Universidade de Aveiro
Departamento de Electrónica, Telecomunicações e Informática,
2013

Michael da Conceição Costa

Implementação de uma RedBox Ethernet para Aplicações de Tempo-Real

RedBox Ethernet Implementation for Real-Time Applications

Michael da Conceição Costa

Implementação de uma RedBox Ethernet para Aplicações de Tempo-Real

RedBox Ethernet Implementation for Real-Time Applications

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Eletrotónica e Telecomunicações , realizada sob a orientação científica do Doutor Arnaldo Oliveira Professor Auxiliar do Departamento de Electrónica, Telecomunicações e Informática e do Doutor Paulo Pedreiras, Professor Auxiliar do Departamento de Electrónica, Telecomunicações e Informática.

o júri / the jury

presidente / president

Professor Doutor Ernesto Fernando Ventura Martins
Professor Auxiliar, Universidade de Aveiro

vogais / examiners committee

Professor Doutor Paulo José Lopes Machado Portugal
Professor Auxiliar, Universidade do Porto - Faculdade de Engenharia

Professor Doutor Paulo Bacelar Reis Pedreiras
Professor Auxiliar, Universidade de Aveiro

agradecimentos

A realização deste trabalho só foi possível com o apoio de várias pessoas. Em primeiro lugar gostaria de agradecer ao Professor Doutor Arnaldo Oliveira e ao Professor Doutor Paulo Pedreiras. Um grande obrigado pela disponibilidade demonstrada e pelas reuniões bem humoradas. Agradeço as sugestões e os conhecimentos científicos que me permitiram realizar este trabalho. Agradeço aos meus pais, padrinhos, irmã e restantes familiares que apesar da distancia sempre me apoiaram e foram essenciais ao longo da minha vida. A todos os colegas presentes presentes no laboratório 319 pelo bom ambiente proporcionado por eles! Um muito obrigado aos meus amigos pelo grande espírito de camaradagem e pela sua amizade.

Resumo

Os sistemas embutidos distribuídos, devido à sua distribuição, são usados em várias áreas tais como a automação industrial, a indústria automóvel ou equipamentos médicos. Este tipo de sistemas necessita de redes dedicadas para que as várias partes do sistema comuniquem entre elas. Recentemente, vários protocolos baseados na tecnologia Ethernet têm vindo a ser desenvolvidos para esse tipo de sistemas. Esses protocolos aproveitam algumas das vantagens da Ethernet tais como uma grande largura de banda, baixo custo de produção e instalação. Alguns destes sistemas embutidos distribuídos são considerados crítico, o que significa que uma falha do sistema pode provocar resultados catastróficos, e.g. danificar equipamentos ou provocar danos em pessoas. Este tipo de sistemas necessita de grandes níveis de fiabilidade. Nesse sentido são usadas técnicas de tolerância a faltas baseadas no uso de redundância na rede e nos nós que a constituem. Foram desenvolvidos vários mecanismos de redundância sobre Ethernet. Citando alguns deles o Rapid Spanning Tree Protocol (RTSP), High-Availability Seamless redundancy protocol (HSR), Parallel Redundancy Protocol (PRP) ou Rapid Ring Recovery (RRR). O Rapid Ring Recovery é uma proposta recente. Este protocolo tem como principal vantagem o seu tempo de recuperação, uma vez que consegue recuperar de falhas nas ligações em menos de 1 milissegundo. A sua utilização está limitada a uma topologia em anel, permitindo acrescentar nós à rede sem grande complexidade. Mas ao só suportar redes em anel limita a sua utilização. Esta implementação usa a tecnologia FPGA e foi validada experimentalmente. Os resultados experimentais demonstram ser possível implementar uma RedBox, que consiste numa interface para uma rede Ethernet a funcionar com Rapid Ring Recovery, usando uma FPGA. A implementação deste equipamento foi validada experimentalmente e foi possível verificar que é possível reagir a erros nas ligações em menos de 1ms, tal como esperado.

Abstract

Networked Embedded Systems (NES) are used in a broad range of application domains, from automation and industrial machinery to vehicles, medical equipment. Due to their distributed nature and environment constraints, NES often depend on dedicated networks. Recently, several Ethernet-based protocols have been developed for those systems, taking advantage of some of Ethernet's appealing attributes, e.g. large bandwidth, cheap silicon and high availability, while removing or reducing the sources of non-determinism arising from its MAC protocol and/or from the current switched architecture. Many NES are intrinsically safety-critical, meaning that a system failure may have catastrophic results, e.g., damage to expensive equipment and/or the environment or endanger human lives. This type of systems requires high levels of reliability and integrity, which can be achieved using fault tolerance techniques, such as nodes and network redundancy. Different Ethernet redundancy mechanisms have been developed, such as Rapid Spanning Tree Protocol (RTSP), High-Availability Seamless redundancy protocol (HSR), Parallel Redundancy Protocol (PRP) or Rapid Ring Recovery (RRR). RRR is a recently proposed protocol, based on the use of multiple virtual rings, that is fully distributed, allows recovery times in the order of 1ms, is easily implementable and does not require a duplicated infrastructure. This paper presents the implementation of a redundancy box (RedBox) implementing the RRR protocol. The RedBox offers a standard Ethernet interface to the nodes locally attached to it, providing a seamless integration of the redundancy architecture. Furthermore, the RedBox allows the direct formation of the network, without depending on switches, therefore simplifying the network deployment and maintenance. The RedBox was implemented using FPGA technology and experimentally validated. The experimental results obtained show the feasibility and correction of the approach, being possible to react to errors in less than 1ms, as expected.

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Acrónimos

BPDU Bridge Protocol Data Units.

FCS Frame Check Sequence.

FPGA Field-programmable gate array.

HSR High-availability Seamless Redundancy protocol.

IEEE Institute of Electrical and Electronics Engineers.

MAC Medium Access Control.

PHY Physical Layer Transceiver.

PRP Parallel Redundancy Protocol.

RedBox Redundancy Box.

RMII Reduced Media Indenpedant Interface.

RRR Rapid Ring Recovery.

RSTP Rapid Spanning Tree Protocol.

STP Spanning Tree Protocol.

TEMAC Tri-Mode Ethernet MAC.

TRILL TRansparent Interconnection of Lots of Links.

UART Universal asynchronous receiver/transmitter.

VLAN Virtual Local Area Network.

Capítulo 1

Introdução

Os sistemas embutidos estão cada vez mais presentes no nosso dia a dia. Devido à vários fatores, como por exemplo a localização física dos equipamentos, alguns desses sistemas são distribuídos. Em certos casos esses sistemas são usados em situações críticas. Os sistemas embutidos distribuídos críticos exigem a existência de meios de comunicação para interligar os equipamentos que os constituem. Quando o sistema é crítico, o meio de comunicação também se torna crítico. Nestes sistemas as falhas são indesejáveis. No caso da ocorrência de uma falta, o tempo de recuperação do sistema deve ser baixo.

Este trabalho aparece no sentido de estudar abordagens sobre o uso da redundância existente nas ligações dos equipamentos. Todos os protocolos abordados nesta dissertação têm como objetivo evitar falhas e reduzir os tempos de recuperação. Todos os protocolos abordados usam a tecnologia Ethernet.

1.1 Enquadramento e Motivação

Na atualidade os sistemas embutidos distribuídos são encontrados em diversos contextos, sendo usados tanto em equipamentos de uso diário como em ambientes críticos tais como a aviação, indústria aerospacial e sistemas de controlo. Com a evolução tecnológica, a complexidade e especificidade dos sistemas embutidos tem vindo a aumentar. Assim, surgiu a necessidade de distribuir os sistemas pela sua funcionalidade de modo a facilitar a sua manutenção ou atualização. Com essa distribuição, as redes tornaram-se fundamentais pois são elas que vão permitir a partilha de variados resultados, informações ou recursos dentro de um sistema distribuído.

Os meios de comunicação usados em sistemas distribuídos críticos têm que ser robustos e suportar a ocorrência de faltas, devido e.g. interferência electro-magnética, sem comprometer o bom funcionamento do sistema, garantindo assim que não irão ocorrer situações indesejadas devido a uma falha de comunicação. Assim, num sistema distribuído crítico o meio de comunicação também é crítico.

Nos sistemas críticos é imperativo reduzir o número de falhas e atenuar os danos provocados por elas. Os tempos de recuperação têm de ser pequenos. Em muitos desses sistemas o uso de redundância é desejável. Entende-se por redundância replicar certas partes do sistema ou repetir certas acções. Isso permite que o sistema não falhe ou recupere mais rapidamente no caso em que ocorra uma falha. A redundância pode ser tipicamente introduzida à nível espacial ou temporal. A redundância espacial é usada quando certas funcionalidades do sistema

são replicadas noutros locais. Essa abordagem pode ser implementada ao duplicar um certo equipamento (tanto o hardware como o software). Numa rede de comunicação, esse tipo de redundância pode ser introduzido com o uso de ligações suplementares entre os equipamentos. Isso permite que cada equipamento tenha mais que um "caminho"para comunicar com outro equipamento. A replicação de funcionalidades num sistema é útil nos casos em que ocorra uma falha permanente e.g destruição do equipamento. É conhecido por redundância temporal as abordagens que repetem uma acção. Por exemplo nos sistemas de comunicação é muito comum o reenvio de uma mensagem quando um erro é detetado. Esta repetição é útil para os casos em que ocorrem erros esporádicos. Assim o reenvio da trama depois de um intervalo temporal é suficiente para superar a falta.

Esta dissertação tem como finalidade desenvolver um dispositivo que reduza o número de falhas e os tempos de recuperação numa rede usando a redundância. Os meios de comunicação usados em sistemas distribuídos críticos são implementados com recurso a diversas tecnologias. Sendo a ligação elétrica através de condutores o método mais utilizado na indústria. Essa escolha deve-se ao facto dos cabos oferecerem bons resultados no que toca a segurança e fiabilidade e um preço reduzido. Como estas características são desejadas em sistemas críticos o seu uso é frequente. Este trabalho enquadra-se no projeto Serv-CPS que consiste em apoiar o desenvolvimento de estruturas de rede baseadas em Ethernet comutada. Assim a tecnologia escolhida para este trabalho foi a Ethernet.

A elevada velocidade e disponibilidade de equipamento para redes Ethernet tem motivado a investigação sobre a adequação e utilização desta tecnologia de rede noutros domínios de aplicação distintos do propósito inicial: as redes de dados de uso geral em instalações empresariais e domésticas.

Um dos domínios de aplicação em que se acredita que a rede Ethernet venha a ter um papel fundamental num futuro próximo são as redes de comunicação para aplicações críticas de tempo-real, usadas por exemplo em infraestruturas industriais, de transporte e produção de energia. Neste tipo de sistemas embutidos distribuídos, a Ethernet (eventualmente dotada de extensões específicas) substituirá com vantagens, essencialmente ao nível da velocidade e custo, os tradicionais meios de comunicação atualmente empregues na interligação dos nodos associados a sensores, atuadores, controladores e interface.

Para esse efeito já existem vários protocolos sobre Ehternet que fazem uso da redundância. O Rapid Spanning Tree Protocol (RSTP) [3], o High-availability Seamless Redundancy protocol (HSR) [2], Parallel Redundancy Protocol (PRP) [1], o TRansparent Interconnection of Lots of Links (TRILL) [6] e Rapid Ring Recovery (RRR) [10] são alguns dos protocolos com esse proposito.

O RRR foi o protocolo escolhido para ser implementado no dispositivo desenvolvido no contexto deste trabalho. O RRR é uma proposta recente de um protocolo que usa a redundância para recuperar de falhas. O RRR consegue obter tempos de recuperação inferiores a 1 ms e é baseado no uso de vários anéis virtuais. O RRR é completamente distribuído e não necessita duplicação da infraestrutura. Isso permite uma implementação fácil de se realizar. Nesta dissertação é descrito o desenvolvimento de uma Redundancy Box (RedBox) para o RRR. Esta RedBox permite integrar nós locais numa rede RRR. Essa interligação é realizada através de uma interface Ethernet standard, o que torna esse processo simples e rápido. A interligação de várias RedBox permite a criação de uma rede RRR sem necessidade de utilização de *switches* facilitando a expansão da rede.

1.2 Objetivos

No âmbito desta dissertação pretende-se efetuar o desenvolvimento de uma RedBox para a interligação de equipamentos Ethernet em sistemas embutidos distribuídos de tempo-real com tolerância a faltas. Para esse efeito foram efetuados os seguintes procedimentos:

- Estudo sobre Ethernet e a sua utilização em dispositivos baseados em Field-programmable gate array (FPGA).
- Levantamento do estado da arte nas comunicações Ethernet.
- Recolha de informação sobre protocolos funcionando sobre Ethernet e que tirem partido de redundância.
- Escolha do protocolo e da topologia da rede.
- Desenvolvimento da infraestrutura baseada em FPGA.
- Teste do funcionamento e obtenção de resultados.

1.3 Organização

Esta dissertação está organizada com a seguinte estrutura:

- No capítulo 2 é dado a conhecer o estado atual da tecnologia abordada nesta dissertação e são apresentados vários protocolos com carateristas desejadas. Cada protocolo é analisado em detalhe e no fim do capítulo é apresentado uma breve comparação. A comparação entre os protocolos analisados incide sobre as topologias de rede suportadas e tempos de recuperação.
- No capítulo 3 está uma descrição aprofundada do protocolo RRR. Essa descrição foca-se essencialmente sobre a estrutura da trama, topologia da rede e o seu funcionamento.
- No Capítulo 4 é explicado qual foi a abordagem efetuada para a implementação do dispositivo. É descrito cada bloco funcional implementado para a realização do dispositivo.
- No capitulo 5 é apresentado um sniffer Ethernet desenvolvido em FPGA com intuito de ser usado como dispositivo de testes.
- No capítulo 6 são apresentados os resultados e uma comparação com os resultados esperados.
- No capítulo 7 é feita uma análise sumaria ao trabalho realizado e sobre novas abordagem possíveis para trabalho futuro.

1.4 Contribuição

A proposta inicial do RRR era implementada com recurso de um *switch* comercial associado a uma placa de desenvolvimento com uma FPGA. Neste trabalho foi possível demonstrar a possibilidade de incorporar todas as funcionalidades num só dispositivo. Após a obtenção dos

resultados obtidos durante o trabalho realizado no âmbito desta dissertação, foi realizado um artigo para comunicar os resultados obtidos. Esse artigo foi publicado e apresentado no evento INForum - Simpósio de Informática [13].

Capítulo 2

Conceitos Fundamentais

Os sistemas embutidos distribuídos são utilizados em vários sectores, tais como a indústria automóvel ou aerospacial. Para permitir aos nós, que constituem o sistema, a capacidade de trocar dados entre si, é necessário a existência de uma infraestrutura que lhes permita tal comunicação. Para esse efeito, usam-se redes de comunicação adequadas. Como todos os elementos do sistema, as redes de comunicação também são suscetíveis a erros e esses erros podem provocar falhas no sistema. Como este tipo de sistema é muito utilizado em sistemas críticos, é essencial existirem mecanismos para que o sistema seja capaz de tolerar tais erros. Esses mecanismos têm que permitir ao sistema cumprir com os seus constrangimentos temporais mesmo na presença de erros. A maioria desses mecanismos são baseados no uso de redundância, que passa pela duplicação dos nós ou da própria rede. Este trabalho está focado na tolerância a erros em sistemas que usem a tecnologia Ethernet. Já existem várias abordagens com esse propósito, tais como o RSTP, HSR, PTP, TRILL ou RRR. O protocolo escolhido para este trabalho foi RRR, por permitir um tempo de recuperação curto e ser simples de implementar.

2.1 Sistemas de tempo real

Na sociedade atual, existe uma necessidade crescente em utilizar sistemas computacionais. Estes sistemas em muitos casos têm interação com outros sistemas, pessoas ou mesmo com o mundo real. Nesta perspectiva aparecem os sistemas de tempo real, os quais se encontram na atualidade num variado leque de equipamentos, desde equipamentos domésticos a industriais.

Segundo Kopetz [12] , um sistema de tempo real tem que ser capaz de reagir adequadamente a um estímulo externo dentro de um intervalo de tempo limitado pela situação. Esse intervalo de tempo no qual é necessário uma ação e resultados é conhecido por *deadline*. Para fornecer uma resposta adequada, o sistema pode ter que recolher informações sobre o evento, o momento em que ocorreu por exemplo, e processar toda essa informação. Esses eventos podem aparecer periodicamente, como por exemplo em sistemas de controlo em malha fechada clássicos, ou então esporadicamente como no caso de sinalização de um alarme.

Em certas situações o sistema pode não ser capaz de cumprir com o *deadline*. Se o não comprimento do *deadline* não tiver nenhuma consequência grave e os resultados continuarem a ter alguma utilidade esse sistema poderá ser considerado não crítico. No caso contrário, em que o resultado de um incumprimento do *deadline* pode causar uma catástrofe, esse sistema será considerado crítico. Um sistema aéreo é um bom exemplo de um sistema crítico, pois se

Tabela 2.1: Sistemas críticos versus não críticos

Características	Sistema crítico	Sistema não crítico
Tempo de Resposta	Necessário	Desejado
Desempenho em Sobre carga	Previsível	Desapontante
Controlo do Tráfego	Meio Ambiente	Sistema Computacional
Segurança	Frequentemente Crítico	Não Crítico
Tamanho dos Dados	Pequeno/Médio	Grande
Tipo de Redundância	Ativa	Checkpoint-Recovery
Integridade dos Dados	Curto Prazo	Longo Prazo
Deteção de Erros	Autónoma	Assistida pelo Utilizador

alguma coisa falhar pode causar a falha do sistema, e provocar a sua destruição e colocar a vida de pessoas em perigo.

É fácil perceber que um sistema crítico tem necessidades diferentes de um não crítico. Por essa razão eles tem que ser projetados de maneiras diferentes. Algumas dessas diferenças podem ser observadas na tabela 2.1 retirada de [12].

Um acontecimento inesperado pode comprometer os resultados de um sistema de tempo real. Nesse sentido é importante o sistema ser previsível para permitir que o seu funcionamento seja o desejado. Por exemplo, o conhecimento do código associado às tarefas, bem como da plataforma em que é executada permite saber com antecedência quais são as suas necessidades, os tempos de execução e outros parâmetros úteis para o agendamento das tarefas. Para além da previsibilidade, existem outras características desejadas num sistema de tempo real, citando algumas[5]:

`Pontualidade`: Os resultados obtidos têm de ser corretos, não só no seu conteúdo como no domínio do tempo. Nestes sistemas existe uma grande exigência em termos temporais.

`Projetado para Picos de Carga`: Os sistemas têm que suportar com picos de carga. Eles tem que ser projetados para garantir o funcionamento para vários casos, mesmo com uma sobre-utilização dos recursos do sistema.

`Previsibilidade`: Para garantir um bom funcionamento, o sistema tem que ser previsível. Isso será muito útil para o sistema saber com antecedência se é capaz de realizar todas as tarefa.

`Tolerância a faltas`: O sistema tem que ser tolerante a faltas. Mesmo que aconteça um erro não desejado o sistema tem que continuar a funcionar. Por exemplo num sistema de comunicação, se não receber uma trama devido à uma falta, o sistema tem que continuar a receber as tramas seguintes.

`Facilidade de Manutenção`: O sistema deve ser projetado numa arquitetura modular. Esta característica é muito importante para ser fácil aplicar possíveis alterações ou reparações, conseguindo-se atualizar só a parte da qual estamos interessados sem nos preocupar com o resto.

Quando chega o momento de avaliar um sistema de tempo real há certas características que são muito interessante e úteis de se avaliar. A seguir são apresentadas algumas destas retiradas de [5]:

`Fiabilidade`: A Fiabilidade $F(t)$ de um sistema é a probabilidade de o sistema fornecer um serviço até ao tempo t.

`Manutenibilidade`: É uma medida que indica o tempo necessário para o sistema recuperar de uma falha não critica.

`Disponibilidade`: Este valor indica a relação entre o tempo em que o sistema está disponível para fornecer um serviço e o seu tempo de funcionamento.

2.2 Sistemas Embutidos Distribuídos

O uso de sistemas embutidos distribuídos tem vindo a crescer constantemente. Estes sistemas têm vindo a ser utilizados em vários sectores tais como a indústria automóvel e aviónica. O aumento da utilização de sistemas embutidos distribuídos deve-se, entre outros fatores, à sua capacidade de partilhar recursos, escalabilidade, facilidade de instalação e manutenção. Com esta capacidade de partilha de recursos, cada equipamento da rede pode trabalhar paralelamente e só partilhar o resultado da sua tarefa. Esta separação torna mais fácil o desenvolvimento do sistema e a sua expansão. Toda a distribuição do sistema fica transparente do lado do utilizador que vai continuar a ver um só sistema. Todavia, esta separação obriga a existência de uma rede de comunicação que interligue todos os dispositivos. Esta rede é fundamental para o bom funcionamento do sistema. Essa rede pode ser implementada com as mais variadas tecnologias, e.g. com recurso a cabos elétricos, sem fios ou fibra óptica.

Atualmente vários sistemas usam equipamentos que requerem grandes quantidades de dados. Para a partilha de recursos funcionar corretamente é necessário existir uma rede que interligue todos os equipamentos do sistema. Essa rede necessita ter uma largura de banda suficientemente grande para suportar o tráfego gerado pelos vários dispositivos. Existem tecnologias de comunicações com capacidade de suportar esse tráfego mas que não são adequadas para sistemas críticos. São tecnologias que têm sido usadas em aplicações que não apresentam requisitos temporais estritos. Todavia, um comportamento temporalmente determinístico é essencial para sistemas críticos. Nesse sentido diversos grupos de investigação têm vindo a propor alternativas aos protocolos tradicionais.

2.3 Redundância

As faltas nem sempre são evitáveis, mas quando ocorrem o sistema como um todo não pode falhar. Uma das soluções para tornar os sistemas tolerantes a falhas é o uso de redundância. O uso de redundância implica usar mais recursos que o necessário com o propósito de manter o funcionamento do sistema durante a ocorrência de faltas. No caso desta dissertação a redundância encontra-se na quantidade de ligações existentes entre os dispositivos de uma rede, mais especificamente numa rede Ethernet. Como existe mais ligações que o necessário, os dispositivos têm vários caminhos para comunicar entre eles. No caso em que ocorre uma falha num caminho, existe outro caminho para o destino desejado.

O uso de redundância pode ser variado. Segundo [16] existem quatro tipos de redundância: diversa, homogénea, ativa e passiva.

- A redundância diversa consiste no uso de diversos elementos com a mesma funcionalidade. Por exemplo, uma mensagem pode ser transmitida por intermédio de texto e de

áudio ao mesmo tempo. Este tipo de redundância tem a vantagem que cada tipo de elemento é tolerante a distúrbios diferentes. No exemplo anterior, a existência de ruído sonoro irá provocar falhas na receção via áudio, mas não irá influenciar a mensagem transmitida por intermédio de texto.

- A redundância homogénea consiste em replicar um elemento. Apesar de ser mais fácil de implementar que a anterior, é mais suscetível de todos os elementos falharem ao mesmo tempo, pois, sendo do mesmo tipo, eles são vulneráveis as mesmas falhas.

- A redundância ativa usa mais elementos que o necessário, mas distribui a carga de trabalho por todos os elementos. No momento da ocorrência de uma falha num dos elementos, a carga nos outros elementos aumenta para conseguir manter o desempenho do sistema. Este tipo de redundância também permite reparar ou substituir um dos elementos com facilidade, uma vez que é possível retirar o elemento sem interromper o funcionamento do sistema. Na figura 2.1 está uma ponte suspensa onde se pode ver vários cabos a suportar a estrutura. Neste tipo de construções é comum usar redundância ativa. A quantidade de cabos é superior ao necessário e podem ser substituídos individualmente sem grande complicação.

- Na redundância passiva o elemento redundante só é usado quando o elemento ativo falha. Os elementos em excesso só são usados para recuperar de uma falha. Um exemplo comum é o uso de um pneu suplente nos carros.

O uso de redundância permite a tolerância a falhas. Apesar de ser frequentemente usada, a redundância implica existir um custo mais elevado ou piores desempenhos utilizando os recursos disponíveis para repetir tarefas, e.g. enviar todas as tramas com réplicas. Mas em sistemas embutidos distribuídos críticos o seu uso é essencial.

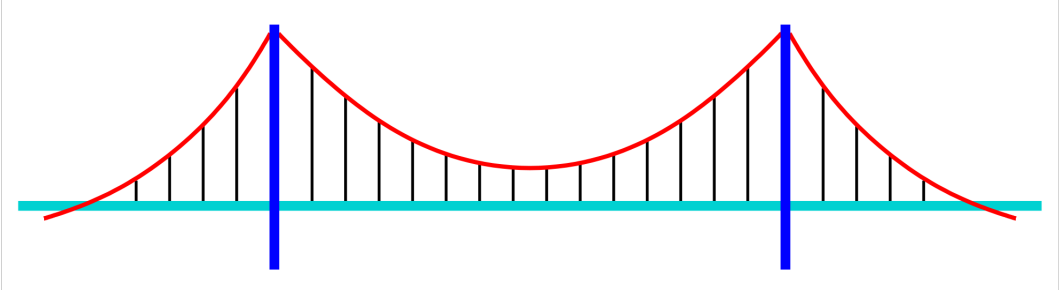

Figura 2.1: Ponte suspensa

2.4 Conceitos sobre Ethernet

A Ethernet é um padrão de comunicação utilizado em redes locais. Esta tecnologia tem vários standards especificados na norma Institute of Electrical and Electronics Engineers (IEEE) 802.3. Desenvolvida inicialmente para interligar uma impressora e um computador, com uma velocidade inicial de 2.94Mbps, esta tecnologia tem sido constantemente melhorada, atingindo-se neste momento velocidades de 10Gbps. Com a sua evolução foram aparecendo várias normas que usam diferentes meios físicos (e.g cabo coaxial, pares de cabo de cobre e fibra óptica) e

velocidades diferentes. Neste momento as normas mais utilizadas são a 100BaseT e a 1000BaseT também conhecidas por *Fast Ethernet* e *Gigabit Ethernet* respetivamente. Estas normas utilizam pares de cabo de cobre e têm velocidades de 100Mbps e 1Gbps. Nesta dissertação foi usada a norma 100BaseT. Inicialmente a interligação entre equipamentos com a tecnologia Ethernet estava estruturada na forma de barramento. Com o aparecimento dos *switches* Ethernet, a topologia da rede já não está limitada a um simples barramento. As principais topologias implementadas numa rede Ethernet são: barramento, anel, estrela, árvore e malha.

Numa topologia de rede do tipo barramento figura 2.2.a) todos os nós estão no mesmo domínio de colisão, pelo que esta topologia pode ser utilizada para pequenas redes. Quando a rede aumenta torna-se difícil todos os equipamentos partilharem um único barramento. Do ponto de vista de tolerância a falhas esta abordagem não é recomendada. Só existe um caminho para cada destino e uma única falha nesse caminho provoca a perda de conectividade entre os terminais.

A topologia de rede em anel figura 2.2.d) permite ligar n dispositivos com n ligações. Nesta abordagem existe sempre dois caminhos, um em cada sentido A possibilidade de enviar uma mensagem num dos dois sentidos permite à rede de suportar uma única falha. Um inconveniente neste tipo de topologias é o envio de tramas para terminais não existentes na rede. Quando esta situação ocorre, se não houver nenhuma gestão da rede, a mensagem entra num ciclo e vai percorrer a rede infinitamente.

A topologia em estrela figura 2.2.b) é composta por um nó central ao qual todos os restantes nós estão ligados. Esse nó central pode ser um *Hub* ou um *switch*. Na topologia em estrela é possível acrescentar nós até a quantidade máxima de nós suportados pelo nó central. Aqui o nó central é muito importante. Qualquer falha que ocorra nele tem efeito no funcionamento da rede. Todo o tráfego passa por ele. Nos restantes nós, caso exista um problema no nó ou na sua ligação ele só irá afetar as mensagem que sejam direcionadas para esse nó.

A topologia em árvore figura 2.2.c) é outra alternativa existente. O seu funcionamento é parecido com a topologia em estrela mas com várias redes em estrela interligadas entre elas. A sua estrutura será hierárquica, existindo um ramo principal do qual vão aparecendo novos ramos e assim sucessivamente. A importância dos dispositivos, para o bom funcionamento da rede, varia consoante a sua posição na árvore. Os dispositivos nas extremidades da árvore, caso falhem, só interferem nas mensagem nas quais eles são os destinatários, enquanto que nos nós internos da árvore existem mensagens destinadas para outros destinatários que os atravessam. Consoante a sua posição na árvore o nó interno pode ter vários nós a necessitar que ele funcione corretamente. Nesta topologia é possível existir redundância nas ligações. Para isso basta interligar ramos mas para que a rede continue a ter um bom funcionamento é necessário existir um gestão adequada. Sem essa gestão poderão criar-se ciclos internos e mensagens a fluir na rede infinitamente, as quais prejudicam o desempenho da rede podendo mesmo levar ao seu colapso.

Numa topologia em malha figura 2.2.e) todos os dispositivos da rede tem uma ligação direta entre si. O número de caminhos possíveis entre os dispositivos é elevado e o número de ligações aumenta exponencialmente com o número de nós ligados à rede. Esta abordagem fornece uma grande robustez mas a complexidade da gestão da rede também é grande.

Como anteriormente referido, para um bom funcionamento é necessário existir uma gestão adequada da rede. Para fazer essa gestão existem vários protocolos que regulam o seu funcionamento. Esses protocolos são depois implementados pelos dispositivos presentes na rede, e.g os *switches* que interligam os dispositivos.

A informação partilhada pela rede é encapsulada em tramas predefinidas para a tecnologia

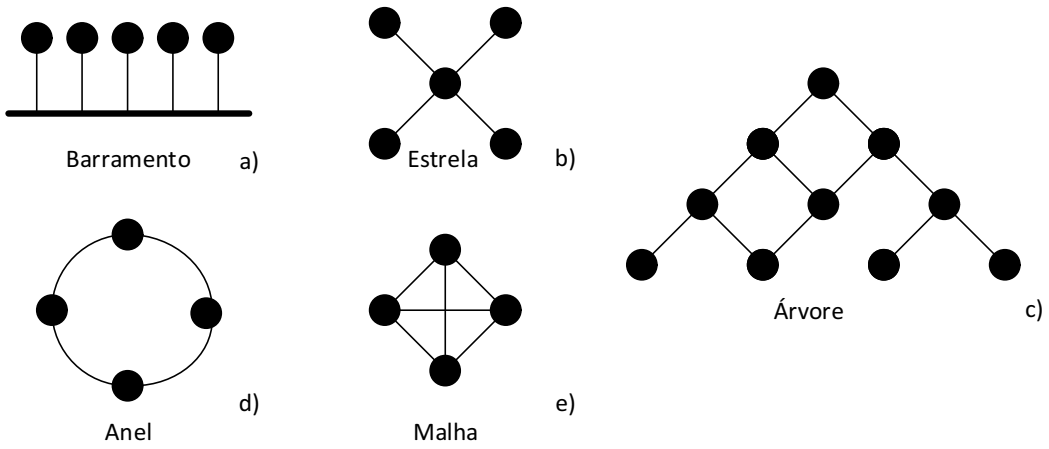

Figura 2.2: Topologias de Rede

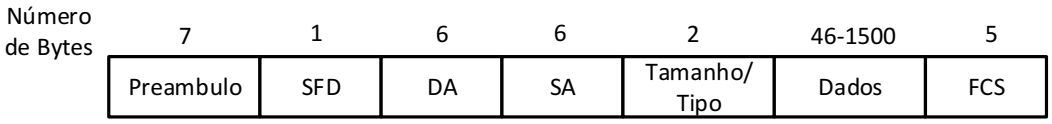

Figura 2.3: Formato trama ethernet

usada. No caso da Ethernet o formato da trama tem vários campos, representados na figura 2.3.

- O preâmbulo é constituído por sete bytes com o valor 0x55 seguido de um *Start Frame Delimiter* com o valor 0xD5. Estes dois campos tem como finalidade indicar o início do envio de uma trama e são usados para sincronização.
- O endereço Medium Access Control (MAC) é um endereço físico único atribuído à interface de um dispositivo à rede. Este endereço é constituído por 6 bytes. Na trama Ethernet o campo DA (Destination Address) contém o endereço MAC do dispositivo para o qual a trama é enviada. O campo SA (Source Address) tem o valor do endereço MAC do dispositivo que enviou a trama.
- O campo seguinte pode indicar o tipo ou o seu tamanho, conforme definido no IEEE Std 802.3-2005. Quando o valor do campo for inferior ao valor decimal 1500 ele irá indicar o tamanho dos dados da trama. Caso seja igual ou superior a 1536 o seu conteúdo irá indicar o tipo da trama Ethernet. Por exemplo no caso do valor do campo ser 33024 (0x8100), o campo indica uma trama do tipo Virtual Local Area Network (VLAN).
- O campo de dados contém a informação transmitida. Este campo tem tipicamente um tamanho entre 0 e 1500 bytes. Em certas tramas este campo pode ter tamanhos superiores, nesses casos as tramas são conhecidas por *Jumbo Frames*. No caso do tamanho deste campo ser inferior a 46 bytes são adicionados bytes com o valor 0x00 até completar o tamanho mínimo, este processo é conhecido por *padding*.
- O ultimo campo da trama é o Frame Check Sequence (FCS). É o resultado proveniente de um algoritmo de *Cyclic Redundancy Check* que usa todos os bits da trama. Caso o

resultado obtido no recetor não coincida com o campo FCS presente na trama, houve um erro na transmissão da trama e ela é considerada corrompida.

A Ethernet continua a ser uma tecnologia em evolução. O seu baixo custo e bons desempenhos fazem dela uma tecnologia interessante para se aplicar em diversas áreas emergentes. Uma dessas áreas é o uso de Ethernet para as comunicações dos dispositivos dentro de um carro. Com o uso de câmeras de vídeo nos carros a largura de banda necessária aumentou, tornando os meios de comunicação usados até agora (os barramentos de de campo) obsoletos nesse sentido. A largura de banda da Ethernet e o seu custo tornam na uma candidata interessante para substituir as tecnologias usadas até agora. Mas a Ethernet não foi desenvolvida inicialmente para sistemas críticos. Nos últimos tempos vários grupos de investigação têm vindo a propor protocolos que tornam a Ethernet adequada para sistemas críticos.

2.5 Protocolos Ethernet

Como foi possível ver na secção anterior, as topologias de uma rede podem ser complexas. Essa complexidade dificulta a gestão da rede. Com o propósito de simplificar a gestão da rede foram desenvolvidos vários protocolos. São esses protocolos que vão ditar as regras do funcionamento da rede. Cada protocolo é desenvolvido consoante a tecnologia e a finalidade para qual a rede vai ser usada. No caso deste trabalho o foco está nos protocolos que funcionem sobre Ethernet e que permitam recuperar de falhas com base na redundância. Os protocolos que se seguem possuem as características desejadas.

2.5.1 Rapid Spanning Tree Protocol

O RSTP [3] surgiu como a evolução do Spanning Tree Protocol. Estes protocolos têm funcionalidades bem definidas e a sua utilização em redes permite resolver um problema que aparece aquando da existência de redundância na rede que são os ciclos internos. Quando existe redundância o tráfego tem vários caminhos possíveis e pode ocorrer uma trama entrar num ciclo, isto é, repetir um troço de caminho indefinidamente sem nunca chegar ao destino, ficando assim a ocupar largura de banda de forma permanente. Nesse sentido foram desenvolvidos protocolos que definem caminhos únicos entre os equipamentos da rede.
O RSTP define vários estados para as portas do *switch*. Essa decisão é tomada numa fase de aprendizagem, na qual cada *bridge* envia tramas que contêm a informação necessária para as decisões. Estas tramas são conhecidas por Bridge Protocol Data Units (BPDU). Nesse processo de aprendizagem a *Bridge* com menor ID irá ser considera de Root. As restantes *bridges* vão definir um custo para cada porta e vão tentar encontrar o caminho com menor custo para a *Bridge* root.

Depois da fase de aprendizagem a rede converge e todas as *bridges* sabem em que porta podem comunicar com a *bridge* root. Nesse momento cada *bridge* vai atribuir às suas portas um estado. O estado da porta pode ser de *root* no caso de ser a porta usada para comunicar com o root (exceto a própria *bridge* root). Outro estado possível é *designada*, no qual a porta é usada por outra *bridge* para comunicar com o root. Existem dois estados nos quais a porta está "desativa", o estado *alternativo* e o estado *Backup*. No estado alternativo a porta é uma alternativa para o caminho com menor custo para a *bridge* root, enquanto que no caso do *backup* é uma porta que está interligada ao mesmo equipamento que outra porta da *bridge*. Apesar de conseguir a recuperação de faltas, o espaço de tempo até a convergência é de várias

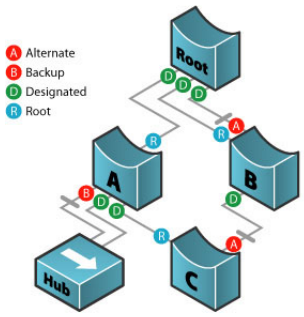

Figura 2.4: Rede com Rapid Spanning Tree Protocol

centenas de milissegundos até alguns segundos [14], valor que é inaceitável para a maioria dos sistemas de tempo real.

2.5.2 Rapid Ring Recovery

Este protocolo foi desenvolvido com objetivo de melhorar o tempo de recuperação de uma falha dentro de uma rede. Nos sistemas de tempo real esse tempo de recuperação tem que ser o menor possível. Foi nesse sentido que este protocolo foi desenvolvido, sendo destinado a redes com topologia em anel e apresentando tempos de recuperação da ordem dos 300 microssegundos. Segundo [11] este protocolo é baseado na utilização de vários anéis virtuais. A característica principal do RRR é que ele incorpora um método de deteção de falhas descentralizado. Só fazem parte da deteção da falha os nós adjacentes à ligação que falhou. Isto permite que haja uma reação da rede muito mais rápida, porque não é necessário esperar que todos os equipamentos pertencentes à rede tenham conhecimento da falha para que ocorra a recuperação. Um problema nas redes com redundância, como por exemplo a topologia em anel, é quando é introduzido uma trama com um endereço MAC desconhecido. Quando isso acontece essa trama pode ficar indefinidamente a fluir na rede. Nos casos em que sejam introduzidas várias tramas com endereços desconhecidos o desempenho da rede vai ser comprometido. No RRR para evitar que isso aconteça as tramas são enviadas através de uma VLAN. Essa VLAN interliga todos os elementos da rede mas sem fechar o anel. Isso implica existirem n VLANs para n dispositivos. Na figura 2.5 estão quatro equipamentos na rede, logo vão existir quatro ligações e respetivamente quatro VLAN. Como os anéis não são fechados já não temos o risco de ter tramas a percorrer caminhos em ciclo infinito, porque só existe um caminho para cada destino. Assim existe sempre uma VLAN na qual uma determinada ligação não é utilizada. No caso de aparecer uma falha nalguma ligação basta utilizar a VLAN na qual essa ligação não é utilizada. Como o dispositivo desenvolvido no âmbito desta dissertação implementa este protocolo é feita uma descrição mais pormenorizada do seu funcionamento no capítulo 3.

2.5.3 IEC 62439

O RRR apresentado anteriormente tem tempos de recuperação de falhas na ordem das centenas de microssegundos. Mas nalguns casos existem sistemas de tempo real críticos onde é necessário ter tempo de recuperação nulos. Para ser possível ter tempos de recuperação nulos é necessário utilizar vários caminhos ao mesmo tempo. Assim se existir uma falha nalgum dos caminhos a trama continuara a chegar ao destino. Nesse sentido foi desenvolvido o IEC

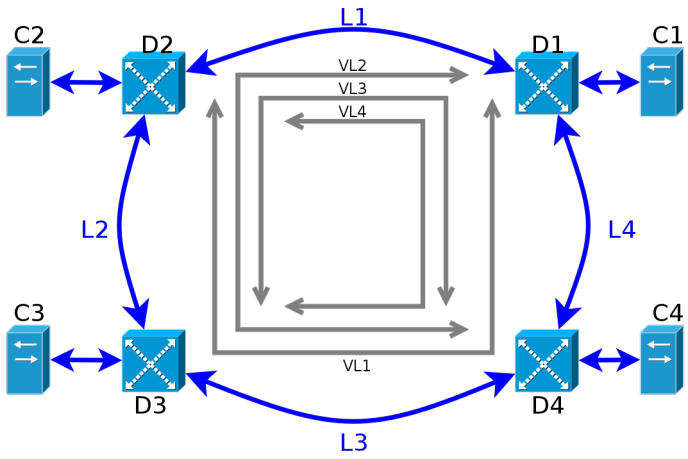

Figura 2.5: Rede com 4 dispositivos

62439.

Parallel Redundancy Protocol (PRP)

Neste standard toda a rede é duplicada. Cada nó tem que ter duas portas Ethernet, cada uma ligada a uma rede. Cada trama que é enviada por um nó é duplicada e é transmitida em cada uma das redes. Na figura 2.6 é apresentado uma figura onde existem 4 nós. Todos os nós estão interligados entre si através de duas redes, a LAN A e a LAN B, que são idênticas.

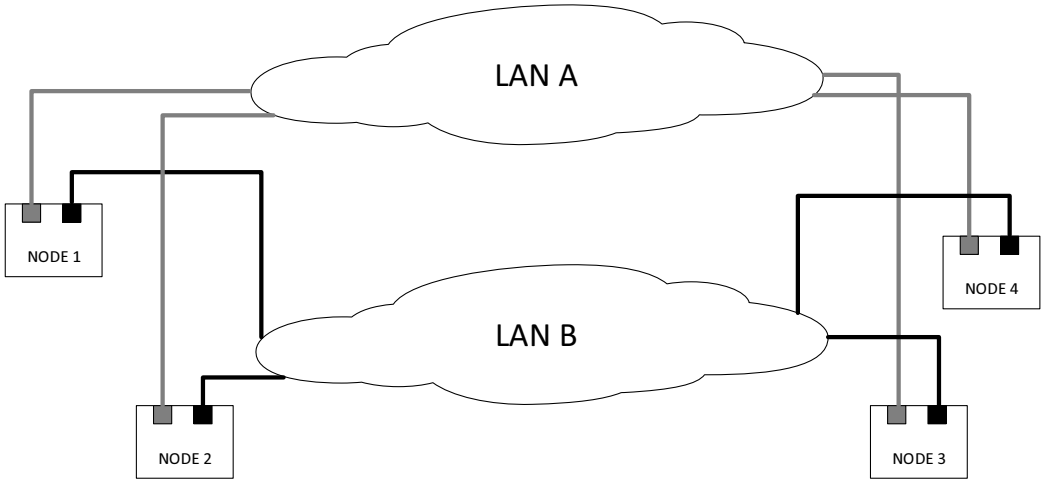

Figura 2.6: Rede duplicada

Para implementar esta técnica é necessário o nó ser capaz de duplicar a trama e na receção também tem que ser capaz de descartar a trama duplicada no caso em que as duas tramas são recebidas. Na figura 2.7 está a estrutura interna de um nó. Nesta figura é visível que são necessárias duas interfaces de rede. Para ser capaz de detetar se uma trama já foi recebida é inserido um *Sequence Number* na trama. Do lado do emissor existe uma tabela onde, para cada endereço MAC, individual, *multicast ou broadcast* existe o valor do *Sequence Number* da

última trama enviada para esse endereço. Do lado do recetor existe outra tabela que irá conter os *Sequence Numbers* das tramas recebidas de cada endereço. Com essas tabelas o recetor é capaz de saber quando é necessário descartar a trama [1].

Esta abordagem tem um bom desempenho em termos de fiabilidade, sendo capaz de suportar uma falha numa das redes e continuar a funcionar corretamente e sem a ocorrência de qualquer atraso. Mas, em contrapartida, os custos necessários para duplicar a rede podem tornar inviável esta solução.

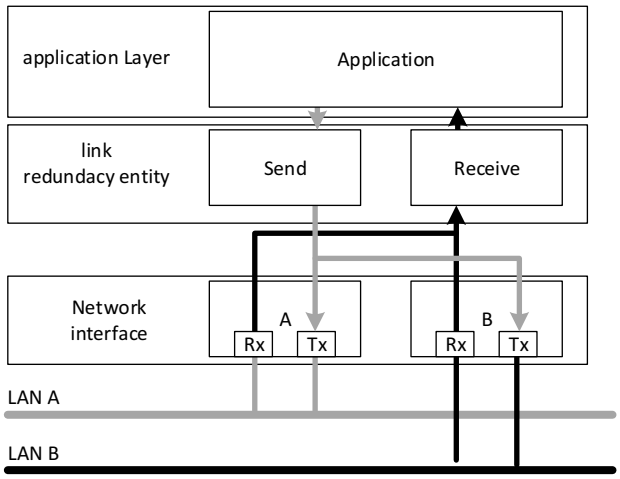

Figura 2.7: Estrutura do nó

Highly Available Seamless Ring (HSR)

A solução anterior apesar de fornecer tempos de recuperação nulos, apresentava um custo elevado. O HSR utiliza os mesmos nós que o PRP e não necessita de duplicar a rede. A ideia continua a mesma, enviar duas tramas iguais, uma em cada porto, mas nesta abordagem os nós estão todos interligados por uma rede em anel. O funcionamento do nó continua muito parecido com o anterior mantendo o mecanismo de adicionar um *Sequence Number* nas tramas enviadas. Na receção de uma trama, o nó verifica se a trama lhe é destinado e caso contrário a trama é reintroduzido anel. Esta abordagem implica ter uma implementação na qual o nó é capaz de fazer a verificação da trama e o seu reencaminhamento no menor tempo possível. O tempo de atraso da trama é proporcional ao número de nós pelos quais a trama passa. Caso o tempo de atraso em cada nó seja elevado, pode inviabilizar a solução tornando o sistema muito lento. Uma implementação em *hardware* é aconselhável [2].

Na figura 2.8 estão 5 nós interligados em anel. A aplicação do nó 1 envia uma trama *multicast* para os nós 2 e 4. A interface de rede duplica a trama e envia cada trama num sentido (seta preta e cinzenta). Cada nó recebe a trama e verifica se a trama lhe é destinada. Caso seja, reencaminha a trama para a camada aplicação. Caso contrario, volta a introduzir a trama no anel. Neste exemplo, como é uma trama *multicast*, esta volta a ser introduzida no anel. Quando as tramas voltam ao nó inicial o nó descarta a trama, retirando-a do anel. Esta abordagem, tal como o PRP, tem tempos de recuperação nulos, e não implica duplicar toda a rede. Por outro lado tem como desvantagem o facto de estar limitado a uma topologia em anel.

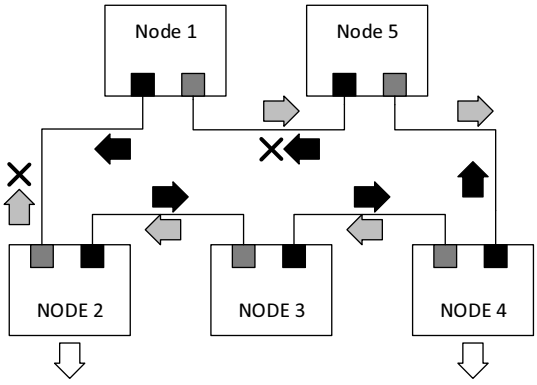

Figura 2.8: Envio de trama *multicast*

2.5.4 Transparent Interconnection of Lots of Links (TRILL)

Este procolo surge como potencial substituto do Spanning Tree Protocol (STP). Apesar de corrigir certos problemas o STP tem bastantes inconvenientes como bloquear alguns portos e assim reduzir o número de caminhos possíveis, além de concentrar o tráfego todo nalgumas ligações. O TRILL foi concebido para ter as vantagens dos *routers* e dos *switches*. Os dispositivos que implementem o TRILL podem funcionar numa rede com outro protocolo de maneira transparente para a rede. Essa capacidade possibilita que haja uma substituição progressiva dos *switches* atuais pelos novos que implementam o TRILL, os quais são conhecidos por *Rbridges*. Segue um resumo poético do protocolo retirado de [6]

Algorhyme V2, by Ray Perlner

I hope that we shall one day see
A graph more lovely than a tree.

A graph to boost efficiency
While still configuration-free.

A network where RBrigdes can
Route packets to their target LAN.

The paths they find, to our eletion,
Are least cost paths to destination!

With packet hop counts we now see,
The network need not be loop-free!

RBridges work transparently,
Without a common spanning tree.

No STP o bloqueio dos portos permite de eliminar ciclos internos na rede, sendo em contrapartida inutilizados recursos. Com o TRILL, as RBridges não bloqueiam portos para evitar ciclos, sendo em vez disso inserido um *hop count* que vai ser decrementado cada vez que as tramas passam por uma RBridge. Esse campo é inserido no cabeçalho da trama TRILL. Esta é uma abordagem já utilizada em vários protocolos, tais como o Internet Protocol, que tem um campo *Time To Live*. O valor do *hop count* tem que ser inicializado pela RBridge que vai inserir a trama na rede. O valor desse campo é a previsão de quantas RBridges serão atravessadas até a trama chegar ao destino. O cabeçalho TRILL também contém o endereço MAC da Rbridge que introduziu a trama na rede e o endereço destino da Rbridge de saída. As RBridges não se limitam a decrementar o Hop Count, elas também alteram o endereço destino do cabeçalho da trama Ethernet. No caso de ser uma trama unicast, a Rbridge troca o endereço destino da trama Ethernet para o endereço da próxima RBridge no caminho para a Rbridge de saída. Quando chegar à Rbridge de saída esta irá trocar o endereço destino da trama para o endereço do dispositivo para o qual a trama é destinada. A próxima Rbridge é escolhida consoante um caminho planeado. Para planear o caminho é necessário as RBridges terem informações sobre a rede recolhidas com a troca de tramas. Esse conhecimento sobre a rede permite à RBridge detetar vários caminhos para o mesmo destino. Assim é possível utilizar a redundância de caminhos para aumentar a largura de banda ao enviar tramas em cada um desses caminhos ou aumentar a tolerância a falhas ao enviar a mesma informação em mais que um caminho.

2.5.5 Interligação de Equipamentos

Algumas destas abordagens implicam que o nó tenha dois portos Ethernet para enviar tramas. Mas muitos dos equipamentos usados não tem essa característica. Uma maneira de contornar esse problema consiste em desenvolver um equipamento com várias portas Ehternet que é utilizado como interface de rede. Esse equipamento irá implementar toda a lógica de reencaminhamento das tramas estipulado pelo protocolo utilizado. Estas interfaces denominam-se usualmente por RedBox. Assim é possível interligar qualquer dispositivo a uma rede com um dos protocolos anteriores sem nenhuma complexidade.

2.5.6 Discussão sobre os protocolos

Todos os protocolos referenciados anteriormente têm vantagens e desvantagens. A escolha do protocolo a utilizar vai depender sempre da finalidade do sistema no qual ele é inserido. Temos o RSTP que suporta qualquer tipo de rede. Tem uma abordagem fácil de entender e é muito usado em sistemas não críticos. Apesar da sua simplicidade, este protocolo desperdiça recursos. Por exemplo quando existem vários caminhos entre dois dispositivos só um é que é usado, quando os outros poderiam ser aproveitados para aumentar a largura de banda entre os dispositivos. Outro inconveniente deste protocolo é o seu tempo de recuperação, na ordem das centenas de milissegundos que torna este protocolo não utilizável em sistemas críticos.

O RRR é uma proposta recente. Este protocolo tem como principal vantagem o seu tempo de recuperação, uma vez que consegue recuperar de falhas em menos de 1 milissegundo. A sua utilização está limitada a uma topologia em anel, permitindo acrescentar nós a rede sem grande complexidade. Mas ao só suportar redes em anel limita a sua utilização.

O PRP é uma boa abordagem para sistemas críticos. O seu tempo de recuperação nulo torna esta solução apetecível para sistemas críticos. Para obter esse tempo de recuperação é

necessário duplicar toda a infraestrutura da rede. Esta necessidade torna os sistemas que usam esta solução, muito dispendiosos e eventualmente inviáveis. O HSR também tem tempos de recuperação nulos mas sem a duplicação da rede e os custos da duplicação. Tal como o RRR este protocolo também está limitado a redes em anel. Outro inconveniente desta abordagem é o envio da mesma trama nos dois sentidos, reduzindo assim a largura de banda disponível na rede.

O último protocolo exposto, denominado por TRILL, está ainda em desenvolvimento. Possível substituto do STP nas redes locais, este protocolo consegue gerir a redundância presente na rede e permite, ao invés do STP, usar essa redundância para aumentar a largura de banda ou em sistemas críticos de prevenir falhas ao enviar a mesma trama em vários caminhos. Suportando qualquer tipo de topologia a complexidade de implementação deste protocolo é maior.

Como foi referenciado no inicio a escolha do protocolo depende da finalidade do sistema onde é inserido e dos requisitos do sistema.

No próximo capítulo é apresentado com mais detalhe o protocolo Rapid Ring Recovery que foi o escolhido para este trabalho.

Capítulo 3

Rapid Ring Recovery

Com a expansão da Ethernet para sectores críticos, foram desenvolvidos vários protocolos com intuito de aumentar a sua fiabilidade. Uma recente proposta, conhecida por RRR, propõe tempos de recuperação de falhas numa ligação inferiores a 1 milissegundo. O seu funcionamento está baseado no uso inteligente de múltiplas VLANs e possui um mecanismo de deteção de falhas descentralizado. Os nós adjacentes à falha são os únicos que fazem parte do processo de deteção e de recuperação. Estas características e os resultados apresentados na proposta inicial tornam este protocolo aliciante para ser usado como base para futuros trabalhos.

3.1 Introdução

A Ethernet é sem dúvida a tecnologia mais usada em redes locais. Recentemente o seu uso em sistemas de tempo real tem vindo a generalizar-se. Apesar da Ethernet ter sido inicialmente projetada para transmissões de grandes quantidades de dados sem restrições temporais, o que a tornava inadequada para sistemas de tempo real, a sua simplicidade, baixo custo e largura de banda fizeram com que começasse a existir interesse em usar a Ethernet em sistemas de tempo real. Como os sistemas de comunicação usados em sistemas de tempo real críticos devem ser tolerantes a faltas é necessário um ajuste ao uso da tecnologia Ethernet. Nesse sentido têm vindo a aparecer várias abordagens, sendo uma delas o RRR.

Este protocolo tem como vantagem principal o facto de detetar e recuperar de uma falha com tempos muito baixos, estando reportados tempos de recuperação na ordem dos 294 microssegundos [10]. O RRR usa um método de deteção de falhas descentralizado. Só os nós adjacentes à ligação que falhou é que fazem parte do processo de deteção da falha. Assim na ocorrência de uma falha o comportamento mantém-se, exceto nos nós adjacentes. É nesses nós que vai ser inicializado o processo de recuperação. O RRR também tem como característica fazer um uso inteligente de VLANs. Um problema das redes com topologia em anel é quando é introduzido uma trama com um endereço destino desconhecido. Nesse tipo de situações as tramas ficam indefinidamente a fluir no anel, o que prejudica o desempenho da rede podendo levar ao seu colapso. No RRR para evitar que isso aconteça as tramas são enviadas através de VLANs. Cada uma dessas VLANs interliga todos os elementos da rede mas sem fechar o anel o que resolve o problema. Isto implica existir n VLANs para n dispositivos.

Na figura 3.1 estão 4 equipamentos na rede, logo vão existir 4 ligações e respetivamente 4 VLANs. Como as VLANs são anéis que não são fechados já não temos o risco de ter tramas a percorrer caminhos infinitamente, porque só existe um caminho para cada destino.

Também se pode verificar que existe sempre uma VLAN na qual uma determinada ligação não é utilizada. No caso de aparecer uma falha nalguma ligação basta utilizar uma outra VLAN na qual essa ligação não seja utilizada. Por exemplo, no caso de ocorrer uma falha se na ligação L4 basta utilizar a VLAN 1 para a falha não interferir com as comunicações.

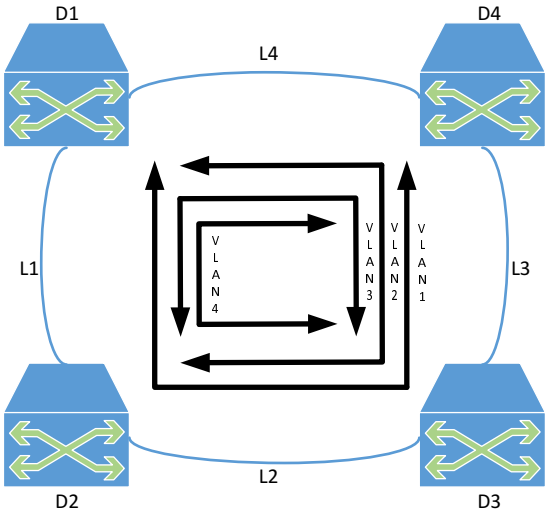

Figura 3.1: Rede com 4 dispositivos

3.2 Trama

Como o funcionamento nesta rede difere do habitual, os cabeçalhos das tramas Ethernet tradicionais não são adequados para este protocolo. Assim foi introduzido um cabeçalho com as informações necessárias para que cada nodo saiba como e para onde encaminhar a trama. A trama usada na proposta original tem o formato das tramas da norma IEEE 802.1ah. Esta trama tem vários campos que foram adicionados à trama original. Esta alteração pode ser vista na figura 3.2.

- O campo *Backbone destination address* (B-DA) contém o endereço MAC do nó no anel para o qual a trama se destina.
- O campo *Backbone source address* (B-SA) contém o endereço MAC do nó que introduziu a trama no anel.
- O campo B-TAG é constituído pelo *ethertype* com valor de 0x88A8 e o backbone VLAN indicator.
- O I-TAG tem um *ethertype* de 0x88E8 e o I-SID é o *service identifier* onde é sinalizado o bit de túnel que indica se a trama está a ser recuperada. Também é no I-SID que é indicado em que VLAN a trama está a ser transmitida.
- O *Customer destination address* (C-DA) é o endereço do equipamento para o qual a trama se destina enquanto que o *Customer source address* é o endereço do equipamento que enviou a trama.

- O campo C-TAG é composto pelo ethertype 0x8100 e o o VLAN ID.
- O ethertype vai ter o valor definido na trama original tal como o payload.

É possivel verificar que os campos da trama original são mantidos, só foram acrescentados campos aos originais.

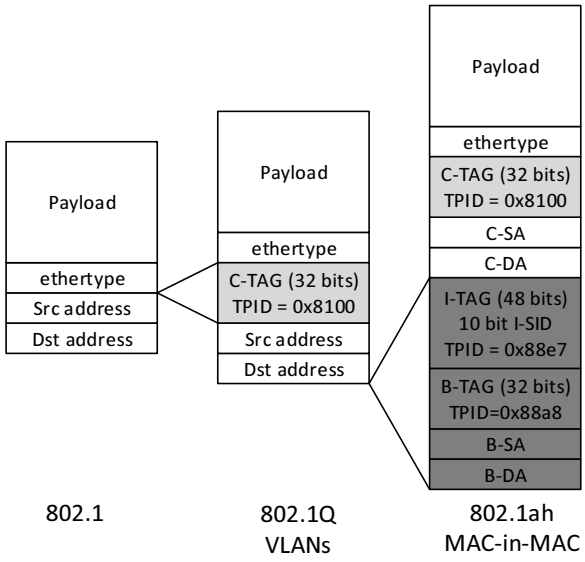

Figura 3.2: Formato de uma trama definida pela norma 802.1ah

3.3 Funcionamento sem Falhas

O funcionamento na ausência de falhas dos nós com o RRR implementado é simples. Cada nó da rede tem um ID definido durante a configuração da rede. Cada nó tem também uma tabela de encaminhamento na qual armazena o endereço MAC do dispositivo que enviou a trama e associa esse endereço à porta em que recebeu a trama. Também associa o endereço com a VLAN na qual a trama está a ser transmitida. Depois de receber uma trama, consoante o endereço destino reencaminha a trama para o porto correto. No momento de enviar uma trama, o nó verifica se o endereço do dispositivo para o qual a trama é enviada existe na tabela de encaminhamento. No caso de existir, a trama é enviada pela porta que está associada a esse endereço e o envio é realizado na VLAN, também ela associada a esse endereço. Quando o endereço não está presente na tabela, o nó envia a trama pela VLAN predefinida que é aquela que tem o valor igual ao ID do nó. No caso de ser o nó com o ID 1 ele envia a trama na VLAN 1. No caso do destino ser desconhecido e a trama ser proveniente do anel, o envio é feito simultaneamente para o porto local e para o anel.

3.4 Recuperação de uma falha

Quando ocorre uma falha numa ligação, o comportamento dos nós adjacentes à falha altera-se. Ao receber uma trama proveniente dos portos locais, o nó introduz a trama no

anel pelo porto cuja ligação não falhou e na VLAN que não utilize essa ligação. No caso da trama ser proveniente do anel, o *switch* verifica se conhece o endereço destino na tabela de encaminhamento. Caso conheça e seja um dos portos locais, a trama é enviada para esse porto. Se for um porto do anel que use a ligação que falhou, a trama é reenviada para trás com o bit túnel alterado para '1' para indicar que a trama encontrou um erro pelo caminho e está a voltar para trás. Durante o regresso da trama à origem os nós não atualizam a tabela de encaminhamento e o valor da VLAN usada para a transmissão da trama é alterada para a VLAN mais adequada. Os nós ao receberem uma trama com o bit de túnel ativo, vão verificar quem introduziu a trama no anel. Caso o endereço corresponda ao seu, o nó vai repor o bit de túnel a '0' e encaminhar a trama para a outra porta ligada ao anel. Se o endereço não coincidir com o seu, o nó limita-se a enviar a trama para o outro porto do anel.

No caso do endereço MAC destino da trama não estar presente na tabela de encaminhamento, a trama vai ser reenviada para trás mas também vai ser enviada para os portos locais.

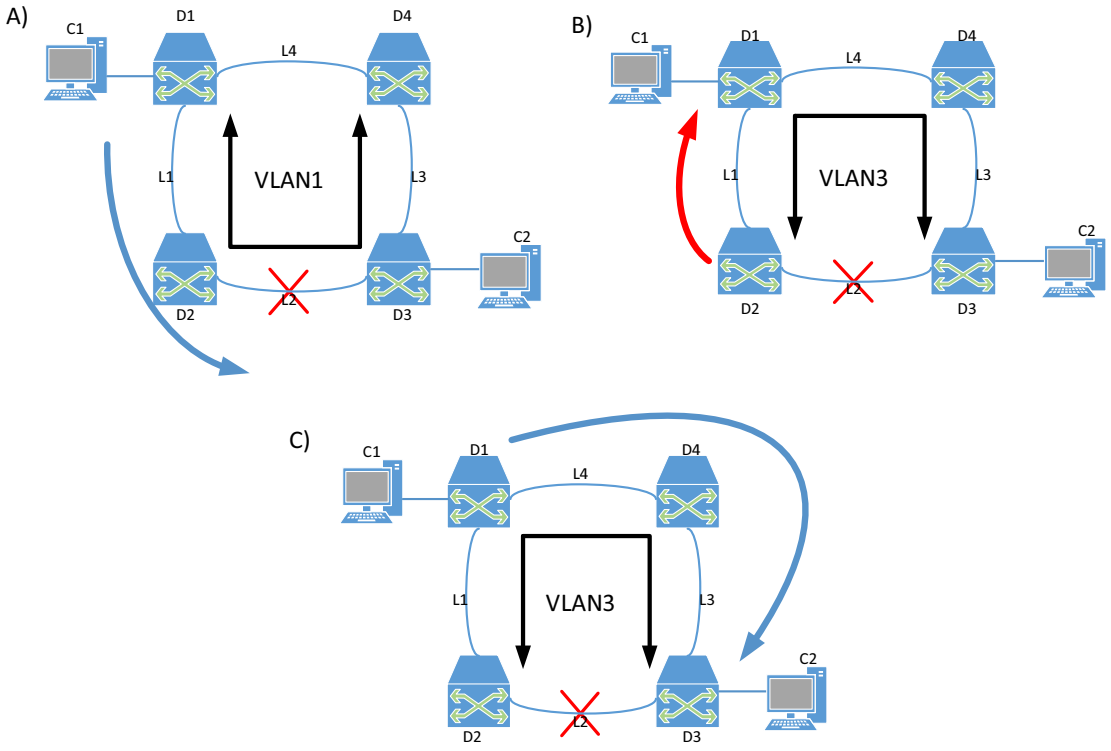

Figura 3.3: Recuperação de uma falha

3.5 Exemplo da recuperação de uma falha

Para exemplificar melhor segue-se um exemplo da recuperação de uma falha numa ligação. Na figura 3.3 estão representados quatro nós e em dois deles estão dois computadores ligados. Quando o computador C1 enviar uma trama para o computador C2, o nó D1 vai adicionar campos à trama para ficar com o formato adequado e vai usar a VLAN 1 para enviar a trama pela ligação L1. A trama vai ser recebida pelo nó D2, mas como existe uma falha na ligação

L2 a trama vai ser reenvidada para trás com o bit de túnel igual a '1'. O nó D2 também vai alterar a VLAN usada para a VLAN3, pois a ligação L2 não é usada nesta VLAN e assim a falha não afetará o envio da trama. Quando a trama voltar ao nó D1 o nó vai detetar que foi ele que introduziu a trama no anel e que ela encontrou uma falha pelo caminho. O nó vai então repor o bit de túnel da trama a '0' e reintroduz a trama no anel mas na outra direção e na VLAN definida pelo nó D2. A trama vai agora para o nó D4 que volta a enviar a trama pela ligação L4 e chega ao nó D3 que vai reencaminhá-la para o destino final que é o computador C2. Todas as tramas que o computador C2 mandar para o computador C1 vão percorrer o mesmo caminho. Mas no caso do computador C2 responder, como as tabelas de encaminhamento são atualizadas, as tramas já vão ser enviadas pelo caminho que não tem falha.Depois deste processo de recuperação todas as tramas enviadas entre C1 e C2 usam o mesmo caminho e já não tentam comunicar através da ligação L2.

O RRR usa um mecanismo baseado no envio de tramas periódicas para detetar uma falha numa ligação . Existe um temporizador para cada porto Ethernet e quando não for recebida nenhuma trama durante um tempo predefinido considera-se que a ligação não está a funcionar. Para que este método funcione corretamente é necessário que haja tráfego suficiente para que haja a receção de tramas antes do temporizador chegar ao fim. Como nem sempre existe o tráfego desejado, é necessário um temporizador com um tempo inferior ao anterior. Esse novo temporizador vai ser usado para despertar o envio de tramas *keep alive*. Essas tramas têm como único propósito manter a ligação ativa. O tempo predefinido no primeiro temporizador vai estabelecer qual é o tempo máximo para o nó detetar uma falha e proceder ao algoritmo de recuperação. No artigo [10] esse tempo está definido com 225 microssegundos. Consoante a aplicação este tempo pode ser parametrizável. As tramas de *keep alive* são usadas no caso de não haver tráfego suficiente, garantindo-se assim uma latência máxima na deteção de falhas de *links*. Caso haja tráfego suficiente, não é necessário o envio de tais tramas e consequentemente o desempenho da rede não é prejudicado.

3.6 Conclusões

O RRR é mais uma abordagem disponível para tornar a tecnologia Ethernet mais fiável e permitir a sua utilização em sistemas críticos. O RRR consegue detetar falhas e recuperar em menos de um milissegundo, e consegue isso sem desperdiçar recursos com o envio de tramas duplicadas. A rede é fácil de configurar sendo possível adicionar capacidade de auto-configuração à rede. O RRR está limitado a redes com topologia em anel o que pode tornar esta implementação impossível em certos sistemas. A implementação descrita no artigo [10] mostra que os dispositivos que implementam o RRR são baseados num *switch* comercial para a ligação a rede em anel e uma placa de desenvolvimento com FPGA para fazer o encaminhamento da trama. Esta implementação implica que o formato da trama tenha um formato standard, sendo o formato escolhido o mesmo que o usado no 802.1ah. Caso a rede em anel seja unicamente composta por equipamentos com RRR seria interessante retirar alguns campos desnecessários existentes na trama usada, reduzindo assim o tamanho da trama. Em vez de utilizar os *switches* comerciais, seria interessante implementar tudo num único dispositivo baseado numa FPGA. Assim com algumas alterações seria possível criar uma rede que seria utilizada como um "switch distribuído", sendo cada nó do anel uma entrada desse novo *switch*.

Capítulo 4

Implementação em FPGA

Neste capítulo é discutido o desenvolvimento de um nó que funcione consoante o protocolo Rapid Ring Recovery. A sua implementação é baseada na tecnologia FPGA. Existem vários blocos, cada um deles com uma função bem definida, tal como o "bloco"PHY e MAC responsáveis por receber e enviar as tramas Ethernet, bem como outros blocos que implementam o algoritmo do RRR. Esta implementação permite criar uma rede que seja constituída com vários destes nós, sendo todavia necessário configurar precisamente a rede para permitir o seu bom funcionamento.

4.1 Visão Global

Nesta dissertação procura-se desenvolver um equipamento que permita interligar vários dispositivos através de Ethernet. Nos capítulos anteriores foi definido o protocolo que iria ser implementado neste equipamento que é o Rapid Ring Recovery. A finalidade deste trabalho é de ter uma rede cujos nós sejam os equipamentos desenvolvidos e a funcionar com o RRR. Os nós são projetados para serem usados como *Redundancy Box* (RedBox). A RedBox vai permitir criar uma rede em anel e oferecer uma interface standard para dispositivos que não sejam compatíveis com RRR.

4.2 Arquitectura do Sistema

A implementação da RedBox foi baseada em placas de desenvolvimento, cuja unidade principal é uma FPGA. O uso de uma FPGA permite uma implementação em hardware personalizável e com prototipagem rápida.

Na Figura 4.1 está uma visão global do sistema. Na figura é possível ver um PHY Ethernet para cada porto. Estes PHYs são produtos comerciais e estão ligados, cada um a um MAC. Os MACs são instanciados na FPGA. Cada porto irá assim ter um PHY e um MAC responsáveis pelo envio e receção das tramas Ethernet. O formato das tramas enviadas pelos equipamentos que estão ligados à rede não são compatíveis com as que são trocadas dentro da rede. Para colmatar esse problema foram adicionados dois blocos a frente do MAC presente no porto local. Esses blocos vão ser usados para adicionar campos às tramas que ingressam na rede e para retirar esses mesmos campos para as tramas provenientes da rede. Assim todas as tramas que chegam ao multiplexer de entrada tem o mesmo formato. O multiplexer de entrada vai ser o elemento do sistema que vai decidir de que porto a trama vai ser recebida. O bloco "Rapid

Ring Recovery"é o bloco que contém toda a lógica necessária para efetuar o encaminhamento das tramas, bem como a recuperação de falhas. O bloco seguinte é o demultiplexer. Este bloco vai fazer o encaminhamento da trama para o porto correto. É também este bloco que vai verificar em que VLAN a trama está a ser transmitida e, caso o porto não seja usado nessa VLAN, descarta a trama. Entre o demultiplexer e os MACs dos portos dedicados ao anel está um bloco *keep alive sender*. O mecanismo de deteção de falhas do protocolo utilizado implica que seja necessário receber tramas dentro de um intervalo de tempo definido. Caso não seja recebida nenhuma trama a ligação é considerada inativa. Para evitar que uma ligação seja considerada inativa na ausência de tráfego é necessário o envio de tramas *keep alive*. Este bloco tem como função monitorizar o envio de tramas e caso seja necessário enviar as tais tramas de *keep alive*. Todos estes blocos são interligados com barramentos de dados e de controlo, sendo necessário sincronizar os blocos com o mesmo domínio de relógio.

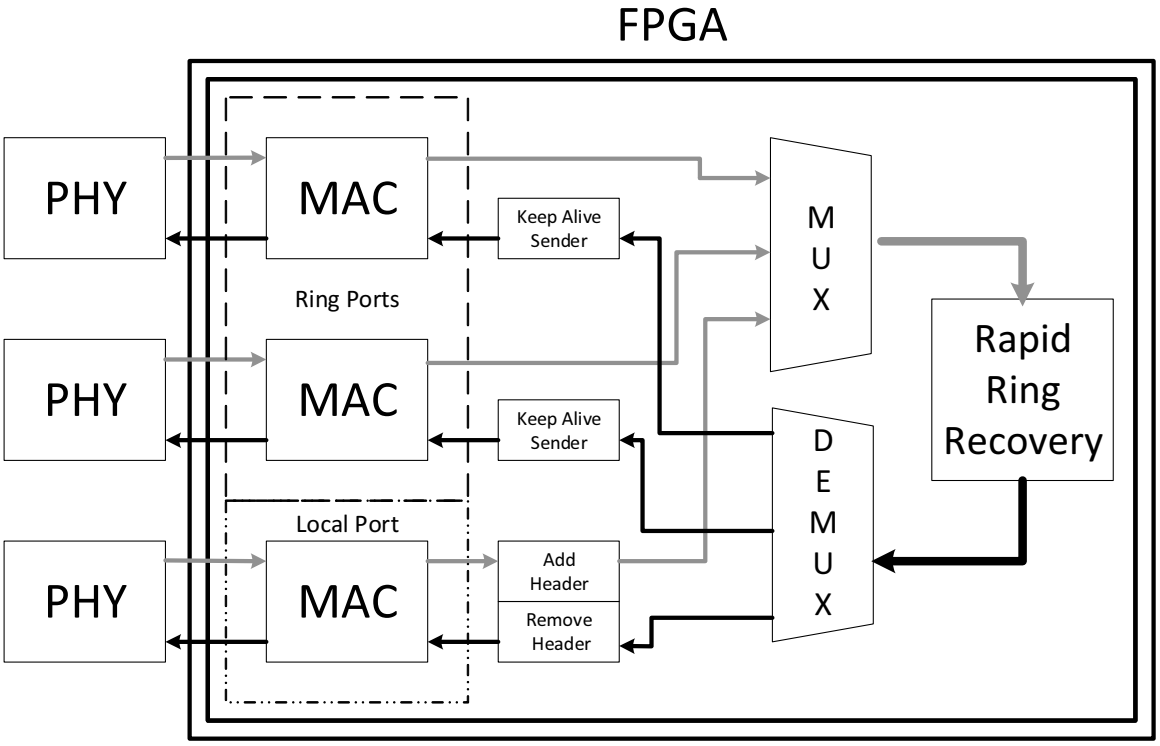

Figura 4.1: Arquitetura do Sistema

4.2.1 PHY

Em todos os sistemas de comunicação é necessário uma ligação física para a transferência de dados. Na tecnologia Ethernet 100 Base-T o meio usado são condutores de cobre em pares entrançados. A transmissão de informação é feita conforme uma norma definida, e.g. , o que permite o funcionamento entre vários equipamentos produzidos por empresas diferentes. A norma específica todo o funcionamento da tecnologia incluindo a camada física. As especificações da camada física englobam características, tais como as ligações elétricas ou a própria dimensão física dos componentes. Para acelerar a prototipagem de projetos que utilizem a

Ethernet foram desenvolvidos componentes eletrónicos conhecidos por Physical Layer Transceiver (PHY) que implementam toda a parte física da tecnologia e oferecem uma interface para as camadas superiores que simplifica a receção e envio da informação. Neste trabalho foi usado um circuito integrado produzido pela SMSC e o modelo é LAN8710A. Este integrado desempenha as funções necessárias da camada física e usa uma interface denominada por Reduced Media Indenpedant Interface (RMII) para comunicar com a camada superior que neste caso será o TEMAC Core.

4.2.2 Medium Acess Control

Tal como para a camada física, existem normas para o acesso ao meio e componentes que os implementam. O hardware que realiza essa tarefa é conhecido por *Medium Acess Control*, o qual desempenha as seguintes funções:

- Enviar e receber tramas.
- Verificar e calcular o FCS.
- Descartar tramas mal formadas.
- Adicionar ou remover o *preemble* da trama Ethernet e o *Start of Frame Delimiter*.
- Adicionar o *Padding* no caso da trama não ter o tamanho mínimo.

Estas são algumas das funções desempenhadas pelo *Medium Acess Control*. Para este trabalho foi usado o xilinx Tri-Mode Ethernet MAC (TEMAC) Core [4]. Este IP Core é fornecido pela Xilinx sendo a sua finalidade desempenhar o papel de MAC Ethernet para velocidades de 10/100/1000Mbps. Este bloco permite usar a tecnologia Ethernet em sistemas embutidos baseados em FPGA com facilidade. O TEMAC core é gerado através de um software no qual é possível, durante o processo de criação, realizar varias personalizações ao core para a sua melhor adequação ao sistema. Juntamente com o bloco TEMAC são gerados blocos auxiliares. Esses blocos são constituídos por filas de espera, registos e lógica adicional. A função desses blocos é de simplificar o uso do bloco TEMAC. Para ajudar o utilizador a perceber melhor o funcionamento do conjunto dos blocos é fornecido um bloco denominado *Swap Module* que se interliga aos outros blocos. Esse bloco usa o TEMAC Core para receber tramas e reenviar a trama para trás mas com os campos *destination address* e *source address* trocados. A analise desse bloco permite ter uma visão global dos passos necessários para usar o TEMAC Core. Sendo possível ver uma abordagem de como usar a interface cliente para enviar e receber tramas, nesse bloco também são realizadas algumas manipulações realizadas nas tramas Ethernet. A estrutura do projeto com o *Swap Module* está representado na figura 4.2

O TEMAC Core contem registos através dos quais é possível configurá-lo. Para esse efeito foi criado uma máquina de estados para permitir alterar cada um desses registos com os valores desejados. Nesses registos é possível configurar a velocidade de transmissão, ativar ou desativar a verificação do *Frame Check Sequence* pelo TEMAC Core, permitir a receção de tramas com tamanho superior a 1500bytes, aplicar filtros aos endereços MAC etc...

O bloco MAC presente na figura 4.1 é constituído por um TEMAC core que desempenha as funções de MAC ao qual se acrescenta mais um conjunto de lógica adicional para permitir uma interface mais fácil e um bloco que permite configurar o TEMAC Core com as funcionalidades desejadas.

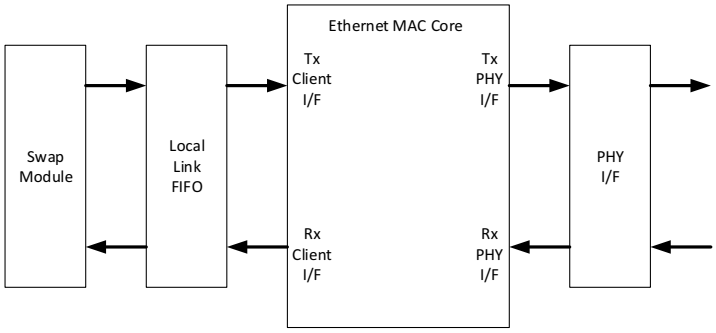

Figura 4.2: Projeto com swap module

4.2.3 Add Header e Remove Header

As tramas usadas no RRR têm um formato bem especificado. Esse formato não coincide com as tramas que são usadas na maioria dos equipamentos que usam Ethernet. Para simplificar o bloco central denominado "Rapid Ring Recovery"foram desenvolvidos 2 blocos cuja função é adicionar ou remover campos às tramas proveniente dos portos locais. Assim todas as tramas que chegam ao bloco central têm o mesmo formato e são analisadas da mesma maneira. Na figura 3.2 a trama que está do lado esquerdo é uma trama Ethernet genérica. Do lado direito está uma trama com um formato definido pela norma 802.1ah que é o formato usado no RRR e consequentemente neste trabalho. Na figura também é possível ver quais são os campos que foram acrescentados.

O bloco "Add Header"está interligado ao MAC do porto local. Quando uma trama for recebida naquele porto e o multiplexer tiver o porto local selecionado. O bloco "Add Header"vai adicionar os campos em falta com valores predefinidos. Os campos acrescentados são inicializados com os seguintes valores:

- O campo B-DA vai ser inicializado com o valor 0xFFFFFFFFFFFF. As tramas não são enviadas para um nó especifico mas para um equipamento ligado nalgum dos nós. Assim o endereço destino do nó não é especificado.

- O campo B-SA contém o endereço do nó que introduz a trama na rede. Neste caso o bloco vai adicionar este campo com o endereço do próprio nó, visto ser o nó que está a introduzir a trama na rede.

- O B-TAG é constituído por dois campos, um ethertype e um B-ID. Para este trabalho estes dois campos são inicializados sempre com os mesmos valores. O conteúdo de cada um destes campos é 0x88A8 e 0x0000 respetivamente.

- O I-TAG é formado por um *ethertype* e o I-SID. O *ethertype* tem o valor constante de 0x88E8 e o I-SID é usado para indicar se a trama está num túnel e em qual VLAN ela está a ser enviada. Ao inicializar o I-SID o bloco vai atribuir o valor '0' ao bit de túnel e vai definir a VLAN usada com o mesmo valor que o ID do próprio nó. O I-SID pode vir a ser alterado pelos outros blocos caso seja necessário.

- O C-TAG é o último a ser adicionado a trama original. Tal como o B-TAG o C-TAG é constituído por um ethertype e um C-ID. Os valores destes campos são 0x8100 e 0x0000, respetivamente.

Quando uma trama for destinada ao porto local é necessário voltar a retirar os campos adicionados a trama original. Para desempenhar essa tarefa está implementado o bloco "Remove Header. Este bloco vai interligar o demultiplexer e a interface da transmissão do MAC, descartando os campos adicionados ao formato base da trama. Este processo de descartar certos campos da trama é realizado com a manipulação dos bits de controlo que interligam os blocos. Esta manipulação permite que só sejam enviados os campos desejados para o bloco MAC e consequentemente são descartados os campos não desejados.

4.2.4 Multiplexer

Em alguns sistemas com várias entradas e com uma única unidade de processamento é necessário selecionar qual das entradas vai estar interligada à unidade de processamento. Nesses casos é usado um multiplexer para efetuar a seleção e proceder à interligação. O multiplexer existe numa grande variedade de aplicações, existindo para circuitos digitais como para circuitos analógicos. Consoante a sua aplicação a sua implementação também irá mudar. O multiplexer é um elemento essencial neste trabalho. Como só existe uma unidade de "atendimento"é necessário a existência de uma entidade para definir qual irá ser o próximo porto a ser atendido. Essa escolha depende do algoritmo implementado no MUX. Uma das principais vantagens de trabalhar com FPGA é a possibilidade de trabalhar a nível de hardware e continuar a ter a liberdade para desenvolver blocos completamente personalizáveis. Neste caso o multiplexer vai permitir a um porto de comunicar com o bloco central num mecanismo semelhante ao "token ring". Em cada ciclo de relógio o token muda para o porto seguinte. Caso o porto tenha uma trama na fila de espera ele é atendido. Quando ele acabar de receber a trama, o token passa para o porto seguinte. Caso o porto não tenha tramas na fila de espera, o token muda para o porto seguinte. Este mecanismo impede que um porto fique indefinidamente a espera de ser atendido. Cada vez que um porto recebe uma trama ele tem que libertar o token. Quando um dos portos está a ser atendido o multiplexer assegura a ligação entre esse porto e a unidade central. Quando for sinalizado o fim da receção da trama a ligação é interrompida. Esta é uma abordagem fácil de se implementar e com resultados satisfatórios.

4.2.5 Rapid Ring Recovery

Esta é a unidade central do dispositivo. É neste bloco que o algoritmo do RRR é realizado. Quando uma trama chega é analisada e, caso necessário, alterada. Para realizar esta tarefa este bloco é constituído por vários elementos sendo eles: uma máquina de estados que implementa o algoritmo do protocolo, uma memória com acesso realizado através do conteúdo e temporizadores. Estes elementos são essenciais para o bom funcionamento do dispositivo.

Content Address Memory

Quando a rede é inicializada os nós da rede não sabem que dispositivos estão a usar a rede. Mas com a informação proveniente das tramas é possível aos nós descobrirem o endereço dos dispositivos que estão a usar a rede e onde é que eles estão relativamente a ele. Com a aprendizagem é possível aumentar o desempenho da rede, o que torna esta característica muito útil. Neste caso, quando um nó recebe uma trama destinada a um dispositivo presente num dos portos locais, a trama é-lhe diretamente enviada. Se a aprendizagem não é utilizada a trama é enviada para o dispositivo mas também é enviada para o anel e percorre toda a rede

desnecessariamente. A informação recolhida é armazenada numa tabela. A tabela é conhecida por tabela de encaminhamento. Neste trabalho a tabela de encaminhamento utilizada nos nós é implementada com uma *Content Address Memory* associada a uma memória. Uma memória do tipo *Content Address Memory* permite aceder à memória não pelo endereço de uma posição de memória mas sim pelo conteúdo de uma posição de memória. Esta memória é muito útil para este tipo de trabalho. Quando existe um acesso à memória, o objetivo é de encontrar em que posição da memória está o endereço MAC de um dispositivo. Numa memória tradicional seria necessário percorrer as várias posições de memória e desperdiçar vários ciclos de relógio. Com esta memória basta um ciclo de relógio para saber se o endereço está presente na memória e para ter acesso à informação associada ao endereço. No caso especifico deste trabalho, a tabela de encaminhamento de cada nó vai conter o endereço do dispositivo, o porto no qual recebeu a trama e a VLAN utilizada para enviar a trama.

Temporizadores

O mecanismo de deteção de falhas é baseado na receção periódica de tramas. Quando o porto fica um tempo predefinido sem receber nenhuma trama, a ligação desse porto é considerada inativa. Para monitorizar o tempo entre a chegada de cada trama são utilizados temporizadores. Assim é preciso um temporizador para cada porto. Quando chega uma trama o temporizador é reinicializado. Quando um temporizador chega ao valor máximo definido, ele ativa um bit para sinalizar que a ligação do porto está inativa. O valor definido nesse temporizador vai ser o tempo que um nó demora no máximo para detetar uma falha na ligação. O valor do temporizador na proposta inicial é de 225 microssegundos [11].

Máquina de estados

É na máquina de estados que é realizado a análise e o processamento da trama. Essas tarefas são realizadas com auxílio da CAM e dos temporizadores. Consoante a análise efetuada à trama, este bloco vai assegurar que o reencaminhamento da trama é realizado corretamente. Para isso ele vai atribuir aos bits de controlo os valores adequados. Na figura 4.3 está representada a máquina de estados que implementa o algoritmo utilizado. Quando o sistema é inicializado a máquina de estados vai para um estado de espera e vai permanecer nesse estado enquanto não chega nenhuma trama. Quando chegar uma trama é necessário verificar que tipo de trama se trata. Neste trabalho são considerados três tipos de trama:

- O primeiro tipo de trama são as tramas que tenham o bit de túnel igual a '1'. Quando uma trama encontra uma falha no seu caminho para o destino ela é reenviada para trás com um bit de túnel alterado para o valor '1'. Este bit serve para indicar que a trama está a realizar um "percurso"de recuperação. Os dispositivos quando recebem uma trama deste tipo não fazem a aprendizagem do endereço MAC, uma vez que a trama não está a realizar o percurso normal e logo o porto no qual o dispositivo recebeu a trama pode também não ser o correto. As tramas ficam com o bit de túnel ativo até voltarem à origem, o que significa ter de voltar ao nó que a introduziu no anel.

- Outro tipo de tramas utilizado neste trabalho são tramas enviadas com o único propósito de manter as ligações ativas. Este tipo de trama não contem nenhum tipo de informação e é unicamente utilizada para o propósito referenciado anteriormente. Esta tramas são denominadas por tramas de *keep alive*. Para o nó conseguir reconhecer este tipo de

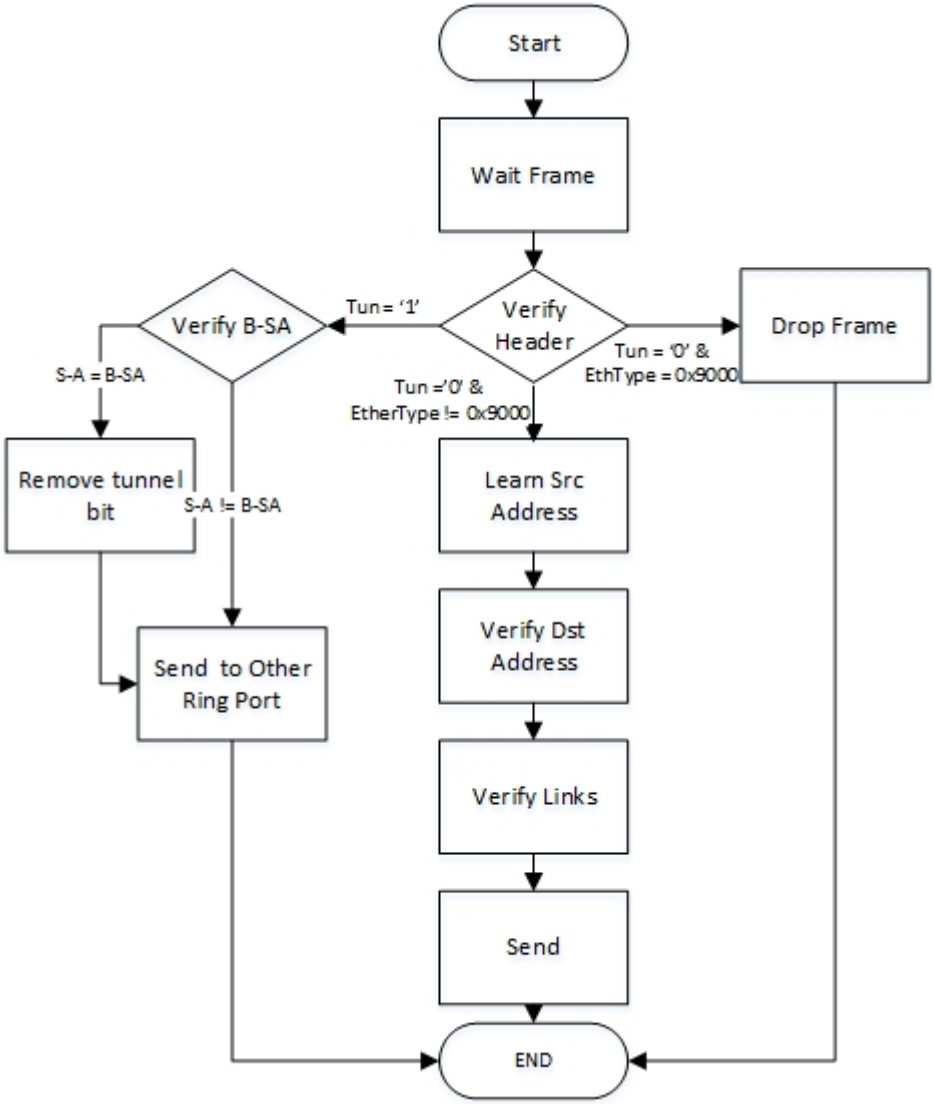

Figura 4.3: Máquina de estados

trama, o campo ethertype da trama Ethernet é alterado para o valor 0x9000. Assim o nó pode verificar com facilidade que se trata de uma trama de *keep alive*.

- As tramas que não se enquadrem com os dois tipos anteriores serão consideradas tramas de dados, enviadas pelos dispositivos ligados aos nós do anel. Este tipo de tramas são tratadas consoante o algoritmo do RRR. Cada trama destas será reencaminhada por forma a chegar ao seu destino.

Se a trama recebida é do primeiro tipo("tunnel = 1") o próximo passo é verificar o endereço do nó que introduziu a trama no anel. Caso o endereço presente na trama coincida com o endereço do nó significa que ele enviou a trama e que ela encontrou uma falha e teve que voltar para trás. O nó vai repor o bit de túnel com o valor '0' e reenviar a trama, mas desta vez pelo outro porto. Caso o endereço não coincida a trama é simplesmente encaminhada para o outro porto.

No caso da trama não ter o bit de túnel ativo e ter o ethertype com o valor 0x9000 estamos no segundo tipo de tramas. Como as tramas de *keep alive* não contêm nenhuma informação relevante, elas são simplesmente recebidas e descartadas após de ter efetuado um reset ao respetivo timer. Por último, se a trama recebida não se enquadrar com nenhum dos tipos anteriores significa que é do terceiro tipo. Neste caso a trama vai ser reencaminhada para o seu destino consoante os endereços provenientes da trama.

Para efetuar o reencaminhamento da trama o nó vai realizar várias tarefas. Logo a seguir a receber a trama o nó vai verificar se o endereço presente no campo *Source Address* da trama se encontra na tabela de encaminhamento. Caso exista, o nó vai atualizar os valores presentes na tabela. Se ainda não existe nenhuma entrada para esse endereço o nó vai acrescentar uma nova entrada na tabela associada a esse endereço. O passo seguinte consiste em verificar a existência do endereço destino na tabela de encaminhamento. No caso do endereço destino existir na tabela e o porto associado é o porto local, a trama vai ser encaminhada diretamente para o porto local. Se o porto associado é um dos portos dedicado à rede em anel, a trama é encaminhada para o anel no mesmo sentido em que chegou. No caso do endereço não existir na tabela a trama é encaminhada para o porto local e reintroduzida no anel. Neste ponto está definido para que portos a trama vai ser enviada. Antes de enviar a trama é necessário verificar que a ligação do porto que vai ser usada está a funcionar. Para efetuar essa verificação, o nó vai consultar os temporizadores e comprovar o bom estado da ligação. Caso as ligações estejam a funcionar a trama é enviada. Caso contrário é efetuado a recuperação da falha enviando a trama destinada para o anel no sentido inverso do que aquele em que a trama chegou. Com a inversão do sentido do percurso da trama o nó também vai alterar a VLAN na qual a trama está a ser transmitida para a VLAN que não usa essa ligação. Pode acontecer que a ligação que falhou não seja usada na VLAN na qual a trama está a ser transmitida. Nesse caso a trama é simplesmente descartada e não há recuperação. O processo de descartar a trama nesse caso é realizado pelo bloco seguinte, o demultiplexer. Todos os casos descritos anteriormente são tramas proveniente do anel.

Quando a trama é proveniente do porto local o nó introduz simplesmente a trama no anel pela porta que está a funcionar e altera na trama o valor da VLAN para a mais adequada. Não é necessário recuperação da trama. Após estas verificações a trama está pronta para ser enviada e são alterados os bits de controlo para comunicar aos outros blocos a informação necessária para o correto encaminhamento da trama.

4.2.6 Demultiplexer

Tal como na receção, é necessário interligar o porto correto com o bloco central para proceder ao envio da trama. Nesta implementação existem 3 portos Ethernet, o que torna necessário um bloco que realize essa interligação. Ao contrário do multiplexer na entrada, este bloco não vai decidir qual é o porto que vai ficar interligado. Quando chega ao momento de enviar a trama, o bloco "Rapid Ring Recovery"é que vai definir para quais portos é necessário enviar a trama e indicar ao demultiplexer essa informação através dos bits de controlo. O envio da trama pode ser para um único porto ou para vários portos, como no caso em que é necessário enviar para o porto local e para o anel. Como o processamento da trama é feito com um mecanismo *Store and Forward* a trama é armazenada. Mas o envio para vários portos é simultâneo. Para conseguir realizar essa tarefa este bloco tem de ser capaz de interligar vários portos ao bloco "Rapid Ring Recovery", mas o barramento de dados e controlo usado foi projetado para ligar só um porto. Assim quando é necessário interligar mais que um porto é preciso adicionar lógica adicional para que os bits de controlo sejam controlados pelos vários portos e não por um único porto. Assim é assegurado o envio correto e em simultâneo da trama nos vários portos. Além da interligação dos portos ao bloco principal, o demultiplexer tem como função gerir os portos utilizados em cada VLAN. Uma das tarefas é verificar em que VLAN a trama está a ser enviada. Caso a ligação do porto para o qual a trama está a ser enviada não pertencer a VLAN em questão a trama é descartada.

4.2.7 Keep alive sender

O mecanismo de deteção de falhas requer o envio e a receção de tramas. O envio de tramas é necessário para que os nós vizinho não considerem as ligações desativas. Para a ligação não ser considerada desativa é necessário evitar que o nó esteja mais de um determinado tempo sem enviar tramas. Como o tráfego gerado nem sempre é suficiente é necessário a existência de uma entidade responsável por monitorizar o envio das tramas e se necessário enviar tramas. Nesta implementação o bloco responsável para essa tarefa denomina-se por "keep alive sender". Este bloco vai ter um temporizador que é reiniciado no envio de uma trama pelo porto Ethernet. Se o temporizador atingir o limite máximo é enviada uma trama de *keep alive* pelo porto. O valor desse temporizador tem que ser inferior ao intervalo de tempo necessário para considerar uma ligação inativa.

4.2.8 Interligação dos blocos

A implementação do equipamento foi realizada consoante a arquitetura acima descrita. Cada um dos blocos foi criado para desenvolver tarefas muito bem definidas. Todos eles são baseados em máquinas de estados juntamente com blocos lógicos para auxiliar as máquinas de estados a atuar no sistema. Estes blocos vão receber, analisar e enviar tramas. As tramas vão atravessar os vários blocos durante o reencaminhamento. Essa transmissão da trama ao longo dos blocos vai ser realizada byte a byte e sincronizado com um relógio de 25Mhz. Devido a esta sincronização é essencial a existência de ligações entre cada bloco. Essas ligações incluem bits de controlo para controlar as transferências e assegurar que elas sejam feitas no momento ideal. A máquina de estados de cada bloco vai estar dependente de bits de controlo controlados pelas máquinas de estados dos blocos vizinhos, o que torna este processo muito bem controlado e previsível.

4.3 Plataforma

Para implementar a RedBox é usada uma placa de desenvolvimento Mars Starter[8] e um módulo MX1[7] sendo ambas fabricadas pela Enclusta. A placa de desenvolvimento Mars Starter consiste numa placa de desenvolvimento com vários interfaces disponíveis tais como portos Ethernet, USB e HDMI. Além dos vários conectores disponíveis também se encontra na placa vários periféricos para serem usados em todo o tipo de aplicações. O processamento é realizado num módulo que se acrescenta a placa. Existem vários módulos disponíveis, cada um deles com componentes diferentes tais como a FPGA, o numero de PHYs Ethernet ou a RAM disponível. Para este trabalho foi escolhido o modulo MX1 que contém uma FPGA Spartan 6 e dois PHYs Ethernet. Nas figuras 4.4 4.5 retiradas dos respetivos manuais de utilizador é possível ver os vários componentes que constituem a placa Mars Starter e o modulo MX1.

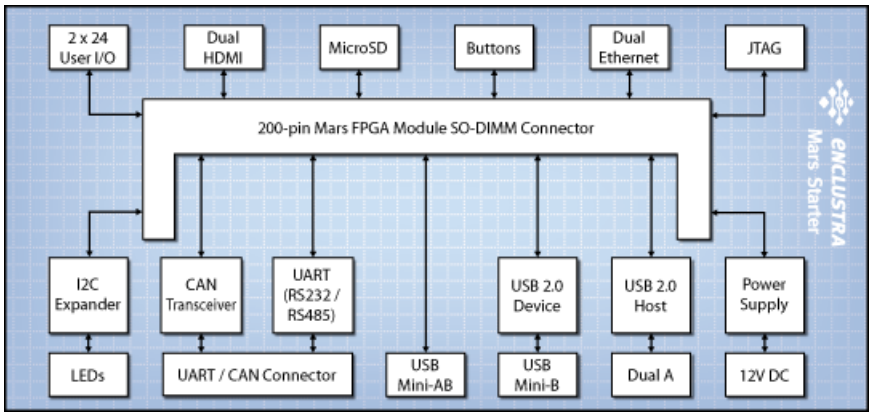

Figura 4.4: Estrutura da placa Mars Starter

Tanto a placa de desenvolvimento como o módulo só estão preparados para usar dois portos Ethernet. Como são necessário pelo menos três portos Ethernet, dois para ligar o dispositivo à rede e outro para ser usado como porto local, foi acrescentado um PHY Ethernet e o respetivo conector físico nos pinos de IO disponíveis na placa Mars Starter. Com este conjunto de material já é possível realizar a implementação da RedBox.

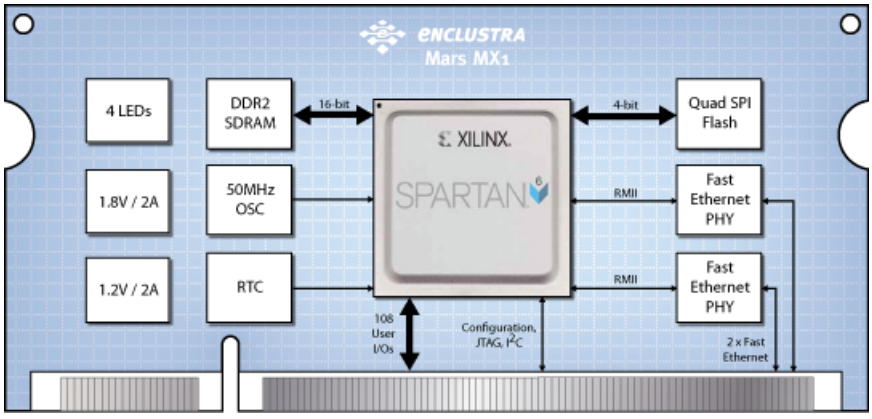

Figura 4.5: Estrutura da placa Mars MX1

4.4 Configuração do sistema

O dispositivo implementado vai ser utilizado como um nó de uma rede que funciona com o protocolo RRR. Para implementar uma rede com os dispositivos desenvolvidos é necessário interligar os nós com os cabos adequados. Quando todos os equipamentos estão interligados por forma a criar a rede é preciso configurar os nós para que a rede funcione corretamente. A rede testada neste trabalho é constituída por 3 nós e a sua configuração é estática. A configuração da rede passa por definir os IDs de cada um dos nós. Cada nó tem um ID que pode variar entre o valor 1 e o número de dispositivos da rede. A distribuição dos IDs não pode ser aleatória, sendo necessário atribuir o ID com o valor 1 a um dos nós e atribuir os IDs seguintes usando esse nó como referencia. A atribuição tem que ser seguida e no sentido horário. O ID do nó seguinte tem sempre um incremento unitário. Por exemplo o porto da direita do nó 1 irá estar ligado ao porto esquerdo do nó 2. A configuração do ID nos nós implementados neste trabalho é realizado com um botão de pressão existindo leds a indicar qual o ID atual. Também é necessário indicar ao nó qual o tamanho da rede. Com estes parâmetros os nós são capazes de associar as VLANs corretas à cada um dos seus portos ficando prontos para funcionar. Para ligar um dispositivo a rede, por exemplo um computador, basta ligar o dispositivo ao porto local de um dos nós. Todos os dispositivos ligados a rede tem conectividade entre eles, sendo assegurado a recuperação da rede devido a uma falha numa ligação em menos de 1 ms. Na presente implementação é necessário configurar manualmente a rede, assim adicionar ou remover um nó implica reconfigurar a rede.

4.5 Conclusões

Neste capítulo foi descrito a implementação de uma RedBox para uma rede RRR. Esta implementação foi projetada para cumprir com todos os requisitos determinados pelo protocolo RRR, sendo a sua implementação modular. O que permite fazer melhoramentos no futuro com facilidade. A sua validação funcional é realizada no capítulo 5.

Capítulo 5

Testes e validação

Para validar o dispositivo foram realizados vários testes. Todos estes testes foram realizados num *setup* experimental criado com a finalidade de testar o equipamento desenvolvido. Assim cada teste decorreu numa rede com 3 nós, sendo dois deles instrumentados com um gerador de tráfego e um *sniffer*. O primeiro teste validou o uso dos dois caminhos possíveis entre dois nós e a tolerância às falhas possíveis numa ligação. O segundo teste verificou o efeito de uma falha numa ligação quando o tráfego é periódico e cujo período é superior ao tempo de recuperação. O último teste, tal como o anterior, testou a rede na ocorrência de uma falha numa ligação mas para mensagens com período inferior ao tempo de recuperação. Os resultados foram os esperados e demonstra que esta abordagem permite que as comunicações recuperem após uma falha dentro do tempo de recuperação de 225μs.

5.1 Setup experimental

Após a implementação do dispositivo, foram realizados vários testes com o propósito de validar o funcionamento do dispositivo e da rede implementada. Para a realização desses testes é necessário criar um ambiente controlável. Os testes têm que decorrer em condições previamente definidas.

Nesse sentido foi criado um *setup* experimental que pode ser visualizado na figura 5.1. O *setup* é constituído por três nós e um computador. A interligação dos nós é realizada numa topologia em anel e dois desses nós são usados como terminais. Para gerar o tráfego e monitorizar o momento de chegada das tramas foram adicionados aos nós usados como terminais um gerador de tráfego configurável e um *sniffer*. O gerador de tráfego permite gerar tráfego Ethernet com periodicidade e conteúdo configurável enquanto que o *sniffer* é utilizado para monitorizar o momento de chegada das tramas e o seu conteúdo.

A configuração do gerador de tráfego é feito com recurso a um módulo UART e um terminal. É enviada uma trama na qual é indicado qual é o período desejado para o envio da trama, sendo o conteúdo das mesmas também incorporado na trama de configuração.

A função do *sniffer* é realizar a captura da tramas e enviar essa informação pelo módulo Universal asynchronous receiver/transmitter (UART) pelo terminal. Essa informação também contém o instante da captura da trama. Assim é possível conhecer o momento de chegada de cada trama e o seu conteúdo.

Os nós usados como terminais são interligados ao computador e, como referido anteriormente, comunicam através de uma UART. A gestão dos testes é realizado no computador,

sendo a configuração do tráfego e a recolha de informação realizada numa aplicação específica. Após a recolha dos dados, é realizado processamento nas amostras a fim de retirar os resultados pretendidos.

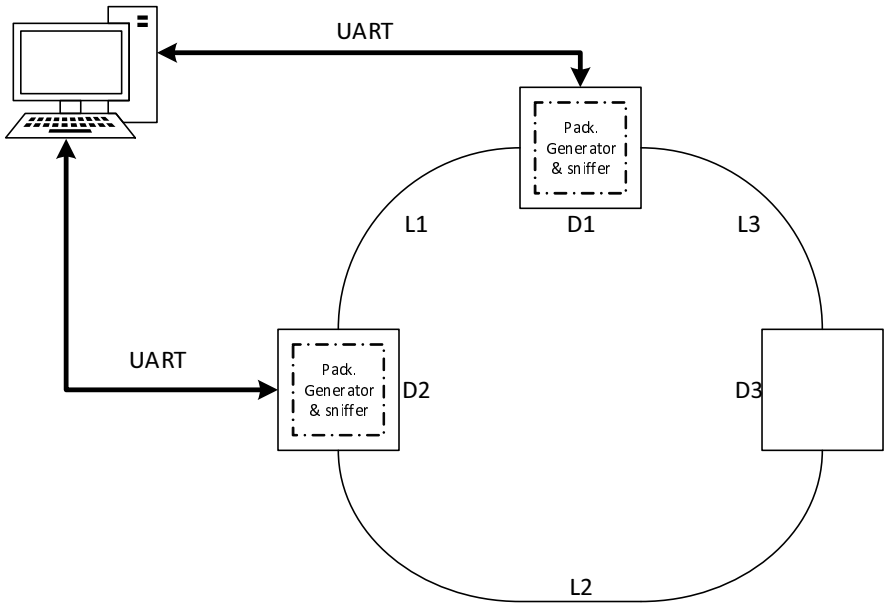

Figura 5.1: Setup experimental usado nos testes

O principal objetivo destes testes consiste em verificar o comportamento da rede e dos nós quando ocorre uma falha numa ligação. Nesse sentido são provocadas falhas nas ligações L1 e L2. Estas falhas são provocadas de uma maneira sequencial. Inicialmente as duas ligações funcionam, passado 4 milissegundos é provocado uma falha na ligação L1 que irá durar 4 milissegundos. Após outros 4 milissegundos é provocado outra falha mas na ligação L2. Esta falha também terá duração de 4 milissegundos. Depois desta falha o processo volta a repetir-se e tem uma duração de 16 milissegundos.

Para a realização dos testes, o tráfego gerado será sempre entre os nós D1 e D2. O sentido do tráfego depende do teste.

5.2 Utilização de vários caminhos

Numa rede em anel existem sempre dois caminhos entre dois destinos. No caso da comunicação não ser possível num caminho é usado o outro. Os dispositivos implementados fazem o envio das tramas seguindo o algoritmo do protocolo RRR. Quando existir uma falha numa ligação adjacente ao nó e que essa ligação seja usada por defeito, o envio da trama é realizado no outro sentido. No caso da falha não ser adjacente, um processo de recuperação irá ocorrer durante o envio da trama. Para verificar o funcionamento da rede foi gerado tráfego no nó D2 com destino ao nó D1 com periodicidade de 1ms. Na figura 5.2 está representado a diferença entre o momento esperado da chegada das tramas e o valor medido. Neste gráfico é possível verificar que há dois atrasos distintos. Um deles que é quase nulo que corresponde as tramas que usaram o caminho D2-D1 e o outro valor corresponde as tramas que percorreram a o caminho D2-D3-D1. Outro dado possível de verificar é que são enviadas 4 tramas num dos

caminhos e 12 no outro caminho. Como não existe tráfego nos dois sentidos a trama é sempre enviada por um caminho predefinido, que neste caso é o caminho D2-D3-D1. Quando ocorre a falha na ligação L2 e durante a sua duração, as tramas são enviadas pelo caminho D2-D1. Quando a ligação volta a ficar ativa o tráfego volta para o primeiro caminho.

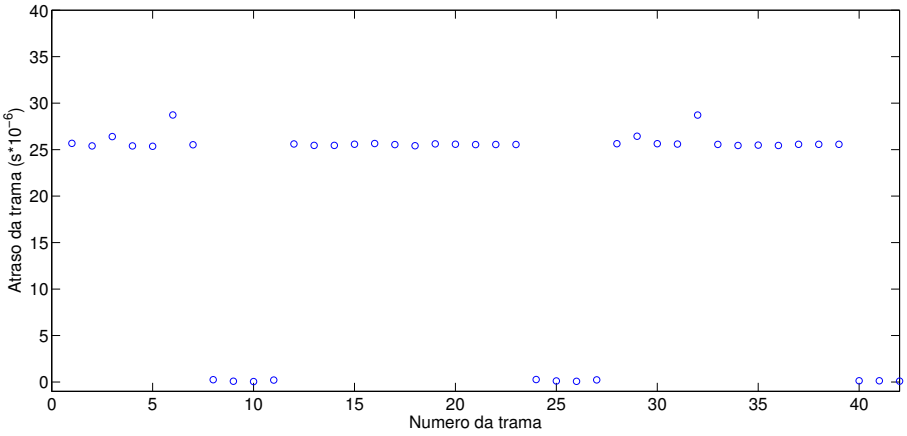

Figura 5.2: Atraso na chegada das tramas no sentido D2-D1

Com o objetivo de verificar se o nó fazia a aprendizagem dos endereços MAC dos equipamentos na rede e do porto no qual recebeu as tramas provenientes desses mesmos equipamentos, foi repetida a experiencia anterior mas com o tráfego a ser gerado nos dos sentidos. Na figura 5.3 está representado a diferença temporal entre o tempo esperado da chegada das tramas e o instante medido das tramas que circulam no sentido D2-D1. Tal como na primeira experiencia houve dois caminhos usados para o envio das tramas e consequentemente dois atrasos diferentes. Mas agora houve 8 tramas a chegar em cada um dos caminhos. Como o tráfego é nos dois sentidos o nó vai manter as comunicações no mesmo caminho até detetar uma falha ou receber uma trama daquele endereço no outro sentido. É possível ver que foi isso que aconteceu. Quando ocorre uma falha na ligação L1 o caminho passa a ser D2-D3-D1 o que torna o atraso maior. Passado 4ms a ligação volta a ficar ativa mas o envio das tramas continua a ser pelo caminho D2-D3-D1. Após mais 4ms aparece uma falha na ligação L2 e o caminho é então alterado para D2-D1, mantendo-se as comunicações nesse caminho até voltar a aparecer uma falha na ligação L1.

As figuras apresentadas anteriormente representam o tráfego gerado no nó D2. Como as falhas são adjacentes ao nó a trama é enviada num caminho sem falhas. Assim não é necessário existir a recuperação da trama. Para visualizar a recuperação da trama é feita uma captura da tramas provenientes do nó D1 e com destino ao nó D2. Esta captura pode ser visualizada na figura 5.4. Nesta figura já é possível ver 3 tempos de atraso diferentes. Os dois menores correspondem aos caminhos descritos anteriormente. O maior valor representa uma trama a ser recuperada. Como o nó D1 não é adjacente à ligação L2, a trama é enviada nesse caminho mesmo com a ocorrência de uma falha na ligação L2. Nesse caso a trama é enviada e no nó D3 é detetado a falha existente na ligação L2, sendo a trama reenviada para trás pelo nó D3. A trama percorre assim o caminho D1-D3-D1-D2. Entretanto o nó D2 também deteta a falha na ligação L2 e começa a enviar as tramas para o nó D1 pela ligação L1 ou seja pelo caminho D2-D1. Assim o nó D1 vai receber tramas pela ligação L1 e passa a comunicar com o nó D2

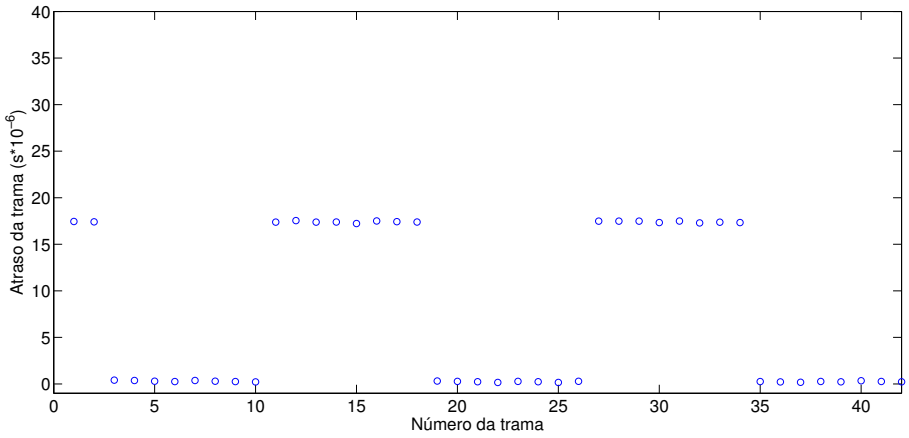

Figura 5.3: Atraso na chegada das tramas no sentido D2-D1

também por essa ligação. Assim só uma trama é que vai passar pelo processo de recuperação. Se o nó D2 não enviasse tramas o nó D1 continuaria a enviar tramas pelo nó D3 tornando necessário recuperar todas as tramas enviadas.

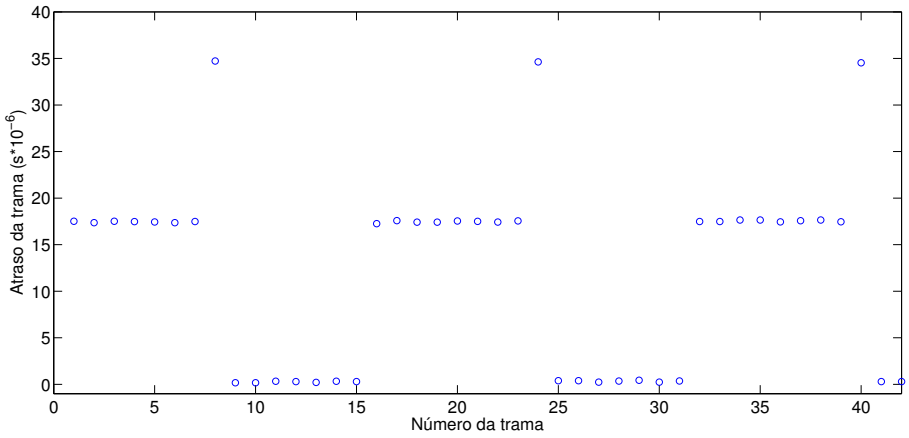

Figura 5.4: Atraso na chegada das tramas no sentido D1-D2

5.3 Envio de tramas com periodicidade superior a 225μs

Nos testes da secção anterior foram analisados os tempos de atraso na chegada da trama. Estes atrasos que são provocados pela mudança do caminho usado para o envio da trama. Nesta secção serão analisados os efeitos da periodicidade do envio da trama sobre a quantidade de tramas perdidas devido às falhas na ligação. Para os testes foram geradas mensagens com periodicidades diferentes. Foi gerado tráfego médio com velocidades de 10,15,20,25,30,45,50,60 Mbps. Para gerar essas cargas foi definido um tamanho para a trama, o qual neste caso foi de 1460Bytes e foi alterado a periodicidade do envio da trama consoante a carga desejada. O

sentido do tráfego era do nó D2 para o nó D1. Os resultados obtidos estão representados na tabela 5.1.

C(B)	T (μs)	BW (Mbps)	# Tx	# Lost	# Faults	Lost per fault
1460	1177	10	1020	20	75	0.27
1460	785	15	1022	22	50	0.44
1460	588	20	1021	21	37	0.57
1460	471	25	1023	23	30	0.77
1460	392	30	1022	22	25	0.88
1460	294	45	1020	20	19	1.05
1460	235	50	1022	22	15	1.47
1460	196	60	1018	18	12	1.50

Tabela 5.1: Tabela das mensagens com tamanho de 1460Bytes

É de notar que a diminuição do período provoca um aumento do número de perdas por falhas, pois aumenta o número de tramas enviadas por unidade de tempo. Como o tempo de recuperação é constante a probabilidade do envio da trama se realizar durante a deteção da falha aumenta.

Mas os valores de perdas por falhas apresentados são os valores médios. Como o tempo de recuperação é fixo, quando a periodicidade for superior ao tempo de recuperação não pode existir mais do que uma perda por falha. Na figura 5.5 está um histograma no qual é representado o intervalo de chegada entre cada trama do tráfego gerado com uma periodicidade de 1177μs. A barra com maior percentagem está posicionada no valor da periodicidade.

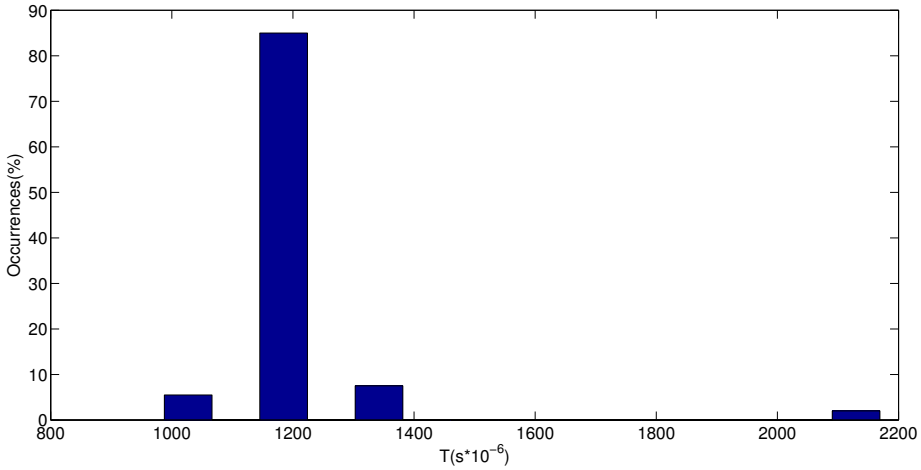

Figura 5.5: Histograma do intervalo entre a chegada de tramas com T=1177μs

Quando ocorre uma falha a comunicação é realizada noutro caminho. Essa mudança provoca uma diferença no momento de chegada da trama seguinte. As duas barras adjacentes à barra

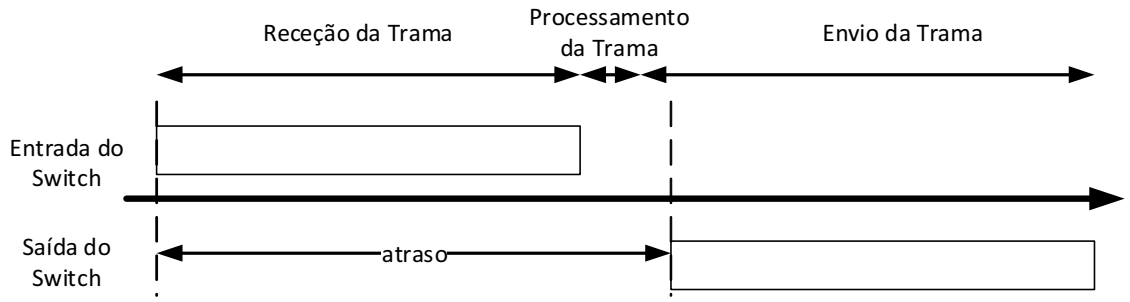

Figura 5.6: Instantes temporais no envio da trama

principal representam essas tramas. No caso de ocorrer uma perda, o tempo de chegada da trama vai ser duas vezes o valor da periodicidade da trama. A perda da trama ocorre com o aparecimento de uma falha e ao recuperar da falha o caminho usado também é alterado. A quantidade de nós atravessados em cada um dos caminhos difere de um nó. Assim a diferença temporal provocada pelo uso de um caminho diferente será o valor do atraso provocado por um nó. Na figura 5.6 estão representados os vários tempos na receção e envio da trama. O tempo de atraso é constituído pelo tempo de receção da trama, o tempo de processamento e o inicio do processo de envio da trama. O processo de receção da trama corresponde à receção da trama. Este tempo é proporcional ao tamanho da trama e é caraterizado por:

$$t_r = \frac{N * 8}{100000000} \tag{5.1}$$

Na equação anterior N é o numero de Bytes da trama. O tempo de processamento tem uma duração aproximada de 30 ciclos de um relógio com uma frequência de 25Mhz. O tempo de envio é o tempo necessário para escrever a trama numa memória e depois enviá-la. A equação que define o tempo de envio é a seguinte:

$$t_e = \frac{20 + N}{25000000} + \frac{N * 8}{100000000} \tag{5.2}$$

Assim o tempo de atraso é constituído pela soma do tempo de receção do tempo de processamento e do tempo de armazenar a trama na memória. Para o caso da trama ter um tamanho de 1460 Bytes o atraso é de 177μs. Neste caso o intervalo temporal entre tramas é igual a $2 * 1177\mu s - 177\mu s = 2177$. Na figura 5.7 é possível ver que as tramas provenientes após uma perda chegaram com uma discrepância para com o intervalo esperado. Essa diferença de 9μs provem de alguns elementos que constituem o sistema e que não estão bem caraterizados tais como o PHY Ethernet ou a interface do TEMAC Core. Com estes resultados é verificado que o intervalo de tempo entre a receção das tramas nunca ultrapassa os 2200μs o que permite concluir que na ocorrência de perdas nunca é perdido mais do que uma trama.

Na figura 5.8 está a representada a barra principal mas com mais resolução. É possível verificar que as tramas chegam com intervalos de tempo muito próximos do período definido. O desvio máximo do valor esperado é de 5μs.

Ao longo das medições foram encontrados casos nos quais havia mais do que uma perda por falha. Apesar do número médio de perdas por falhas ser inferior ao valor um, existe casos em que foi perdida mais do que uma trama. Como o tempo de recuperação é inferior ao período, não deveria haver perdas superiores a uma trama por falha. Na figura 5.10 é apresentado o

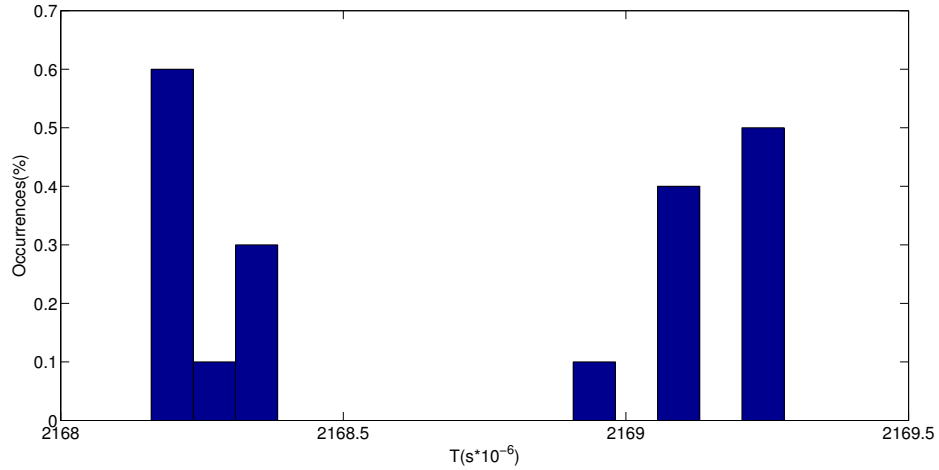

Figura 5.7: Histograma do intervalo entre a chegada de tramas após uma perda para T=1177μs

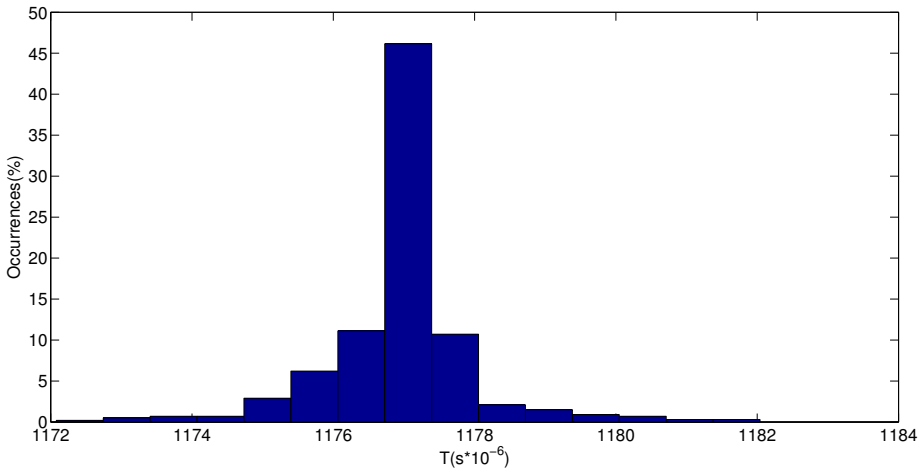

Figura 5.8: Histograma do intervalo entre a chegada de tramas com mais resolução para T=1177μs

histograma do intervalo entre chegada das tramas após uma perda para mensagens com 494μs de periodicidade. Existem duas barras: uma a respeito do caso em que houve uma perda e a outra para o caso em que houve duas perdas. A perda de uma trama era esperada, mas a perda de 2 tramas numa falha não era prevista. A razão de ocorrer duas perdas é devido ao envio da trama não ser instantâneo. Na Figura 5.9 é visível que uma falha pode afetar o envio de duas tramas mesmo quando a periodicidade é superior ao tempo de recuperação. Neste caso a trama tem 1460Bytes e entre o momento em que é escolhido o porto no qual a trama vai ser enviada e a finalização do envio da trama existe um intervalo de tempo próximo dos 177μs. Este valor é dado pela equação 5.2. Assim o intervalo de tempo no qual uma falha pode provocar perdas não é 225μs mas 225μs + 177μs o que dá 402μs. Como o resultado da soma é superior a periodicidade da trama é possível ocorrer duas perdas por falha quando o tráfego gerado tem periodicidade de 392μs e o tamanho da trama é 1460Bytes.

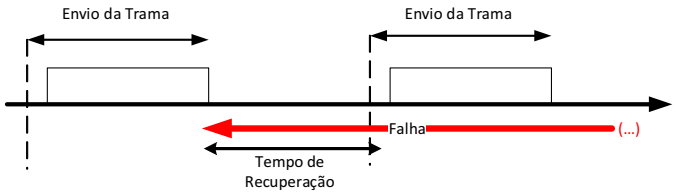

Figura 5.9: Efeito de uma falha no envio das tramas

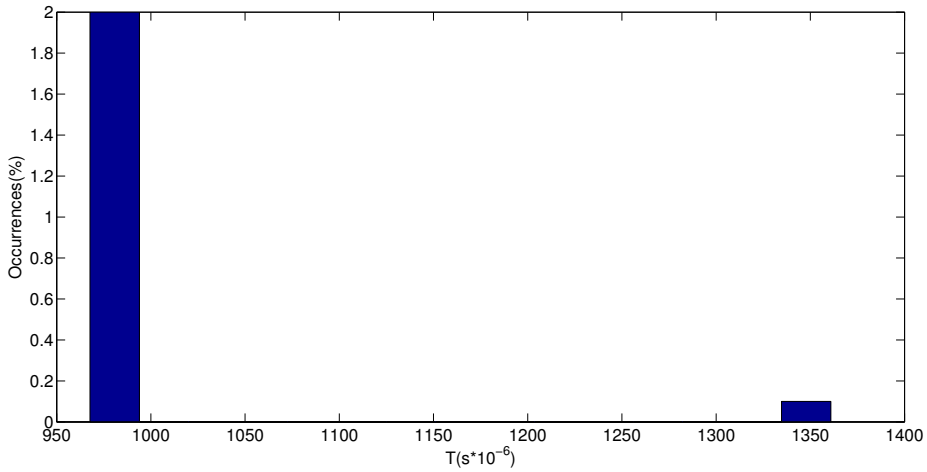

Figura 5.10: Histograma do intervalo entre a chegada de tramas após perdas para T=392μs

5.4 Envio de tramas com periodicidade inferior a 225μs

Estes testes são muito semelhantes ao testes descritos na secção anterior. Foram mensagens com periodicidades diferentes provocando cargas distintas e com o tamanho das tramas fixo. Mas nestas experiencias o tamanho da trama é de 136Bytes o que torna as tramas mais pequenas do que na experiência anterior onde o tamanho era de 1460Bytes. As cargas são as mesmas que na secção anterior mas como a trama é mais pequena a periodicidade do envio

das tramas também vai ser menor. Na tabela 5.2 estão os vários tráfegos gerados, o número de falhas ocorridas e o número médio de perdas por falha. Tal como na experiência anterior

C(B)	T (μs)	BW (Mbps)	# Tx	# Lost	# Faults	Lost per fault
132	115	10	1013	13	8	1.63
132	77	15	1011	11	5	2.20
132	57	20	1014	14	4	3.50
132	46	25	1011	11	3	3.67
132	38	30	1008	8	2	4.00
132	26	45	1016	16	2	8.00
132	23	50	1008	8	1	8.00
132	19	60	1011	11	1	11.00

Tabela 5.2: Tabela das mensagens com tamanho de 132Bytes

com a diminuição do período de envio da trama a quantidade de perdas por falha aumenta. Na figura 5.11 está representado o envio de uma trama com o período de 115μs.

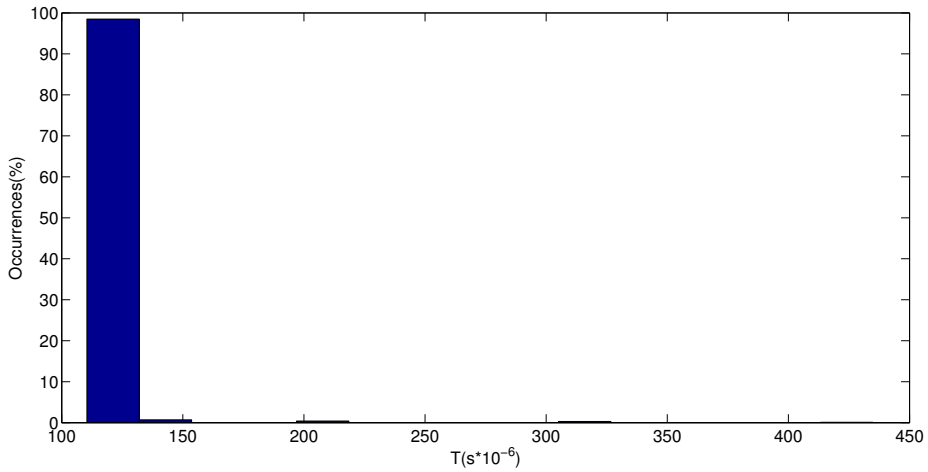

Figura 5.11: Histograma do intervalo entre a chegada de tramas para T=115μs

A maioria da tramas foram recebidas com um intervalo igual à periodicidade do envio. Mas algumas tramas foram recebidas com um intervalo de 204μs, 319μs e 400μs. Essas tramas estão melhor representadas na figura 5.12. Estes valores correspondem a perda de 1 trama, 2 tramas e três tramas. Os valores não são múltiplos de 115μs porque depois de detetar a falha existe uma mudança de caminho que provoca um atraso na chegada da trama. A mudança de caminho é sempre de um nó, então a diferença no tempo de chegada da trama será dada pelo tempo de atraso de um nó. Neste caso o tempo de atraso é de 26μs, assim o valor de 204μs provem de 2x115μs-26μs. Tal como no caso anterior o tempo máximo para deteção da falha é de 225μs, a esse valor é adicionado o tempo de envio da trama para saber qual o intervalo no qual pode haver perdas.

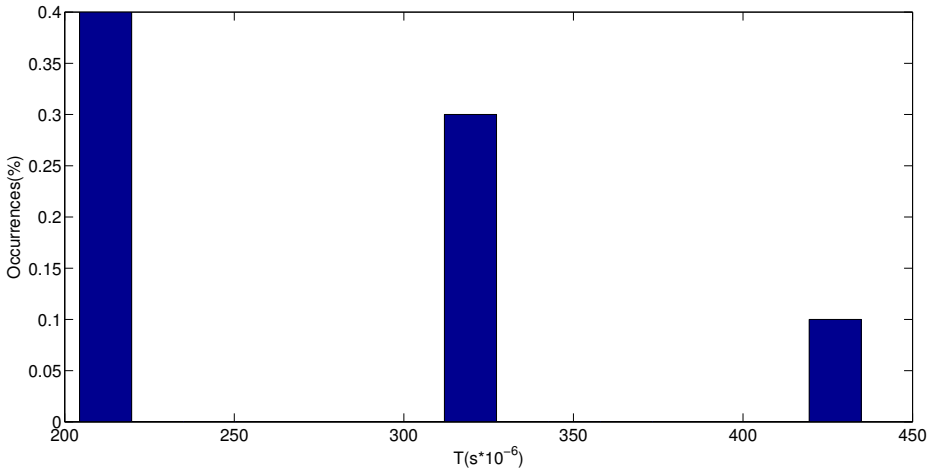

Figura 5.12: Histograma do intervalo entre a chegada de tramas após perdas para T=115μs

A periodicidade ser inferior aos 225μs não implica que ocorra uma perda em cada falha. O tempo máximo de deteção é de 225μs mas o tempo mínimo de deteção é inferior a esse valor. Nesse sentido é interessante analisar e quantificar esse tempo mínimo de deteção. Nesta implementação o envio de tramas de *keep alive* é realizada após 150μs do envio da última trama. Na figura 5.13 é visível o envio de tramas com um intervalo de 150μs que é o intervalo de tempo para provocar o envio de uma trama de *keep alive*. Se a falha ocorre no mesmo instante que o envio da segunda trama, a trama não irá ser recebida. Assim o melhor caso será a falha ocorrer próximo do fim do envio da trama. Assim o tempo mínimo de recuperação é de $225 - 150 = 75$. Nesse caso após 75μs a ligação é considerada inativa.

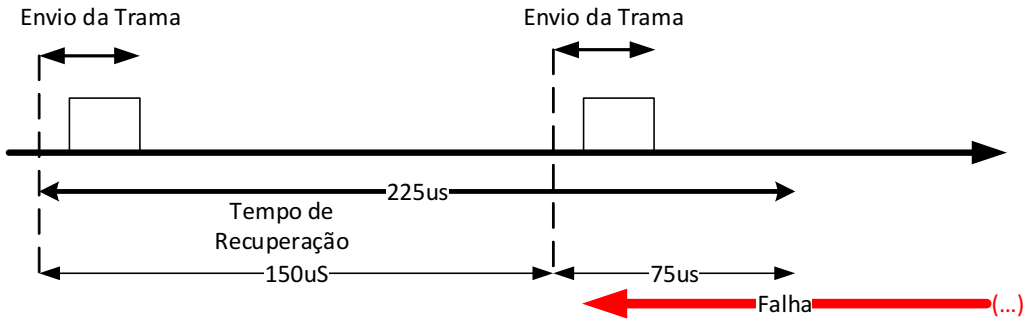

Figura 5.13: Tempo mínimo de recuperação

Este tempo mínimo de recuperação implica que as tramas com periodicidade inferior ao tempo mínimo de recuperação vão sempre ter pelo menos 1 perda. Na figura 5.14 está representado o intervalo de tempo entre duas tramas recebidas na ocorrência da perda de uma trama. O envio das tramas é de 38μs, como o tempo de recuperação mínimo é de aproximadamente de 75μs implica que em cada falha exista pelo menos 1 perda. Nas medições apresentadas na figura 5.14 houve 4 perdas e 6 perdas nas falhas ocorridas.

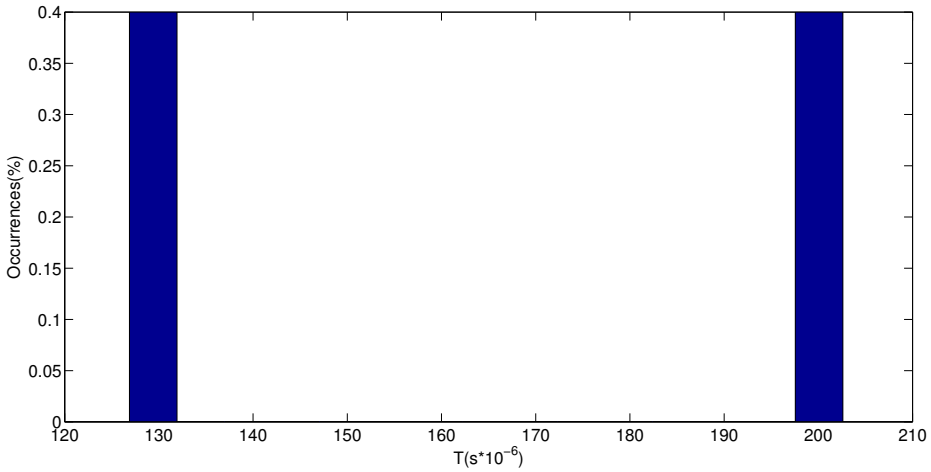

Figura 5.14: Histograma do intervalo entre a chegada de tramas após perdas para T=38μs

5.5 Conclusão sobre os resultados

Ao longo do capítulo foram apresentados os resultados recolhidos durante os testes e a sua análise. Houve resultados que não eram esperados inicialmente, mas após uma análise mais cuidada foram justificados, por exemplo a perda de duas tramas numa falha quando o envio tinha uma periodicidade superior ao tempo de recuperação. Os resultados demonstram o correto funcionamento da implementação.

Capítulo 6

Conclusões e Trabalho Futuro

Foi projetado e desenvolvido um dispositivo que permite criar uma rede com RRR e oferecer uma interface de rede para dispositivos com Ethernet mas que não suportem RRR. Para validar a solução foram realizados vários testes para validar as varias funcionalidades do equipamento. Os resultados demonstraram que esta implementação funciona como planeado. A recuperação da rede após a falha de uma ligação realizou-se conforme esperado, sendo usado um caminho alternativo para o envio da trama. Ao longo do desenvolvimento da implementação foi possível encontrar uma alteração possível para tornar a rede capaz de suportar a falha de um nó e não só de uma ligação.

6.1 Conclusões

Este trabalho teve como objetivo implementar um dispositivo que seja usado como um nó de uma rede. A rede é gerida com o algoritmo definido pelo protocolo RRR. Além de formar o anel da rede, o nó também oferece uma interface à rede para os dispositivos que não suportem o RRR. Para os nós funcionarem conforme o protocolo, eles necessitam cumprir varias especificações:

- Cada nó usa um mecanismo de deteção de erro baseado na receção de tramas num espaço de tempo determinado. Esse tempo irá ser considerado como o tempo de deteção de falhas.
- Os nós que formam a rede necessitam criar várias VLANs. Cada VLAN é diferente e usa todas as ligações da rede em anel exceto uma. A ligação que não é usada permite que o anel seja interrompido e por consequente a inexistência de ciclos.
- Quando um nó deteta um erro numa ligação e recebe uma trama que tinha essa ligação como destino, ele envia a trama para trás e usa a VLAN adequada.
- Se a trama é proveniente do porto local o nó acrescenta um cabeçalho à trama e envia a trama para o anel com a direção adequada.

Foi realizado uma série de testes para validar as características anteriores. A implementação funciona e corresponde as características definidas inicialmente. Assim demonstra-se que é possível incorporar as funções de comutação no interior da FPGA mantendo o seu funcionamento. Esta união torna assim o equipamento mais pequeno, mais robusto e menos caro.

6.2 Trabalho Futuro

Com a integração da parte de comutação no interior da FPGA já não existe a dependência do *switch* comercial e a necessidade da conformidade da trama com a norma IEEE 802.3ah. O que torna possível retirar alguns campos da trama. Na proposta inicial do RRR quando um nó falha, a rede toda também falha. Só é suportado a falha de uma ligação. Para corrigir esse inconveniente é necessário acrescentar um bit no cabeçalho da trama para indicar se a trama já foi recupera ou não.

Este trabalho estava focado em demonstrar que era possível realizar a comutação com uma FPGA. Nesse sentido foi realizado um nó experimental com o porto local e os portos dedicados ao anel a funcionar com 100Mbps em Full Duplex. Para tornar este nó mais robusto, e permitir o seu uso na prática, seria necessário usar os portos do anel com uma velocidade superior. Por exemplo com uma velocidade de 1Gbps. Como o anel é partilhado por todos os nós é importante ter uma capacidade de transmissão superior. Também é interessante ter mais que um porto local. Para permitir a dois equipamentos próximos usarem o mesmo nó.

Bibliografia

[1] *International electrotechnical commission, iec 62439-3 clause 4.*

[2] *International electrotechnical commission, iec 62439-3 clause 5.*

[3] *Standard ieee 802.1d. mac bridges. ieee 2004.*

[4] *Logicore ip tri-mode ethernet mac v4.4, user guide*, 2010.

[5] Giorgio C.Buttazzo, *Hard real-time computing systems - predictable sheduling algorithms and applications*, Kluwer Academic Publishers, 1997.

[6] R. Perlman et al, *Internet engineering task force (ietf) request for comments: 6325.*

[7] Christoph Glattfelder, *Mars mx1 user manual*, 2011.

[8] ———, *Mars starter board user manual*, 2012.

[9] Rafael Gouveia, *Injector de tráfego de pacotes ethernet em vhdl para fpga atlys*, DETUA (2013).

[10] Minh Huynh, Stuart Goose, Prasant Mohapatra, and Raymond Liao, *Rrr: Rapid ring recovery submillisecond decentralized recovery for ethernet ring*, IEEE Transactions on Computers **60** (2011), no. 11, 1561–1570.

[11] ———, *Rrr: Rapid ring recovery submillisecond decentralized recovery for ethernet ring*, IEEE TRANSACTIONS ON COMPUTERS VOL. 60 (2011).

[12] Herman Kopetz, *Real-time systems - design principles for distributed embedded applications*, Kluwer Academic Publishers, 1997.

[13] Paulo Pedreiras Michael Costa, Arnaldo Oliveira, *A rrr redbox for safety-critical networked embedded systems*, INForum - Simpósio de Informática, 2013.

[14] G. Prytz, *Network recovery time measurements of rstp in an ethernet ring topology*, Emerging Technologies and Factory Automation, 2007. ETFA. IEEE Conference on, 2007, pp. 1247–1253.

[15] R.Santos, *Enhanced ethernet switching tecnology for adaptibe hard real-time applications*, Ph.D. thesis, Universidade de Aveiro, 2010.

[16] Kritina Holden William Lidwell and Jill Butler, *Universal principles of design*, Rockport Publishers, 2003.

Apêndice A

Sniffer Ethernet

Numa rede de comunicações existe uma grande quantidade de tráfego a fluir pelas suas ligações. Esse tráfego nem sempre é acessível nos terminais da rede. A única maneira de se conseguir monitorizar esse tráfego é quebrando uma das ligações e realizar uma captura do tráfego. Nas redes Ethernet é comum introduzir um *hub* numa ligação, assim todo o tráfego é acessível em todos os portos do *hub*. Para realizar a captura do tráfego é ligado um computador a um desses portos e com o software adequado é realizada a captura do tráfego, esta implementação está visível na figura A.1. Esta abordagem tem vários inconvenientes, ao introduzir o *hub* estamos a tornar a ligação suscetível a ter colisões. Outra desvantagem é o facto da captura da trama não ser precisa. Apesar do conteúdo da trama estar correto, o instante da captura não tem um valor fiável. A razão desse instante não ser preciso é devido ao facto da captura ser realizada por software e ter que passar por várias camadas protocolares. Para resolver este problema existem equipamentos conhecidos por *sniffer* Ethernet. A função deste equipamento é realizar a captura da trama e o registo do seu instante de chegada. Este tipo de equipamento é desenvolvido com este único propósito, o que permite que ele seja

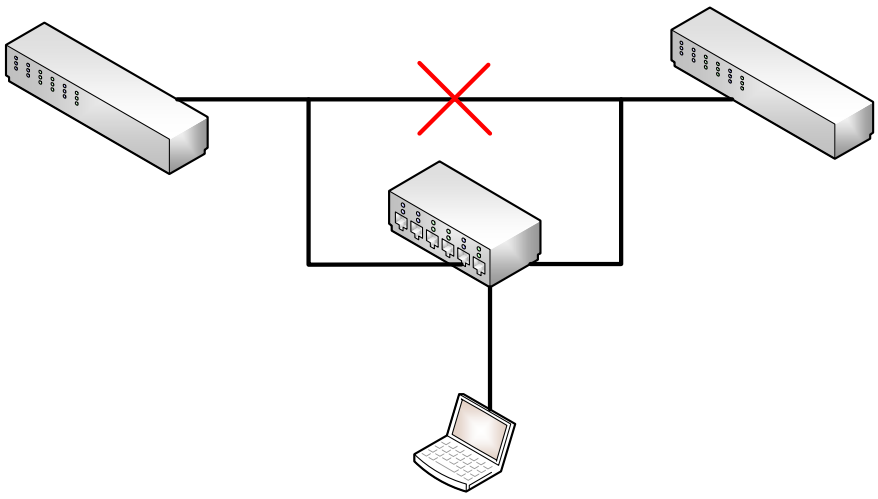

Figura A.1: Captura de trama usando um HUB

direcionado unicamente para capturar a trama. Assim o seu desempenho será maior. O *sniffer* tem que ser transparente para a rede. Isso obriga que a quebra provocada na ligação tenha a menor influencia possível no seu funcionamento. Uma abordagem possível passa por interligar

diretamente os dois portos mas capturando todo o tráfego que flui nessa ligação. No instante na qual uma trama passa pelo *sniffer* é realizado a captura do instante de tempo da chegada da trama. Isto permite ter uma grande precisão sobre os instantes temporais das chegadas das tramas. Mas infelizmente este tipo de equipamento tem preços de venda bastantes elevados. Nesse sentido foi realizado um *sniffer* numa placa de desenvolvimento baseada em FPGA para monitorizar o tráfego gerado numa rede Ethernet.

A.1 Arquitectura do sniffer

O *sniffer* implementado é baseado em vários blocos. Os blocos e a sua interligação podem ser visualizados na figura A.2.

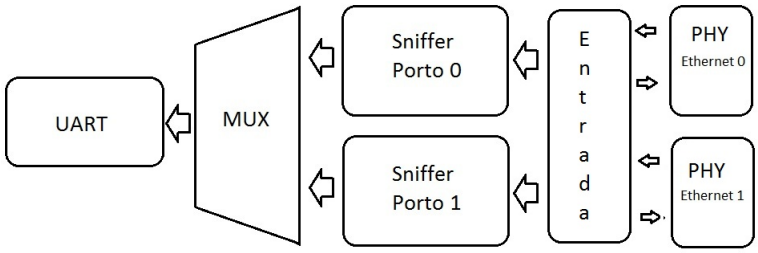

Figura A.2: Arquitetura do *sniffer*

A.1.1 Entrada

Este bloco está na entrada do sistema. A sua função é permitir ao dispositivo ser transparente na ligação onde é introduzido. Essa característica é possível ao interligar diretamente os dois PHYs Ethernet. Nessa interligação o bloco também permite ao resto do sistema de ter acesso ao conteúdo transmitido na ligação. Assim é possível realizar a captura da trama e provocar um atraso na ligação de baixo valor.

A.1.2 sniffer

O bloco *sniffer* vai efetuar a captura da trama e registar o seu instante de chegada. Para realizar essas funções o bloco *sniffer* é ele próprio constituído por vários elementos. Os elementos necessários para o funcionamento do bloco são visíveis na figura A.3. Existe um MAC Ethernet que denomina-se por TEMAC. A sua função neste sistema é de analisar os dados trocados entre os PHYs e fornecer essa informação ao sistema através de uma interface baseada em FIFOs. O conteúdo das tramas trocadas na ligação pode ser acedido nesse FIFOs. A captura da trama também inclui a captura do seu instante de chegada. Existe um temporizador que é acedido cada vez que é recebida uma trama, para permitir ao sistema capturar o instante de chegada de cada trama. Para comunicar toda esta informação ao utilizador é necessário um meio de transmissão. Neste dispositivo é usado uma porta serie para realizar as comunicações entre o utilizador e o sistema. O formato das tramas transmitidas para o utilizador está representado na figura A.4. Essa trama contém:

- um *Start of Frame* constituído por "\s".

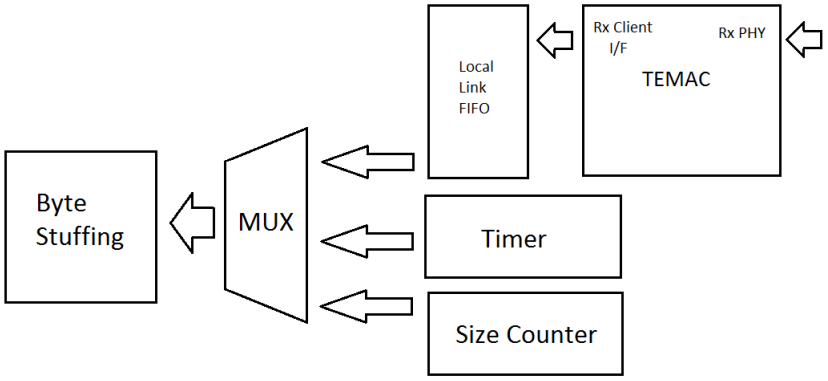

Figura A.3: Elementos que constituem o bloco sniffer

- O campo Trama Ethernet contem o conteúdo da trama. Esse conteúdo pode ser filtrado. O utilizador pode indicar ao sistema através da porta serie qual é o tamanho da amostra da trama. Em muitos casos só os cabeçalhos da trama são necessários para analise não sendo útil enviar todo o conteúdo da trama.
- Os dois campos seguintes indicam o instante da chegada da trama em segundos e microssegundos.
- O campo ID é utilizado para indicar em qual dos portos a trama foi recebida. Isto permite ao utilizador saber em que sentido foi capturada a trama.
- O conteúdo da trama Ethernet não é enviada em completo mas é útil saber qual era o tamanho original da trama. Essa informação é disponível no campo Size.
- Para sinalizar o fim da trama é utilizado um *End of Frame* constituído pelos caracteres "\e".

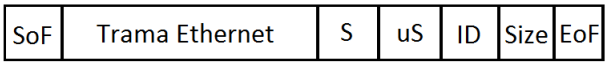

Figura A.4: Trama enviada para o utilizador

O bloco MUX na figuraA.3 é responsável por juntar toda a informação e de criar a trama destinada para o utilizador. Na receção da trama pode ocorrer que a aplicação desenvolvida para receber a trama de detetar um falso fim de trama. Os dois carateres usados para detetar o fim de trama ou incio de trama podem aparecer no conteúdo da trama. Para resolver esse problema existe o bloco Byte Stuffing. Este bloco vai receber uma trama destinada para o utilizador e sempre que encontra o caractere "\"vai duplica-lo. Assim quando aparecer a sequencia de carateres que definem o inicio e fim de trama não existe confusão. Na figura A.5 está o algoritmo implementado no bloco Byte Stuffing. A interface de saída deste bloco foi desenvolvida para ser diretamente ligada à UART.

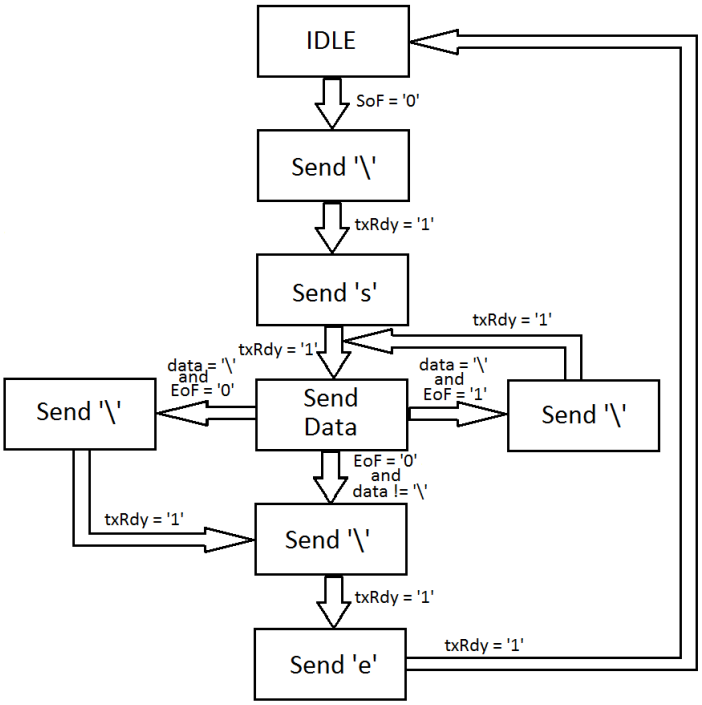

Figura A.5: Algoritmo do bloco Byte Stuffing

A.1.3 MUX

Existem dois portos a capturar tramas. Mas só existe um meio de comunicação para o utilizador. Este bloco vai definir de qual porto é enviada a trama para o utilizador. Na saída de cada bloco *sniffer* está um FIFO que disponibiliza qual é a sua occupação. O MUX vai atribuir o direito a transmitir ao bloco sniffer que tiver o FIFO mais cheio. Essa atribuição é feita enquanto o FIFO tiver a maior taxa de ocupação e o envio da trama atual não finalizar.

A.1.4 UART

Como foi referenciado anteriormente é usado uma porta serie para comunicar com o utilizador. Neste caso é usado uma UART. A UART está configurada para 8bits de dados, 1 stop bit e um BaudRate de 12Mbps. Sempre que uma trama Ethernet é recebida pelo *sniffer* é enviada uma trama para o utilizador com o seu conteúdo, o momento da receção e o porto no qual foi recebido. Para o utilizador receber essa informação, ele precisa de ter um computador e configurar uma porta serie com as configurações descritas acima. É possível criar uma aplicação que faça a receção dessas tramas e realize uma serie de operações para analise do desempenho da rede. Facilitando o processo de analise.

Apêndice B

Gerador de Tráfego Periódico

Realizar testes ao dispositivo desenvolvido implica o uso de outros equipamentos para simular várias situações aos quais o dispositivo pode ser sujeito. Um gerador de tráfego configurável é um equipamento muito útil para se conseguir simular esses ambientes. Graças a este tipo de equipamento é possível testar o dispositivo com um tráfego conhecido e retirar conclusões sobre o seu funcionamento. Como um gerador de tráfego é muito útil para realizar os testes foi desenvolvido um gerador de tráfego baseado no trabalho [9]. A implementação do gerador é baseado na tecnologia FPGA.

B.1 Funcionamento do Gerador

O gerador implementado permite gerar tramas Ethernet com o conteúdo desejado e permite definir a periodicidade do envio da trama. A estrutura interna do gerador pode ser visualizada na figura B.1. A comunicação entre o gerador de tráfego e o utilizador é realizada através de

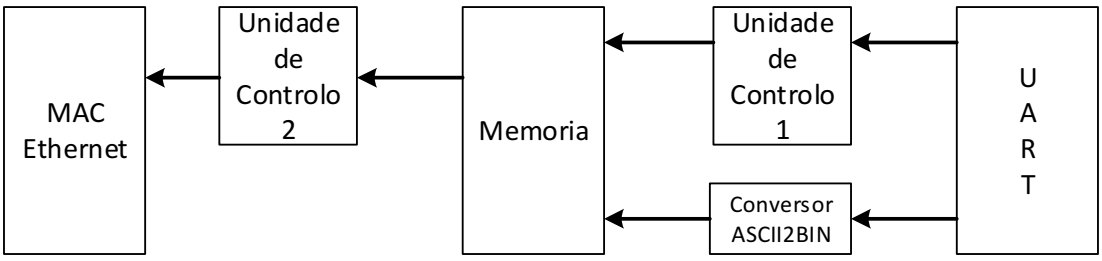

Figura B.1: Arquitetura interna do Gerador

uma porta serie. A configuração do gerador é efetuada ao enviar uma trama por essa porta serie. O formato da trama encontra-se na figura B.2. O primeiro campo, *Start of Frame*, serve para indicar o inicio da trama. No campo seguinte está a periodicidade desejada para o envio da trama. O valor da periodicidade vem em múltiplos de microssegundos. No campo seguinte está a trama desejada para ser enviado periodicamente e o ultimo campo indica o fim da trama. Na implementação do gerador foi criado um bloco que é responsável por receber a trama vinda do utilizador. Este bloco vai configurar o temporizador responsável por ativar o envio da trama periodicamente. Também é necessário converter o campo que contem a trama a ser enviada. Os dados da trama são introduzidos em ascii sendo necessário converter esses

Figura B.2: Formato da trama de configuração do Gerador

em valores em binário. O bloco responsável por empacotar esses dados numa trama Ethernet tem uma interface onde existe um barramento de 8 bits o que torna a conversão para binário essencial para simplificar a introdução dos dados nesse barramento. Os dados convertidos são armazenados numa memoria. No final da conversão a trama que será enviada periodicamente está armazenada na memoria, cada vez que que seja necessário enviar a trama basta aceder a memoria para ter acesso a trama e proceder ao seu envio. Para realizar essa tarefa foi criado outro bloco, cuja função é enviar uma trama sempre que o temporizador chegar ao valor pretendido. Para proceder ao envio, o bloco fornece ao MAC Ethernet os dados da trama que estão armazenados na memoria.