Curso de Verilog - Dia 1

Caio Rodrigo, Jorge Reis

Prática 0 - Hello World

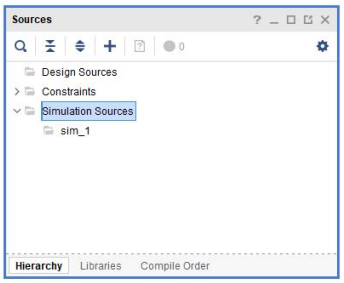
Vamos agora realizar um de exemplo Hello World em Verilog para se habituar com o Vivado e aprender a executar arquivos em Verilog.

Criando o Projeto

1. Crie um projeto no Vivado;
2. **Project Name**: escolha um nome e o local do projeto;
3. **Project Type**: marque "RTL Project"e "Do not specify sources at this time";
4. **Default Part**: aqui é possível escolher a FPGA para a síntese. Como nenhuma síntese será feita, a escolha é irrelevante;
5. Finish.

Adicionando Arquivos

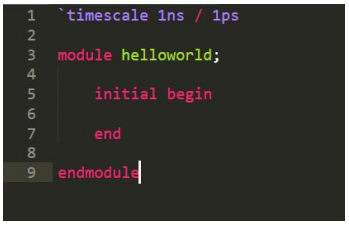
O Vivado abrirá o projeto e agora é preciso criar o arquivo do código fonte. Para isso, clique em no símbolo do "+". Para as práticas 0 e 1 será preciso criar arquivos de **simulação** e para a prática 2 será preciso criar arquivos de **design**.



Na janela que se abriu, siga o seguintes passos:

1. Selecione "Add or create simulation sources";
2. Clique em "Create File", digite o nome do arquivo a ser criado e deixa o "File Type"como Verilog;
3. **Define Module** aqui é possível criar entradas e saídas para o módulo do arquivo criado. Isso não será necessário para o exemplo;
4. O arquivo foi criado dentro da pasta "Simulation Sources".

Abra o arquivo, por padrão, o Vivado gera um módulo já com timescale, um header e o protótipo do módulo. Acrescente ao código gerado a diretiva **Initial** do Verilog, conforme o código abaixo:



O initial é importante para a execução de **System Tasks** e possibilita a atribuição de variáveis como será visto mais adiante no curso. Agora é possível utilizar a **System Task $display** para fazer o **Hello World**.

Executando Arquivos

A única forma de "executar"arquivos em Verilog é por meio de simulação. Para isso, clique em **Simulation** > **Run Simulation** > **Run Behavioral Simulation** em **Flow Navigator**, localizado no lado esquerdo do Vivado.

Se tudo tiver ocorrido corretamente, a mensagem será exibida na janela **Tcl Console** localizada na parte inferior do programa.

Prática 1 - Trabalhando com Números

System Task $display

A system task $display serve para imprimir textos no terminal do simulador. Ela tem seu funcionamento similar à função **printf** da linguagem C.

```
$display("Ola"); // Printa "Ola" no terminal
```

O print do valor de uma variável é possível por meio do caractere "%"da seguinte forma:

```
$display("%d", var); // Printa a variavel "var" em decimal "%d"
$display("%d e %b", var1, var2); // Printa a variavel "var1" em decimal
                                 // e "var2" em binario
```

Caso queira utilizar o caractere "%", use ele duas vezes na string do display: "%%".

Tabela 1: Especificação dos formatos de strings

Format	Display
%d ou %D	Mostra a variável em decimal
%b ou %B	Mostra a variável em binário
%s ou %S	Mostra a string
%h ou %H	Mostra a variável em hexadecimal
%c ou %C	Mostra um caractere ASCII
%m ou %M	Mostra o nome hierárquico do módulo (não precisa de argumento)
%v ou %V	Mostra a força do sinal
%o ou %O	Mostra a variável em octal
%t ou %T	Mostra o tempo de simulação
%e ou %E	Mostra um número real em notação científica (EX: 3e10)
%f ou %F	Mostra um número real em formato decimal (EX: 2.13)
%g ou %G	Mostra um número real em notação científica ou decimal, o que for menor

Atividades

Utilizando a simulação e a task **$display**, faça as atividades abaixo e confira os resultados no Tcl Console.

1. Escreva os seguintes números:
   (a) Número decimal "123"como um número sized 8-bits
   (b) Hexadecimal de 16-bits de valor desconhecido (tudo X)
   (c) Número de 4-bits '-2' em hexadecimal. Compare com '-2' escrito na forma de complemento de dois em um registrador de 4 bits
   (d) Número hexadecimal unsized "1234"
2. Crie variáveis dos tipos V1 e V2, salve os números e compare-os utilizando a task display com os formatos abaixo.
   (a) V1 = (integer) 5; V2 = (reg[4:0]) 5; %d e %b
   (b) V1 = (integer) -5; V2 = (reg[4:0]) -5; V3 = (reg signed[4:0]) -5; %d e %b
   (c) V1 = (reg[4:0]) 3; V2 = (reg[0:4]) 3; %b e %b de V1[4] e V2[4]
   (d) V1 = (integer) 3; V2 = (real) 3.0; %d e %b

Prática 2 - Gate Modelling

A modelagem no nível de abstração de gates se aproxima bastante com o que é feito em Eletrônica Digital. O desenvolvedor se preocupa em instanciar gates e conectá-los para criar o circuito desejado. Alguns exemplos de instanciações:

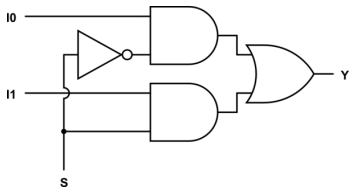
```
and a1 (out,in1,in2);          // Instanciacao basica
or or1 (out,in1,in2,in3);      // O numero de entradas pode ser maior que 2
not (out, in);                 // Nao e preciso colocar o nome do gate para intancia-lo
// Repare que a primeira porta e sempre um output
```

Alguns gates que podem ser instanciados: and, nand, or, nor, xor, xnor...

Atividades

Utilize modelagem em nível de gates para criar os circuitos abaixo. Lembre-se de criar os arquivos como **design source** e não **simulation source**

1. Multiplexador 2x1:



2. Full Adder: