SEQ2VHDL - Um Ambiente para Projeto de Sistemas Digitais

Leandro de Oliveira Tancredo[1], Alexandre César Rodrigues da Silva [2] e Norian Marranghello [3]

Resumo *— Este trabalho apresenta uma nova ferramenta de síntese para projetos de sistemas digitais denominada SEQ2VHDL. A partir da descrição em diagrama de transição de estados de uma máquina de estados finitos, representada no modelo de Mealy, é gerado uma descrição otimizada do sistema na linguagem de hardware VHDL. Elimina-se dessa forma a tarefa árdua com detalhes d projeto. A SEQ2VHDL foi comparada com duas outras ferramentas disponíveis comercialmente. Foram projetados diversos chip-set de códigos de transmissão digital utilizados no setor de telecomunicações. Os resultados comprovaram o desempenho satisfatório com relação ao custo de implementação, ao tempo de execução e uso de memória. A ferramenta trará grandes contribuições nas áreas educacionais da engenharia elétrica e computação, possibilitando além da criação de sistemas autômatos, o aprendizado da linguagem VHDL, pois trata-se de uma ferramenta de domínio público ao contrário das ferramentas comerciais.*

Palavras Chave: FPGA, VHDL, FSM, HDL, Síntese Lógica).

INTRODUÇÃO

Este artigo tem como proposta apresentar uma ferramenta de síntese de alto nível ou HLS (abreviatura do inglês *high-level synthesis*) denominada SEQ2VHDL (SEQUENCIAL TO VHDL) e, a partir, dela implementar vários circuitos integrados de códigos de linhas utilizados em transmissão de dados, no setor de telecomunicações, originados a partir do diagrama de estados, utilizando a linguagem VHDL [2] [8][9]. Criou-se, dessa forma, um ambiente compatível para a síntese de sistemas digitais que veio facilitar a tarefa do projetista, eliminando a descrição detalhada como no exemplo de uma metodologia de projetos digitais seqüenciais apresentada em [10].

Atualmente, dispõe-se de dois importantes tradutores disponíveis comercialmente que interpretam uma máquina de estado finito, descrita através de seu diagrama de estado. Estas ferramentas são denominadas: Statecad [11] da Actel e Active-HDL [4], da Cypress.

Esses *softwares* destacam-se pelo uso do diagrama de estados para descrever o comportamento de uma máquina de estados finitos, traduzindo-as para a sua representação em código VHDL. Nenhum processo de otimização é efetuado nesta etapa, deixando a cargo das ferramentas de síntese, ou seja, o MAX + Plus II [1] ou XILINX [12]. Estas ferramentas apresentam entradas gráficas que possibilitam expressar idéias em um modo natural, bem próximo da linguagem humana.

Como a descrição em VHDL independe da tecnologia, as ferramentas de síntese comerciais apropriadas levariam para o nível de implementação de estado no domínio físico.

Conforme GENOE [7] existem aproximadamente 6000 licenças para sintetizadores lógicos no mercado, entretanto, há uma carência de ferramentas que interpretem os níveis de abstração altos. Um tradutor de linguagem, foi proposto por BONATTI [3] e implementado em um pacote de domínio público, denominado Stoht. FUHRER et alli [6] descrevem uma solução para minimização de estados de uma FSM para circuitos lógicos.

Muitas outras ferramentas de síntese de Alto Nível são descritas na literatura, comprovando a importância deste tipo de abordagem.

Na próxima seção, descreve-se os programas TABELA e o SEQ2VHDL.

AMBIENTE TABELA E SEQ2VHDL

Descreve-se nesta seção um método sistemático para projetos que a partir de uma descrição textual de uma máquina de estados finitos síncronas, origina-se uma segunda descrição, onde os elementos de memória são claramente identificáveis (*flip-flops*) e, finalmente, uma última descrição em VHDL na qual está presente uma entrada particular, denominada relógio.

[1] Leandro de Oliveira Tancredo, DEE – FEIS - UNESP, Av. Brasil Norte, 364, Bairro: Zona Norte,CEP: 15385-000, Ilha Solteira - SP, Brasil, tancredo@dee.feis.unesp.br

[2] Alexandre César Rodrigues da Silva, DEE – FEIS – UNESP, Av. Brasil Norte, 364, Bairro: Zona Norte,CEP: 15385-000, Ilha Solteira - SP, Brasil, acrsilva@dee.feis.unesp.br

[3]Norian Marranghello, DCCE –IBILCE -UNESP, Rua Cristovão Colombo, 2265 - Jardim Nazareth 15054-000, São José do Rio Preto, Brasil, norian@dcce.ibilce.unesp.br

Programa TABELA

O programa TABELA [5] resultado de trabalho de tese apresentado na UNICAMP possui as seguintes etapas do projeto conforme a Figura 1.

Na metodologia empregada parte-se do nível RTL do domínio comportamental e chega-se ao nível lógico do mesmo domínio, ou seja, parte-se do diagrama de estados e chega-se nas correspondentes funções booleanas e memórias em suas formas mínimas. Vários algoritmos de minimização de funções booleanas poderiam ser empregados nesta etapa do projeto, optou-se pelo método Quine-McCluskey [5].

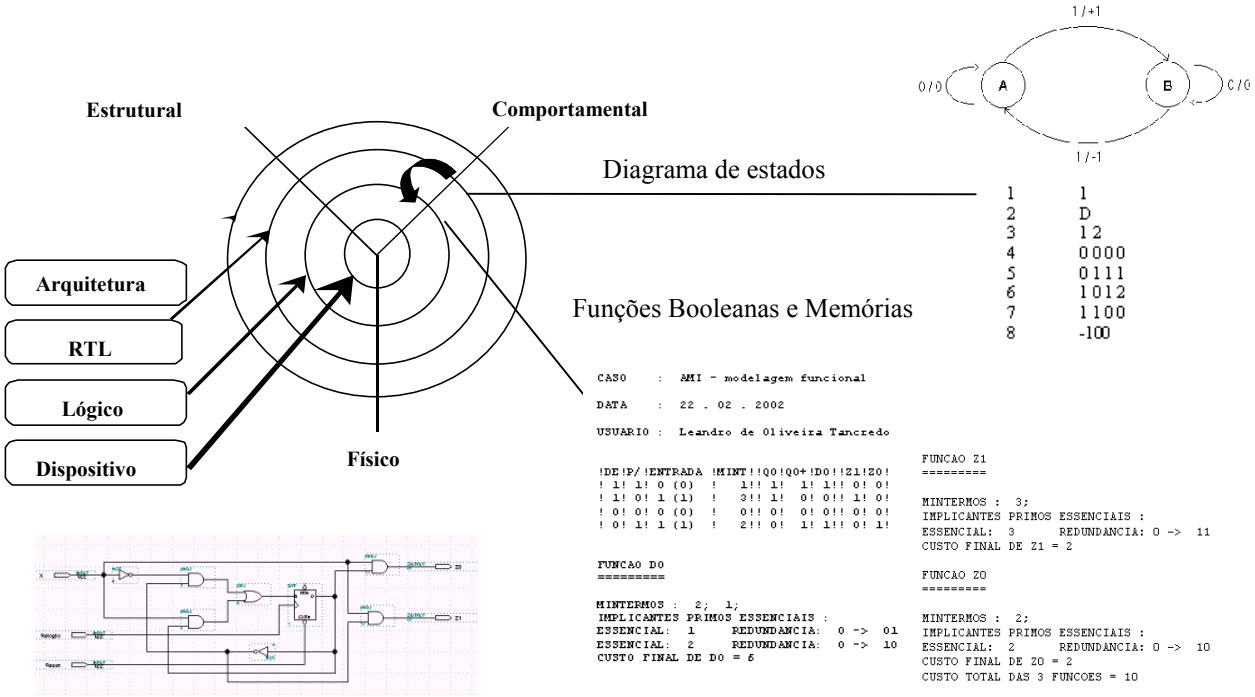


FIGURA. 1
DIAGRAMA Y APRESENTANDO A FERRAMENTA TABELA.

Programa SEQ2VHDL

Nesta seção descreve-se a ferramenta SEQ2VHDL, que recebe como entrada as funções booleanas (a saída gerada pelo programa TABELA) e cria um modelo funcional do circuito descrito na linguagem de descrição VHDL, conforme a figura 2 que apresenta o diagrama Y.

A listagem abaixo descreve o arquivo AMI.VHD gerado pelo programa SEQ2VHDL:

```
-- Ferramenta de Síntese Lógica
-- A ferramenta tem por objetivo transformar a descrição -
-- de uma máquina de estados finitos
-- sintetizada pelo programa TABELA para o seu modelo
-- RTL descrito em VHDL.
-- Programa: SEQ2VHDL.EXE
-- Programador: Leandro de Oliveira Tancredo
-- Versão: 9.0 de 3 de março de 2002
ENTITY AMI IS
    PORT(
-- CLK e CLR estão presentes em todos os modelos
    CLK, CLR : IN BIT;
-- Os parâmetros seguintes são definidos de acordo
-- com a descrição contida no programa tabela.
-- Xn representam as variáveis de entrada
-- Qn representam as variáveis de estado
-- Zn representam as funções de saída
    X0    : IN BIT;
    Q0    : OUT BIT;
    Z0, Z1 : OUT BIT );
END AMI;

-- RTL e a designação para todas as arquiteturas
ARCHITECTURE RTL OF AMI IS
-- VEn são sinais auxiliares que assumem os mesmos
-- valores das variáveis de estado. Eles são utilizados para
-- permitir
-- um melhor modelamento do sistema
SIGNAL  VE0: BIT;
--Jn, Kn e Dn representam as funções de controle e são
-- definidas
```

```
-- no arquivo gerado pelo Programa TABELA
SIGNAL D0 : BIT;
 BEGIN
-- Nesta parte do modelo tem-se a descrição de cada um dos Flip-flops.
--Deve-se observar que os sinais auxiliares são utilizados,
-- sendo que no final de cada processo seus valores são
--transferidos para as variáveis de estados
-- Importante salientar que em relação ao software, existem duas procedures,
-- uma que implementa o flip-flop D e outra que
--implementa o JK e estas procedures recebem os índices
-- que representam o respectivo elemento de memória.
--Tais índices estão definidos no arquivo gerado pelo
 --programa TABELA
-- Inferindo flip-flop tipo D
PROCESS(CLK, CLR)

BEGIN
   IF CLR = '0' THEN
     VE0 <= '0';
   ELSIF CLK'EVENT and CLK = '1' THEN
     VE0 <= D0;
   END IF;
   Q0 <= VE0;
END PROCESS;
-- Processos que implementam as funções combinacionais de controle
-- dos Elementos de memória e de Saídas.
 D0 <=  ( NOT(X0) AND (VE0)) OR ((X0) AND NOT(VE0));
 Z1 <=  ( (X0) AND (VE0));
 Z0 <=  ( (X0) AND NOT(VE0));
END RTL;
```

Pode-se notar que na entidade do modelo apresentado foram definidos os portos de entrada e saída e os sinais de sincronismo (CLK e CLR). O porto denominado Q_0 não é necessário para o perfeito funcionamento do sistema. Este foi inserido para permitir que o projetista observe, através de simulação, os estados pelos quais a máquina está transitando. Cabe salientar que o TAB2VHDL foi desenvolvido para ser empregado no ensino de sistemas digitais. Se a avaliação das transições de estado não forem necessárias, as variáveis de estado podem ser omitidas.

São definidos sinais auxiliares denominados V_{En} cujo objetivo é evitar que se gerem vários *drivers* para um mesmo sinal, o que exigiria o desenvolvimento de uma função de resolução. Esta função define qual será o valor do sinal em caso de conflitos.

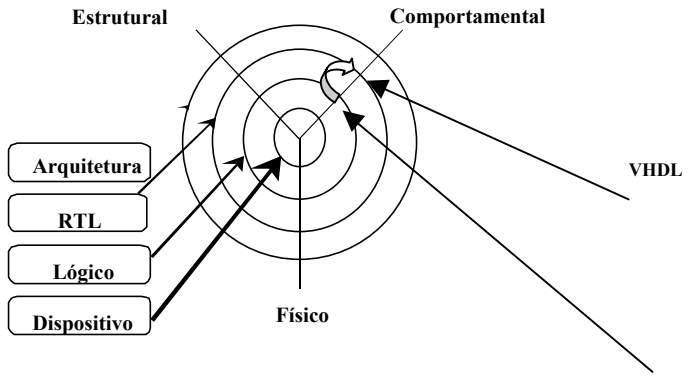
É criado um processo para cada um dos *flip-flops* utilizados. Dois tipos de processos foram criados, um para modelar *flip-flop* D, sensível a transição de subida, e o outro para modelar *flip-flop* JK, sensível a transição de subida. Vários outros modelos de *flip-flop* poderiam ser definidos na ferramenta SEQ2VHDL.

Por fim, são definidas as funções de controle dos elementos de memória, D_0, e de saída, Z_1 e Z_2.

Estas funções foram obtidas do arquivo gerado pelo programa TABELA.

Na simulação apresentada na figura 2, verifica-se o perfeito funcionamento do código AMI, cujo modelo foi gerado pelo programa SEQ2VHDL.

Vários códigos de linhas foram projetados, empregando três diferentes ambientes de projeto. Na próxima seção, ao implementar os códigos, compara-se o projeto gerado pelo SEQ2VHDL com outras ferramentas.




```
-- PROJETO DE TESE DE MESTRADO - Ferramenta de Síntese Lógica
-- A ferramenta tem por objetivo transformar a descrição de uma máquina
de estados finitos
-- sintetizada pelo programa TABELA para o seu modelo RTL descrito em
VHDL.
-- Programa: SEQ2VHDL.EXE
-- Programador: Leandro de Oliveira Tancredo
-- Versão: 9.0 de 3 de março de 2002
ENTITY AMI IS
     PORT(
-- CLK e CLR estão presentes em todos os modelos
     CLK, CLR : IN BIT;

-- Os parâmetros seguintes são definidos de acordo
-- com a descrição contida no programa tabela.
-- Xn representam as variáveis de entrada
```

Funções Booleanas

```
!DE!P/ !ENTRADA  !MINT!!Q0!Q0+!D0!!Z1!Z0!
! 1! 1! 0 (0)   !   1!! 1!  1! 1!! 0! 0!
! 1! 0! 1 (1)   !   3!! 1!  0! 0!! 1! 0!
! 0! 0! 0 (0)   !   0!! 0!  0! 0!! 0! 0!
! 0! 1! 1 (1)   !   2!! 0!  1! 1!! 0! 1!


FUNCAO D0
=========

MINTERMOS :  2;  1;
IMPLICANTES PRIMOS ESSENCIAIS :
ESSENCIAL:  1     REDUNDANCIA:  0 ->  01
ESSENCIAL:  2     REDUNDANCIA:  0 ->  10
CUSTO FINAL DE D0 = 6
```

FIGURA. 2
DIAGRAMA Y QUE REPRESENTA A FERRAMENTA SEQ2VHDL.

ESTUDO DE CASOS

Escolheu-se para estudo de casos a implementação de alguns chip-sets de códigos de transmissão digitais, utilizando-se duas ferramentas e o ambiente TABELA + SEQ2VHDL. A maioria desses exemplos são padrões, todavia, com o passar do tempo surgem novos serviços, que obrigam a adaptação do código à nova tecnologia utilizada.

Logo, os projetos devem permitir uma maior flexibilidade de adaptação, sendo que uma solução adequada seria a implementação em *software*. Contrariamente, por estes códigos estarem localizados nas camadas inferiores da OSI/ISO, e, por isso, necessita-se de velocidade nas suas aplicações em tempo real, adotando-se a implementação em *hardware*.

O desenvolvimento de projetos eletrônicos tem impulsionado o setor de telecomunicações nos últimos anos, permitindo, assim, a criação de vários serviços digitais, como exemplo, temos várias tecnologias RDSI, HDSL, ADSL, VDSL e várias outras que estão surgindo no mercado. A utilização de componentes lógicos programáveis, como FPGAs para implementação de hardware, têm proporcionado uma melhoria no desempenho do circuito, ou seja, a redução de custos e uma integração física maior em relação aos projetos convencionais.

Os códigos de linha HDB3, HDB1, 2BQ1, 3B4B, MLT-3, AMI e algumas técnicas alternativas de detecção de sincronismo foram projetados utilizando-se as ferramentas comerciais, aqui denominadas simplesmente de SC e AC, e os resultados foram comparados com os obtidos pelo ambiente TAB2VHDL, conforme Tabela I.

TABELA I

RESULTADO COMPARATIVO DA FERRAMENTA IMPLEMENTADA COM AS COMERCIAIS

	HDB3			HDB1			2BQ1			3B4B		
	SC	AC	TAB	SC	AC	TAB	SC	AC	TAB	SC	AC	TAB
LC	7	15	7	6	4	4	7	5	5	12	7	7
% UTIL	21	46	21	18	12	12	21	15	15	37	21	21
FAN-IN	34	101	39	33	15	17	34	18	10	88	52	32
CUSTO	99	301	50	35	18	11	43	22	9	181	183	44
TEMPO (s)	11	6	3	2	3	6	3	8	3	8	10	2
MEMÓRIA (KB)	4,492	4,162	2,363	3,02	3,095	2,925	3,1	2,366	2,417	2,594	2,757	2,216

	MLT3			AMI			S10010			CCS		
	SC	AC	TAB	SC	AC	TAB	SC	AC	TAB	SC	AC	TAB
LC	6	4	4	3	3	3	8	4	4	8	4	4
% UTIL	18	12	12	9	4	9	25	12	12	25	12	12
FAN-IN	24	13	14	8	6	8	37	19	21	40	19	22
CUSTO	24	13	14	1	1	2	15	24	27	18	28	27
TEMPO (s)	3	2	2	1	4	2	1	0	1	0	4	1
MEMÓRIA (KB)	2,963	3,025	2,912	3,032	3,396	2,324	2,63	3,432	3,136	3,333	2,994	3,438

	KEMN			TRANSU		
	SC	AC	TAB	SC	AC	TAB
LC	14	5	5	9	4	4
% UTIL	43	15	15	28	12	12
FAN-IN	64	29	33	44	17	22
CUSTO	29	80	63	23	21	33
TEMPO (s)	1	6	1	1	6	1
MEMÓRIA (KB)	3,191	3,157	2,941	3,352	2,934	3,065

Pode-se concluir através da análise das tabelas apresentadas que o ambiente TABELA em conjunto com SEQ2VHDL obteve um desempenho notável comparado com as ferramentas avaliadas.

Outro resultado importante, foi obtido ao comparar o custo do código gerado pelo TAB2VHDL ao utilizar como elemento de memória flip-flop JK (TAB JK) ou D (TAB D), observa-se conforme Tabela II que em algumas situações o JK é a melhor solução como no 2BQ1, 3B4B e MLT3, e que em outras o D foi a melhor solução como na AMI, S10010 E CCS.

TABELA II

COMPARAÇÃO UTILIZANDO ELEMENTO DE MEMÓRIA JK E D NO TAB2VHDL

	HDB3		HDB1		2BQ1	
	TAB D	TAB JK	TAB D	TAB JK	TAB D	TAB JK
CUSTO	50	50	11	11	9	6

	3B4B		MLT3		AMI	
	TAB D	TAB JK	TAB D	TAB JK	TAB D	TAB JK
CUSTO	44	43	14	7	2	5

	S10010		CCS	
	TAB D	TAB JK	TAB D	TAB JK
CUSTO	24	29	27	28

CONCLUSÃO

Conclui-se que com o surgimento de novos serviços, que deverão em um curto espaço de tempo, serem projetados e adaptados, impulsionados pelos grandes resultados apresentados pelo desenvolvimento das tecnologias digitais, como exemplo, o HDSL que a partir de um único cabo de par trançado se compartilha 30 canais de voz em um link de 2 Mbps, substituindo um único canal de voz analógico como nas tecnologias passadas. Portanto, a necessidade de novas ferramentas de síntese como o TAB2VHDL, para otimizar *chip-sets* envolvidos nestes tipos de projetos, passa a ter fundamental importância.

Assim, o estudo mostrou que é impossível ter uma única ferramenta que substitua todas as outras, entretanto, se existisse, ficaria muito complicado trabalhar, tendo como causa o alto grau de complexidade que envolveria o projeto.

O projeto de sistemas digitais é uma área de pesquisa em constante evolução. O aumento contínuo da complexidade dos sistemas, cria a necessidade de se rever as técnicas de síntese e otimização já desenvolvidas.

O emprego cada vez maior das FPGAs tem motivado o desenvolvimento de ferramentas de síntese automatizada.

AGRADECIMENTOS

Os autores agradecem ao PPGEE da FEIS – UNESP, UNIFEV, FAI-JALES, à CAPES e à P.I. Componentes.

REFERÊNCIAS

[1] ALTERA. Altera Digital Library. Disponível em: <www.altera.com> Acesso em: 09/12/2001.

[2] ARMSTRONG, James R. "Chip-level modeling with VHDL". *10 ed. New Jersey: Prentice Hall*, 1989.148p.

[3] BONATTI, Ivanil, S.. "Stoht – An SDL-to-Hardware Translator." *Asia Pacific Design Automation Conference*, Mukuhari Japan, 33-36pp. 1995

[4] CYPRESS. "An Introduction to Active-HDL". FSM.pdf. São José, 28 de Janeiro de 2002. 1 arquivo (97 Kbytes). Disquete 3 ½. Acrobat Reader 5.0

[5] SILVA, A. C. R., "Contribuição à minimização e simulação de circuitos lógicos", 1989. 113 f. Dissertação (Mestrado em Engenharia Elétrica) – UNICAMP, Campi-nas.

[6] FUHRER, Robert, M. et alli. "OPTIMISTA: State Minimization of Asynchro-nous FSMs for Optimum Output Logic". *NSF Award* CCR-97-34803.DCS – Columbia University, New York .

[7] GENOE, Mark et alli. "On the use of VHDL-based behavioral synthesis for telecom ASIC design". *Eighth International Symposium On System Synthesis*, 6 pp. 1995

[8] PERRY ,D. L.. "VHDL". *4.ed. New York: McGraw-Hill Inc*, 1991. 459p.

[9] SHAHDAD, Moe. "An overview of VHDL Language e Technology". *IEEE 23rd Design Automation Conference*, Paper 17.1, 320-326p, 1986.

[10] TANCREDO, Leandro de O. Tancredo, "Métodos e Ferramentas de Projetos Digitais Seqüenciais e Programa TABELA". 18 jan 2002. 82 f. Estudos Especiais I - DEE FEIS UNESP. Ilha Solteira.

[11] XILINX. "StateCAD User´s Guide". Xilinx Inc, 2000.

[12] XILINX. "Xilinx Programmable Logic Book". Disponível em: <www.xilinx.com > Acesso em 09/12/2002