

Laboratório 03

Tutorial da Ferramenta Quartus – Uso da Placa

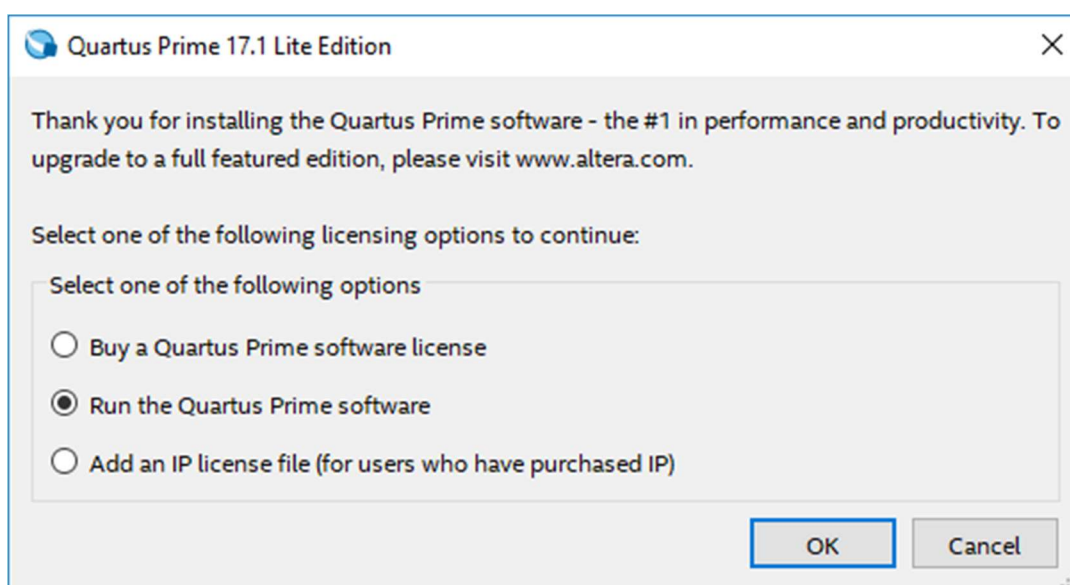
Este laboratório é um tutorial para familiarização de uso da placa.

Parte I – Antes de começar

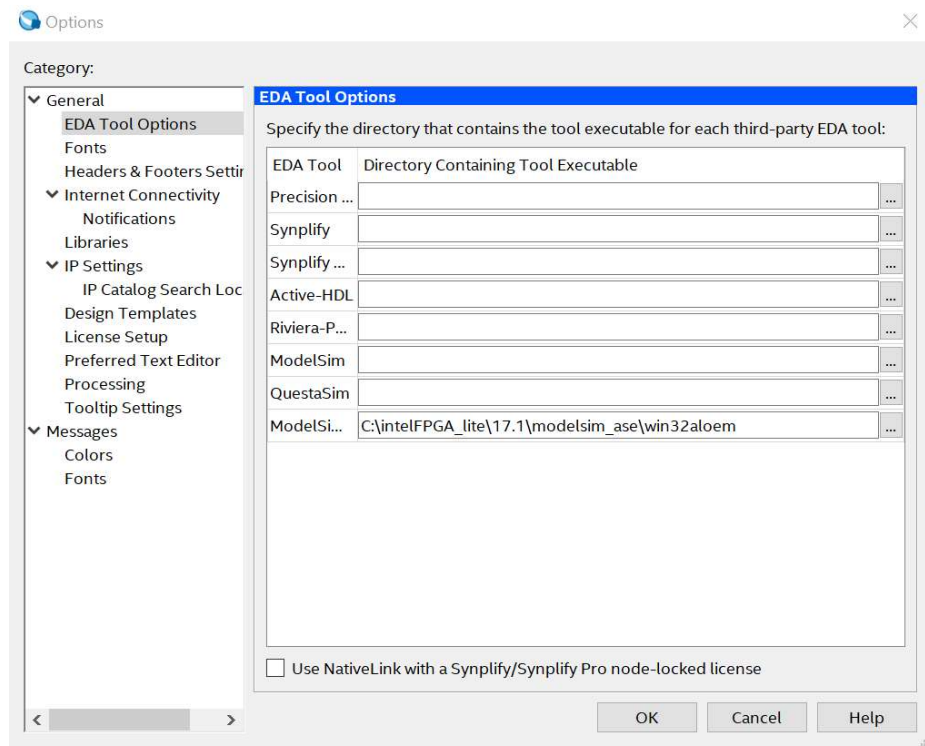
I-1. Faça o download do manual da placa DE2-115. O arquivo está disponível na página do curso no Canvas, em Arquivos/Arquivos uso da placa. Utilize-o como referência para todos os laboratórios. O manual contém informações sobre como funcionam e como utilizar cada um dos periféricos disponíveis na placa.

I-2. Faça o download do arquivo de *Pin Assignments* que também está disponível no Canvas, em Arquivos/Arquivos uso da placa. Esse arquivo será necessário para configurar os projetos e gravar os circuitos na placa.

I-3. Ao iniciar o Quartus pela primeira vez, poderá ser exibida a tela abaixo. Escolha a opção “Run the Quartus Prime software” para prosseguir com a licença gratuita do Quartus.



I-4. Verifique a configuração do caminho para a instalação do simulador Modelsim. Dentro da janela principal do Quartus, selecione **Tools > Options...** Selecione a aba **EDA Tool Options** e configure o caminho para a ferramenta **ModelSim-Altera**. Em uma instalação padrão, o caminho é `C:\intelFPGA_lite\20.1\modelsim_ase\win32aloem`.



Parte II – Criando um projeto no Quartus

II-1. Da janela principal do Quartus, selecione **File > New Project Wizard...** Na janela **New Project Wizard**, parte **Introduction** (se for exibida), pressione **Next >** até chegar na janela de configuração do projeto. Configure conforme abaixo:

- Directory:** caminho onde o projeto será salvo;
- Name:** um nome para o projeto;
- Top-Level Entity:** a entidade (módulo) que será a primeira da hierarquia do projeto, aquela que se comunicará diretamente com os pinos de entrada e saída da placa DE2-115. Essa configuração pode ser modificada posteriormente.

Dicas:

- Não utilize nomes de projeto e nomes de entidade muito longos ou que contenham espaços ou caracteres especiais.
- VHDL não é *case sensitive*. Portanto, não utilize letras maiúsculas e minúsculas para diferenciar nomes de entidades (como a top-level).

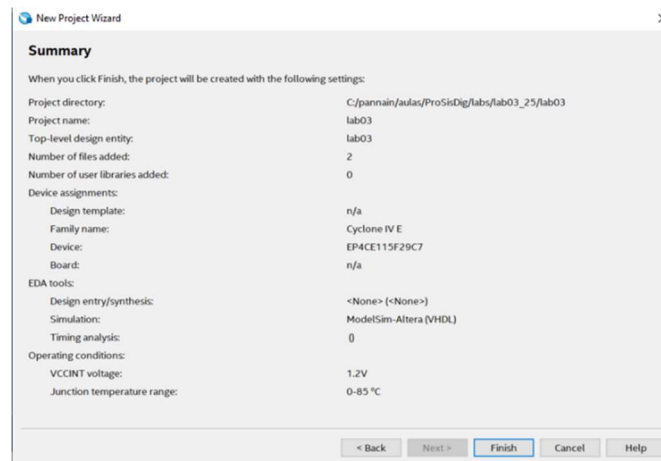
- c) Evite utilizar caminhos de diretório muito longos, que contenham caracteres especiais e, se possível, espaços.
- d) Evite utilizar diretórios em dispositivos mais lentos, como pendrives e unidades mapeadas em rede. Trabalhe na pasta c:\Usuarios\ "seu RA"\Lab03 e copie os arquivos depois. O processo de compilação do Quartus gera muitos arquivos e arquivos grandes. Utilizar dispositivos lentos pode prejudicar o processo.

The screenshot shows the 'New Project Wizard' dialog box with the title 'Directory, Name, Top-Level Entity'. It contains three text input fields: 'What is the working directory for this project?' with the value 'C:\Usuarios\seu RA\lab03', 'What is the name of this project?' with the value 'lab03', and 'What is the name of the top-level design entity for this project? This name is case sensitive and must exactly match the entity name in the design file.' with the value 'lab03'. There is a button 'Use Existing Project Settings...' and a set of navigation buttons at the bottom: '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'.

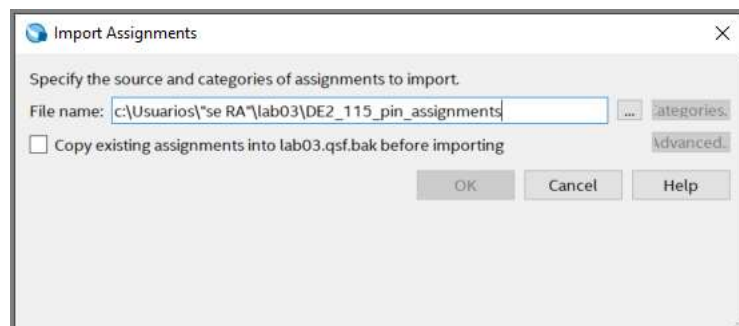
II-2. Avance selecionando **Next >** até chegar na janela **Family, Device & Board Settings**. Na metade esquerda da tela, selecione a **Device family** “Cyclone IV E”. Na metade direita, copie e cole o **Name filter** “EP4CE115F29C7”. Este é o nome do dispositivo FPGA presente na placa. Em **Available devices**, selecione o único dispositivo que aparecerá e confirme clicando em **Finish**.

The screenshot shows the 'New Project Wizard' dialog box with the title 'Family, Device & Board Settings'. It has two tabs: 'Device' and 'Board'. The 'Device' tab is active. It contains a section 'Select the family and device you want to target for compilation. You can install additional device support with the Install Devices command on the Tools menu. To determine the version of the Quartus Prime software in which your target device is supported, refer to the [Device Support List](#) webpage.' Below this, there are two main sections. The left section is 'Device family' with a dropdown menu showing 'Cyclone IV E' and a 'Device' dropdown showing 'All'. The right section is 'Show in \'Available devices\' list' with dropdowns for 'Package' (Any), 'Pin count' (Any), and 'Core speed grade' (Any), and a text input for 'Name filter' containing 'EP4CE115F29C7'. There is a checkbox 'Show advanced devices' which is checked. Below these sections is a table titled 'Available devices:' with columns: Name, Core Voltage, LEs, Total I/Os, GPIOs, Memory Bits, and Embedded multiplier. The table contains one row: EP4CE115F29C7, 1.2V, 114480, 529, 529, 3981312, 532. At the bottom, there are navigation buttons: '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'.

Name	Core Voltage	LEs	Total I/Os	GPIOs	Memory Bits	Embedded multiplier
EP4CE115F29C7	1.2V	114480	529	529	3981312	532



II-3. De volta à janela principal do Quartus, selecione **Assignments > Import assignments....** Na janela **Import Assignments**, busque o arquivo de Pin assignments que você baixou no passo I-2 e confirme clicando em **OK**. Você pode desmarcar a caixa **Copy existing assignments into lab03.qsf.bak before importing**.

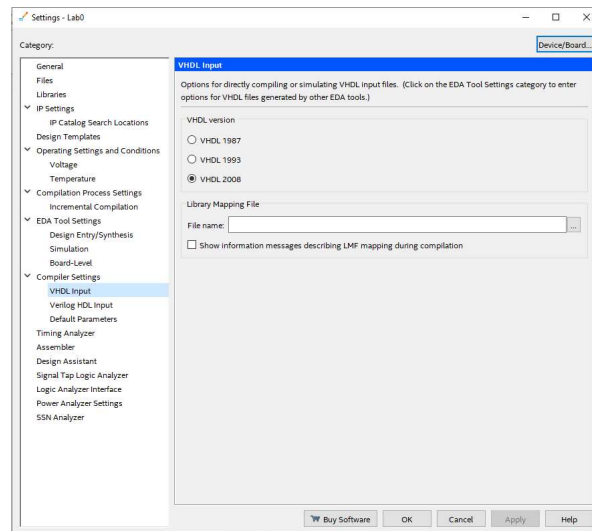


II-4. Em **Assignments > Assignment Editor**, veja a lista de assignments que você acabou de importar. Os assignments são as associações entre os pinos de conexão da FPGA na placa DE2-115 e os periféricos traduzidos como nomes de sinais que representam o que é cada elemento. A coluna **To** mostra os nomes dos sinais que devem ser usados como sinais de entrada e saída na entidade top-level do projeto. Os nomes dos sinais são os mesmos listados no manual da placa que você baixou no item I-1. Por exemplo, se você quiser utilizar os LEDs vermelhos da placa como saída, basta criar uma saída do tipo vetor com 18 posições (17 até 0).

Pin	From	To	Assignment Name	Value	Resolved	Entity	Comment	Tag
11	LED17	Location	Pin_015	Yes				
14	LED17	IO Standard	2.5 V	Yes		DE2_115		
15	LED17	Location	Pin_016	Yes				
16	LED17	IO Standard	2.5 V	Yes		DE2_115		
17	LED17	Location	Pin_017	Yes				
18	LED17	IO Standard	2.5 V	Yes		DE2_115		
19	LED17	Location	Pin_018	Yes				
20	LED17	IO Standard	2.5 V	Yes		DE2_115		
21	LED17	Location	Pin_019	Yes				
22	LED17	IO Standard	2.5 V	Yes		DE2_115		
23	LED17	Location	Pin_020	Yes				
24	LED17	IO Standard	2.5 V	Yes		DE2_115		
25	LED17	Location	Pin_021	Yes				
26	LED17	IO Standard	2.5 V	Yes		DE2_115		
27	LED17	Location	Pin_022	Yes				
28	LED17	IO Standard	2.5 V	Yes		DE2_115		
29	LED17	Location	Pin_023	Yes				
30	LED17	IO Standard	2.5 V	Yes		DE2_115		
31	LED17	Location	Pin_024	Yes				
32	LED17	IO Standard	2.5 V	Yes		DE2_115		
33	LED17	Location	Pin_025	Yes				
34	LED17	IO Standard	2.5 V	Yes		DE2_115		
35	LED17	Location	Pin_026	Yes				
36	LED17	IO Standard	2.5 V	Yes		DE2_115		
37	LED17	Location	Pin_027	Yes				
38	LED17	IO Standard	2.5 V	Yes		DE2_115		
39	LED17	Location	Pin_028	Yes				
40	LED17	IO Standard	2.5 V	Yes		DE2_115		
41	LED17	Location	Pin_029	Yes				
42	LED17	IO Standard	2.5 V	Yes		DE2_115		
43	LED17	Location	Pin_030	Yes				
44	LED17	IO Standard	2.5 V	Yes		DE2_115		

II-5. De volta à janela principal do Quartus, selecione **Assignments > Settings....** Nas abas **Category**, à esquerda, selecione **VHDL Input** dentro de **Compiler Settings**. Ajuste **VHDL version** para **VHDL 2008**. A versão 2008 do VHDL permite utilizar construções que não existiam em versões anteriores. Confirme clicando em **Apply** e **OK**.

Note que, dependendo da resolução da tela, o botão OK pode não aparecer porque a janela não cabe na tela. Nesse caso, você pode precisar usar artifícios como redimensionar a janela, maximizar a janela, ocultar a barra de tarefas do Windows ou repensar seriamente suas escolhas de vida ao usar o Quartus para poder ver o botão.

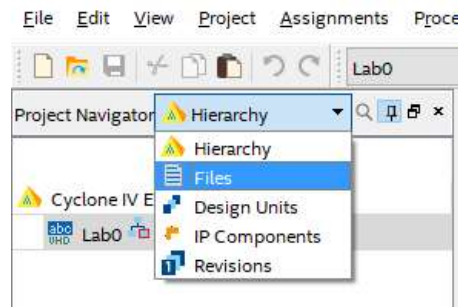


II-6. A janela **Settings** permite alterar outras configurações importantes do projeto criado. Por exemplo, na aba **General** pode ser alterado o nome da entidade top-level; na aba **Libraries** podem ser adicionadas e removidas bibliotecas adicionais e criadas bibliotecas customizadas para projetos maiores; na aba **Files** podem ser conferidos os arquivos adicionados, o caminho para cada um, a biblioteca que eles pertencem e a ordem em que eles são compilados, assim como adicionados e removidos arquivos; e o botão **Device/Board...** no canto superior direito permite alterar o dispositivo escolhido no momento da criação do projeto.

Parte III – Gerenciando arquivos no projeto

III-1. Para criar novos arquivos VHDL, selecione **File > New...** e **VHDL File** dentro de **Design Files**. Ao salvar o arquivo pela primeira vez, o Quartus exibirá uma caixa com a opção **Add file to current project** para adicionar o novo arquivo diretamente ao projeto.

III-2. Para ver e navegar pelos arquivos do projeto, selecione a opção **Files** no menu drop-down do painel **Project Navigator**. Se o painel **Project Navigator** não estiver visível, ative-o em **View > Utility Windows > Project Navigator**.



III-3. Para adicionar arquivos já existentes ao projeto, selecione **Assignments > Settings...** e vá na aba **Files** do painel à esquerda.

Dicas:

- Procure criar um arquivo para cada entidade VHDL que for utilizar, e nomear o arquivo com o mesmo nome da entidade que está descrita nele.
- Se for criar múltiplas entidades em um mesmo arquivo, note que as declarações de bibliotecas (por exemplo, `library ieee` e `use ieee.std_logic_1164.all`) devem ser repetidas antes de cada declaração de entidade.
- Procure copiar os arquivos VHDL que for utilizar no projeto para dentro da pasta do projeto. Se for utilizar arquivos que estão em outras pastas, preste atenção nos caminhos e tome cuidado para não adicionar várias vezes o mesmo arquivo ao projeto.
- Não utilize nomes de arquivos muito longos ou que contenham espaços ou caracteres especiais.
- VHDL não é *case sensitive*. Portanto, não utilize letras maiúsculas e minúsculas para diferenciar nomes de entidades.
- A ordem em que os arquivos aparecem na aba **Files** importa. O quartus considera os arquivos na ordem em que eles estão listados ali, e isso faz diferença principalmente para simulação. Portanto, deixe os arquivos listados de maneira que os arquivos que estão mais para baixo na lista dependam apenas de arquivos que estão mais para cima.

III-4. Edite os arquivos `lab03.vhd` e `lab03_tb.vhd` e adicione-os ao projeto. Vamos começar analisando o arquivo `lab03.vhd`, que descreve a entidade `lab03`.

```

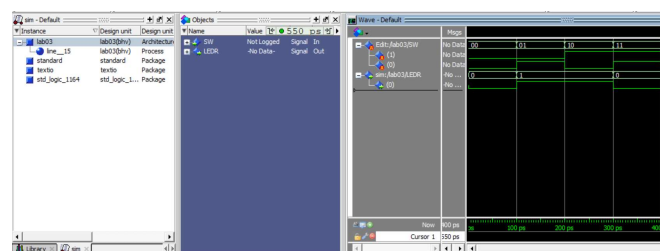
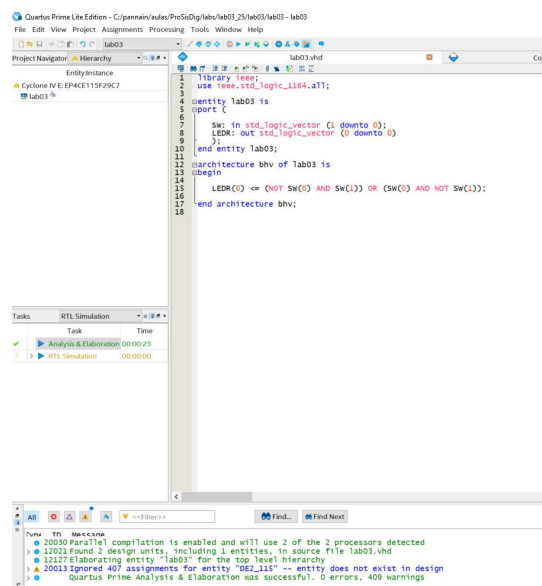
1  library ieee;
2  use ieee.std_logic_1164.all;
3
4  entity lab03 is
5  port (
6
7      SW: in std_logic_vector (1 downto 0);
8      LEDR: out std_logic_vector (0 downto 0)
9  );
10 end entity lab03;
11
12 architecture bhv of lab03 is
13 begin
14
15     LEDR(0) <= (NOT SW(0) AND SW(1)) OR (SW(0) AND NOT SW(1));
16
17 end architecture bhv;
18

```

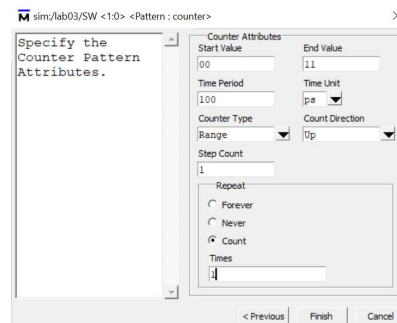
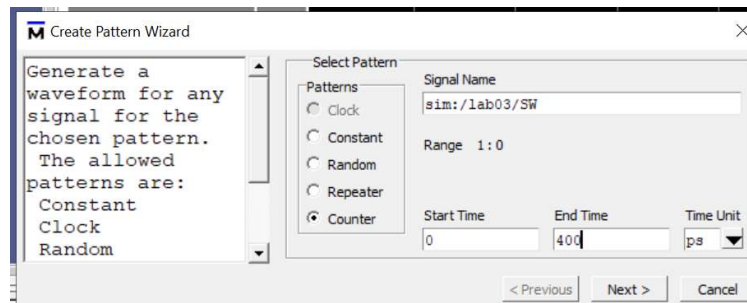
Responda (para você mesmo, não é preciso entregar nada), pesquisando sobre o que achar necessário esclarecer melhor:

- Em que periféricos da placa DE2-115 ela está conectada?
- Qual a saída esperada (em valores binários) da saída `LEDR0` em função das entradas `SW`?
- Qual o comportamento esperado do circuito quando gravado na placa DE2-115?

Faça a Analysis&Elaboration e depois RTL Simlutaion e verifique a existência ou não de inconsistências.



OBS: SW deverá ser editado como um contador, que começa em 00 e termina em 11, com 100 ps de período, contando uma vez.



Parte IV – Simulando um circuito com um Testbench

IV-1. Edite o arquivo `lab03_tb.vhd` e adicione-o ao projeto. O arquivo descreve a entidade `lab03_tb`, que é um *test bench* para a entidade `lab03`. Um *test bench* é um código em VHDL que instancia um circuito para teste e aplica nele conjuntos de entradas com o objetivo de observar as saídas.

```

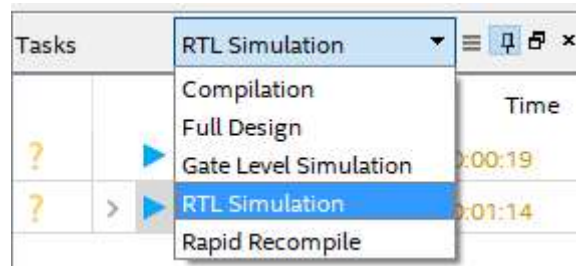
1  Library ieee;
2  use ieee.std_logic_1164.all;
3
4  entity lab03_tb is
5
6  end entity lab03_tb;
7
8  architecture tb of lab03_tb is
9
10     signal sw: std_logic_vector(1 downto 0);
11     signal ledr: std_logic_vector(0 downto 0);
12
13     begin
14         uut: entity work.lab03 port map( SW => sw, LEDR => ledr);
15     end architecture tb;

```

Responda (para você mesmo, não é preciso entregar nada), pesquisando sobre o que achar necessário esclarecer melhor:

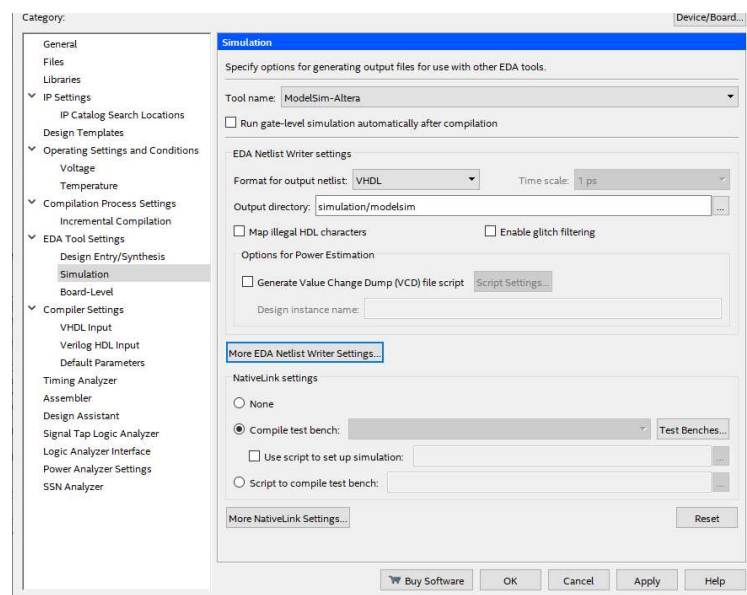
- Por que a entidade `lab03_tb` não tem entradas e saídas?
- Como é feita a instanciação da entidade `lab03` dentro do *test bench*?
- O que significa a palavra-chave `work` ao lado do nome da entidade instanciada?
- Quais são os padrões de entrada aplicados aos sinais `SW` ?
- Quais são as saídas esperadas para o teste?

IV-2. Para simular o *test bench*, precisamos primeiro configurá-lo. Selecione a opção **RTL Simulation** no menu drop-down do painel **Tasks**. Se o painel **Tasks** não estiver visível, ative-o em **View > Utility Windows > Tasks**.



IV-3. Expanda a opção **RTL Simulation** dentro do painel (não no menu drop-down) e dê um clique duplo em **Edit Settings**. Você verá a mesma janela **Settings** do item II-5.

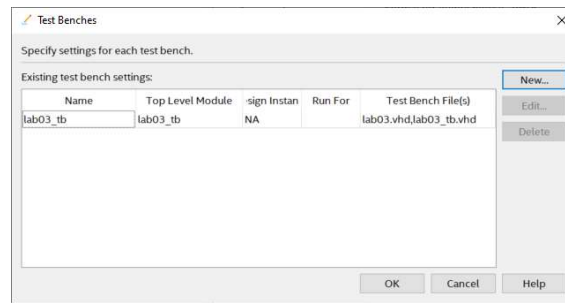
- Configure **Tool name** como *ModelSim-Altera*.
- Em **EDA Netlist Writer settings**, configure **Format for output netlist** como *VHDL*.
- Em **NativeLink settings** marque a opção **Compile test bench**.



IV-4. Clique no botão **Test Benches...** ao lado do menu drop-down **Compile test bench**. Na janela **Test Benches** que aparecerá, clique em **New...**. Você verá a janela **New Test Bench Settings**.

- Em **Test bench name** dê um nome para o seu teste, por exemplo, *lab03_tb*.
- Em **Top level module in test bench** dê o nome da entidade que efetuará o teste (não daquela que está sendo testada), no caso, *lab03_tb*.
- Em **Simulation period**, deixe marcado *Run simulation until all vector stimuli are used*. Isso significa que o simulador parará automaticamente quando os sinais do código de teste pararem de variar. Note que, alternativamente, você pode especificar um tempo em **End simulation at**.
- Em **Test bench and simulation files** adicione os arquivos necessários para o teste, no caso *lab03.vhd* e *lab03_tb.vhd*, **nesta ordem**.

- e) Confirme tudo clicando em OK nas janelas **New Test Bench Settings**, **Test Benches** e **Settings**.





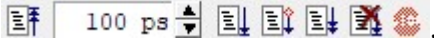
Dicas:

- Em projetos maiores, você pode (e deve) criar um *test bench* diferente para cada unidade a ser testada.
- Os *test benches* existentes no projeto vão estar listados no menu drop-down **Compile test bench** da aba **Simulation** das configurações do projeto. Para simular outro, é só escolher outro e confirmar.
- Ao adicionar arquivos em um *test bench*, procure adicionar apenas aqueles que são realmente necessários para o teste que será feito, sem colocar entidades que não serão utilizadas.

IV-5. Para executar a simulação, selecione a opção **RTL Simulation** no menu drop-down do painel **Tasks** e dê um clique duplo em **RTL Simulation** dentro do painel. O Quartus executará automaticamente a operação **Analysis & Elaboration** (você verá uma barra de progresso) e, logo após, abrirá o simulador ModelSim para simular o projeto. Se ao invés de ver o ModelSim você vir uma mensagem de erro, confira a configuração de caminho para o ModelSim altera conforme o item **I-4**. A janela do ModelSim se parecerá com a imagem abaixo. Se painéis estiverem faltando na sua visualização, selecione **Layout > Reset**.

IV-6. Tome um tempo para analisar a janela do ModelSim e entender o que os controles fazem. Algumas atividades interessantes que podem ser úteis são:

- Acompanhar possíveis erros e warnings de simulação no painel inferior. Em VHDL, warnings são muito importantes e várias vezes indicam o que há de errado em um projeto que “não funciona”.
- Diminuir e aumentar o zoom da visualização da onda de simulação nos botões com os ícones da lupa .
- Ir para a próxima ou anterior transição de um sinal, na onda, clicando sobre o nome dela e usando os botões .
- Alterar a base de visualização dos valores clicando com o botão direito sobre o nome do sinal e selecionando, em **Radix**, as opções para exibir os valores em decimal, hexadecimal, decimal sem sinal, etc.
- Expandir a visualização de vetores para exibirem múltiplos sinais binários.

- f) Reiniciar e re-executar a simulação com os botões **Restart** e **ContinueRun** em . Note que se a opção *Run simulation until all vector stimuli are used* (do item IV-4.c) não for utilizada ou os sinais de simulação nunca pararem de variar a simulação poderá rodar infinitamente. Clique no botão **Stop** para parar.
- g) Incluir sinais internos da unidade em teste na simulação, clicando e arrastando-os entre os painéis mais à esquerda para a janela que exibe a onda. Pode ser necessário reiniciar a simulação.

Note que existe algum problema recorrente no ModelSim que faz com que a simulação trave ou passe a exibir valores desconexos na onda de visualização sem motivo aparente. Isso geralmente acontece apenas na primeira visualização, logo após a janela do ModelSim aparecer. Por isso, quando estiver executando um teste e vir valores que não fazem sentido, antes de culpar o seu código, tente re-executar a simulação utilizando os botões do item IV-6.e para verificar se isso resolve o problema.

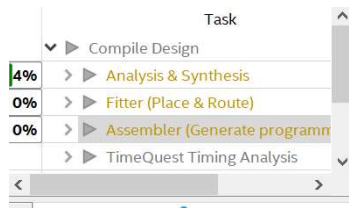
IV-7. Após analisar a simulação, responda (para você mesmo, não é preciso entregar nada), pesquisando sobre o que achar necessário esclarecer melhor:

- a) Os padrões de entrada são os que você esperava no item IV-1.f? Se não, por quê?
- b) Os padrões de saída são os que você esperava no item IV-1.g? Se não, por quê?

Parte V – Sintetizando e gravando na placa

V-1. Para sintetizar um circuito e gravar na placa, precisamos trocar o fluxo de trabalho para compilação (síntese). Em projetos de hardware, síntese e simulação são coisas muito diferentes. Na simulação, o código VHDL é tratado como um software, o que permite muitas construções que não são permitidas para síntese. Essas construções resultam em código “não-sintetizável”. É comum que desenvolvedores que começam a escrever o código apenas pensando em seu comportamento, e não em que tipo de módulos e dispositivos de hardware representa, acabem escrevendo código não-sintetizável. Também é comum que código que passe corretamente pelo fluxo de simulação nem consiga completar o fluxo de síntese ou, pior ainda, complete e, ao ser gravado na placa, apresente um comportamento inesperado ou que não faça o menor sentido. Por outro lado, utilizando simulação é muito mais fácil de analisar o comportamento do código ao ver sinais internos e suas relações. Por isso, os dois fluxos são importantes.

Para executar a síntese, de volta a janela principal do Quartus (o Modelsim pode ser fechado), selecione a opção **Compilation** no menu drop-down do painel **Tasks** e dê um clique duplo em **Assembler (Generate programming files)**, após expandir o item **Compile Design** se necessário. Se o painel **Tasks** não estiver visível, ative-o em **View > Utility Windows > Tasks**.



V-2. O Quartus executará automaticamente os procedimentos de **Analysis & Synthesis**, **Fitter** e **Assembler**. O primeiro faz a análise do código descrito e a síntese, o segundo mapeia o resultado utilizando os componentes disponíveis na placa e o terceiro formata o resultado em um arquivo para gravação. É normal que essas tarefas demorem muito mais que uma compilação de software, mesmo para designs pequenos. Porém, caso esteja demorando MUITO tempo (mais de 10 ou 15 minutos), pode ser um indicativo de algo de (muito) errado com o código.

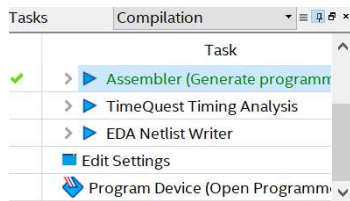
Se houver erros, verifique primeiro se a entidade top-level está configurada corretamente e o dispositivo utilizado é o correto, EP4CE115F29C7. Veja os itens **II-5** e **II-6**. Se os erros persistirem, verifique-os no painel inferior ou abrindo as pastas referentes a cada um dos procedimentos e selecionando **Messages** na aba **Compilation Report** que se abrirá ao iniciar a compilação.

V-3. Se não houver erros, com certeza haverá muitos warnings. Os warnings são exibidos de maneira agrupada no painel inferior e na aba de **Compilation Report**. Os warnings são muito importantes e ajudam muito a encontrar problemas no projeto. Não deixe de tomar um tempo para analisar os warnings e tentar entender o que eles significam.

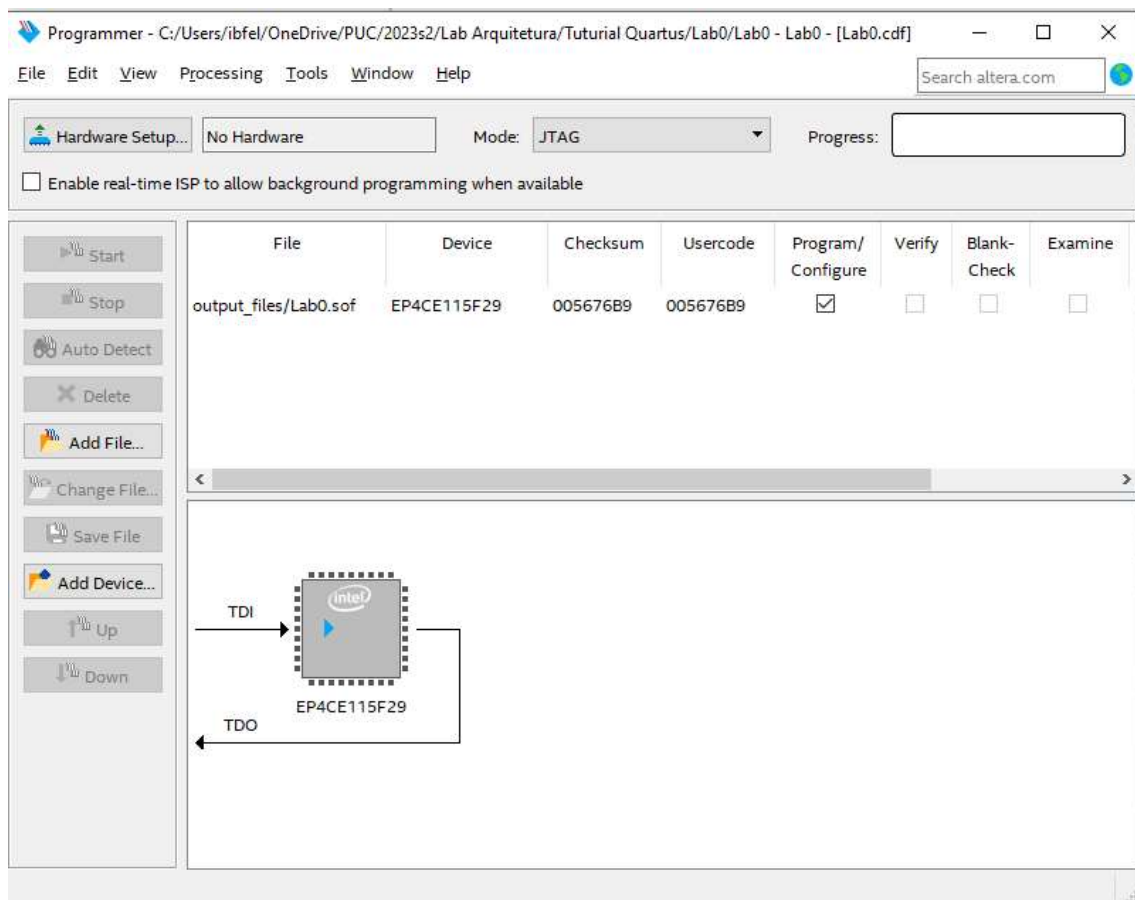
Em especial, preste **muita** atenção em resolver *warnings* que indiquem a existência de “**latches**” ou “**unsafe behavior**” no projeto. Esses são os *warnings* que têm o potencial de causar maior frustração ao analisar o comportamento de um circuito, após gravação na placa, e não devem ser ignorados. Latches são geralmente gerados por construções condicionais que não cobrem um caso contrário, por exemplo, `if` sem `else`, `when` sem um `else` ou `with/select` que não cubra todas as possibilidades. Note que não basta apenas haver um `else`, mas ele tem que efetivamente dar um valor para os sinais envolvidos quando a condição do `if/when` não é atingida.

V-4. Após executar o **Assembler**, o Quartus terá gerado os arquivos necessários para gravação do circuito na placa. Conecte a placa DE2-115 no computador e na tomada e ligue-a no botão vermelho. A placa deverá exibir um código de teste piscando leds e mostrando números sequenciais nos visores. Se isso não acontecer, **não** prossiga e peça ajuda ao professor.

V-5. Role um pouco o painel **Tasks** e dê um clique duplo em **Program Device (Open Programmer)**. Você verá a janela do gravador na placa.



Se na parte superior esquerda da janela você vir a mensagem “No Hardware”, conforme a imagem abaixo, primeiro verifique se a placa está corretamente conectada ao computador e ligada. Se estiver, clique em **Hardware Setup...** e dê um duplo no item que aparecerá na lista de **Available hardware items** e feche a janela clicando em **Close**. Se a lista estiver vazia, é provável que o driver da placa não esteja corretamente instalado (Windows) ou que o seu usuário não tenha permissão para acessá-lo (Linux).



V-6. Se tudo estiver correto, o botão **Start** deverá estar ativo e clicável. Basta clicar nele para gravar o circuito na placa.

Se houver problemas, verifique:

- Se a opção Mode está configurada como JTAG.
- Se o **Device** listado na parte superior e na figura da parte inferior está correto, ou seja, EP4CE115F29C7. Se não estiver, feche o **Programmer**, configure o device conforme os itens **II-5** e **II-6**, e recomece a parte **V**.

- c) Se há um arquivo *.sof listado na lista em **File**. Se não houver, verifique se o projeto foi corretamente compilado.
- d) Se o checkbox da coluna **Program/Configure** está ativo e marcado. Se não estiver marcado, marque. Se não estiver ativo, verifique as configurações a-c acima.

V-7. Verifique o funcionamento do circuito na placa e responda (para você mesmo, não é preciso entregar nada), pesquisando sobre o que achar necessário esclarecer melhor:

- a) O comportamento do circuito é o que você esperava no item **III-4.e**? Se não, por quê?
- b) Que alterações seriam necessárias para atingir o comportamento que era esperado, se já não foi atingido?
- c) O comportamento do circuito é o mesmo demonstrado pela simulação no item **IV-7**? Se não, por quê?

Para entregar: (arquivos vhds – projeto e testbench)

Laboratório 04

Tutorial da Ferramenta Quartus – Uso da Placa

Este laboratório é um tutorial para familiarização de uso da placa.

- 1 - Sintetize um full adder na placa. As entradas devem ser SW0, SW1 e SW2. As saídas LDR0 e LDR1.
- 2 - Sintetize, na placa, um full adder, utilizando os displays de 7 segmentos para mostrar as entradas e as saídas do somador.