UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

FERNANDO GOULART DA ROCHA

PROJETO DE DIPLOMAÇÃO

GIGABIT ETHERNET: CONTROLADOR DE ACESSO AO MEIO

Porto Alegre
2011

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA

DEPARTAMENTO DE ENGENHARIA ELÉTRICA

GIGABIT ETHERNET: CONTROLADOR DE ACESSO AO MEIO

Projeto de Diplomação apresentado ao Departamento de Engenharia Elétrica da Universidade Federal do Rio Grande do Sul, como parte dos requisitos para Graduação em Engenharia Elétrica.

ORIENTADOR: Tiaraju Vasconcellos Wagner

Porto Alegre

2011

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

FERNANDO GOULARTDA ROCHA

GIGABIT ETHERNET: CONTROLADOR DE ACESSO AO MEIO

Este projeto foi julgado adequado para fazer jus aos créditos da Disciplina de "Projeto de Diplomação", do Departamento de Engenharia Elétrica e aprovado em sua forma final pelo Orientador e pela Banca Examinadora.

Orientador: _____

Prof. Tiaraju Vasconcellos Wagner, UFRGS

Mestre pela Universidade Federal do Rio Grande Sul - Porto alegre, Brasil

Banca Examinadora:

Prof. MSc. Tiaraju Vasconcellos Wagner, UFRGS
Mestre pela Universidade Federal do Rio Grande do Sul – Porto Alegre, Brasil

Engº Paulo Roberto Dalla Santa, DSPRO
Engenheiro pela Universidade Federal do Rio Grande do Sul – Porto Alegre, Brasil

Prof. Dr. Marcelo Götz, UFRGS
Doutor pela Universidade Federal do Rio Grande do Sul – Porto Alegre, Brasil

Porto Alegre, Julho de 2011

DEDICATÓRIA

Dedico este trabalho a querida Cristiane, pela dedicação e apoio nos momentos mais difíceis.

AGRADECIMENTOS

À empresa DSPRO, pela oportunidade e apoio na formação profissional.

Ao meu orientador Prof. MSc. Tiaraju Vasconcellos Wagner, pela ajuda, estímulo e aprendizado.

À UFRGS, professores e funcionários, por providenciar a estrutura e ajuda necessárias ao estudo.

À minha banca examinadora, Prof. Dr. Marcelo Götz e Engº Paulo Roberto Dalla Santa, por aceitar o convite para participação na avaliação do meu trabalho.

RESUMO

Este documento descreve a análise, o projeto e a implementação de um dos módulos essenciais para uma interface do tipo Gigabit Ethernet: o Controlador de Acesso ao Meio. O módulo, implementado em um FPGA, integra um projeto de distribuição de áudio digital desenvolvido em parceria com a empresa DSPRO. O controlador será capaz de trabalhar nos modos *Gigabit Ethernet* (1000Mbps) e *Fast Ethernet* (100Mbps) e apresentará funcionalidades específicas do projeto. O projeto segue a norma 802.11ab do Instituto de Engenheiros Eletricistas e Eletrônicos (IEEE).

Palavras-chave: Engenharia Elétrica. Gigabit Ethernet. Eletrônica. Comunicação. FPGA

ABSTRACT

This document describes the analysis, the project and the implementation of one of the essential Gigabit Ethernet modules: the Medium Access Controller. This module, implemented on an FPGA, is part of a digital audio distribution project developed in partnership with DSPRO company. The controller will be able to work on *Gigabit Ethernet* (1000Mbps) and *Fast Ethernet* (100Mbps) modes and will present project's specifics features. The project is based upon Institute of Electrical and Electronics Engineers' (IEEE) 802.11ab standard.

Keywords: Electrical Engineering. Gigabit Ethernet. Electronic. Communication. FPGA

SUMÁRIO

1 INTRODUÇÃO 12
2 CONTEXTO DO PROJETO 13
2.1 Controlador de Acesso ao Meio 13
2.2 1000BASE-T Ethernet 13
2.2.1 *Reduced Gigabit Media Independent Interface* 15
2.3 Especificação do Projeto 17
3 DESENVOLVIMENTO 20
3.1 Estrutura do projeto 20
3.2 Desenvolvimento do Hardware 21
3.2.1 Seleção de Componentes 21
3.2.2 Desenvolvimento da PCI 26
3.2.3 Montagem 30
3.3 Desenvolvimento em Lógica Programável 30
3.3.1 Linguagem de Descrição de *Hardware* 30
3.3.2 Módulos Desenvolvidos 32
4 SIMULAÇÕES E IMPLEMENTAÇÃO FINAL 41
***4.1 Testbenchs* 41**
4.2 Simulações do projeto 42
4.3 Implementação 44
4.4 Testes finais 45
5 CONCLUSÃO 48
6 REFERÊNCIAS 50

LISTA DE ILUSTRAÇÕES

Figura 1 - Relação do PHY 1000BASE-T com o modelo de referência Open System Interconnection (OSI) e o modelo IEEE 802.3 CSMA/CD LAN 14
Figura 2 - Diagrama de Tempo dos Sinais da Interface RGMII 16
Figura 3 - Diagrama de blocos da parte referente à interface *Ethernet* do projeto..... 20
Figura 4 - Diagrama simplificado de uma célula lógica..... 25
Figura 5 - Esquema elétrico das terminações da interface Ethernet..... 28
Figura 6 - Relação dos blocos da lógica programável..... 33
Figura 7 - Simulação do ínicio da recepção de um pacote no modo *Gigabit Ethernet*..... 42
Figura 8 - Simulação do final da recepção de um pacote no modo *Gigabit Ethernet*..... 43
Figura 9 - Simulação do início da recepção de um pacote no modo *Fast Ethernet* 43
Figura 10 - Simulação do final da recepção de um pacote no modo *Fast Ethernet*..... 44

LISTA DE TABELAS

Tabela 1 - Definições de Sinais da Interface RGMII 16
Tabela 2 - Utilização final de lógica.....45

LISTA DE ABREVIATURAS

MAC: Controlador de Acesso ao Meio

FCS: Seqüência Verificadora de Pacote

IEEE: Instituto de Engenheiros Eletricistas e Eletrônicos

PHY: Codificador da Camada Física

GMII: *Gigabit Media Independent Interface*

RGMII: *Reduced Gigabit Media Independent Interface*

RTBI: *Reduced Ten Bit Interface*

PCD: *Physical Coding Sublayer*

DDR: *Double Data Rate*

CRC: *Cyclic Redundancy Check*

SFD: *Start Frame Delimiter*

PCI: Placa de Circuito Impresso

FPGA: *Field Programmable Gate Array*

VHDL: *Very High Speed Integrated Circuit Hardware Description Language*

CI: Circuito Integrado

LDH: Linguagem de descrição de *Hardware*.

ECAD: *Electronic Desing Automation*

DCM: *Digital Clock Manager*

PLL: *Phase-Locked Loop*

DLL: *Delay Locked Loop*

RAM: *Random Access Memory*

1 INTRODUÇÃO

O presente trabalho foi desenvolvido em parceria com a empresa Ds Pro Áudio Ltda., aqui referida como DSPRO. Nele, são descritas a arquitetura e as etapas do projeto de um módulo Controlador de Acesso ao Meio de uma interface *Gigabit Ethernet*.

O módulo integra o projeto de um novo produto desenvolvido pela empresa e pode ser considerado parte principal do mesmo.

Por se tratar de um projeto industrial e ainda em desenvolvimento, algumas informações, como *part numbers* de componentes e códigos-fonte, serão omitidas, sem, no entanto, prejudicar de forma alguma o entendimento e o detalhamento do projeto.

Conforme informado, o módulo integra um projeto específico da empresa, sendo sua aplicação restrita ao produto da mesma. Contudo, pode-se facilmente estender a funcionalidade deste removendo restrições de projeto e tornando-o um controlador de acesso ao meio genérico para aplicações que utilizem uma interface *Gigabit Ethernet*. As alterações necessárias para tanto estão descritas neste documento.

Todo o trabalho aqui descrito foi desenvolvido pelo autor, com exceção do leiaute e da montagem descritos nas seções 3.2.2.4 e 3.2.3, respectivamente. O esquema elétrico da seção 3.2.2.3 foi revisado por um engenheiro da empresa e a decisão final sobre a seleção dos componentes da seção 3.2.1 foi feita pelo responsável do projeto. As demais etapas foram realizadas integralmente pelo autor.

2 CONTEXTO DO PROJETO

2.1 Controlador de Acesso ao Meio

O controlador de acesso ao meio – ou MAC, do inglês *medium access controller* – é o dispositivo responsável pela primeira interpretação lógica dos dados contidos em um pacote transmitido por uma rede Ethernet. É o responsável pela identificação da origem e destino do pacote, bem como da integridade do mesmo.

Um MAC pode ser configurado para filtrar pacotes e considerar apenas aqueles que são destinados para ele (*Unicast*), que são destinados para um grupo do qual faz parte (*Multicast*), ou que são destinados para todos integrantes da rede (*Broadcast*).

Assim como é função do MAC verificar a integridade do pacote, este deve realizar o cálculo necessário para determinação da seqüência verificadora de pacote (FCS, do inglês *frame check sequence*) antes de transmitir os dados.

2.2 1000BASE-T Ethernet

O padrão *Gigabit Ethernet* por cabo de par trançado de cobre, ou *Ethernet* 1000BASE-T, foi especificado em 1999 pelo Instituto de Engenheiros Eletricistas e Eletrônicos (IEEE) através da publicação do padrão 802.3ab. Trata-se da definição de uma interface capaz de transferir dados a uma taxa de 1 Gigabit/s (125 Megabytes/s). Foi desenvolvida com a intenção de permitir compatibilidade com padrões especificados anteriormente que também utilizavam cabo de par trançado de cobre, como 10BASE-T (802.3i), 100BASE-TX (802.3u) e 100BASE-T4 (802.3u).

A interface de comunicação sugerida pelo IEEE entre o MAC e o dispositivo responsável por transmitir e receber os dados na camada física (PHY) é a *Gigabit Media Independent Interface* (GMII), definida na publicação 802.3z do instituto. Embora seja uma interface de uso opcional no padrão, pode-se dizer que quase a totalidade das alternativas adotadas deriva dela, ou por conveniência ou por necessidade devido às exigências do padrão 802.3ab direcionadas ao uso da interface GMII. A Figura 1, retirada da publicação 802.3ab do IEEE, demonstra onde é utilizada esta interface e relaciona o PHY e o MAC com as camadas do modelo de referencia OSI para telecomunicações.

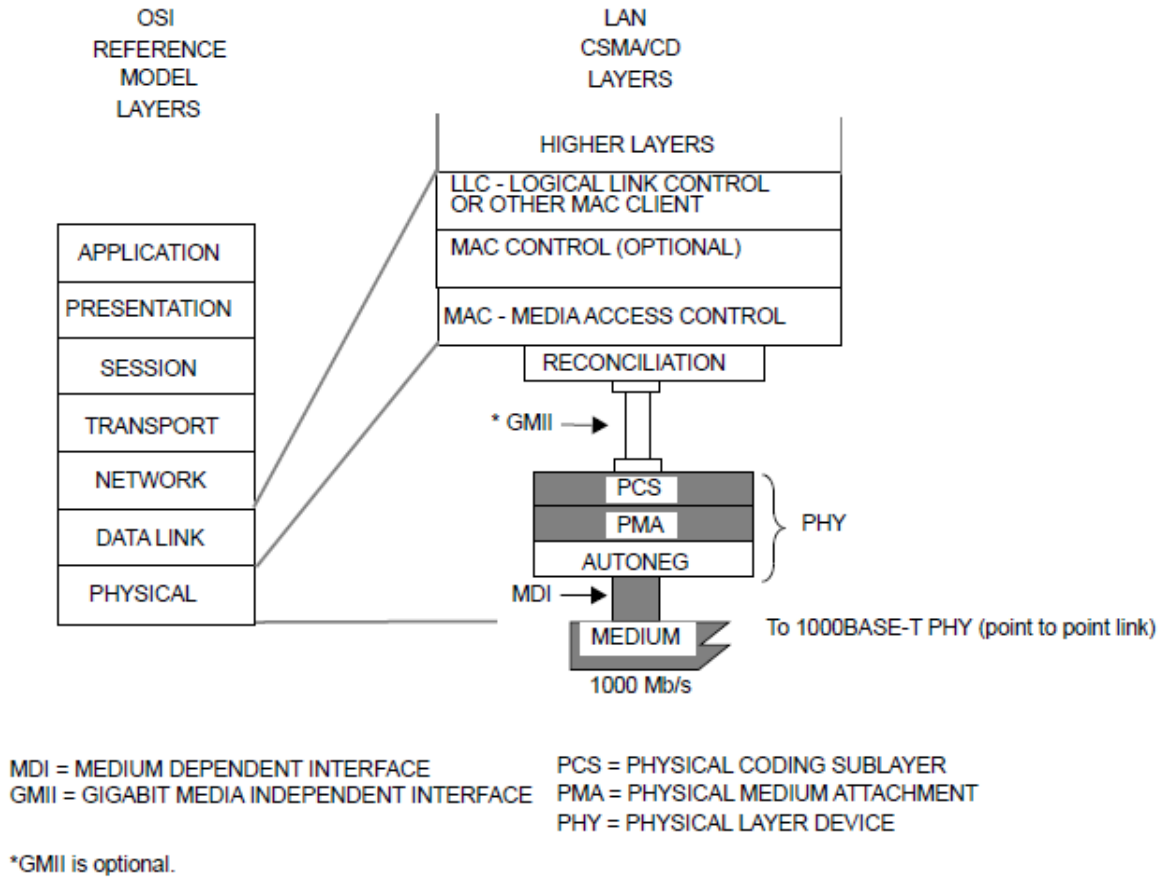

Figura 1 - Relação do PHY 1000BASE-T com o modelo de referência Open System Interconnection (OSI) e o modelo IEEE 802.3 CSMA/CD LAN

2.2.1 *Reduced Gigabit Media Independent Interface*

Desenvolvido em parceria pelas empresas *Broadcom*, *HP* e *Marvell*, o padrão *Reduced Gigabit Media Independent Interface* (RGMII), que teve sua publicação final em 2002, visa reduzir o número de pinos utilizados pela interface GMII de 25 para 12. No mesmo documento é descrita a interface *Reduced Ten Bit Interface* (RTBI), a qual dá acesso direto a Subcamada de Codificação Física – ou PCS, do inglês *Physical Coding Sublayer*. Essa interface, no entanto, será desconsiderada por não interessar ao projeto.

A Tabela 1 apresenta a descrição dos sinais envolvidos na comunicação entre o PHY e o MAC na interface RGMII.

Como pode ser observado na descrição dos sinais TD e RD, se a interface estiver operando no modo Gigabit o dado é transmitido com taxa de transferência dobrada – DDR, do inglês *Double Data Rate*. A grande dificuldade nesta interface está em deslocar o relógio de referência em relação aos dados para garantir que os mesmos estejam estáveis no momento em que forem amostrados. Nas primeiras versões do padrão os dados deveriam ser obrigatoriamente alinhados com o relógio durante a transmissão do sinal, e deveria ser deslocado na recepção através da utilização de trilhas consideravelmente mais longas dos sinais de relógio. Na sua versão final, no entanto, o padrão permite que o relógio seja atrasado na transmissão, facilitando o roteamento do projeto. A Figura 2, retirada da publicação do padrão RGMII (2002), demonstra a relação entre os sinais da interface. Os sinais TXD[8..5], TXD[4], TXD[9], RXD[8..5], RXD[4] e RXD[9], na forma como representados na figura, fazem parte da interface RTBI e devem ser desconsiderados.

TXC	Relógio de referência de transmissão gerado pelo MAC de freqüência 125MHz, 25MHz ou 2.5MHz, dependendo da velocidade de operação.
TD[3..0]	Bits 3:0 do dado de transmissão (TXD) na borda ascendente de TXC e bits 7:4 na borda descente. Se estiver operando no modo 10Mbps ou 100Mbps os bits 3:0 são transmitidos no primeiro ciclo de TXC e os bits 7:4 no ciclo seguinte, podendo ser repetidos na borda descente. É gerado pelo MAC.
TX_CTL	Controle de transmissão. Habilitação da transmissão quando TXC encontra-se no nível lógico ‘1’ e uma combinação lógica da habilitação e do erro de transmissão no nível lógico ‘0’. É gerado pelo MAC.
RXC	Relógio de referência de recepção gerado pelo PHY de freqüência 125MHz, 25MHz ou 2.5MHz, dependendo da velocidade de operação.
RD	Bits 3:0 do dado de recepção (RXD) na borda ascendente de RXC e bits 7:4 na borda descente. Se estiver operando no modo 10Mbps ou 100Mbps os bits 3:0 são transmitidos no primeiro ciclo de RXC e os bits 7:4 no ciclo seguinte, podendo ser repetidos na borda descente. É gerado pelo PHY.
RX_CTL	Controle de recepção. Indica se o dado é válido quando TXC encontra-se no nível lógico ‘1’ e uma combinação lógica da validação do dado e do erro de recepção no nível lógico ‘0’. É gerado pelo PHY.

Tabela 1 - Definições de Sinais da Interface RGMII

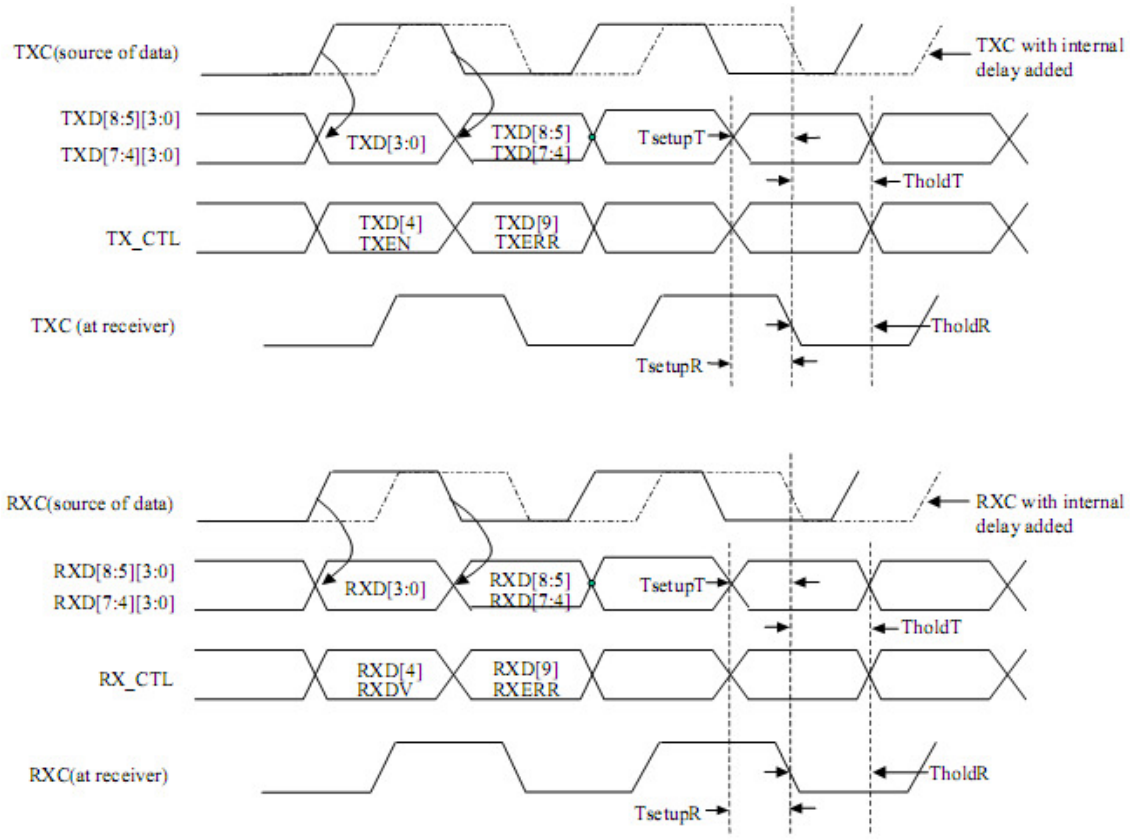

Figura 2 - Diagrama de Tempo dos Sinais da Interface RGMII

2.3 Especificação do Projeto

O MAC será parte integrante do projeto de um sistema de distribuição de áudio digital. Ele deverá ser capaz de operar nos modos 1000BASE-T (*Gigabit Ethernet*) e 100BASE-TX (*Fast Ethernet*), ambos em *full-duplex*, e de utilizar toda a capacidade da banda de transferência de dados de cada um dos modos, respeitadas as condições do padrão.

Na recepção, o MAC deverá verificar a integridade do pacote calculando os quatro bytes da verificação de redundância cíclica (CRC-32) e comparando-os com a seqüência de verificação de pacote. Os pacotes que contenham dados de áudio do sistema deverão ser identificados através do endereço de destino, deverá ser verificado se foram obtidos com a mesma taxa de amostragem com que opera o equipamento do qual o MAC faz parte e se o pacote de fato contém amostras de áudio. As amostras contidas no pacote deverão ser demultiplexadas e encaminhadas para seu canal correspondente. As manipulações dos dados contidos nos pacotes de áudio são específicas do MAC do projeto, não se aplicando a um MAC genérico, com exceção da identificação do pacote pelo endereço de destino.

Deverá haver a possibilidade de habilitar ou desabilitar filtros para os pacotes recebidos. O filtro de pacotes *Unicast*, deverá ter como restrição os endereços MAC das portas Ethernet do equipamento, os quais poderão ser reconfigurados através de registradores. O filtro de pacotes *Broadcast* não distinguirá diferentes endereços *Broadcast*. O MAC não possuíra filtro para pacotes *Multicast*.

O projeto inclui um microcontrolador para o qual deverão ser destinados os pacotes que não contenham dados de áudio. O microcontrolador possui um MAC de uma interface *Fast Ethernet* embarcado que utiliza o protocolo *Reduced Media Independent Interface* (RMII) para comunicação com PHY. O MAC deste projeto em específico deverá emular um

PHY para o microcontrolador e encaminhar os pacotes que não contenham dados de áudio para o mesmo.

Na transmissão, o MAC será responsável por multiplexar as amostras de áudio dentro do seu respectivo pacote para serem transmitidas para outros equipamentos. Os pacotes provenientes do microcontrolador terão seu endereço de origem substituído pelo correspondente ao da porta em que forem transmitidos. Os bytes da FCS deverão ser removidos destes pacotes e recalculados devido à substituição realizada anteriormente.

Cada pacote transmitido será precedido por um preâmbulo constituído por uma seqüência de sete bytes 01010101, ou 0x55 no formato hexadecimal, e de um byte delimitador do começo de pacote – SFD, do inglês *Start Frame Delimiter* – igual a 01110101, 0xD5 no formato hexadecimal. Ao encerrar a transmissão de um pacote haverá um período correspondente a transmissão de 12 bytes em que não será transmitido nenhum dado. Essas são restrições do protocolo Ethernet.

O MAC se comunicará com o PHY através da interface RGMII, descrita brevemente na seção 2.2.1, e deverá limitar automaticamente o número de canais transmitidos quando operar no modo *Fast Ethernet* (100Mbps). O padrão elétrico utilizado nos sinais de comunicação da interface RGMII será o *Stub Series Terminated Logic* (SSTL) de classe II, com terminações de entrada e de saída para cada sinal.

A conexão entre o PHY e o conector RJ-45 será feita através de um transformador isolador e casador de impedâncias para cada um dos quatro pares utilizados no protocolo além de suas respectivas terminações. O roteamento das trilhas da placa de circuito impresso (PCI) deverá ser feito de forma a garantir a impedância necessária para o devido casamento entre as descontinuidades das linhas de transmissão e um atraso desconsiderável entre as linhas de um mesmo sinal.

O MAC será implementado em um vetor de portas programável em campo – FPGA, do inglês *Field Programmable Gate Array* – e será descrito em linguagem de descrição de hardware de circuitos integrados de velocidade muito alta – VHDL, do inglês *Very High Speed Integrated Circuit Hardware Description Language*.

3 DESENVOLVIMENTO

3.1 Estrutura do projeto

O diagrama de blocos abaixo representa a parte relativa à interface Gigabit Ethernet do projeto. O bloco referente à camada física inclui o PHY, circuito integrado (CI) responsável por codificar os dados e determinar as configurações da conexão, e os dispositivos passivos necessários para garantir o casamento de impedâncias, a redução de ruído e a proteção do circuito. O MAC é o controlador de acesso ao meio implementado em um FPGA, responsável por encaminhar dados de gerência do microcontrolador e dados de áudio de outros blocos implementados no FPGA para a camada física e, reciprocamente, dados desta para eles.

Como existem quatro interfaces *Ethernet* no projeto, o diagrama de blocos da Figura 3 pode ser replicado e os dados podem ser multiplexados para os demais blocos e demultiplexados para as interfaces quando forem originados nestes.

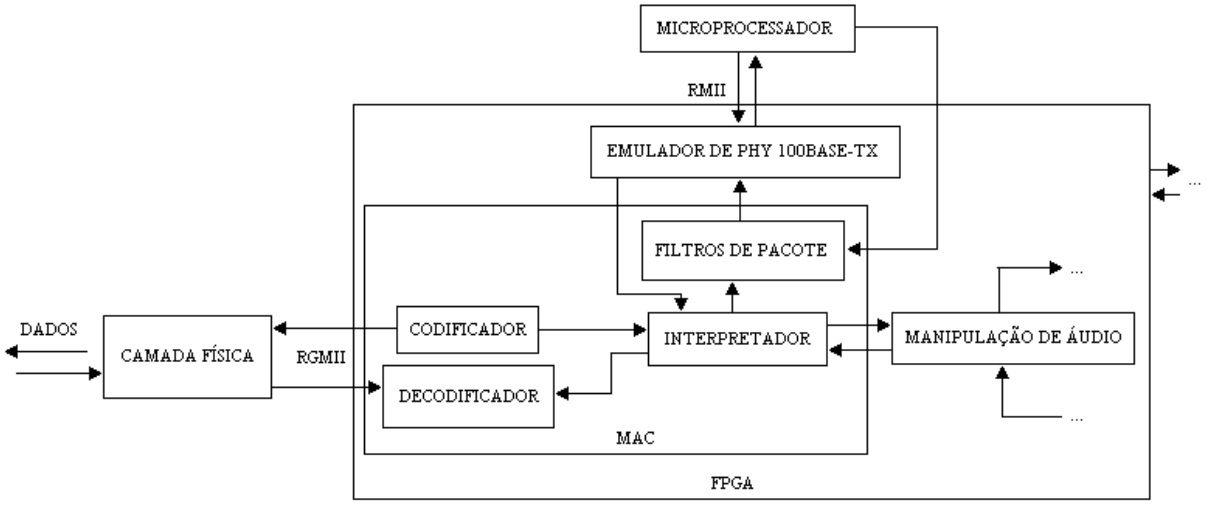

Figura 3 - Diagrama de blocos da parte referente à interface *Ethernet* do projeto

3.2 Desenvolvimento do Hardware

3.2.1 Seleção de Componentes

3.2.1.1 PHY

Uma vez definida a estrutura do projeto, a próxima etapa é a seleção de componentes. Por se tratar de uma interface relativamente popular e de alta freqüência, o uso de circuitos integrados projetados para modular e acionar os sinais deste uso específico tende a ser mais vantajoso financeira e temporalmente. Como mencionado nos capítulos anteriores, tal circuito integrado é normalmente chamado de PHY ou PHYceiver.

Como muitos PHYs possuem diferenciados encapsulamentos, interfaces, padrões elétricos e até tensões de alimentação, a escolha deste componente é fundamental para definir as diretrizes do projeto. Os seguintes elementos são significativos na escolha de um PHY 1000Base-T:

- Interfaces de comunicação com MAC suportadas
- Padrão elétrico das interfaces anteriores
- Existência de detecção e correção nos erros de polaridade dos sinais da interface 1000Base-T
- Capacidade operar em Full-Duplex e/ou Half-Duplex
- Sensibilidade de recepção e ruído de transmissão (Pode suportar cabos acima de 100m)
- Encapsulamento
- Suporte a outros padrões de comunicação (10BASE-T, 100BASE-TX, 1000BASE-LX, etc...)

- Compatibilidade de pinos com outros PHYs (Inclusive de diferentes padrões)
- Tensões de alimentação necessárias
    - Existência de regulador interno
- Custo
- Relação da empresa com os distribuidores
- Funções adicionais (Interrupções, Controle de LEDs, Comprimento de cabo, etc...)

Para este projeto optou-se pela interface RGMII, como já mencionado, devido à redução de pinos utilizados sem acréscimos nas freqüências de operação, eliminando a necessidade de utilização de componentes harmônicos com freqüências muito altas nos sinais de comunicação. Como a maioria das famílias de FPGA disponíveis no mercado suporta uma grande variedade de padrões elétricos de comunicação, não foram impostas muitas restrições nesse sentido ao PHY, no entanto, deu-se preferência a padrões que utilizem como referência a tensão de 3.3V, utilizada como alimentação de muitos dos atuais CIs no mercado.

A detecção e correção de erros de polaridade foi considerada opcional, por não alterar significativamente a facilidade de roteamento dos sinais na PCI e por erros desse gênero não serem comuns na fabricação de cabos.

Para o encapsulamento, optou-se por aqueles que possuíssem uma boa área para dissipação térmica, não ocupassem uma área significativa na placa, possibilitassem a compatibilidade de pinos com algum outro PHY que utilizasse o padrão 100BASE-TX e que, preferencialmente, possuíssem montagem simplificada, excluindo, dessa forma, os encapsulamentos BGA, PGA, etc.

Por se tratar de um produto que deve ser compatível com outros fabricados anteriormente, o modo 100BASE-TX deve ser uma funcionalidade obrigatória. Como a

especificação do projeto indicava a obrigatoriedade de suporte para que os dados utilizassem a integralidade da banda disponível tanto para transmissão como para recepção, o PHY obrigatoriamente deve suportar o modo Full-Duplex tanto quando estiver operando no modo Fast Ethernet quanto no modo Gigabit Ethernet.

Observando-se o custo, uma boa sensibilidade de recepção do CI significa uma taxa de erros reduzida. Esta característica não costuma ser explícita pelos fabricantes, mas pode ser observada quando os mesmos garantem a operação do PHY com cabos de comprimento superior ao especificado no padrão (100m).

Por fim, optou-se por um PHY que atende todos os requisitos mencionados e utiliza o SSTL Classe II com referência em 3.3V como padrão elétrico de comunicação.

3.2.1.2 Transformador

Embora não seja obrigatório, a maioria das interfaces Ethernet utiliza transformadores para isolar eletricamente o circuito, eliminar laços de terra e reduzir o ruído de modo comum.

Cada transformador possui sua resposta em freqüência devido às características de fabricação. Tanto no modo 100BASE-TX quanto no modo 1000BASE-T a freqüência fundamental é 125MHz, sendo que o segundo utiliza modulação em amplitude com cinco níveis contra três níveis do primeiro. Considerando-se importante para a integridade do sinal uma resposta em freqüência desde a fundamental até a quinta harmônica, necessitamos de um transformador com uma banda mínima de 125MHz até 625MHz.

Para o padrão 1000BASE-T são utilizados quatro pares de cabos para transmitir os dados, sendo necessários quatro transformadores. No mercado, existem transformadores específicos para uso com interfaces Ethernet. Trata-se na realidade de quatro transformadores encapsulados no mesmo componente. A alta proximidade destes elementos magnéticos traz

como problema a interferência, conhecida como *crosstalk*, dos sinais de um transformador em outro. A máxima taxa de *crosstalk* que pode ser admitida depende dos algoritmos de rejeição e do leiaute de cada PHY. Para o PHY escolhido neste projeto podemos aceitar uma taxa de *crosstalk* de até -25dB.

Atendendo esses requisitos, foi escolhido o transformador que apresentava melhor custo e disponibilidade para pronta-entrega.

3.2.1.3 Demais elementos passivos

Os resistores que compõem a interface não necessitam ser fabricados para suportar alta dissipação de energia, sendo sua única restrição a tolerância de 1% para o devido casamento de impedâncias.

Os capacitores presentes na interface têm a função de proteger o circuito, impedindo correntes elevadas de baixa freqüência. Os demais são utilizados para prevenir oscilações nas tensões de alimentação, seja por variação da carga seja por interferência eletromagnética. Por esse motivo não existem grandes restrições quanto as suas tolerâncias, sendo necessária apenas que a tensão de ruptura do capacitor que conecta o pino central do transformador à blindagem externa não seja inferior a 2kV, para proteção de descargas eletrostáticas.

3.2.1.4 FPGA

Um FPGA é um circuito integrado reconfigurável capaz de representar diversos circuitos lógicos, sejam eles seqüenciais ou combinacionais. Cada FPGA é, tipicamente, constituído de centenas a centenas de milhares de células lógicas. A composição exata de cada célula lógica varia entre os fabricantes, porém, essencialmente, pode-se dizer que todas

elas possuem uma *Lookup Table*, um *Flip-Flop* e um multiplexador, conforme a Figura 4. Uma *Lookup Table* é a implementação de uma tabela verdade com *n* entradas e uma saída. O número de entradas de cada *Lookup Table* pode ser diferente para cada modelo de FPGA.

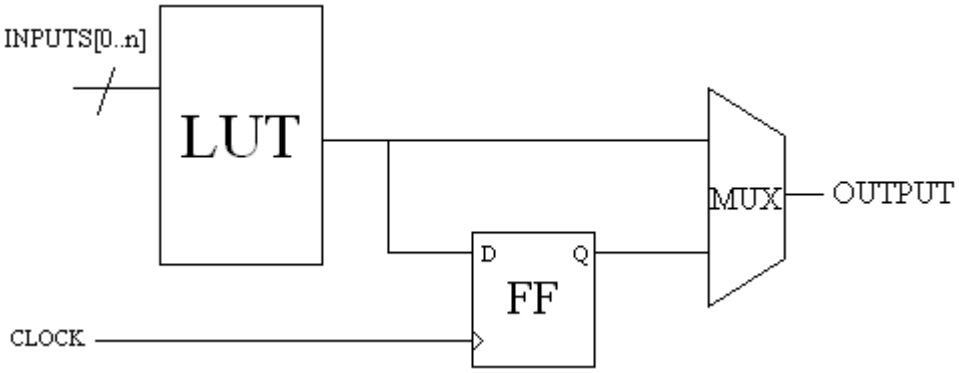

Figura 4 - Diagrama simplificado de uma célula lógica

Além das células lógicas, essenciais para cada FPGA, é muito comum encontrar no mercado circuitos integrados que possuem outros dispositivos embarcados como recursos adicionais as células lógicas. Tais recursos costumam ajudar o desenvolvimento de diversos projetos que não seriam possíveis se utilizassem somente células lógicas, seja por motivos operacionais, otimização da utilização da lógica ou restrições temporais das partes configuráveis do CI. Destacam-se entre tais recursos:

- Malha de captura de fase – PLL, do inglês *Phase-Locked Loop*.
- Malha de captura de atraso – DLL, do inglês *Delay-Locked Loop*.

Gerenciador Digital de Relógio – DCM, do inglês *Digital Clock Manager*.

- Multiplicador.
- Memória de Acesso Aleatório – RAM, do inglês *Random Access Memory.*
- *Flip-Flop* DDR de entrada e saída.
- Módulo para processamentos de sinais.

O custo de um FPGA está diretamente relacionado com a quantidade de células lógicas e recursos disponíveis no mesmo.

Para este projeto foi estimada a quantidade de células lógicas necessárias para implementar todos os módulos digitais do produto, não apenas os Controladores de Acesso ao Meio das quatro interfaces.

A estimativa inicial, tendo como base a utilização de lógica de produtos anteriores, foi de aproximadamente dez mil células lógicas e trinta memórias do tipo RAM de pelo menos 8kbits. É obrigatória a presença de *Flip-Flops* nos pinos de entrada e saída e alguma forma de multiplicar e dividir os relógios de referência (DCMs, PLLs ou DLLs).

Dessa forma, optou-se por um FPGA de custo relativamente baixo que respeita todas as condições mencionadas.

3.2.2 Desenvolvimento da PCI

3.2.2.1 *Software* de Automação de Projetos Eletrônicos

Para desenvolver a PCI, utilizou-se um *software* de automação de projetos eletrônicos – ECAD, do inglês Electronic Design Automation.

A utilização de *softwares* ECAD possibilita o desenvolvimento de esquemas elétricos, o leiaute de placas de circuito impresso de forma integrada ao esquema, além de possibilitar a tradução do esquema para linguagens de descrição de *hardware* e de *software* embarcado. O *software* apresenta diversas ferramentas que auxiliam no desenvolvimento do esquema elétrico, como verificação de erros de conectividade, baseado no sentido das portas dos componentes, e indicação de ligações realizadas no esquema que não foram introduzidas no leiaute.

3.2.2.2 Criação de Símbolos e *Footprints* para os Componentes

Após a seleção de todos os componentes é necessária a tradução dos mesmos para utilização no *software* ECAD. Cada componente necessita de pelo menos dois símbolos associados a ele, um para utilização no esquema elétrico e outro para utilização no leiaute da placa. Este último é chamado de *footprint* por representar o posicionamento dos componentes na placa de forma semelhante a uma "marca de pegada". O *software* utilizado possibilita a criação destes símbolos de forma gráfica e simples.

A criação do símbolo para o esquema elétrico pode ser feita da forma que o projetista considerar mais conveniente, sendo obrigatório apenas incluir no símbolo todos os pinos do componente e, para correta verificação do esquema, o tipo de conexão de cada pino (entrada, saída, alimentação, entrada e saída ou elemento passivo).

Para a *footprint*, é necessária uma observação cautelosa do componente. As dimensões, normalmente especificadas no *datasheet*, servem como referência para criação das *footprints*. Em componentes de montagem na superfície da placa deve ser associado um *pad* para cada pino existente no símbolo do esquema elétrico. Para componentes cujos pinos atravessam a placa o mesmo deve ser feito com os furos presentes na *footprint*. O espaçamento entre os *pads* ou furos deve ser compatível com as dimensões e tolerâncias do componente. Para alguns componentes, como conectores, é importante indicar o espaço que este ocupará na placa para facilitar o posicionamento de todas as partes.

Não só para os componentes devem ser criados símbolos e *footprints*. Para pontos de teste, parafusos de fixação e até serigrafias é preciso indicar suas ligações e posições no *software*. Para pontos de testes é necessário indicar o espaço necessário para as ponteiras do osciloscópio os alcançarem. O mesmo deve ser feito com parafusos para permitir a rotação e o acesso de uma chave a eles.

3.2.2.3 Desenvolvimento do Esquema Elétrico

Utilizando o *software* ECAD, foram conectados todos os componentes relativos ao projeto. Por exemplo, para o esquema da interface *Ethernet*, representado na Figura 5, foram conectados os resistores e capacitores como terminações com a intenção de garantir o casamento de impedâncias reduzindo as reflexões da linha.

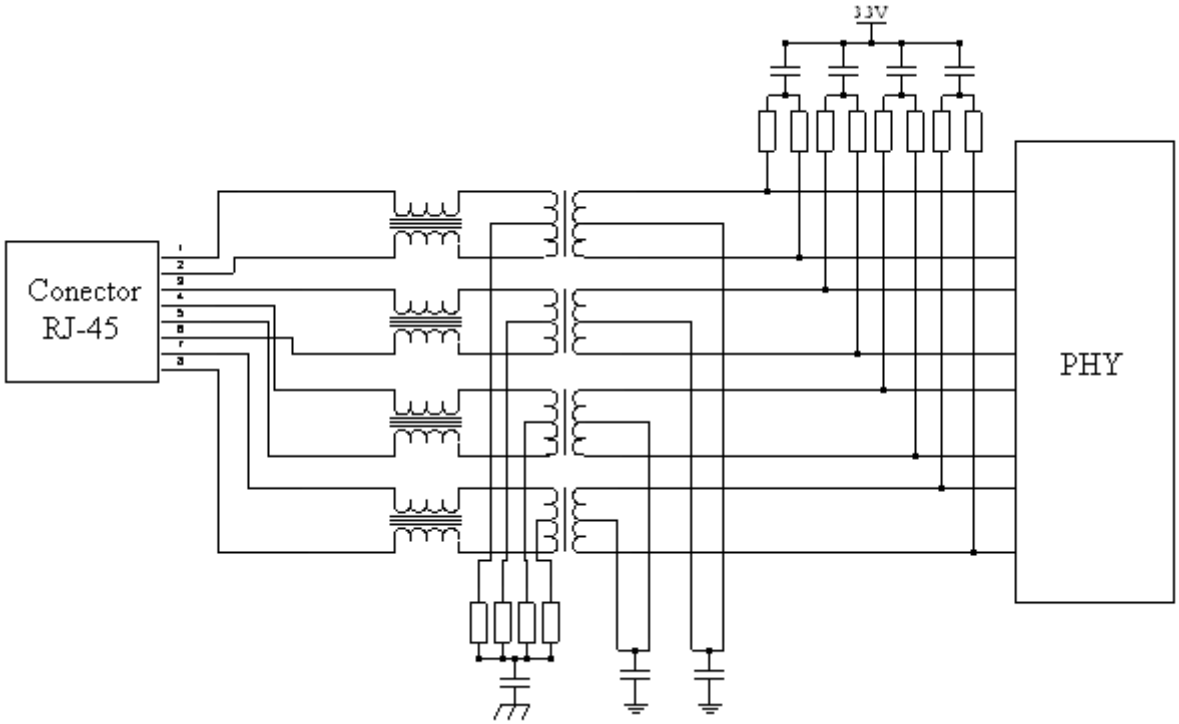

Figura 5 - Esquema elétrico das terminações da interface Ethernet

O esquema elétrico foi desenvolvido no modo hierárquico, ou seja, o projeto foi dividido em partes, cada parte foi representada em uma página com indicação de portas de entrada e saída de sinais e salva individualmente em um arquivo. Na página que contém o topo do projeto, todas as partes são conectadas, podendo, inclusive, serem replicadas inúmeras vezes. A base de um esquema elétrico é a arquitetura de *hardware* do projeto, e é

bastante interessante que a página de topo do esquema esteja semelhante à divisão em blocos efetuada na arquitetura, a fim de facilitar a compreensão do projeto como um todo.

Terminada essa etapa, utiliza-se o software para verificar se todos os pinos dos componentes e portas de cada parte do projeto foram devidamente conectados.

3.2.2.4 Desenvolvimento do Leiaute

Antes de iniciar a traçar as trilhas que ligam as alimentações e os sinais dos componentes, processo conhecido como roteamento, é necessário definir o formato da placa, a quantidade de camadas disponíveis para traçar as trilhas e o posicionamento dos componentes.

O custo de fabricação de uma placa está diretamente ligado ao tamanho, irregularidades no formato e quantidade de camadas utilizadas. Por esse motivo, o número de camadas deve ser o mínimo possível, desde que todas as ligações sejam feitas de forma a garantir o mínimo de ruído e o funcionamento estável dos componentes.

Costuma-se utilizar planos que podem ou não ocupar a totalidade da área da placa para sinais de alimentação, isso permite que sinais de alta freqüência consigam caminhos de retorno com uma indutância inferior a de uma trilha simples. Para projetos que utilizem tais sinais, como o presente caso, pode-se considerar essa técnica praticamente obrigatória.

O posicionamento dos componentes deve iniciar por aqueles cuja localização seja definida por elementos externos, como, por exemplo, as partes mecânicas em que a placa será fixada. Estes componentes normalmente são conectores e chaves. A posição dos demais deve ser considerada de forma a facilitar o roteamento e garantir que não haverão trilhas muito longas para sinais críticos e nem com uma diferença de comprimento considerável entre aquelas que compõem um barramento.

O *software* permite que sejam definidas restrições como distância entre trilhas, comprimento máximo e impedância característica.

Feita a verificação de que todas as conexões já foram roteadas e que todas as restrições foram atendidas, o leiaute é então enviado para fabricação.

3.2.3 Montagem

Com exceção de alguns conectores e poucos componentes, as peças utilizadas no projeto são todas produzidas de acordo com a tecnologia de montagem sobre superfície – SMT, do inglês *Surface Mount Tecnology* –, a qual possibilita que um grande número de placas de complexidade elevada sejam produzidas em curtos espaços de tempo. Basicamente, os componentes são posicionados sobre a placa, cuja pasta de solda já foi introduzida sobre os *pads*. A placa é então levada a um forno de temperatura controlada onde a pasta de solda é fundida, fixando o componente a placa.

Os componentes do tipo "pino através do furo" – PTH, do inglês *Pin Through Hole* – são soldados manualmente uma vez que o processo de soldagem SMT é finalizado.

3.3 Desenvolvimento em Lógica Programável

3.3.1 Linguagem de Descrição de *Hardware*

Linguagens de Descrição de *Hardware* (LDH) são expressões das estruturas temporais e espaciais e do comportamento de sistemas eletrônicos traduzidas em texto. Semelhantemente a algumas linguagens de programação, LDHs possuem em sua sintaxe

formas de explicitar processos e eventos que ocorrem paralela ou seqüencialmente. No entanto, existe a possibilidade de se indicar noção de tempo e de espaço nas LDHs, características próprias de *hardware*.

A grande maioria das linguagens de descrição de *hardware* possui um foco exclusivo em sistemas digitais, havendo uma carência muito grande em LDHs próprias para circuitos analógicos e maior ainda para circuitos mistos.

Como a utilização de LDHs neste projeto é especifica para configuração de FPGAs, ou seja, sistemas digitais, será descrita brevemente apenas a linguagem utilizada para a realização deste projeto: VHDL.

3.3.1.1 VHDL

Desenvolvida durante a década de 1980, a VHDL tinha uma proposta inicial de documentar os projetos de circuitos integrados controlados pelo Departamento de Defesa Americano. Em pouco tempo, porém, já haviam simuladores capazes de interpretar as descrições e, em seguida, ferramentas sintetizadoras de circuitos digitais baseados na linguagem.

A estrutura de uma descrição em VHDL pode ser divida, basicamente, em duas partes: entidade e arquitetura. Na primeira são descritos todos os sinais de entrada e saída, além de valores genéricos utilizados como constantes na descrição. Na segunda, os processos e atribuições de sinais é que são descritos, incluindo as declarações daqueles que não fazem parte das entradas e saídas da entidade.

Assim como nas linguagens de programação, podem ser utilizadas bibliotecas com funções e constantes pré-definidas. Estas bibliotecas devem ser descritas em arquivos

separados e identificadas como pacotes ou, em inglês, *packages*. Para serem utilizadas, as mesmas devem ser declaradas no topo da descrição, antes mesmo da declaração da entidade.

A seguir, um exemplo em VHDL de uma porta OR com sua saída registrada na borda do relógio definida por um genérico.

```
library IEEE;
use IEEE.std_logic_1164.all;

entity OR_GATE is
     generic (
          CLK_EDGE  : std_logic := '1'
          );
     port (
          CLK  : in std_logic;
          A    : in std_logic;
          B    : in std_logic;
          OUT  : out std_logic
          );
end OR_GATE;
architecture OR_ARCH of OR_GATE is
begin

     process(CLK)
     begin
          if CLK'event and CLK=CLK_EDGE then
               OUT       <= A or B;
          end if;
     end process;

end OR_ARCH;
```

A entidade OR_GATE pode ser utilizada em outras descrições do mesmo projeto ou até mesmo reutilizada em outros projetos sem a necessidade de se reescrever a arquitetura, bastando atribuir sinais da entidade em que ela é inserida a cada um dos sinais de entrada e saída declarados.

3.3.2 Módulos Desenvolvidos

Para o desenvolvimento do Controlador de Acesso ao Meio foram descritos em VHDL sete módulos. São eles:

- RGMII_RX
- RGMII_TX
- ETH_RX
- ETH_TX
- uPC_INTERFACE
- RMII_TX
- RMII_RX

O diagrama de blocos que representa como foram organizados estes blocos encontra-se na Figura 6.

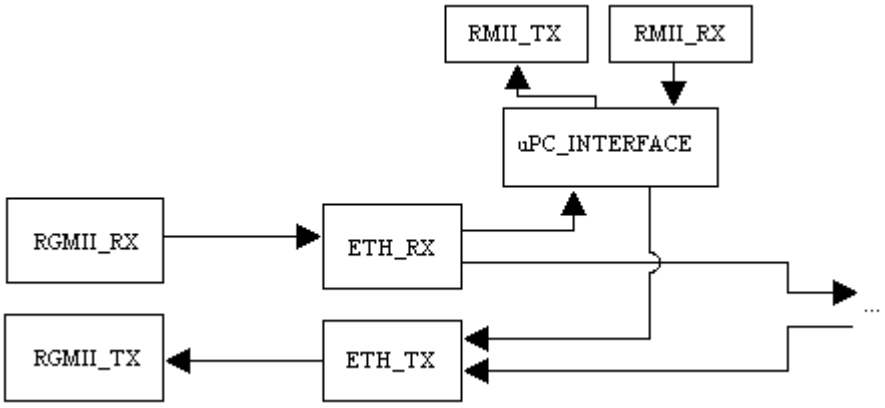

Figura 6 - Relação dos blocos da lógica programável

3.3.2.1 RGMII_RX

Este módulo recebe diretamente os dados do PHY e entrega dados bem estruturados com indicação de começo e final de pacote e erro de recepção para os módulos seguintes, além de indicação da velocidade de operação, *status* de conexão e outros erros indicados pelo PHY. O relógio de referência é o RXC, fornecido pelo PHY, e varia sua freqüência de acordo

com a velocidade que o mesmo estabeleceu para a conexão. É o único módulo que não possui como referência o relógio do sistema.

O barramento RD é amostrado nas bordas de subida e descida do relógio. Como o processo é sensível somente à borda ascendente, considera-se RD como um sinal de oito bits, denominado RXD, sendo os quatro menos significativos amostrados durante a borda de subida e os mais significativos na borda de descida.

No entanto, se a velocidade de operação não for a do modo *Gigabit Ethernet*, então só serão considerados os valores de RD amostrados na borda de descida de RXC e a máquina de estados só será habilitada quando o sinal RXD de oito bits estiver completo.

A máquina de estados funciona de forma relativamente simples e possui apenas três estados: INATIVO, ESPERA_SFD e DADOS.

- INATIVO: Utiliza o barramento RD como indicação de *status* como velocidade de conexão, conexão estabelecida, portadora de sinal detectada, etc... de acordo com o padrão da interface RGMII. Quando o sinal RX_CTL estiver no nível lógico ‘1’ durante as duas transições de RXC, será modificado o estado para ESPERA_SFD.
- ESPERA_SFD: Aguarda que o sinal RXD assuma o valor do SFD, 0xD5, para seguir para o estado DADOS e indicar o início do pacote de dados. Caso RXD seja diferente do valor do preâmbulo, 0x55, ou do valor do SFD, ou ainda que o sinal de controle RX_CTL não se mantenha no nível lógico ‘1’ durante esse período, será indicado erro e máquina voltará ao estado INATIVO.
- DADOS: O valor de RXD é repassado e é dada a indicação de que o dado é válido. É iniciado um contador para verificar o tamanho do pacote de dados. Caso o contador indique um tamanho de pacote maior que o permitido pelo padrão (1522 *bytes*), é indicado erro e o final do pacote e a máquina retorna

para o estado INATIVO. Só não será indicado erro ao final de um pacote quando o sinal RX_CTL efetuar a transição para o nível lógico ‘0’ na borda de descida de RXC e permanecer assim na borda seguinte, e ainda que o contador não apresente valor inferior ao permitido (64 *bytes*), quando, então, será indicado o final da recepção sem erros.

Antes de prosseguir para o próximo módulo, os dados são inseridos em uma memória do tipo *First In First Out* (FIFO), cujo controle de escrita é síncrono com o sinal RXC e o controle de leitura com o relógio do sistema de freqüência fixa em 125MHz.

3.3.2.2 RGMII_TX

Este módulo transmite os dados diretamente para o PHY. O barramento TD é acionado por *flip-flops* DDR em cujas entradas para a borda de subida do relógio estão os quatro bits menos significativos do sinal TXD e, para as de descida, os mais significativos.

O módulo possui como entradas a indicação de validade de dado, a indicação da velocidade de operação e o dado de oito bits.

Quando a validade dos dados de entrada é indicada, o sinal TXD assume os valores registrados, o sinal TX_CTL assume o nível lógico ‘1’ e permanece assim até que, no próximo registro de dados, não seja indicada validade.

Embora o módulo possua como referência constante o relógio de 125MHz, é necessário que a taxa de transmissão de dados seja a mesma definida pelo PHY. Ou seja, para o modo *Fast Ethernet* as transições dos sinais de transmissão para o PHY só poderão ocorrer a uma taxa de 25MHz.

Para que seja possível trabalhar em mais de um modo sem a necessidade de alternar o relógio, criou-se um sinal de habilitação de operação. Este sinal permanece sempre no nível lógico ‘1’ quando o módulo opera no modo *Gigabit Ethernet* e varia de forma a atingir a correta freqüência de operação quando estiver utilizando outros modos.

3.3.2.3 ETH_RX

O módulo ETH_RX é responsável por filtrar pacotes e encaminhar dados de áudio para os demais módulos contidos no FPGA. Pacotes que não contenham dados de áudio serão encaminhados para o módulo uPC_INTERFACE.

Os seis primeiros *bytes* indicam o endereço MAC de destino do pacote. Caso este endereço seja do tipo *Multicast* e pertencente aos endereços próprios do protocolo de transmissão de áudio, o pacote será tratado como portador de dados de áudio e será verificado, através das informações contidas nele, se as condições com que estes dados foram amostrados são válidas para reprodução no equipamento. Caso o endereço seja do tipo *Multicast,* mas não pertencer aos endereços próprios do protocolo de transmissão de áudio, será verificado se o filtro para pacotes deste tipo está habilitado e, caso negativo, os dados deste pacote serão encaminhados para o módulo uPC_INTERFACE. De forma análoga serão tratados os pacotes do tipo *Broadcast*.

Se for detectada uma transmissão do tipo *Unicast,* o endereço de destino do pacote será comparado ao endereço MAC da porta *Ethernet* em questão. Caso sejam idênticos, o pacote será repassado para o módulo seguinte. Se o filtro *Unicast* estiver desabilitado os pacotes serão enviados independentemente do endereço MAC registrado para a porta.

Paralelamente a verificação de endereços e repasse de pacotes o módulo realiza uma operação chamada CRC-32 com os dados contidos no pacote. Essa operação nada mais é do

que uma operação matemática com todos os dados ali contidos para verificar a integridade e validade dos mesmos.

A CRC-32 é uma operação detectora de erros que gera uma seqüência de 32 bits para uma quantidade infinita de dados. Uma mínima alteração em qualquer bit destes dados produz uma seqüência completamente diferente.

Cada pacote possui a seqüência, gerada pela CRC-32, dos dados contidos nele, nos seus últimos quatro *bytes*. Se o cálculo feito pelo MAC diferir da seqüência encontrada no pacote, algum erro ocorreu durante a transmissão e os dados daquele pacote não podem ser considerados válidos e devem ser descartados.

3.3.2.4 ETH_TX

Os dados de áudio que são amostrados no equipamento em que se encontra o MAC devem ser inseridos neste módulo. A função elementar deste módulo é construir e estruturar pacotes de dados e controlar a transmissão dos mesmos.

Antes de enviar os dados de cada pacote para o módulo RGMII_TX, é necessário transmitir o preâmbulo e o SFD, ambos mencionados na seção 2.3.

A estrutura do pacote de áudio faz parte de um protocolo proprietário da empresa DSPRO e por isso não serão dados muitos detalhes da mesma. A descrição do módulo, no entanto, não será prejudicada.

Para o pacote de áudio é inserido, inicialmente, um endereço MAC do tipo *Multicast*. Isso possibilita que *switches* gerenciáveis possam isolar a rede por onde circulam os dados de áudio. Em seguida, é adicionado ao pacote o endereço MAC da porta em que ele será transmitido. Após os dois endereços, existe um campo chamado de *Ethertype,* utilizado para identificar qual protocolo está sendo utilizado para a geração daquele pacote. Segue-se então

com a inserção de dados onde as amostras de áudio de diferentes canais são misturadas com dados como número de canais válidos e taxa de amostragem das referidas amostras.

Por fim insere-se a FCS, os quatro bytes calculados de forma idêntica ao módulo ETH_RX..

Para pacotes provenientes do módulo uPC_INTERFACE, apenas é substituído o endereço MAC de origem, onde consta o endereço do microcontrolador, pelo endereço da porta. Como houve mudança nos dados do pacote é preciso recalcular e substituir a FCS destes pacotes;

Uma vez encerrada a transmissão, espera-se por um período equivalente a transmissão de doze *bytes* para iniciar a transmissão de um novo pacote de dados.

3.3.2.5 uPC_INTERFACE

Como existem quatro portas *Ethernet* em cada equipamento, e apenas um MAC no microcontrolador, é preciso gerenciar a distribuição no tempo dos dados de comunicação entre cada interface e o microcontrolador. Para essa tarefa foi desenvolvido o módulo uPC_INTERFACE.

O módulo comunica-se com o MAC do microcontrolador, o qual utiliza a interface RMII. Esta interface não funciona para o modo o *Gigabit Ethernet* e possui uma freqüência de operação fixa em 50MHz. Para este problema utilizou-se uma solução idêntica ao do módulo RGMII_TX, onde foi criado um sinal específico para habilitar a operação.

Os pacotes oriundos de cada uma das quatro interfaces são armazenados em memórias distintas. Ao terminar o envio de dados de uma interface para o microcontrolador verifica-se

se existe algum pacote armazenado da interface seguinte e, caso negativo, verifica-se na seguinte e assim sucessivamente.

Para que microcontrolador tome conhecimento de qual interface foi recebido aquele pacote, o módulo adiciona um byte antes da FCS indicando qual das quatro interfaces é a origem. Esta alteração modifica a FCS válida para os dados, logo, para evitar que o pacote seja descartado pelo MAC do micro controlador, deve-se repetir a operação CRC-32 e substituir o valor da FCS.

Da mesma forma o microcontrolador indica para qual interface ele deseja que seja transmitido o seu pacote. O *byte* de identificação de interface é, então, suprimido e o pacote enviado para módulo ETH_TX da interface correspondente. Nesse caso, não há necessidade de substituir a FCS, pois isso será feito de qualquer maneira no módulo seguinte.

Para a interface com o MAC do microcontrolador utilizam-se os módulos RMII_TX e RMII_RX, para transmissão e recepção, respectivamente.

3.3.2.6 RMII_TX

O módulo RMII_TX possui comportamento análogo ao módulo RGMII_TX, com a exceção de que não necessita indicação do modo de operação por estar sempre operando no modo *Fast Ethernet*.

Os sinais das duas interfaces são semelhantes, porém a interface RMII possui um barramento com apenas dois bits para transmissão de dados contra quatro bits da interface RGMII. Nem os dados, nem nenhum outro sinal da interface RMII são DDR, nem podem ser utilizados para indicações de *status* da conexão.

Outra diferença entre as interfaces é que a transmissão e a recepção utilizam o mesmo relógio de referência, podendo ser fornecido pelo MAC, pelo PHY, ou ainda de uma fonte externa.

3.3.2.7 RMII_RX

Utilizado para recepção dos pacotes originados no microcontrolador, este módulo possui um funcionamento bastante semelhante ao RGMII_RX. Assim como o módulo RMII_TX, não necessita de indicação de velocidade.

Os erros de recepção podem ser detectados de duas maneiras, através do sinal que tem origem no PHY, chamado RX_ER, ou ainda no barramento de dados que mantém valor fixo em “01” até o fim da transmissão.

Como nesta aplicação, estes módulos estão sendo usados para emular um PHY não haverá ocorrências de erros que possam ser detectados desta maneira.

4 SIMULAÇÕES E IMPLEMENTAÇÃO FINAL

4.1 Testbenchs

Durante o desenvolvimento da lógica programável, de forma iterativa, são realizadas simulações para verificar e validar o trabalho. Para tais simulações, utilizam-se descrições em VHDL onde é inserida a entidade de topo do projeto e são atribuídos sinais às suas entradas. Estas descrições são chamadas de *testbenchs*.

Uma vantagem de se utilizar *testbenchs* para simulações é a possibilidade de utilizar funções de VHDL que costumam não ser implementáveis, como operações do tipo ponto flutuante e relações temporais entre sinais. Um exemplo desse tipo de função são eventos que ocorrem em intervalos de tempo bem definidos ou atrasos de sinais descritos de forma explícita. Para gerar um sinal de relógio de 100MHz, por exemplo, pode-se introduzir o seguinte comando:

```
clock     <= not clock after 5 ns;
```

Para um sinal que represente o mesmo relógio, porém com um atraso de 20ns pode-se escrever:

```
clock_dly <= transport clock after 20ns;
```

Outra vantagem das *testbenchs* em VHDL é a portabilidade, pois permite que diferentes ferramentas de simulação sejam utilizadas sem a necessidade de se gerar um vetor de testes compatível com cada uma delas.

4.2 Simulações do projeto

Para este projeto foram desenvolvidas *testbenchs* que simulavam diferentes pacotes, incluindo pacotes com erro, na recepção do sistema. Nas figuras Figura 7, Figura 8, Figura 9 e Figura 10 é possível ver o resultado da simulação dos sinais de entrada do módulo RGMII_RX. Na Figura 7, há o início da recepção de um pseudo-pacote no modo *Gigabit Ethernet* que contém apenas dados de um contador.

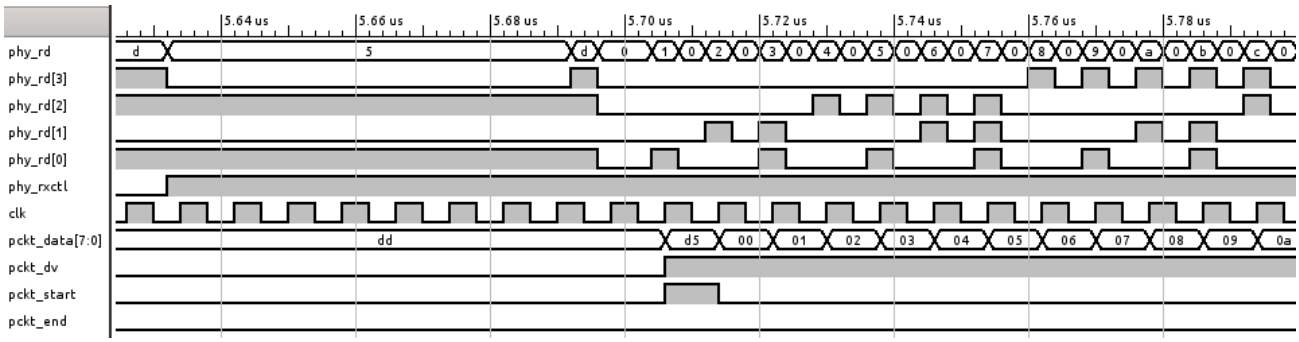

Figura 7 - Simulação do ínicio da recepção de um pacote no modo *Gigabit Ethernet*

Pode-se observar que o módulo começa a repassar os dados somente depois de receber um preâmbulo válido e que o primeiro *byte* transmitido para o módulo seguinte é o SFD. Este pacote possui como endereço MAC de destino o valor 00:01:02:03:04:05, o bit menos significativo do segundo octeto do endereço de destino identifica se este endereço é do tipo *Multicast* ou *Unicast*. Neste caso, onde o segundo octeto é 0x01, o endereço será tratado como *Multicast* e o pacote só será reencaminhado para o microcontrolador se o respectivo filtro do módulo ETH_RX não estiver habilitado.

Na Figura 8 está representado o fim da recepção deste mesmo pacote. Os últimos quatro bytes deste pacote representam a seqüência verificadora de pacote. Como os valores da seqüência seguem os valores do contador e não o valor correto para a verificação, o módulo

ETH_RX identificará um erro no pacote e irá descartar o mesmo independentemente do filtro de pacotes *Multicast* estar habilitado ou não.

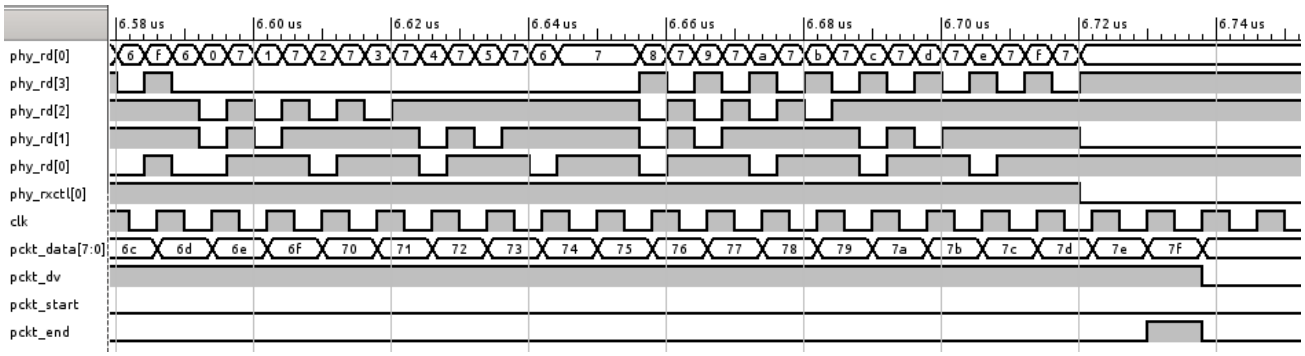

Figura 8 - Simulação do final da recepção de um pacote no modo *Gigabit Ethernet*

Para comparação, o mesmo pacote foi utilizado como teste para recepção no modo *Fast Ethernet*. Nas figuras Figura 9 e Figura 10 é possível observar um valor maior no período do relógio e que a indicação de validade de dados é alternada, diferentemente do modo *Gigabit Ethernet*. Isso ocorre devido à necessidade de aguardar dois ciclos completos de relógio para a recepção dos 8 bits do sinal RXD.

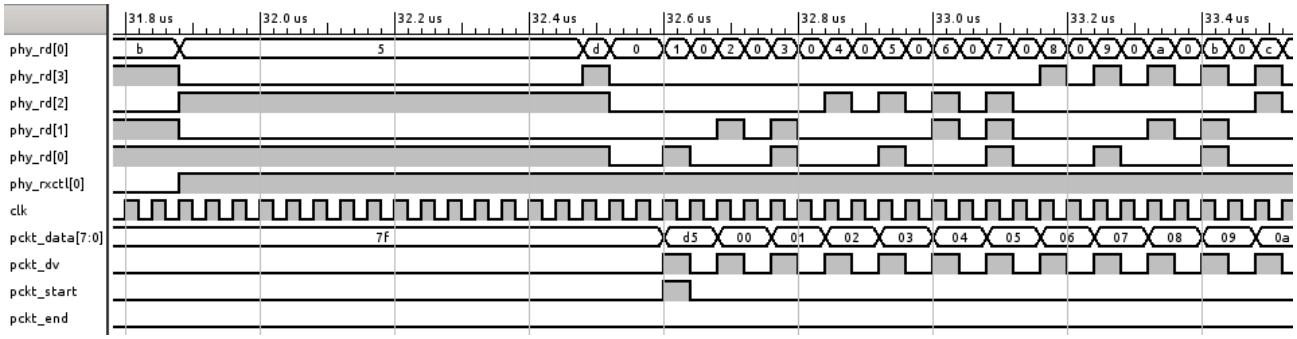

Figura 9 - Simulação do início da recepção de um pacote no modo *Fast Ethernet*

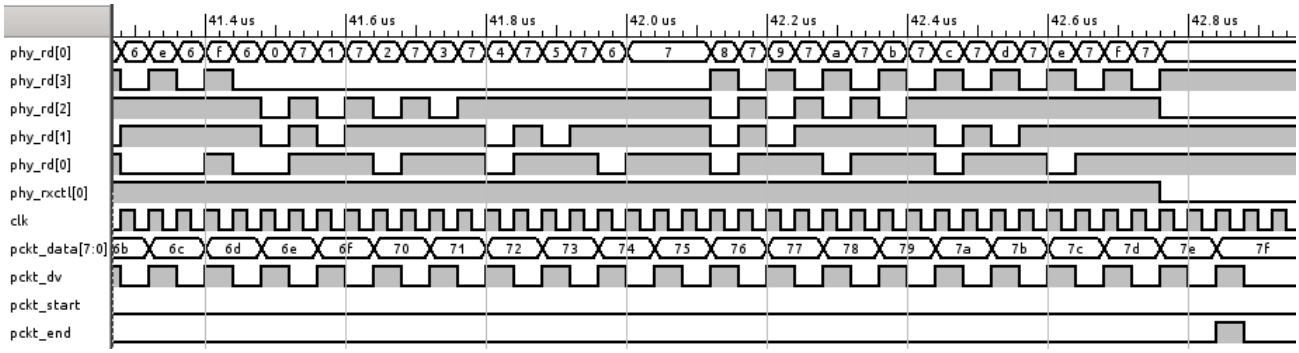

Figura 10 - Simulação do final da recepção de um pacote no modo *Fast Ethernet*

O mesmo teste é realizado com diferentes pacotes, incluindo pacotes pertencentes ao protocolo de áudio e pacotes com erros na recepção. Sempre que ocorrer alguma discrepância entre o resultado esperado e o obtido na simulação o projeto será revisado visando corrigir o erro.

4.3 Implementação

Utilizando a ferramenta de síntese disponibilizada pela empresa fabricante do FPGA, os arquivos contendo as descrições em VHDL são traduzidos para circuitos lógicos. Para o posicionamento e roteamento do circuito lógico a ferramenta leva em consideração restrições definidas pelo usuário, como, por exemplo, a posição dos pinos utilizados como portas de entrada e saída dos sinais, o tempo máximo de *setup* para sinais referenciados a cada relógio, a máxima potência a ser dissipada ou a diferença máxima do atraso entre dois ou mais sinais.

Caso alguma restrição não seja cumprida durante a implementação, é preciso revisar o projeto e utilizar técnicas que permitam atingir o desempenho ou a utilização de lógica desejada. No caso de uma restrição de tempo de *setup*, por exemplo, é possível registrar o mesmo sinal em diversos *flip-flops* facilitando o roteamento para alcançar diferentes regiões do FPGA. Para reduzir a utilização de lógica existem diversas técnicas que podem ajudar no

cumprimento da restrição, a reutilização de lógicas idênticas em diferentes módulos ou a utilização de LUTs como memória distribuída ao invés de *flip-flops*, são alguns exemplos.

Concluída a implementação com todas as restrições sendo atingidas, a lógica utilizada por cada um dos módulos deste projeto está descrita na Tabela 2.

Módulo	Repetições	LUTs	FFs
RGMII_RX	4	74	50
RGMII_TX	4	45	38
ETH_RX	4	193	104
ETH_TX	4	87	64
uPC_INTERFACE	1	170	145
RMII_TX	1	43	31
RMII_RX	1	65	53

Tabela 2 - Utilização final de lógica

4.4 Testes finais

Por fim, o trabalho é verificado em condições práticas. Sinais internos do FPGA são mapeados para os pontos de teste e, quando há interesse, analisados em um osciloscópio. Diferentes pacotes são enviados por cada uma das quatro interfaces e, utilizando pontos de testes ou o pacote de resposta da mesma interface, é verificado se o projeto apresenta o comportamento esperado.

Como primeiro teste, os PHYs de cada uma das quatro interfaces são configurados para trabalhar somente no modo *Fast Ethernet*. É enviado um pedido de eco, ou *ping*, e espera-se uma resposta por parte do microcontrolador. Cada interface é testada separadamente e, para utilizar, inicialmente, apenas o endereço MAC *Unicast* da porta, o mesmo é definido manualmente no computador antes de enviar o pedido para o endereço IP do microcontrolador.

Após, o mesmo procedimento é realizado utilizando endereços MAC configurados, propositalmente, de forma errada. Espera-se que os filtros do MAC descartem os pacotes e não os repassem para o microcontrolador, pois, para esta aplicação, os filtros *Unicast* devem estar sempre habilitados.

Para verificar se o PHY está reencaminhando pacotes do tipo *Broadcast*, o computador é configurado para descobrir o endereço MAC automaticamente, da forma padrão, através de pacotes de questionamento para toda a rede.

Para o teste da operação concorrente das quatro interfaces, computadores são conectados simultaneamente em cada porta e realizam os pedidos de eco ao mesmo tempo.

Cada PHY é, então, configurado para operar no modo *Gigabit Ethernet*. Os testes anteriores são repetidos exatamente da mesma forma.

Ainda é necessário testar o uso simultâneo de interfaces operando em diferentes modos, para isso, o mesmo teste da operação concorrente é repetido, porém com alguns PHYs configurados para operar no modo *Gigabit* e outros no modo *Fast*.

Como os módulos referentes às interfaces de áudio ainda não haviam sido desenvolvidos à época da conclusão deste trabalho, pacotes condizentes com o protocolo de transferência de áudio, porém com amostras artificiais, são gerados no computador e transmitidos para as interfaces *Ethernet* do equipamento. Os sinais presentes nos pontos de teste são utilizados validar o módulo de recepção de áudio.

Para o teste do módulo de transmissão de áudio, é habilitada a transmissão com amostras constantes contendo o número do canal a que pertence. Por exemplo, para o canal de número cinqüenta será atribuído o valor 0x000032 para todas as amostras. Os pacotes são capturados e analisados com a ajuda de um computador.

Havendo algum erro ou comportamento inesperado o projeto é revisado visando corrigir as anomalias. Na fase de testes não é considerada somente a lógica programável, mas também o *hardware*, incluindo possíveis erros de montagem.

Após algumas iterações nos procedimentos de teste, simulação e desenvolvimento, o projeto de um Controlador de Acesso ao Meio com as especificações contidas na Seção 2.3 pode ser considerado concluído.

5 CONCLUSÃO

O desenvolvimento de qualquer interface de comunicação requer rigor na adequação às normas do padrão. No caso de uma interface tão popular como a Ethernet, a compatibilidade com os mais diversos dispositivos, de diferentes fabricantes, torna-se um ponto crítico. Os cuidados para garantir estabilidade e robustez são obrigatórios em qualquer projeto que deseje agregar qualidade ao produto e, neste caso, não foi diferente.

Embora este trabalho apresente apenas uma parcela do projeto de um produto, a compreensão e participação de todas as etapas de um projeto, desde a especificação até o acabamento final, é de suma importância para a formação de um engenheiro. A capacidade de transformar o que foi em algum momento nada além de uma idéia em algo concreto é, sem dúvidas, o maior trunfo da profissão.

No futuro, podem haver modificações neste projeto, como a adoção de um número diferente de interfaces ou a utilização de um novo protocolo de transferência de áudio. As possíveis mudanças, no entanto, não acrescentam significativas alterações no trabalho aqui descrito.

Para o desenvolvimento final do produto é preciso considerar diversos fatores.

Nos produtos anteriores existem módulos de conversão de áudio analógico-digital e digital-analógico. Este trabalho foi desenvolvido com a intenção de proporcionar compatibilidade com estes módulos, porém permite que sejam desenvolvidos novos módulos com diferentes funções.

Para que a compatibilidade com novos e antigos módulos de conversão de áudio seja possível é necessário que outros projetos complementares a este, que também fazem parte do desenvolvimento do produto, sejam finalizados.

A utilização do mesmo protocolo, mais especificamente de sua adaptação, para os modos *Fast Ethernet* e *Gigabit Ethernet*, permite a transferência bidirecional de até 80 e 800 canais de áudio, respectivamente. Como são raríssimas aplicações que utilizem o número de canais proposto para o modo *Gigabit Ethernet*, é muito provável que o protocolo de transferência de áudio para este modo seja modificado, reduzindo o espaço de banda utilizado para o áudio e agregando novas funções.

As especificações de tal protocolo, no entanto, ainda não foram definidas. O que, como já dito, não significa a possibilidade de necessidade de grandes modificações no projeto aqui descrito, por alterar, no máximo, o modo como o controlador identifica dados de áudio.

Enfim, embora o trabalho aqui descrito seja voltado para a distribuição de áudio, basta direcionar todos os dados os de saída dos módulos de recepção para *buffers* com acesso externo e utilizar dados de transmissão de outros *buffers* também com acesso externo que teremos um Controlado de Acesso ao Meio que pode ser utilizado em qualquer aplicação *Ethernet*. Para isso, é necessário apenas que a interface com a camada física se dê pelo padrão RGMII e o controlador estará pronto para operar nos modos *Ethernet* (10Mbps), *Fast Ethernet* e *Gigabit Ethernet*.

6 REFERÊNCIAS

[1] INSTITUTO DE ENGENHEIROS ELETRICISTAS E ELETRÔNICOS. **Std. 802.3ab**, 1999.

[2] NORRIS, Mark. **Gigabit Ethernet Technology and Applications**. Artech House, 2002.

[3] HP, BROADCOM E MARVELL. **Reduced Gigabit Independent Interface v2.0,** 2002.

[4] VHDL ANALYSIS AND STANDARDIZATION GROUP. < http://www.eda.org/twiki/bin/view.cgi/P1076/WebHome > Acessado em 29 de maio de 2011.

[5] HARTMUT, Sadrozinski; JINYUAN Wu. **Applications of Field-Programmable Gate Arrays in Scientific Research**. Taylor & Francis. 2002.