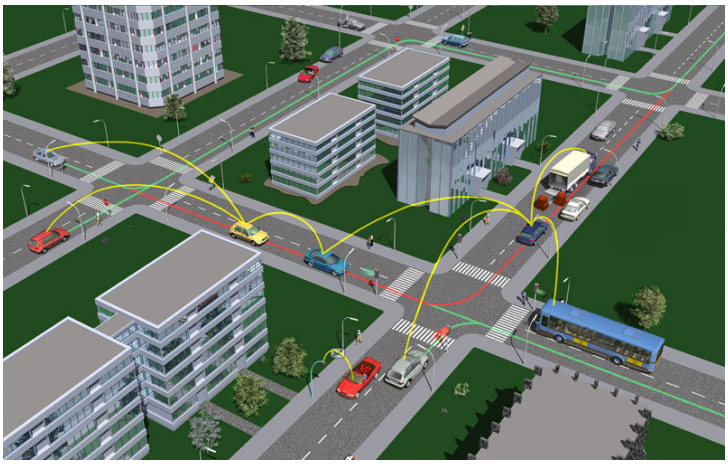
Universidade de Aveiro
Departamento de
Electrónica, Telecomunicações e Informática
2011

Manuel José Alves
Ventura da Silva

Co-projecto em FPGA da MAC IEEE 802.11p para Comunicações Veiculares



Manuel
Ventura

Co-projecto em FPGA da MAC IEEE 802.11p para Comunicações Veiculares

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Electrónica e de Telecomunicações, realizada sob a orientação científica do Professor Doutor Arnaldo Oliveira, Professor Auxiliar do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro, e do Professor Doutor José Alberto Gouveia Fonseca, Professor Associado do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro.

o júri / the jury

presidente / president

Prof. Doutor João Nuno Pimentel da Silva Matos
Professor Associado da Universidade de Aveiro

vogais / examiners committee

Prof. Doutor Arnaldo Silva Rodrigues de Oliveira
Professor Auxiliar da Universidade de Aveiro (Orientador)

Prof. Doutor José Alberto Gouveia Fonseca
Professor Associado da Universidade de Aveiro (Coorientador)

Prof. Doutor João Paulo de Castro Canas Ferreira
Professor Auxiliar do Departamento de Engenharia Electrotécnica e de Computadores da Faculdade de Engenharia da Universidade do Porto

agradecimentos / acknowledgements

Gostaria de agradecer aos professores Arnaldo Oliveira e José Alberto Fonseca, pela orientação oferecida ao longo deste ano, aos colaboradores no desenvolvimento do projecto João Pio e Nelson Silva, aos meus pais e aos colegas de curso que me acompanharam durante estes últimos cinco anos.

Resumo

Com o avanço e disseminação em grande escala de tecnologias de telecomunicações, estas encontram-se em cada vez mais aspectos do nosso dia-a-dia. Recentemente estas tecnologias têm sido aplicadas a veículos, como meio de não só aumentar a segurança na condução como também a comodidade de condutores e passageiros.

Para que as comunicações veiculares sejam uma realidade é necessária a elaboração de standards que permitam o desenvolvimento de plataformas compatíveis de comunicações entre veículos, servindo também como base ao desenvolvimento de aplicações que tirem partido de redes veiculares. Assim, foram propostas as normas IEEE WAVE (*Wireless Access in Vehicular Environments*)em conjunção com a emenda IEEE 802.11p de forma a endereçar algumas das especificidades das redes veiculares, como por exemplo os baixos tempos de conectividade e a natureza altamente dinâmica do meio.

Esta dissertação enquadra-se no âmbito do projecto HEADWAY, cujo objectivo é desenvolver um dispositivo para comunicação entre veículos, dentro do qual foi concebida uma plataforma de desenvolvimento que permitirá a integração de toda a pilha protocolar WAVE, potenciando assim a criação de um sistema de comunicações entre veículos standarizado. A plataforma de desenvolvimento concebida contém uma antena, módulos de RF, circuitos DAC e ADC, uma FPGA, um processador de uso geral e um módulo GPS. Este trabalho encontra-se focado no desenvolvimento e implementação de uma camada MAC em FPGA para comunicações veiculares de acordo com as normas WAVE e IEEE 802.11p.

As diferentes funcionalidades da camada MAC foram divididas em termos de complexidade e tempos de execução, dando origem à divisão da camada de acesso desenvolvida em *Upper* MAC (UMAC) e *Lower* MAC (LMAC). A UMAC será implementada em *software* (C) correndo sobre um microprocessador embutido em FPGA e, em geral, conterá as funcionalidades mais complexas da camada de acesso em termos de algoritmo que não necessitem de tempos de execução extremamente baixos, como por exemplo o processamento e descodificação de tramas. A LMAC será constituída por lógica em *hardware* modelado em VHDL, e executará funções da MAC críticas em termos temporais, como por exemplo o *timestamping* de tramas recebidas, e cálculos complexos que benificiem de operações paralelas, como a computação e verificação de erros através do cálculo de CRC.

A camada MAC desenvolvida foi implementada em FPGA e os seus mecanismos foram validados.

Abstract

The advancements and dissemination of telecommunication technologies has caused them to be employed more and more in our day-to-day life. Recently, these technologies have been applied to vehicles, as a way of not only improving driving safety but also the drivers' and passengers' comfort.

If vehicular communications are to become a reality, communication standards must be created in order to allow the development of compatible communication platforms, while also serving as a basys for application development. The standards IEEE WAVE, alongside the IEEE 802.11p amendment, were proposed in order to meet these demands and address some of the specific issues with vehicular networks, such as short connectivity times and the highly dynamic nature of the propagation environment.

This thesis fits within the HEADWAY project, the goal of which is the creation of a device that will perform communication between vehicles. In order to incorporate every layer of the WAVE (Wireless Access in Vehicular Environments) protocol stack, a development platform was conceived that will enable the creation of a standardized communications system for vehicles. The development platform created features an antenna, RF modules, DAC and ADC circuits, an FPGA, a general purpose microprocessor and a GPS module. This work is focused in the development and implementation in FPGA of a MAC layer in accordance with the WAVE standards.

The MAC layer's different functionalities were divided according to their complexity and execution time, causing our MAC's division in Upper MAC (UMAC) and Lower MAC (LMAC). The UMAC will be implemented in software (C) running in an FPGA embedded microprocessor and will contain the MAC's functions that are more complex, algorithmically speaking, but are not required to be excuted in a very short time interval, such as frame processing and decoding. The LMAC will be implemented by VHDL modeled hardware logic and will perform time critical functions, such as the timestamping of received frames, and complex calculations that benefit from the paralelism offered by hardware logic, such as CRC computation and error checking.

This MAC layer was implemented in an FPGA and its mechanisms were validated.

Conteúdo

Lista de Figuras

Lista de Tabelas

Acrónimos

AC *Access Category*

ACK *Acknowledgment*

AP *Access Point*

AIFS *Arbitrary Inter Frame Space*

BSS *Basic Service Set*

CCH *Control Channel*

CTS *Clear to Send*

CRC *Cyclic Redundancy Check*

CW *Contention Window*

CSMA-CA *Carrier Sense Multiple Access with Collision Avoidance*

DSRC *Dedicated Short Range Communications*

EDCA *Enhanced Distributed Channel Access*

FCC *Federal Communications Commission*

FPGA *Field Programmable Gate Array*

FCS *Frame Check Sequence*

GPRS *General Packet Radio Service*

GPS *Global Positioning System*

HSPA *High Speed Packet Access*

IFS *Inter Frame Space*

ITS *Intelligent Transport Systems*

LLC *Logical Link Control*

LMAC *Lower* MAC

MAC *Medium Access Control*

NAV *Network Allocation Vector*

OBU *On-Board Unit*

OSI *Open Systems Interconnection*

PHY Camada Física

PLB *Peripheral Local Bus*

RF *Radio Frequency*

RSU *Road Side Unit*

RTS *Request to Send*

SCH *Service Channel*

SIFS *Short Inter Frame Space*

SoC *System on Chip*

TAF *Timing Advertisement Frame*

TDMA *Time Division Multiple Access*

UMAC *Upper* MAC

UTC *Coordinated Universal Time*

VANET *Vehicular Ad-hoc Network*

VHDL *VHSIC Hardware Description Language*

VHSIC *Very High Speed Integrated Circuit*

V2I *Vehicle to Infrasructure*

V2V *Vehicle to Vehicle*

WAVE *Wireless Access in Vehicular Environments*

Capítulo 1

Introdução

1.1 Comunicações Veiculares e ITS

O avanço e disseminação em grande escala de comunicações sem fios revolucionou o nosso estilo de vida, oferecendo uma grande comodidade e flexibilidade no acesso a serviços de Internet e várias aplicações baseadas em telecomunicações. Mais recentemente, este tipo de tecnologias começou a ser aplicada a automóveis com a finalidade de melhorar a experiência de viagem de condutores e passageiros. Este tipo de comunicações, ditas Comunicações Veiculares, tem como nós das suas redes os próprios veículos e eventualmente unidades fixas na via: a comunicação entre veículos é designada por *Vehicle to Vehicle* (V2V) enquanto que a comunicação entre um dispositivo instalado na estrada e um veículo diz-se *Vehicle to Infrasructure* (V2I). Os dispositivos colocados dentro de um veículo designam-se vulgarmente por *On-Board Unit*s (OBUs) enquanto que os dispositivos instalados na estrada dizem-se RSUs. Em regra geral, são utilizados dispositivos digitais de comunicação de curto alcance (*Dedicated Short Range Communications* (DSRC)), que permitem comunicações até cerca de 1000m de distância.

O desenvolvimento de sistemas inteligentes de transporte (*Intelligent Transport Systems* (ITS)) segue em paralelo com o avanço em comunicações veiculares. À medida que os veículos e a infraestrutura montada nas vias se tornam mais inteligentes também aumentam os benefícios que as redes veiculares lhes trazem, permitindo um acesso eficiente a informação na estrada.

As redes veiculares têm então como principal finalidade melhorar a segurança para os ocupantes de veículos. Grande parte dos acidentes rodoviários são causados por erros humanos e através de sistemas de condução, prevenção e notificação de acidentes mais inteligentes espera-se reduzir grande parte da sinistralidade verificada hoje em dia. Para além desta componente de segurança, as redes veiculares poderão ainda oferecer conforto aos passageiros, reduzir o tempo que se passa na estrada e, consequentemente, o combustível consumido. É esperado que o desenvolvimento de comunicações veiculares sirva como base para inúmeras aplicações, tanto de segurança como de entertenimento.

Através de comunicações veiculares será possivel a notificação e prevenção automática de colisões: um veículo poderá detectar uma situação anormal na estrada e notificará os veículos na sua vizinhança, oferecendo assim mais tempo de reacção aos condutores. Um automóvel poderá também enviar notificações aos seus vizinhos avisando-os de que acabou de efectuar uma travagem brusca, podendo estes reagir mais rapidamente. Poderão também

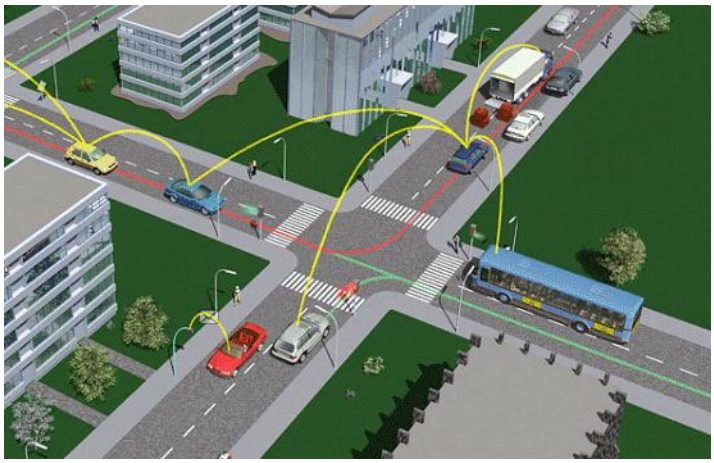


Figura 1.1: Cenário de utilização de comunicações veiculares (retirado de [1]).

ser desenvolvidas aplicações para o auxílio e prevenção de colisões em intersecções, uma das zonas onde ocorre um maior número de acidentes. Será também possível uma melhor gestão de situações de emergência, permitindo um melhor acesso de equipas de socorro a locais de acidentes. Outra possível aplicação será a detecção mais eficiente de zonas de congestionamento na estrada permitindo a condutores e a sistemas de navegação inteligentes alterar as suas rotas de acordo com a informação de tráfego. Informações relativas ao estado do tempo poderão também ser disponibilizadas aos condutores em tempo real, permitindo a preparação destes para situações de neve ou gelo na estrada, por exemplo. O pagamento de portagens sem necessidade de paragem é um sistema de comunicações veiculares que se encontra já em funcionamento em muitas estradas do planeta contribuindo para melhorar a experiência dos condutores ao eliminar uma das maiores zonas de congestionamento em auto estradas. Informação variada pode também ser oferecida a condutores e passageiros de acordo com a sua posição, como por exemplo localização de restaurantes, hotéis e marcos históricos da região onde o veículo se encontra. Poderá ser também oferecido o acesso à internet no automóvel como forma de aumentar a comodidade de passageiros de veículos, principalmente em viagens mais longas.

O desenvolvimento de ITS e comunicações veiculares levou ainda a protótipos de sistemas de condução mais inteligentes. Um destes tipos de protótipo é a condução em pelotão, em que vários veículos circulam autonomamente separados por uma curta distância. Na figura 1.2 está representada uma possível situação num sistema deste tipo, em que um pelotão é liderado por um veículo pesado e um veículo negoceia a sua entrada no pelotão ao mesmo tempo que outro manifesta o seu desejo de o deixar. Este tipo de sistemas permite não só reduzir o combustível consumido através da diminuição do atrito aerodinâmico dos veículos, como reduz o número de acidentes ao retirar o elemento humano da condução e os congestionamentos nas estradas devido à utilização mais eficiente destas. Também já foram desenvolvidos vários veículos de condução completamente autónoma, sem necessidade de um líder de pelotão para circular. Estes sistemas não são possíveis sem uma constante comunicação entre veículos e beneficiarão imenso do desenvolvimento de redes de telecomunicações veiculares.

Apesar da motivação principal para o desenvolvimento de ITS ser o melhoramento da segurança na condução, espera-se o desenvolvimento de aplicações de *infotainment* que tornem

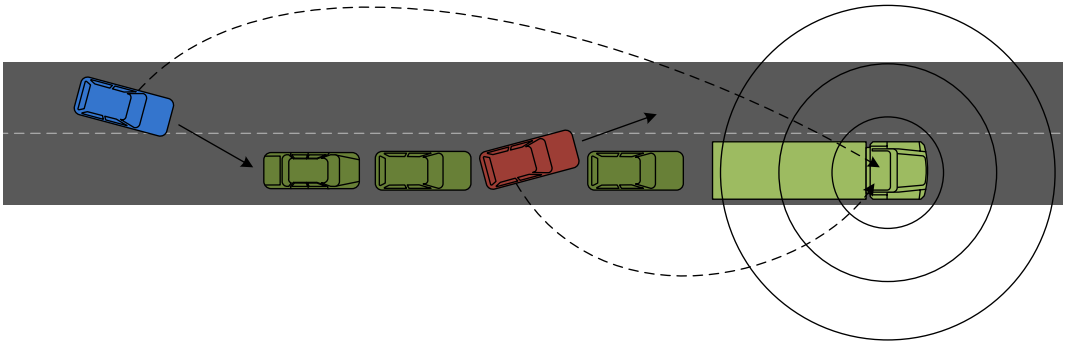


Figura 1.2: Condução em pelotão

estes sistemas mais atractivos para os consumidores levando a uma maior disseminação desta tecnologia e tornando-a atraente comercialmente do ponto de vista do investidor.

Com o desenvolvimento de comunicações veiculares surgem novos desafios decorrentes da natureza dinâmica e móvel destas novas redes. É necessária a criação de novos protocolos que permitam o correcto funcionamento dos vários serviços, desde as aplicações de segurança até ao *infotainment*, para que o potencial das comunicações veiculares seja completamente explorado.

1.2 Camada MAC no Contexto da Pilha Protocolar WAVE/ 802.11p

Os sistemas de comunicações actuais, devido à sua elevada complexidade são muitas vezes divididos em pequenas partes, denominadas camadas. Esta divisão agrupa funções semelhantes dentro de um sistema de comunicações simplificando a compreensão, desenvolvimento e normalização destes sistemas. Uma destas divisões de sistemas de comunicações é o modelo *Open Systems Interconnection* (OSI), cujas diferentes camadas se encontram representadas na figura 1.3. De acordo com este modelo, cada camada comunica com as camadas directamente acima e abaixo dela. Uma das divisões da camada *Data Link* do modelo OSI é a *Medium Access Control* (MAC).

A principal função da camada de controlo de acesso ao meio (MAC) é, como o nome indica, efectuar o controlo do acesso a um meio físico partilhado por vários nós ou terminais de uma rede multiponto. Este controlo de acesso é feito baseado em protocolos de acesso múltiplo que permitem então que várias estações se encontrem ligadas ao mesmo meio físico. Os protocolos de acesso múltiplo podem ainda detectar e evitar colisões de dados. Os mecanismos de acesso ao meio baseiam-se na multiplexagem do canal físico na frequência, no tempo ou em ambos.

Para assegurar a compatibilidade entre os vários sistemas de comunicações, cada uma das camadas do modelo OSI obedece a protocolos e normas que especificam, entre outros aspectos, características físicas dos sinais trocados entre sistemas, sequências de trocas de mensagens ou ainda mecanismos de detecção e correcção de erros.

Em comunicações veiculares, os vários protocolos utilizados estão agrupados na pilha protocolar *Wireless Access in Vehicular Environments* (WAVE) (*Wireless Access for Vehicular Environments*), composto pelas normas IEEE 1609.1, IEEE 1609.2, IEEE 1609.3, IEEE 1609.4 e IEEE 802.11p. As diferentes camadas da pilha protocolar WAVE estão representadas

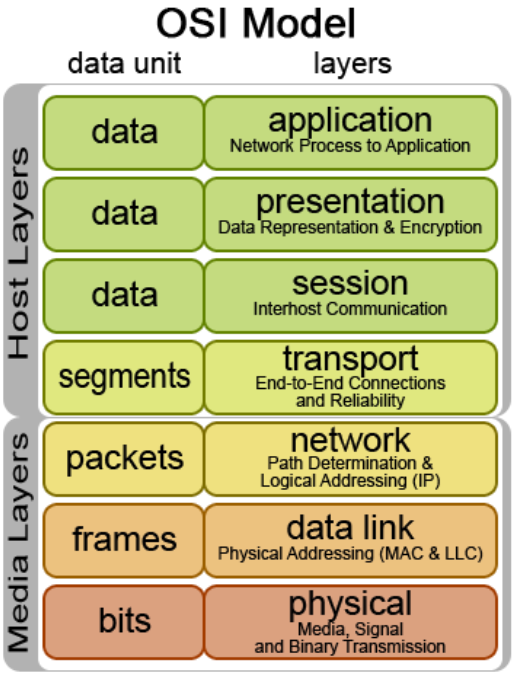


Figura 1.3: Camadas do modelo OSI (retirado de [2]).

na figura 1.4, bem como a sua correspondência com as camadas do modelo OSI.

Os standards propostos para comunicações veiculares pelo IEEE para a camada MAC têm como base o IEEE 802.11p [12], uma emenda realizada ao standard IEEE 802.11 [13] utilizado em WLANs convencionais. Por cima deste standard foi criada a norma IEEE 1609.4 [4] que permite a operação da MAC 802.11p em modo multi-canal. Foi definido nesta norma a existência de um canal de controlo (*Control Channel* (CCH)) e canais de serviço (*Servive Channel* - *Service Channel* (SCH)). Os vários canais encontram-se em frequências distintas adjacentes e o acesso 'básico' é feito alternando-se periodicamente entre o CCH e um SCH.

Em comunicações veiculares a camada MAC possui algumas especificidades: é necessário garantir uma correcta troca de mensagens entre terminais da rede para que informações críticas sobre segurança sejam sempre entregues, mesmo em ambientes extremamente congestionados. Outra especificidade da camada de acesso em redes veiculares deve-se à constante mudança de topologia da rede fruto da grande mobilidade dos seus nós, fazendo com que os terminais da rede se encontrem em conectividade durante um período de tempo muito curto quando comparado com o tempo de conectividade em WLANs. As condições do canal físico também se alteram bastante rapidamente, uma vez que a velocidade relativa entre nós comunicantes poderá ser grande causando a perda súbita de conectividade entre eles. Outras condições poderão ainda ser subitamente alteradas entre nós que se encontram a comunicar como por exemplo o surgimento de obstáculos súbitos entre eles (outros veículos, árvores, etc.). A densidade de nós das redes também variará imenso, já que uma estrada secundária não terá a mesma densidade de veículos que uma via de acesso a uma grande cidade.

Um dos grandes reptos da camada MAC será garantir a transmissão e recepção de mensagens críticas de segurança de uma maneira determinística mesmo em meios altamente densos.

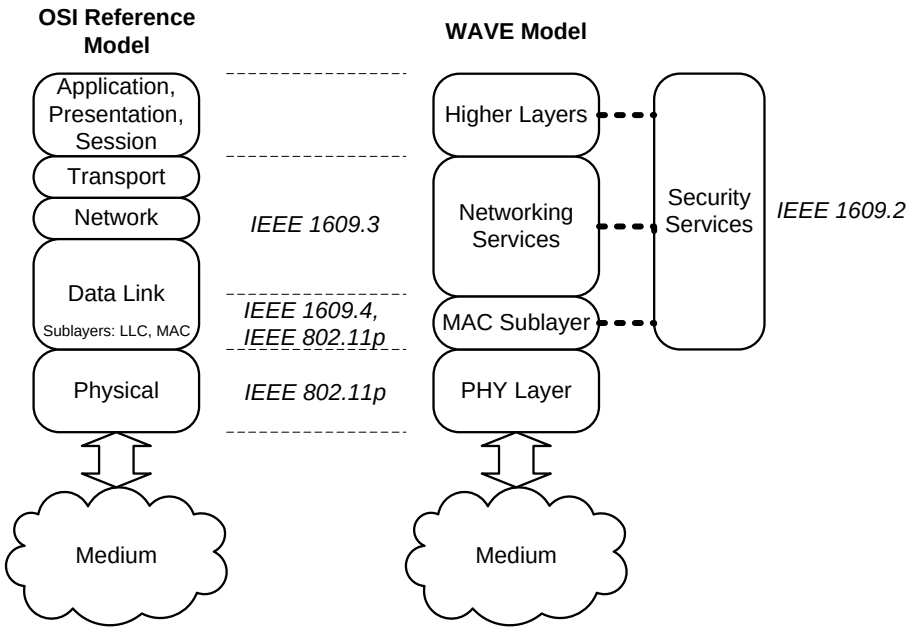


Figura 1.4: Pilha protocolar WAVE e correspondência das suas camadas com as do modelo OSI (adaptado de [3]).

Por exemplo, se um veículo realizar uma travagem súbita este poderá notificar os veículos na sua vizinhança deste evento. No entanto, se um automóvel efectuou uma paragem repentina é provável que outros veículos perto deste também a tenham efectuado originando um grande fluxo de mensagens neste período e um grande potencial para colisão de pacotes. Esta é uma situação que a camada de acesso tem de ter em conta para garantir a segurança de todos os ocupantes de veículos. Em [5] foi verificado que os standards propostos prioritizam correctamente as mensagens, mas num ambiente mais congestionado o canal é utilizado de uma maneira ineficiente e ocorre um grande número de colisões entre as mensagens de maior prioridade.

As diferentes aplicações a correr sobre comunicações veiculares têm ainda necessidades de largura de banda e latência bastante distintas. Enquanto que mensagens de segurança têm de ser propagadas com um tempo máximo de atraso, aplicações de entertenimento como seja o *streaming* de um vídeo têm requisitos de largura de banda mínima para funcionarem correctamente.

A camada de acesso em comunicações veiculares tem assim o desafio de oferecer um acesso ao meio eficiente durante os baixos tempos de conectividade, moderar o acesso ao meio partilhado tendo em conta a constante chegada e partida de estações da rede e conciliar a transmissão de mensagens de segurança a uma baixa latência com a largura de banda necessária às aplicações de *infotainment*.

1.3 Objectivos

No contexto deste projecto pretende-se realizar o desenvolvimento de uma plataforma de desenvolvimento que integre as várias camadas da pilha protocolar WAVE, servindo como

base para a construção de um sistema de comunicações para veículos. No contexto da camada MAC os osbjectivos gerais são:

- Especificação da camada MAC;
- Divisão das funcionalidades da MAC IEEE 802.11p em *Upper* MAC (UMAC), *software* correndo num microprocessador, e *Lower* MAC (LMAC), lógica *custom* implementada em *hardware*;
- Modelação da *Lower* MAC, através da linguagem *VHSIC Hardware Description Language* (VHDL);
- Desenvolvimento da *Upper* MAC, recorrendo a *software* (C) correndo num processador de uso geral embutido em *Field Programmable Gate Array* (FPGA);
- Integração da LMAC e UMAC recorrendo ao ambiente integrado de projecto Xilinx EDK;
- Implementação em FPGA da camada MAC desenvolvida;
- Validação dos mecanismos implementados.

1.4 Organização da Dissertação

Esta dissertação contém, além desta introdução, os seguintes capítulos:

- **Capítulo 2 - Estado da Arte** - Neste capítulo é feito um levantamento dos protocolos e normas mais recentemente propostos no âmbito da camada de acesso para comunicações veiculares, assim como uma vista sobre várias avaliações aos standards IEEE e os mecanismos principais de controlo de acesso ao meio descritos nas normas.
- **Capítulo 3 - Especificação da MAC** - Neste capítulo é especificada a divisão entre LMAC e UMAC, e as diferentes funcionalidades implementadas em cada uma. É apresentado o modelo de programação simplificado da LMAC, e os procedimentos necessários à UMAC para poder controlar a LMAC.
- **Capítulo 4 - Implementação da MAC** - Neste capítulo é apresentada a arquitectura da camada MAC desenvolvida, e, em maior detalhe, a arquitectura da *Lower* MAC e a implementação de cada um dos seus módulos internos.
- **Capítulo 5 - Validação** - Neste capítulo são especificados os vários testes realizados à MAC, ao nível da verificação funcional e do seu desempenho e correcção temporal, e apresentados e discutidos os resultados obtidos.
- **Capítulo 6 - Conclusão** - Neste capítulo é feito um resumo do trabalho realizado, a discussão final dos resultados obtidos e possíveis trabalhos futuros.

É ainda apresentado o seguinte apêndice, fornecendo informação adicional sobre a implementação do sistema:

- **Apêndice A - Modelo de Programação** - Neste apêndice é apresentado em maior detalhe o modelo de programação da LMAC, incluindo uma descrição de todos os registos existentes, dos procedimentos a realizar de forma a operar correctamente a LMAC e as *drivers* criadas.

Capítulo 2

Estado da Arte

2.1 Introdução

De forma a habilitar comunicações veiculares o IEEE desenvolveu então um conjunto de standards, denominado WAVE, assentando sobre as especificações para as camadas Camada Física (PHY) e MAC introduzidas pela emenda IEEE 802.11p. Escolheu-se utilizar esta emenda por diversas razões, entre elas:

- **Comunicações DSRC na gama dos 5.9 GHz** - as comunicações DSRC, pelo seu curto a médio alcance, permitem a utilização da diversidade espacial em redes veiculares, melhorando a eficiência de utilização do canal físico. A gama de frequências utilizada possibilita também a transferência de dados a ritmos elevados;

- **Comunicações entre estações fora do contexto de um *Basic Service Set* (BSS)** - a norma IEEE 802.11p prevê a troca de mensagens entre estações sem que estas tenham de aderir a um BSS (ou rede). Devido aos baixos tempos de conectividade entre estações foi necessário introduzir alterações a este nível, de forma a eliminar o *overhead* verificado sempre que uma estação tem de aderir a uma rede.

Os novos standards WAVE introduzem também novas funcionalidades ao nível da camada MAC, nomeadamente a norma IEEE 1609.4. É especificada a utilização de vários canais de comunicação em diferentes frequências adjacentes:

- **Canal de Controlo** - neste canal são trocadas as mensagens críticas de segurança e anunciados serviços que serão oferecidos nos diversos canais de serviço;

- **Canais de Serviço** - canais de uso geral, onde é possível a troca de serviços entre estações e de mensagens não críticas;

A abordagem multi-canal escolhida permite explorar adicionalmente a diversidade espacial da rede. Uma vista mais detalhada sobre os standards IEEE para comunicações veiculares é dada na secção 2.3.

Apesar dos standards propostos específicos para a camada de acesso em comunicações veiculares, existe ainda espaço para melhoramentos e várias sugestões a este nível foram já apresentadas.

2.2 Soluções propostas

Como já foi apresentado no capítulo anterior, os standards actuais para comunicações veiculares deixam algumas questões em aberto, não só ao nível de optimizações dentro do próprio standard como em questões mais relacionadas com as especificidades de redes veiculares. Foram identificados vários problemas decorrentes deste tipo de redes:

- Baixos tempos de conectividade entre nós da rede;
- Utilização pouco eficiente do canal de comunicação;
- Cumprimento de restrições temporais e de largura de banda para o correcto funcionamento de aplicações.

Ao longo deste capítulo irão ser apresentadas algumas das soluções propostas no âmbito de acesso ao meio em comunicações veiculares, bem como uma vista geral sobre os *standards* actuais.

2.2.1 Tempos de Conectividade

Um dos problemas identificados parte do baixo tempo de conectividade entre OBUs em movimento e RSUs. De forma a mitigar este problema foram propostos vários protocolos que permitem a utilização de nós *relay* que auxiliam na comunicação de OBUs que se encontram fora do raio de cobertura de uma RSU, com a própria RSU. Consegue-se assim um aumento da cobertura efectiva de uma RSU e do tempo de conectividade entre estações.

2.2.2 Utilização Eficiente do Canal

Uma questão decorrente do acesso multi-canal especificado na norma IEEE 1609.4 prende-se com a utilização pouco eficiente do canal, uma vez que o tempo dedicado ao canal de controlo e aos canais de serviço é dado por um valor fixo. As soluções propostas passam por esquemas em que os intervalos de controlo e de serviço têm durações mínimas e máximas, variando de acordo com a situação: se existir um grande número de mensagens de segurança o intervalo de controlo é extendido, caso contrário dar-se-á mais tempo ao intervalo de serviço para que outras aplicações possam correr suavemente. Outra abordagem passa por diferentes modos de acesso ao canal de acordo com a proximidade ou não de uma OBU com uma RSU.

2.2.3 Funcionamento de Aplicações

O cumprimento das restrições necessárias para as várias aplicações, tanto em termos de latência como em termos de largura de banda é também alvo de estudo em diferentes trabalhos. Em termos de entrega determinística de mensagens de segurança foram sugeridas abordagens onde RSUs poderiam reservar *slots* temporais durante os quais poderiam difundir mensagens de segurança sem contenção do meio. A mesma abordagem foi utilizada para o funcionamento de aplicações que necessitem de uma maior largura de banda. Outras abordagens utilizam um período de *polling* de OBUs por parte da RSU como forma de os nós da rede conseguirem aceder ao meio sem qualquer tipo de contenção durante esse período.

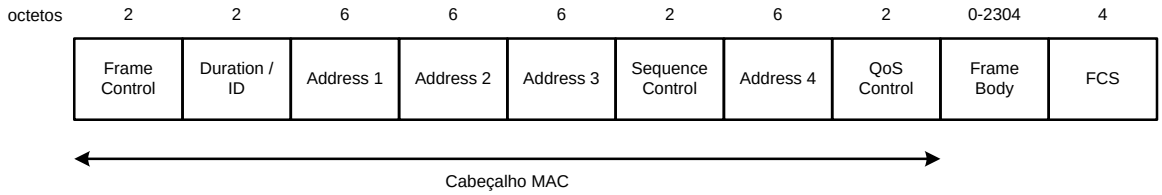


Figura 2.1: Formato de uma trama MAC 802.11.

2.3 Standards IEEE 802.11p e IEEE 1609.4

A camada MAC para comunicações veiculares é baseada no protocolo IEEE 802.11 [13] com a emenda IEEE 802.11p [12] em conjunto com a extensão IEEE 1609.4 [4] que oferece a base para a operação multi-canal dos sistemas WAVE.

Uma possível arquitectura para a camada MAC é ilustrada na figura 2.4. A entidade MAC 802.11p utilizada em dispositivos WAVE é a MAC especificada no standard IEEE 802.11, com as emendas descritas em IEEE 802.11p. Existem duas entidades MAC 802.11p no exemplo apresentado: uma para o canal de controlo e outra para o canal de serviço. Os dados são prioritizados de acordo com a sua categoria de acesso (*Access Category* (AC)), como indicado pelas filas na figura 2.4, que oferecem diferentes parâmetros para a contenção do meio e transmissão de dados para tramas de diferente prioridade.

Na figura 2.1 estão representados os campos possíveis para uma trama MAC. Todas as tramas possuem os campos *Frame Control*, *Duration / ID* e *Frame Check Sequence*. Todos os outros campos estarão ou não presentes de acordo com o tipo de trama em questão.

Nesta secção serão abordados os seguintes aspectos da MAC WAVE:

- Acesso ao meio;
- Contenção interna;
- Operação multi-canal;
- Tipos de acesso ao canal;
- Sincronização.

2.3.1 Acesso ao Meio

O acesso ao meio utiliza *Carrier Sense Multiple Access with Collision Avoidance* (CSMA-CA), onde um nó da rede verifica se o meio está livre antes de emitir uma mensagem. Caso o meio seja detectado livre o nó terá de esperar um certo intervalo de tempo (*Inter Frame Space* (IFS)) mais um intervalo aleatório de *backoff*, que varia entre 0 e o valor de uma janela de contenção (*Contention Window* (CW)), para poder iniciar a sua transmissão. Se após este período o nó não detectar nenhuma mensagem a ser transmitida, começará a sua própria transmissão; caso seja detectado que o meio se encontra ocupado antes dos períodos de IFS e de *backoff* experirarem, o nó irá esperar até que o meio se encontre livre novamente e repetirá o processo descrito. Este processo é descrito na figura 2.2 a).

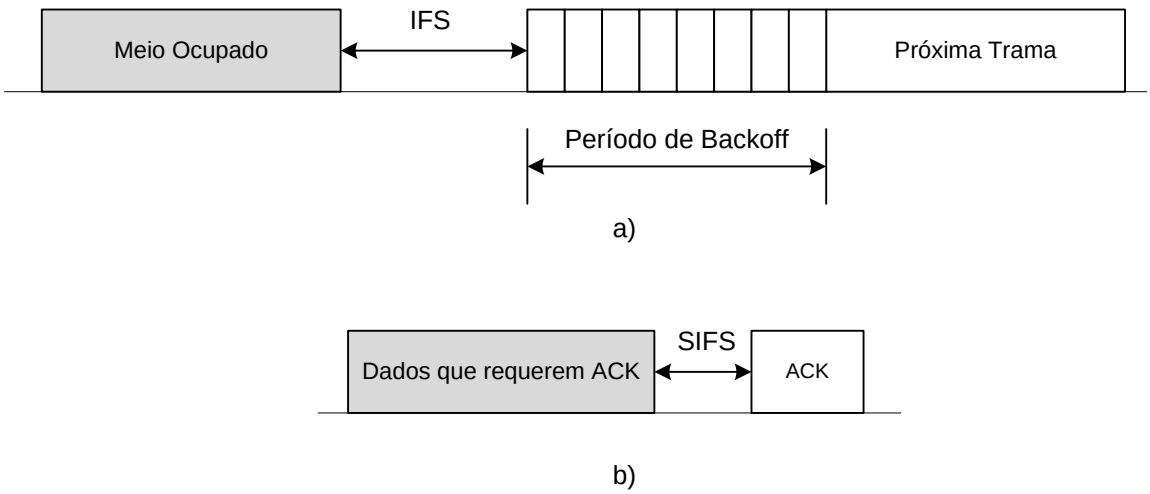


Figura 2.2: Mecanismo de Inter Frame Space e período de Backoff.

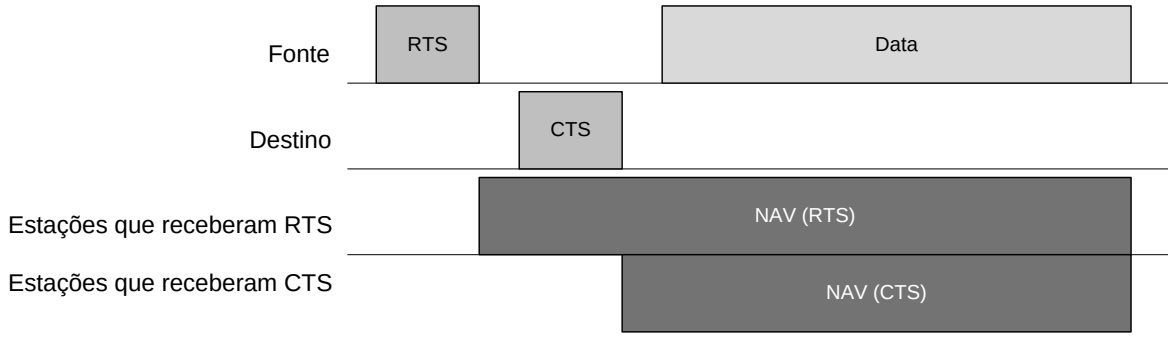


Figura 2.3: Troca de mensagens RTS e CTS por parte de duas estações e reserva virtual do meio nas restantes estações que capturam pelo menos um destes dois pacotes.

Um caso particular de IFS é o *Short Inter Frame Space* (SIFS), tempo utilizado quando uma MAC tem de responder, por exemplo, com uma trama do tipo *Acknowledgment* (ACK) a uma trama recentemente recebida. Neste caso, a estação que enviará a reposta imediatamente após a ter decorrido o tempo SIFS, sem um período de *backoff*, de modo a impedir a entrada no meio de outras estações que desejem enviar uma nova trama (Figura 2.2 b)).

Para resolver o problema do terminal invisível é utilizado um *handshake* no envio de mensagens através de tramas *Request to Send* (RTS) e *Clear to Send* (CTS), exemplificado na figura 2.3. Através destes dois pacotes é também feita uma reserva virtual do meio por parte das estações que os enviam. Estas duas tramas possuem um campo denominado *Duration*, que contém a duração em microssegundos da transferência que se irá proceder e é utilizado para actualizar o valor de um *Network Allocation Vector* (NAV). O NAV é decrementado a cada microssegundo e uma estação apenas pode aceder ao meio quando o seu NAV é zero e a sua camada PHY determina que o meio não se encontra ocupado.

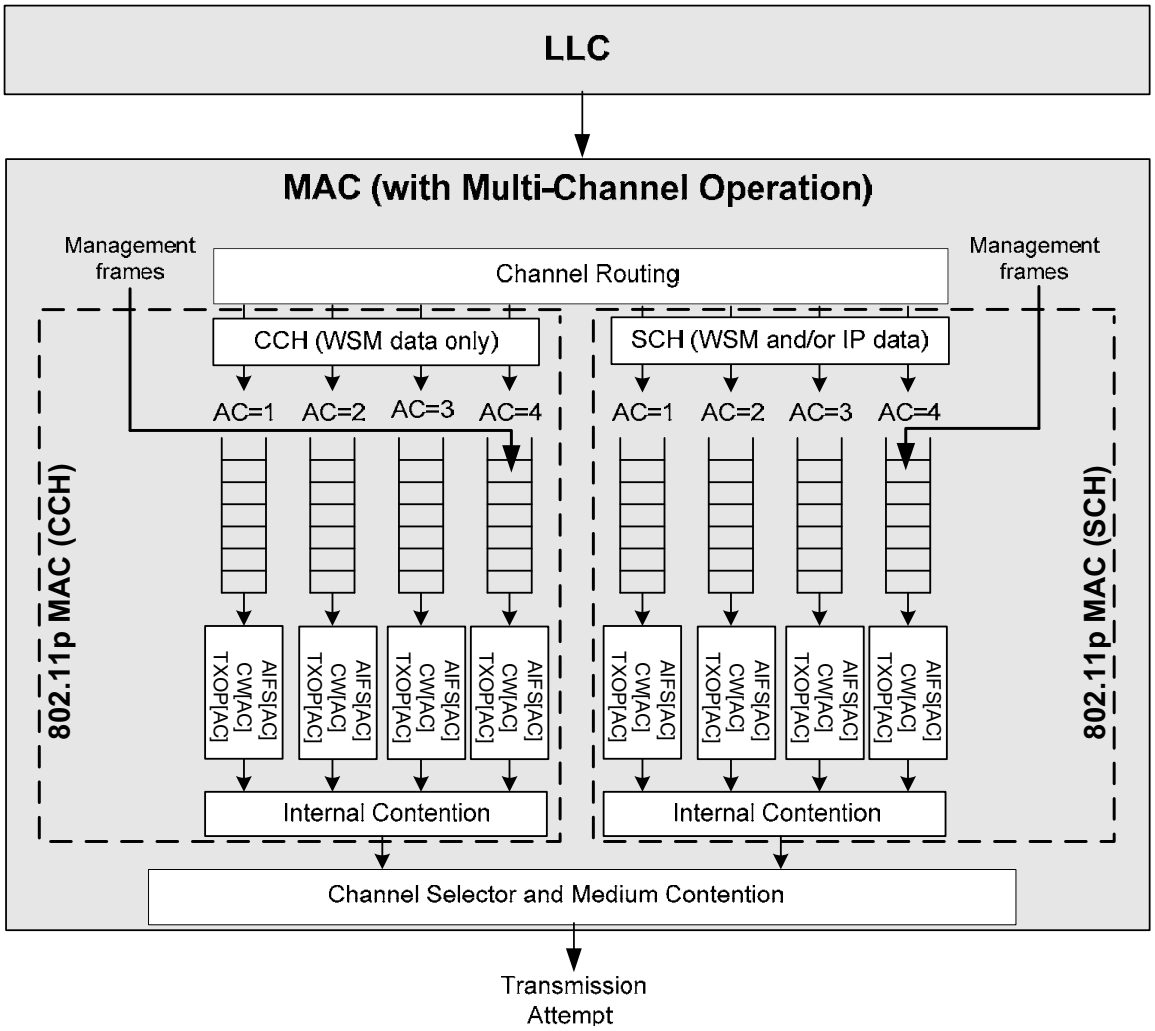


Figura 2.4: Exemplo de arquitectura para camada MAC 802.11p (imagem retirada de [4]).

Categoria de Acesso	CWmin	CWmax	AIFSN
AC0	aCWmin	aCWmax	9
AC1	aCWmin	aCWmax	6
AC2	(aCWmin+1)/2+1	aCWmin	3
AC3	(aCWmin+1)/4+1	(aCWmin+1)/2+1	2

Tabela 2.1: Parâmetros EDCA propostos em IEEE 802.11p para comunicações veiculares.

2.3.2 Contenção Interna

A contenção interna de tramas a ser enviadas é feita com base no mecanismo (*Enhanced Distributed Channel Access* (EDCA)) descrito na norma IEEE 802.11 [13]. As mensagens a ser enviadas são divididas em 4 categorias de acesso, sendo AC0 a categoria de menor prioridade e AC3 a de maior prioridade. Na figura 2.4 pode-se observar a existência de 4 filas para o canal de controlo e 4 filas para os restantes canais de serviço. Isto deve-se ao facto de certas mensagens poderem ser enviadas apenas no canal de controlo, enquanto que outras apenas poderão ser transmitidas num canal de serviço. O valor do período de *backoff* e o *Inter Frame Space* (intervalo mínimo entre tramas no meio) também dependem da categoria de acesso, tomando valores mais baixos para mensagens mais prioritárias e mais altos para tramas com menor prioridade. Os parâmetros EDCA para as várias categorias de acesso estão descritos na norma IEEE 802.11p [12]:

- **CWmin** - Valor mínimo da janela de contenção para cada categoria de acesso;
- **CWmax** - Valor máximo da janela de contenção para cada categoria de acesso;
- ***Arbitrary Inter Frame Space* (AIFS)** - Intervalo entre tramas para cada categoria de acesso, em unidades de *slots* temporais (definidos na norma);
- **aCWmin e aCWmax** - constantes globais indicando os valores mínimo e máximo da janela de contenção numa dada estação.

Por omissão e de acordo com a norma IEEE 802.11p os parâmetros EDCA tomam os valores apresentados na table 2.1.

Na figura 2.5 está representado um exemplo de contenção ao canal por uma estação com mensagens nas 4 categorias de acesso. Como tanto o período de *backoff* como o AIFS é mais baixo para a categoria de acesso 3, esta terá acesso ao meio mais cedo. Quando a mensagem da categoria de acesso é transmitida para o meio, este é declarado como estando ocupado fazendo com que os períodos de *backoff* para as restantes categorias de acesso não sejam decrementados. Quando o meio fica livre novamente, as diferentes categorias de acesso esperarão novamento pelo seu AIFS e continuarão a decrementar o seu valor de *backoff* anterior sem ser necessário calcular um novo.

No caso de ocorrer uma colisão interna, i.e. o período de *backoff* expirar simultaneamente para duas categorias de acesso diferentes dentro de uma mesma estação, a mensagem passada à camada PHY será a correspondente à categoria de acesso mais prioritária de entre as que colidiram. Para as restantes categorias de acesso será calculado um novo período aleatório de *backoff* com uma janela de contenção maior.

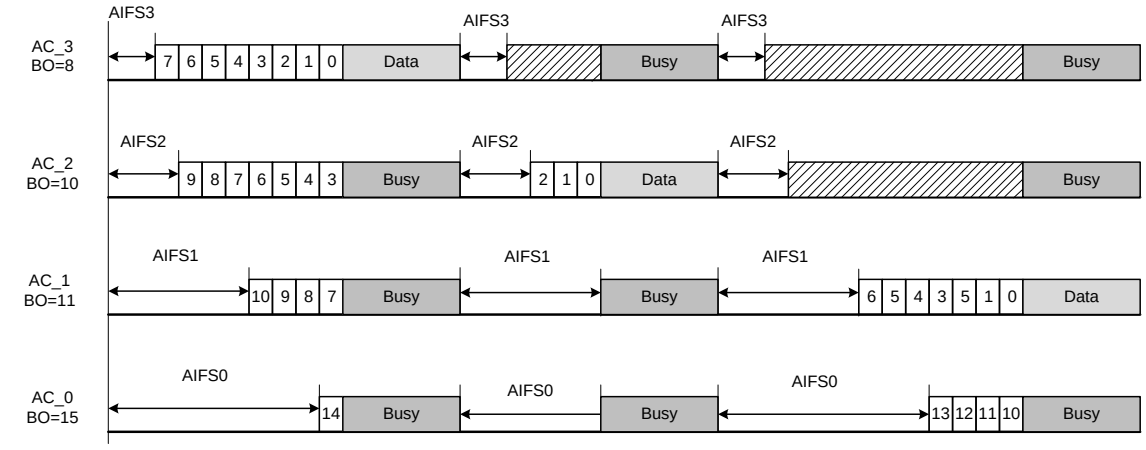


Figura 2.5: Exemplo de contenção interna pelo canal com o mecaniscmo EDCA.

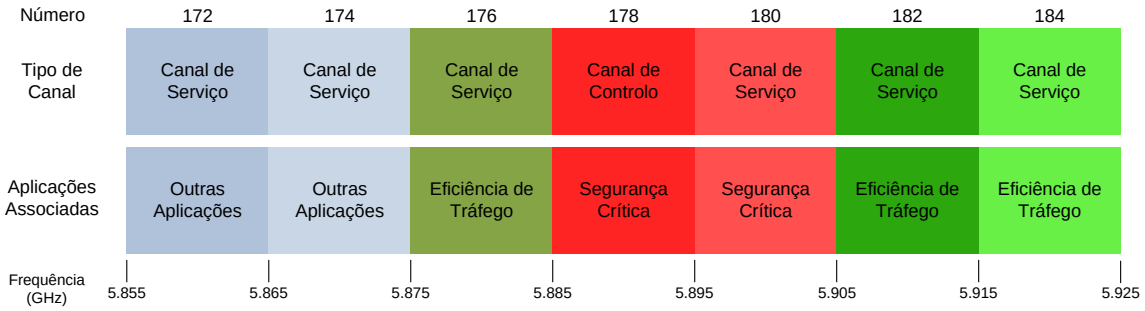


Figura 2.6: Conjunto de Canais Definidos no standard WAVE.

Se fôr detectada uma colisão no meio, por exemplo pela não recepção de uma trama ACK, a trama será reenviada com um novo período de *backoff* calculado através de uma maior janela de contenção, como acontece no caso de colisões internas.

2.3.3 Operação Multi-Canal

Na norma IEEE 1609.4 é definida a operação multi-canal da MAC para sistemas WAVE. Com vista ao desenvolvimento de comunicações e redes veiculares a *Federal Communications Commission* (FCC) nos Estados Unidos alocou a banda de 5.850GHz a 5.925GHz para comunicações DSRC, banda essa que é ainda dividida em 7 canais de 10MHz, cada um com um fim bem definido. Esta atribuição está definida no standard WAVE e está representada na figura 2.6. Na Europa, a banda alocada é a de 5.855 GHz a 5.925 GHz.

O canal de controlo será o canal a utilizar no envio de mensagens de segurança crítica ou anúncios de serviços a ser oferecidos nos diversos canais de serviço. Os canais de serviço (6) terão um uso geral, desde troca de informações relevantes entre veículos e prestação de serviços.

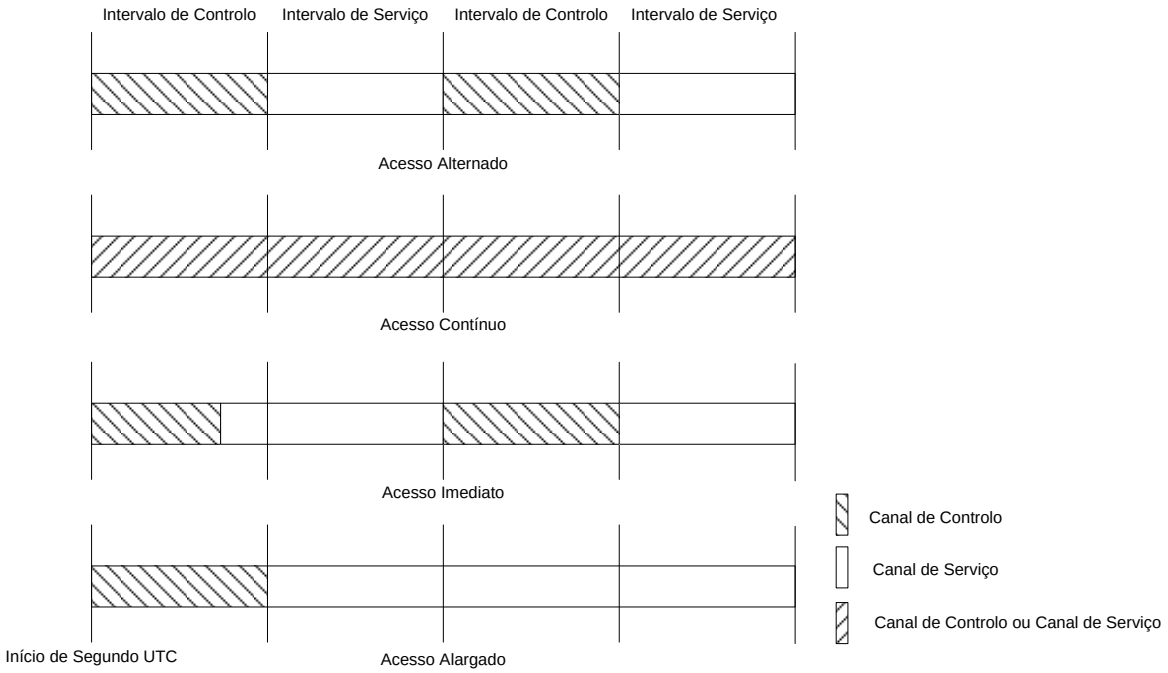


Figura 2.7: Vários tipos de acesso aos canais de controlo e de serviço.

2.3.4 Tipo de Acesso

O acesso aos canais é feito em modo *Time Division Multiple Access* (TDMA), onde os dispositivos WAVE poderão aceder ao canal de controlo e a um canal de serviço de forma periódica e alternada. Os intervalos de tempo standard dedicados a cada canal têm uma duração de 50ms tanto para o canal de controlo como para o canal de serviço, estando associado ao início de um segundo *Coordinated Universal Time* (UTC) o início de um *Sync Interval*, denominação dada ao intervalo de tempo consituído por um intervalo de controlo e um intervalo de serviço. Foi dado este valor à duração dos intervalos de controlo e de serviço para que os veículos possam transmitir mensagens de segurança no canal de controlo a cada 100ms, o tempo de reacção médio de um adulto. De cada vez que ocorre uma mudança de canal, é necessário aguardar um tempo de guarda para que todas as estações tenham tempo de sintonizar no canal certo. Este tempo de guarda tem em conta erros na sincronização e o tempo máximo que uma camada PHY pode demorar a efectuar uma mudança de canal. Ambos os acessos estão esquematizados na figura 2.7. O standard reconhece que durante os tempos de guarda, imediatamente após a mudança de canal por parte dos nós, poderá existir uma grande competição pelo acesso ao meio. Assim, durante o tempo de guarda os nós devem assumir que o meio se encontra ocupado para que possa ocorrer o período de *backoff* aleatório associado ao CSMA-CA.

2.3.5 Sincronização

Para que um dispositivo WAVE possa aceder correctamente ao canal é então necessária uma sincronização global pelo tempo UTC. Se um dispositivo não estiver sincronizado com um erro máximo, não irá conseguir efectuar a operação multi-canal descrita na norma IEEE

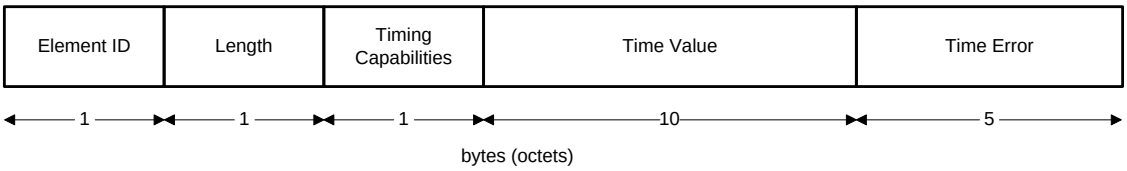


Figura 2.8: Informação de sincronização contida numa TAF.

1609.4. Mensagens para auxiliar a sincronização de estações estão também descritas na norma IEEE 1609.4, em pacotes denominados *Timing Advertisement Frame* (TAF). Na figura 2.8 apresenta-se parte da informação contida numa trama TAF.

É também prevista na norma a possibilidade de sincronização da estação através de *Global Positioning System* (GPS).

2.4 Avaliação do Desempenho dos Standards

Um dos problemas identificados com a norma IEEE 1609.4 é a grande probabilidade de colisões no início de intervalos CCH. Em [14] foi analisada a difusão de mensagens de segurança num ambiente congestionado. Se existir um elevado número de veículos numa dada área e estes tiverem mensagens de segurança calendarizadas para serem enviadas no próximo intervalo de controlo, ocorrerão inevitavelmente colisões entre os pacotes, apesar do período aleatório de *backoff*. É referido que as aplicações de segurança deverão ter este factor em conta para poderem funcionar correctamente. Assume-se em [14] que as mensagens de segurança são difundidas em modo broadcast e portanto não utilizam o *handshake* RTS-CTS. Sendo assim, a estação responsável pelo envio da mensagem de segurança não saberá se a sua mensagem foi recebida correctamente por todos os nós da rede.

Em [5] foi simulado o esquema de contenção interna da MAC 802.11p. Para uma rede com 300 nós, com os intervalos médios entre geração de mensagens pelas camadas superiores descritos na tabela 2.2, o número de mensagens propriamente enviadas para o meio decresce bastante para pacotes das categorias de acesso menos prioritárias. É de crer que à medida que o número de nodos aumenta também as categorias de acesso mais altas sofrerão deste problema. Ainda assim, para uma rede com até 300 nós não existe grande variação no total de mensagens enviadas para categorias de acesso de segurança podendo-se concluir que o mecanismo EDCA consegue prioritizar correctamente este tipo de mensagens. No mesmo trabalho, foi simulado o atraso de mensagens em função do número de nós da rede e os resultados estão representados na figura 2.9. Verificou-se que para redes com mais de 200 nós o atraso observado é superior ao máximo admitido para mensagens de segurança (100 ms).

O *throughput* e atraso de mensagens foi também simulado tanto em [5] como em [6]. Verificou-se que o *throughput* de mensagens no canal de controlo sofre bastante com o aumento de veículos na rede, como se pode observar na figura 2.10. O atraso das mensagens de segurança no canal de controlo (a preto na figura 2.11) mantém-se baixo até que o tempo entre o envio de mensagens baixa para 1ms, momento a partir do qual o atraso verificado para esta categoria de acesso no canal de controlo ultrapassa o atraso máximo permitido para mensagens de segurança V2V.

Em [15] é analisada a utilização de DSRC para aplicações não críticas, partindo da fia-

Categoria de Acesso	Tempo médio entre mensagens	Mensagens por segundo
AC3	100 ms	10.0
AC2	100 ms	10.0
AC1	90 ms	11.1
AC0	45 ms	22.2

Tabela 2.2: Parâmetros utilizados para simulação em [5].

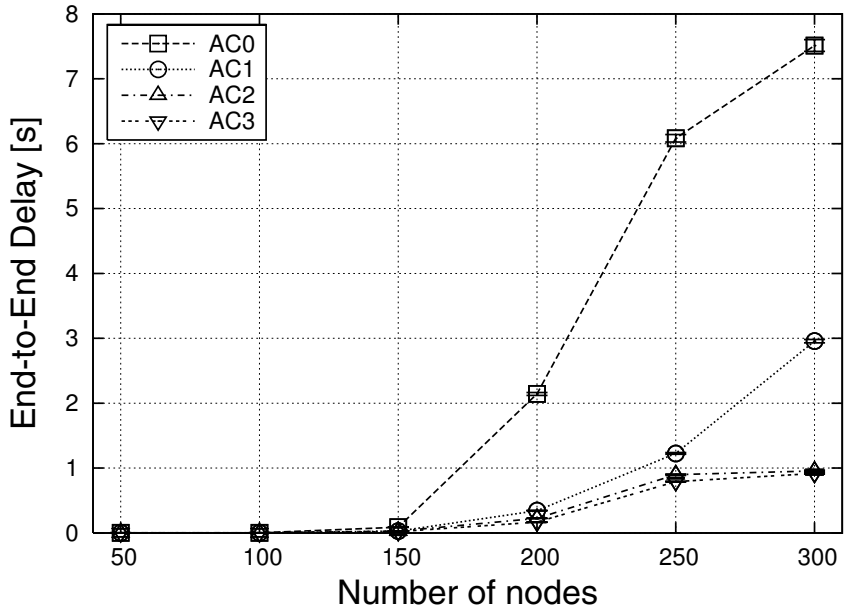


Figura 2.9: Atraso das mensagens em função do número de nós da rede [5].

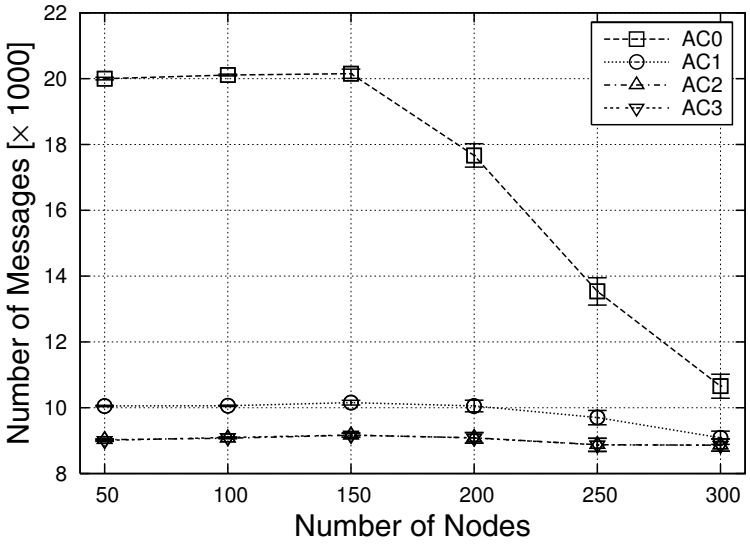


Figura 2.10: Número de mensagens entregues em funçao do número de nós na rede [5].

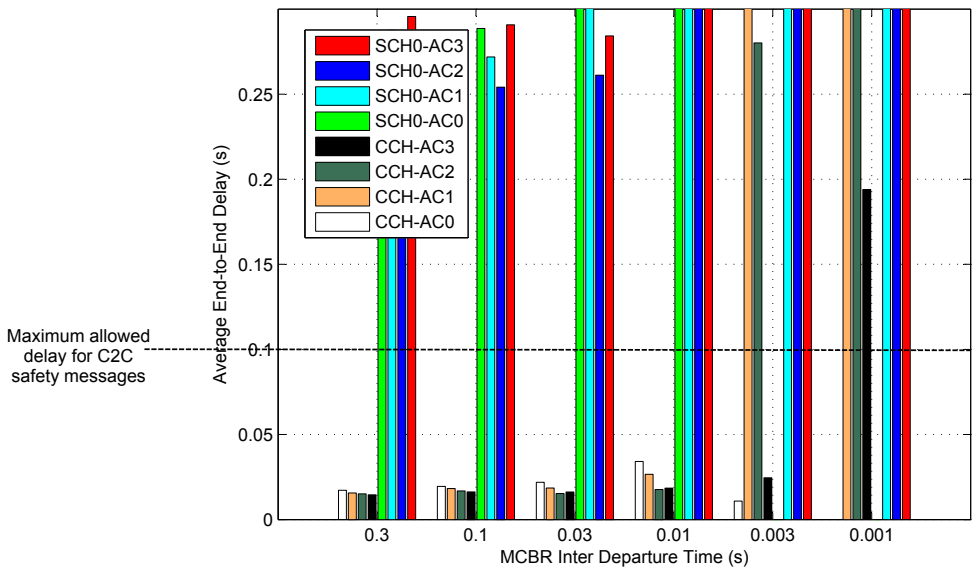


Figura 2.11: Atraso de mensangens das várias categorias de acesso para CCH e SCH0 [6].

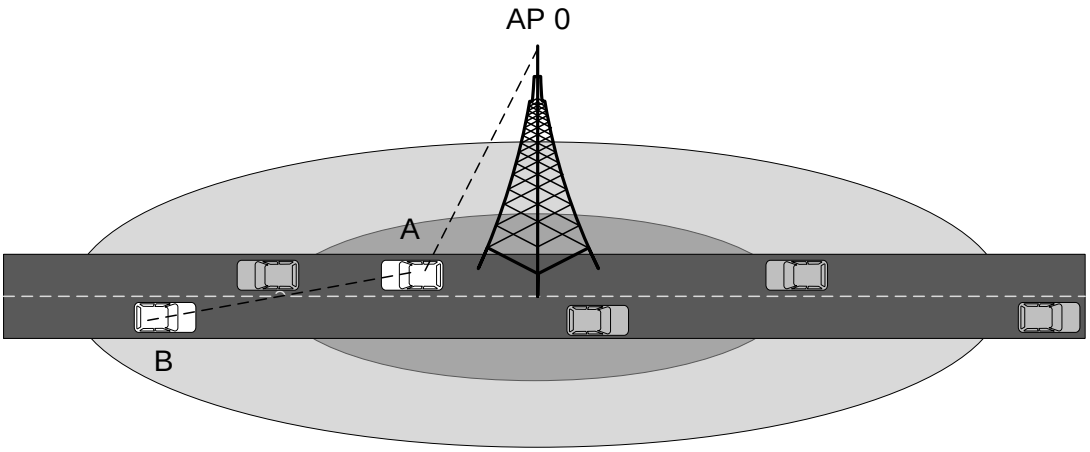


Figura 2.12: Cenário de transmissão cooperativa

bilidade da recepção de mensagens de segurança, e explorou-se a hipótese da utilização de diferentes durações para os intervalos de serviço e de controlo de acordo com a densidade de tráfego. Verificou-se que num meio congestionado, para se obter uma percentagem de recepção de mensagens de segurança da ordem dos 95% é necessário dedicar todo o tempo disponível para o canal de controlo. Este efeito pode ser mitigado pela utilização de repetições no envio das mensagens de segurança. Para um número de repetições das mensagens de segurança óptimo, será possível aumentar a fiabilidade para ambientes com alta densidade para um valor que permita algum tempo de operação ao canal de serviço. É também investigada a transferência de ficheiros na estrada. Tomando como exemplo um ficheiro mp3 de 3.65MB, calculou-se serem necessários mais de 20 segundos para a transferência deste, caso a percentagem de tempo dedicada ao canal de serviço fosse demasiado baixa. Um veículo que viaje a 90km/h atravessaria 500m em 20 segundos, o que significa que não lhe seria possível completar a transferência de um ficheiro mp3 se a RSU que o estava a servir não possuisse um alcance superior a 500m. A simulação para este exemplo foi conduzida com apenas um veículo a efectuar uma transferência, tornando-se esta situação bastante problemática ao funcionamento de aplicações de não segurança em ambientes congestionados.

2.5 MAC Cooperativa

Um dos problemas da norma IEEE 802.11 é demonstrado na figura 2.12. O veículo A pode comunicar com o *Access Point* (AP) 0 directamente; no entanto o veículo B apenas pode atingir o AP 0 através do veículo A. Os standards actuais não permitem uma comunicação cooperativa nem uma retransmissão dinâmica entre nós da rede veicular. Apesar de ser possível com recurso a *bitrates* mais baixos cobrir uma maior área, a utilização de um ritmo de transmissão mais baixo resultará tipicamente num pior desempenho de toda a rede.

Através do uso da retransmissão de dados entre *On Board Units* (OBUs) será possível aumentar a área de serviço efectiva de uma *Road Side Unit* (RSU) e o ritmo de transmissão de dados, explorando a diversidade espacial da rede. As OBUs da rede assistir-se-ão mutuamente com o fim de oferecer o melhor serviço possível: uma OBU que apenas possa comunicar com a origem/destino da informação com baixos ritmos de transmissão poderá receber/transmitir

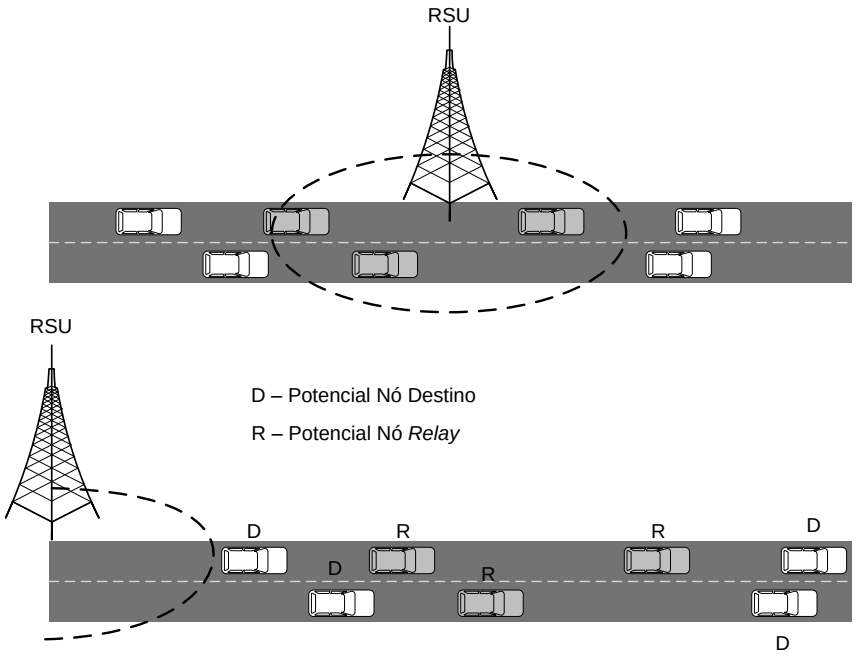


Figura 2.13: Cooperação na entrega de pacotes *broadcast*

os seus dados de/para uma outra OBU, denominada *relay*, a um ritmo mais alto, que por sua vez irá tambem comunicar com a origem/destino final a um ritmo mais elevado. Obtém-se assim um melhor desempenho da rede do que se a OBU com baixo ritmo de transmissão comunicasse directamente com a origem/destino da informação.

2.5.1 Protocolos propostos

O protocolo Vehicular Cooperative Media Access Control (VC-MAC), proposto em [16] foi concebido para cenários de *download* de dados de um AP em modo *broadcast*. Na situação ilustrada na figura 2.13 observa-se que alguns nós não conseguiram receber a informação enviada pela RSU. Os nós que receberam a informação correcta irão então anunciar esta recepção durante um intervalo denominado *Information Exchange* (IE) e são tidos como potenciais nós *relay*. Durante este intervalo, OBUs que não tenham recebido a informação da RSU apercebem-se de que existem potenciais *relays* à sua volta e comunicam a intenção de receber os dados durante a segunda parte do IE. Segue-se outro intervalo temporal denominado *Relay Set Selection* onde o conjunto de nós *relay* óptimo é seleccionado, de forma a minimizar colisões utilizando a diversidade espacial da rede. Finalmente segue-se o intervalo de *Data Forwarding* durante o qual os nós *relay* escolhidos efectuam o *broadcast* da informação recebida da RSU.

Em [7] é proposto o protocolo *Adaptive Distributed Cooperative MAC* (ADC-MAC). Este protocolo permite 3 tipos de transmissão de dados: *Direct Transmission* (DT, onde os dados são transmitidos directamente de um nó da rede para outro), *Cooperative Relay* (CR, onde os dados são transmitidos com o auxílio de um nó relay) e *Two-Hop Relay* (TR, onde não só os dados mas os pacotes RTS e CTS são transmitidos com a ajuda de um nó relay). A troca de pacotes entre estações é exemplificada na figura 2.14. No *handshake* inicial RTS-CTS

tomam partido 3 intervenientes, o transmissor, o receptor e o *relay*. O *relay* para uma dada transmissão é determinado no receptor, que mantém uma tabela, actualizada periodicamente por pacotes recebidos da rede. Uma OBU pode assim aprender dinamicamente sobre os nós vizinhos, tomando a decisão relativa a qual será o nó *relay* para uma dada transmissão.

Também em [17] foi desenvolvido um protocolo adaptativo que permite a utilização de *relays* em redes veiculares. Se a densidade de veículos em torno de um RSU for elevada, as comunicações serão executadas apenas em modo *one-hop*, ou seja sem nós *relay*. Se a densidade de veículos for baixa, ficarão activas comunicações do tipo *two-hop* garantindo à RSU uma maior área de cobertura. Estes dois modos de operação têm como finalidade não fazer com que o meio fique demasiadamante congestionado através de uma oferta permanente de nós *relay*, garantindo um bom desempenho da rede tanto em situações de congestionamento como de pouco tráfego.

2.6 Uso Eficiente do Canal

A operação multi-canal da MAC para comunicações veiculares é definida na norma IEEE 1609.4. Os intervalos de acesso ao canal são divididos em intervalos de duração fixa de 100ms, sendo que metade deste intervalo é atribuído ao canal de controlo e a outra metade a canais de serviço. O CCH servirá principalmente para a troca de informações de segurança e mensagens de controlo entre nós da rede veicular, enquanto que os canais de serviço serão de uso mais geral por parte de outras aplicações. Os vários nós da rede terão de competir pelo acesso tanto do CCH como do SCH, podendo isto ser um problema devido à grande dinâmica da topologia de redes deste tipo. Um dos desafios do uso eficiente do canal de transmissão é o correcto funcionamento de aplicações de *infotainment* que utilizem grande largura de banda ao mesmo tempo que se garante a comunicação criticamente temporal entre veículos de mensagens de segurança. Para solucionar este problema foram propostas várias abordagens:

- Duração variável para intervalos de controlo e de serviço;
- Coordenação dos intervalos de controlo e de serviço por parte de uma RSU;
- Utilização da diversidade espacial da rede para reutilização de canais.

2.6.1 Intervalos de Duração Variável

Em [8] foi proposto o esquema *Variable* CCH *Interval* MAC (VCI-MAC). Um dos problemas identificados com a duração fixa do intervalo de controlo foi a sua baixa eficiência devido às grandes variações na topologia de uma rede veicular: numa situação de grande congestionamento, o CCH poderá não conseguir oferecer largura de banda suficiente para a troca de todas as mensagens necessárias; por outro lado, quanto a densidade de nós é baixa o intervalo relativo ao CCH poderá estar a desperdiçar alguma da largura de banda necessária a aplicações mais pesadas que estejam a utilizar um canal de serviço. Foi então proposto que o intervalo relativo ao canal de controlo fosse ainda dividido num *Safety Interval* e *WAVE Service Announcement* (WSA) *Interval*, como esquematizado na figura 2.15. Durante o primeiro intervalo serão trocadas as mensagens de segurança, enquanto que o segundo intervalo será utilizado por nós fornecedores de serviços para os anunciar através de pacotes WSA. Uma RSU enviará no início de cada intervalo de controlo um pacote VCI que indicará a duração dos intervalos de controlo e de serviço de acordo com a quantidade de tráfego que a rodeia. No

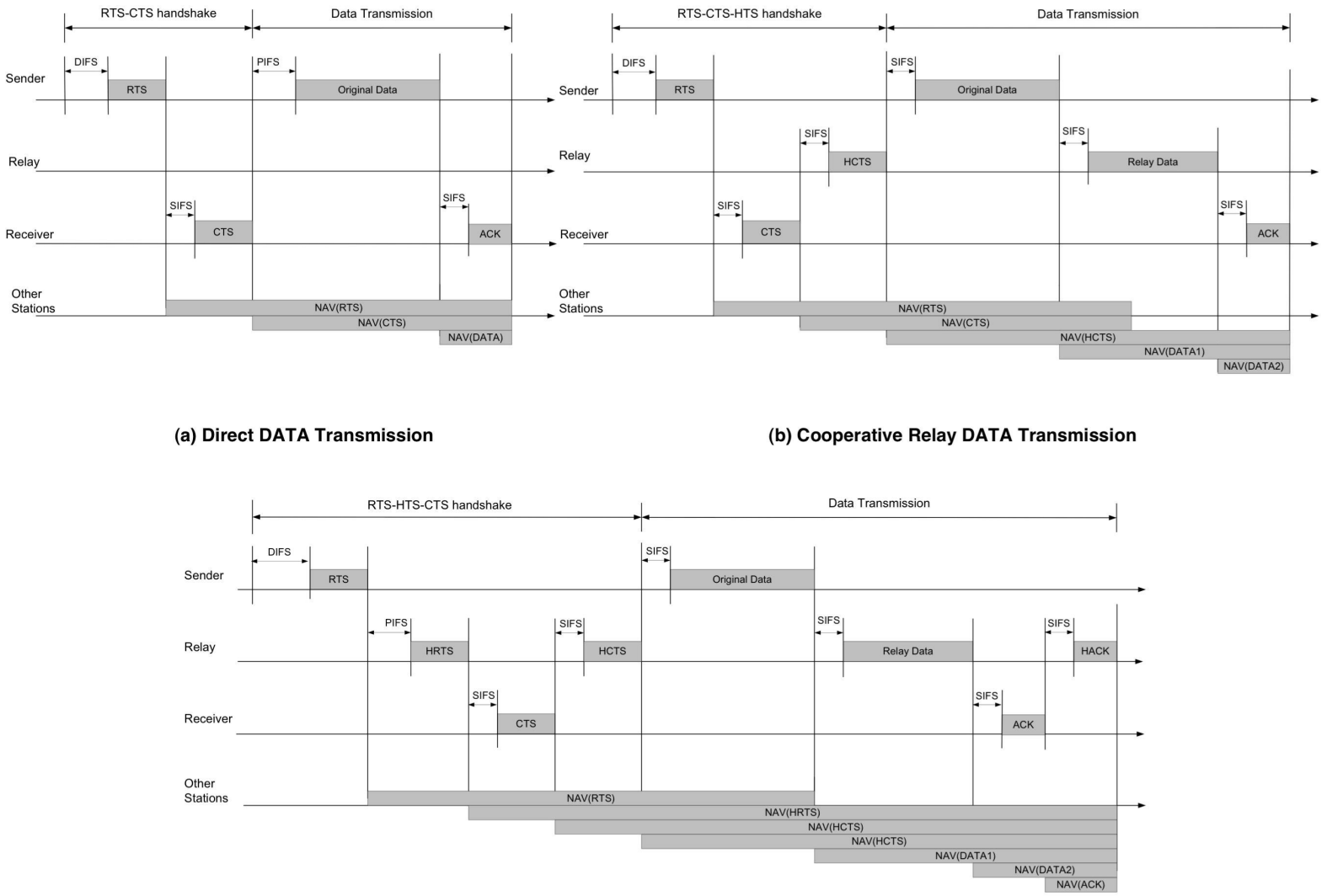


Figura 2.14: Troca de pacotes proposta em [7].

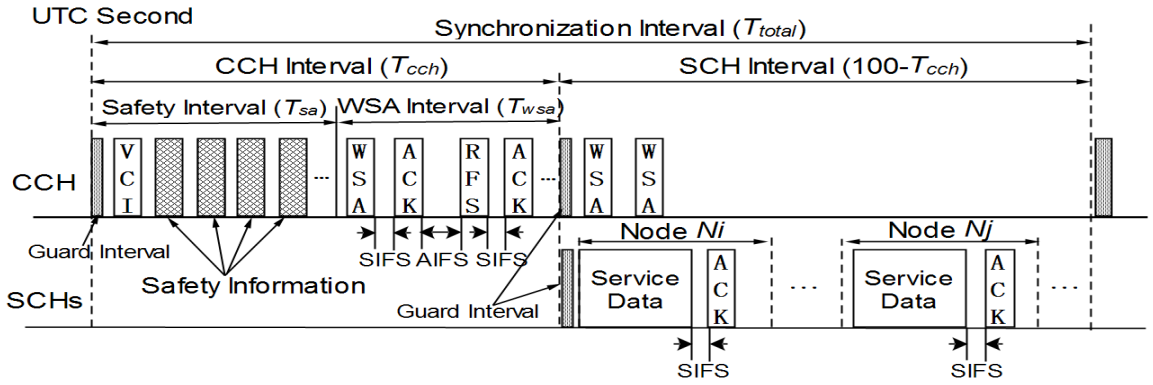


Figura 2.15: Divisão do intervalo de controlo e trocas de pacotes propostas em [8].

entanto esta abordagem é apenas possível quando coordenada através de uma RSU, estando em progresso o desenvolvimento deste esquema para um sistema distribuído.

Também em [15] é proposto um ajuste da duração dos intervalos de controlo e de serviço de acordo com a densidade de nós verificada a cada momento, como forma de assegurar a correcta comunicação de mensagens de segurança entre veículos.

2.6.2 Coordenação por RSU

Outro esquema para servir tanto mensagens de segurança como aplicações que utilizem maior largura de banda foi proposta em [18]. Aqui são utilizados dois diferentes métodos de transmissão de mensagens: se o veículo estiver na proximidade de uma RSU, denominada por *Coordinating Access Point* (CAP), será a RSU a controlar toda a rede, ditando os intervalos durante os quais podem ser trocadas mensagens de segurança e intervalos durante os quais serviços podem ser oferecidos; na ausência de uma RSU, os veículos criarão uma rede ad-hoc onde apenas serão trocadas mensagens de segurança. Um CAP conterá uma lista dos veículos na sua proximidade e coordenará a transmissão de mensagens de segurança através de *polling* num intervalo denominado *Collision Free Period* (CFP). Para que a latência máxima seja 100ms, cada veículo terá a possibilidade de transmitir as suas mensagens relativas a segurança a pelo menos cada 100ms. Assim, tal como na proposta anterior, a RSU terá a maior responsabilidade na sincronização da rede para permitir a operação em vários canais. A prestação de serviços que não de segurança apenas decorrerá nas imediações de um CAP.

2.6.3 Diversidade Espacial

Um protocolo baseado na criação de *clusters* de veículos coordenado por uma RSU, *Cluster* MAC (CMAC), é descrito em [19] e demonstrado na figura 2.16. A RSU aloca canais a veículos baseando-se nos seus *clusters*, permitindo a reutilização de canais em *clusters* não adjacentes. A coordenação de todas as comunicações numa dada área por parte de uma RSU permite uma comunicação sem contenção do meio. Propõe-se a utilização de duas antenas direccionais na RSU como forma de manter comunicações simultâneas com áreas opostas da estrada sem interferência entre elas. É mantida uma fila na RSU para pedidos de envio de mensagens de segurança e uma outra fila para pedidos diversos, dando-se à primeira uma maior prioridade.

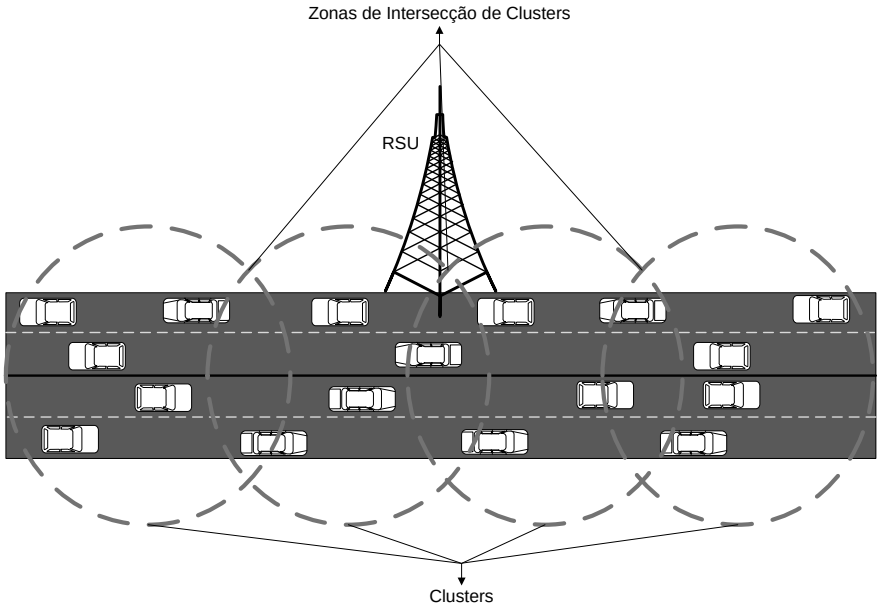


Figura 2.16: Clusters numa Rede Veicular.

No entanto esta abordagem, tal como outras, não será possível num modo *ad hoc* sem a existência de RSUs.

2.7 Qualidade de Serviço

Outro aspecto importante que a camada MAC tem de ter em conta é a qualidade de serviço das várias aplicações. Um dos problemas identificados relativamente ao IEEE 802.11p é o facto de o seu mecanismo de contenção de canal dificultar imenso a entrega de dados a um ritmo elevado, principalmente em ambientes muito congestionados. Em redes veiculares altamente dinâmicas torna-se importante obter o maior *throughput* possível para permitir o funcionamento correcto de aplicações de entertenimento, que se espera aumentarão o conforto para passageiros de veículos.

Para além do correcto funcionamento de serviços de lazer, é também necessário garantir que as mensagens de segurança são entregues a todos os veículos nelas interessados dentro de um intervalo de tempo curto e com uma baixa percentagem de perda de pacotes. A correcta disseminação de mensagens de segurança é um dos grandes desafios das *Vehicular Ad-hoc Network*s (VANETs), pois a rede é completamente distribuída e informações sobre eventos súbitos que ocorram na estrada podem ser a diferença entre a vida e a morte. As mensagens de segurança em redes veiculares são assim semelhantes a aplicações de tempo real, onde as mensagens têm de ser entregues com um atraso inferior a uma *deadline*, após a qual a mensagem perde a sua utilidade.

O aumento do conforto para condutores e passageiros de veículos poderá ser um factor motivador para os consumidores, alastrando as redes veiculares a um maior número de pessoas e tornando o desenvolvimento de sistemas deste tipo atractivo de um ponto de vista comercial. É então importante que a MAC ofereça também uma boa base ao funcionamento deste tipo

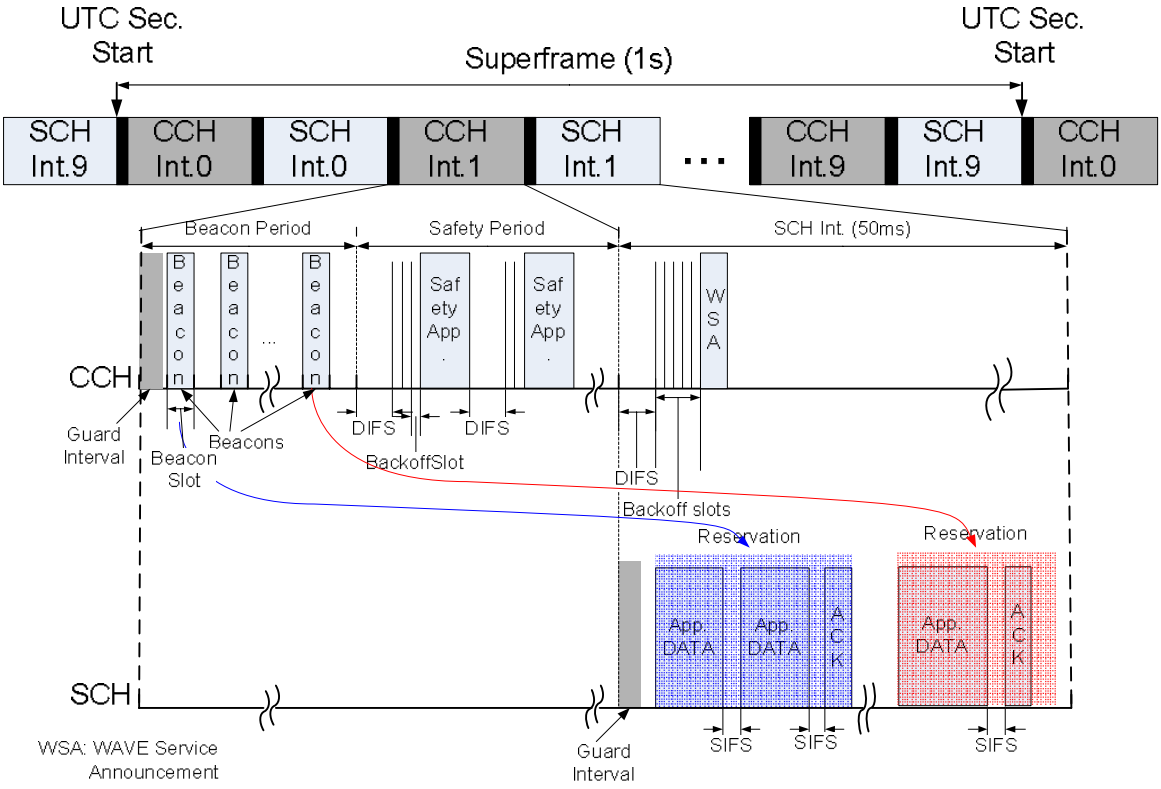


Figura 2.17: Divisão do intervalo de controlo e trocas de pacotes propostas em [9].

de aplicações.

Tem-se então dois tipos de especificação principais em comunicações veiculares

- Largura de banda para aplicações de *infotainment*;
- Baixa latência para entrega rápida de mensanges de segurança.

2.7.1 Qualidade de Serviço para *Infotainment*

No protocolo *Vehicular* MESH MAC (VMESH-MAC) [9] o problema do ritmo de tráfego para serviços de entertenimento é abordado. Neste protocolo, cada segundo UTC (denominado no protocolo por *super-frame*) é dividido em 10 intervalos e cada intervalo de canal de controlo é dividido entre um período de *beacon* (BP) e um período de segurança (SP), reservado exclusivamente para troca de mensagens de segurança. Durante o período de *beacon*, os veículos anunciam a sua presença na rede e tentam reservar no nó fornecedor de serviço um determinado canal de serviço e uma determinada posição durante um *super-frame* durante o qual o veículo poderá aceder ao meio, livre de mecanismos de contenção. Este processo está representado na figura 2.17. Através deste protocolo é possível efectuar reserva de serviços em que o fornecedor pode ser tanto uma RSU como uma OBU. No entanto, ao dividir o intervalo do canal de controlo em 2, reduzirá as oportunidades de transmissão de mensagens de segurança, podendo estas ser atrasadas.

O protocolo WAVE-*Hybrid Coordination Function* (W-HCF) foi proposto em [10]. O mecanismo proposto é representado na figura 2.18. Durante um intervalo de controlo uma

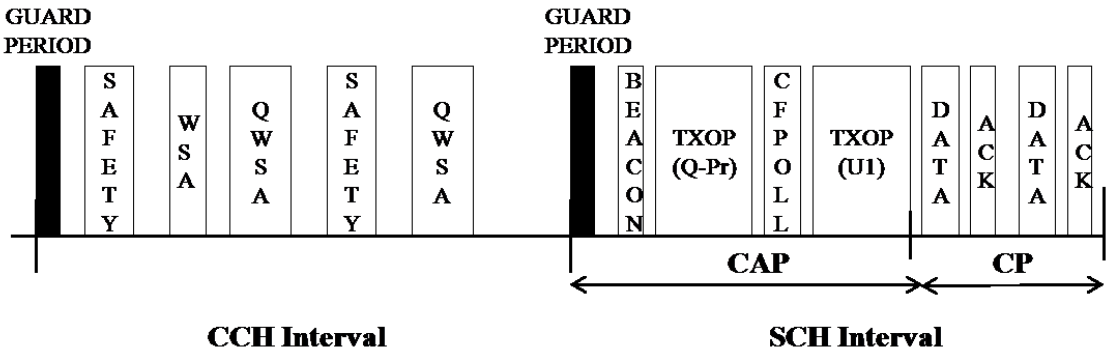


Figura 2.18: Troca de pacotes e coordenação por RSU propostas em [10].

RSU com capacidade de oferecer qualidade de serviço, denominada *Q-Provider* (Q-Pr), envia tramas QWSA para anunciar esta funcionalidade a OBUs na sua proximidade. As OBUs que quiserem requisitar este serviço poderão reservar um período durante o intervalo de serviço durante o qual conseguirão efectuar transmissão de dados sem qualquer contenção do meio. Isto é conseguido com a reserva por parte do Q-Pr de um intervalo de tempo dentro de um SCH, denominado por *Controlled Access Period* (CAP), durante o qual a RSU irá efectuar o *polling* das OBUs das quais recebeu uma reserva. As RSUs que não efectuaram nenhuma reserva e não se encontram a executar nenhuma aplicação que requere qualidade de serviço, terão de esperar pelo fim do CAP para transmistir ou receber dados. O Q-Pr estima o tempo que uma dada RSU estará dentro do seu alcance, através de pacotes trocados entre os nós que indicam a posição, velocidade e direcção de um veículo, e continuará a executar *polling* de um veículo que tenha efectuado reserva até que este tempo se esgote. Caso o veículo se encontre ainda nas imediações do Q-Pr poderá efectuar uma nova reserva para continuar a desfrutar da qualidade de serviço oferecida pela RSU. Uma desvantagem desta abordagem é a de não permitir qualidade de serviço nas transmissões V2V, necessitando da coordenação de uma RSU.

2.7.2 Qualidade de Serviço para Segurança

Um protocolo distribuído para reserva de posições durante o intervalo do canal de controlo, *Dedicated Multi-Channel* MAC (DMMAC) é apresentado em [11]. Aqui cada veículo terá a possibilidade de enviar mensagens de segurança sem contenção do meio através da reserva de uma posição. O canal de controlo é dividido em duas fases: uma em que o acesso não tem contenção do meio e é reservado por cada veículo denominado por *Adaptive Broadcast Frame* (ABF); a restante duração do canal de controlo é denominada *Contention-based Reservation Period* (CRP) e é utilizada para efectuar reservas durante o ABF e anúncios de outros serviços que estejam a ser prestados. A duração dos intervalos de controlo e de serviço são fixas, mas a duração do ABF e CRP é variável de acordo com as necessidades. Esta divisão do canal de controlo encontra-se representada na figura 2.19.

[20] propõe-se a utilizar o standard MAC RR-ALOHA, uma alternativa à MAC IEEE 802.11p, como forma de evitar contenção no meio. Neste protocolo, slots temporais são alocados dinamicamente sem necessitar de um nó coordenador, sendo por isso um protocolo distribuído e adequado para redes *ad-hoc*. Neste protocolo os nós acrescentam às suas tramas MAC um campo *Frame Information* (FI) que contém a sua percepção da rede, isto é, que

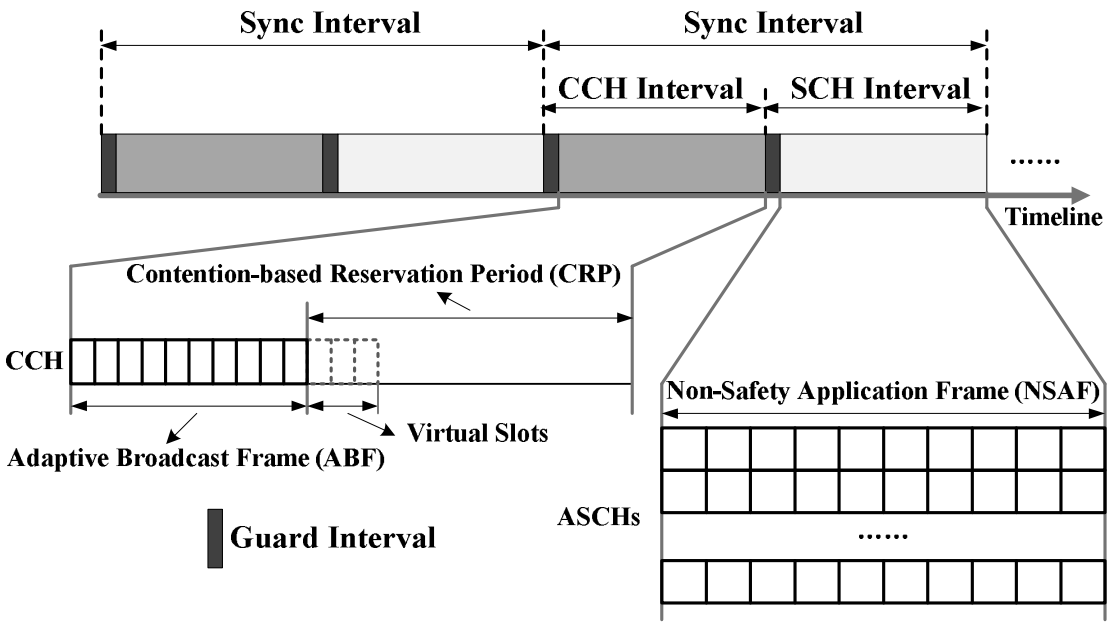


Figura 2.19: Divisão do canal de controlo proposta em [11].

slots temporais estão ocupados por que nó. No entanto, devido às especificidades das redes veiculares (alta mobilidade, baixos tempos de conectividade), foi necessário identificar alguns dos problemas deste protocolo e sugerir soluções, criando o RR-ALOHA+. As alterações incluem a difusão de FIs por 2 saltos, permitindo uma maior vista da rede que circunda um veículo, melhores mecanismos de detecção de possíveis colisões, reutilização de slots temporais e negociação de identificadores entre nós da rede.

Capítulo 3

Especificação da MAC

3.1 Introdução

Pretende-se no contexto deste projecto o desenvolvimento de um protótipo para comunicações veiculares de acordo com os standards WAVE e IEEE 802.11p, sendo implementadas as várias camadas da pilha protocolar. Escolheu-se implementar a camada MAC e a parte digital da PHY num único *System on Chip* (SoC). O desenvolvimento de uma camada MAC própria do projecto facilita a integração desta dentro da plataforma concebida, para além de permitir a realização de testes de desempenho focados nas diferentes funcionalidades da camada de acesso oferecendo vários níveis de flexibilidade. Diferentes abordagens podem ser tomadas quanto ao desenvolvimento da MAC, nomeadamente ao nível das funcionalidades que se desejam ser implementadas em *software* correndo sobre um microprocessador e aquelas que são implementadas recorrendo a lógica *custom* em *hardware*, interface entre as restantes camadas protocolares e acesso a módulos adicionais presentes no protótipo, como por exemplo módulo GPS como forma de realizar a sincronização da estação. A camada MAC desenvolvida potenciará ainda a investigação na área de camadas de acesso para comunicações veiculares, por exemplo, ao nível de propostas de alteração e melhoramento dos standards actuais.

Escolheu-se utilizar uma FPGA para o desenvolvimento e implementação da nossa camada MAC pela flexibilidade que esta oferece relativamente à divisão das funcionalidades que se desejam ser implementadas em *software* correndo sobre um microprocessador e aquelas que se pretende sejam implementadas por *hardware* dedicado. A integração e interface com a camada PHY será também mais flexível, uma vez que as camadas MAC e PHY serão implementadas no mesmo *chip*. O crescente aumento de capacidade das FPGAs possibilita a implementação de lógica bastante complexa com um bom desempenho devido ao paralelismo de operações que se consegue alcançar, para além das vantagens oferecidas ao nível de testes: alterações ao projecto são efectuadas, implementadas em *hardware* e testadas num muito curto espaço de tempo. Para além do *hardware* em si, as ferramentas de desenvolvimento de SoCs existentes facilitam enormemente a concepção deste tipo de sistemas, particularmente a integração do microprocessador com o *hardware* dedicado desenvolvido.

Na figura 3.1 encontra-se representada a arquitectura geral do sistema, onde se apresenta a MAC e parte da PHY completamente implementadas em FPGA e a subcamada PHY *Radio Frequency* (RF) e camada *Logical Link Control* (LLC) externas ao *chip* da FPGA.

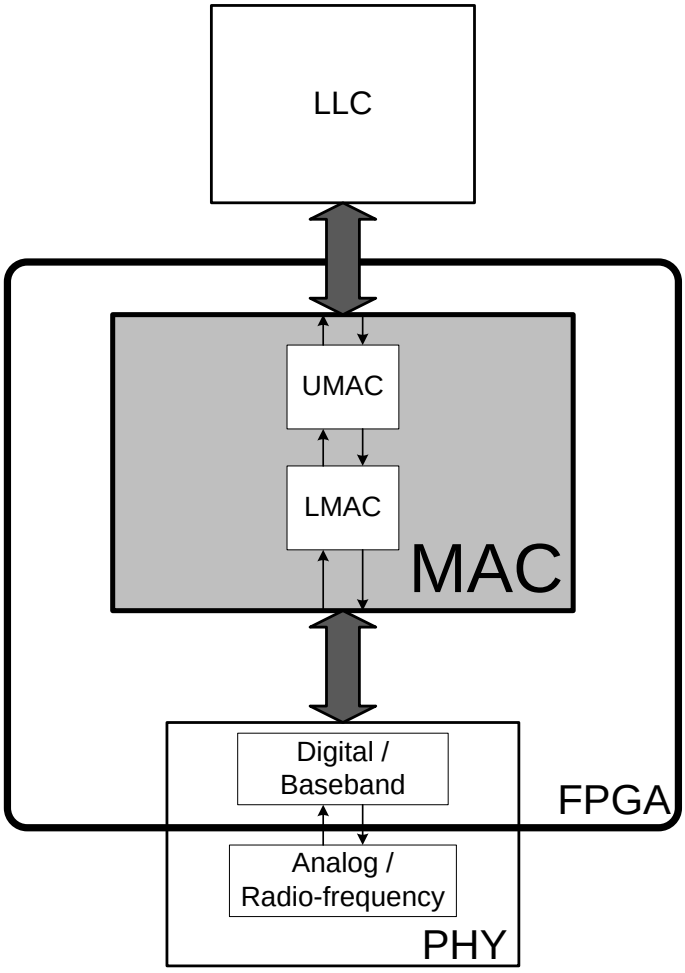


Figura 3.1: Arquitectura geral do sistema.

3.2 Funcionalidades da camada MAC

A camada MAC implementada deverá realizar as seguintes funções:

- Gestão da memória para recepção/transmissão;
- Comunicação com a camada PHY;
- Recepção de tramas da PHY;
- Transmissão de tramas para a PHY;
- Manutenção dos *timers* do sistema, tanto relativos à contenção do meio como do relógio do sistema;
- Geração e verificação do campo *Frame Check Sequence* (FCS) para validação de tramas;
- Construção de tramas MAC;
- Comunicação com camadas superiores;
- Extracção de cabeçalho para tramas recebidas;
- Construção de cabeçalho para tramas a enviar;
- Processamento de tramas recebidas;
- Controlo de acesso ao meio, tanto virtual como físico;
- Geração de respostas a certas tramas (envio de tramas ACK e RTS/CTS).

3.3 Decomposição entre UMAC e LMAC

Para a concepção da nossa camada de acesso, decidiu-se dividir a MAC em duas partes distintas: UMAC e LMAC. Como o nome indica a UMAC tratará da comunicação entre a MAC e as camadas superiores e de operações de mais alto nível, estando definida em *software*. A LMAC, por outro lado, será completamente definida em *hardware* e estará encarregue de funções de mais baixo nível e da comunicação com a camada PHY.

Esta divisão partiu da necessidade de algumas das funções da camada de acesso necessitarem de uma maior predictabilidade e precisão em termos temporais, sendo este determinismo impossível de assegurar a um nível mais alto. Assim foram identificadas as funcionalidades que necessitariam de ser implementadas em *hardware* e aquelas que, não requerendo uma precisão temporal tão alta, poderiam ser implementadas em *software*. Esta divisão não é final, e à medida que mais testes vão sendo efectuados e mais mecanismos implementados, poderá revelar-se necessário transferir algumas das funcionalidades da UMAC para LMAC ou vice-versa.

Outra vantagem deste tipo de arquitectura é a possibilidade de a LMAC estar a receber uma trama da PHY ao mesmo tempo que a UMAC processa outros dados, permitindo assim um paralelismo de operações.

A arquitectura geral da camada MAC com maior enfâse na LMAC encontra-se esquematizada na figura 3.2. Tramas vindas da camada PHY serão colocadas numa memória interna

de posições de tamanho fixo, sendo validado o seu campo FCS à medida que os dados vão chegando à MAC. Quando a trama tiver sido transmitida na totalidade à camada, MAC irá ser gerada uma interrupção notificando a UMAC da existência de novos dados na memória. Caso a UMAC deseja transmitir uma trama, vinda de camadas superiores ou gerada internamente, escrevê-la-á também na memória interna da LMAC, juntamente com o cabeçalho MAC gerado. A UMAC poderá então escrever num registo da LMAC, dando início ao processo de envio dos dados em memória para a camada PHY. À medida que os dados vão sendo enviados é gerado o campo FCS para a trama e é enviado para a PHY assim que os dados em memória tiverem sido transmitidos. O modelo de programação da LMAC é especificado na secção 3.4.

Caberá assim à LMAC:

- Realizar o interface entre a camada MAC e a camada PHY;
- Gestão da memória de tramas;
- Gestão dos *timers* do sistema;
- Geração do *Cyclic Redundancy Check* (CRC)32 para tramas enviadas e verificação de erros para tramas recebidas;

Enquanto que a UMAC ficará encarregue de:

- Comunicação com as camadas superiores;
- Processamento de tramas recebidas;
- Contrução do cabeçalho MAC;
- Controlo de acesso ao meio.

3.4 Modelo de Programação da LMAC

Na figura 3.3 estão representados os registos presentes na LMAC, através dos quais a LMAC pode controlar a sua actividade ou ler o seu estado. A função de cada um deles, bem como os procedimentos a efectuar pela UMAC para os utilizar correctamente estão descritos no apêndice A.

3.4.1 Registos de Estado

Estes registos permitem à UMAC verificar o estado de várias funções da LMAC entre elas:

- **Estado da Memória (MEMSTA)** - Neste registo é possível à UMAC determinar o número de posições livres em memória (FREES), o número total de posições (NSTLS), o sucesso ou insucesso da alocação de uma posição (VLDSL) e o ID da posição alocada (ALCID). Cada posição de memória pode conter uma trama MAC;
- **Estado de Transmissão (STATUS)** - Verificação da disponibilidade da LMAC em transmitir uma trama num dado momento (TXRDY);

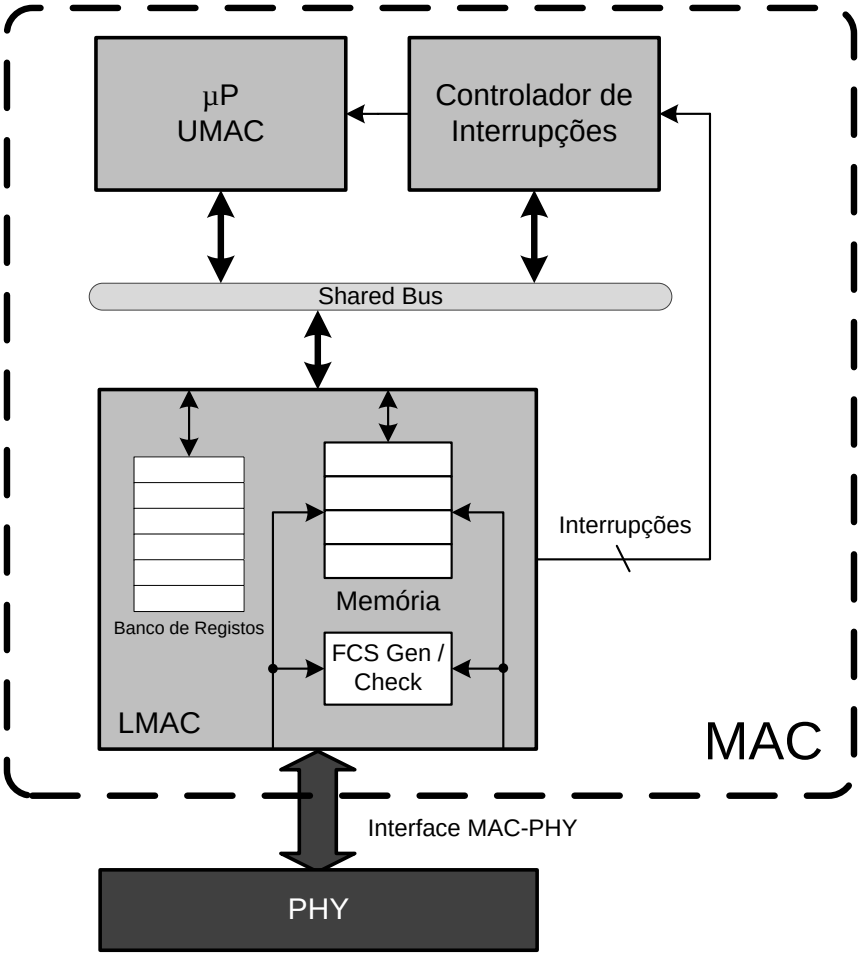


Figura 3.2: Arquitectura geral da camada MAC.

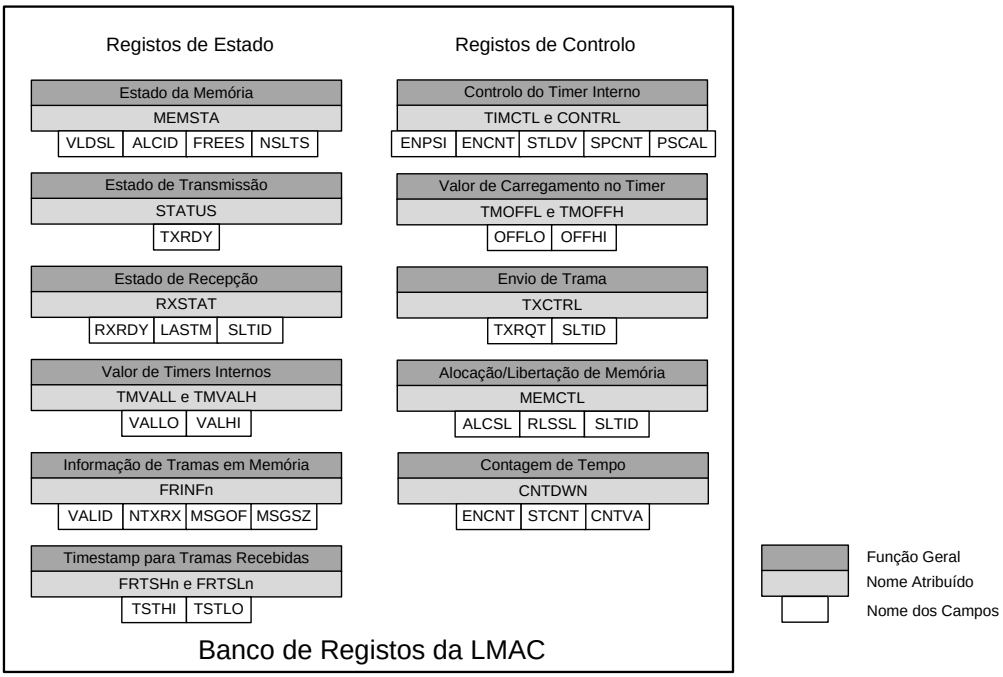


Figura 3.3: Modelo de programação simplificado da LMAC.

- **Estado de Recepção (RXSTAT)** - Neste registo a UMAC pode verificar se existem tramas recentemente recebidas à espera de ser atendidas (RXRDY), se se encontram outras ainda em fila de espera para leitura pela UMAC (LASTM), e em que posição de memória se encontram (SLTID). Este registo é actualizado de cada vez que é lido, i.e. quando a UMAC lê este registo é colocado no campo SLTID uma nova posição de memória com uma trama à espera de ser atendida (se existir) e actualizados os campos RXRDY e LASTM;

- **Valor de *Timers* Internos (TMVALL e TMVALH)** - Dois registos contendo os 32 bits menos (VALLO) e mais significativos (VALHI) do *timer* do sistema;

- **Informação de Tramas em Memória (FRINFn)** - Um registo por cada posição de memória, onde se encontra informação relativa à existência de dados válidos na posição de memória correspondente (VALID), se a trama nessa posição foi recebida da PHY ou colocada pela UMAC (NTXRX), o *offset* dos dados em relação ao endereço inicial da posição de memória (MSGOF) e o tamanho da trama (MSGSZ);

- ***Timestamp* para Tramas Recebidas (FRTSHn e FRTSLn)** - Dois registos contendo os 32 bits menos (TSTLO) e mais significativos (TSTHI) do *timestamp* de recepção para cada posição de memmória.

3.4.2 Registos de Controlo

Os registos de controlo possibilitam à UMAC actuar na LMAC, efectuando diversos pedidos e operações.

- **Controlo do *Timer* Interno (TIMCTL e CONTRL)** - Estes registos permitem controlar a operação do *timer* interno da LMAC, podendo a LMAC activar a geração de interrupções a cada segundo (ENPSI), activar o *timer* (ENCNT), carregar um valor no *timer* (OFFLO, OFFHI) mediante a activação de uma flag (STLDV) e configurar o valor de *prescale* deste (PSCAL), através da activação de uma outra flag (SPCNT);

- **Valor de Carregamento no *Timer* (TMOFFH e TMOFFL)** - Registos que podem ser escritos pela UMAC com os 32 bits menos significativos (OFFLO) e mais significativos (OFFHI) do valor que se deseja carregar no *timer*;

- **Envio de Trama (TXCTRL)** - Através deste registo a UMAC poderá ordenar o envio dos dados numa dada posição de memória (SLTID) mediante a activação de uma *flag* (TXRQT);

- **Alocação/Libertação de Memória (MEMCTL)** - Neste registo a UMAC pode requerer a alocação de uma posição de memória ao activar uma *flag* (ALCSL), ou libertar uma dada posição de memória (SLTID) através da activação de uma outra flag (RLSSL).

- **Contagem de Tempo (CNTDWN)** - A UMAC pode configurar este registo com um valor em microssegundos (CNTVA) e realizar a activação de um contador (ENCNT) que activará uma interrupção CNTVA microssegundos depois. É também possível parar o contador através da activação de uma outra flag (STCNT) e realizar a leitura do seu valor actual (CNTVA).

3.4.3 Interacção LMAC-UMAC

Transmissão de dados à PHY

Para transmitir dados à camada PHY a UMAC deverá, de acordo com o diagrama de fluxo da figura 3.4:

1. Pedir à LMAC a alocação de uma posição de memória;

2. Escrever os dados na posição alocada;

3. Configurar o tamanho da trama;

4. Ler o estado da transmissão: caso a LMAC não esteja pronta a transmitir, a UMAC poderá escolher entre verificar o estado novamente ou desisitir da transmissão;

5. Caso a LMAC esteja pronta a transmitir, a UMAC poderá dar a ordem de transmissão da posição de memória;

6. No final da transmissão, a UMAC poderá libertar a posição de memória ou poderá escolher libertá-la mais tarde.

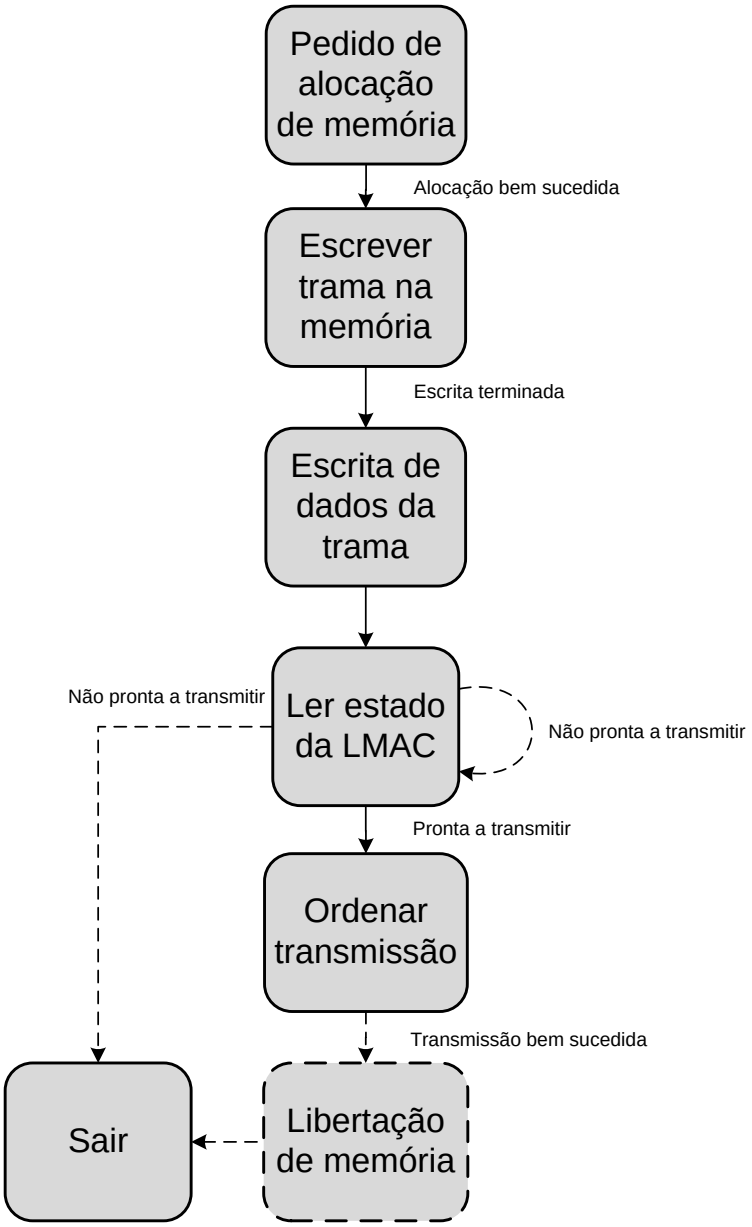


Figura 3.4: Diagrama de fluxo para a interacção entre UMAC e LMAC na transmissão de uma trama.

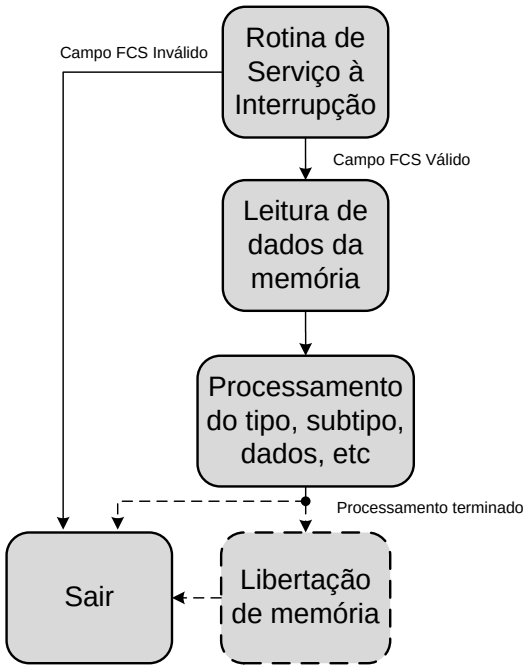


Figura 3.5: Diagrama de fluxo para a interacção entre UMAC e LMAC na recepção de uma trama.

Recepção de dados da PHY

De forma a ler correctamente dados vindos da camada PHY a UMAC seguirá os seguintes passsos, descritos no diagrama de fluxo da figura 3.5:

1. No final da recepção de uma trama, a LMAC gerará uma interrupção notificando a UMAC da recepção de uma nova trama;

2. Caso o campo FCS seja válido a UMAC poderá ler os dados da memória; no caso de erros na trama a UMAC ignorará os dados recebidos;

3. Depois de lidos os dados, estes serão processados pela UMAC;

4. No final do processamento, a UMAC poderá decidir entre libertar os dados em memória ou realizar esta libertação numa outra altura.

3.5 Comunicação com camada PHY

A LMAC tem como função efectuar a passagem de mensagens entre a camada MAC e a PHY utilizando as primitivas e fluxos descritos na norma 1609.4 [4]. As primitivas de comunicação são implementadas com recurso a sinais lógicos trocados entre a LMAC e a PHY. Na figura 3.6 estão representados os sinais trocados entre as duas camadas.

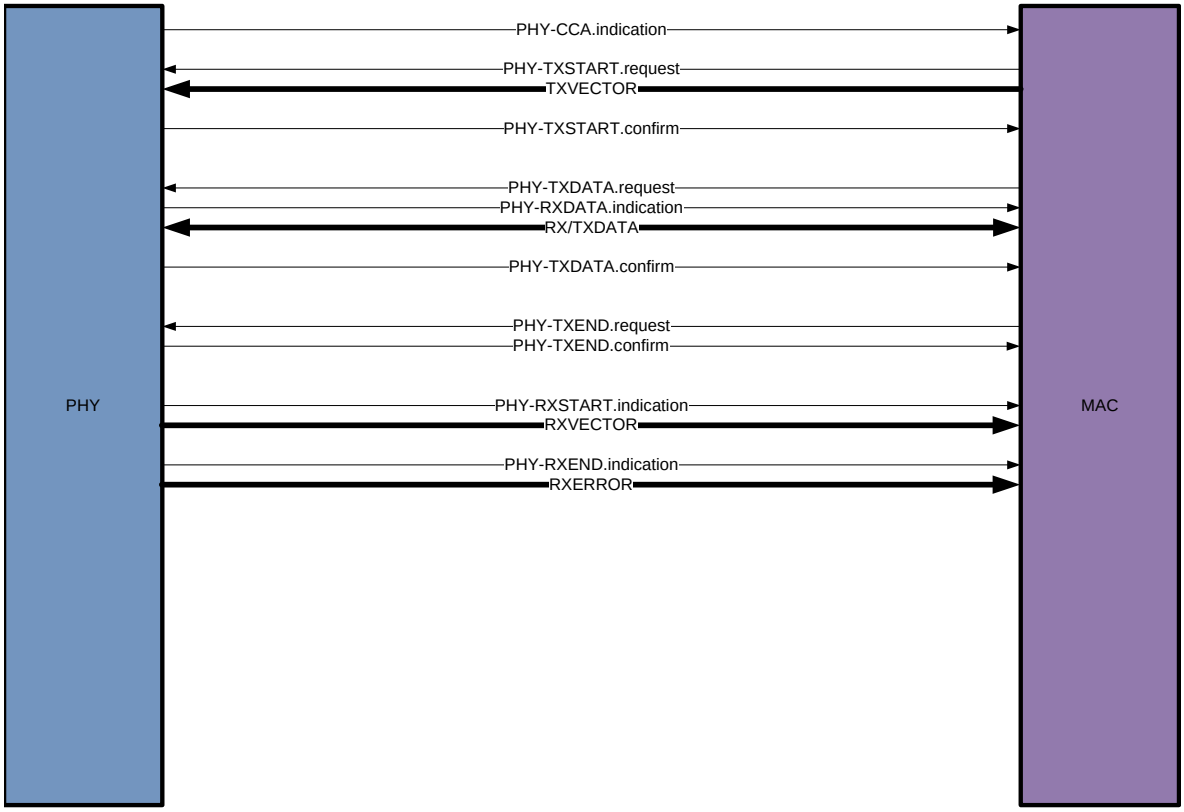


Figura 3.6: Interface entre as camadas MAC e PHY.

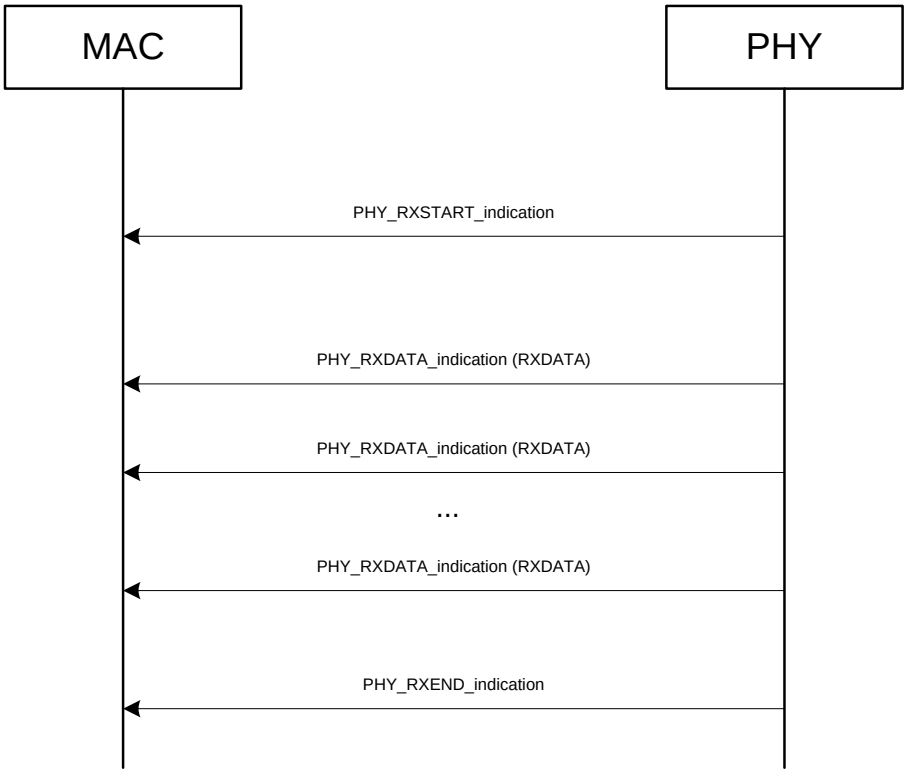


Figura 3.7: Sequência de primitivas do processo de recepção.

3.5.1 Recepção de Tramas da PHY

A LMAC encontra-se também encarregue da recepção de tramas enviadas pela camada PHY e colocação destas numa posição de memória livre. A troca de primitivas segue de acordo com a norma IEEE 802.11 (secção 12.3.5) e está representada na figura 3.7. Compete também à LMAC guardar o *timestamp* de recepção de tramas vindas da camada PHY.

Um possível diagrama temporal para a recepção de uma trama está representado na figura 3.8.

3.5.2 Transmissão de tramas para a PHY

A LMAC tem também como função o envio de tramas da sua memória para a camada PHY. A troca de primitivas segue de acordo com a norma IEEE 802.11 (secção 12.3.5) e está representada na figura 3.9.

O diagrama temporal de uma possível transmissão de dados da MAC para a PHY está representada na figura 3.10.

3.6 Memória

A memória da camada de acesso possui várias posições de tamanho estático, sendo armazenada em cada uma delas no máximo uma trama. A memória interna utilizada tem

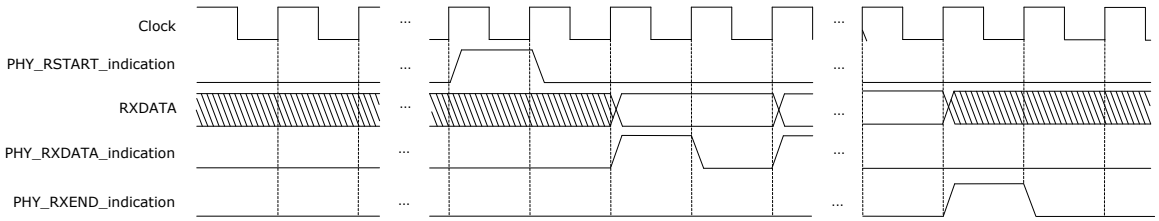


Figura 3.8: Diagrama temporal da transmissão de uma trama à MAC pela PHY.

capacidade para albergar até 6 tramas.

Esta memória tem como finalidade guardar tramas recebidas vindas da camada PHY, para posterior leitura e processamento por parte da UMAC, e guardar tramas geradas pela UMAC para transmissão para a camada PHY e consequente difusão no meio. Possui assim dois portos de leitura e de escrita, um para comunicar com o Microprocessador e consequentemente com a UMAC e outro para leitura e escrita de dados por parte da LMAC.

O gestor de memória em hardware realiza as seguintes funções:

- Gestão das posições livres e ocupadas;
- Alocação de posições de memória para tanto na recepção de tramas da PHY e como em pedidos de alocação por parte da UMAC.

3.7 *Timer*

A LMAC possui um timer de 64 bits, que corresponde ao *offset* em microssegundos relativamente a 1 de Janeiro de 1958, seguindo a recomendação TF.460-4(2002) do ITU-R como descrito na norma 802.11p [12]. Este timer é mantido por um contador incremental cujos parâmetros podem ser configurados pela UMAC:

- Valor do factor de pré-divisão;
- Activação de interrupções periódicas;
- Carregamento de um valor no *timer*.

O valor actual do *timer* é armazenado em dois registos de 32 bits acessíveis pela UMAC. Uma boa gestão da precisão do *timer* é importante pois a camada PHY tem de comutar de canais periodicamente e erros no valor do *timer* farão com que as mudanças de canal não estejam sincronizadas e o dispositivo não escutará o canal de controlo nos intervalos correctos.

3.8 Geração e Verificação de Frame Check Sequence

A geração e verificação do campo MAC FCS para uma trama inteira seria um processo extremamente pesado computacionalmente e é então implementada na LMAC por *hardware* dedicado. O campo FCS tem 32 bits de comprimento e é adicionado no final da trama MAC. O polinómio de 32 bits para geração e verificação da sequência está descrito na norma IEEE 802.11 (secção 7.1.3.7) [13]:

$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$

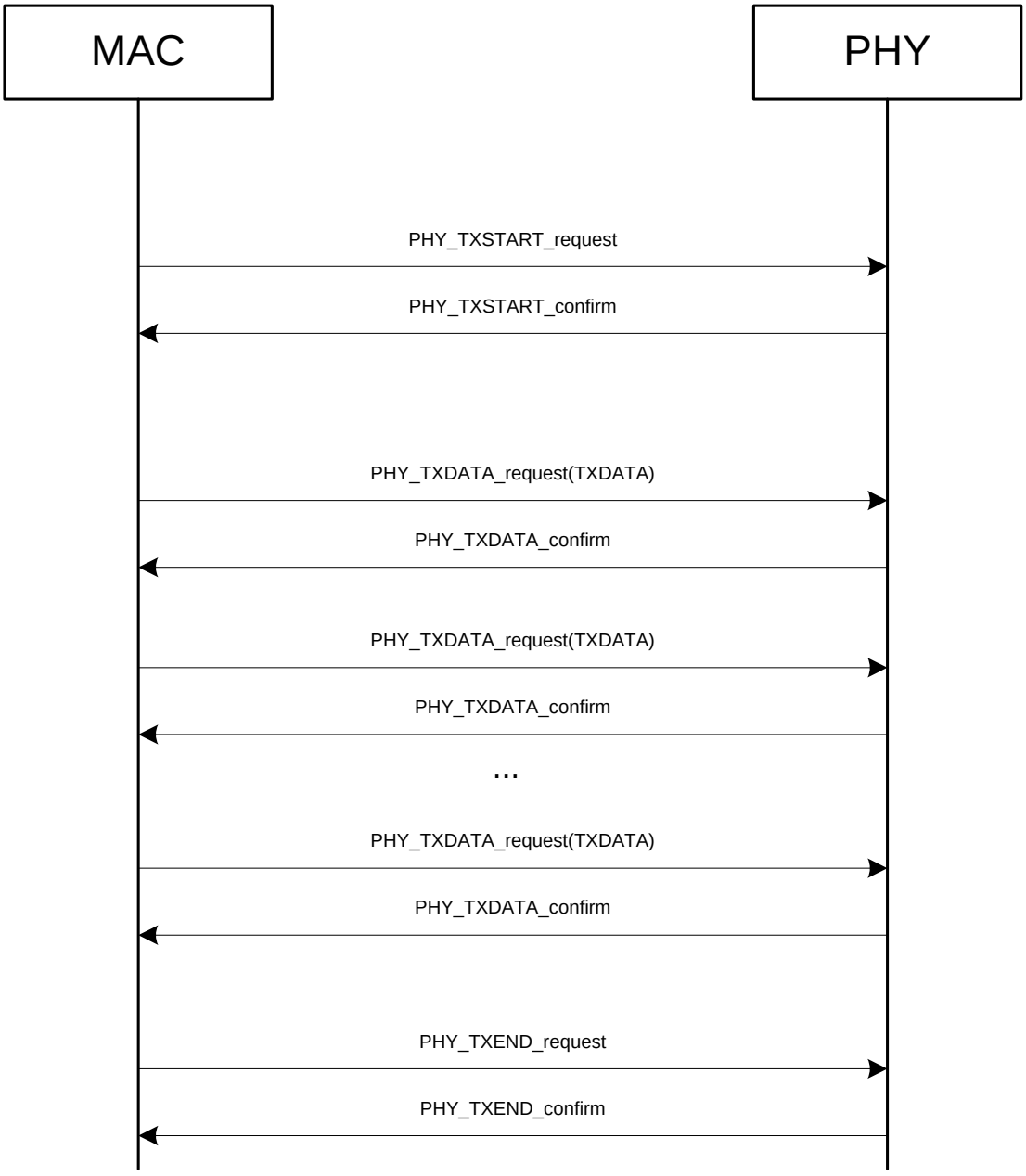


Figura 3.9: Sequência de primitivas do processo de transmissão.

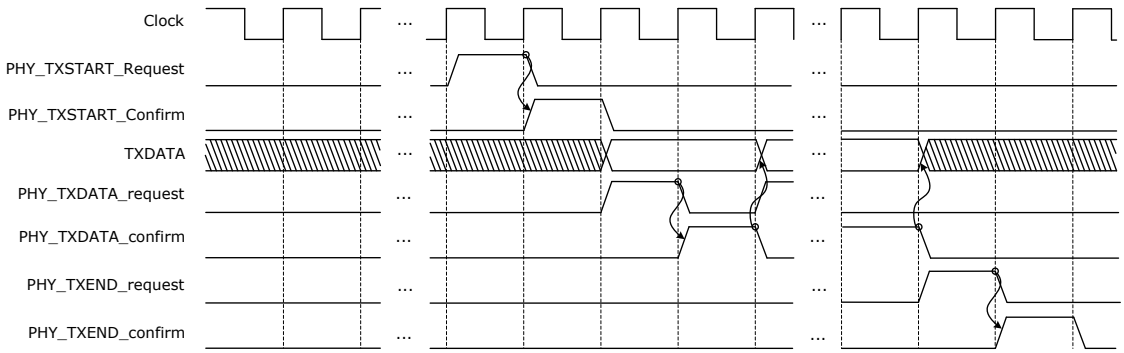


Figura 3.10: Diagrama temporal da transmissão de uma trama da MAC para a PHY.

Capítulo 4

Implementação

De forma a integrar as várias camadas da pilha protocolar WAVE foi desenvolvida uma plataforma com esse propósito, que oferece grande liberdade ao nível das suas configurações e acesso aos módulos que a constitui. A arquitectura da plataforma desenvolvida encontra-se representada na figura 4.1.

Esta plataforma contém:

- Antena com uma largura de banda entre 5.75GHz e 6.05GHz;
- Módulo RF capaz de transmitir e receber sinais e que realiza a conversão de banda base para radio-frequência e vice-versa;
- Conversores analógico-digital e digital-analógico, implementando o interface entre a PHY digital e a PHY analógica;
- Módulo FPGA que suportará a camada MAC e a parte digital da camada PHY;
- Módulo *General Packet Radio Service* (GPRS)/*High Speed Packet Access* (HSPA) com GPS, permitindo o acesso remoto à plataforma;
- Microprocessador TX27, que suportará as camadas de rede, LLC e aplicação.

4.1 Arquitectura da MAC

Ao longo deste capítulo irão ser abordados detalhes da implementação da camada MAC, com um maior foco na LMAC.

Na figura 4.2 está representada a arquitectura simplificada da camada de acesso. Note-se a presença de um controlador de interrupções, pois o microprocessador utilizado apenas possui uma linha de interrupções.

O interface entre o microcontrolador Microblaze e a LMAC é feito através do barramento *Peripheral Local Bus* (PLB), um barramento com uma largura de 32 bits disponibilizado pelas ferramentas de projecto da Xilinx para interligar microprocessadores com periféricos.

A LMAC foi modelada através da linguagem VHDL, enquanto que a UMAC foi definida em C correndo sobre o microprocessador Microblaze. No desenvolvimento deste sistema utilizaram-se ferramentas da Xilinx:

- Xilinx ISE Project Navigator para o desenvolvimento da LMAC;

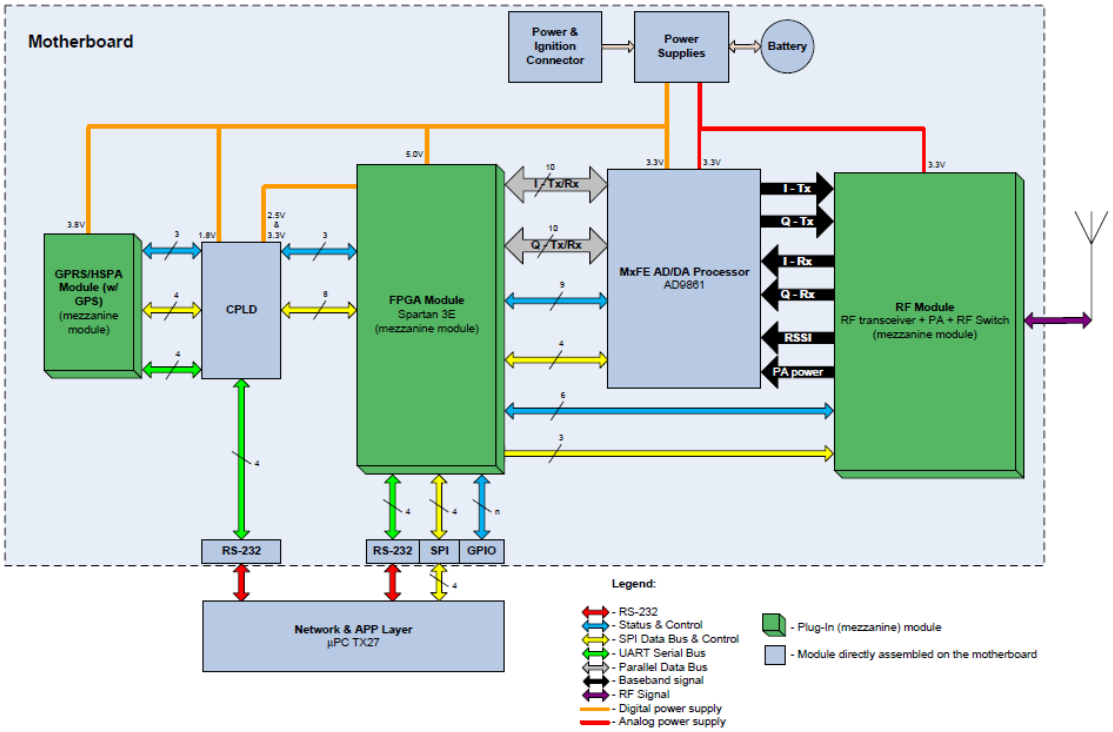


Figura 4.1: Arquitectura da plataforma desenvolvida.

- Xilinx Platform Studio para a criação do sistema embutido constituído pelo microprocessador, LMAC e barramentos PLB;
- Xilinx Software Development Kit para o desenvolvimento da UMAC.

4.1.1 Arquitectura da LMAC

Na figura 4.3 está representado um diagrama simplificado da arquitectura da LMAC. É esquematizada a interacção entre os vários processos em *hardware* e o fluxo dos dados dentro da LMAC. O interface com a UMAC é feito através do barramento PLB, através do qual o microprocessador poderá aceder a um dos portos da memória interna da LMAC e aos registos de controlo e de estado. Grande parte dos processos representados são controlados ou escrevem o seu estado em diversos campos no banco de registos, de acordo com o modelo de programação descrito em 3.4. O interface MAC-PHY é o descrito na figura 3.6.

4.2 Memória e Gestor de Memória

4.2.1 Memória

A memória utilizada na LMAC possui dois portos para leitura e escrita: um através do barramento PLB para o microprocessador e um segundo porto para acesso pela LMAC, para escrita de tramas recebidas ou leitura de dados para passagem à camada PHY e consequente

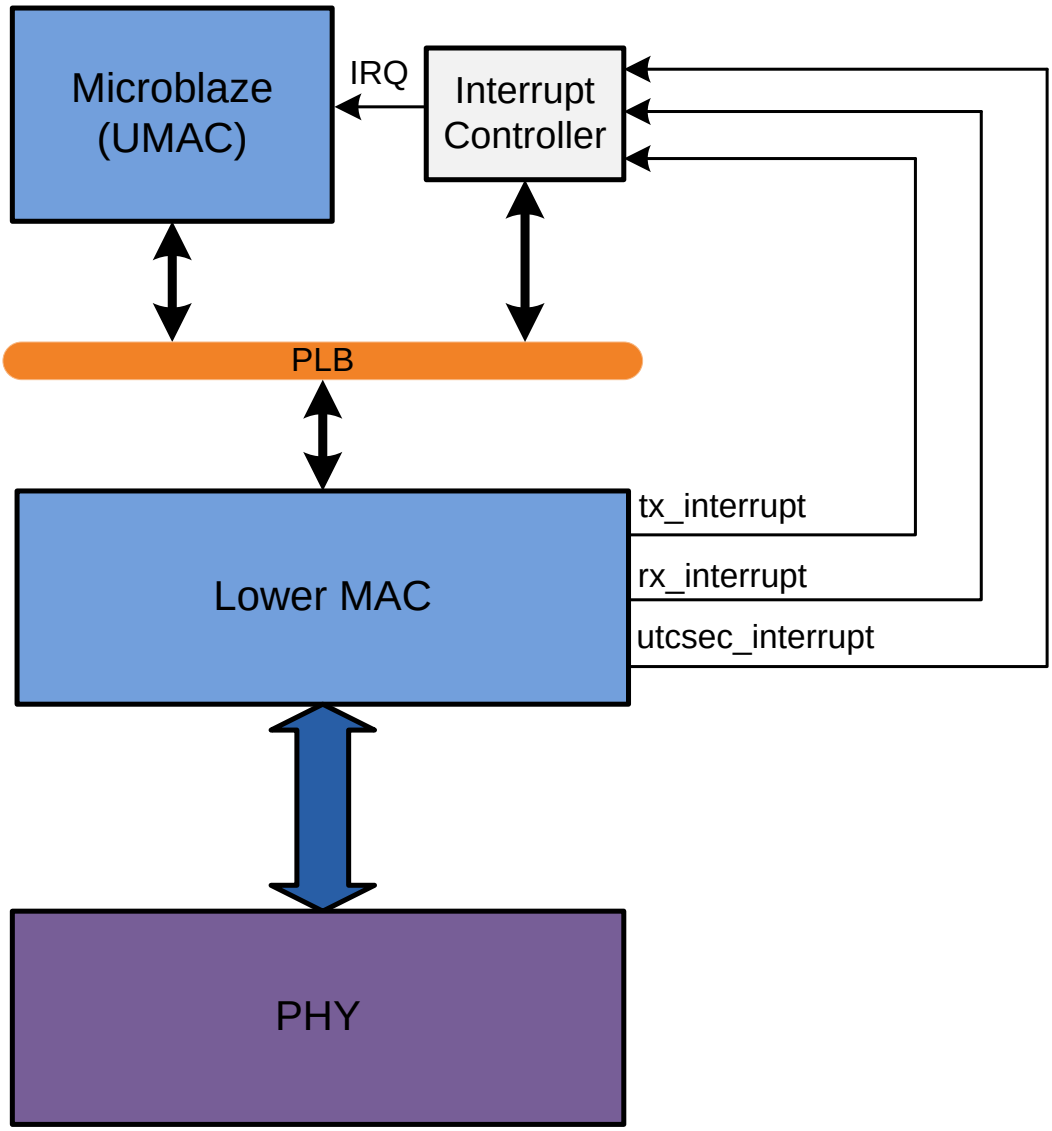


Figura 4.2: Arquitectura simplificada do sistema

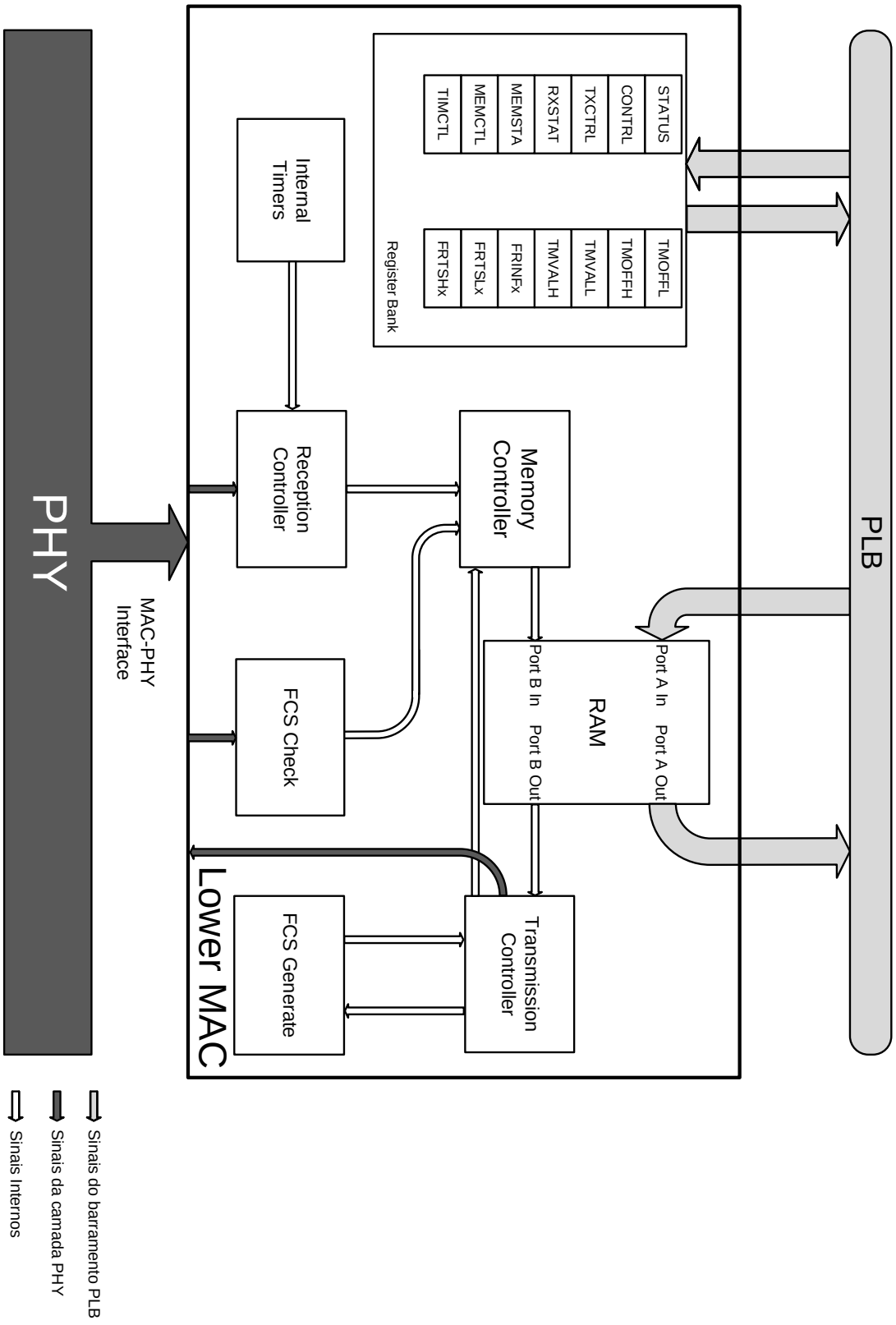


Figura 4.3: Arquitectura simplificada da LMAC.

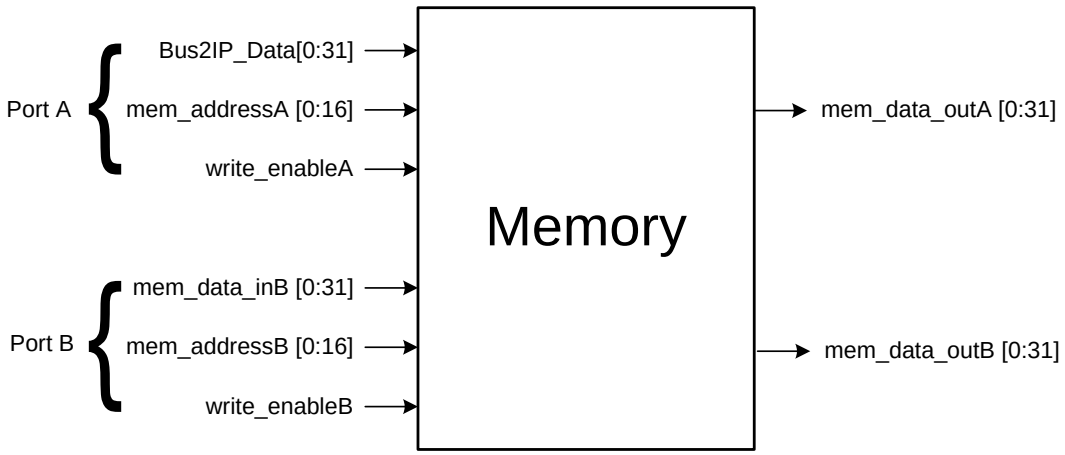


Figura 4.4: Sinais de interface da memória interna da LMAC.

transmissão para o meio. Esta memória é dividida em posições de tamanho fixo, contendo cada uma delas uma trama.

Na figura 4.4 está representado o interface da memória da LMAC. O porto A é utilizado pela UMAC enquanto que o porto B está destinado à LMAC.

4.2.2 Gestor de Memória

O controlador de memória em *hardware* desempenha as seguintes funções:

- Alocação de posições de memória à LMAC;
- Alocação de posições de memória à UMAC;
- Libertação de posições de memória;

Os sinais que constituem o interface do controlador de memória encontram-se representados na figura 4.5.

Alocação de posições de memória à LMAC

Quando a LMAC activa o sinal `mac_rx_slot_request`, sinalizando o pedido de alocação de memória para uma trama recebida da PHY, o controlador de memória irá:

1. Verificar a *flag* VALID para todos os registos FRINFn, até encontrar uma posição de memória sem dados válidos;
2. Caso encontre uma posição livre, irá colocar a '1' a *flag* VALID do registo FRINFn correspondente, decrementará o número de posições livres no sinal `frees`, colocará o número da posição alocada no sinal `curr_rx_slot_id` e activará o sinal `mac_rx_slot_grant`;

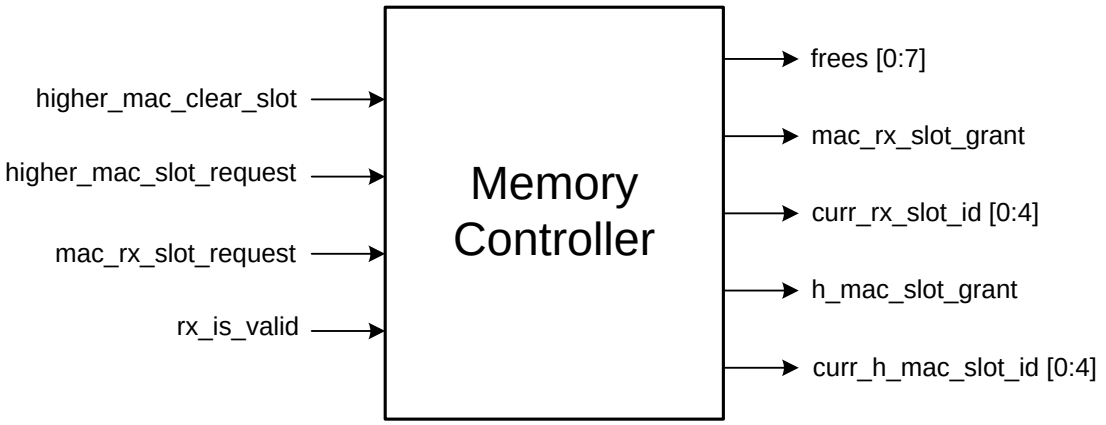


Figura 4.5: Sinais de interface do controlador de memória.

3. Se não fôr encontrada nenhuma posição livre o número de posições livres no sinal `frees` manter-se-á e o sinal `mac_rx_slot_grant` permanecerá desactivado;

4. No final da recepção dos dados, se a verificação de erros através do campo FCS revelar que a trama recebida contém dados inválidos irá ser colocada a '0' a *flag* VALID do registo FRINFn correspondente e o número de posições livres presente no sinal `frees` será incrementado. Caso não seja detectado nenhum erro nos dados não ocorre nenhuma alteração de valores.

Alocação de posições de memória à UMAC

A UMAC pode pedir a alocação de uma posição de memória à LMAC através do *set* no registo MEMCTL da *flag* ALCSL. A escrita desta *flag* irá activar o sinal `higher_mac_slot_request` e iniciar o processo de alocação para a UMAC:

1. O controlador de memória verificará a *flag* VALID para todos os registos FRINFn, até encontrar uma posição de memória sem dados válidos;

2. Caso encontre uma posição livre, irá colocar a '1' a *flag* VALID do registo FRINFn correspondente, decrementará o número de posições livres no sinal `frees`, colocará o número da posição alocada no sinal `curr_h_mac_slot_id` e activará o sinal `h_mac_slot_grant`;

Libertação de posições de memória

A libertação de posições de memória possui maior prioridade no controlador de memória que as outras duas operações, de modo a evitar que um pedido de alocação bloqueie um pedido de libertação. A libertação de uma posição de memória pode ser pedida pela UMAC através do *set* no registo MEMCTL da *flag* RLSSL, activando o sinal `higher_mac_clear_slot`, e da colocação no campo SLTID da posição de memória a libertar. O controlador de memória irá então:

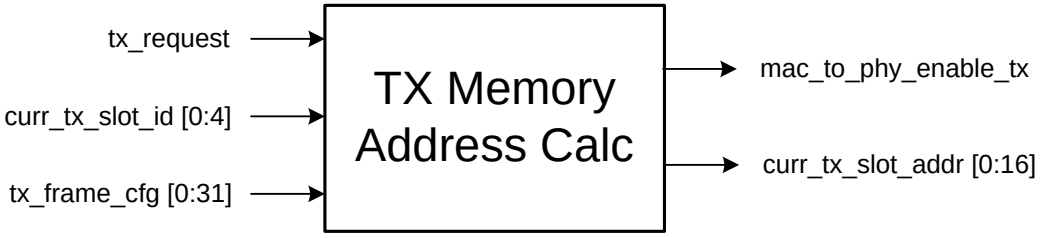


Figura 4.6: Sinais de interface do processo de cálculo do endereço de memória para transmissão.

1. Colocar a '0' a *flag* valid do registo FRINFn correspondente ao valor colocado no campo SLTID e decrementar o número de posições livres;

2. Se a posição indicada pelo campo SLTID estivesse já livre, o número de posições livres manter-se-ia.

4.2.3 Outros processos

Estão ainda implementados na LMAC processos adicionais que realizam certos cálculos e a multiplexagem de alguns sinais.

Cálculo de Endereços

Na figura 4.6 encontra-se representado o interface do processo que calcula o endereço inicial de memória a ser transmitido pela LMAC.

Este processo recebe como entradas os sinais `tx_request`, activado quando ocorre um pedido de transmissão por parte da UMAC, `curr_tx_slot_id`, a posição de memória a transmitir, e `tx_frame_cfg`, que contém os dados do registo FRINFn correspondente à posição dada pelo sinal `curr_tx_slot_id`, que corresponde ao valor escrito pela UMAC no campo SLTID no registo TXCTRL.

O sinal de saída `mac_to_phy_enable_tx` será activado caso a *flag* VALID do sinal `tx_frame_cfg` esteja a '1'. O valor do sinal `curr_tx_slot_addr` corresponderá ao endereço inicial de memória a partir do qual os dados naquela posição de memória se encontram e é dado pela soma do campo MSGOF do sinal `tx_frame_cfg` com o produto do sinal `curr_tx_slot_id` e o número de *bytes* de cada posição de memória.

O valor do endereço inicial de memória para a transmissão de dados à PHY (sinal `curr_tx_slot_addr`)é simplesmente dado pelo produto do número da posição alocada (sinal `curr_rx_slot_id`) com o número de *bytes* de cada posição.

Multiplexagem de Sinais

Através do processo esquematizado na figura 4.7 é seleccionado um dos registos FRINFn para constituir o sinal `tx_frame_cfg` a partir do sinal `curr_tx_slot_id`.

O processo descrito na figura 4.8 selecciona o endereço correcto de acesso à memória para a LMAC, escolhendo entre o endereço calculado pelo controlador de transmissão (`tx_mem_`

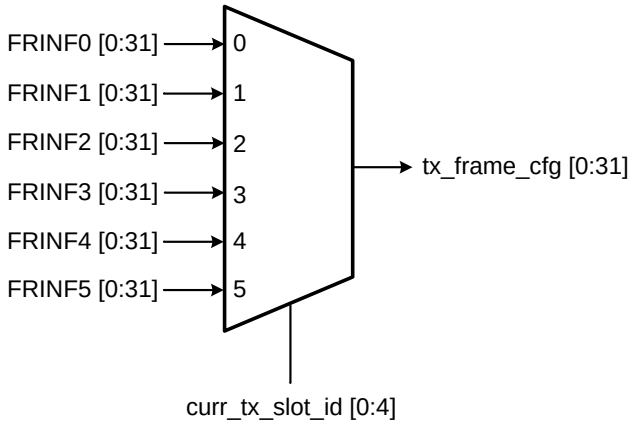


Figura 4.7: Multiplexagem dos vários registos FRINFn no sinal `tx_frame_cfg`.

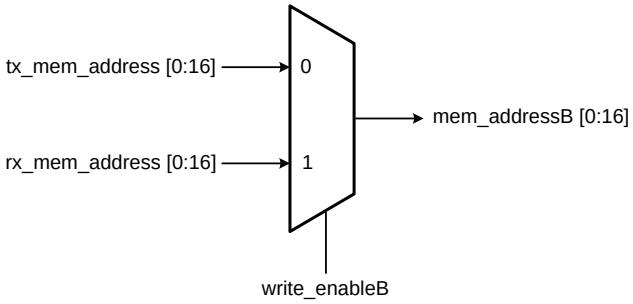


Figura 4.8: Multiplexagem dos endereços calculados pelo controlador de transmissão e controlador de recepção no sinal `mem_addressB`.

`address`) e o calculado pelo controlador de recepção (`rx_mem_address`). Utiliza-se o sinal `write_enableB` como critério pois quando este está activo, significa que o controlador de recepção está a aceder à memória para escrita no endereço `rx_mem_address`.

4.3 Controlador de Recepção

A passagem de uma trama da camada PHY para a camada MAC é iniciada pelo sinal `PHY_RXSTART_indication`. De seguida a PHY activará o sinal `PHY_RXDATA_indication` indicando a presença de dados válidos no sinal `RXDATA`.

O controlador de recepção da LMAC tem como função comunicar com a camada PHY e o controlador de memória, de forma a assegurar que uma trama recentemente recebida seja colocada correctamente em memória. Os sinais de interface deste controlador encontram-se descritos na figura 4.9.

Este controlador foi implementado com o recurso a uma máquina de estados, cujo diagrama é representado na figura 4.10.

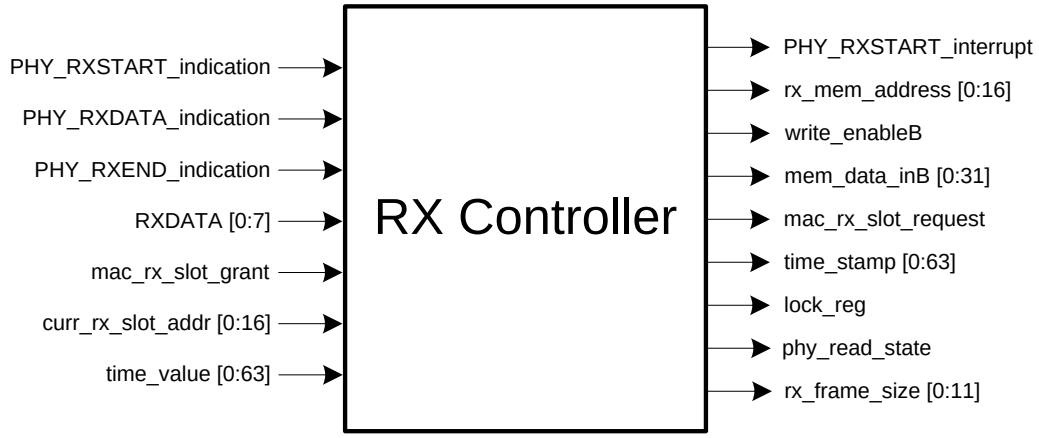


Figura 4.9: Sinais de interface do controlador de recepção.

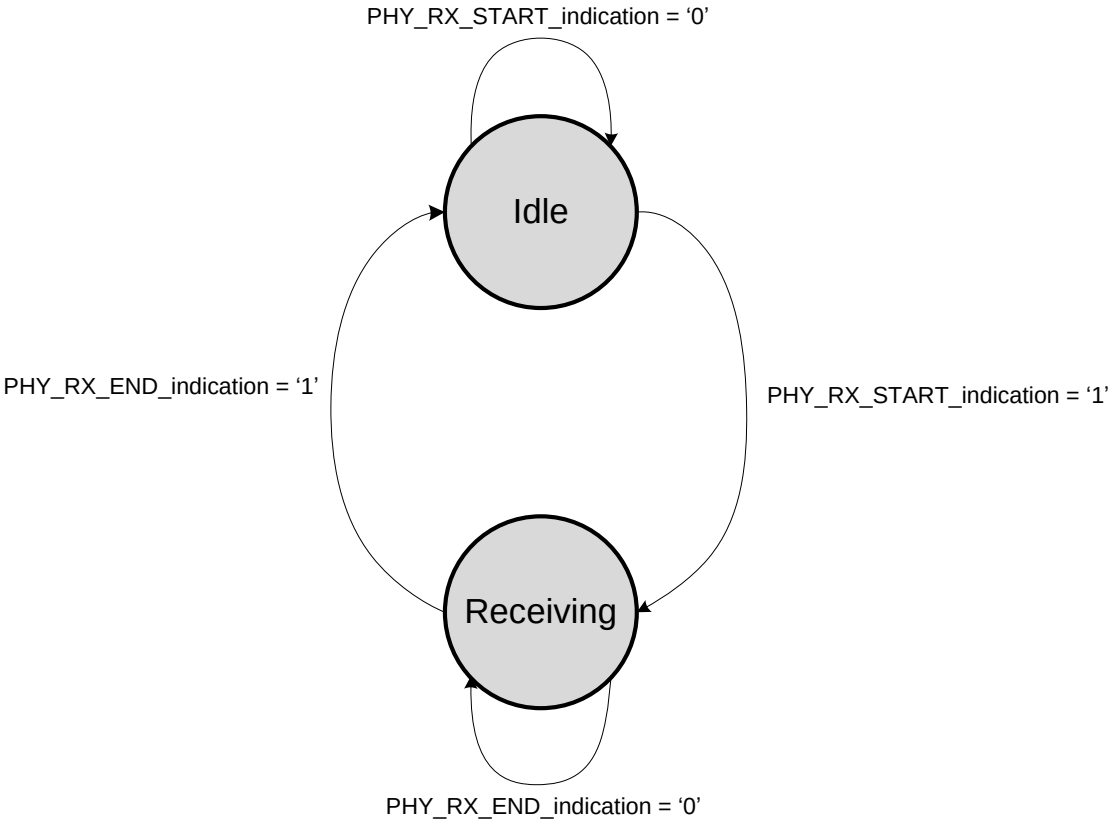


Figura 4.10: Máquina de estados de recepção.

A recepção de uma trama pode-se dividir em 4 passos:

- Início de recepção e alocação de posição de memória;
- Recepção dos dados e colocação destes em memória;
- Fim da recepção de dados;
- Verificação do campo FCS.

4.3.1 Alocação de posição de memória

Quando a camada PHY está pronta para começar a passar uma trama recentemente recebida à camada MAC, activará o sinal `PHY_RXSTART_indication`. Ao receber este sinal a máquina de estados de recepção da LMAC passará ao estado *Receiving* e:

1. O controlador de recepção activa o sinal `mac_rx_slot_request`, pedindo ao gestor de memória um *slot* para colocar a trama a ser lida da PHY e guarda o valor do tempo actual do sistema, contido no sinal `time_value`, no sinal `time_stamp`;
2. Caso o gestor de memória consiga alocar uma posição, este comunicará ao controlador de recepção o endereço inicial de memória da posição através do sinal `curr_rx_slot_addr` e a validade desta alocação activando o sinal `mac_rx_slot_grant`. O controlador de recepção irá então desactivar o sinal `mac_rx_slot_request`.
3. Caso não tenha sido alocada uma posição de memória até terem sido passados 3 *bytes* da camada PHY, o sinal `mac_rx_slot_request` será desactivado e a trama será efectivamente descartada. Isto deve-se ao facto de o acesso à memória ser *word-wide* enquanto que a troca de dados entre LMAC e PHY é *byte-wide*, não sendo assim estritamente necessário que exista uma posição alocada até que 3 *bytes* tenham sido recebidos.

4.3.2 Recepção dos dados e colocação destes em memória

1. Os dados são lidos da camada PHY através do sinal `RXDATA`, cuja validade é indicada pelo sinal `PHY_RXDATA_indication`;
2. Quando existirem 4 *bytes* no controlador de recepção prontos a ser escritos em memória, o sinal `write_enableB` será activado, o endereço de escrita em memória será passado em `rx_mem_address` e os dados a ser escritos estarão presentes no sinal `mem_data_inB`;
3. À medida que os dados vão sendo escritos em memória o sinal `rx_mem_address` vai sendo incrementado de forma a reflectir o novo endereço da palavra a ser escrita na memória da LMAC;

4.3.3 Fim da recepção de dados

1. Quando a trama tiver sido passada na totalidade à LMAC, a PHY activará o sinal `PHY_RXEND_indication`;
2. O controlador de recepção notificará o processador da recepção de novos dados através da activação da interrupção `PHY_RX_interrupt`;

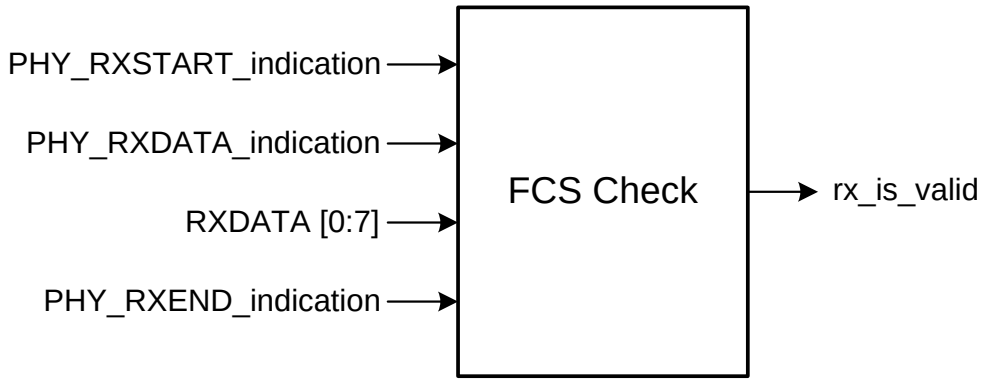


Figura 4.11: Sinais de interface do módulo de verificação do campo FCS.

3. Os dados recebidos ainda em *buffer* na LMAC serão escrito em memória: como a escrita em memória é *word-wide* se a trama recebida da PHY não tiver um número de *bytes* múltiplo de 4 ficariam dados por escrever na memória caso não se procedesse a este último passo.

4.3.4 Verificação do campo FCS

A verificação do campo FCS para uma trama recebida é feita *on-the-fly* à medida que esta é transmitida da camada PHY para a camada MAC. O interface do módulo de verificação do campo FCS está esquematizado na figura 4.11.

De modo a verificar a validade dos dados recebidos o módulo:

1. Começa por realizar um *reset* ao receber o sinal `PHY_RXSTART_indication`;

2. Calcula o CRC para toda a trama (incluindo o campo FCS), cujos dados são recebidos através do sinal `RXDATA` e validados pelo sinal `PHY_RXDATA_indication`;

3. Quando a PHY sinaliza o fim da trama através do sinal `PHY_RXEND_indication` o valor de CRC calculado para uma trama válida será o polinómio descrito na norma IEEE 802.11 [13]:

   $G(x) = x^{31} + x^{30} + x^{26} + x^{25} + x^{24} + x^{18} + x^{15} + x^{14} + x^{12} + x^{11} + x^{10} + x^8 + x^6 + x^5 + x^4 + x^3 + x + 1$

4. Caso a trama seja dada como válida o sinal `rx_is_valid` será activado.

4.3.5 Fila de Recepção

Está também implementada em *hardware* uma fila para tramas recebidas, contendo os números de posições de memória para tramas recebidas que ainda não foram atendidas pela UMAC. O seu interface encontra-se esquematizado na figura 4.12.

O seu funcionamento é o seguinte:

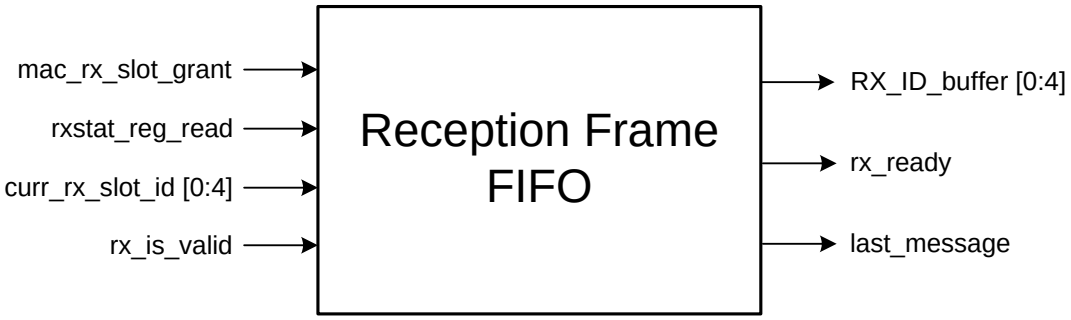


Figura 4.12: Sinais de interface da fila de recepção.

- Quando o sinal `rx_valid` e `mac_rx_slot_grant` são activados simultaneamente, sinalizando a chegada de uma trama válida e a alocação desta a uma posição de memória, é adicionado à fila o ID contido no sinal `curr_rx_id`;

- Quando a UMAC realiza uma leitura no registo RXSTATUS activando o sinal `rxstat_reg_read`, é realizado um *shift* na fila colocando no sinal `RX_ID_buffer` a próxima posição de memória ainda não atendida pela UMAC;

- Caso exista uma ou mais tramas na fila o sinal `rx_ready` é activado; caso haja apenas uma trama à espera de ser atendida é activado o sinal `last_message`.

4.4 Controlador de transmissão

A transmissão de uma trama da camada MAC para a camada PHY é iniciada pela UMAC, pela activação da flag TXRQT no registo TXCTRL, indicando também a posição de memória a ser transmitida através do campo SLTID. Caso a posição de memória a ser transmitida contenha dados válidos, será activado o sinal `mac_to_phy_enable_tx` (Secção 4.2.3).

O controlador de transmissão da LMAC tem como função ler os dados em memória da posição a ser transmitida à PHY e transmiti-los através do interface e sequência de troca de sinais descrito na secção 3.5.2. O interface do módulo propriamente dito é apresentado na figura 4.13.

À semelhança do controlador de recepção, também o controlador de transmissão é implementado com recurso a uma máquina de estados. Um diagrama dos seus estados e transições possíveis é representado na figura 4.14.

A transmissão de dados da LMAC para a camada PHY pode ser dividida em 4 passos:

1. Início de transmissão;

2. Transmissão de dados à PHY;

3. Transmissão de CRC calculado;

4. Final de transmissão.

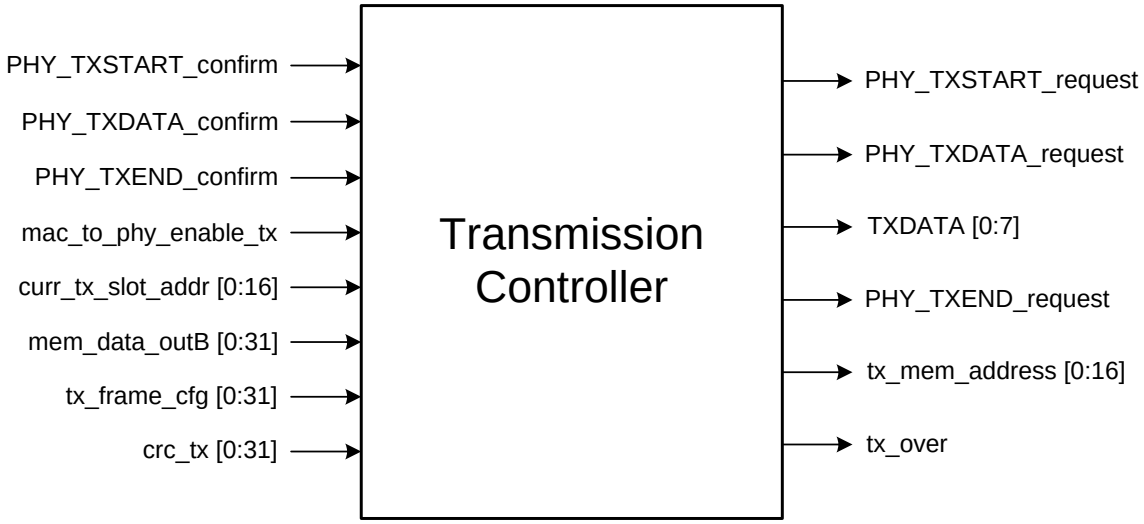


Figura 4.13: Sinais de interface do controlador de transmissão.

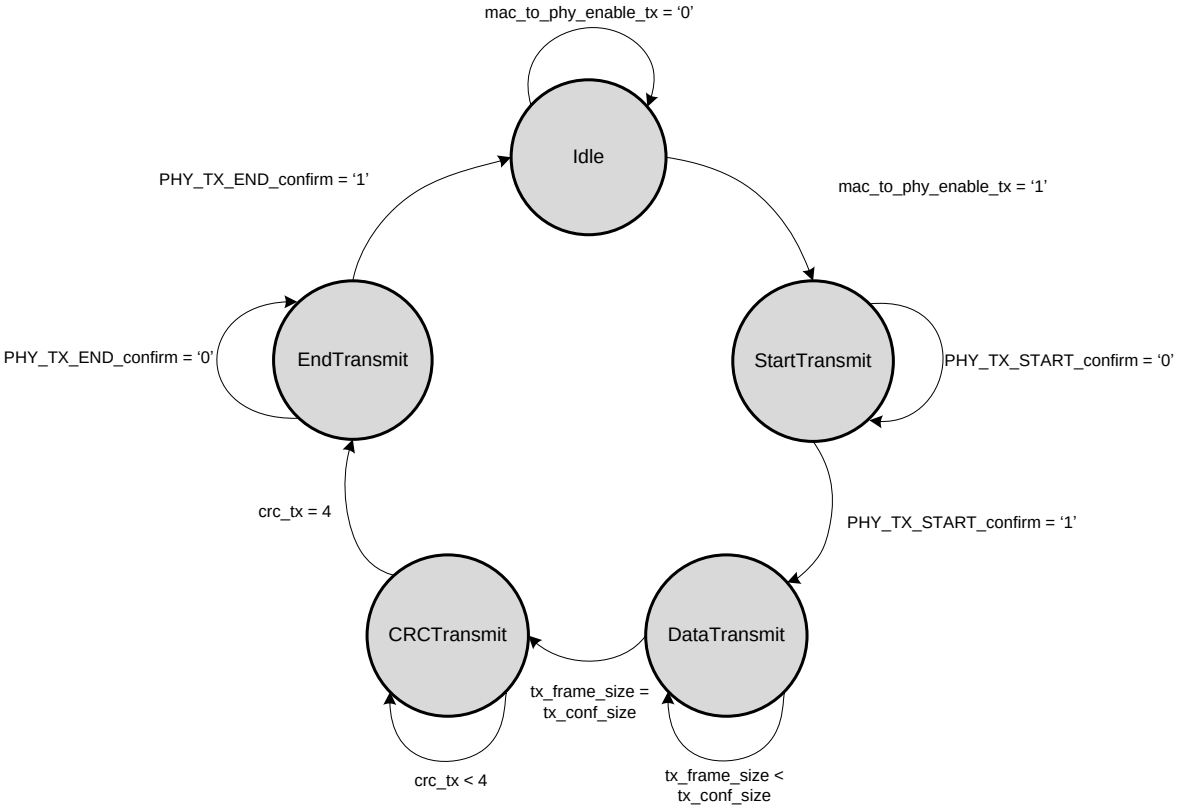


Figura 4.14: Máquina de estados de transmissão.

4.4.1 Início de transmissão

A operação do controlador de transmissão é iniciada pela activação do sinal `mac_to_phy_enable_tx` pelo gestor de memória. Ao ser activado este sinal o controlador de transmissão irá transitar para o estado *StartTransmit* e irá:

1. Activar o sinal `PHY_TXSTART_request`, notificando a PHY do desejo de realizar uma transmissão;

2. Quando a camada PHY responder activando o sinal `PHY_TXSTART_confirm`, o sinal `PHY_TXSTART_request` será colocado novamente a '0' e a máquina de estados transitará para estado *DataTransmit*.

4.4.2 Transmissão de dados à PHY

De forma a transmitir dados à camada PHY o controlador de transmissão irá:

1. Ler os dados no endereço de memória indicado pelo sinal `tx_mem_address`, que é obtido pela soma entre o endereço no sinal `curr_tx_slot_addr` e um *offset* correspondente ao número de palavras já transmitidas na totalidade à PHY;

2. Colocar o *byte* correspondente no sinal TXDATA e activar o sinal `PHY_TXDATA_request`;

3. Ao receber o sinal `PHY_TXDATA_confirm` da camada PHY, o controlador de transmissão é notificado da correcta passagem dos dados à PHY e enviar-lhe-à um novo *byte*. O controlador de transmissão incrementará ainda um contador interno que indica o número de bytes já transmitidos (sinal `tx_frame_size`);

4. Caso a última palavra lida da memória não tenha sido ainda transmitida na totalidade, será passado à PHY o próximo *byte* desta mesma palavra;

5. Caso a palavra tenha já sido transmitida completamente, o controlador de transmissão incrementará o sinal `tx_mem_address` e lerá a próxima palavra a transmitir da memória;

6. Estes passos repetir-se-ão até que o sinal `tx_frame_size` iguale o valor em `tx_cnfg`: quando o número de *bytes* transmitidos à PHY fôr igual ao tamanho da trama configurado pela UMAC ocorrerá a transição de estado para *CRCTransmit* e o sinal `tx_over` será activado.

4.4.3 Passagem de CRC calculado

O interface do módulo encarregado de calcular o campo FCS está representado na figura 4.15.

O campo FCS é calculado *on-the-fly*, à medida que os dados vão sendo transmitidos à camada PHY pelo sinal `TXDATA` e validados pelo sinal `PHY_TXDATA_request`. A sua transmissão segue da seguinte forma dentro do controlador de transmissão:

1. Quando o controlador de transmissão activa o sinal `PHY_TXSTART_request`, o módulo de geração do campo FCS realiza o seu *reset*;

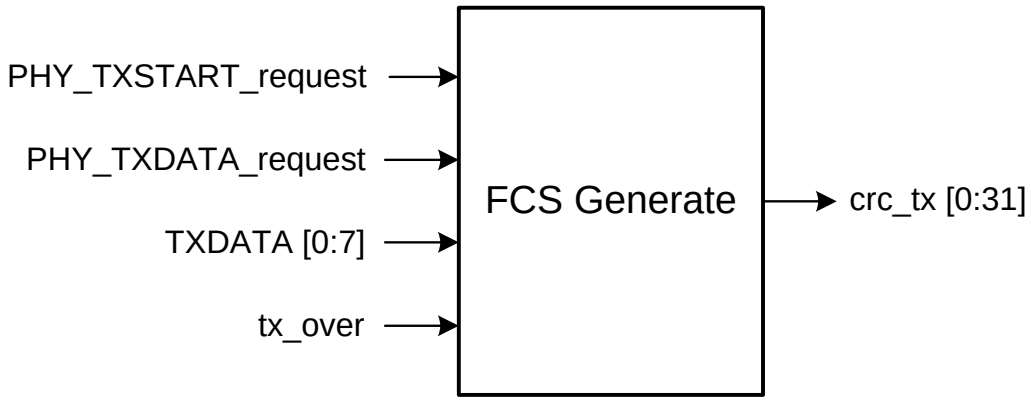


Figura 4.15: Interface do módulo gerador do campo FCS

2. O campo FCS está disponível para transmissão à PHY assim que os dados em memória tenham sido passados na sua totalidade através do sinal `TXDATA` e encontra-se no sinal `crc_tx`;

3. O controlador de transmissão colocará o campo FCS um *byte* de cada vez no sinal `TXDATA` e activando o sinal `PHY_TXDATA_request`. O sinal `tx_over` evita que o módulo continue a calcular o CRC quando o campo FCS está a ser enviado;

4. Quando a PHY activar o sinal `PHY_TXDATA_confirm`, irá ser passado o próximo *byte* do campo FCS;

5. Este processo repete-se até que os 4 *bytes* do campo FCS tenham sido transmitidos à PHY, causando a transição da máquina de estados para o estado *EndTransmit*.

4.4.4 Final de transmissão

Neste estado é finalizada a troca de dados entre LMAC e PHY:

1. O controlador de transmissão activará o sinal `PHY_TXEND_request`;

2. Quando a PHY activar o sinal `PHY_TXEND_confirm` ocorrerá a transição para o estado *Idle* e será activada a interrupção `PHY_TX_interrupt`, notificando a UMAC do sucesso da transmissão dos dados;

4.5 Controlador de registos

Um outro módulo presente na LMAC que realiza algumas operações ao nível dos registos é o controlador de registos. O seu interface encontra-se representado na figura 4.16.

O funcionamento deste módulo é o seguinte:

1. Através dos sinais `mac_rx_slot_request`, `mac_rx_slot_grant` e `phy_read_state` o controlador de registos determina se está a decorrer, num dado momento, a recepção de uma trama da camada PHY;

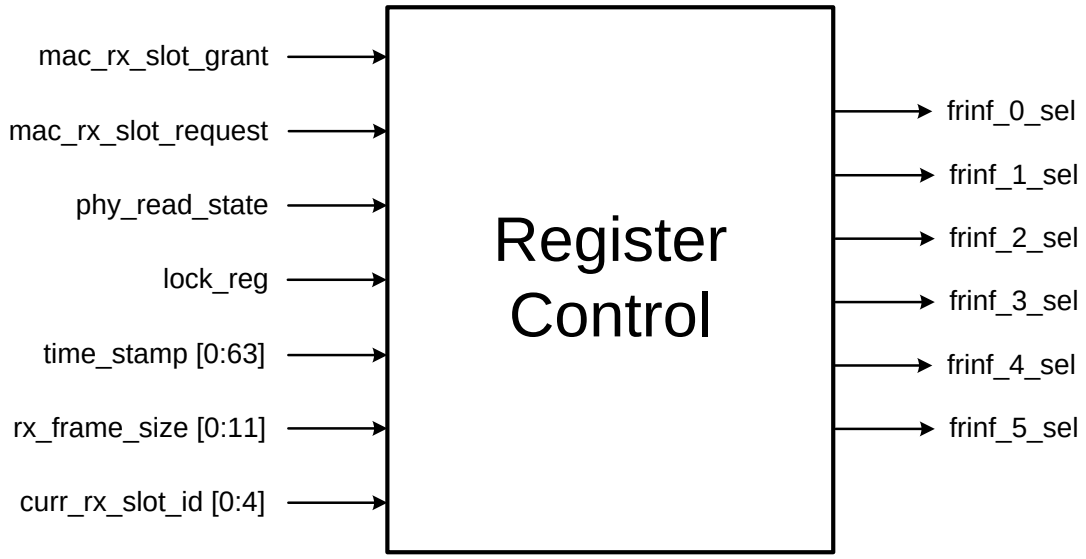


Figura 4.16: Sinais de interface do móduloe controlador de registos.

2. Caso a LMAC esteja a receber dados da PHY, o controlador de registos activará um dos seus sinais de saída de acordo com o sinal `curr_rx_slot_id`. Ao ser activada uma destas saídas garante-se que o registo FRINFn correspondente não será alterado pela UMAC até ao fim da recepção de dados;

3. O controlador de registos coloca ainda o valor do sinal `time_stamp` nos registos TSTHIn e TSTLOn correspondentes, assim como o tamanho da trama recebida no registo FRINFn, passado através do sinal `rx_frame_size`. O controlador de registos preenche tambem o campo MSGOF no registo FRINFn em questão com o valor 0, uma vez que o *offset* em relação ao endereço inical de memória da posição para tramas recebidas é sempre nulo;

4. Quando a recepção de dados da PHY termina, as saídas do controlador de registos são colocadas a '0' e a UMAC pode alterar os registos FRINFn.

4.6 *Timers* internos

O diagrama de blocos simplificado para os *timers* internos da LMAC encontra-se representado na figura 4.17. Estão também representados os registos através dos quais a UMAC pode interagir com o sistema.

O sistema é constituído por dois contadores, efectuando um deles a divisão da frequência interna da FPGA para 1 microssegundo, enquando que outro contador incrementa o valor de TMVAL a cada microssegundo. O valor deste factor de divisão é configurado pela UMAC para permitir a sincronização da estação. Caso se deseje utilizar a linha de interrupção `utcsec_interrupt`, existe ainda um outro contador que é também incrementado a cada microssegundo, gerando uma interrupção quando atinge 1×10^6.

Será possível no futuro sincronizar os *timers* implementados com o módulo GPS presente

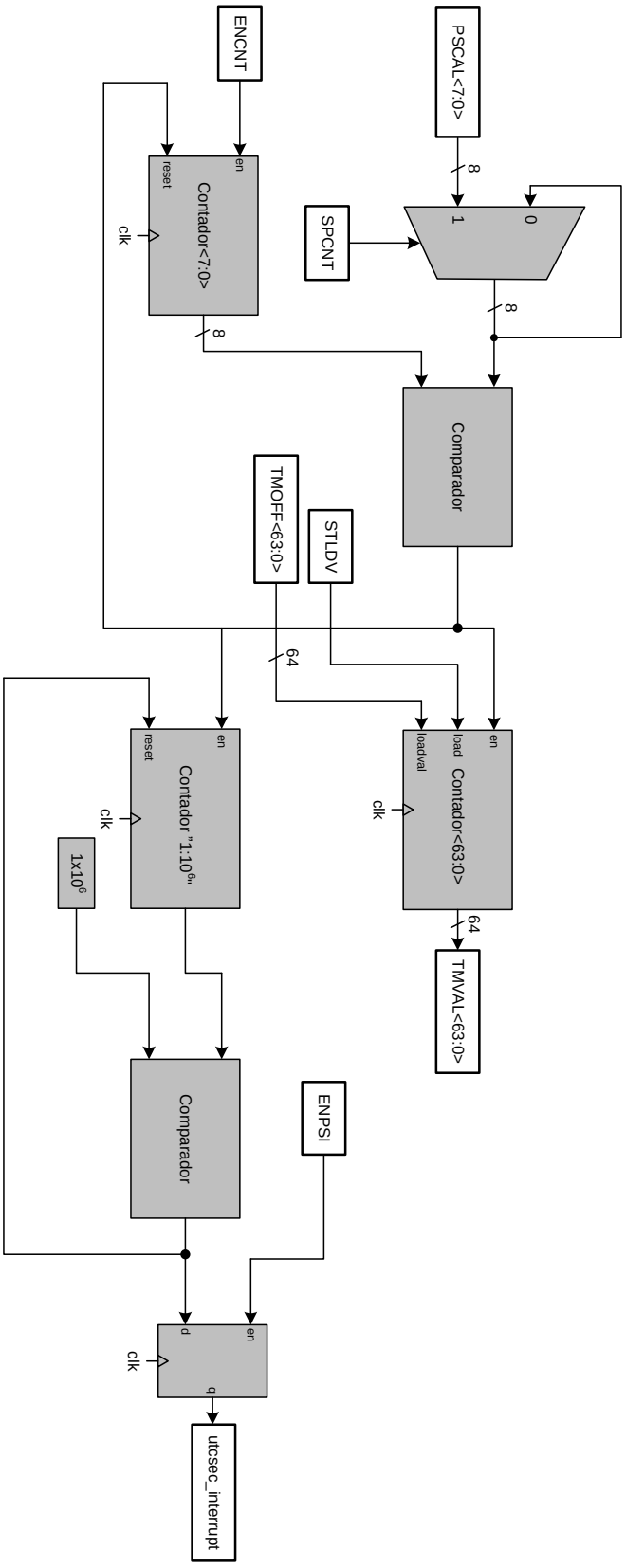


Figura 4.17: Diagrama de blocos simplificado para *timers*.

na plataforma de desenvolvimento, dando assim acesso à MAC a um tempo universal com um baixo erro.

4.7 Controlo de acesso ao meio

O controlo de acesso ao meio realizado é puramente virtual, i.e., através da reserva do meio por parte de uma estação durante um certo tempo. Esta reserva pode ser realizada através do envio de tramas RTS e CTS: o campo *duration* neste tipo de tramas é utilizado para actualizar o NAV local, indicando que o meio estará ocupado nos próximos microssegundos. O valor do NAV vai sendo decrementado e quando fôr igual a zero, a estação poderá tentar aceder ao meio. O processo de controlo de acesso ao meio virtual foi já explorado na secção 2.3.1.

Para oferecer suporte ao controlo de acesso ao meio virtual, cujo algoritmo está implementado na UMAC, foi implementado um contador decrescente cujo esquema simplificado está apresentado na figura 4.18.

O valor do campo CNTVA é carregado no contador quando a *flag* ENCNT é colocada a '1', iniciando a contagem. Quando o contador atinge zero, gera uma interrupção à UMAC. Caso a *flag* STCNT tenha sido colocada a '1' antes do contador ter terminado a sua contagem, o seu valor actual é colocado no campo CNTVA para possível leitura da UMAC.

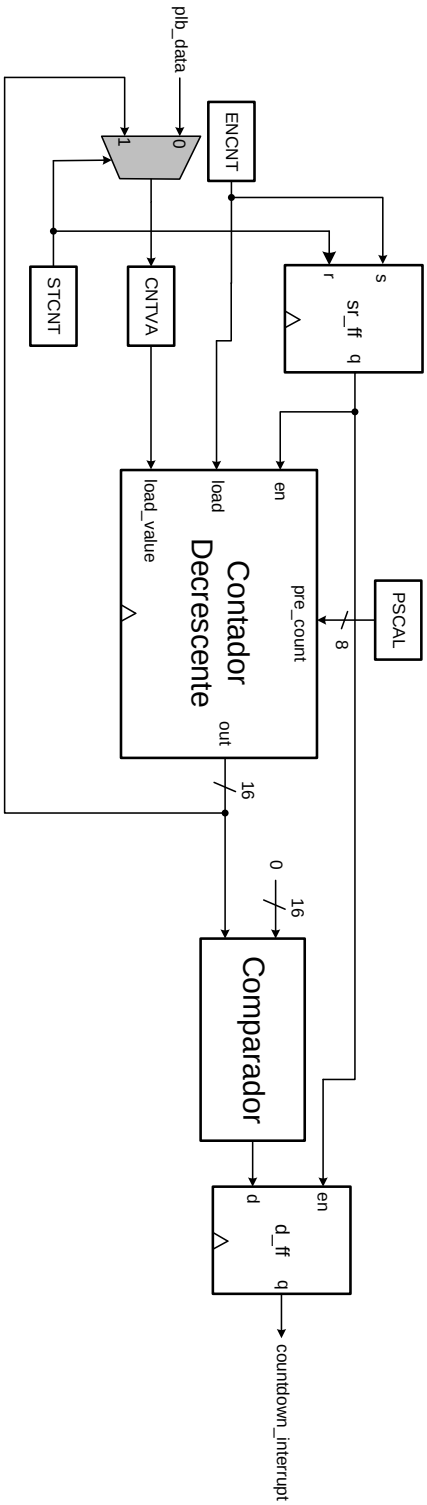


Figura 4.18: Esquema simplificado do contador decrescente da LMAC.

Capítulo 5

Validação

Ao longo deste capítulo irão ser apresentados os vários testes efectuados à MAC tanto no seu funcionamento geral como na sua correcção temporal. Os testes realizados foram, em geral, mais focados na LMAC.

Como o desenvolvimento da camada PHY seguiu em paralelo com o do da camada MAC, tornou-se necessário emular a camada PHY para efeitos de teste de troca de tramas. Para esse efeito utilizou-se como camada PHY uma porta série com lógica adicional que emula os sinais de interface entre a camada MAC e a camada PHY. Consegue-se assim através de um terminal receber e transmitir tramas MAC, ou trocar tramas entre duas estações. Este sistema encontra-se esquematizado na figura 5.1.

Irão ser realizados vários testes à LMAC divididos em duas categorias principais:

- Verificação funcional:
  - Transmissão e recepção de tramas;
  - Descodificação e criação de tramas;
  - Geração e validação do campo FCS;
  - Gestao de Memória e fila de recepção;
- Desempenho e correcção temporal:
  - Latência da LMAC;
  - Temporizadores, interrupções e *timestamping*;

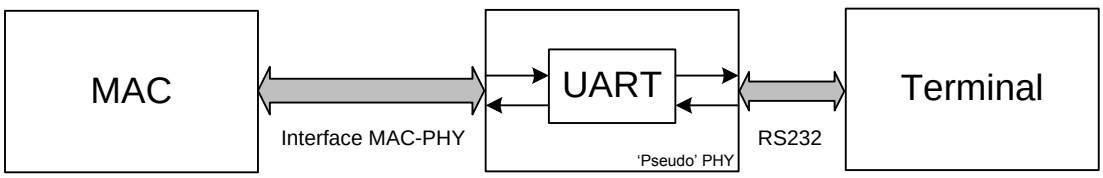


Figura 5.1: Esquematização da configuração do sistema para teste.

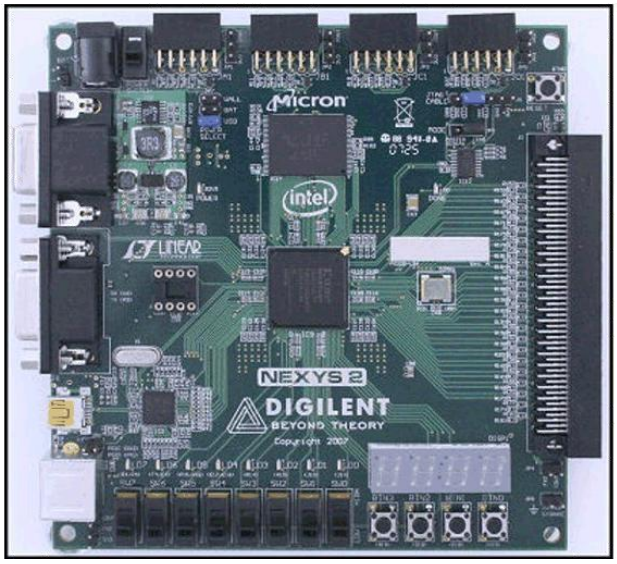


Figura 5.2: Placa de desenvolvimento Nexy2-500.

5.1 Placa de Desenvolvimento

A placa de desenvolvimento utilizada na validação da MAC foi a Nexys2-500 da Digilent 5.2. Esta placa contém uma FPGA Spartan3E-500, suficiente para albergar tanto a LMAC como o processador sobre o qual a UMAC correrá para além de diversos periféricos. Em termos de memória de dados e intruções foram utilizadas as memórias Intel StrataFlash e Micron PSDRAM, ambas de 16MB de capacidade. A memória interna da FPGA revelou-se insuficiente para conter todo o código e dados necessários ao funcionamento da UMAC, para além da memória da LMAC. Esta placa de desenvolvimento permite a comunicação com um terminal através de porta série ou a comunicação entre outra placa, através de porta série ou de conectores pmod.

5.2 Verificação Funcional

5.2.1 Transmissão e recepção de dados

Pretende-se através deste teste validar os mecanismos mais básicos da LMAC, nomeadamente o envio de dados da MAC para a PHY e a recepção de dados de um terminal. O envio de uma sequência de dados da MAC envolverá:

- Alocação de memória à UMAC;
- Escrita da memória da LMAC;
- Interface com a camada PHY para transmissão de dados.

A recepção de dados de um terminal envolve:

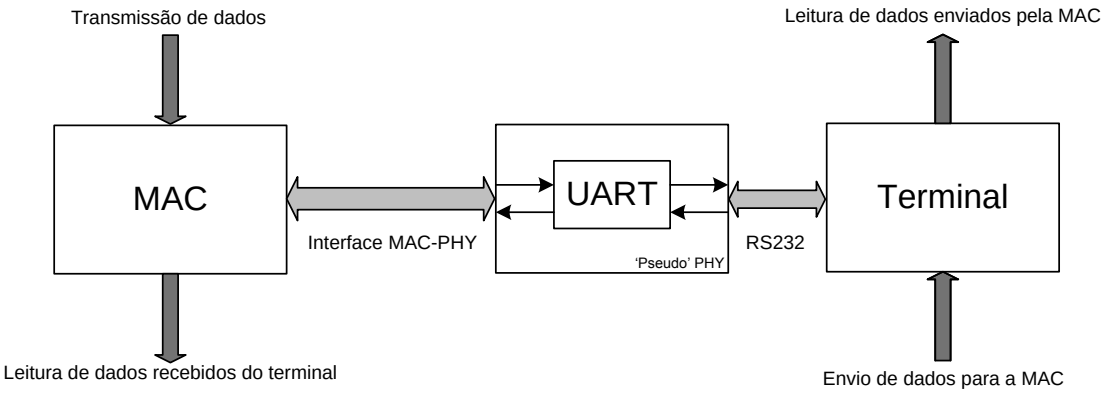


Figura 5.3: Esquematização dos testes realizados para validação da transmissão e recepção de tramas.

- Alocação de memória à LMAC;
- Leitura de dados da memória da LMAC;
- Interface com a camada PHY para recepção de dados;
- Activação de interrupção de recepção de novos dados.

Para além dos itens referidos, estará também envolvida a leitura e escrita de registos da LMAC em ambos os testes. Os testes realizados estão esquematizados na figura 5.3.

Resultados

Verificou-se que ao transmitir dados da camada MAC estes eram observados com sucesso no terminal através de porta série.

Ao enviar dados do terminal para a MAC, foi também possível lê-los na totalidade a partir da memória interna da LMAC.

Discussão

Através de dois simples testes foi possível verificar o funcionamento básico de vários mecanismos da LMAC e da arquitectura concebida:

- Acesso à memória por ambos os portos de acesso, tanto para leitura como para escrita;
- Alocação de posições de memória para a UMAC e LMAC;
- Acesso aos registos de controlo e estado da LMAC;
- Interface com a camada PHY funcional;
- Geração de interrupções e interacção com o controlador de interrupções bem sucedida.

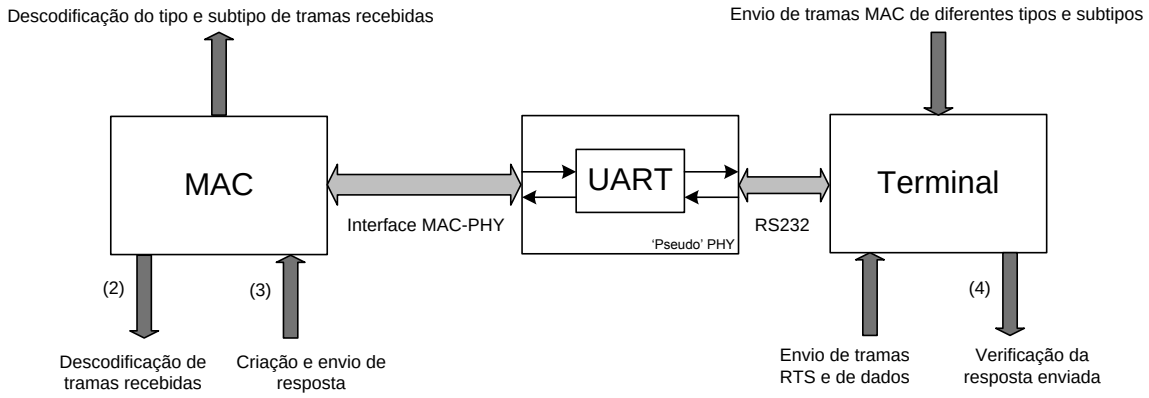


Figura 5.4: Esquematização dos testes realizados para descodificação e criação de tramas.

5.2.2 Descodificação e criação de tramas

Pretende-se através deste teste verificar uma das funcionalidades implementada na UMAC, neste caso a descodificação de tramas recebidas em termos de tipo e subtipo MAC. Este processo encontra-se descrito na figura 5.4.

Irão ser enviadas tramas MAC de vários tipos e subtipos, que terão de ser reconhecidas pela UMAC:

- Tramas de controlo:
  - Tramas ACK;
  - Tramas RTS;
  - Tramas CTS;
- Tramas de dados.

A UMAC terá também de gerar respostas às tramas recebidas:

- ACK em resposta a uma trama de dados;
- CTS em resposta a uma trama RTS;

Resultados

A UMAC revelou-se capaz de reconhecer correctamente os vários tipos de tramas recebidos pela LMAC.

A UMAC respondeu ainda com sucesso a tramas de dados e RTS, com tramas ACK e CTS respectivamente.

Discussão

Através do correcto reconhecimento de tramas pela UMAC foi possível a geração de respostas a vários tipos de tramas e confirmar adicionalmente o funcionamento dos mecanismos de transmissão e recepção de tramas implementados na LMAC.

Tipo de Trama	Reconhecido?
Controlo-ACK	Sim
Controlo-RTS	Sim
Controlo-CTS	Sim
Dados	Sim

Tabela 5.1: Resultados dos testes de descodificação de tramas pela UMAC.

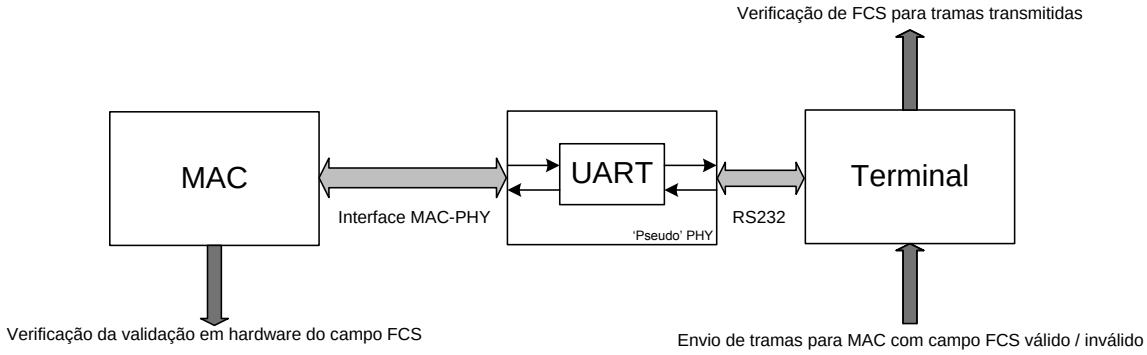


Figura 5.5: Esquematização dos testes realizados para validação da geração e verificação do campo FCS.

5.2.3 Geração e validação do campo FCS

Pretende-se através deste teste validar os mecanismos implementados de geração do campo FCS para tramas transmitidas à PHY e de verificação de CRC para tramas recebidas. Pretende-se também verificar que o campo FCS calculado está de acordo com o definido no protocolo.

Irão ser transmitidas várias tramas e comparado o seu FCS, gerado na LMAC, com o calculado através de calculadores *online* ([21], [22] e [23]). Para a verificação do mecanismo de validação irão ser enviadas para a MAC várias tramas, tanto com um FCS válido como inválido, e observado se esta detecta erros nas tramas ou não.

O procedimento de teste da geração e validação do campo FCS encontra-se representado na figura 5.5.

Resultados (geração)

Na tabela 5.2 encontram-se os resultados obtidos para as várias tramas transmitidas: na primeira coluna encontram-se representados os dados transmitidos pela MAC, na segunda coluna os campo FCS correspondente gerado e na terceira coluna o campo FCS gerado em calculardores *online*.

Resultados (validação)

Na tabela 5.3 encontram-se os resultados obtidos para as várias tramas enviadas do terminal para a MAC: na primeira coluna encontram-se os dados transmitidos do terminal para

Dados Enviados (ASCII)	**FCS Gerado (Hex)**	**FCS Gerado em calculadores *online* (HEX)**
123456789	`CBF43926`	`CBF43926`
abcdefghi	`8DA988AF`	`8DA988AF`
ZYWXVUTSR	`797517D0`	`797517D0`

Tabela 5.2: Resultados dos testes de geração do campo FCS.

Dados Enviados (Hex)	**Válido/Inválido**	**CRC Calculado na recepção**
`31323334353661D37209`	Válido	`1CDF4421`
`616263646566EF398E4B`	Válido	`1CDF4421`
`75547961486633F57049`	Inválido	`5EFC9F29`
`3B22AAEF243F3E5F3C27`	Inválido	`DAD45A0F`

Tabela 5.3: Resultados dos testes de verificação do campo FCS.

a placa de desenvolvimento, na segunda coluna está especificado se os dados enviados constituem uma sequência válida ou não e na terceira coluna está presente o CRC calculado na recepção.

Discussão

Verificou-se que o campo FCS gerado em *hardware* para tramas transmitidas é o correcto quando comparado com os valores gerados em calculadores *online*, validando assim o mecanismo de geração e transmissão do campo FCS da LMAC.

Na verificação do campo FCS verificou-se que, tal como esperado, o CRC gerado para válidas recebidas é sempre o mesmo e igual a `1CDF4421`, a partir do qual se pode obter o polinómio resto para tramas validas descrito na norma IEEE 802.11 [13] (`C704DD7B`) através da negação de toda a palavra e inversão da ordem dos *bits* dentro de cada *byte*, sendo portanto um polinómio equivalente. Para tramas com campo FCS inválido, o CRC gerado na recepção difere do resto final válido e a sequência de dados é dada como inválida.

5.2.4 Gestão de memória e fila de recepção

Deseja-se através de uma série de testes verificar o correcto funcionamento do gestor de memória e da fila de recepção de mensagens. Em relação ao gestor de memória:

- Verificar que sucessivas alocações e libertações de memória não causam o mau funcionamento da memória, como por exemplo a não utilização de uma posição e outras situações;
- Verificar que a posição alocada é a correcta.

Quando à fila de recepção de mensagens pretende-se verificar que não existem incoerências em relação às posições alocadas na recepção de uma trama e aquelas colocadas na fila.

Os testes realizados encontram-se esquematizados na figura 5.6.

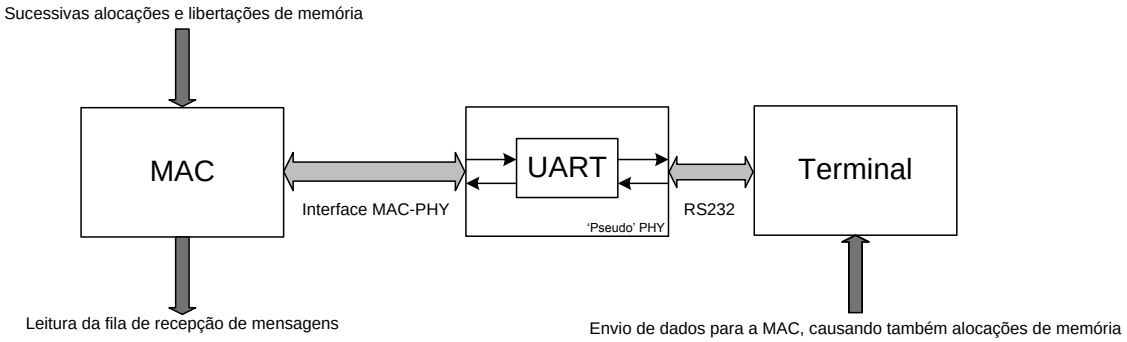


Figura 5.6: Esquematização dos testes realizados para validação do gestor de memória e fila de mensagens recebidas.

Resultados

Através de diversas sequências de alocação de tramas, tanto à UMAC como a tramas recebidas, e libertação de posições de memória verificou-se que em todas as situações os *slots* de memória foram sendo correctamente alocados e libertados, ocorrendo sempre a reutilização de posições previamente ocupadas e libertadas com normalidade e nunca ocorrendo a alocação para escrita de uma posição já ocupada.

Também a fila de recepção de mensagens funcionou correctamente, contendo apenas posições de memória com tramas recebidas e válidas. A fila foi também actualizada com sucesso a cada leitura por parte da UMAC.

Discussão

Tanto o gestor de memória como a fila de mensagens recebidas encontram-se a funcionar correctamente, sendo possível:

- Alocar e libertar posições sem nunca corromper a memória da LMAC;
- Ter várias mensagens em fila de espera na LMAC, sendo a fila correctamente actualizada sempre que ocorre uma leitura do registo RXSTAT ou a recepção de uma nova trama.

5.3 Desempenho e Correcção Temporal

5.3.1 Latência da LMAC

Pretende-se através deste teste medir o tempo desde que a UMAC ordena a transmissão de uma trama até que a primitiva `PHY_TXSTART_request` seja enviada à camada PHY. Para esse efeito enviaram-se 100 tramas consecutivas da LMAC para a PHY e mediu-se o tempo descrito através de um contador em *hardware*. O processo de teste encontra-se esquematizado na figura 5.7.

Resultados

Verificou-se que o atraso introduzido pela LMAC no envio de tramas para a camada PHY é de 3 ciclos de relógio, que na placa de desenvolvimento utilizada corresponde a um atraso

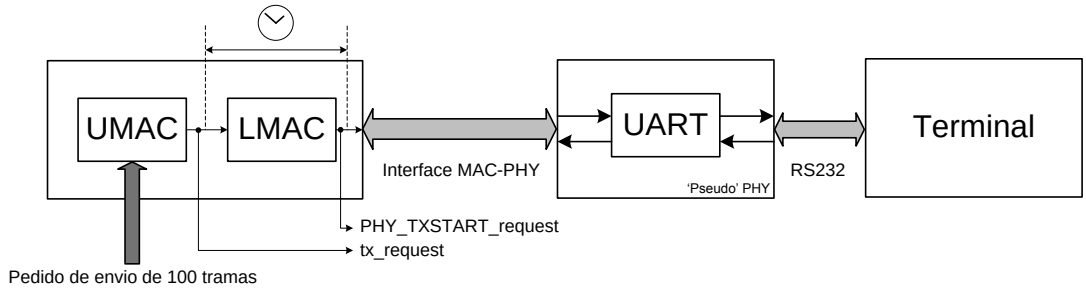


Figura 5.7: Esquematização dos testes realizados para quantificação da latência introduzida pela LMAC.

de 60 ns.

Teste	**Resultado (ns)**
Latência da LMAC no envio de tramas	60

Tabela 5.4: Resultados do teste à latência introduzida pela LMAC no envio de tramas para a camada PHY.

Discussão

O atraso entre o pedido da UMAC da transmissão de uma trama e o envio da primitiva `PHY_TXSTART_request` pela UMAC é muito pequeno, demonstrando-se assim que a LMAC introduz uma baixa latência na transmissão de dados.

5.3.2 Temporizadores, interrupções e *timestamping*

Com duas placas de desenvolvimento na configuração representada na figura 5.8 utilizou-se a interrupção despoletada a cada segundo pela LMAC para o envio de tramas entre as MACs. Esta experiência serviu não só para testar a activação de interrupções periódicas mas também o *timestamping* correcto de tramas recebidas.

Resultados

Na tabela 5.5 estão representados os resultados obtidos para este teste.

Discussão

Verifica-se que a diferença entre os *timestamps* de tramas consecutivas é sempre aproximadamente igual. Os valores não são iguais a 1×10^6 possivelmente por a frequência de funcionamento das duas placas de desenvolvimento não ser exactamente igual. Pôde-se também através deste teste verificar o correcto funcionamento dos *timers* internos, das interrupções periódicas e do *timestamping* de tramas recebidas.

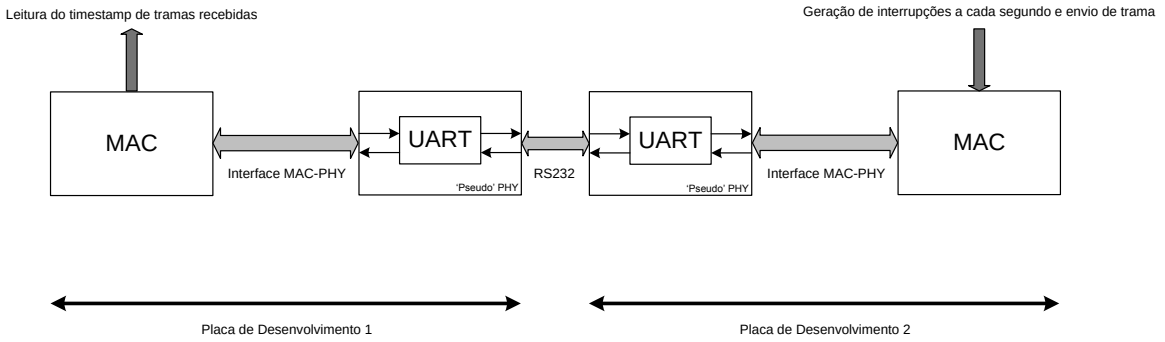


Figura 5.8: Esquematização dos testes realizados para validação dos temporizadores, interrupções periódicas e *timestamping* de tramas recebidas.

Número da trama	***Timestamp* de Recepção (Hex)**	**Diferença entre tramas consecutivas (μs)**
0	0DD8DD9	-
1	0ECD017	999998
2	0FC1255	999998
3	10B5493	999998
4	11A96D2	999999
5	129D910	999998

Tabela 5.5: Diferença entre o *timestamp* de 6 tramas consecutivas.

5.3.3 Tempo de geração de reposta a tramas

Outro aspecto importante a validar em relação à camada MAC como um todo, é o tempo que esta demora a descodificar uma trama, gerar a resposta adequada e transmiti-la. Uma vez que respostas directas da MAC a tramas têm de ser enviadas para o meio com um intervalo de tempo entre tramas próximo de SIFS (definido na norma como 64 μs) é necessário assegurar que as respostas são geradas na MAC rapidamente, pois é necessário neste tempo SIFS ter ainda em conta o atraso introduzido pela camada PHY.

Para esse efeito, enviou-se de um terminal tramas de dados e mediu-se, com recurso a um contador implementado em *hardware*, o tempo entre o momento em que a trama é recebida na totalidade pela LMAC e o momento em que a UMAC ordena o envio da trama ACK de resposta. Entre estes momentos decorrem os seguintes passos:

- O lançamento e atendimento pela UMAC da interrupção de chegada de trama;
- Leitura pela UMAC dos dados recebidos da memória da LMAC;
- Descodificação da trama recebida;
- Geração da trama ACK de resposta;
- Escrita da trama ACK na memória da LMAC;
- Ordem de envio da trama ACK;

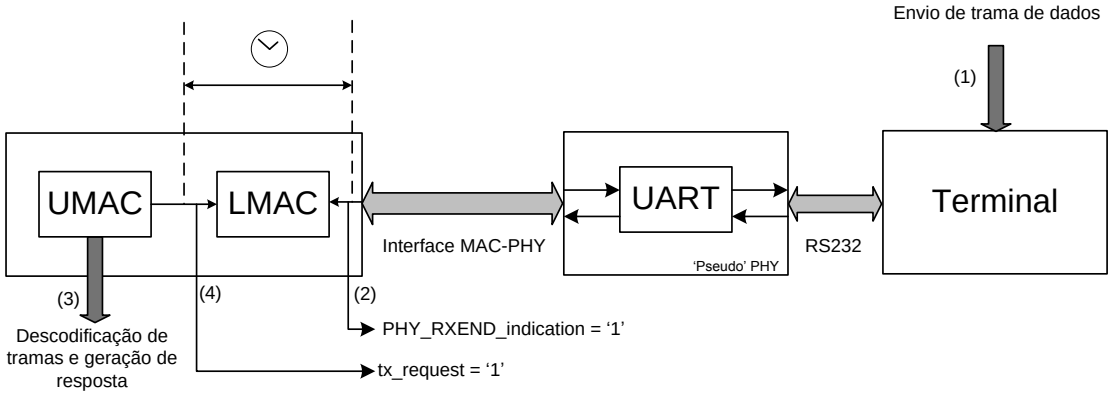


Figura 5.9: Esquematização dos testes realizados para medição do tempo de geração de resposta a tramas.

Um esquema deste teste encontra-se representado na figura 5.9. Em termos de sinais, pretende-se medir o tempo desde que o sinal `PHY_RXEND_indication` é activado até que seja activado o sinal `tx_request`.

Resultados

Ao longo da medição destes tempos foram sendo realizadas optimizações adicionais à UMAC, tanto ao nível do código como ao nível do próprio Microblaze através da introdução de *cache* e de outras funcionalidades que melhoram o seu desempenho em troca da ocupação de uma maior área na FPGA. Os resultados obtidos são apresentados na tabela 5.6. Os tempos apresentados são resultados aproximados.

Optimização	**Ciclos de relógio**	**Tempo (μs)**
Nenhuma	67000	1340
Optimizações no código	62500	1250
Optimizações no microprocessador	22500	450
Adição de *cache* (8KB)	5000-12000	100-240

Tabela 5.6: Tempo de geração de repostas MAC.

Discussão

Os tempos verificados foram excessivamente longos; apesar de ter sido possível reduzir os tempos de resposta uma ordem de grandeza através de optimizações ao nível do código e do microprocessador, o esquema actual não permitirá colocar uma resposta MAC no meio com um intervalo entre tramas igual a SIFS. Torna-se assim necessária a migração da geração de respostas MAC da UMAC para a LMAC.

Capítulo 6

Conclusões e Trabalho Futuro

6.1 Resumo do trabalho realizado

Ao longo deste trabalho foi desenvolvida a base, tanto em *hardware* como em *software*, para a concepção da MAC WAVE. O projecto foi implementado em FPGA e foram validados os mecanismos em *hardware* que oferecerão à UMAC as funcionalidades necessárias para realizar o controlo de acesso ao meio.

O trabalho realizado ao longo deste projecto pode ser resumido pelos seguintes pontos:

- Implementação dos mecanismos de geração e verificação de FCS;
- Optimização de alguns aspectos da LMAC;
- Contribuição ao nível da UMAC;
- Ajustes no módulo PHY.

O trabalho realizado permitirá de futuro prosseguir com o desenvolvimento de uma camada MAC de acordo com os protocolos WAVE que possa ser integrada num sistema para comunicações veiculares, potenciando a propagação deste tipo de tecnologias.

A camada de acesso deste projecto foi desenvolvida da raíz, assim como a plataforma de implementação base, permitindo maior liberdade de projecto quanto à sua arquitectura, interface com a camada PHY e acesso a módulos adicionais (GPS por exemplo). No entanto, devido a esta abordagem, foi necessário realizar varios ajustes ao longo do trabalho na camada MAC, como por exemplo a necessidade de migração de algumas funções da UMAC para a LMAC.

Em relação à UMAC foram implementadas as seguintes funcionalidades:

- Processamento do cabeçalho e endereço MAC para tramas de diversos tipos;
- Realização do *handshake* especificado para certas tramas de dados através de tramas ACK;
- Gestão virtual do meio através de tramas *Request to Send* e *Clear to Send* e manutenção do NAV;
- Detecção de tramas duplicadas e reenvio de tramas cuja transmissão não foi bem sucedida;

A LMAC é capaz de:

- Comunicar com a camada PHY, transmitindo e recebendo tramas;
- Detecção de erros em tramas recebidas e geração de FCS para tramas enviadas;
- Alocação de memória;
- Funções relacionadas com tempo:
  - *Timestamping* de tramas recebidas;
  - Tempo do sistema;
  - Contagens e interrupções.

6.2 Discussão final dos resultados

A MAC desenvolvida respondeu bem aos testes realizados, desde os mais simples, como a transmissão e recepção de dados, até a testes de consistência do controlador de memória. A latência introduzida pela LMAC na transmissão de tramas é também satisfatória: o tempo mais curto entre tramas consecutivas é igual a SIFS (64μs) e o atraso no envio de mensagens adicionado pela LMAC é apenas de aproximadamente 0.1% esse valor. No entanto, o atraso no lançamento de interrupções, no acesso ao barramento PLB e no processamento de tramas recebidas foi determinado ser demasiado grande e as respostas MAC terão de ser migradas para *hardware*.

A LMAC revelou assim um funcionamento correcto, oferecendo as bases necessárias ao desenvolvimento de uma cada de acesso para redes veiculares.

Não foi possível realizar alguns testes, ao nível do controlo de acesso e sobrecarga de mensagens, por ainda não ter sido realizada a integração com a verdadeira camada PHY na plataforma de desenvolvimento

6.3 Trabalho Futuro

Como já foi referido na secção 5.3.3, será necessário migrar a geração de respostas a nível da MAC (tramas CTS em resposta a tramas RTS e tramas ACK em resposta a tramas de dados) para *hardware* de forma a garantir que a camada de acesso desenvolvida responde dentro dos tempos descritos na norma. Como a MAC desenvolvida foi criada de raíz não era completamente claro no início do projecto quais a funcionalidades que teriam de ser implementadas em *hardware* de forma a respeitar os tempos necessários, embora para alguns casos isto fosse relativamente óbvio (computação e verificação do campo FCS, por exemplo). No futuro poderá ser ainda necessário passar outras funcionalidades presentemente implementadas em *software* para *hardware* como forma de respeitar as restrições temporais impostas.

Existe ainda um grande número de funcionalidades descritas na pilha protocolar WAVE que necessitam de ser implementadas, nomeadamente grande parte da norma IEEE 1609.4, sendo a mais significativa delas a comutação entre o canal de serviço e o canal de controlo asm também suporte para outros tipos de tramas possíveis dentro do standard.

Também será necessário realizar a integração com a 'verdadeira' camada PHY, possibilitando a realização do controlo de acesso ao meio com *carrier sensing*. Ao se implementar

a MAC na nova plataforma será também possível e vantajoso utilizar o módulo GPS para sincronização da estação, de forma a se obter um mecanismo de comutação de canal correcto.

Apêndice A

Modelo de Programação da LMAC

A.1 Registos Disponíveis

Na figura A.1 está representado o modelo de progamação da LMAC, composto por 29 registos que permitem à UMAC controlar a LMAC e determinar o seu estado. A escrita e leitura dos registos da LMAC é feita através do barramento PLB.

A.1.1 Registo 0 - STATUS

Através deste registo, a UMAC pode verificar se a LMAC se encontra num dado momento a efectuar a transmissão de dados para a camada PHY. Uma descrição geral deste registo encontra-se na tabela A.1.

Nome	STATUS
Número	0
Tipo	Leitura
Campos	**Descrição**
TXRDY	*Transmission Ready* - Indica à UMAC se a LMAC se encontra a pronta a transmitir dados para a PHY ('1') ou não ('0')

Tabela A.1: Nome, número, tipo e campos do registo STATUS.

A.1.2 Registo 1 - CONTRL

Neste registo a UMAC pode activar o timer da LMAC e interrupções a cada segundo UTC. Uma descrição geral deste registo encontra-se na tabela A.2.

A.1.3 Registo 3 - TXCTRL

O registo 3 é utilizado pela UMAC quando deseja transmitir uma trama à camada PHY. Uma descrição geral deste registo encontra-se na tabela A.3.

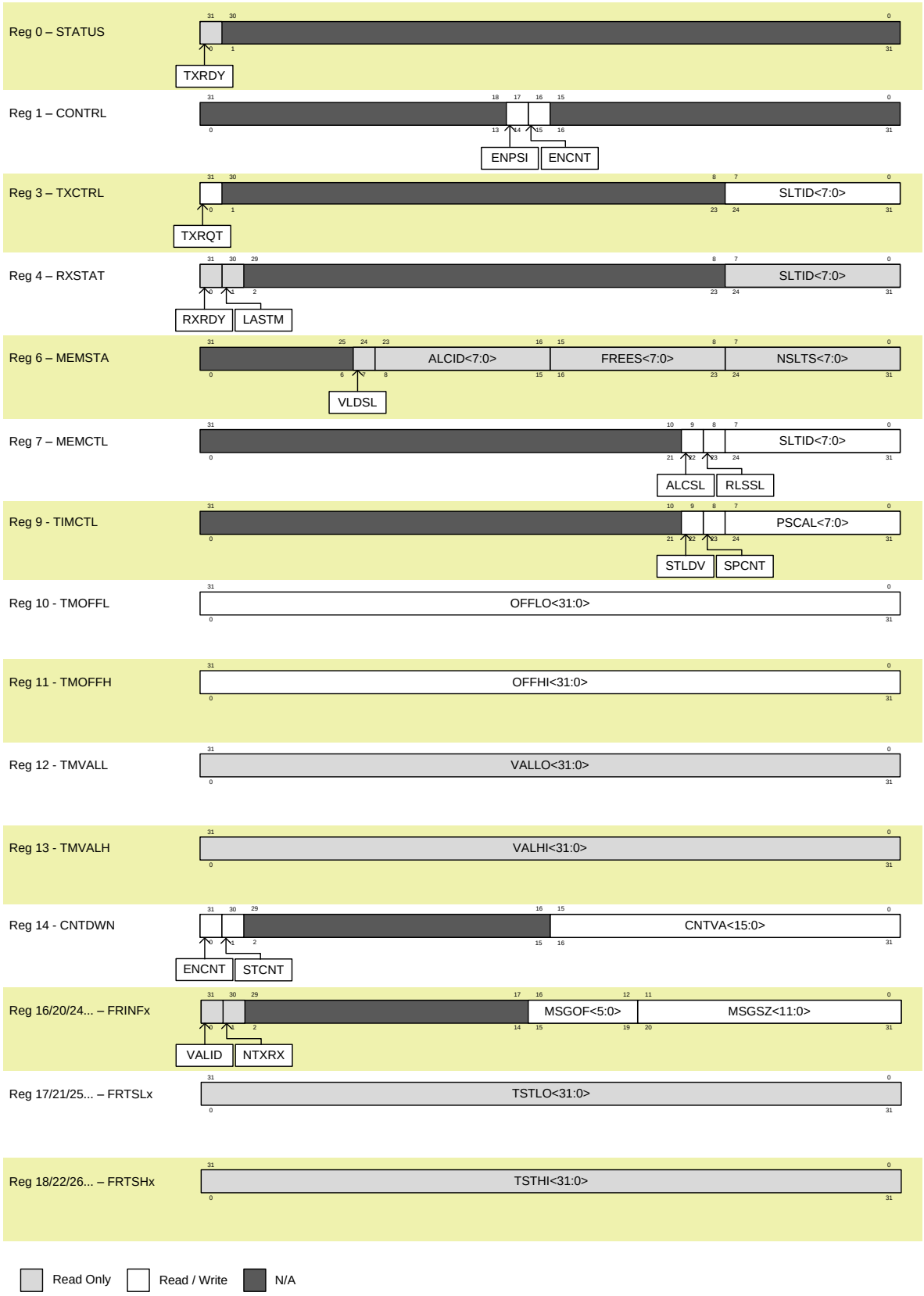


Figura A.1: Modelo de programação da LMAC.

Nome	CONTRL
Número	1
Tipo	Leitura e escrita

Campos	**Descrição**
ENPSI	*Enable Periodic Second Interrupt* - Quando a '1' causa a geração de interrupções periódicas a cada segundo pela LMAC
ENCNT	*Enable Counter* - Quando a '1' activa o contador

Tabela A.2: Nome, número, tipo e campos do registo CONTRL.

Nome	TXCTRL
Número	3
Tipo	Leitura e escrita

Campos	**Descrição**
TXRQT	*Transmission Request* - Quando a '1' ordena à LMAC que os dados contidos na posição de memória indicada no campo *Slot ID* devem ser transmitida para a PHY
SLTID<7:0>	*Slot ID* - Posição de memória a ser transmitida à PHY pela LMAC

Tabela A.3: Nome, número, tipo e campos do registo TXCTRL.

A.1.4 Registo 4 - RXSTAT

Neste registo a UMAC pode verificar se existem tramas recebidas à espera de ser atendidas. Uma descrição geral deste registo encontra-se na tabela A.4.

Nome	RXSTAT
Número	4
Tipo	Leitura

Campos	**Descrição**
RXRDY	*Reception Ready* - Quando a '1' indica que existe uma mensagem recebida da PHY que ainda não foi atendida pela UMAC
LASTM	*Last Message* - Quando a '1' indica que a trama na posição de memória *Slot ID* é a última recebida e não existe mais nenhuma trama à espera de atendimento pela UMAC
SLTID<7:0>	*Slot ID* - Posição de memória da trama recebida

Tabela A.4: Nome, número, tipo e campos do registo RXSTAT.

Este registo é alterado sempre que é efectuada a sua leitura, ou seja, se existir apenas uma trama em fila de espera a flag RXRDY é colocada a '0' depois da UMAC efectuar uma leitura no registo RXSTAT. Também o campo SLTID é actualizado com a próxima mensagem em fila de espera, caso exista uma, após uma leitura deste registo.

A.1.5 Registo 6 - MEMSTA

Este registo contém o estado da memória interna da LMAC, para além da posição de memória alocada à UMAC caso seja pedida uma. Uma descrição geral deste registo encontra-se na tabela A.5.

Nome	MEMSTA
Número	6
Tipo	Leitura
Campos	**Descrição**
VLDSL	*Valid Slot* - Quando a '1' indica que o campo *Allocated Slot ID* é válido
ALCID<7:0>	*Allocated Slot ID* - Caso a alocação de memória na LMAC tenha sido bem sucedida, indica uma posição de memória que foi disponibilizada pela LMAC para utilização pela UMAC
FREES<7:0>	*Free Slots* - Indica o número de posições livres na memória
NSLTS<7:0>	*Number of Slots* - Indica o número total de posições de memória

Tabela A.5: Nome, número, tipo e campos do registo MEMSTA.

A.1.6 Registo 7 - MEMCTL

Neste registo a UMAC tem a possibilidade de efectuar operações de alocação e libertação na memória da LMAC. Uma descrição geral deste registo encontra-se na tabela A.6.

Nome	MEMCTL
Número	7
Tipo	Leitura e escrita
Campos	**Descrição**
ALCSL	*Allocate Slot* - Quando a '1' ordena à LMAC a alocação de uma posição de memória para a UMAC
RLSSL	*Release Slot* - Quando a '1' ordena à LMAC a libertação da posição de memória indicada em *Slot ID*
SLTID<7:0>	*Slot ID* - Indica a posição de memória a ser libertada pela LMAC

Tabela A.6: Nome, número, tipo e campos do registo MEMCTL.

A.1.7 Registo 9 - TIMCTL

No registo 9 a UMAC pode controlar o *timer* interno da LMAC, configurando o seu valor de *pre-scale* e podendo também efectuar o carregamento de um valor no *timer*. Uma descrição geral deste registo encontra-se na tabela A.7.

A.1.8 Registo 10 - TMOFFL

Neste registo podem ser escritos os 32 bits menos significativos do valor a ser carregado no *Timer* da LMAC. Uma descrição geral deste registo encontra-se na tabela A.8.

Nome	TIMCTL
Número	9
Tipo	Leitura e escrita

Campos	**Descrição**
STLDV	*Set Timer Load Value* - Quando a '1' ordena à LMAC o carregamento do valor presente nos registos TMOFFL e TMOFFH no seu *Timer* interno
SPCNT	*Set Prescaler Counter* - Quando a "1" aplica o valor guardado em *Prescaler Value* como divisor da frequência de relógio no *Timer* da LMAC
PSCAL<7:0>	*Prescaler* - Factor de divisão do *clock* interno do sistema. Deve ter um valor que permita incrementar o contador (registos TMVALL e TMVALH) em unidades de 1μs

Tabela A.7: Nome, número, tipo e campos do registo TIMCTL.

Nome	TMOFFL
Número	10
Tipo	Leitura e escrita

Campos	**Descrição**
OFFLO<31:0>	*Timer Offset Low* - 32 bits menos significativos a ser carregados no *Timer* da LMAC

Tabela A.8: Nome, número, tipo e campos do registo TMOFFL.

A.1.9 Registo 11 - TMOFFH

Neste registo podem ser escritos os 32 bits mais significativos do valor a ser carregado no *Timer* da LMAC. Uma descrição geral deste registo encontra-se na tabela A.9.

Nome	TMOFFH
Número	11
Tipo	Leitura e escrita

Campos	**Descrição**
OFFHI<31:0>	*Timer Offset High* - 32 bits mais significativos a ser carregados no *Timer* da LMAC

Tabela A.9: Nome, número, tipo e campos do registo TMOFFH.

A.1.10 Registo 12 - TMVALL

Este registo contém os 32 bits menos significativos do *Timer* da LMAC. Uma descrição geral deste registo encontra-se na tabela A.10.

Nome	TMVALL
Número	11
Tipo	Leitura
Campos	**Descrição**
TMVALL<31:0>	*Timer Value Low* - 32 bits menos significativos do *Timer* da LMAC

Tabela A.10: Nome, número, tipo e campos dos registo TMVALL.

A.1.11 Registo 13 - TMVALH

Este registo contém os 32 bits mais significativos do *Timer* da LMAC. Uma descrição geral deste registo encontra-se na tabela A.11.

Nome	TMVALH
Número	13
Tipo	Leitura
Campos	**Descrição**
TMVALH<31:0>	*Timer Value High* - 32 bits mais significativos do *Timer* da LMAC

Tabela A.11: Nome, número, tipo e campos dos registo TMVALH.

A.1.12 Registo 14 - CNTDWN

Através deste registo a UMAC poderá activar ou parar um contador na LMAC que gere uma interrupção assim que o tempo configurado tenha passado.Uma descrição geral deste registo encontra-se na tabela A.12.

Nome	CNTDWN
Número	13
Tipo	Leitura
Campos	**Descrição**
ENCNT	*Enable Countdown - Flag* de activação do contador
STCNT	*Stop Countdown - Flag* que quando activada pára o contador
CNTVA<15:0>	*Countdown Value* - Valor a carregar no contador

Tabela A.12: Nome, número, tipo e campos dos registo CNTDWN.

A.1.13 Registos 16/20/24/28/32/36 - FRINF0/1/2/3/4/5

A LMAC possui ainda um registo para cada posição de memória, com flags de estado e campos configuráveis pela UMAC. Uma descrição geral deste registo encontra-se na tabela A.13.

Nome	FRINF0/1/2/3/4/5
Número	16/20/24/28/32/36
Tipo	Leitura

Campos	**Descrição**
VALID	*Valid* - Quando a '1' indica que a posição de memória à qual este registo corresponde contém dados válidos
NTXRX	*Transmission or Reception* - Indica se a posição de memória à qual este registo corresponde contém uma trama gerada pela UMAC ('0') ou uma trama recebida da PHY ('1')
MSGOF<5:0>	*Message Offset* - Indica o *offset*, em relação ao endereço inicial da correspondente posição de memória, a partir do qual os dados são válidos
MSGSZ<11:0>	*Message Size* - Indica o tamanho, em bytes, da trama contida na posição de memória a que este registo corresponde

Tabela A.13: Nome, número, tipo e campos dos registos FRINF0/1/2/3/4/5.

A.1.14 Registos 17/21/25/29/33/37 - FRTSL0/1/2/3/4/5

Este grupo de 6 registos contém os 32 bits menos significativos do valor do *timestamp* de tramas recebidas para cada posição de memória. Uma descrição geral deste registo encontra-se na tabela A.14.

Nome	FRTSL0/1/2/3/4/5
Número	17/21/25/29/33/37
Tipo	Leitura

Campos	**Descrição**
TSTLO<31:0>	*Timestamp Low* - 32 bits menos significativos do valor do *timestamp* de recepção para a posição de memória à qual o registo corresponde

Tabela A.14: Nome, número, tipo e campos dos registos FRTSL0/1/2/3/4/5.

A.1.15 Registos 18/22/26/30/34/38 - FRTSH0/1/2/3/4/5

Este grupo de 6 registos contém os 32 bits mais significativos do valor do *timestamp* de tramas recebidas para cada posição de memória. Uma descrição geral deste registo encontra-se na tabela A.15.

A.2 Procedimentos a seguir pela UMAC

A.2.1 Recepção de uma trama

Ao receber uma nova trama, a LMAC irá activar a interrupção `phy_tx_interrupt` sinalizando a UMAC de que se encontram dados prontos a ser lidos. Ao receber a interrupção indicada acima a UMAC terá de seguir o seguinte procedimento para ler a trama recentemente recebida da memória:

Nome	FRTSH0/1/2/3/4/5
Número	18/22/26/30/34/38
Tipo	Leitura

Campos	**Descrição**
TSTLO<31:0>	*Timestamp Low* - 32 bits mais significativos do valor do *timestamp* de recepção para a posição de memória à qual o registo corresponde

Tabela A.15: Nome, número, tipo e campos dos registos FRTSH0/1/2/3/4/5.

---

1. No registo 4 - RXSTAT, verificar se a *flag* RXRDY se encontra a '1', o que significa que existe pelo menos uma trama recebida que a UMAC ainda não atendeu. De seguida deve consultar o campo SLTID que informa qual o ID do *slot* de memória que contém a trama recebida.

2. Ainda no mesmo registo, caso a flag LSM se encontre a '1', então a UMAC sabe que não existem mais tramas à espera de ser atendidas. Caso contrário, a UMAC deve realizar o processamento da primeira trama recebida e voltar a consultar o registo RXSTAT (que é automaticamente actualizado pela LMAC após esta detectar que a UMAC efectuou uma leitura neste registo) de forma a saber qual a trama seguinte. Isto repetir-se-á até que LSM se encontre a '1'.

3. No Registo 16/20/24/28/32/36 - FRINFx de informação associados à trama, a UMAC verifica se essa posição contém na realidade uma trama válida através do campo VALID e continua com o processamento desta.

A.2.2 Requisição de posição de memória pela UMAC

Para que a UMAC possa escrever numa posição da memória interna da LMAC necessita primeiro de pedir a alocação de um *slot*. Para que se lhe seja alocada uma posição de memória a UMAC deve seguir os seguintes passos:

1. Escrever no Registo 7 - MEMCTL colocando a flag ALCSL a '1';

2. No Registo 6 - MEMSTA, caso a LMAC consiga alocar um slot para a UMAC utilizar, a *flag* VLDSL é colocada a '1' e o campo ALCID¡7:0¿ é preenchido com o ID do *slot* alocado. Caso não existam slots disponíveis, a flag VLDSL permanece a '0' e a UMAC tem de tomar medidas adequadas a esta situação, eventualmente a libertação de uma posição de memória.

A.2.3 Transmissão de uma trama para a PHY

Quando a UMAC pretende enviar uma trama para a camada PHY tem de efectuar as seguintes operações:

1. No Registo 0 - STATUS verificar se a flag TXRDY está a '1', indicando que a LMAC está em condições de transmitir a trama. Caso não esteja, a UMAC terá de esperar até que a LMAC esteja pronta a efectuar uma transmissão. Assim que a flag TXR se encontrar a "1", pode passar para o passo seguinte.

2. No Registo 3 - TXCTRL preencher o campo SLTID com o ID do *slot* de memória que contém a trama que deseja transmitir. Finalmente, fazer o *set* da flag TXRQT que sinaliza à LMAC que pretende enviar a trama. A LMAC tratará da transmissão da trama para a PHY a partir deste momento.

A.2.4 Libertação de uma posição de memória pela UMAC

A libertação de posições de memória da LMAC apenas pode ser efectuada pela UMAC. Para isso basta à UMAC:

1. Escrever no Registo 7 - MEMCTL, preenchendo o campo SLTID com o ID da posição de memória que pretende libertar. Em seguida deve fazer *set* da flag RSL.

A.2.5 Manutenção do *Timer*

Carregamento inicial e activação do *Timer*

Para carregar um valor no *Timer* e efectuar a sua activação a UMAC deve:

1. Escrever no Registo 10 - TMOFFL e Registo 11 - TMOFFH os 32 bits menos significativos e os 32 bits mais significativos respectivamente, do valor do offset que pretende carregar no timer;
2. No Registo 9 – TIMCTL colocal a *flag* STL a '1', ordenando à LMAC o carregamento do conteúdo dos registos 10 - TMOFFL e 11 - TMOFFH nos registos 12 - TMVALL e 13 - TMVALH (o contador);
3. Finalmente, no Registo 1 – CONTRL fazer *set* da flag ENCNT de forma a activar o contador.

Activação de interrupções periódicas

Para activar uma interrupção periódica activada pela LMAC a cada segundo basta à UMAC:

1. Colocar a '1' a *flag* ENPSI no Registo 1 - CONTRL.

Alteração da cadência do Timer

A UMAC pode também alterar o valor do pré-divisor do *Timer* da LMAC. Para isso tem apenas de:

1. Escrever no campo PSCAL do Registo 9 - TIMCTL o novo valor do pré-divisor a ser carregado;
2. Colocar a '1' a *flag* STLDV no mesmo registo.

A.3 Drivers

Apesar de a UMAC poder aceder aos registos e à memória directamente, foi criado um conjunto de macros que simplifica algumas das operações previamente descritas. Nesta secção são descritas as macros implementadas para este efeito.

LMAC_TX_FRAME(BaseAddress, SlotID)

Esta macro tem como função ordenar à LMAC o envio da trama presente na posição de memória indicada pelo parâmetro SlotID.

- Parâmetros de entrada:
  - **BaseAddress** - Endereço base dos registos da LMAC;
  - **SlotID** - Número da posição de memória a transmitir à PHY.
- Retorna:
  - **N/A**

LMAC_GET_TX_STATUS(BaseAddress)

Esta macro tem como função verificar o estado de transmissão da LMAC.

- Parâmetros de entrada:
  - **BaseAddress** - Endereço base dos registos da LMAC;
- Retorna:
  - **1** se a LMAC não se encontra a realizar uma transmissão, **0** caso contrário;

LMAC_GET_MEM_STATUS(BaseAddress)

Esta macro tem como função obter informações relativas ao estado da memória.

- Parâmetros de entrada:
  - **BaseAddress** - Endereço base dos registos da LMAC;
- Retorna:
  - Conteúdo do registo MEMSTA;

LMAC_REQ_SLOT(BaseAddress, SlotID, isValid)

Esta macro tem como função pedir à LMAC a alocação de uma posição de memória.

- Parâmetros de entrada:
  - **BaseAddress** - Endereço base dos registos da LMAC;
- Retorna:
  - **SlotID** - Posição de memória alocada pela LMAC para escrita pela UMAC, caso a alocação tenha sido bem sucedida;
  - **isValid** - **1** se alocação de memória tiver sido bem sucedida, **0** caso contrário.

LMAC_CLEAR_SLOT(BaseAddress, SlotID)

Esta macro tem como função ordenar à LMAC a libertação de uma posição de memória.

- Parâmetros de entrada:
  - **BaseAddress** - Endereço base dos registos da LMAC;
  - **SlotID** - Número da posição de memória a libertar.
- Retorna:
  - **N/A**

LMAC_GET_FIRST_RX_FRAME(BaseAddress, SlotID, isValid, isLast)

Esta macro tem como função determinar qual a primeira trama na fila de espera de tramas recebidas da LMAC.

- Parâmetros de entrada:
  - **BaseAddress** - Endereço base dos registos da LMAC;
- Retorna:
  - **SlotID** - Caso existam tramas em fila de espera, é retornado neste parâmetro a posição de memória na saída da fila de recepção da LMAC;
  - **isValid** - **1** caso a trama na posição **SlotID** contenha dados válidos, **0** caso contrário;
  - **isLast** - **1** caso a posição de memória lida da fila de recepção represente a última trama em fila de espera, **0** caso existam tramas adicionais à espera de ser atendidas.

LMAC_SET_TIMER(BaseAddress, flags, prescaler)

Esta macro tem como função ordenar configurar os *timers* da LMAC.

- Parâmetros de entrada:
  - **BaseAddress** - Endereço base dos registos da LMAC;
  - **flags** - *Flags* indicando a operação a realizar nos *timers* (configuração do valor de *prescale*, activação de interrupções periódicas e/ou carregamento de um valor no *timer*).
  - **prescaler** - Factor de divisão do *timer*, configurado quando a flag para configuração do valor de *prescale* estiver activa;
- Retorna:
  - **N/A**

LMAC_LOAD_TIMER_COUNTER_HI(BaseAddress, tmrCounter)

Esta macro tem como função ordenar carregar no registo OFFHI os 32 bits mais significativos de um valor a carregar no *timer*.

- Parâmetros de entrada:
  - **BaseAddress** - Endereço base dos registos da LMAC;
  - **tmrCounter** - Valor a carregar no registo OFFHI.
- Retorna:
  - **N/A**

LMAC_LOAD_TIMER_COUNTER_HI(BaseAddress, tmrCounter)

Esta macro tem como função ordenar carregar no registo OFFLO os 32 bits menos significativos de um valor a carregar no *timer*.

- Parâmetros de entrada:
  - **BaseAddress** - Endereço base dos registos da LMAC;
  - **tmrCounter** - Valor a carregar no registo OFFLO.
- Retorna:
  - **N/A**

LMAC_GET_TIMER_VAL_HI(BaseAddress)

Esta macro tem como função ordenar ler os 32 bits mais significativos do *timer* da LMAC.

- Parâmetros de entrada:
  - **BaseAddress** - Endereço base dos registos da LMAC;
- Retorna:
  - 32 bits mais significativos do *timer* da LMAC.

LMAC_GET_TIMER_VAL_LO(BaseAddress)

Esta macro tem como função ordenar ler os 32 bits menos significativos do *timer* da LMAC.

- Parâmetros de entrada:
  - **BaseAddress** - Endereço base dos registos da LMAC;
- Retorna:
  - 32 bits menos significativos do *timer* da LMAC.

LMAC_GET_FRAME_INFO(BaseAddress, SlotID)

Esta macro tem como função ordenar ler informação sobre uma posição de memória.

- Parâmetros de entrada:
  - **BaseAddress** - Endereço base dos registos da LMAC;
  - **SlotID** - Número da posição de memória
- Retorna:
  - Conteúdo do registo FRINFn correspondente.

LMAC_SET_COUNTDOWN(BaseAddress, CountdownVal)

Esta macro tem como função activar o contador decrescente da LMAC para que este gere uma interrupção.

- Parâmetros de entrada:
  - **BaseAddress** - Endereço base dos registos da LMAC;
  - **CountdownVal** - Valor, em microssegundos, a carregar no contador;
- Retorna:
  - **N/A**.

LMAC_STOP_COUNTDOWN(BaseAddress, CountdownVal)

Esta macro tem como função parar o contador decrescente da LMAC.

- Parâmetros de entrada:
  - **BaseAddress** - Endereço base dos registos da LMAC;
- Retorna:
  - **CountdownVal** - O valor, de 16 bits, do contador no momento de paragem.

LMAC_SET_FRAME_INFO(BaseAddress, SlotID, Data)

Esta macro tem como função ordenar carregar informação sobre uma posição de memória.

- Parâmetros de entrada:
  - **BaseAddress** - Endereço base dos registos da LMAC;
  - **SlotID** - Número da posição de memória;
  - **Data** - Dados a carregar no registo FRINFn correspondente.
- Retorna:
  - 32 bits mais significativos do *timer* da LMAC.

LMAC_GET_FRAME_TS_HI(BaseAddress, SlotID)

Esta macro tem como função ordenar ler os 32 bits mais significativos *timestamp* de uma trama em memória.

- Parâmetros de entrada:
  - **BaseAddress** - Endereço base dos registos da LMAC;
  - **SlotID** - Número da posição de memória da qual se pretender ler o *timestamp*;
- Retorna:
  - 32 bits mais significativos do *timestamp* associado à trama.

LMAC_GET_FRAME_TS_LO(BaseAddress, SlotID)

Esta macro tem como função ordenar ler os 32 bits menos significativos *timestamp* de uma trama em memória.

- Parâmetros de entrada:
  - **BaseAddress** - Endereço base dos registos da LMAC;
  - **SlotID** - Número da posição de memória da qual se pretender ler o *timestamp*;
- Retorna:
  - 32 bits menos significativos do *timestamp* associado à trama.

LMAC_WORD_WRITE_FRAME(MemBaseAddress, SlotID, DataWord, WordOffset)

Esta macro tem como função ordenar a escrita de uma palavra (4 *bytes*) na memória da LMAC.

- Parâmetros de entrada:
  - **MemBaseAddress** - Endereço base da memória da LMAC;
  - **SlotID** - Número da posição de memória onde se pretende escrever;
  - **DataWord** - Palavra a ser escrita em memória;
  - **WordOffset** - *Offset* em relação ao início da posição de memória onde se pretende escrever a palavra;
- Retorna:
  - **N/A**

LMAC_BYTE_WRITE_FRAME(MemBaseAddress, SlotID, DataByte, ByteOffset)

Esta macro tem como função ordenar a escrita de um *byte* na memória da LMAC.

- Parâmetros de entrada:
  - **MemBaseAddress** - Endereço base da memória da LMAC;
  - **SlotID** - Número da posição de memória onde se pretende escrever;
  - **DataByte** - *Byte* a ser escrita em memória;
  - **WordOffset** - *Offset* em relação ao início da posição de memória onde se pretende escrever o *byte*;
- Retorna:
  - **N/A**

LMAC_WORD_READ_FRAME(MemBaseAddress, SlotID,WordOffset)

Esta macro tem como função ordenar a leitura de uma palavra (4 *bytes*) na memória da LMAC.

- Parâmetros de entrada:
  - **MemBaseAddress** - Endereço base da memória da LMAC;
  - **SlotID** - Número da posição de memória de onde se pretende ler;
  - **WordOffset** - *Offset* em relação ao início da posição de memória de onde se pretende ler a palavra;
- Retorna:
  - Palavra lida da posição de memória indicada.

LMAC_BYTE_READ_FRAME(MemBaseAddress, SlotID, ByteOffset)

Esta macro tem como função ordenar a escrita de uma palavra (4 *bytes*) na memória da LMAC.

- Parâmetros de entrada:
  - **MemBaseAddress** - Endereço base da memória da LMAC;
  - **SlotID** - Número da posição de memória onde se pretende escrever;
  - **ByteOffset** - *Offset* em relação ao início da posição de memória de onde se pretende ler o *byte*;
- Retorna:
  - *Byte* lido da posição de memória indicada.

Bibliografia

[1] Defence Codex. Advances in inter-vehicle communication systems. `http://www.science.mod.uk/codex/issue5/journals/journals4.aspx`, Dezembro 2009.

[2] BCA MCA India Students Community. Osi reference model. `http://bcamca.in/osi-reference-model-layers-architecture-213`, Abril 2011.

[3] IEEE Vehicular Technology Society. IEEE draft standard for wireless access in vehicular environments (WAVE) — architecture. 2009.

[4] IEEE Vehicular Technology Society. IEEE standard for wireless access in vehicular environments (WAVE) — multi-channel operation. 2010.

[5] S. Eichler. Performance evaluation of the IEEE 802.11p WAVE communication standard. In *2007 IEEE 66th Vehicular Technology Conference, 2007 (VTC-2007 Fall)*, pages 2109 –2113, 2007.

[6] Sebastian Grafling, Petri Mahonen, and Janne Riihijarvi. Performance evaluation of IEEE 1609 WAVE and IEEE 802.11p for vehicular communications. In *2010 Second International Conference on Ubiquitous and Future Networks (ICUFN)*, pages 344 –348, 2010.

[7] Ting Zhou, H. Sharif, M. Hempel, P. Mahasukhon, Wei Wang, and Tao Ma. A novel adaptive distributed cooperative relaying MAC protocol for vehicular networks. *IEEE Journal on Selected Areas in Communications*, 29(1):72 –82, 2011.

[8] Qing Wang, Supeng Leng, Huirong Fu, Yan Zhang, and H. Weerasinghe. An enhanced multi-channel MAC for the IEEE 1609.4 based vehicular ad hoc networks. In *INFOCOM IEEE Conference on Computer Communications Workshops 2010*, pages 1 –2, 2010.

[9] Yunpeng Zang, L. Stibor, B. Walke, H.-J. Reumerman, and A. Barroso. Towards broadband vehicular ad-hoc networks - the vehicular mesh network (VMESH) MAC protocol. In *IEEE Wireless Communications and Networking Conference, 2007 (WCNC 2007)*, pages 417 –422, 2007.

[10] M. Amadeo, C. Campolo, A. Molinaro, and G. Ruggeri. A WAVE-compliant MAC protocol to support vehicle-to-infrastructure non-safety applications. In *IEEE International Conference on Communications Workshops, 2009 (ICC Workshops 2009)*, pages 1 –6, 2009.

[11] Ning Lu, Yusheng Ji, Fuqiang Liu, and Xinhong Wang. A dedicated multi-channel MAC protocol design for VANET with adaptive broadcasting. In *IEEE Wireless Communications and Networking Conference, 2010 (WCNC 2010)*, pages 1 –6, 2010.

[12] IEEE Computer Society. IEEE standard for information technology — telecommunications and information exchange between systems — local and metropolitan area networks — specific requirements part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications amendment 6: Wireless access in vehicular environments. 2010.

[13] IEEE Computer Society. IEEE standard for information technology — telecommunications and information exchange between systems — local and metropolitan area networks — specific requirements part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. 2007.

[14] Qi Chen, D. Jiang, and L. Delgrossi. IEEE 1609.4 DSRC multi-channel operations and its implications on vehicle safety communications. In *2009 IEEE Vehicular Networking Conference (VNC)*, pages 1 –8, 2009.

[15] Z. Wang and M. Hassan. How much of DSRC is available for non-safety use? In *5th ACM Internation Workshop on VANET 2008*, 2008.

[16] Jin Zhang, Qian Zhang, and Weijia Jia. VC-MAC: A cooperative MAC protocol in vehicular networks. *IEEE Transactions on Vehicular Technology*, 58(3):1561 –1571, 2009.

[17] Hang Su and Xi Zhang. Network-coding-based relay MAC protocols for drive-thru internet services in vehicular networks. In *2010 IEEE Global Telecommunications Conference (GLOBECOM 2010)*, pages 1 –5, 2010.

[18] T.K. Mak, K.P. Laberteaux, R. Sengupta, and M. Ergen. Multichannel medium access control for dedicated short-range communications. *IEEE Transactions on Vehicular Technology*, 58(1):349 –366, 2009.

[19] N. Chandra Rathore, R.S. Tomar, S. Verma, and G.S. Tomar. CMAC: A cluster based MAC protocol for VANETs. In *2010 International Conference on Computer Information Systems and Industrial Management Applications (CISIM)*, pages 563 –568, 2010.

[20] H.A. Cozzetti and R. Scopigno. RR-aloha+: A slotted and distributed MAC protocol for vehicular communications. In *2009 IEEE Vehicular Networking Conference (VNC)*, pages 1 –8, 2009.

[21] On-line crc calculation. `http://www.lammertbies.nl/comm/info/crc-calculation.html`, Junho 2011.

[22] Crc calculation. `http://zorc.breitbandkatze.de/crc.html`, Junho 2011.

[23] Online crc32 checksum calculator. `http://crc32-checksum.waraxe.us/`, Junho 2011.