Curso Complementar de VHDL

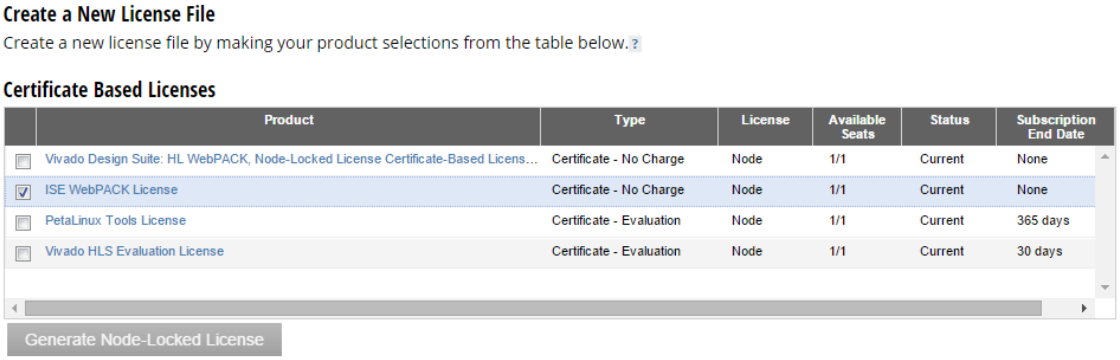
1. Baixando o Xilinx ISE

a. Entrar no site http://www.xilinx.com/
b. No menu *SUPPORT*, acessar *Downloads & Licensing*
c. Selecionar *ISE*
d. Escolher a versão 14.5
e. Em **ISE Design Suite - 14.5 Full Product Installation**, realizar o download da versão desejada:
    a. Windows https://secure.xilinx.com/webreg/register.do?group=dlc&htmlfile=&emailFile=&cancellink=&eFrom=&eSubject=&version=14.5&akdm=&filename=Xilinx_ISE_DS_Win_14.5_P.58f_4.tar
    b. Linux https://secure.xilinx.com/webreg/register.do?group=dlc&htmlfile=&emailFile=&cancellink=&eFrom=&eSubject=&version=14.5&akdm=1&filename=Xilinx_ISE_DS_Lin_14.5_P.58f_4.tar
f. O instalador ocupa aproximadamente 6 GB.
g. Instalar a versão WebPack.

Os alunos que quiserem podem levar um pen-drive de 8+ GB para fazermos a transferência do instalador do WebPack.

2. Obtendo uma licença para o Xilinx ISE WebPack

a. Entrar no site http://www.xilinx.com/
b. Realizar o cadastro no site.
c. Acessar *Xilinx Product Licensing Site* através do link: http://www.xilinx.com/getlicense
d. Em *Create a New License File*, escolher *ISE WebPack License* e clicar em *Generate Node-Locked License*.



3. Laboratório L434

A senha para acesso ao laboratório é 1357. Os alunos que não possuírem conta no departamento de elétrica podem acessar os computadores do laboratório com o login "*aula"* e senha "*a.2010"*. Durante o desenvolvimento das tarefas, os arquivos devem ser salvos em C:\Temp. Após terminar de utilizar o ISE, os alunos devem salvar seus projetos em um *pendrive* ou enviar para si mesmos por e-mail pois a pasta *Temp* é apagada diariamente. Os alunos que possuírem login podem salvar em seus perfis.

4. Auxílio na Tarefa da Aula 0 – variáveis intermediárias

As variáveis intermediárias são utilizadas dentro de uma estrutura e não fazem parte da "entidade". Como vimos na aula, a entidade define os sinais de entrada e saída, ou seja, como a nossa "caixa preta" se comunica com o exterior. Chamaremos de variáveis intermediárias, portanto, aqueles sinais que são empregados na lógica interna de uma estrutura mas não são acessíveis ao exterior. Como veremos, no exemplo abaixo, elas podem ser usadas para conectar as entradas às saídas. Abaixo segue um exemplo do *Full Adder* utilizando variáveis intermediárias.

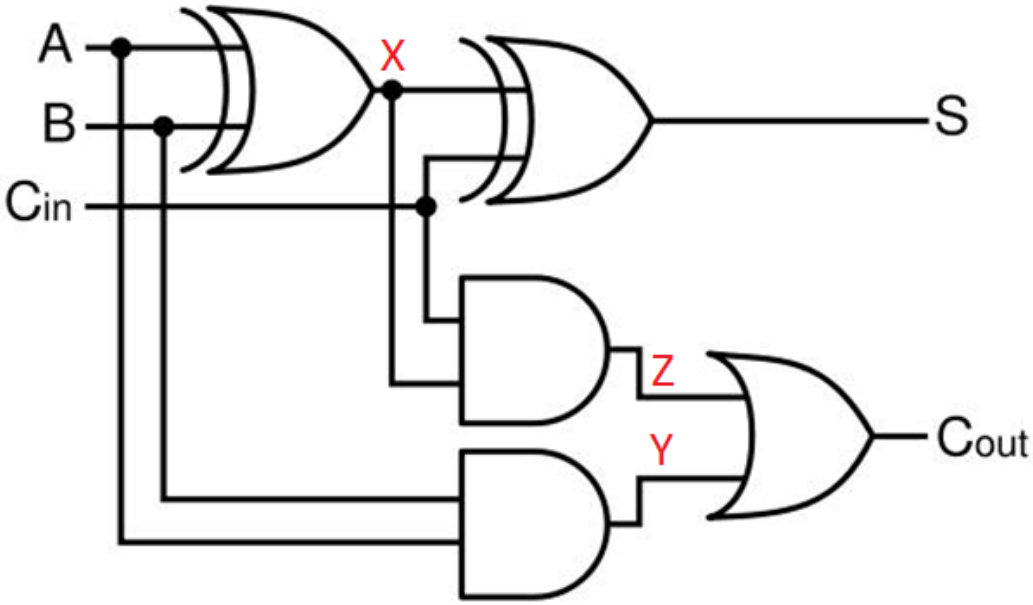
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity fadder is
	Port (
		LOCAL_CLOCK    : in   STD_LOGIC;

		A      : in   STD_LOGIC;
		B      : in   STD_LOGIC;
		Cin    : in   STD_LOGIC;

		Cout   : out  STD_LOGIC;
		Sum    : out  STD_LOGIC
	);
end fadder;

architecture Behavioral of fadder is

	signal X : std_logic := '0';
	signal Y : std_logic := '0';
	signal Z : std_logic := '0';

begin

	Cout <= Y or Z;
	Sum  <= Cin xor X;

	X <= a xor b; -- VHDL não é case sensitive (não faz diferença se é maiúsculo ou minúsculo)!!
	Y <= A and B;
	Z <= X and cin;

end Behavioral;
```

As variáveis intermediárias são declaradas como:

signal ***nome*** *:* ***tipo****;*

entre *architecture Behavioral* e *begin*. O tipo pode ser *std_logic*, *std_logic_vector*, etc. Não deve-se defini-las como *IN* ou *OUT* já que, como vimos, elas não são acessíveis fora da estrutura a que pertencem. É um bom costume inicializá-las com o valor zero (*:= '0'*).

As variáveis intermediárias X, Y e Z do código acima são mostradas em vermelho no diagrama do *Full Adder* abaixo.



O desenvolvimento de um *Full Adder* de 4 bits fica muito mais simples com a utilização de variáveis intermediárias. Abaixo segue o diagrama de um *Full Adder* de 4 bits. As entradas estão em vermelho e as saídas em preto.