *The design of Rijndael: AES - The Advanced Encryption Standard*. Springer-Verlag Berlin Heidelberg.

Dall'Osso, M., Biccari, G., Giovannini, L., Bertozzi, D. e Benini, L. (2012). Xpipes: A latency insensitive parameterized network-on-chip architecture for multi-processor SoCs. Em *International Conference on Computer Design (ICCD)*, páginas 45–48.

Dally, William J.and Seitz, C. L. (1986). The torus routing chip. *Distributed Computing*, 1(4):187–196.

Dally, W. e Towles, B. (2003). *Principles and Practices of Interconnection Networks*. Morgan Kaufmann Publishers Inc.

Dally, W. J. e Towles, B. (2001). Route packets, not wires: on-chip interconnection networks. Em *Proceedings of the 38th Design Automation Conference*, páginas 684–689.

Daniels, R. C., Murdock, J. N., Rappaport, T. S. e Heath, R. W. (2010). 60 GHz Wireless: Up Close and Personal. *IEEE Microwave Magazine*, 11(7):44–50.

Deb, S., Ganguly, A., Pande, P. P., Belzer, B. e Heo, D. (2012). Wireless NoC as Interconnection Backbone for Multicore Chips: Promises and Challenges. *Journal on Emerging and Selected Topics in Circuits and Systems*, 2(2):228–239.

Diener, M., Cruz, E. H. M., Pilla, L. L., Dupros, F. e Navaux, P. O. A. (2015). Characterizing Communication and Page Usage of Parallel Applications for Thread and Data Mapping. *Performance Evaluation*, 88-89.

Diffie, W. e Hellman, M. (1976). New directions in cryptography. *Information Theory, IEEE Transactions on*, 22(6):644–654.

Diguet, J.-P., Evain, S., Vaslin, R., Gogniat, G. e Juin, E. (2007). NOC-centric Security of Reconfigurable SoC. Em *First International Symposium on Networks-on-Chip*, páginas 223–232.

DiTomaso, D., Kodi, A., Matolak, D., Kaya, S., Laha, S. e Rayess, W. (2015). A-WiNoC: Adaptive Wireless Network-on-Chip Architecture for Chip Multiprocessors. *Transactions on Parallel and Distributed Systems*, 26(12):3289–3302.

Ebrahimi, M., Daneshtalab, M., Liljeberg, P., Plosila, J. e Tenhunen, H. (2012). CATRA-congestion aware trapezoid-based routing algorithm for on-chip networks. Em *Design, Automation Test in Europe Conference Exhibition (DATE)*, páginas 320–325.

Evain, S. e Diguet, J. P. (2005). From NoC security analysis to design solutions. Em *IEEE Workshop on Signal Processing Systems Design and Implementation*, páginas 166–171.

Floyd, B. A., Hung, C. e O, K. K. (2002). Intra-Chip Wireless Interconnect for Clock Distribution Implemented with Integrated Antennas, Receivers, and Transmitters. *Journal of Solid-State Circuits*, 37(5):543–552.

Freitas, H. C., Navaux, P. O. e Santos, T. G. (2008). NOC architecture design for multi-cluster chips. Em *International Conference on Field Programmable Logic and Applications*, páginas 53–58.

Freitas, H. C., Schnorr, L. M., Alves, M. A. Z. e Navaux, P. O. A. (2010). Impact of Parallel Workloads on NoC Architecture Design. Em *Euromicro Conference on Parallel, Distributed and Network-based Processing*, páginas 551–555.

Frey, J. e Yu, Q. (2015). Exploiting state obfuscation to detect hardware trojans in NoC network interfaces. Em *International Midwest Symposium on Circuits and Systems*, páginas 1–4.

Frey, J. e Yu, Q. (2017). A hardened network-on-chip design using runtime hardware trojan mitigation methods. *Integration, the VLSI Journal*, 56:15–31.

Gabis, A. B., Bomel, P. e Sevaux, M. (2018). Application-aware Multi-Objective Routing based on Genetic Algorithm for 2D Network-on-Chip. *Microprocessors and Microsystems*, 61:135–153.

Ganguly, A., Chang, K., Deb, S., Pande, P. P., Belzer, B. e Teuscher, C. (2011). Scalable Hybrid Wireless Network-on-Chip Architectures for Multicore Systems. *IEEE Transactions on Computers*, 60(10):1485–1502.

Ganguly, A., Mansoor, N., Shamim, M. S., Ahmed, M. M., Narde, R. S., Vashist, A. e Venkataraman, J. (2017). Intra-chip Wireless Interconnect: The Road Ahead. Em *Proceedings of the 10th International Workshop on Network on Chip Architectures*, páginas 1–6.

Gebotys, C. e Gebotys, R. (2003). A framework for security on NoC technologies. Em *IEEE Computer Society Annual Symposium on VLSI*, páginas 113–117.

Goossens, K., Dielissen, J. e Radulescu, A. (2005). AEthereal network on chip: concepts, architectures, and implementations. *IEEE Design Test of Computers*, 22(5):414–421.

Gropp, W., Lusk, E. e Skjellum, A. (2017). *Using MPI: portable parallel programming with the message-passing interface*, volume 1. MIT Press, 3 edition.

Guerrier, P. e Greiner, A. (1999). A scalable architecture for system-on-chip interconnections. Em *Sophia Antipolis Forum on Micro Electronics*, páginas 90–93.

Gugnani, S., Lu, X. e Panda, D. K. (2016). Performance Characterization of Hadoop Workloads on SR-IOV-Enabled Virtualized InfiniBand Clusters. Em *International Conference on Big Data Computing Applications and Technologies*, páginas 36–45.

Gura, N., Patel, A., Wander, A., Eberle, H. e Shantz, S. (2004). Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs. Em Joye, M. e Quisquater, J.-J., editores, *Cryptographic Hardware and Embedded Systems*, volume 3156, páginas 119–132. Springer Berlin Heidelberg.

Halim, D. K., Ming, T. C., Song, N. M. e Hartono, D. (2017). Software-based turbo decoder implementation on low power multi-processor system-on-chip for Internet of Things. Em *International Conference on New Media Studies*, páginas 137–141.

Hankerson, D. e Menezes, A. (2011). *Elliptic Curve Discrete Logarithm Problem*, páginas 397–400. Springer US.

Hankerson, D., Menezes, A. J. e Vanstone, S. (2006). *Guide to elliptic curve cryptography*. Springer Science & Business Media.

Hansen, C. J. (2011). WiGiG: Multi-gigabit wireless communications in the 60 GHz band. *IEEE Wireless Communications*, 18(6):6–7.

Hemani, A., Jantsch, A., Kumar, S., Postula, A., Oberg, J., Millberg, M. e Lindqvist, D. (2000). Network on chip: An architecture for billion transistor era. Em *IEEE NorChip Conference*, volume 31.

Hestness, Joel and Grot, Boris and Keckler, Stephen W. (2010). Netrace: Dependency-driven Trace-based Network-on-chip Simulation. Em *International Workshop on Network on Chip Architectures*, páginas 31–36.

Hossain, H., Ahmed, M., Al-Nayeem, A., Islam, T. Z. e Akbar, M. M. (2007). Gpnocsim - A General Purpose Simulator for Network-On-Chip. Em *International Conference on Information and Communication Technology*, páginas 254–257.

Howard, J., Dighe, S., Vangal, S. R., Ruhl, G., Borkar, N., Jain, S., Erraguntla, V., Konow, M., Riepen, M., Gries, M., Droege, G., Lund-Larsen, T., Steibl, S., Borkar, S., De, V. K. e Wijngaart, R. V. D. (2010). A 48-Core IA-32 message-passing processor with DVF in 45nm CMOS. Em *International Solid-State Circuits Conference*, páginas 108–109.

Hu, J. e Marculescu, R. (2004). DyAD: Smart Routing for Networks-on-chip. Em *Proceedings of the 41st Annual Design Automation Conference*, páginas 260–263.

Huang, Y., Yin, W. Y. e Liu, Q. H. (2008). Performance Prediction of Carbon Nanotube Bundle Dipole Antennas. *IEEE Transactions on Nanotechnology*, 7(3):331–337.

IEEE (2017). IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture – Amendment 2: Local Medium Access Control (MAC) Address Usage. *IEEE Std 802c-2017 (Amendment to IEEE Std 802-2014 as amended by IEEE Std 802d-2017)*, páginas 1–26.

Infiniband (2016). InfiniBand Architecture Specification Volume 2. Relatório Técnico Release 1.3.1, Infiniband Trade Association.

Jain, L., Al-Hashimi, B. M., Gaur, M. S., Laxmi, V. e Narayanan, A. (2007). NIRGAM: a simulator for NoC interconnect routing and application modeling. Em *Design, Automation and Test in Europe Conference*, páginas 16–20.

Jain, R. K. (1991). *Art of Computer Systems Performance Analysis: Techniques for Experimental Design Measurements, Simulation and Modeling*. John Wiley.

Jiang, N., Balfour, J., Becker, D. U., Towles, B., Dally, W. J., Michelogiannakis, G. e Kim, J. (2013). A Detailed and Flexible Cycle-Accurate Network-on-Chip Simulator. Em *International Symposium on Performance Analysis of Systems and Software*, páginas 86–96.

Johnson, D., Menezes, A. e Vanstone, S. (2014). The Elliptic Curve Digital Signature Algorithm (ECDSA). *International Journal of Information Security*, 1(1):36–63.

Kapoor, H., Rao, G., Arshi, S. e Trivedi, G. (2013). A Security Framework for NoC Using Authenticated Encryption and Session Keys. *Circuits, Systems, and Signal Processing*, 32(6):2605–2622.

Kim, K., Yoon, H. e O, K. K. (2000). On-chip wireless interconnection with integrated antennas. Em *International Electron Devices Meeting*, páginas 485–488.

King, B. (2009). Mapping an Arbritrary Message to an Elliptic Curve When Defined over GF (2^n). *International Journal of Network Security*, 8(2):169–176.

Koblitz, N. (1987). Elliptic curve cryptosystems. *Mathematics of computation*, 48(177):203–209.

Krichene, H., Baklouti, M., Marque, P., Dekeyser, J. L. e Abid, M. (2016). SCAC-Net: Reconfigurable Interconnection Network in SCAC Massively Parallel SoC. Em *International Conferenbce on Parallel, Distributed, and Network-Based Processing*, páginas 759–762.

Kumar, S., Jantsch, A., Soininen, J. P., Forsell, M., Millberg, M., Oberg, J., Tiensyrja, K. e Hemani, A. (2002). A network on chip architecture and design methodology. Em *Proceedings of IEEE Computer Society Annual Symposium on VLSI*, páginas 105–112.

Kurimoto, Y., Fukutsuka, Y., Taniguchi, I. e Tomiyama, H. (2013). A hardware-software cosimulator for Network-on-Chip. Em *International SoC Design Conference*, páginas 172–175.

Laha, S., Kaya, S., Matolak, D. W., Rayess, W., DiTomaso, D. e Kodi, A. (2015). A New Frontier in Ultralow Power Wireless Links: Network-on-Chip and Chip-to-Chip Interconnects. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 34(2):186–198.

LAN/MAN Standards Committee (2016). IEEE Standard for Ethernet. *IEEE Standard 802.3*, páginas 1–4017.

Law, L., Menezes, A., Qu, M., Solinas, J. e Vanstone, S. (2003). An efficient protocol for authenticated key agreement. *Designs, Codes and Cryptography*, 28(2):119–134.

LeMay, M. e Gunter, C. A. (2015). *Network-on-Chip Firewall: Countering Defective and Malicious System-on-Chip Hardware*, páginas 404–426. Springer International Publishing.

Li, M., Zeng, Q.-A. e Jone, W.-B. (2006). DyXY: A Proximity Congestion-aware Deadlock-free Dynamic Routing Method for Network on Chip. Em *Proceedings of the 43rd Annual Design Automation Conference*, páginas 849–852.

Li, X., Duraisamy, K., Bogdan, P., Majumder, T. e Pande, P. P. (2016). Network-on-Chip-Enabled Multicore Platforms for Parallel Model Predictive Control. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 24(9):2837–2850.

Li, Y., Peng, S. e Chu, W. (2010). Metacube—a versatile family of interconnection networks for extremely large-scale supercomputers. *The Journal of Supercomputing*, 53(2):329–351.

Lin, J. J., Wu, H. T., Su, Y., Gao, L., Sugavanam, A., Brewer, J. E. e O, K. K. (2007). Communication Using Antennas Fabricated in Silicon Integrated Circuits. *IEEE Journal of Solid-State Circuits*, 42(8):1678–1687.

Lis, Mieszko and Shim, Keun Sup and Cho, Myong Hyon and Ren, Pengju and Khan, Omer and Devadas, Srinivas (2010). DARSIM: A Parallel Cycle-Level NoC Simulator. Em *Workshop on Modeling, Benchmarking and Simulation*.

Lu, Z., Thid, R., Millberg, M. e Jantsch, A. (2005). NNSE: Nostrum Network-on-Chip Simulation Environment. Em *Swedish System-on-Chip Conference*.

Mangano, D. e Urzi, I. (2015). System for Designing Network-on-Chip Interconnect Arrangements. US Patent 9,202,002.

Matyjas, J. D., Kumar, S. e Hu, F. (2015). *Wireless network performance enhancement via directional antennas: models, protocols, and systems*. CRC Press.

McCurley, K. S. (1990). The discrete logarithm problem. Em *Proceddings of Symposium in Applied Math*, volume 42, páginas 49–74.

Miller, V. S. (1986). Use of elliptic curves in cryptography. Em *Advances in Cryptology*, páginas 417–426. Springer.

Montgomery, P. L. (1994). A survey of modern integer factorization algorithms. *CWI quarterly*, 7(4):337–366.

Moscibroda, T. e Mutlu, O. (2009). A Case for Bufferless Routing in On-chip Networks. *SIGARCH Computer Architecture News*, 37(3):196–207.

MPICH (2018). High-performance portable mpi. `http://www.mpich.org/`. Acessado em 01/03/2018.

Mullender, S. J., van Rossum, G., Tananbaum, A. S., van Renesse, R. e van Staveren, H. (1990). Amoeba: a distributed operating system for the 1990s. *Computer*, 23(5):44–53.

Muthukuru, J. e Sathyanarayana, B. (2012). Fixed and variable Size Text Based Message Mapping Technique using ECC. *Global Journal of Computer Science and Technology*, páginas 5363–5375.

Nakajima, K., Kurebayashi, S., Fukutsuka, Y., Hieda, T., Taniguchi, I., Tomiyama, H. e Takada, H. (2013). Naxim: A Fast and Retargetable Network-on-Chip Simulator with QEMU and SystemC. *International Journal of Networking and Computing*, 3(2):217–227.

Ni, L. M. e McKinley, P. K. (1993). A survey of wormhole routing techniques in direct networks. *Computer*, 26(2):62–76.

NIST (1999). Recommended elliptic curves for federal government use. `http://csrc.nist.gov/encryption`. Acessado em 20/02/2017.

Pande, P. P., Ganguly, A., Chang, K. e Teuscher, C. (2009). Hybrid wireless network on chip: A new paradigm in multi-core design. Em *International Workshop on Network on Chip Architectures*, páginas 71–76.

Pereñíguez García, F. e Abellán, J. L. (2017). Secure Communications in Wireless Network-on-chips. Em *Proceedings of the 2Nd International Workshop on Advanced Interconnect Solutions and Technologies for Emerging Computing Systems*, páginas 27–32.

Pires, I. L. P., Alves, M. A. Z. e Albini, L. C. P. (2017). Trace-Driven Extension for Noxim Simulator. Em *VII Brazilian Symposium on Computing Systems Engineering*, páginas 102–108.

Radulescu, A., Dielissen, J., Goossens, K., Rijpkema, E. e Wielage, P. (2004). An efficient on-chip network interface offering guaranteed services, shared-memory abstraction, and flexible network configuration. Em *Proceedings Design, Automation and Test in Europe Conference and Exhibition*, volume 2, páginas 878–883.

Rajesh, J., Koushik, C. e Sanghamitra, R. (2018). *Hardware Trojan Attacks in SoC and NoC*, páginas 55–74. Springer International Publishing.

Ramakrishna, M., Kodati, V. K., Gratz, P. V. e Sprintson, A. (2016). GCA:Global Congestion Awareness for Load Balance in Networks-on-Chip. *IEEE Transactions on Parallel and Distributed Systems*, 27(7):2022–2035.

Rambo, E. A., Saidi, S. e Ernst, R. (2016). Providing formal latency guarantees for ARQ-based protocols in Networks-on-Chip. Em *Design, Automation Test in Europe Conference Exhibition (DATE)*, páginas 103–108.

Rantala, V., Lehtonen, T. e Plosila, J. (2006). Network on Chip Routing Algorithms. Relatório técnico, TUCS Technical Report nº 779.

Rappaport, T. S. (2002). *Wireless Communications: Principles and Practice*. Prentice Hall PTR, 2 edition.

Reinbrecht, C., Susin, A., Bossuet, L., Sigl, G. e Sepúlveda, J. (2017). Timing attack on NoC-based systems: Prime+Probe attack and NoC-based protection. *Microprocessors and Microsystems*, 52:556–565.

Reis, R., Soares, M., Lubaszewski e Jess, J. A. (2007). *Design of Systems on a Chip: Design and Test*. Springer US, 1 edition.

Ren, P., Lis, M., Cho, M. H., Shim, K. S., Fletcher, C. W., Khan, O., Zheng, N. e Devadas, S. (2012). HORNET: A Cycle-Level Multicore Simulator. *Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 31(6):890–903.

Rezaei, A., Daneshtalab, M., Safaei, F. e Zhao, D. (2016). Hierarchical approach for hybrid wireless Network-on-chip in many-core era. *Computers & Electrical Engineering*, 51:225–234.

Riley, G. F. e Henderson, T. R. (2010). *The ns-3 Network Simulator*, páginas 15–34. Springer Berlin Heidelberg.

Rivest, R. L., Shamir, A. e Adleman, L. (1978). A Method for Obtaining Digital Signatures and Public-key Cryptosystems. *Communications of the ACM*, 21(2):120–126.

Rohling, M., May, T., Bruninghaus, K. e Grunheid, R. (1999). Broad-band OFDM radio transmission for multimedia applications. *Proceedings of the IEEE*, 87(10):1778–1789.

Ryu, E.-K., Yoon, E.-J. e Yoo, K.-Y. (2004). An Efficient ID-Based Authenticated Key Agreement Protocol from Pairings. Em Mitrou, N., Kontovasilis, K., Rouskas, G. N., Iliadis, I. e Merakos, L., editores, *Networking 2004*, páginas 1458–1463. Springer Berlin Heidelberg.

Saini, S., Jin, H., Jespersen, D., Feng, H., Djomehri, J., Arasin, W., Hood, R., Mehrotra, P. e Biswas, R. (2013). An Early Performance Evaluation of Many Integrated Core Architecture Based SGI Rackable Computing System. Em *International Conference for High Performance Computing, Networking, Storage and Analysis*, páginas 1–12.

Sajeesh, K. e Kapoor, H. K. (2011). An Authenticated Encryption Based Security Framework for NoC Architectures. Em *International Symposium on Electronic System Design*, páginas 134–139.

Salmani, H. (2017). *Hardware Trojan Attacks and Countermeasures*, páginas 247–276. Springer International Publishing.

Sengupta, A. e Ray, U. K. (2016). Message mapping and reverse mapping in elliptic curve cryptosystem. *Security and Communication Networks*, 9(18):5363–5375.

Sepúlveda, J., Flórez, D. e Gogniat, G. (2015). Reconfigurable security architecture for disrupted protection zones in NoC-based MPSoCs. Em *International Symposium on Reconfigurable Communication-centric Systems-on-Chip*, páginas 1–8.

Shamim, M. S., Mansoor, N., Narde, R. S., Kothandapani, V., Ganguly, A. e Venkataraman, J. (2017). A Wireless Interconnection Framework for Seamless Inter and Intra-Chip Communication in Multichip Systems. *IEEE Transactions on Computers*, 66(3):389–402.

Shamim, M. S., Muralidharan, J. e Ganguly, A. (2015). An Interconnection Architecture for Seamless Inter and Intra-Chip Communication Using Wireless Links. Em *Proceedings of the 9th International Symposium on Networks-on-Chip*, páginas 1–8. ACM.

Shamir, A. (1985). Identity-Based Cryptosystems and Signature Schemes. Em Blakley, G. R. e Chaum, D., editores, *Advances in Cryptology*, páginas 47–53. Springer Berlin Heidelberg.

Silva, E. e Albini, L. C. P. (2013). Towards a fully self-organized identity-based key management system for MANETs. Em *International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, páginas 717–723.

Sinha, V., Doucet, F., Siska, C., Gupta, R., Liao, S. e Ghosh, A. (2000). YAML: A Tool for Hardware Design Visualization and Capture. páginas 9–14.

Stallings, W. (2002). *Cryptography and Network Security: Principles and Practice*. Pearson Education, 3 edition.

Swaminathan, K., Thakyal, D., Nambiar, S. G., Lakshminarayanan, G. e Ko, S.-B. (2014). Enhanced Noxim simulator for performance evaluation of network on chip topologies. Em *Recent Advances in Engineering and Computational Sciences*, páginas 1–5.

Tewksbury, S., Uppuluri, M. e Hornak, L. A. (1992). Interconnections/micro-networks for integrated microelectronics. Em *Global Telecommunications Conference*, páginas 180–186.

Trappe, W. e Washington, L. C. (2005). *Introduction to Cryptography with Coding Theory*. Prentice-Hall, Inc.

Valencia, P., Muller, E. e Wang, N. (2017). ZigZag: An efficient deterministic Network-on-chip routing algorithm design. Em *2017 8th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, páginas 1–5.

VanderWijngaart, R. F. e Haopiang, J. (2003). NAS Parallel Benchmarks, Multi-Zone Versions. Relatório técnico.

Walker, B., Popek, G., English, R., Kline, C. e Thiel, G. (1983). The LOCUS Distributed Operating System. *SIGOPS Operating Systems Review*, 17(5):49–70.

Wang, C., Hu, W. H. e Bagherzadeh, N. (2011). A Wireless Network-on-Chip Design for Multicore Platforms. Em *International Euromicro Conference on Parallel, Distributed and Network-Based Processing*, páginas 409–416.

Wiener, M. (1990). Cryptanalysis of short RSA secret exponents. *IEEE Transactions on Information Theory*, 36(3):553–558.

WiFi Alliance (2014). Wi-Fi Alliance WiGig Wireless Bus Extension Technical Specification. `http://www.wi-fi.org`. Acessado em 10/12/2017.

WiFi Alliance (2016). WiFi Certified WiGig: Wi-Fi Expands to 60 GHz. Relatório técnico, Wi-Fi Alliance. Acessado em 10/03/2018.

Wilton, S. J. E. e Jouppi, N. P. (1996). CACTI: an enhanced cache access and cycle time model. *Journal of Solid-State Circuits*, 31(5):677–688.

Wireless HD Consortium (2010). Wireless HD Specification Version 1.1 Overview. `http://www.wirelesshd.org`. Acessado em 20/03/2017.

Ye, T. T., Benini, L. e Micheli, G. D. (2004). Packetization and routing analysis of on-chip multiprocessor networks. *Journal of Systems Architecture*, 50(2):81–104. Special issue on networks on chip.

Ye, Y., Xu, J., Wu, X., Zhang, W., Liu, W. e Nikdast, M. (2012). A Torus-Based Hierarchical Optical-Electronic Network-on-Chip for Multiprocessor System-on-Chip. *Emerging Technologies in Computing Systems*, 8(1):1–26.

Yu, Q. e Frey, J. (2013). Exploiting error control approaches for Hardware Trojans on Network-on-Chip links. Em *IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFTS)*, páginas 266–271.

Yu, X., Rashtian, H., Mirabbasi, S., Pande, P. P. e Heo, D. (2015). An 18.7-Gb/s 60-GHz OOK Demodulator in 65-nm CMOS for Wireless Network-on-Chip. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 62(3):799–806.

Zeferino, C. A. (2003). *Redes em chip: Arquiteturas e Modelos para Avaliação de Área e Desempenho*. Tese de doutorado, Universidade Federal do Rio Grande do Sul (UFRGS).

Zeferino, C. A. e Susin, A. A. (2003). SoCIN: a Parametric and Scalable Network-on-Chip. Em *Symposium on Integrated Circuits and Systems Design*, páginas 169–174.

Zelst, A. V. e Schenk, T. C. W. (2004). Implementation of a MIMO OFDM-based wireless LAN system. *IEEE Transactions on Signal Processing*, 52(2):483–494.

Zhang, Z., Greiner, A. e Taktak, S. (2008). A Reconfigurable Routing Algorithm for a Fault-tolerant 2D-Mesh Network-on-Chip. Em *Proceedings of the 45th Annual Design Automation Conference*, páginas 441–446, New York, NY, USA. ACM.

Apêndice A: Comunicação MPI

De modo geral, os *clusters* possuem memórias distribuídas, ou seja, cada processo tem seu próprio espaço de endereço e não há formas de um processo acessar diretamente a memória de outros processos ou tarefas. Assim, o envio e recebimento de mensagens entre processos podem ser utilizados para realizar operações nestes *clusters*, com troca de mensagens entre os processos ou mesmo sincronização de tarefas. O envio dessas mensagens pode ser realizado utilizando MPI.

A biblioteca MPI especifica os nomes, sequência de chamada e resultados de sub-rotinas a serem chamadas por programas escritos em Fortran ou funções de programa escritas em C. Essa biblioteca especifica também a funcionalidade das rotinas de comunicação de alto nível, enquanto que as funções MPI dão acesso às implementações de baixo nível para manipular os *sockets*, *buffering*, cópia de dados, rotinas de mensagens, etc. A comunicação acontece por meio de um sistema de conexão, dessa forma, sua velocidade é determinada pelo tipo de rede utilizada bem como a sobrecarga da pilha do protocolo. A figura A.1 ilustra este cenário.

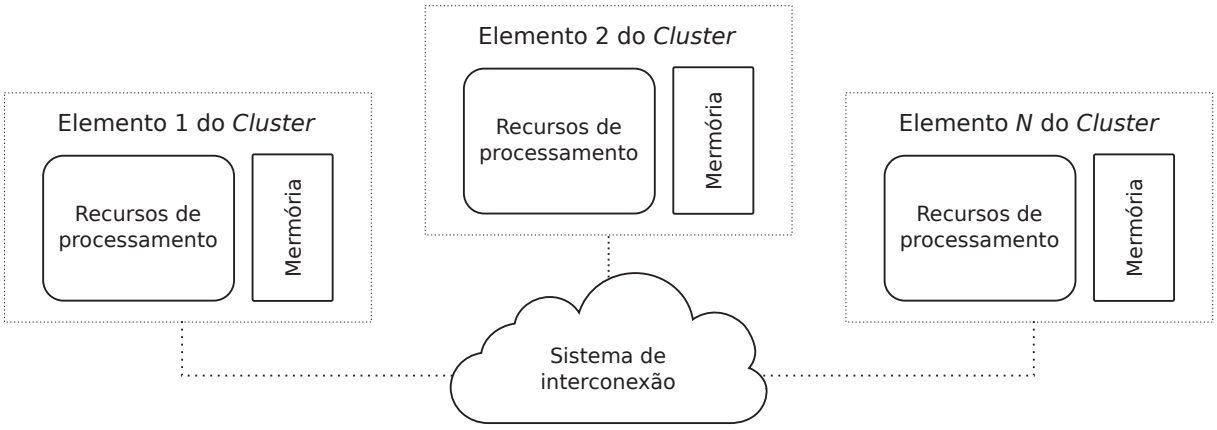


Figura A.1: Exemplo de cenário para comunicação baseada em MPI.

A sintaxe básica da construção de uma aplicação baseada em MPI está detalhada em Gropp et al. (2017). As funções da biblioteca MPI são chamadas de primitivas e servem para estabelecer a comunicação entre os processos de diferentes formas. A comunicação básica para compartilhar dados acontece com as primitivas MPI_Send e MPI_Recv, para enviar e receber uma mensagem e informações associadas a ela, respectivamente.

As primitivas de comunicação podem ser bloqueantes ou não bloqueantes. Uma chamada a uma primitiva bloqueante suspende a execução do processo até que o *buffer* da mensagem de envio ou recebimento esteja seguro para o uso. As primitivas MPI_Send e MPI_Recv são bloqueantes. Por outro lado, um chamada a uma primitiva não bloqueante inicia imediatamente a comunicação, e o *status* da transferência dos dados ou o sucesso da comunicação deve ser verificado posteriormente pelo programador. Em outras palavras, não há a suspensão da execução do processo, porém a mensagem deve ser corretamente recebida. As primitivas MPI_Isend e MPI_Irecv são primitivas não bloqueantes, note que ambas começam com a inicial "I"que denota o termo em inglês *immediate*.

Além disso, o envio de uma mensagem MPI pode ser síncrona com uso da primitiva *MPI_Ssend*, sem uso de *Buffer*, neste caso o programa deve garantir que não há dependência de *buffer* para sua comunicação. Há também a primitiva *MPI_Bsend*, que utiliza um *buffer* para armazenar a mensagem retirando a necessidade do programador determinar uma ordem correta de envio e recebimento. E por fim, a primitiva *MPI_Rsend*, na qual o remetente tem a certeza que o destinatário está apto a receber a mensagem, usado em abordagens como *Remote Direct Memory Access* (RDMA) suportados por sistemas de interconexão como o IB.

A biblioteca MPI possui também comunicações coletivas, são operações realizadas por todos processos durante uma computação. Essas operações podem ser: movimento de dados, usados para rearranjar um montante de dados dos processos ou para computar operações coletivamente (como mínimo, máximo, soma, operadores lógicos, etc., ou mesmo operações definidas pelo programador). As operações coletivas, e a primitiva *MPI_Send*, estão representadas na Figura A.2 e sucintamente descritas abaixo.

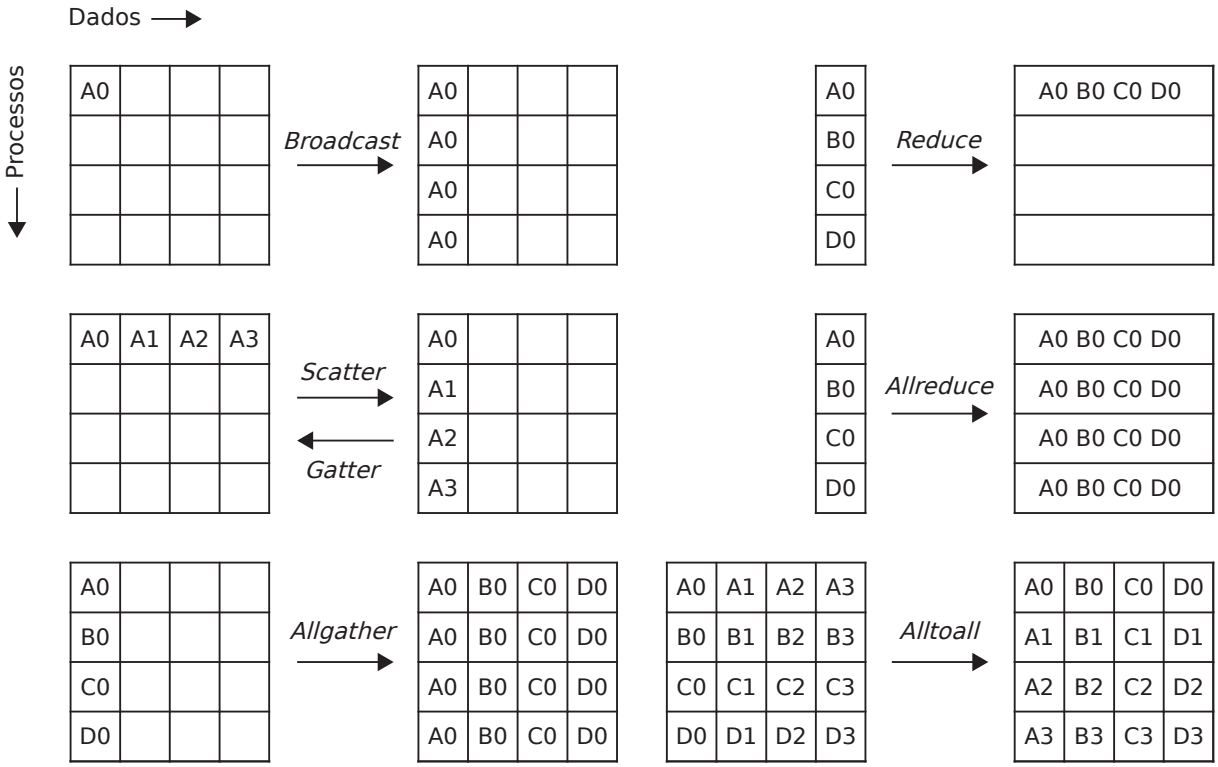


Figura A.2: Primitivas MPI.

- *MPI_Bcast*: um processo espalha sua informação a todos outros;
- *MPI_Gather*: vários processos enviam suas informações a um único processo;
- *MPI_Allgather*: vários processos enviam suas informações a todos processos;
- *MPI_Scatter*: um processo espalha pedaços distintos de sua informação para cada processo;
- *MPI_Alltoall*: todos processos enviam mensagem a todos, inclusive a si mesmo;
- *MPI_Reduce*: os dados de todos processos são destinados a apenas um processo específico (*root*);

- *MPI_Allreduce*: todos processos recebem parte de todos processos.

Algumas primitivas possuem variações para usar um vetor de mensagem, elas são: *MPI_Gatherv*, *MPI_Allgatherv*, *MPI_Scattev* e *MPI_Alltoallv*, a letra "v"acrescida ao final do nome da primitiva identifica-as.

Além das primitivas de comunicação apresentadas, a *MPI_Barrier* é uma primitiva que usa uma barreira para sincronização das mensagens entre os processos. Esta é uma coletiva especial que não permite que as chamadas dos processos continuem até todos os processos na comunicação chamarem esta barreira. Dessa forma, é possível garantir que todos processos completaram uma computação.

Apêndice B: Sistemas de criptografia

B.1 Introdução

A criptografia garante a confidencialidade por meio de ocultação da informação com uso de chave secreta, transformando um texto claro em texto cifrado e posteriormente decifrando-o com a chave (Stallings (2002)). Dessa forma, se somente o remetente e destinatário possuírem a chave criptográfica, somente eles acessarão a informação original. Para a compreensão dos termos relacionados à criptografia, algumas definições devem ser esclarecidas:

- *Texto claro*: é a informação a ser cifrada;
- *Algoritmo de criptografia*: é o algoritmo que cifrará o texto claro, transformando a informação em texto cifrado;
- *Texto cifrado*: também conhecido como texto criptografado, é o resultado do texto claro cifrado pelo algoritmo de criptografia;
- *Chave criptográfica*: ou simplesmente chave, é a informação necessária para cifrar e decifrar;
- *Algoritmo de decriptografia*: é o algoritmo que decifrará o texto cifrado, transformando-o em texto claro de saída;
- *Texto decifrado*: também conhecido como texto decriptografado, é o resultado do algoritmo de decriptografia, ou seja, o texto claro recuperado.

Os sistemas de criptografia são divididos em simétricos e assimétricos. O simétrico usa apenas uma chave compartilhada entre o remetente e o destinatário para cifrar e decifrar. O assimétrico, também conhecido como criptografia de chave pública, usa duas chaves: pública e privada. A chave pública é fornecida livremente aos remetentes para a cifragem do texto claro, enquanto que a chave privada é secreta e usada somente pelo destinatário para decifrar o texto cifrado.

Um esquema de criptografia simétrica possui: texto claro, algoritmo de criptografia, chave secreta, texto cifrado e o algoritmo de decriptografia, e geralmente são baseados em substituição e permutação da informação. Para que esse sistema seja considerado seguro, deve-se atentar aos seguintes requisitos: um oponente que conheça o algoritmo de criptografia e tenha acesso a um ou mais textos cifrados não deve conseguir decifrar o texto ou descobrir a chave; e o emissor e o receptor devem manter as suas cópias da chave compartilhada em segredo e segurança.

Os algoritmos de criptografia simétricos modernos são viáveis para o uso pois é impraticável a decriptografia de uma mensagem com base no texto cifrado mais o algoritmo de criptografia, o sistema de criptografia simétrica mais conhecido e bem aceito é o AES (Daemen e Rijmen (2002)). No entanto, é necessário utilizar um tamanho seguro para a chave, a fim de evitar ataques por força bruta. Dessa forma, o algoritmo pode ser de domínio público e apenas a chave precisa ser secreta. A desvantagem desse sistema é a necessidade da chave secreta ser

conhecida por todos comunicantes, de tal forma que se ela for revelada, todo o sistema estará comprometido.

Um esquema de criptografia assimétrica possui: texto claro, algoritmo de criptografia, chave pública, chave privada, texto cifrado e o algoritmo de decriptografia. As chaves assimétricas podem ser usadas: em sistemas criptográficos, em assinatura digital e em trocas de chaves simétricas. Os principais sistemas de criptografia que utilizam as chaves assimétricas são RSA (Rivest et al. (1978)) e ECC (Miller (1986); Koblitz (1987)).

A criptografia assimétrica ofereceu uma mudança radical nos sistemas de criptografia. Estes sistemas não são baseados em substituições e permutações como os sistemas de criptografia simétricos até então, e ao invés disso utilizam funções matemáticas e sua segurança fundamenta-se na dificuldade de resolver problemas matemáticos como o problema do logaritmo discreto (McCurley (1990)) e o problema do logaritmo discreto para curvas elípticas (Hankerson e Menezes (2011)).

Além disso, não há a necessidade de uma única chave secreta compartilhada, retirando a desvantagem dos sistemas simétricos de comprometer todo o sistema caso essa chave seja descoberta. No entanto, a distribuição da chave pública deve ser realizada de forma a garantir a autenticidade e irretratabilidade. Existem acordos e gerenciamentos de chaves que vão desde a distribuição simples da chave até sofisticadas unidades de autoridades certificadoras.

Um sistema de chave assimétrico não é mais seguro contra ataques de criptoanálise (Wiener (1990)) do que um sistema simétrico ou vice-versa, pois a segurança de qualquer sistema de criptografia depende do tamanho da chave e do trabalho computacional envolvido para a quebra da cifra. A grande vantagem da criptografia assimétrica está em não necessitar de uma chave compartilhada, pois a transmissão dessa chave é um problema em ambientes não confiáveis, como a internet. Além disso, com a criptografia assimétrica é possível realizar a assinatura digital e garantir a autenticidade da mensagem.

Entre os sistemas de criptografia assimétricos, a ECC destaca-se por permitir o uso de chaves menores do que outros sistemas de criptografia assimétrica para o mesmo nível de segurança, e por consequência, menor espaço de armazenamento e tempo de processamento. Para exemplificar essa proporção: uma chave ECC de 224 bits do ECC equivalem a 2048 bits do RSA, mas 112 bits do AES (Gura et al. (2004)). A principal desvantagem da cifra assimétrica é seu desempenho em relação a cifra simétrica, a Tabela B.1 mostra a relação entre o criptossistema simétrico AES, e os criptossistemas assimétricos ECC e RSA.

Tabela B.1: Comparação dos tamanhos das chaves em bits entre os criptossistemas AES, ECC e RSA.

AES	**ECC**	**RSA**
56	112	512
80	160	1024
112	224	2048
128	256	3072
192	384	7680
256	512	15360

Uma alternativa aos dois tipos de criptografia é mesclá-los. O ECIES é um sistema híbrido baseado em curvas elípticas e cifra simétrica. Ele utiliza chave pública e privada para estabelecer uma chave de sessão compartilhada, após ele utiliza um algoritmo de criptografia

simétrico para cifrar e decifrar a mensagem, como o AES. Dessa forma, retira-se a desvantagem de chave única do sistema simétrico e a desvantagem de baixo desempenho de sistemas assimétricos em relação aos simétricos.

Nenhum sistema de criptografia deve ser considerado inquebrável contra técnicas e algoritmos como a força bruta, a quebra deve ser considerada impraticável para a computação moderna. Assim, a segurança está no nível de dificuldade em resolver um problema matemático difícil, como a fatoração de inteiros (Montgomery (1994)), o problema do logaritmo discreto (McCurley (1990)) e logaritmo discreto para curvas elípticas (Hankerson e Menezes (2011)). Esses problemas requerem demasiado tempo para encontrar a chave ou decriptografar o texto sem a chave, o que torna este ataque impraticável e garante a segurança desses sistemas.

B.2 AES

A AES é um sistema de criptografia simétrica realizada por rodadas, na qual cada rodada completa consiste em 4 funções distintas, além da expansão da chave:

1. Substituição de bytes: ou *Sub Bytes*, utiliza uma caixa-*S* para realizar uma substituição *byte* a *byte* do bloco;

2. Deslocamento de linhas (Permutação): ou *Shift Row*, realiza uma permutação simples;

3. Embaralhamento de colunas: ou *Mix Columns*, realiza uma combinação linear com operações aritméticas sob corpo binário finito (corpo de Galois);

4. Operação XOR com uma chave, ou *Add Round Keys*, realiza um XOR *bit* a *bit* simples do bloco atual com uma parte da chave expandida.

A expansão da chave secreta ocorre em um vetor de palavras (*words*). Ela é necessária para o escalonamento das chaves para entrada em cada rodada. Stallings (2002) mostra o passo-a-passo para cifra e decifra com a AES e detalhes da implementação do algoritmo.

B.2.1 Implementação na rede em chip

O primeiro passo para implementar o AES em uma rede em chip é atender sua premissa básica: todos comunicantes devem conhecer previamente a chave secreta. Esta informação pode ser atribuída de duas formas:

1. Cada *hub* sem fio pode conter a chave secreta armazenada no momento de fabricação;

2. Cada *hub* sem fio pode receber a chave secreta após fabricação, de forma confiável, como por exemplo com o uso de um *token* físico;

Se essa premissa for atendida, os passos seguintes estão organizados na Tabela B.2. Eles ilustram o funcionamento padrão do AES para qualquer sistema, ou seja, são definições básicas de implementação.

As principais vantagens em utilizar a AES na rede em chip são: melhor desempenho para cifra e decifra em relação a cifra assimétrica, um aceitável nível de segurança oferecido com chave menor e o algoritmo que implementa o AES ser altamente paralelizável. No entanto, há algumas desvantagens:

Tabela B.2: Etapas da cifra e decifra com AES.

Passo	Descrição
1	Preparar a entrada para o algoritmo: • A entrada para cifra é a chave e um bloco de texto claro. A chave e o bloco podem ser representados cada um por uma matriz quadrada de *bytes*; • Tanto o bloco como a chave possuem tamanhos independentes um do outro, os quais podem ser 128, 192 ou 256 *bits*. O tamanho de 128 *bits* para a chave da AES é o mais implementado; e • A chave deve ser expandida para gerar e adicionar a chave da primeira rodada.
2	Realizar as rodadas: • São realizadas 10 rodadas; • A cada rodada a chave é expandida e adicionada como chave da rodada; e • Ao término da décima rodada, o texto está cifrado.
3	Processo de decifra: • A decifra ocorre de forma similar; • Ao invés de entrar o texto claro, é entrado o texto cifrado; • A chave é a mesma; • Em cada rodada são realizadas as operações inversas: – embaralhamento inverso de colunas; – substituição inversa de *bytes*; e – deslocamento inverso de linhas. • Além dessas operações, a chave da rodada é expandida em ordem reversa e adicionada em cada rodada da decriptografia.

- A chave secreta deve ser conhecida previamente por todos os PEs. Isso pode ser uma grande desvantagem pois se considerar o uso de 3PIP, aumenta a chance da informação ser obtida indevidamente por um PE comprometido. Além disso, esta desvantagem inviabiliza o uso de rede em chip hierárquica ou expansível que seja independente de fabricantes, pois compartilharia uma única chave secreta entre todos os fabricantes e isto não é seguro;

- Há o problema da custódia da chave, ou *Key Escrow*, pois o PE deve manter consigo a chave simétrica por um longo período, e dessa forma ela pode ser obtida se não houver formas de proteger esta informação;

- Se a chave for descoberta, por perda ou violação, todo o sistema está comprometido.

- A chave do algoritmo AES implementado em rede em chip já foi quebrada com ataque *side channel* (Reinbrecht et al. (2017)) lendo a memória do PE e consequentemente comprometendo todo o sistema.

B.3 Criptografia de Curvas Elípticas

A Criptografia de Curvas Elípticas é formada pela teoria de curvas elípticas, aritmética envolvida e o corpo finito. A ECC é um criptossistema de chave pública proposta independentemente por Miller (1986) e Koblitz (1987) a qual provê um aceitável nível de segurança usando chaves a partir de 160 bits, porém o recomendável é utilizar 256 bits, tornando-o possível utilizar

em dispositivos computacionalmente restritos. Detalhes sobre a teoria das curvas elípticas e sua aritmética pode ser encontrado em Hankerson et al. (2006).

B.3.1 Descrição do sistema

Para usar a ECC é necessário:

1. Definir a natureza do corpo finito;
2. Implementar as operações aritméticas do corpo finito;
3. Selecionar uma curva;
4. Definir o ponto gerador do corpo finito;
5. Realizar o mapeamento das mensagens para os pontos da curva.

A natureza do corpo finito deve ser escolhida entre corpo primo ou binário. As operações aritméticas, tanto em corpo primo ou binário, resultam em um polinômio com valores fechados no conjunto, entre essas operações, as mais importantes são a adição usando XOR (bit a bit) e a multiplicação modular, uma vez que a partir delas são derivadas a subtração e a exponenciação.

Há diferentes curvas elípticas que podem ser utilizadas na ECC. Uma curva elíptica é formada pelos parâmetros que definem a curva e um ponto gerador. Um ponto é entendido por dois valor representados tradicionalmente por x e y. A curva escolhida deve permitir a existência de um grupo de corpo finito, grande o suficiente para permitir o mapeamento de informações para pontos na curva. A Certicom (CERTICOM (2000)) e o *National Institute of Standards and Technology* (NIST) (NIST (1999)) recomendam curvas testadas e homologadas por eles, que cumprem todos requisitos.

O ponto gerador, denotado por G é análogo a base g do problema do logaritmo discreto, e começando a partir dele é possível gerar um conjunto, sendo suficiente para calcular a potência dos elementos do corpo finito. Com o conjunto gerado, é necessário mapear as mensagens para os pontos da curva. Este processo é chamado *Message Embedding*.

A representação de texto plano em pontos da curva, ou simplesmente mapeamento, não acontece de maneira simples. Não há um tempo polinomial conhecido e algoritmo determinístico para atribuir pontos em uma curva elíptica arbitrária (Trappe e Washington (2005)). Entretanto, existem métodos probabilísticos rápidos para encontrar pontos a serem usados para codificar mensagens. Alguns desses métodos são apresentados, analisados e discutidos em Trappe e Washington (2005); King (2009); Muthukuru e Sathyanarayana (2012); Amiri e Elkeelany (2013); Sengupta e Ray (2016).

Com a curva elíptica é possível também utilizar assinatura e funções de resumo para garantir a autenticidade e integridade, respectivamente. A *Elliptic Curve Digital Signature Algorithm* (ECDSA) é utilizada para assinatura digital sob curvas elípticas, esta assinatura garante a integridade das mensagens e verifica a autenticidade do remetente. A integridade da mensagem é garantida com uso de um resumo, ou *Hash*, no processo de assinatura. Este resumo é implementado com o algoritmo *Secure Hash Algorithm* (SHA). A teoria que fundamenta e os detalhes para implementação do ECDSA podem ser encontrados em Johnson et al. (2014).

O funcionamento da ECC para qualquer sistema computacional está descrito em Hankerson et al. (2006). Ele pode ser dividido em duas fases: a inicial responsável por definir parâmetros que utilizarão durante a criptografia; e o cálculo da criptografia, no qual são realizados as operações de cifragem e decifragem. Considerando a implementação em um sistema

computacional, a primeira fase pode ser realizada no momento de fábrica ou na inicialização do sistema. Resumidamente, este funcionamento acontece da seguinte forma:

- Fase inicial:
  - Escolha da curva (parâmetros da curva elíptica e o ponto gerador (G))
  - Cada comunicante deve escolher aleatoriamente um inteiro K. Este valor será sua chave privada;
  - Cada comunicante calcula um ponto na curva como sua chave pública:

    $$P_A = (kG)$$

    onde P_A é a chave pública do comunicante A.
- Cifragem
  - A mensagem m deve ser mapeada para um ponto na curva P_m. Esse mapeamento pode ou não ser na fase inicial.
  - O ponto da mensagem P_m será cifrado para P_{c1}:

    $$P_{c1} = (P_m + K_A P_B)$$

    onde K_A é a chave privada do comunicante A e P_B é a chave pública do comunicante B.
  - O ponto cifrado final P_c é a junção dos pontos P_A e P_{c1}:

    $$P_c = [P_A, P_{c1}]$$

- Decifragem
  - O comunicante B recupera o P_m a partir do P_c:

    $$P_m = (P_{c1} - K_B)P_A$$

    onde K_B é a chave privada do comunicante B.
  - O ponto P_m deve mapeado para a mensagem original m.

B.3.2 Implementação na rede em chip

As premissas básicas para utilizar a ECC na comunicação inter-chips são: os parâmetros da curva elíptica devem estar acordados entre os sistemas, afim de que todos os *hubs* sejam capazes de gerar a sua chave pública a partir do ponto gerador; cada sistema deve possuir uma chave pública e outra privada; as chaves públicas devem ser distribuídas de forma segura.

A tabela B.3 apresenta os passos para cifrar e decifrar com a ECC por meio de um exemplo. A comunicação acontece entre duas ENoCs $2D$ 4×4, nela um PE do sistema A envia uma mensagem para um PE do sistema B. Os *hubs* dos sistemas serão responsáveis por encaminhar essa mensagem pelo canal sem fio de forma segura utilizando ECC.

As principais vantagens em utilizar a ECC são: os sistemas não precisam usar uma chave secreta compartilhada, pois cada um possuei seu par de chaves, uma pública e a outra

Tabela B.3: Etapas da ECC.

Passo	Descrição
1	A curva elíptica selecionada é acordada entre os sistemas juntamente com suas inforamções de identificação. As chaves públicas são trocadas por meio de acordo de chaves entre os sistemas;
2	O PE remetente envia ao seu *hub* sem fio a mensagem destinada a um PE de outro sistema.
3	O *hub* sem fio recebe a mensagem, realiza o mapeamento para pontos da curva elíptica utilizando um algoritmo de mapeamento .
4	O *hub* do sistema *A* cifra a mensagem *M* e produz o ponto cifrado (*PC*) da seguinte forma: *PC* = [*M* + *chave privada do hub A*] × [chave pública do *hub B*].
5	O *hub B* recupera o ponto decifrado da seguinte forma: *M* = [*PC* - *chave privada do hub B*] × [*chave pública do hub A*].
8	O *hub B* mapeia os pontos decifrados para a mensagem original utilizando o mesmo algoritmo de mapeamento, gera um novo pacote e encaminha-o ao destinatário local.

privada. Essa vantagem elimina o problema da custódia de chave; se uma chave for descoberta, não comprometerá todo o sistema; o sistema em chip pode ser expansível, uma vez que não há a necessidade de uma chave conhecida previamente por todos. A principal desvantagem deste sistema é o seu desempenho em relação aos sistemas de criptografia simétrica, ou híbrido, como o ECIES.

B.4 Elliptic Curve Integrated Encryption Scheme

O uso de criptografia assimétrica permite uma série de vantagens sob a criptografia simétrica, no entanto o desempenho não é um deles. O ECIES combina as vantagens de cada sistema, integrando o criptossistema assimétrico ECC e o simétrico AES. Com isso, há a possibilidade de usar o bom desempenho para cifrar e decifrar sem necessitar da chave secreta compartilhada.

B.4.1 Descrição do sistema

A ECIES combina a troca de chaves para gerar uma chave efêmera, ou de sessão, e utilizá-la como chave secreta compartilhada em uma cifra simétrica. A troca de chave utiliza *Elliptic Curve Diffie-Hellman* (ECDH), uma variação do acordo de chaves Diffie-Hellman com curvas elípticas. O ECMQV, também pode ser utilizado para a troca de chave sob curvas elípticas (Law et al. (2003)). A cifra simétrica utilizada é o AES com tamanhos de 128, 192 ou 256 bits, porém o nível de segurança provido pelos 128 *bits* é considerado suficientemente seguro para a comunicação e possui um bom desempenho.

Em síntese, a ECIES usa as seguintes funções:

- *Acordo de chave:* função usada para gerar uma chave secreta pelas duas partes;
- *Derivação de chave (KDF)*: mecanismo que produz um conjunto de chaves;
- *Hash*: resumo da mensagem para verificação de integridade;
- *Cifragem*: algoritmo de cifra simétrica;

- *Código de autenticação de mensagem (MAC)z.*

Todo fundamentação da ECC também se aplica a ECIES, ou seja, são utilizados os corpos finitos e a aritimética modular sob o corpo finito selecionados para gerar as chaves de sessão compartilhadas. A cifragem e decifragem são diferentes, pois elas são realizadas com o AES.

B.4.2 Implementação na rede em chip

Para implementar o ECIES na ENoC é necessário que os parâmetros da curva elíptica devem estar acordados previamente. Além disso cada sistema deve ser capaz de criar um par de chaves efêmera para cada sessão de comunicação. A Tabela B.4 mostra os passos para cifrar com ECIES, enquanto que a Tabela B.5 mostra os passos para a decifra. Assim como no exemplo anterior, o *hub A* encaminha uma mensagem cifrada para o *hub B* de sistemas diferentes.

Tabela B.4: Etapas da cifragem com ECIES.

Passo	Descrição
1	*Hub A* deve criar um par de chaves efêmeras consistindo em um elemento de corpo finito e o ponto gerador da curva elíptica. O par de chave deve ser gerado exclusivamente de forma pseudoaleatória.
2	O *hub A* usará uma função de acordo de chave para criar um valor secreto compartilhado com o *hub B*, resultado da multiplicação escalar do elemento do corpo finito e a chave pública do *hub B*.
3	Então, *hub A* deve obter a chave secreta e usá-la, opcionalmente com outros parâmetros, como entrada para a função de derivação de chave (KDF). A saída dessa função é a concatenação da chave de cifra simétrica e uma chave MAC.
4	Com a chave compartilhada e a texto claro, *hub A* usará o AES para produzir o texto cifrado.
5	Com o texto cifrado e os parâmetros opcionais, o *hub A* deve usar a função MAC para produzir uma *tag*.
6	Finalmente, o *hub A* enviará um criptograma contendo a sua chave pública efêmera, a *tag*, e o texto cifrado para o *hub B*.

Tabela B.5: Etapas da decifragem com ECIES.

Passo	Descrição
1	Após receber o criptograma do *hub A*, o *hub B* deve recuperar a chave pública efêmera, a *tag*, e a mensagem cifrada.
2	Usando a chave pública efêmera recuperada e sua própria chave privada, *hub B* irá multiplicar ambos elementos e produzir o valor secreto compartilhado.
3	Tomando como entrada a chave compartilhada e os parâmetros opcionais usados por*hub A*, o *hub B* deve produzir a mesma cifra e chave MAC por meio da função KDF.
4	Com a chave MAC, a mensagem cifrada, e os mesmos parâmetros opcionais usados por *hub A*, *hub B* irá primeiro computar o *tag*, e então ele irá comparar o seu valor com o *tag* recebido. Se o valor for diferente, *hub B* deve rejeitar o criptograma devido a falha na verificação MAC.
5	Se o valor *tag* gerado por *hub B* é o correto, então ele irá continuar o processo para decifrar a mensagem cifrada usando o AES e a chave simétrica compartilhada. Com isso, *hub B* recupera o texto plano que o *hub A* enviou a ele.

Por se tratar de uma abordagem híbrida, os *hubs* não precisam previamente possuir a chave secreta. Assim, não há o problema de posse da chave, pois é usado uma chave de sessão. Além disso, o uso do AES com uma chave de 128 bits, torna a cifra e decifra com melhor desempenho. No entanto, as principais desvantagens desse sistema são:

- O acordo de chave utilizado pode ser vulnerável ao ataque do homem do meio (Chen et al. (2007));
- Se for implementado alguma forma de unidade central para mediar o acordo de chave, esta unidade pode resultar em gargalo.
- Possui pior desempenho do que uma cifra simétrica.

B.5 Considerações

Este apêndice apresentou os conceitos básicos dos sistemas de criptografia AES, ECC e ECIES e os passos para sua implementação em uma NoC. O AES é o sistema de criptografia simétrico mais utizado. Ele possui melhor desempenho com adequado nível de segurança comparado ao ECC e ECIES. Entretanto, o uso de uma única chave de criptografia compartilhada com todos comunicantes é sua grande desvantagem, pois se a chave for comprometida todo o sistema também será.

ECC é considerado o sistema de criptografia assimétrico mais eficiente. Ele usa uma pequena chave para prover o mesmo nível de segurança que outros sistemas de criptografia assimétrica oferecem, porém com chaves maiores. Antes de qualquer comunicação, o remetente deve obter a chave pública do destino. Uma vez obtida, ela é localmente armazenada pelo *hub*. Esta característica pode ser uma vantagem para a ENoC pois a chave púublica pode ser obtida no ponto inicial ou no momento de expansão. ECIES usa o criptosistema de curvas elípticas para gerar uma chave AES comum de sessão. A principal desvantagem é a sobrecarga da chave des sessão, considerando o tempo para as trocas de chaves e geração da chave de sessão.