



FUNDAÇÃO DE ENSINO “EURÍPIDES SOARES DA ROCHA”  
CENTRO UNIVERSITÁRIO “EURÍPIDES DE MARÍLIA” – UNIVEM  
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

# Guia para Utilizar as Ferramentas Project Manager e GXSLOAD

DENISON MENEZES

DIEGO DA SILVA MARTINS

ILDEBERTO DE GENOVA BUGATTI  
(ORIENTADOR)

MARÍLIA

2008



# Guia para Utilizar as Ferramentas *Project Manager e GXSLoad*

---

## 1 Sumário

<b>2</b>	<b>Considerações Iniciais.....</b>	<b>220</b>
2.1	Notas dos Autores.....	220
2.2	Necessário .....	220
2.2.1	Conhecimento .....	220
2.2.2	Ferramentas .....	220
2.2.3	Hardware.....	220
<b>3</b>	<b>FPGA.....</b>	<b>221</b>
3.1	Estrutura Interna de um FPGA .....	221
3.2	Conceitos sobre Roteamento.....	222
3.3	FPGA utilizado .....	222
<b>4</b>	<b>Project Manager.....</b>	<b>224</b>
4.1	Criando um novo Projeto .....	224
4.2	Schematic Editor .....	225
4.2.1	Adicionando componentes no diagrama e interligando eles .....	226
4.2.2	Criando Macros .....	227
4.2.3	Buffers e PADS.....	229
4.2.4	Criando o Netlist.....	231
4.3	Functional Simulation.....	232
4.4	Criando o Bitstream .....	234
<b>5</b>	<b>GXSLoad .....</b>	<b>237</b>
<b>6</b>	<b>Bibliografia .....</b>	<b>240</b>



## 2 Considerações Iniciais

### 2.1 Notas dos Autores

Este guia se dispõe em ensinar conceitos básicos para a utilização da FPGA<sup>1</sup> na placa XS40.

Para tal será utilizado a ferramenta da XILINX Project Manager, para criação dos diagramas esquemáticos dos circuitos lógicos, teste e geração do arquivo Bitstream. E a XSTOOLS para transferência deste arquivo para FPGA.

### 2.2 Necessário

#### 2.2.1 Conhecimento

Conhecimento em lógica digital.

#### 2.2.2 Ferramentas

Project Manager.

XSTOOLS.

#### 2.2.3 Hardware

Placa de desenvolvimento XS40 versão 1.2 do fabricante Xess Corporation.

FPGA XC4010XLPC84 integrante da família XC4000XL do fabricante Xilinx.

---

<sup>1</sup> Field Programmable Gate Array.



## 3 FPGA

### 3.1 Estrutura Interna de um FPGA

Segundo Ordonez (2003), os FPGAs são circuitos programáveis formados por conjuntos de células lógicas em arranjo matricial. Cada célula contém capacidade computacional para implementar funções lógicas e realizar roteamento para comunicação entre elas. O roteamento e as funções lógicas das células são configuráveis via *software*.

De acordo com Martins (2003), a estrutura básica de um FPGA (Figura 1) é composta pelos seguintes elementos:

- 1) **CLB (Configurable Logic Block)**: unidade lógica configurável.
- 2) **IOB (In/Out Block)**: unidade de entrada e saída.
- 3) **SB (Switch Box)**: unidade de conexão entre os diversos CLBs.
- 4) **Canais de roteamento**: interligam as unidades de conexão para formar a rede de interconexão programável.

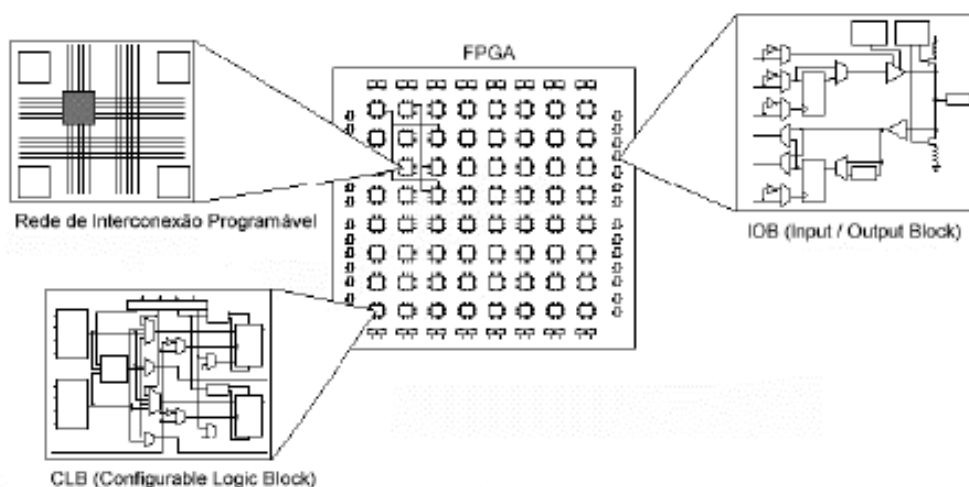


Figura 1 – Elementos básicos de um FPGA

Ordonez (2003) cita as seguintes configurações para a arquitetura interna de um FPGA:

- 1) **Matriz simétrica**: possui canais de roteamento verticais e horizontais com grande flexibilidade.
- 2) **PLD hierárquico**: matriz de blocos lógicos interligados e que podem ser agrupados entre si.
- 3) **Row-based**: blocos lógicos em disposição horizontal com área dedicada para o roteamento.



**4) Sea of gates:** blocos lógicos de complexidade pequena dispostos em grande número por unidade de área. Não há área dedicada para o roteamento.

### 3.2 Conceitos sobre Roteamento

Segundo Ordonez (2003), a interconexão entre os blocos lógicos através de uma rede de camadas de metal é chamada de roteamento. Fisicamente, transistores controlados por bits de memória (PIP) ou chaves de interconexão (*switch matrix*) são responsáveis pelas conexões físicas entre os blocos lógicos.

Podem ser definidos ainda alguns elementos básicos presentes na malha de roteamento da família XC4000 (Xilinx), os quais seguem abaixo descritos.

**1) Conexões globais:** rede de interconexão entre linhas e colunas ligadas por meio de chaves de interconexão.

**2) Matriz de conexão:** chaves de interconexão para o roteamento entre blocos lógicos através de conexões globais.

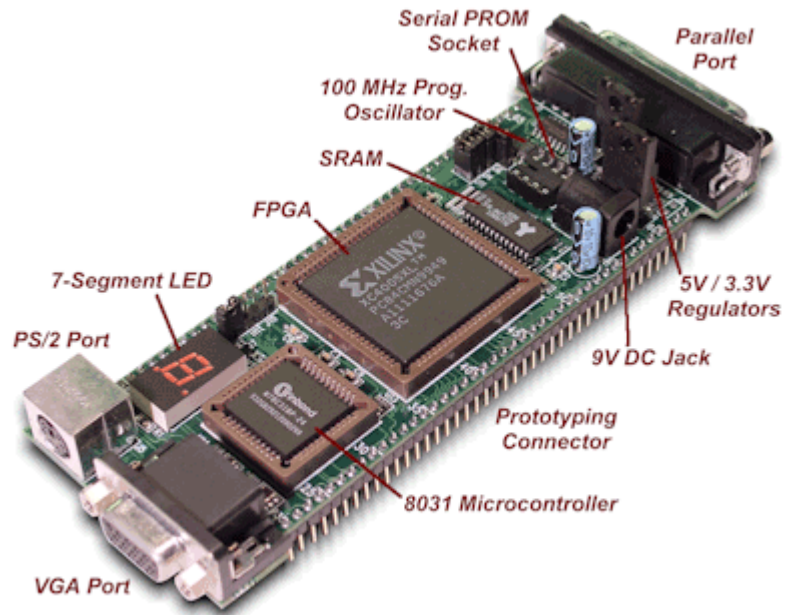
**3) Conexões diretas:** interligação entre CLBs vizinhos que permite menores tempos de atraso, uma vez que não faz uso de recursos globais de roteamento.

**4) Linhas longas:** conexões que interligam sinais longos e percorrem todo o circuito sem passar pelas matrizes de conexão.

### 3.3 FPGA utilizado

Foi utilizado o FPGA XC4010XLPC84 integrante da família XC4000XL do fabricante Xilinx, para a implementação do circuito. Trata-se de um FPGA com encapsulamento PLCC (*Plastic Lead Chip Carrier*) de 84 pinos, sendo 61 deles disponíveis para entrada e saída.

Para auxiliar o teste funcional de validação, foi utilizada a placa de desenvolvimento XS40 versão 1.2 do fabricante Xess Corporation (Figura 2). A placa contém basicamente uma FPGA XC4010XLPC84 funcionando a 3.3 V, um microcontrolador 80C31, uma memória RAM estática de 32 Kbytes, um oscilador de 12 MHz, um display de 7 segmentos e um conector VGA para leitura de sinais de vídeo. A programação do FPGA é feita via PC através da porta paralela.



**Figura 2 – Placa de desenvolvimento XS40**



## 4 Project Manager

O Software *Project Manager* do ambiente Xilinx Foundation F3.1i, Build 3.1.177. é um sistema de desenvolvimento que consiste da integração de um conjunto de ferramentas de softwares e hardware para criar, simular e implementar projetos digitais em FPGA ou CPLD.

O *Project Manager* também é capaz de gerar os bits de configuração para diversos dispositivos reconfiguráveis da Xilinx. Para isso, a ferramenta faz o mapeamento da solução no dispositivo, e gera os bits de configuração (*bitstream*) que podem ser enviados para o dispositivo reconfigurável.

### 4.1 Criando um novo Projeto

Ao abrir o Project Manager surgirá à janela da figura 3 nela estão, as opções de criar um novo projeto ou abrir um já existente.

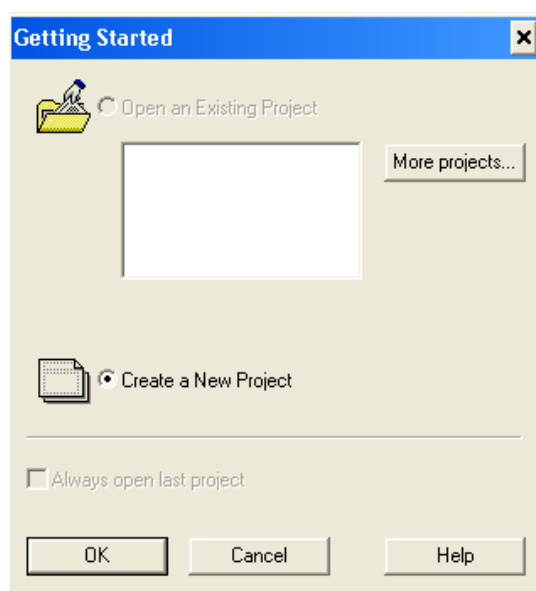
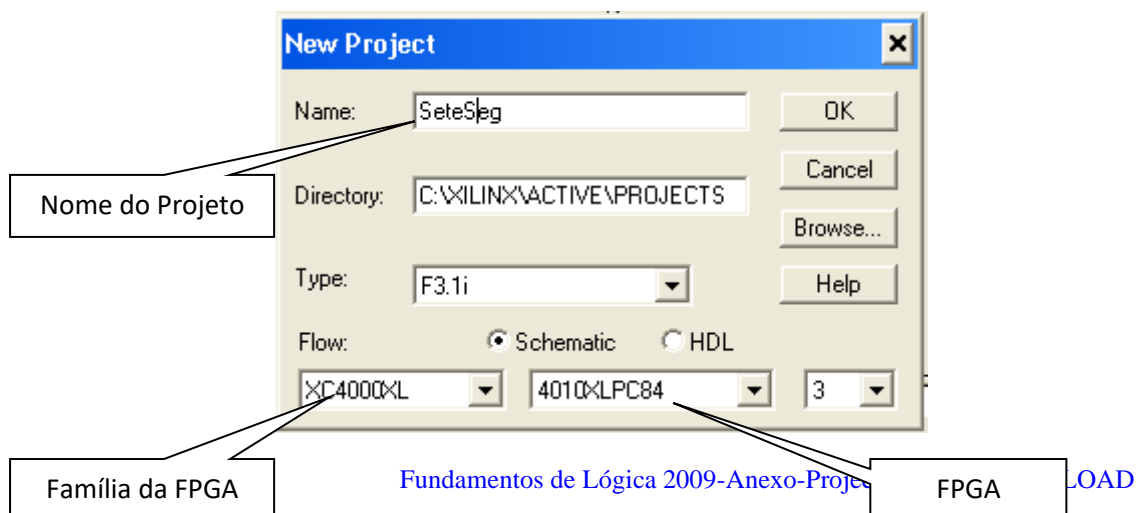


Figura 3 – Criando novo projeto

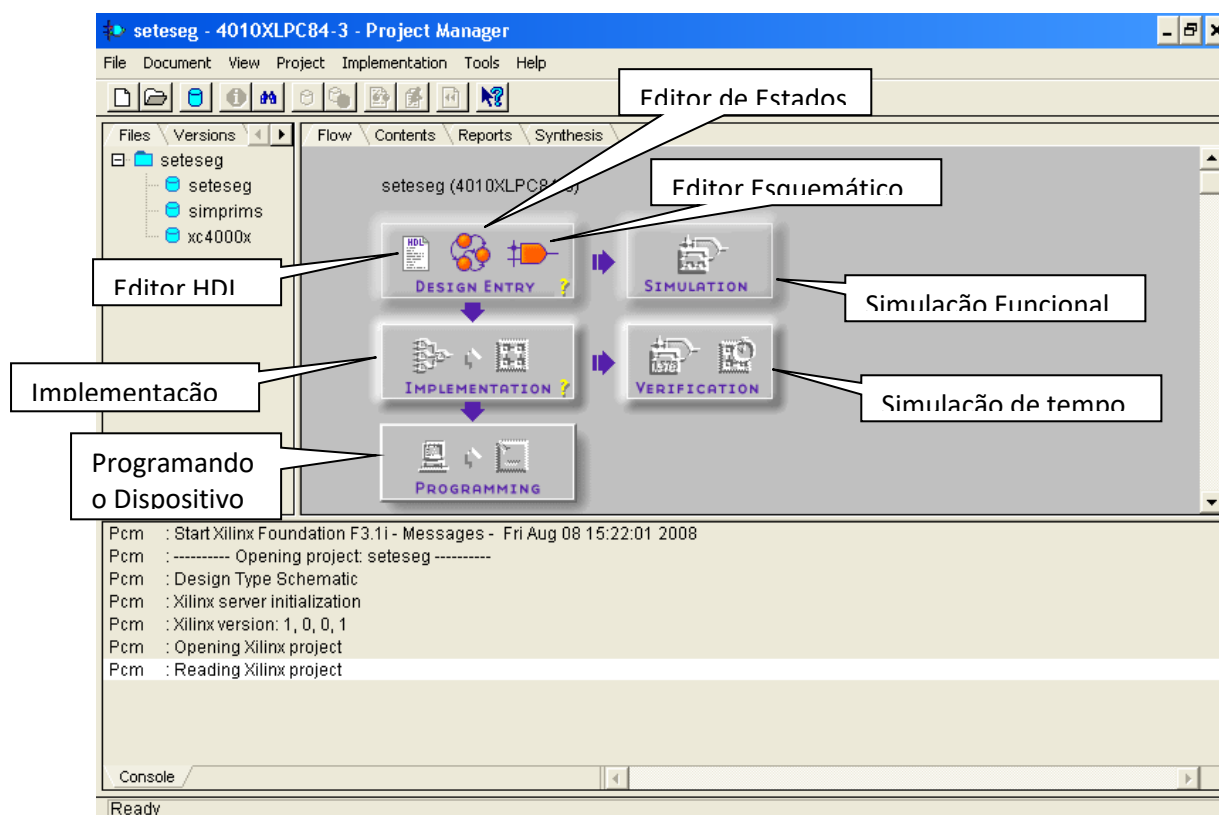
Escolhendo a opção “Create a New Project” ou pelo caminho “File / New Project” aparecerá à janela da figura 4.





**Figura 4 – Criando um novo projeto opções**

Para construir o diagrama esquemático do circuito lógico abra o editor esquemático “Schematic Editor”. Observe a figura5:




**Figura 5 – Ambiente Project Manager da Xilinx**

## 4.2 Schematic Editor

Através deste editor podemos desenvolver de maneira gráfica um circuito lógico. Onde é possível selecionar diversos componentes a partir das bibliotecas, fazer as interligações entre eles, bem como agrupá-los em blocos criando macros para reutilização de funções.

Nosso exemplo é um circuito que apresentará no visor de sete seguimentos o valor de um até nove.

### 4.2.1 Adicionando componentes no diagrama e interligando eles

Para adicionar um componente clique no ícone  (Symbols toolbox). Irá abrir a caixa da Figura 6, digite o nome do componente no campo de busca, e selecione-o caso encontre.



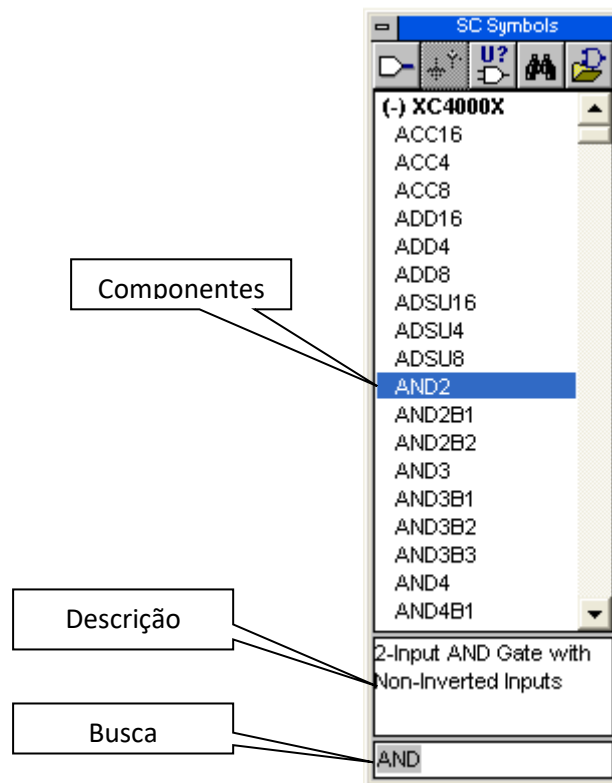




Figura 6 – Caixa de componentes (Symbols toolbox)

Para interligar os componentes selecione o ícone  (*Draw wires*) ou se for o caso o  (*Draw buses*). Como demonstrado na figura 7.

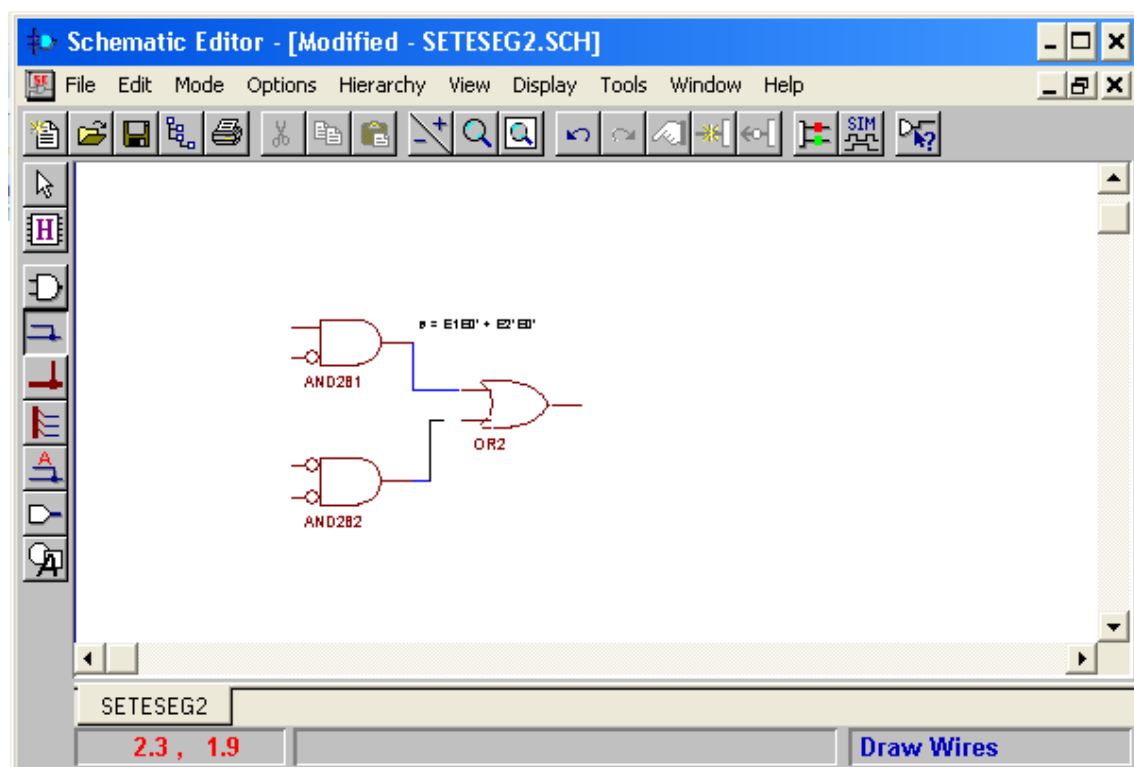



Figura 7 – Interligando Componentes



#### 4.2.2 Criando Macros

Ao construir um circuito pode ser necessário criar macros, para reutilizar algum circuito específico, por exemplo, um contador, ou simplesmente para tornar mais fácil sua compreensão.

Para construir macros utiliza Hierarchy Connector, que é um terminal I/O, mas não é um dispositivo físico, portanto deve ser usado somente para fornecer conexão entre níveis hierárquicos, correspondendo aos pinos no símbolo da macro.

Para adicionar clique no ícone  ou clique com o botão direito do mouse diretamente no “Draw wires” quando o mesmo estiver selecionado e escolhendo a opção “add Terminal”, em seguida será apresentada uma tela para a configuração do terminal a ser criado conforme a figura 8.

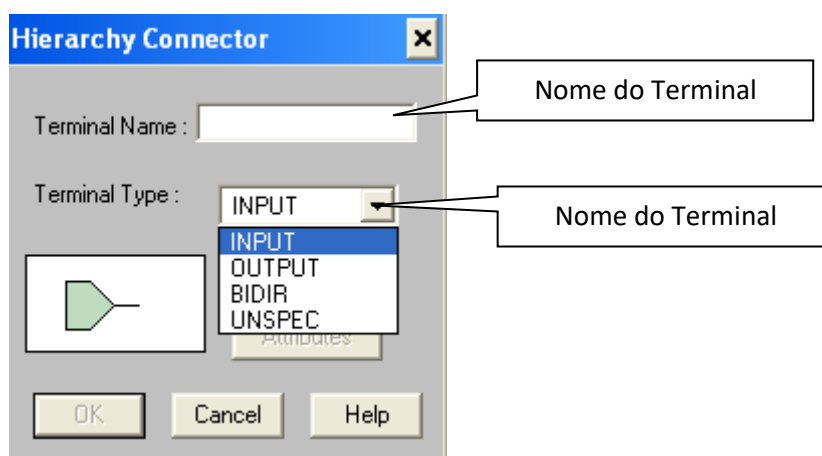


Figura 8 – configuração do Hierarchy Connector

Depois de criado o circuito com os terminais de entrada e saída poderá ser criada a macro, para isso clique na opção da barra de ferramentas “Hierarchy”-> “Create Macro Symbol from Current Sheet...” conforme demonstrado na figura 9, então será apresentada a tela da figura 10.

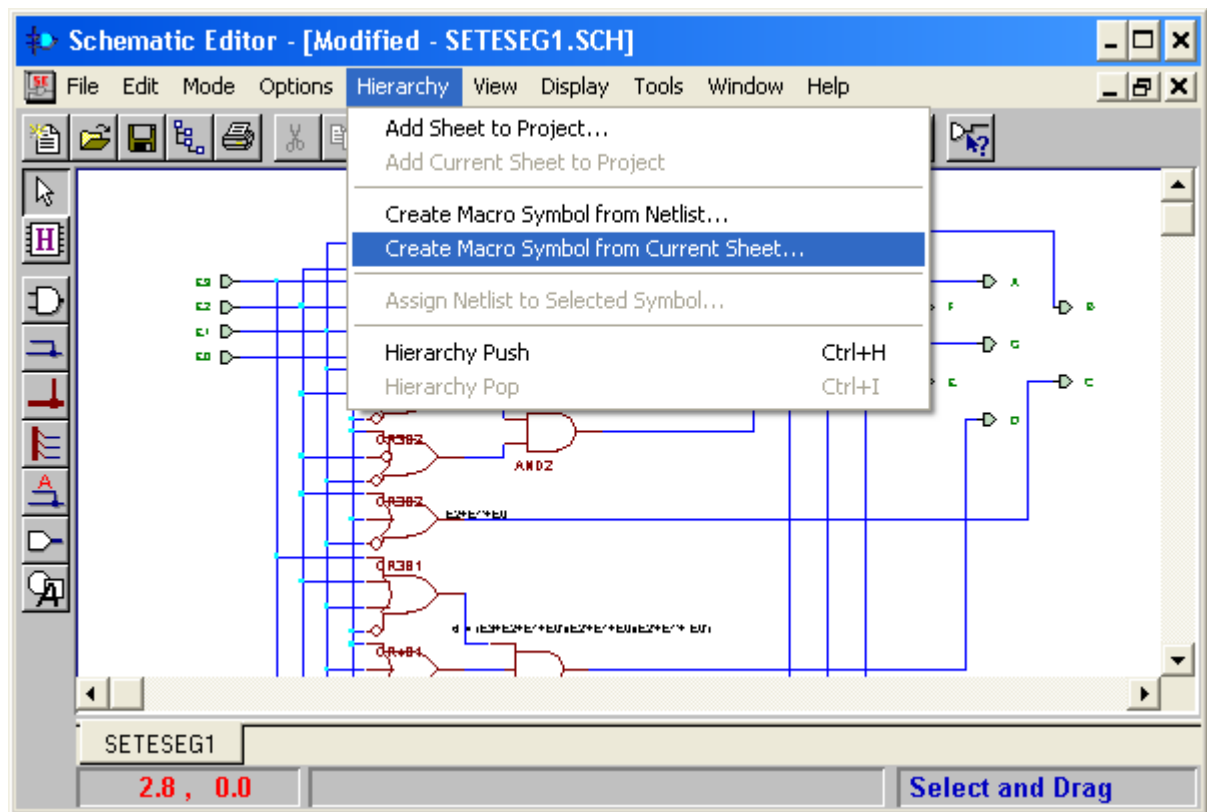


Figura 9 – criando macro

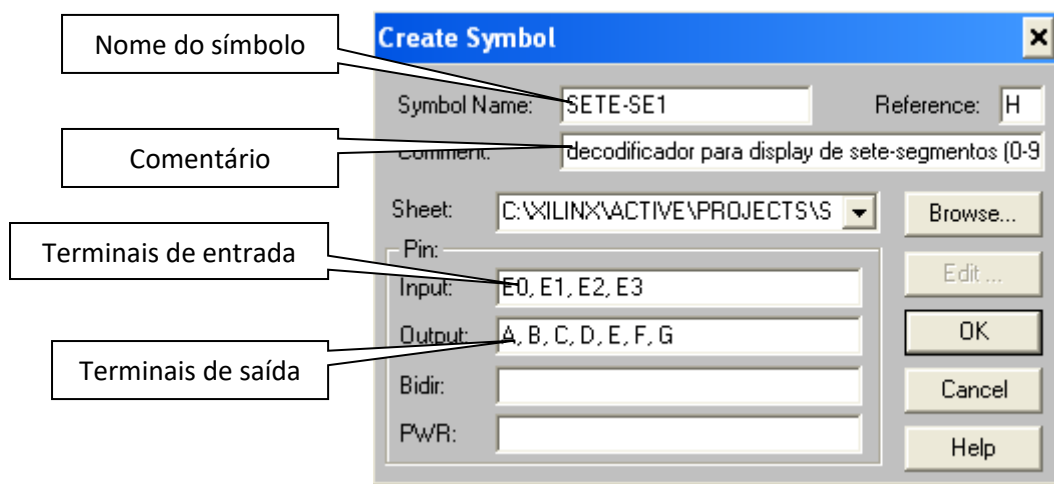


Figura 10 – configuração da macro

Feito os passos anteriores estará disponível a macro criada no Symbols toolbox, o símbolo criado no nosso exemplo está na figura 11.

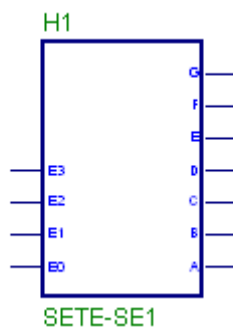


Figura 11 – Macro SETE-SE1

### 4.2.3 Buffers e PADS

#### *BUFFERS*

Buffers são necessários para sinais de entradas e saídas que vão para pinos de seu dispositivo (FPGA). Estão disponíveis no Symbols toolbox, para os buffers de entrada escolhe-se IBF (figura 13) e os de saída OBUF (figura 12). Não esqueça de adicionar BUFFERS no seu esquemático ou seu esquemático não irá compilar.

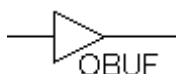


Figura 12



Figura 13

#### *PADS*

Você irá também precisar adicionar pads I/O para buffers de entrada e saída. Estes pads (blocos) representam os pinos atuais no dispositivo XILINX. Os blocos são componentes físicos na biblioteca XILINX e é colocado semelhante a qualquer outro componente. Os nomes dos Pads são IPAD (entrada), OPAD (saída) e IOPAD(bidirecional). TODOS OS DISPOSITIVOS DEVEM SER REPRESENTADOS COM UM DESTES PADS! Os Pads deverão ser dados um nome e possivelmente um número de pino (localização dos pinos). Estes podem ser feitos dando um clique duplo no pad ou pressionando o botão direito do mouse na opção "Symbol Properties..." conforme demonstrado na figura 14, para aparecer a janela Symbol Properties.

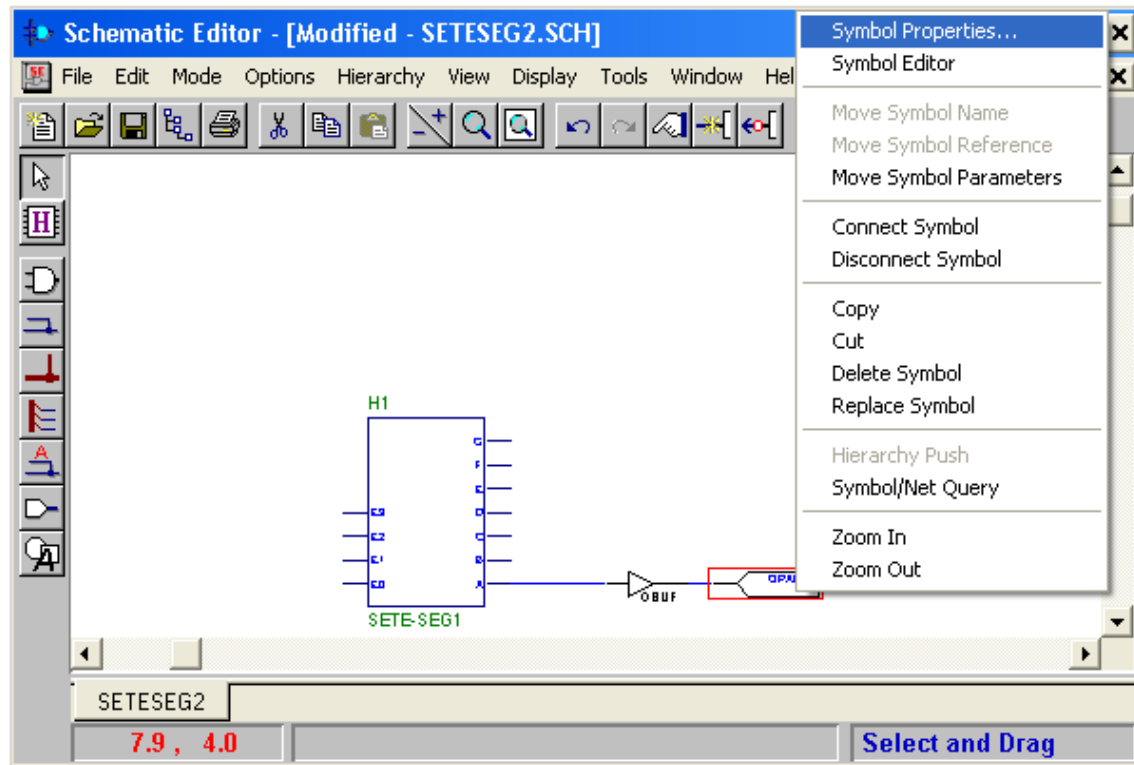


Figura 14 – inserindo OPAD

Na janela *Symbol Properties*, na área *Parameters*, no campo *Name*, selecione *LOC* e em *Description*, e informe o número do pino da FPGA como *pXX*, onde *XX* representa o número do pino. Como na figura 15 o pino 19, e depois pressione o botão *Add*.

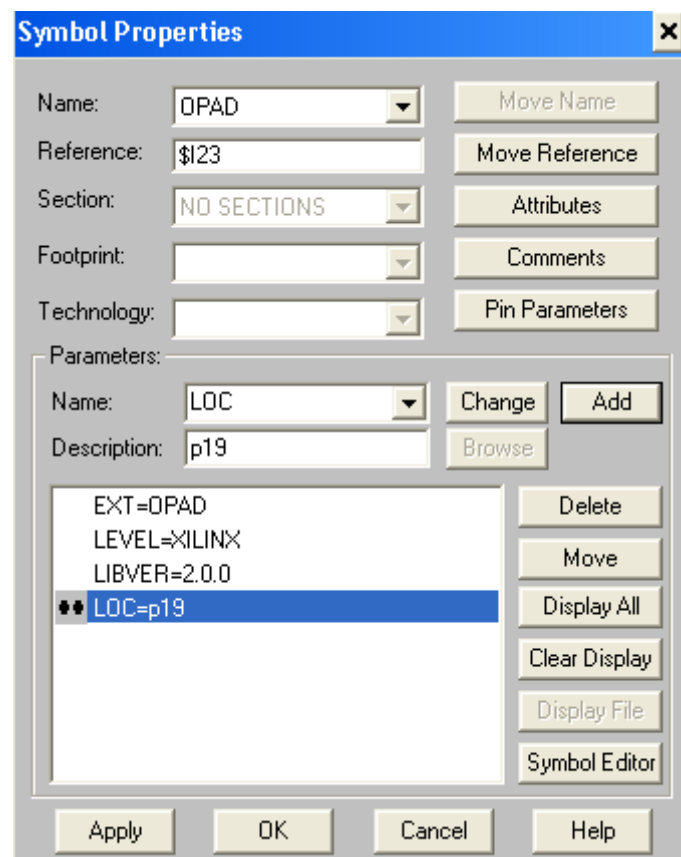




Figura 15 – Symbol Properties

No nosso exemplo iremos utilizar apenas opads, para selecionar os LEDs do display de sete segmentos, os pinos correspondentes são demonstrados na figura 16.

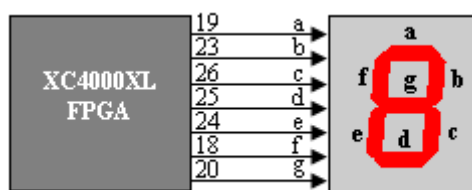


Figura 16 – display de sete segmentos

#### 4.2.4 Criando o Netlist

Após a construção do circuito lógico, deve ser criado e exportado o *netlist* utilizado para compilar o circuito para o FPGA escolhido. A geração do *netlist* fornece informações importantes para o desenvolvimento, como avisos e mensagens de erro. Após a geração e exportação do *netlist*.


Para criá-la basta clicar no Menu em “*OPTIONS*” → “*CREATE NETLIST*” ou pressionando Shift+F2.


Quando terminar, é uma boa prática verificar tem erros de projeto ou não. Isto é feito clicando no Menu em “*OPTIONS*” → “*INTEGRITY TEST*”.

Você pode também gerar um netlist EDIF clicando no Menu em “*OPTIONS*” → “*EXPORT NETLIST*”.

Se você não fizer isto, não há problema, o sistema pedirá para que seja feito posteriormente.

### 4.3 Functional Simulation

Abra o *Functional Simulation* clicando no ícone  ou pelo ambiente *Project Manager* (ver figura 5).

O primeiro passo é adicionar os sinais necessários para a visualização, isto pode ser feito clicando no botão  (Select Components) ou pelo menu “*signal*” → “*Add Signal*”. Abrirá a janela da figura 17, de dois clique no componente desejado na área “*Chip Selection*”, aparecerá seus pinos no quadro do lado direito, e então selecione os pinos que serão visualizados na simulação, depois aperte no botão “*Close*” para retornar ao simulador. Caso você utilize Hierarchy Connector seu sinal se encontra na área “*Signals Selection*” sendo possível seleciona-lo.

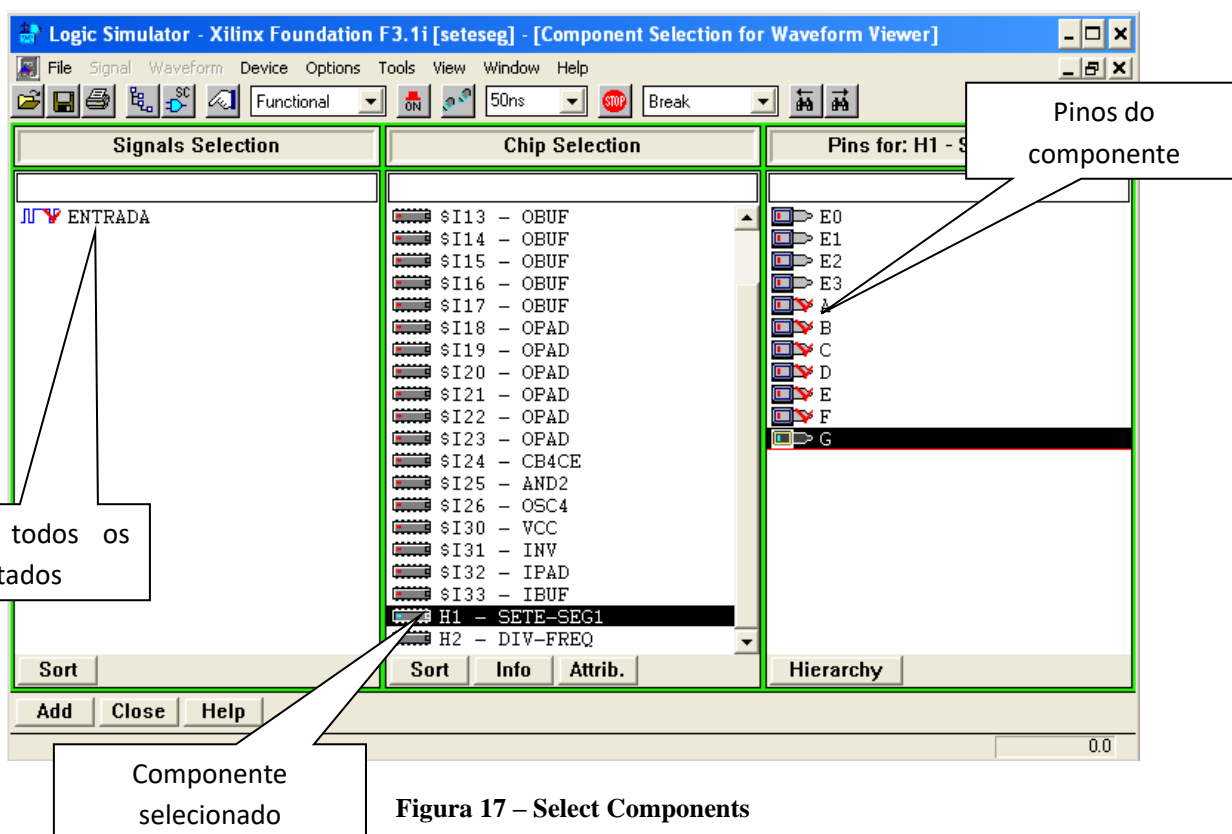



Figura 17 – Select Components

Também é possível adicionar pontos de prova no esquemático do circuito clicando no ícone  (Simulation Toolbox), depois clique no componente, aparecerá a janela da figura 18, selecione o pino desejado e clique no botão Add.

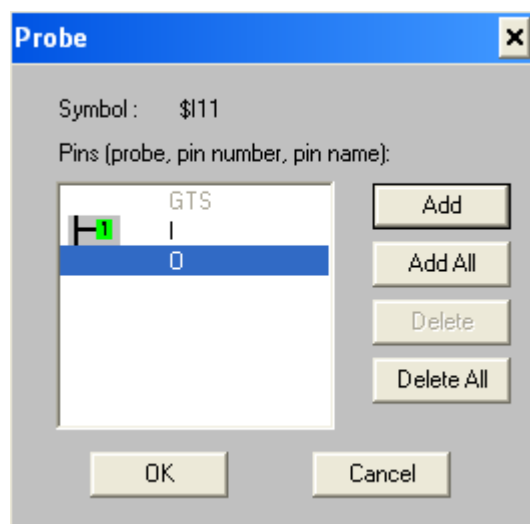



Figura 18 – Simulation Toolbox

O segundo passo é adicionar estímulos, para isso abra o "Select Stimulators" clicando no ícone . Aparecerá a janela da figura 19. Selecione o pino de entrada e selecione a ação desejada no Select Stimulators. Você pode associar teclas como estímulos para sinais no seu projeto. Depois de você associar este estímulo, o sinal varia entre 1 e 0 sempre que você pressionar a tecla correspondente no teclado.

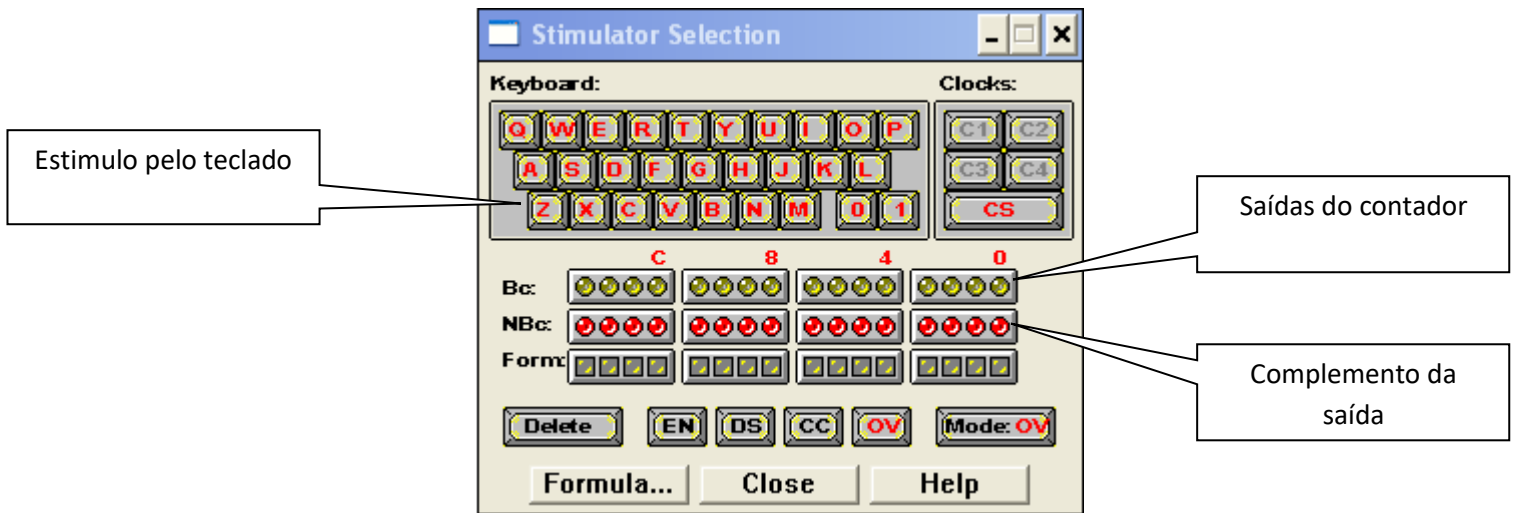



Figura 19 – Select Stimulators

Depois de Ter entrado com as entradas e saídas a serem simuladas e associado estímulos às entradas resta realizar a simulação.

Para iniciar a simulação clique em  (*Simulation Step*), para definir o intervalo dos passos de simulação, selecione uma opção do combobox do seu lado direito.

#### 4.4 Criando o Bitstream

Inicie clicando em *Implementation* no ambiente *Project Manager* (ver figura 5). O botão *Implementation* permite a entrada na função de implementação, que terá como produto final o arquivo *bitstream*(com extensão .bit) para *download* no FPGA. Após a entrada na função de implementação (*Implement Design*), a mesma é ativada clicando no botão *Run*. Observar que é mostrado o tipo de dispositivo FPGA escolhido inicialmente para o projeto, bem como os nomes da versão e revisão.

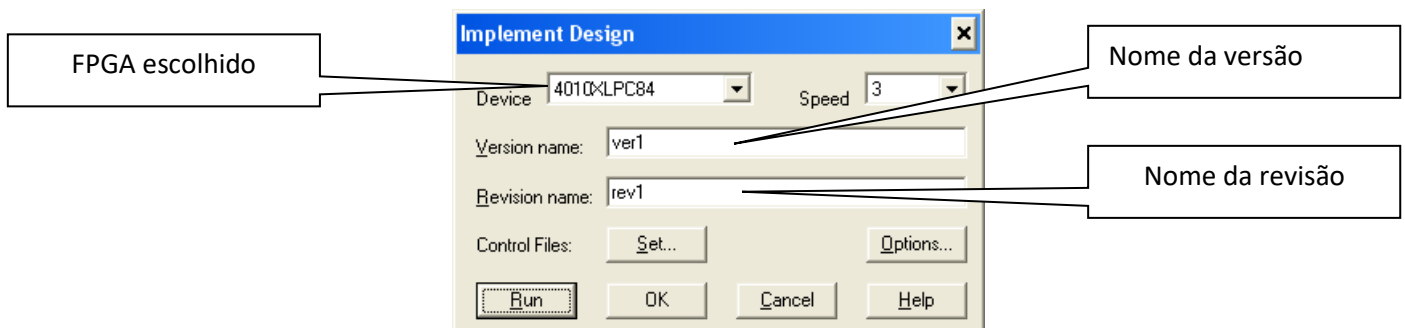
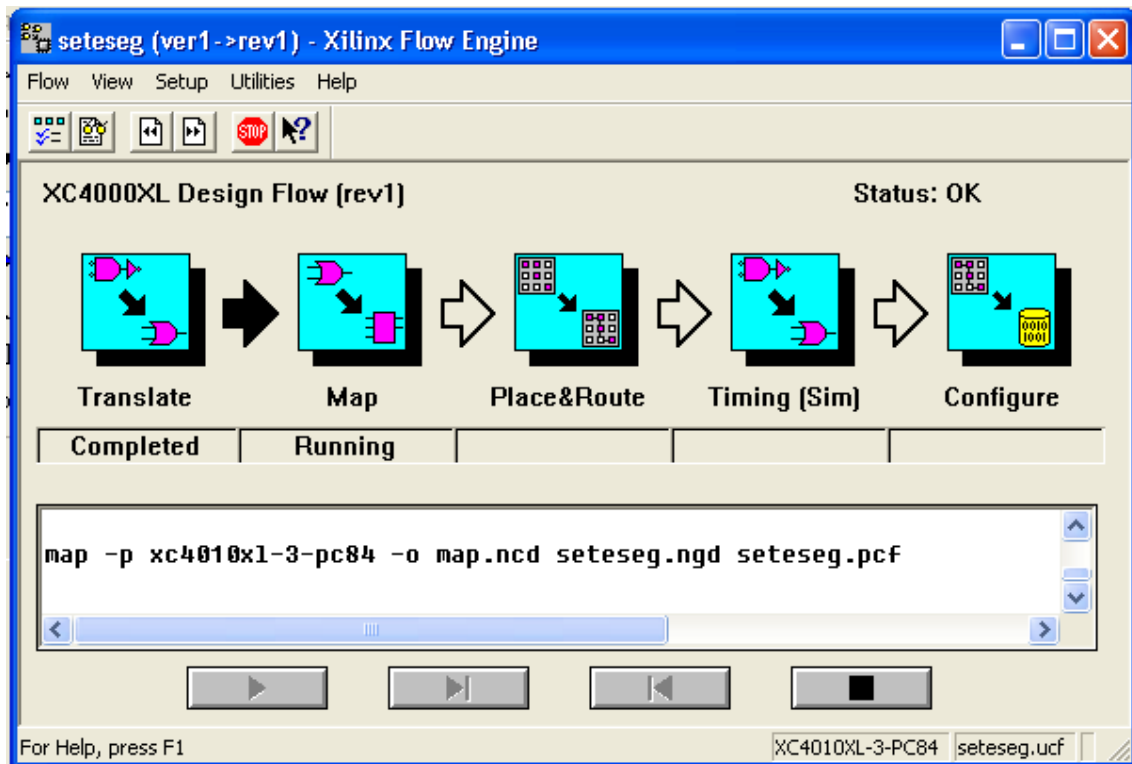


Figura 20 – Ativando a implementação



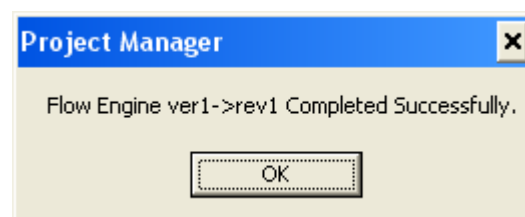


Os passos do processo de implementação são: tradução, mapeamento, roteamento, temporização e configuração. Após a ativação, os passos do processo de implementação seguem na seqüência indicada pelas setas ilustradas na figura 21.



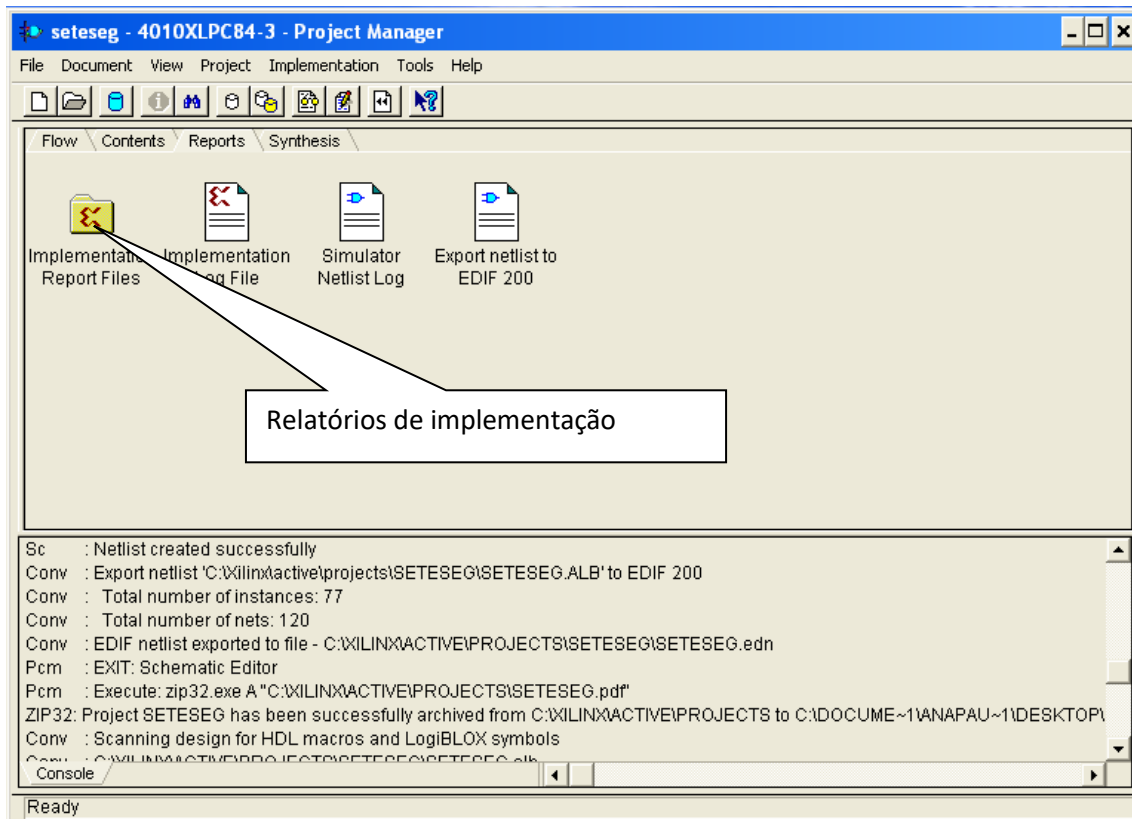
**Figura 21 – Passos do processo de implementação**

Ao término do processo de implementação, é exibida mensagem na tela. Caso ocorram erros, será exibida mensagem de alerta.

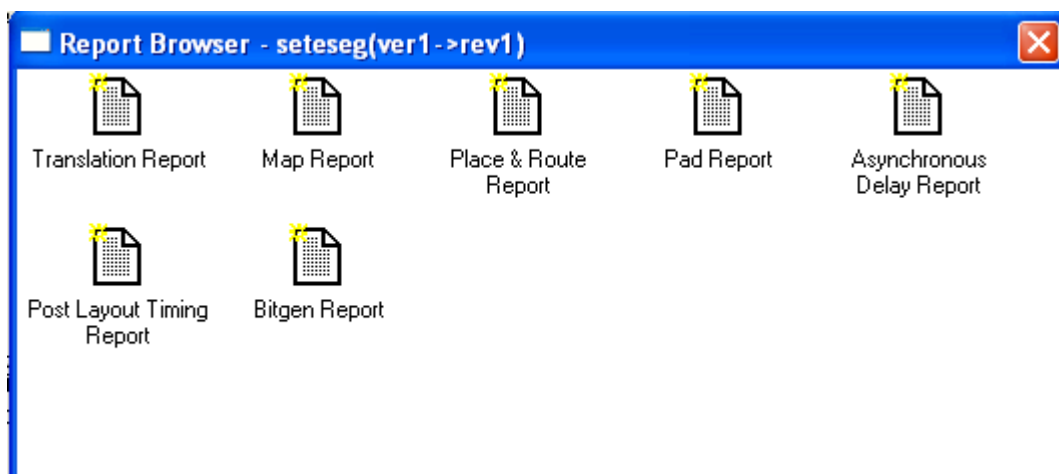


**Figura 22 – Fim do processo de implementação**

Os relatórios gerados por cada fase da implementação podem ser visualizados para verificar estatísticas de ocupação do FPGA, temporização e eventuais erros ocorridos.



**Figura 23 – Buscando os relatórios do processo de implementação**



**Figura 24 – Relatórios gerados pelo processo de implementação**



## 5 GXSL0AD

O GXSL0AD(figura 26) faz parte do software GXSTOOLs, e é utilizado para fazer a transferência do arquivo *bitstream* no FPGA da placa de desenvolvimento. O arquivo com extensão .bit é carregado na ferramenta e transferido para o FPGA via porta paralela.

Para descarregar o arquivo .bit no FPGA da placa de desenvolvimento XS40, basta buscar e arrastar para a área “FPGA/CPLD” de uma janela aberta da ferramenta GXSL0AD, o arquivo .bit que o Project Manager automaticamente salva no caminho Xilinx\active\projects\setseg\xproj\ver1\rev1 dentro de seu diretório de instalação.

Observar que “setseg” é o nome do projeto exemplo.

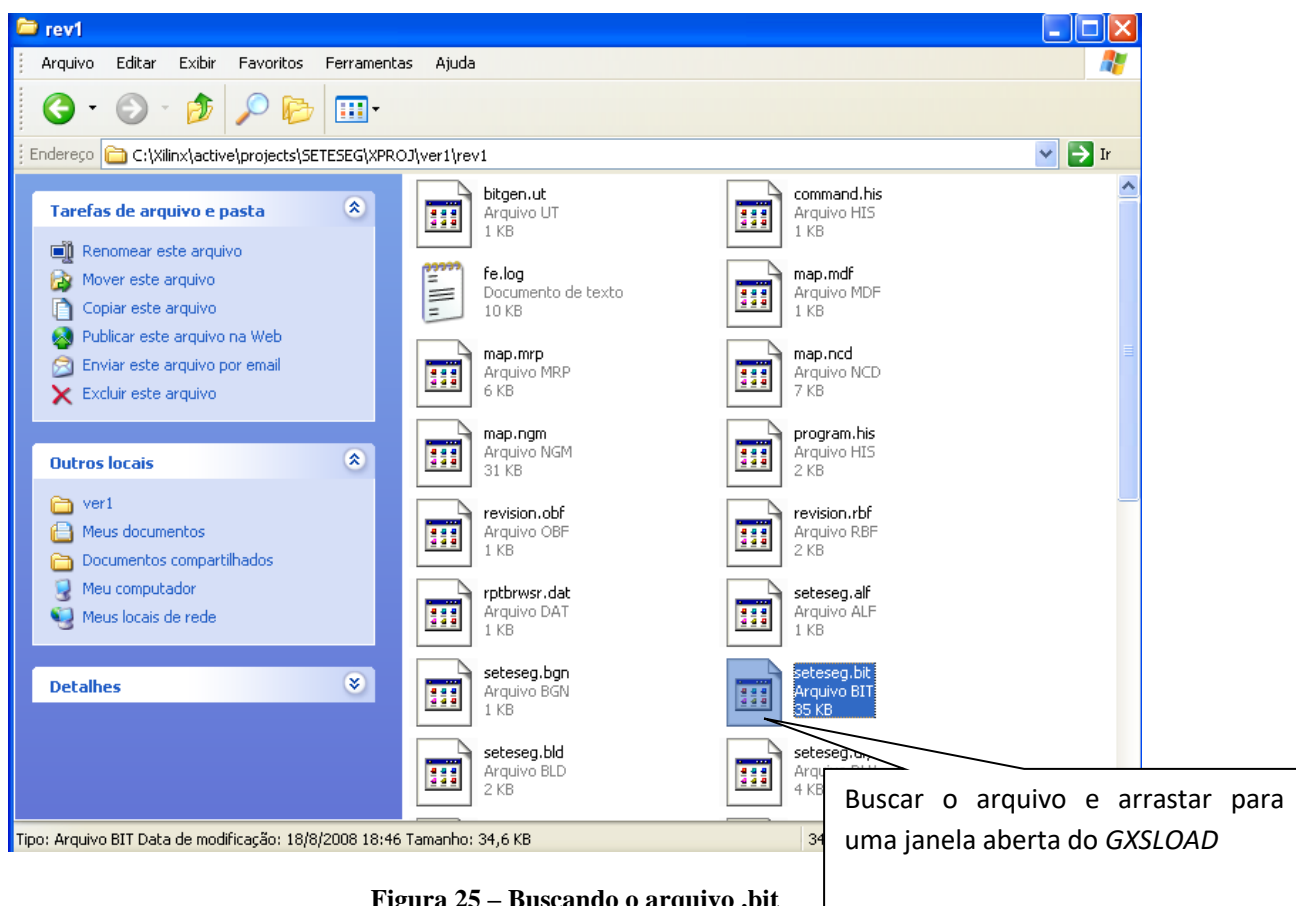
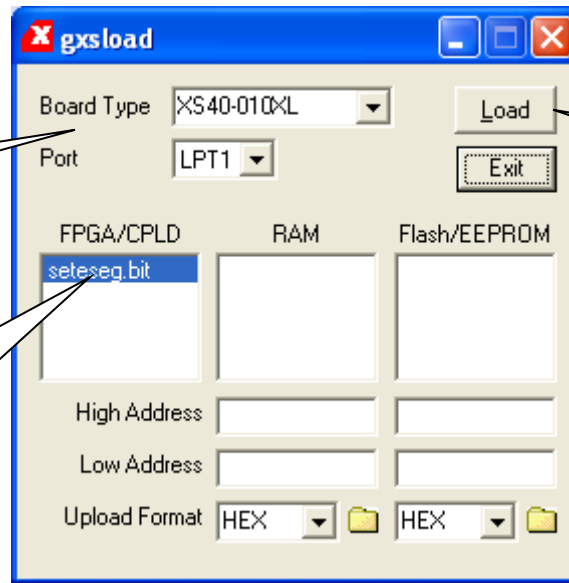


Figura 25 – Buscando o arquivo .bit

Na janela do GXSL0AD deve ser escolhido o tipo de placa de desenvolvimento e o nome da porta paralela a ser utilizada para descarregar o arquivo .bit no FPGA. Feito isso, basta clicar no botão *LOAD* para descarregar.



Tipo de placa de desenvolvimento e nome da porta paralela

Botão para descarregar o arquivo *bitstream*

Arquivo *bitstream* que foi arrastado

**Figura 26 – GXSLoad**

Após receber o arquivo *.bit*, a placa de desenvolvimento está pronta para os testes práticos.

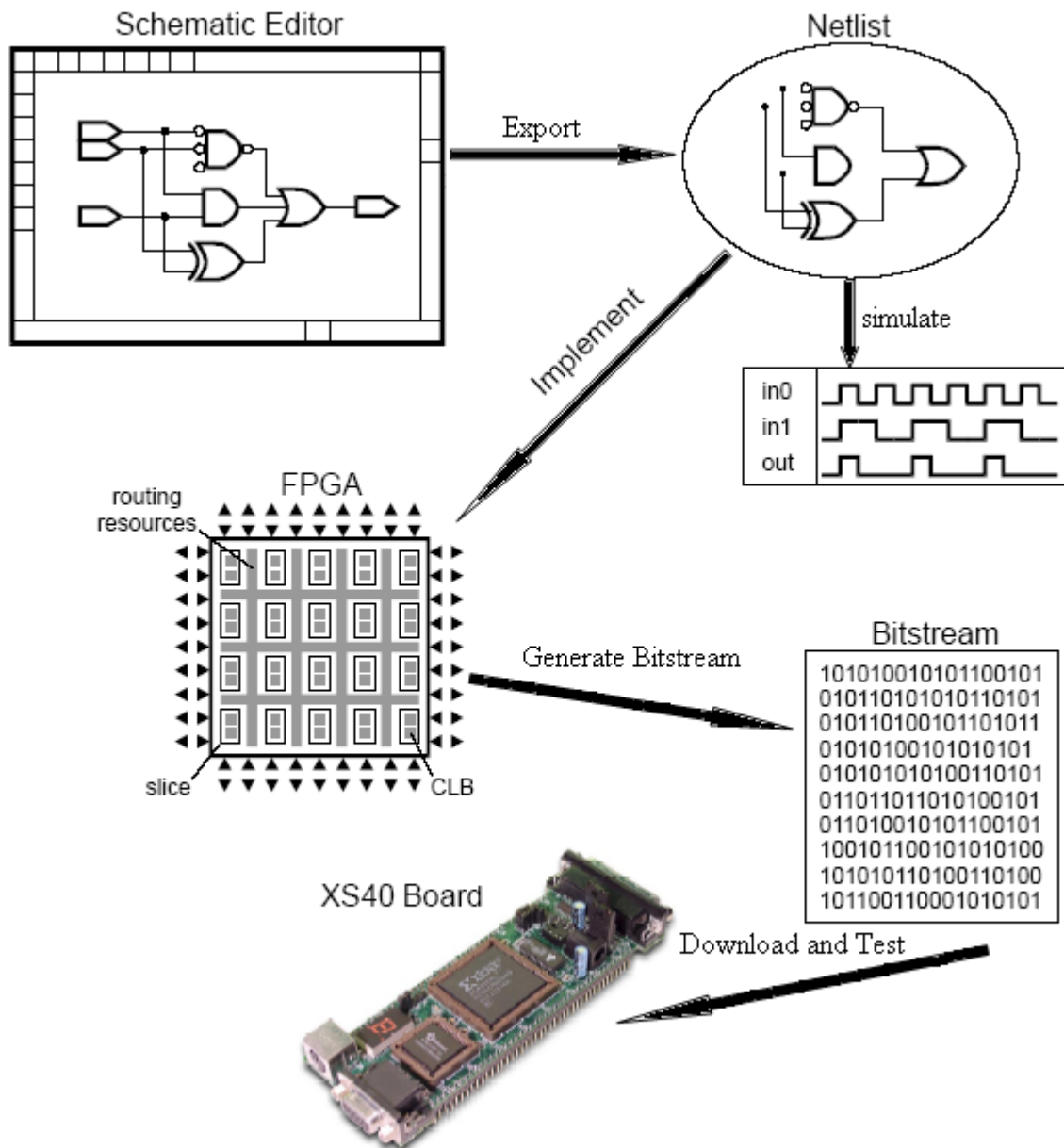


Figura 27 – Passos de desenvolvimento para projetos baseados em FPGA



## 6 Bibliografia

ORDONEZ, Edward David Moreno et al. **Projeto, Desempenho e Aplicação de Sistemas Digitais em Circuitos Programáveis (FPGAs)**. 1. ed. atual. Pompéia, SP: Bless, 2003. 300p.

MARTINS, Diego da Silva; MENEZES, Denison. **AUTOMAÇÃO E CONTROLE DE PULVERIZAÇÃO EM MÁQUINAS AGRÍCOLAS**. 2007. Projeto de Iniciação científica - CENTRO UNIVERSITÁRIO “EURÍPIDES DE MARÍLIA” – UNIVEM, Marília, SP, 2007.

XS40, XSP Board V1.2 User Manual

Dava Vanden Bout; Pragmatic Logic Design with XILINX Foundation 2.1i

### *Agradecimentos*

*Agradeço aos alunos Denison Menezes e Diego da Silva Martins pela dedicação, eficiência, senso de responsabilidade, ética e companheirismo no desenvolvimento das atividades de iniciação científica intitulado “Automação e Controle de Pulverização em Máquinas Agrícolas” que entre muitos outros relevantes resultados gerou também esse manual técnico.*

*Obrigado*

*Ildeberto de Genova Bugatti*



*“Que importa saber o que é linha reta  
quando não se sabe o que é retidão”*