# Protocolos de Comunicação em Circuitos Integrados: Experiências e Resultados

Ricardo N. B. Lima, Emerson Carli, Aloysio C. P. Pedroza
Luci Pirmez, Antônio C. de Mesquita

Grupo de Teleinformática e Automação (GTA)
COPPE/EE - Programa de Engenharia Elétrica
Universidade Federal do Rio de Janeiro
Caixa Postal 68504 - CEP 21945-970 - Rio de Janeiro - RJ - Brasil
FAX: + 55 21 2906626
{belem, emerson, aloysio}@gta.ufrj.br

*Sumário*— ***Este trabalho apresenta os resultados obtidos a partir da utilização de uma metodologia que permite a implementação em hardware e a prototipagem rápida de protocolos de comunicação. São considerados os resultados de implementação de diferentes estilos de descrição de protocolos, resultados de simulação e recomendações quanto ao uso desta metodologia. Os resultados incluem análise de área e avaliação da velocidade de operação dos circuitos sintetizados.***

## 1 Introdução

Este trabalho apresenta os resultados obtidos com a aplicação de uma metodologia que permite a implementação em hardware e a prototipagem rápida de protocolos de comunicação. Esta metodologia é baseada no uso das linguagens Estelle [1, 2] e VHDL (*Very High-speed Integrated Circuits Hardware Description Language*) [3, 4], no uso da técnica de Síntese de Alto Nível (HLS - *High Level Synthesis*) para automação de projetos de sistemas digitais [5] e na utilização de um ambiente específico para projeto de protocolos a nível de software e outro ambiente voltado para implementação de circuitos integrados.

O projeto de circuitos integrados se inicia a partir de uma especificação do sistema em um alto nível de abstração, normalmente a partir de uma descrição comportamental, utilizando um conjunto de ferramentas de projeto integradas em ambientes específicos. O projeto de protocolos de comunicação, em muitos casos, envolve especificações do sistema escritas a partir de Técnicas de Descrição Formal (TDF) [6, 7, 8] tais como a linguagem Estelle. Estas linguagens permitem gerar especificações em um nível de abstração maior que o normalmente utilizado nas descrições de sistemas digitais. Como conseqüência, os projetistas de protocolos se deparam com problemas de interação entre os dois ambientes de projeto, que em geral, são baseados em linguagens de descrição diferentes e necessitam de uma interface de tradução de uma linguagem para outra.

Na metodologia [9] usada neste trabalho, o processo se inicia a partir de uma especificação Estelle do protocolo a ser implementado em hardware, submetida a um ambiente específico para o projeto de protocolos, onde será compilada, testada e simulada. O próximo passo refere-se ao mapeamento desta especificação em uma descrição comportamental em VHDL, similar à especificação funcional do protocolo, que é então submetida a um ambiente específico para o projeto de circuitos. Este ambiente compreende as ferramentas de síntese comportamental e lógico Synopsys Behavioral Compiler [10] e Synopsys Design Compiler [11] e, também, a ferramenta de geração de *layout* Opus [12]. Neste trabalho os protocolos ABRACADABRA da ISO e os protocolos AAL5 e IPv6 [13, 14] são sintetizados com o intuito de se avaliar a complexidade do hardware resultante. Os resultados referentes a várias experiências com os protocolos acima citados são discutidos neste trabalho. Estes resultados incluem análise de área e avaliação da velocidade de operação de diversos circuitos sintetizados.

Este artigo está estruturado da seguinte forma. A seção 2 apresenta as linguagens de especificação utilizadas e o conceito de HLS. Na seção 3, todos os passos referentes à utilização da metodologia proposta são abordados. A seção 4 apresenta os diferentes estilos de descrição para protocolos em VHDL. Na seção 5, os resultados obtidos são examinados. As considerações finais se encontram na seção 6.

## 2 Linguagens e Síntese de Alto Nível

A implementação de protocolos de comunicação em software envolve um ambiente específico voltado para a verificação e teste dos protocolos. A síntese em hardware de sistemas envolve um ambiente que visa principalmente a simulação e a implementação física de um circuito. Em geral, estes dois ambientes são baseados em linguagens de descrição diferentes e necessitam de uma interface de tradução de uma linguagem para outra. Esta seção apresenta as principais características das linguagens de especificação utilizadas neste trabalho e os conceitos referentes à Síntese de Alto Nível.

### 2.1 Linguagens de Especificação

Este trabalho utiliza para a implementação de protocolos de comunicação em software a linguagem Estelle. Em Estelle, uma especificação é composta por um conjunto de módulos, onde cada módulo é composto por um cabeçalho e por um corpo. Os pontos de interação, as variáveis exportadas e o atributo de classe são definidos no cabeçalho. O comportamento do módulo é definido no seu corpo através de uma Máquina de Estados Finita (MEF).

A linguagem de descrição de hardware utilizada neste trabalho, VHDL, é uma linguagem padrão para descrever sistemas eletrônicos digitais. O componente básico de uma descrição VHDL é o par entidade e arquitetura. Uma entidade VHDL descreve, através de um conjunto de portas de entrada e saída, a interface externa de um dado sistema. A arquitetura descreve a função da entidade no domínio comportamental ou estrutural. Para uma troca eficiente de informações entre entidades VHDL que descrevem protocolos de comunicação é necessário algum tipo de sincronização entre elas. Esta sincronização é estabelecida por sinais de controle em ambas as entidades comunicantes.

### 2.2 Síntese de Alto Nível

A Síntese de Alto Nível (HLS) tomou um grande impulso devido à adoção de uma linguagem única capaz de descrever sistemas digitais nos diversos domínios de representação. Dentre estas linguagens VHDL é uma das mais adotadas tanto pelos centros de pesquisa quanto pela indústria. Isto permitiu a construção de ambientes de projeto de hardware integrados, onde as etapas intermediárias do processo de síntese são implementadas automaticamente.

Um circuito integrado VLSI (*Very Large Scale Integrated*) pode ser descrito a partir de um dos três domínios de representação: comportamental, estrutural e físico. Cada domínio pode ser hierarquicamente dividido em alguns níveis de abstração, o que permite que um projeto de circuito integrado possa ser classificado pelo seu domínio e pelo seu nível. Estes domínios são ilustrados de um modo mais claro através do diagrama-Y de Gajski [5].

A HLS corresponde ao mapeamento de uma dada especificação no domínio comportamental para o domínio estrutural em um nível superior ao nível RTL (*Register Transfer Logic*). Neste trabalho será empregada a ferramenta Synopsys Behavioral Compiler para mapear uma especificação de projeto em alto nível para o nível RTL e então compilar esta especificação de mais baixo nível com a ferramenta de Síntese Lógica Synopsys. Deve-se levar em conta que protocolos de comunicação são representados por máquinas de estado dominadas por fluxo de controle, o que torna o processo de síntese mais simples.

A arquitetura alvo da técnica de HLS é um modelo computacional geral que contém caminho de dados, memória e elementos de controle. Ao utilizar HLS, o projetista não precisa criar manualmente o caminho de dados através de registradores e nem unidades funcionais mais complexas como somadores e multiplicadores. Além disso, o projetista não precisa construir um controlador ao nível RTL para sequenciar e controlar as ações do caminho de dados e da memória.

A ferramenta Synopsys Behavioral Compiler constrói essa arquitetura através de métodos de *scheduling*, alocação de hardware e compartilhamento de registradores. O projetista não precisa especificar a arquitetura exata de um projeto, podendo então explorar diferentes implementações possíveis objetivando uma arquitetura ótima.

A técnica de HLS oferece, também, redução no tempo de especificação, simulações mais rápidas, exploração arquitetural e alta qualidade dos resultados.

## 3 Metodologia Adotada

Os passos envolvidos na metodologia proposta para a implementação em hardware de protocolos de comunicação podem ser visualizados na figura 1. Esta metodologia parte de um ambiente específico para desenvolvimento de protocolos na linguagem Estelle, aqui denominado de CAD-Estelle. Após a geração de uma especificação em Estelle corretamente simulada e verificada *(1,2)*, esta especificação é convertida para a linguagem VHDL utilizando a estratégia de mapeamento Estelle-VHDL *(3)* desenvolvida por Pirmez em [15].

Esta descrição VHDL será usada como entrada *(4)* para um ambiente específico de desenvolvimento de circuitos integrados, aqui denominado CAD Circuito Integrado. É importante ressaltar que estes dois ambientes de projeto, CAD-Estelle e CAD Circuito Integrado, são independentes. O CAD Circuito Integrado é composto por um conjunto padrão de ferramentas para projeto de circuitos VLSI. Integram este ambiente o Synopsys Behavioral Compiler, o sintetizador lógico Synopsys e o editor de layout Opus da Cadence.

A ferramenta Synopsys Behavioral Compiler é responsável pelo mapeamento de uma especificação

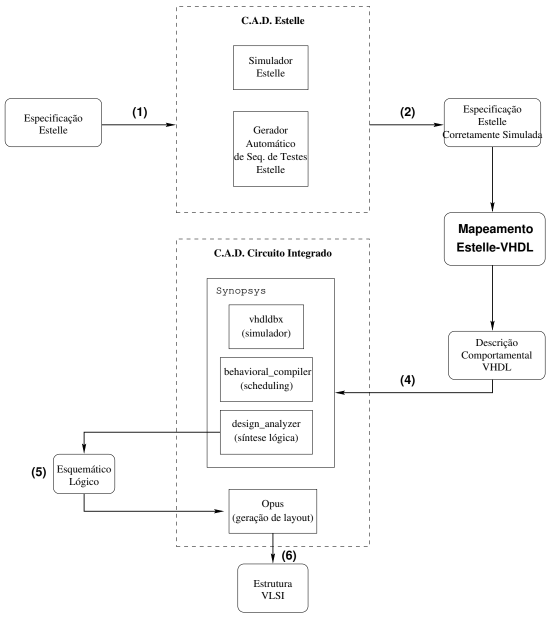


Figura 1: A Metodologia Proposta.

de circuito de um nível mais abstrato para o nível RTL ou para um nível equivalente. Esta ferramenta é importante pois ela é a encarregada de realizar a HLS, logo, todas as vantagens advindas do uso da HLS sobrecaem no uso desta ferramenta. Após esta etapa do processo, ocorre a síntese lógica do circuito, através da utilização do sintetizador lógico Synopsys *(5)*. Por fim, a estrutura VLSI do circuito desejado é obtida através da utilização do editor de layout Opus da Cadence *(6)*.

## 4 Estilos de Descriçã0

Um protocolo pode ser descrito através de diferentes estilos de descrição comportamental na linguagem VHDL [16, 17]. Estes estilos provêem descrições semanticamente equivalentes que diferem somente na qualidade do hardware resultante. A qualidade da descrição obtida após o uso da HLS é medida através da análise da área do circuito sintetizado e também pelo número de estados de controle gerados pelo processo de *scheduling* realizado pelo Synopsys Behavioral Compiler. Será demonstrado que a qualidade da descrição obtida após o uso da HLS dependerá do tipo de construção VHDL utilizada e da seqüência em que estas construções aparecem na descrição de entrada.

Para ilustrar este fato, consideremos o protocolo ABRACADABRA da ISO modelado por uma máquina de estados que realiza operações com outra máquina de estados. Estas operações realizam o processo de abertura e fechamento de conexão. Este protocolo possui duas fases: a fase de inicialização e a fase de processamento. A fase de processamento consiste de um *loop* que espera (*wait* em VHDL) por um sinal de entrada, processa o dado de entrada e gera o resultado correspondente através de um sinal de saída. Este modelo será utilizado para ilustrar o uso de alguns estilos de descrição e os conseqüentes resultados obtidos após o uso da HLS.

A semântica específica de cada estilo de descrição pode ser melhor entendida em [9, 16, 17]. Estes diferentes estilos de descrição são introduzidos no Synopsys Behavioral Compiler e depois usados pelo Synopsys Design Compiler. Diferentes circuitos são sintetizados e a complexidade do hardware resultante será analisada na seção 5.

## 5 Resultados

Nesta seção são discutidos os resultados obtidos após a utilização da metodologia que permite a implementação em hardware de protocolos de comunicação. Os tópicos abordados nesta seção são:

- o processo de simulação das especificações Estelle e VHDL dos protocolos;
- o efeito dos diferentes estilos de descrição comportamental em VHDL nos circuitos sintetizados;
- uma comparação entre a implementação de protocolos utilizando-se a técnica de HLS com biblioteca de células padrão e utilizando-se a técnica de PLD;
- algumas recomedações de projeto.

### 5.1 Resultados de Simulação

A simulação é utilizada para se verificar a funcionalidade das especificações. As simulações foram realizadas em dois pontos ao longo do processo de síntese. Inicialmente ao nível da especificação Estelle e posteriormente ao nível VHDL. No nível Estelle, é usado como entrada no processo de simulação um conjunto de vetores de teste gerados pela ferramenta GASTPC (*Gerador Automático de Seqüências de Teste*) [18]. Este programa gera uma relatório contendo algumas seqüências de envio e recebimento de primitivas de acordo com o serviço oferecido pelo protocolo considerado, as quais são inseridas no Simulador Estelle [19] e mostram todas as possíveis transições que a máquina de estados deve executar de modo a oferecer o serviço proposto corretamente.

Nos casos estudados, as simulações realizadas demostraram que as especificações Estelle e as descrições VHDL estão totalmente de acordo com as especificações funcionais dos protocolos.

### 5.2 Estilos de Descrição VHDL

Como discutido, a linguagem VHDL permite a descrição de um dado comportamento de algumas maneiras diferentes. De modo a obter um desempenho satisfatório para o hardware sintetizado após o processo de síntese, é necessário adotar um estilo de descrição VHDL que leve em conta a estrutura do algoritmo de *scheduling* usado pela ferramenta de síntese.

Como exemplo, uma comparação entre os resultados de síntese gerados pelo Synopsys Behavioral Compiler para seis diferentes estilos de descrição é mostrada na tabela 1. Esta comparação é realizada em termos do número de estados de controle gerados pelo algoritmo de síntese para cada diferente estilo de descrição, pela análise da área do circuito sintetizado e pela velocidade de operação do hardware gerado. A freqüência máxima de operação do circuito é determinado pelo caminho crítico de maior retardo encontrado no circuito sintetizado.

| **Estilos** | **Estados Gerados** | **Área ($mm^2$)** | **Velocidade** |
|---|---|---|---|
| Primeiro | 14 | 1,33 | 125 MHz |
| Segundo | 9 | 1,10 | 149 MHz |
| Terceiro | 9 | 1,13 | 130 MHz |
| Quarto | 9 | 1,37 | 130 MHz |
| Quinto | 9 | 1,27 | 154 MHz |
| TerceiroV2 | 9 | 1,32 | 135 MHz |
| Sexto | 4 | 0,66 | 154 MHz |

Tabela 1: Comparação entre diferentes estilos de descrição.

A partir da tabela apresentada, algumas importantes conclusões podem ser extraidas:

- use a construção "*wait until...control signals*" fora da cláusula "*case state*". A comparação entre o primeiro e o segundo estilos mostra que a área do circuito sintetizado diminuiu e que a velocidade de operação aumentou;

- use a cláusula "*case state*" ao invés da construção "*if..then..elsif..end if*". A comparação entre o segundo e terceiro estilos e entre o quinto e o terceiroV2 mostraram que a área do circuito sintetizado diminuiu e que a velocidade de operação aumentou;

- comparando o terceiro estilo com o terceiroV2 e o segundo estilo com o quinto podemos observar que resetando os diferentes sinais de controle de saída dentro da cláusula "*case state*", a área do circuito diminui. Enquanto que resetando estes mesmos sinais imediatamente após o "*case*", a velocidade de operação aumenta;

- empregue a estrutura "*if..then..elsif..end if*" usando a variável *state* como variável de teste ao invés da variável *input signal*. A comparação entre o terceiro e o quarto estilos mostra que a área do circuito diminuiu;

- use a estrutura "*if..control signals*" ao invés da construção "*wait until...control signals*". A comparação entre o quinto e o sexto estilos mostram que a área do circuito diminuiu.

Tanto em termos de área do circuito e velocidade de operação, o sexto estilo de descrição é o que apresenta os melhores resultados. A figura 2 ilustra o esquema lógico do processo de conexão e desconexão do protocolo ABRACADABRA descrito com o sexto estilo de descrição. O circuito sintetizado possui uma área estimada de 0,66 $mm^2$ obtido usando-se uma biblioteca de células padrão (*standard cells*) AMS com tecnologia 0,8$\mu$m.

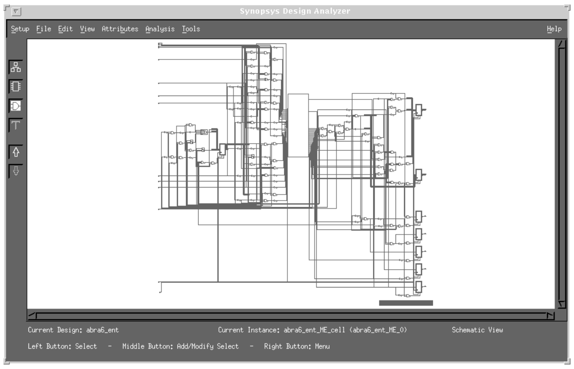


Figura 2: Esquema lógico de uma parte do protocolo ABRACADABRA.

### 5.3 Comparação entre Técnicas de Implementação

Os Dispositivos de Lógica Programável (PLDs) são circuitos integrados digitais que podem ser programados afim de implementar funções complexas, isto é, PLDs podem implementar qualquer expressão booleana definida através de programação.

O processo de HLS utilizando uma biblioteca de *standard cells* e o processo de PLD possuem vantagens semelhantes, como redução no tempo gasto com as especificações, simulações mais rápidas, exploração arquitetural e alta qualidade dos resultados. Porém, o projeto com *standard cells* oferece um maior grau de liberdade na alocação e roteamento de células no *layout* do chip, implicando em um melhor compromisso entre área e velocidade.

Uma comparação foi realizada utilizando uma mesma descrição da parte responsável pelo processo de abertura/fechamento de conexão do protocolo ABRACADABRA. Esta especificação foi implementada utilizando uma biblioteca de células padrão (*standard cells*) AMS com tecnologia 0,8$\mu$m e utilizando ALTERA [20] Max+Plus ll (*Multiple Array Matrix Programmable Logic User System*) [21]. Durante a compilação, o compilador do ambiente ALTERA ficou no modo auto para a escolha do componente mais adequado ao projeto. O modo auto escolheu o componente EPF10K10LC84-3.

Este exemplo mostra uma nítida vantagem em relação à velocidade de operação do projeto utilizando *standard cells*, adotado em conjunto com o Synopsys Behavioral Compiler, em relação à implementação utilizando PLDs. A velocidade em que cada circuito pode operar em cada caso foi de 154 MHz para o projeto com *standard cells* e 33 MHz para a implementação com PLD. Por outro lado deve ser notado que a implementação em PLDs é imediata e o sistema pode ser reprogramado sem a necessidade de uma nova rodada de fabricação, como no caso das *standard cells*.

## 5.4 Recomendações de Projeto

### 5.4.1 Uso de Variáveis Inteiras

Quando uma implementação em software de um protocolo é realizada, é prática comum o uso de variáveis inteiras e sinais na especificação formal do protocolo. Ao nível da especificação Estelle, esta prática não causa maiores problemas. Entretanto, quando são utilizadas linguagens de descrição de hardware, como VHDL, é necessário limitar o *range* de cada variável inteira ou sinal de modo a se reduzir o número de bits de registradores associados a cada variável pela ferramenta de síntese.

Por exemplo, se não houver limitação no *range* de uma variável inteira qualquer em uma determinada descrição VHDL, o Synopsys Behavioral Compiler e o Synopsys Design Compiler alocarão 32 bits para cada registrador responsável em representar esta variável. Para ilustrar este fato, duas descrições VHDL da fase de conexão/desconexão do protocolo ABRACADABRA contendo somente a definição de uma única variável inteira foram sintetizadas. Na primeira descrição, o *range* desta variável inteira não foi limitado. Na segunda descrição, o *range* foi ajustado entre 0 e 7, ou seja a variável inteira só pode assumir estes valores. A tabela 2 mostra uma comparação entre a área dos circuitos sintetizados e o número de bits alocados para os registradores para cada descrição.

| **Descrição** | **Registradores Sintetizados** | **Área ($mm^2$)** |
|---|---|---|
| Com range | 1 de 7 bits | 1,27 |
| Sem range | 3 de 32 bits | 2,37 |

Tabela 2: Comparação entre os circuitos sintetizados.

Observe que na descrição que não houve limitação no *range*, três registradores de 32 bits foram associados à variável inteira, enquanto que na outra descrição somente um registrador de 7 bits foi utilizado. Isto representa uma redução na área de circuito sintetizado de aproximadamente 54%.

### 5.4.2 Armazenamento de Dados

O primeiro passo na síntese de circuitos consiste no particionamento da especificação em dois blocos: a estrutura lógica e a estrutura de armazenamento de dados. A estrutura de armazenamento de dados é geralmente implementada através de uma memória externa. Então, antes de se efetuar o mapeamento Estelle-VHDL de uma determinada especificação, todas as estruturas de armazenamento de dados correspondentes ás variáveis internas, como *buffers*, devem ser implementadas como um módulo separado.

Este fato é ilustrado através da implementação do módulo SAR (*Segmentation and Reassembly sublayer*) do protocolo AAL5 (*ATM Adaptation Layer 5*). Este módulo é responsável pela segmentação do dado recebido. Numa primeira implementação, dois *buffers* foram alocados para o armazenamento de dados, o primeiro *buffer* armazena 88 caracteres e o segundo tem capacidade para 48 caracteres. Ambos foram especificados como variáveis internas na descrição VHDL. O resultado obtido foi um circuito sintetizado com uma área total de 77,41 $mm^2$.

É importante notar que o Synopsys Behavioral Compiler e o Synopsys Design Compiler sintetizam estes *buffers* como um grupo de resgistradores internos ao circuito. Uma solução melhor seria sintetizar a lógica de controle e conectá-la a uma memória externa. Uma segunda implementação foi realizada utilizando uma memória externa conectada à lógica de controle sintetizada. O resultado obtido foi um circuito sintetizado com uma área total de 9,79 $mm^2$.

# 6 Considerações Finais

Este trabalho apresentou os resultados obtidos a partir do emprego de uma metodologia que permite a implementação em hardware de protocolos de comunicação. A comparação entre os resultados de implementação de diferentes estilos de descrição de protocolos mostrou que a adoção de um estilo de descrição a nível comportamental VHDL afeta a qualidade do circuito sintetizado. Isto foi avaliado através da análise da área e da velocidade de operação dos circuitos gerados a partir dos diferentes estilos de descrição discutidos. Para o caso estudado (processo de abertura/fechamento de conexão do protocolo ABRACADABRA) notou-se que de acordo com o estilo adotado, a redução na área de circuito sintetizado chegou a 49% e o aumento da velocidade de operação chegou a 23%. O esquema lógico desta parte do protocolo ABRACADABRA, considerando o melhor estilo de descrição, foi apresentado.

Uma comparação entre a implementação de protocolos utilizando a técnica de Síntese de Alto Nível com biblioteca de células padrão e a técnica de dispositivos programáveis, conhecida como PLD, é apresentada. Esta comparação mostrou que a velocidade de operação do circuito sintetizado utilizando a técnica de HLS com *standard cells* foi 4,5 vezes maior do que a velocidade de operação do circuito implementado utilizando PLDs. Por outro lado deve ser notado que a implementação em PLDs é imediata e o sistema pode ser reprogramado sem a necessidade de uma nova rodada de fabricação, como no caso das *standard cells*.

Algumas recomendações de projeto como, as restrições no uso de variáveis inteiras e sinais e o problema da implementação da estrutura responsável pelo armazenamento de dados, foram examinadas. Com a limitação das variáveis inteiras em VHDL, para o caso estudado, obteve-se uma redução na área de circuito sintetizado de aproximadamente 54%. Para o caso de armazenamento de dados, observou-se que a implementação desta estrutura deve ser efetuada utilizando-se uma memória externa conectada à lógica de controle. Para o caso abordado, obteve-se uma redução na área de circuito sintetizado de aproximadamente 87%.

## Referências


[1] Budkowski, S., Dembinski, P., "An Introduction to Estelle: A Specification Language for Distributed Systems", *Computer Network and ISDN System, North-Holland*, vol. 14, pp. 3–23, 1987.

[2] ISO/TC97/SC21/WG16-1, "Estelle: a Formal Description Technique based on a Extended State Transiction Model", *DP9074*, 1987.

[3] Ashenden, P. J., *The VHDL Cookbook.* Computer Science Dept. of University of Adelaide, 1990.

[4] IEEE Std 1076-1987, *IEEE Standard VHDL Language Reference Manual*, março 1988.

[5] Gajski, Daniel D. et al., *High-Level Synthesis - Introduction to Chip and System Design.* University of California, Irvine: Kluver Academic Publishers, 1992.

[6] King, P., "Formalization of Protocol Engineering Concepts", *IEEE transactions on Computers*, vol. 40, no. 4, pp. 387–403, Abr 1991.

[7] Vissers, C. et al., "Formal Description Techniques", *Procceding of IEEE*, vol. 71, no. 12, pp. 1356–1364, Dez 1983.

[8] Bochmann, G., Sunshine, C., "Formal Methods in Communication Protocol Design", *IEEE transactions on Communications*, vol. 28, no. 4, pp. 624–631, Abr 1980.

[9] Lima, R.N.B., Carli, E., Pirmez, L., Pedroza, A. e Mesquita, A. C., "Design for Hardware Implementation of Communication Protocols from a Formal Description Technique", *Relatório Técnico GTA 99-01, submetido a Computer Communications Magazine*, Jan 1999.

[10] Synopsys, Inc., *Behavioral Compiler User Guide Online Documentation, v1998.02*, 1998.

[11] Synopsys, Inc., *Synopsys Online Documentation, v1998.02*, 1998.

[12] Cadence Design Systems, Inc., *Cadence Online Library*, 1994.

[13] Carli, E., "Projeto de Protocolos visando a Implementacao em Hardware: aplicacao ao Protocolo IPv6 no Ambito da Abordagem IP over ATM", Tese de Mestrado, Universidade Federal do Rio de Janeiro, PEE/COPPE/UFRJ, Rio de Janeiro, Brasil, 1998.

[14] Lima, R.N.B., "Projeto de Protocolos visando a Implementacao em Hardware: Aplicacao ao Caso do Protocolo AAL5 para Redes ATM", Tese de Mestrado, Universidade Federal do Rio de Janeiro, PEE/COPPE/UFRJ, Rio de Janeiro, Brasil, 1998.

[15] Pirmez, L., *Uma Metodologia para a Implementação de Sistemas Distribuídos em Hardware a partir de uma Descrição Formal.* Tese de Doutorado, Universidade Federal do Rio de Janeiro, PEE/COPPE/UFRJ, Rio de Janeiro, Brasil, 1996.

[16] Pirmez, L. and Rahmouni, M. and Kission, P. and Pedroza, A. and Mesquita, A. C. and Jerraya, A. A., "Analysis of Different Protocol Description Styles in VHDL to High-Level Synthesis", in *EURODAC'96*, Geneve, 1996.

[17] Pirmez, L. and Pedroza, A. and Mesquita, A. C., "High Level Synthesis of Protocols Described by a Formal Description Technique", in *IX IFIP International Conference on VLSI'97*, Gramado, RS, Brazil, 1997.

[18] Soares da Silva, G., "Geração Automática de Sequências de Testes para Protocolos de Comunicação", Tese de Mestrado, Universidade Federal do Rio de Janeiro, PEE/COPPE/UFRJ, Rio de Janeiro, Brasil, 1992.

[19] Resende Loureiro, F., "Um Simulador para a Linguagem Estelle", Tese de Mestrado, Universidade Federal do Rio de Janeiro, PEE/COPPE/UFRJ, Rio de Janeiro, Brasil, 1995.

[20] Altera, Corp., *Data Book*, 1996.

[21] Altera, Corp., *Max + PlusII Getting Started*, 1997.