PSI-3451 Projeto de CI Lógicos Integrados

Aula 4

A parte prática da aula 4 pretende colocar o aluno em contato com mais algumas estruturas sintáticas da linguagem, particularmente funções e procedimentos, além de praticar o modelo estrutural de projeto VHDL.

Tenha certeza que sabe responder as perguntas listadas abaixo após a observação dos arquivos ou dos resultados de simulações.

PARTE PRÁTICA

São fornecidos 2 arquivos VHDL de módulos relativos ao projeto **snake** e 1 arquivo referente a um pacote VHDL (*package*). Cada um dos módulos será modificado durante a aula gerando outros 4 arquivos. No total serão analisados e simulados 6 arquivos VHDL. A seguir vamos descrever cada um dos modelos fornecidos e os conceitos de VHDL que podem ser observados em cada uma das simulações.

Em cada experimento e respectivo arquivo VHDL será submetido à seguinte sequência de tarefas: **captura-compilação-simulação.**

O procedimento para **capturar, compilar** e **simular** cada arquivo VHDL é o mesmo utilizado nas aulas anteriores.

Preparação das Pastas e Arquivos para Simulação

- Ligar o computador e entrar no ambiente **Windows.**
- Utilizando o **Windows Explorer** acesse a sua área de trabalho na unidade de rede **X.**
- Na sua área de trabalho crie uma pasta X:**psi3451\aula_4** para armazenar os resultados desta prática. Como serão realizadas 6 simulações, três para cada arquivo fornecido, recomendamos que sejam criadas pastas separadas para cada uma das simulações com o objetivo de salvar adequadamente os resultados. Por exemplo, crie as pastas:

  X:\psi3451\aula_4\fa_3 (para o *full_adder_1*)

  X:\psi3451\aula_4\fa_4 (para o *full_adder_1* modificado)

  X:\psi3451\aula_4\rca_1 (para o *ripple carry adder* utilizando *full_adder_1*)

  X:\psi3451\aula_4\ng_1 (para o *num_gen_1*)

  X:\psi3451\aula_4\ng_2 (para o *num_gen_1* modificado com procedimento do aluno)

  X:\psi3451\aula_4\ng_3 (para o *num_gen_2* modificado com biblioteca criada pelo aluno)

  X:\psi3451\aula_4\lib_1 (para o *procedure_pack*, pacote com procedimento criado pelo aluno)

Ainda com o **Windows Explorer**, selecione **Rede=> NEWSERVERLAB => psi3451 => aula_4**. Transfira os arquivos deste diretório, *full_adder_3* e *num_gen_1* de módulos em VHDL, para as pastas correspondentes, **X:\psi3451\aula_3\fa_3** e **X:\psi3451\aula_3\ng_1**.

1) Captura, compilação e simulação do somador no modelo estrutural

a) Abra o arquivo *full_adder_3.vhd* para a pasta **X:\psi3451\aula_4\fa_3**. Ele está codificado no modelo VHDL estrutural (*structural*).

b) Analise toda a descrição, linha a linha. Tenha certeza que tenha encontrado e entendido os itens seguintes:

- a declaração de componentes
- o modelo VHDL *dataflow* do AND2, XOR2 e OR3 (*entity* e *architecture*)
- os parâmetros de tempo *generic* de cada um dos modelos acima e a sua utilização nas respectivas arquiteturas
- o posicionamento de todas as declarações de bibliotecas e pacotes
- as instanciações e os nomes das instâncias
- *port map* e *generic map*
- no *port map*, casos de associação implícita e explícita para os sinais
- a correspondência do *full adder* com o esquemático apresentado na apostila de conceitos (verifique as conexões entre as instâncias do VHDL).

c) Faça o procedimento de captura, compilação e simulação cf. descrito em exemplos trabalhados nas práticas anteriores.

Recomendação: realize a simulação para verificar:

- o funcionamento do *full adder* (use vetores de teste semelhantes aos usados na aula 2 para o *full adder* comportamental)
- as diversas variações dos sinais internos e das saídas de acordo com a mudança dos valores dos sinais de entrada

. Guarde os resultados do Wave para comparação na seção seguinte.

Perguntas: seguindo as recomendações, a simulação mostrou o comportamento esperado do circuito como descrito na apostila de conceitos? Quais foram os tempos de atraso verificados para as portas lógicas? Foi o esperado? Por que?

2) Captura, compilação e simulação do somador no modelo estrutural com alterações nas instaciações (generic)

a) Faça uma cópia do arquivo arquivo *full_adder_3.vhd* e salve-o com o nome *full_adder_4.vhd* na pasta **X:\psi3451\aula_4\fa_4**.

b) Em seguida modifique o arquivo (com o programa **Notepad++**), retirando as instruções de *generic map* em todas as suas ocorrências (deixe apenas os *port map*).

c) Repita o procedimento de captura, compilação e simulação do item 1)

Perguntas: quais foram os tempos de atraso verificados para as portas lógicas? Por que?

3) Projeto, captura, compilação e simulação do somador de 4 bits no modelo estrutural

a) Faça uma cópia do arquivo arquivo *full_adder_3.vhd* e salve-o com o nome *rc_adder_1.vhd* na pasta **X:\psi3451\aula_4\rca_1**.

b) Neste arquivo, você deve adicionar um somador *ripple-carry* de 4 bits (estrutural):

- denominando o arquivo como **rc_adder_1.vhd**
- denominando a *entity* como **rc_adder**
- utilizando o *full adder* do item 3) como componente.
- no ***rc_adder***, utilize o tipo *std_logic_vector (3 downto 0)* para os sinais de entrada (parcelas da soma) e de saída (resultado da soma) (use nomes significativos para os sinais); para os sinais de *carry-in* e *carry-out* do ***rc_adder***, use *std_logic*.

d) Repita o procedimento de captura, compilação e simulação.

Recomendação: realize a simulação para verificar:

- o funcionamento do *ripple carry adder*
- teste vários casos de propagação de *carry-out* e de ocorrência de *overflow*

. Guarde os resultados do Wave para futuras referências e comparações.

Perguntas: quantos níveis de hierarquia há no seu projeto? Onde as bibliotecas foram declaradas?

4) Captura, compilação e simulação do circuito num_gen_1 (muxes + função)

Funções devem ser vistas como estruturas que objetivam estruturar a descrição do modelo e aumentar a legibilidade do VHDL. Há uma forte semelhança entre funções da linguagem VHDL com as existentes em linguagens de programação de software como C ou Pascal. Não deve ser confundido com componentes que são representações estruturais de módulos.

Como características principais, as funções:

- retornam um valor;
- são chamadas dentro de expressões;
- não modificam valores que lhe são passados;
- são executados em um ciclo de simulação, ou seja, todos os parâmetros passados são vistos como variáveis.

a) Abra o arquivo *num_gen_1.vhd* para a pasta **X:\psi3451\aula_4\ng_1**.

b) Analise toda a descrição, linha a linha. Tenha certeza que tenha encontrado e entendido os itens seguintes:

- o parâmetros *generic* width
- a inferência dos três muxes como ilustrado na apostila de conceitos
- a declaração de função **rand_num_f** (veja onde você deverá completar os dados)
- o corpo da função
- a chamada de função
- a declaração de atraso inercial

c) Complete ao arquivo *num_gen_1.vhd* com os dados solicitados para a função (baseados no seu número USP)

d) Faça o procedimento de captura, compilação e simulação cf. descrito em exemplos trabalhados nas práticas anteriores.

Recomendação: realize a simulação para verificar:

- o funcionamento do circuito de acordo com a tabela da apostila de conceitos

. Guarde os resultados do Wave para comparação na seção seguinte.

Perguntas: seguindo as recomendações, a simulação mostrou o comportamento esperado do circuito como descrito na apostila de conceitos? O tempo de atraso especificado foi observado na simulação?

5) Captura, compilação e simulação do circuito num_gen_2 (muxes + procedimento)

Procedimentos também são estruturas que objetivam estruturar a descrição do projeto e aumentar a legibilidade do VHDL. Um procedimento pode ser visto como a descrição de um processo que pode ser invocado várias vezes. Um procedimento, diferente uma função, tem como características principais:

- pode gerar múltiplos valores, porém, não pelo comando de retorno;
- pode ter comando de retorno como forma de encerrar um procedimento;
- é chamado como comando;
- pode modificar valores que lhe são passados como parâmetros;
- apresentam noção de tempo.

a) Faça uma cópia do arquivo arquivo *num_gen_1.vhd* e salve-o com o nome *num_gen_2.vhd* na pasta **X:\psi3451\aula_4\ng_2**.

b) Como forma de experimentar diversas estruturas em VHDL, você deve trocar a função por um procedimento. Usando os slides da aula para suporte para a sintaxe de procedimento:

- denomine a entity de **num_gen_with_procedure**
- denomine o procedimento de **rand_num_p**
- verifique como compatibilizar a sintaxe de chamada de procedimento para a sintaxe do terceiro mux.
- deixe o tempo de atraso especificado dentro do procedimento

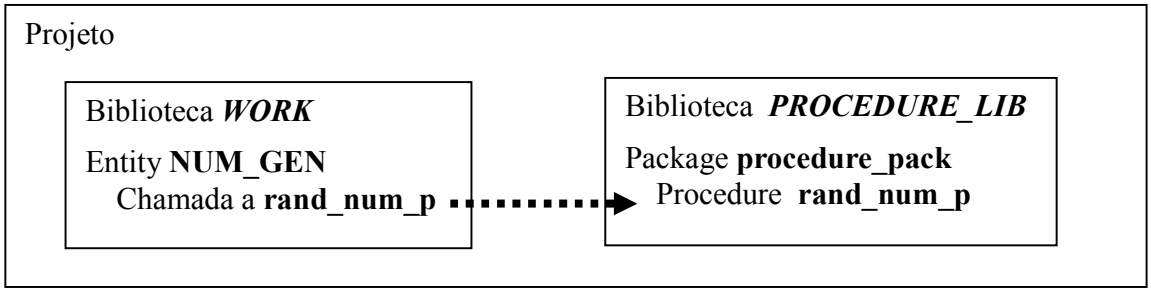
c) Repita o procedimento de captura, compilação e simulação do item 4).

. Guarde os resultados do Wave para comparação na seção seguinte.

Perguntas: a simulação mostrou o comportamento idêntico ao do item 4)? O tempo de atraso especificado foi observado na simulação?

6) Captura, compilação e simulação do circuito num_gen_2 com procedimento declarado em pacote de biblioteca específica

Nesta seção, o procedimento **rand_num_p** será colocado em uma biblioteca que pode ser acessada publicamente por descrições de circuito em um sistema de arquivo separado. Para isto, ele será declarado dentro de pacote **procedure_pack** a ser instalado em uma biblioteca **procedure_lib**. Neste experimento, o procedimento será chamado por um circuito, cuja unidade de projeto estará em ***work***, como mostrado no diagrama abaixo.



Primeiramente, a nova biblioteca deve ser construída. O package deve ser compilado em uma biblioteca ***work*** e depois remapeada com o nome **procedure_lib**.

a) Faça uma cópia do arquivo arquivo *num_gen_2.vhd* e salve-o com o nome *procedure_pack.vhd* na pasta **X:\psi3451\aula_4\lib_1**.

b) Faça o procedimento de criação de biblioteca *work* apontando-a para **X:\psi3451\aula_4\lib_1**

c) Edite o arquivo *procedure_pack.vhd*:

- Crie o pacote **procedure_pack** com a sintaxe de package (declaração e *body*). Use os slides da aula 3 para se referenciar.
- O único elemento do pacote será o procedimento **rand_num_p**

d) Compile o arquivo

Pergunta: o pacote aparece dentro da biblioteca *work*?

Nos passos seguintes, a nova biblioteca com o pacote recém-criada será utilizada pelo novo módulo *num_gen*.

e) Selecione a biblioteca *work* anterior (com o pacote) e apague o mapeamento.

f) Faça uma cópia do arquivo arquivo *num_gen_2.vhd* e salve-o com o nome *num_gen_3.vhd* na pasta **X:\psi3451\aula_4\ng_3**.

g) Faça o procedimento de criação de biblioteca *work* apontando-a para **X:\psi3451\aula_4\ng_3**

h) Crie uma nova biblioteca pelo menu. Na opção **Create**, selecione "**a map to an existing library**". Indique **procedure_lib** como nome e mapeie-o para **X:\psi3451\aula_4\lib_1\work.** Dê ok.

Perguntas: Apareceu a nova biblioteca **procedure_lib** junto à outras? O pacote aparece dentro dela?

i) Edite o arquivo *num_gen_3.vhd* da sequinte forma:

- elimine todas as declarações de **rand_num_p** (obs. a chamada ao procedimento dentro da arquitetura deve ser mantida)
- acrescente a declaração da biblioteca **procedure_lib** e ao uso do pacote **procedure_pack**

j) Repita o procedimento de compilação e simulação do item 4 para o arquivo *num_gen_3.vhd*.

Guarde os resultados do Wave para comparação na seção seguinte.

Perguntas: a simulação mostrou o comportamento idêntico ao do item 4)? De que maneira a atual forma de usar pacotes é diferente daquelas usadas na aula 3?