

Sistemas Reconfiguráveis

Eng. de Computação

Profs. Francisco Garcia e Antônio Hamilton Magalhães

Aula 2 – Plataforma de desenvolvimento

Desenvolvimento de projetos

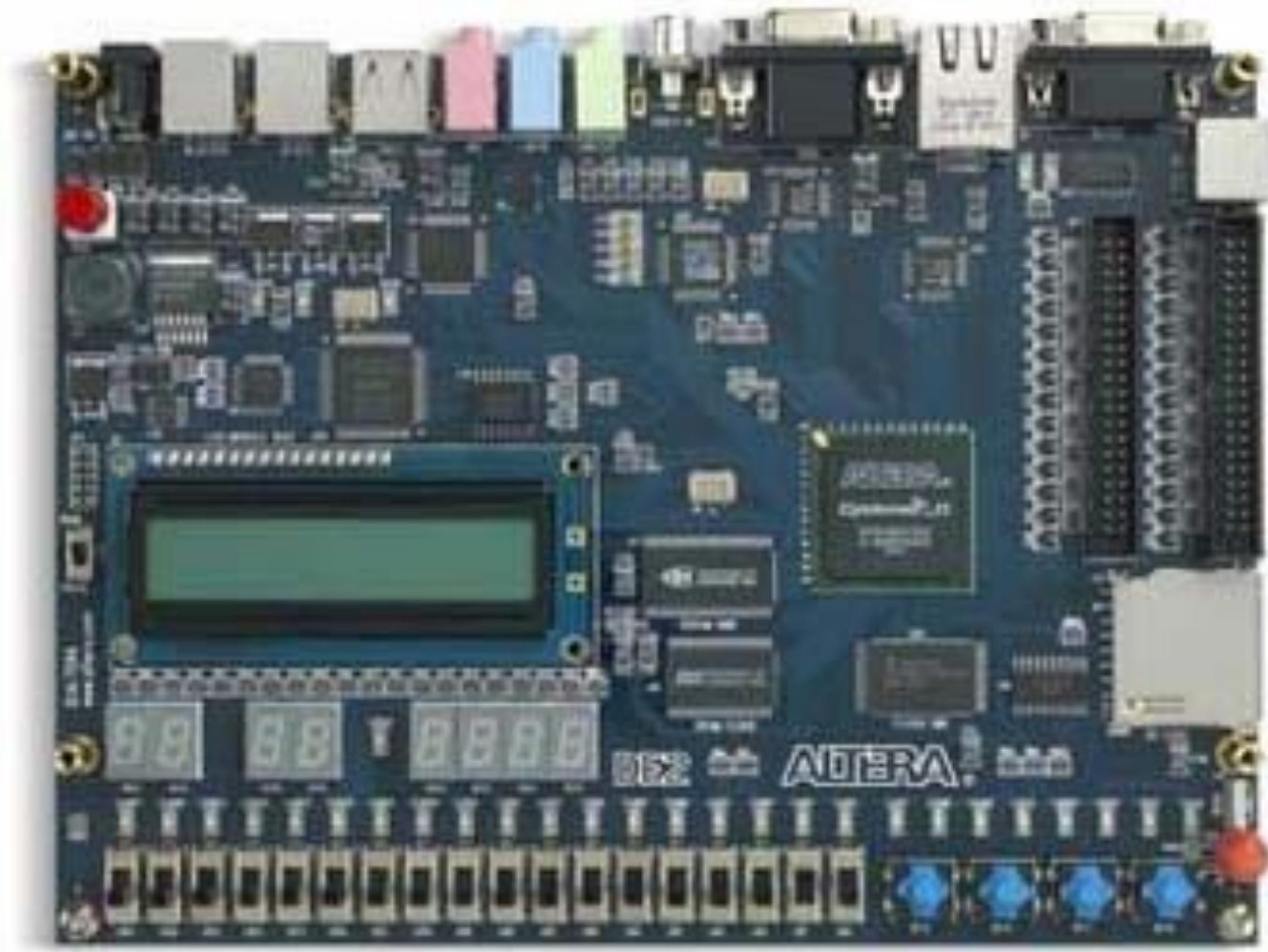


Na disciplina Sistemas Reconfiguráveis, os exercícios e trabalhos serão baseados em pacotes de *hardware* e *software* disponíveis no laboratório:

- *Hardware*: Módulo DE2, fornecido por Terasic
- *Software*: Quartus II (versão 9.1sp2), fornecido por Altera (atualmente Intel)

Para realização dos exercícios e trabalhos, o aluno deverá ter o *software* Quartus II instalado.

Módulo DE2



Especificações principais:

- FPGA Cyclone II EP2C35F672C6 FPGA e dispositivo de configuração serial EPCS16;
- Interface USB Blaster embutida na placa para programação e API de controle de usuário;
- Suporte para modos JTAG e AS;
- Memória SDRAM de 8Mbyte (1M x 4 x 16);
- Memória SRAM de 512K byte (256K X16);
- Memória Flash de 4Mbyte;
- Soquete de cartão SD;
- 4 chaves tipo push-button e 18 chaves de duas posições;
- 9 LEDs verdes + 18 vermelhos;
- Módulo LCD com 16 x 2 caracteres;
- Osciladores de clock de 50-MHz e 27-MHz;

continua...

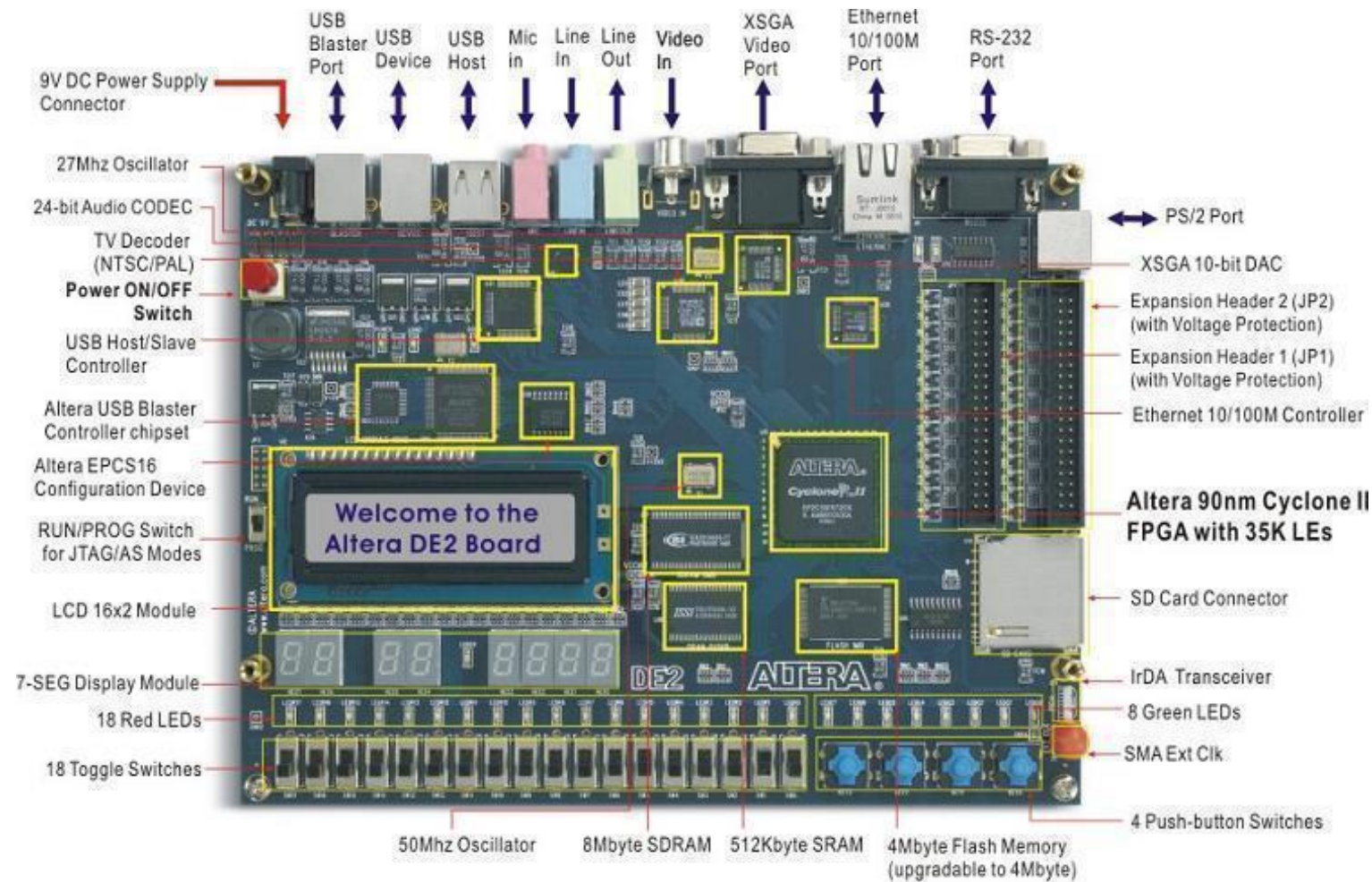
Módulo DE2



continuação

- CODEC de áudio com qualidade de CD de 24-bit, com 3 conectores: entrada de linha, entrada de microfone e saída de linha;
- Saída de vídeo com conector VGA e conversor digital analógico triplo com 10 bits;
- Conector de entrada de vídeo analógico e decodificador NTSC/PAL;
- Controlador ethernet 10/100 com soquete RJ45;
- Conectores USB tipo A e tipo B com controlador host/slave;
- Conector DB9 para comunicação serial RS-232;
- Conector para mouse e teclado padrão PS/2;
- Transceptor infravermelho;
- 2 conectores de expansão com 40 pinos e diodos de proteção.

Módulo DE2



Altera (Intel) EP2C35F672C6

Especificações principais:



- 33.216 elementos lógicos (*logic elements* - LE)
- Cada LE tem uma tabela de pesquisa (*look-up table* - LUT), com quatro entradas, que é um gerador de funções que pode implementar qualquer função de quatro variáveis, e um registrador programável (flip-flop)
- 105 blocos de RAM M4K (483.840 bits)
- 35 multiplicadores 18x18 bits, que podem ser transformados em 70 multiplicadores de 9x9 bits
- 4 PLL
- 475 pinos de entrada/saída (input/output - I/O), em 8 bancos
- Cada banco de I/O pode ser configurado para um padrão elétrico diferente
- Dispositivos com velocidade -6, -7, -8

Software Quartus II



O *software* Quartus II é usado em todas as etapas do projeto:

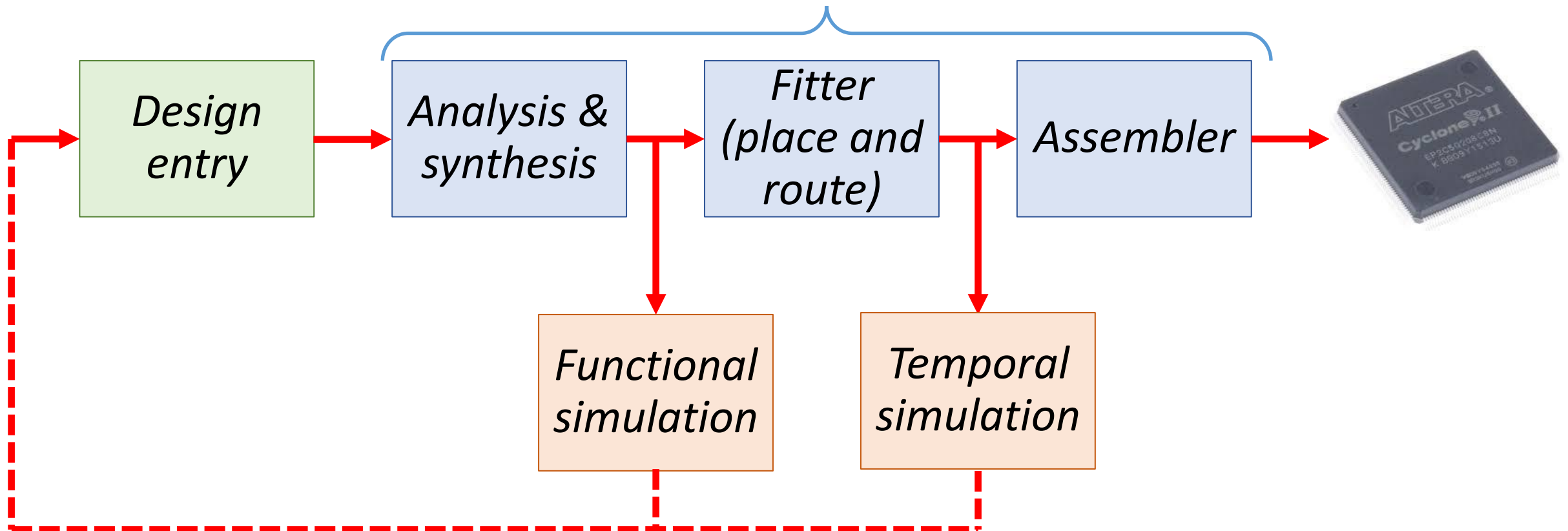
- *Design entry*
- *Analysis & synthesis*
- *Fitter (place and route)*
- *Assembler*
- *Functional simulation*
- *Temporal simulation*
- *Configuration*

Software Quartus II

Fluxo de um projeto

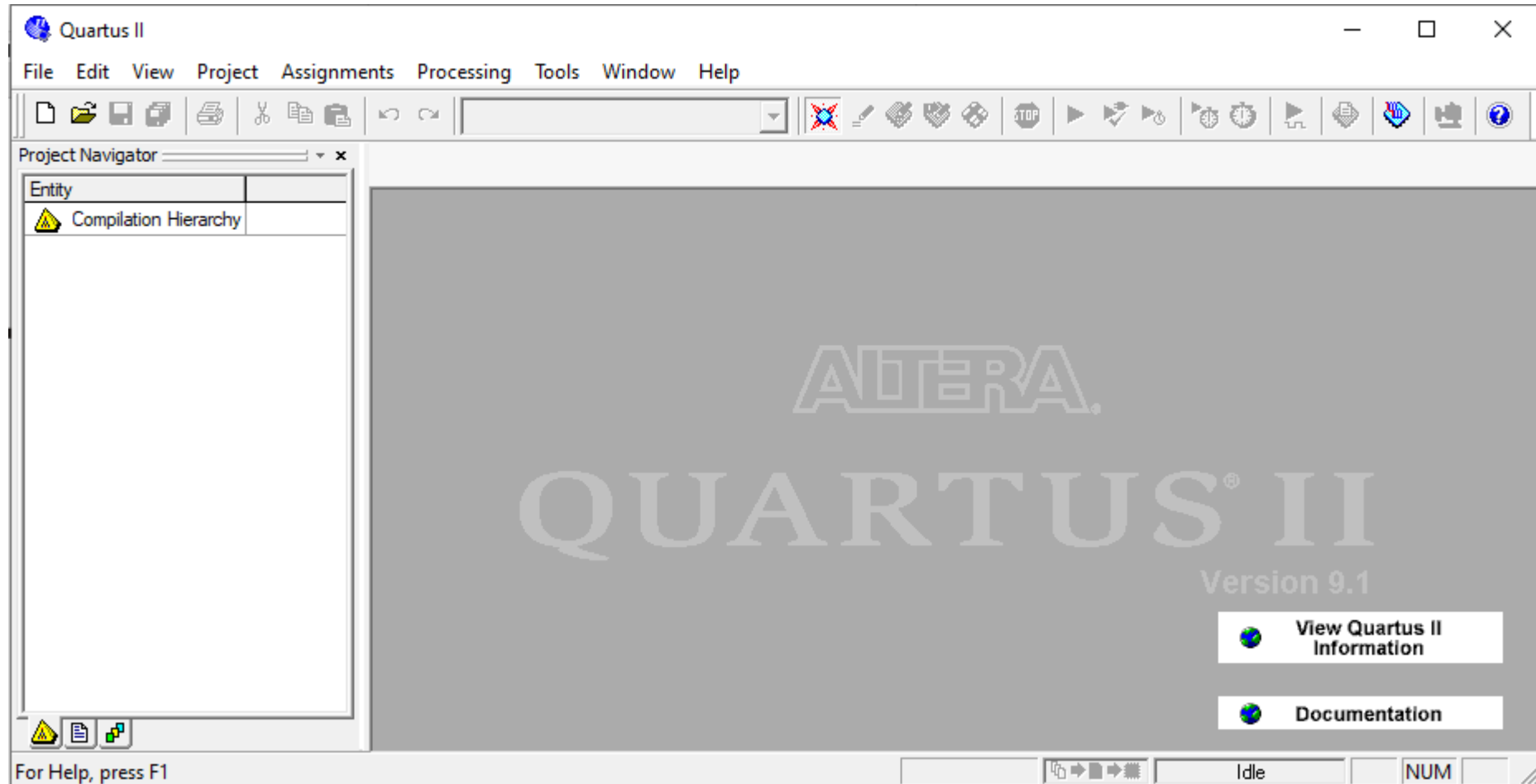
Compile

Configuration



Software Quartus II

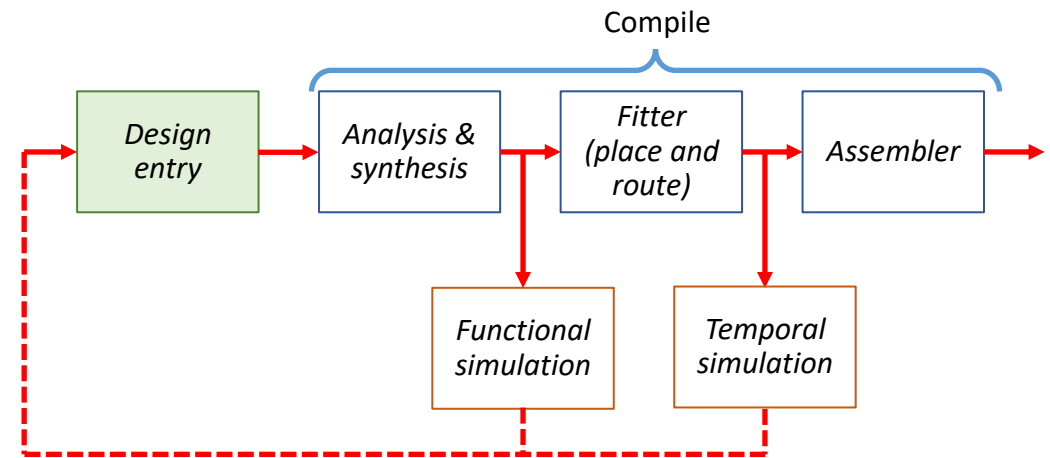
Tela inicial do Quartus II:



Quartus II

Design entry

- Gráfico (diagrama esquemático)
- Texto (linguagem de descrição de *hardware* - HDL)
 - VHDL
 - Verilog
 - AHDL

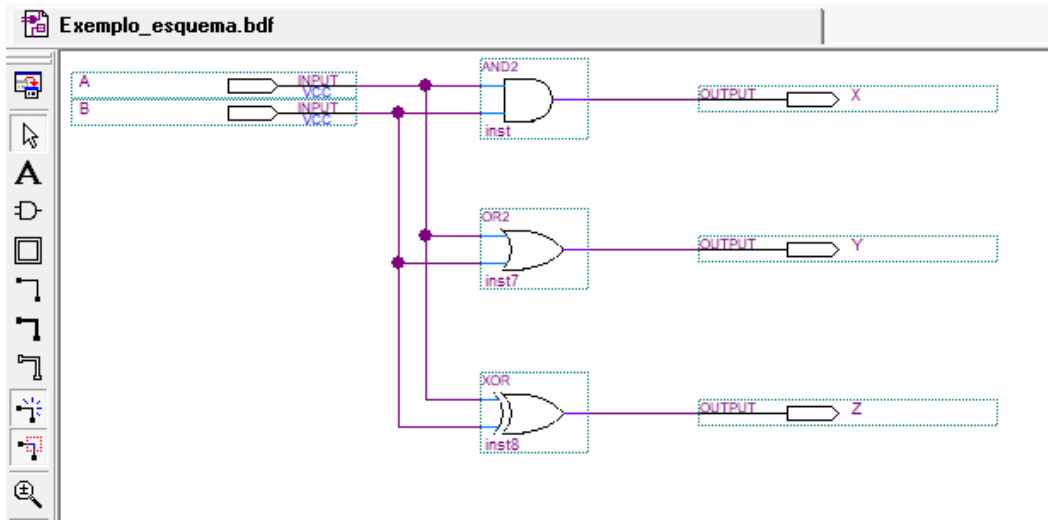


Obs.:

- Projetos simples podem ser descritos com um único arquivo de *design entry*.
- Projetos mais complexos, no entanto, podem ser descritos em vários arquivos de *design entry*, chamados módulos ou blocos funcionais, para facilitar o desenvolvimento e teste. Isso é chamado de **projeto hierárquico**, que pode ter vários níveis de hierarquia.

Quartus II

Design entry - gráfico



Design entry – texto (VHDL)

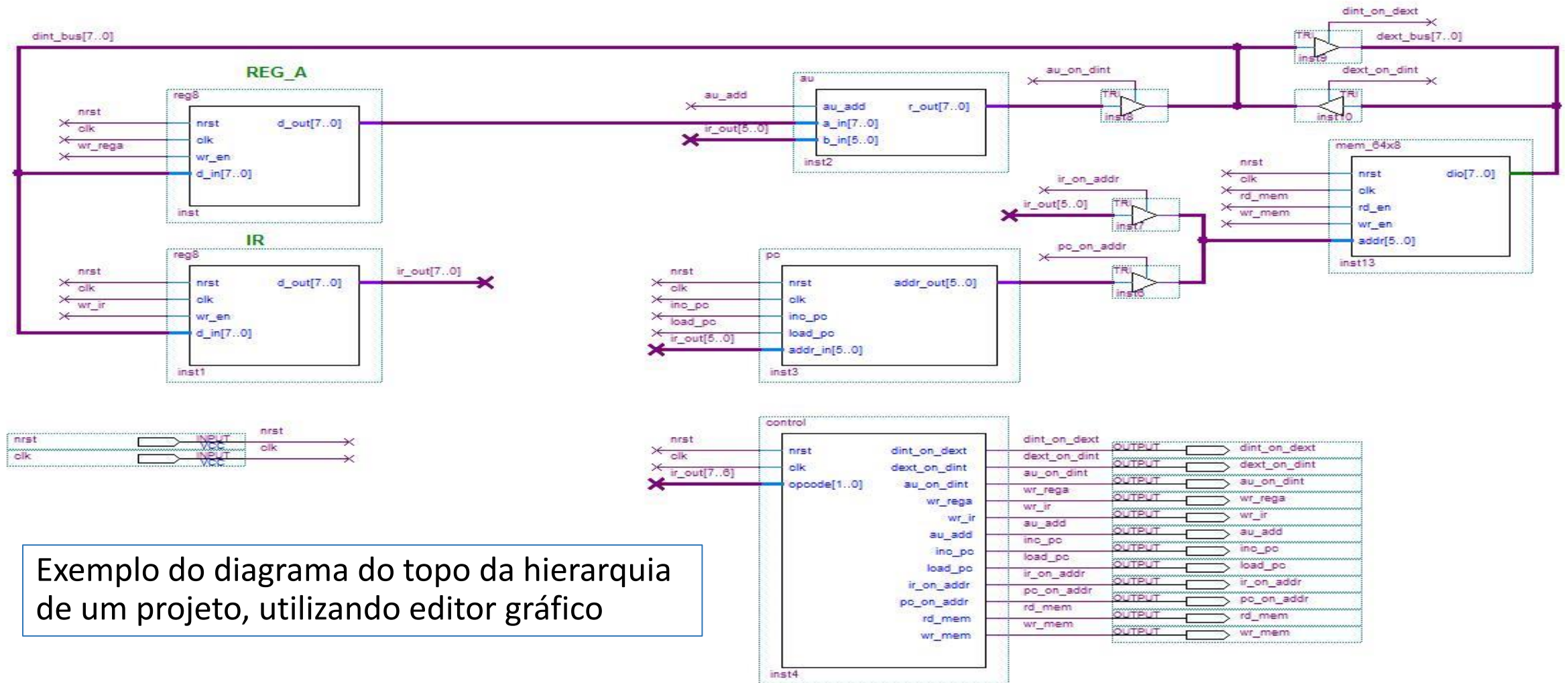
The screenshot shows the Quartus II text design entry interface for a file named 'Exemplo_linguagem.vhd*'. The code defines an entity named 'Exemplo_linguagem' with two input ports 'a' and 'b' of type 'BIT', and three output ports 'x', 'y', and 'z' of type 'BIT'. The architecture 'arch' implements the logic: 'x' is the AND of 'a' and 'b', 'y' is the OR of 'a' and 'b', and 'z' is the XOR of 'a' and 'b'.

```
1  ENTITY Exemplo_linguagem IS
2    PORT (
3      a,b : IN BIT;
4      x, y, z : OUT BIT
5    );
6  END ENTITY;
7  ARCHITECTURE arch OF Exemplo_linguagem IS
8  BEGIN
9    x <= a AND b;
10   y <= a OR b;
11   z <= a XOR b;
12  END arch;
```

Obs.:

- Os blocos de um projeto hierárquico são interligados em um arquivo chamado de **topo da hierarquia**.
- Não é necessário haver uma uniformidade na forma de descrever os diversos blocos (inclusive o topo) em um projeto hierárquico. Uns podem usar linguagem de descrição de *hardware* e outros diagrama esquemático.

Quartus II – exemplo

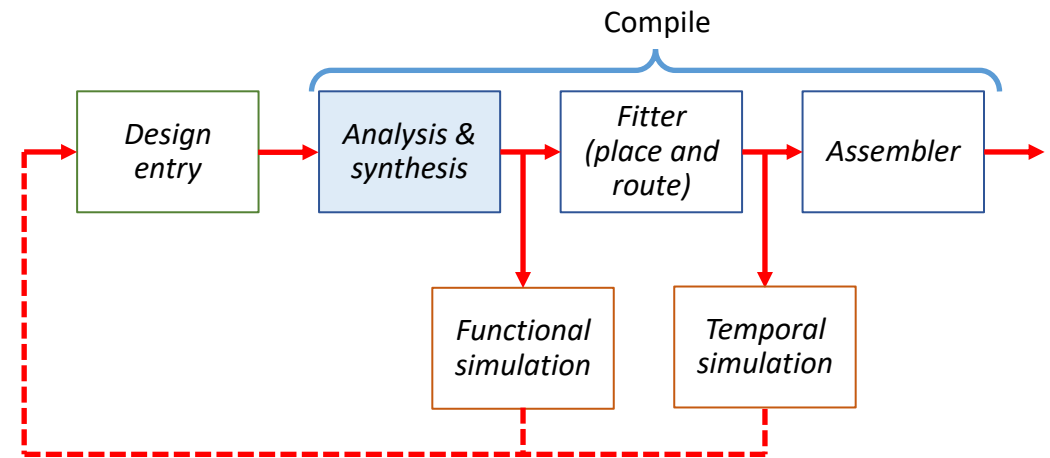


Exemplo do diagrama do topo da hierarquia de um projeto, utilizando editor gráfico

Quartus II

Analysis & synthesis

- Verifica erros
- Simplifica as equações visando:
 - Área (número de elementos lógicos) ou
 - Tempo (atrasos)
- Implementa a lógica desejada, usando os recursos do dispositivo alvo, como os elementos lógicos
- Cria a base de dados do projeto

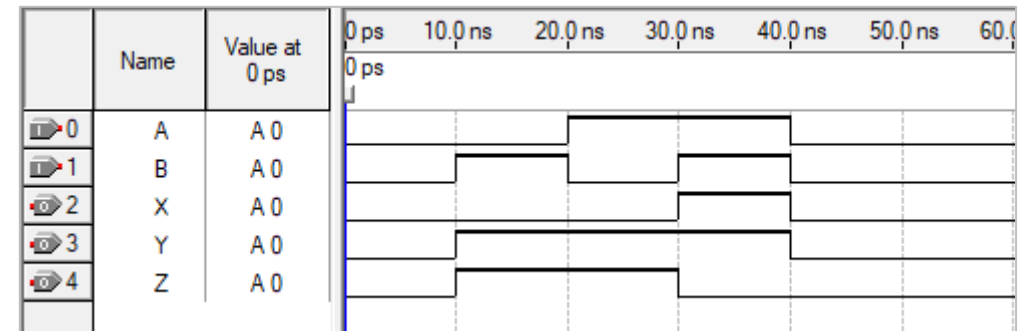
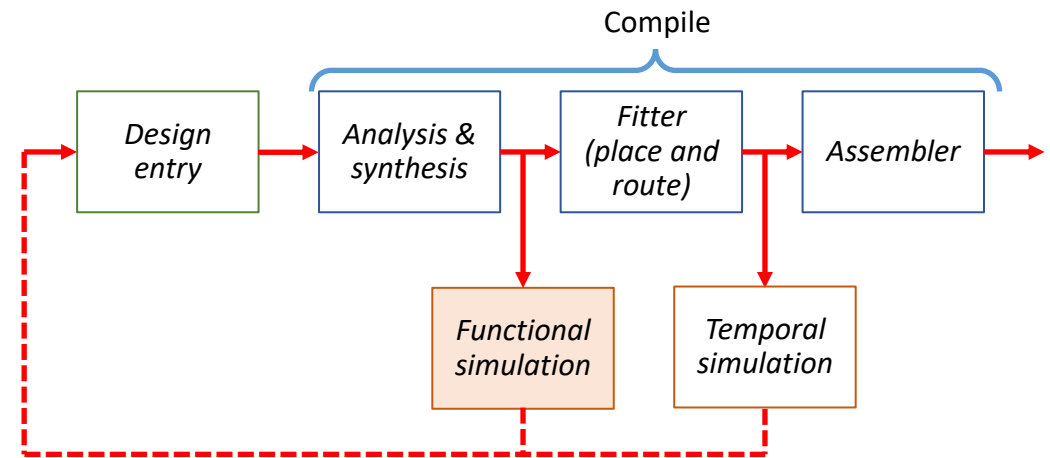


Obs.: Como parte do processo de simplificação e otimização, circuitos lógicos e nós podem ser mudados ou removidos.

Quartus II

Functional simulation

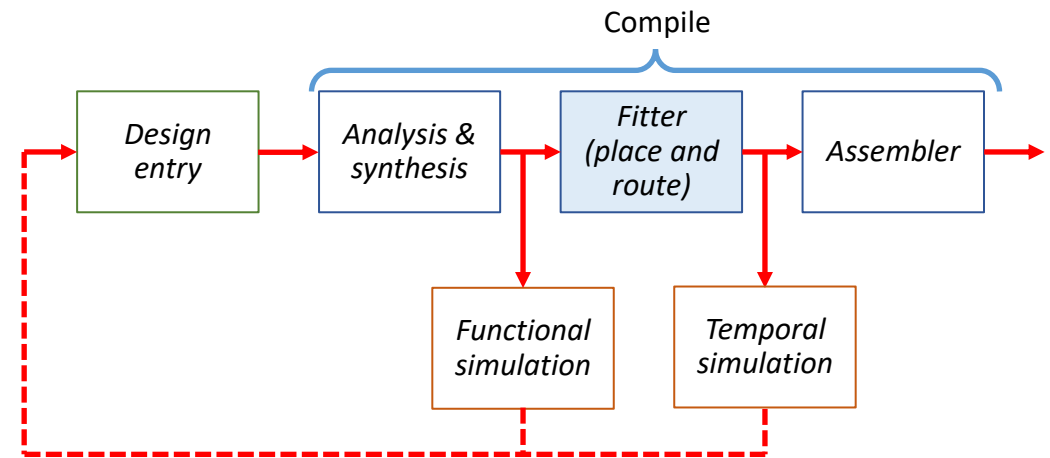
- Gera o resultado (resposta) de acordo com a excitação (estímulos) especificados.
- Faz a simulação sem considerar atrasos de propagação (ideal)



Quartus II

Fitter (place and route)

- Ajusta os requisitos de lógica e temporização do projeto com os recursos disponíveis do dispositivo.
- Aloca cada função lógica para a célula com melhor localização para roteamento e temporização;
- Seleciona os caminhos de interconexões e especificações de pinos apropriados.



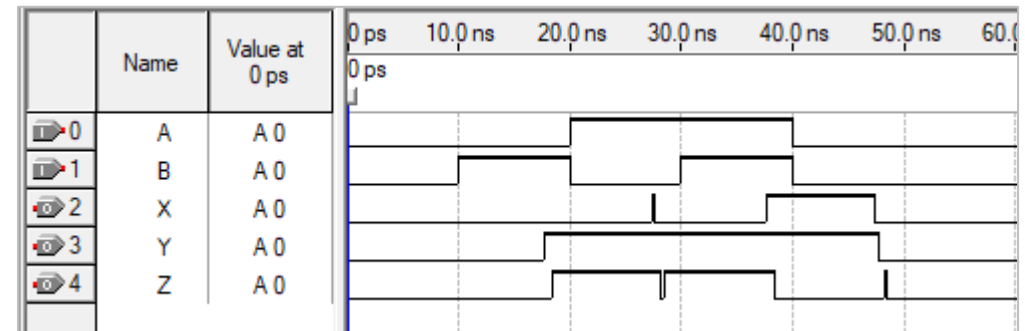
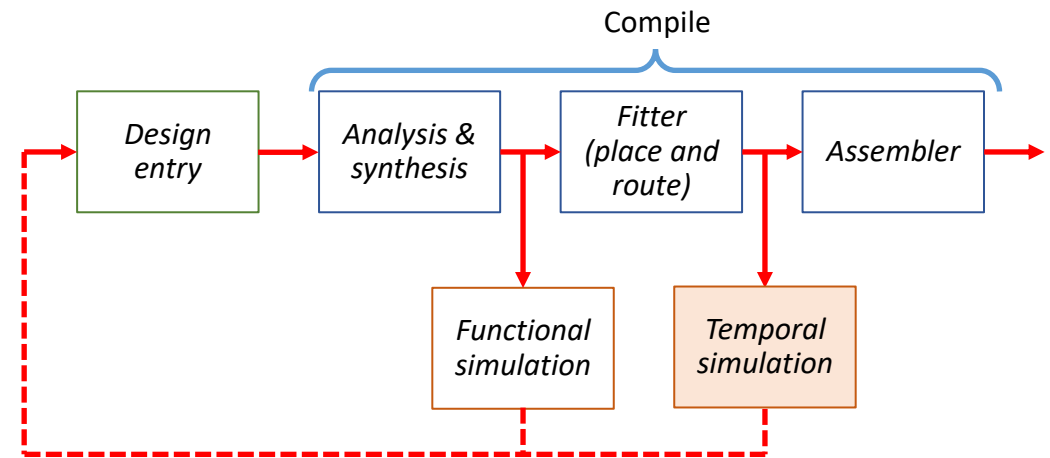
Obs.:

- Em projetos mais complexos, essa etapa pode levar muito tempo para ser executada, pois a ferramenta testa inúmeras possibilidades para tentar satisfazer os requisitos de tempo especificados.

Quartus II

Temporal simulation (timing analysis)

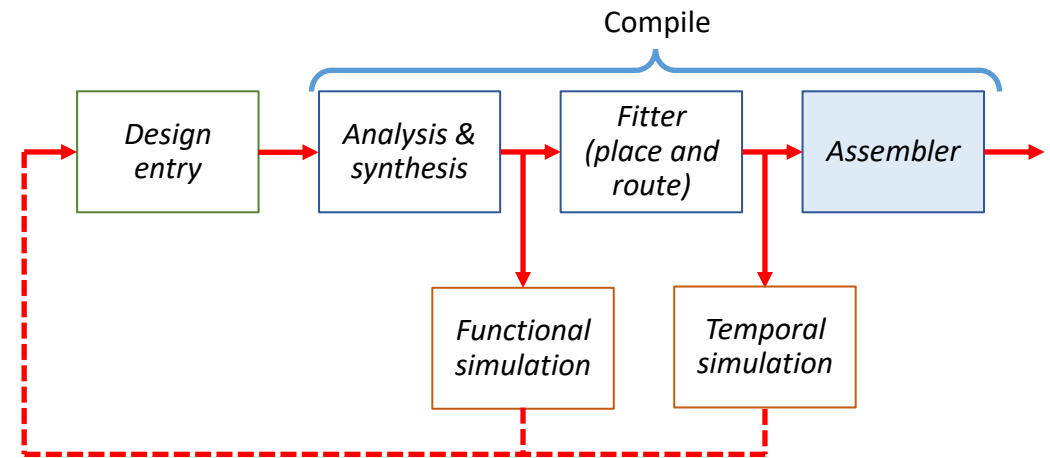
- Gera o resultado (resposta) de acordo com a excitação (estímulos) especificados.
- Faz a simulação considerando os atrasos de propagação, de acordo com um modelo para cada dispositivo específico.



Quartus II

Assembler

- Gera o arquivo que será gravado no dispositivo



Exercício



Vamos fazer agora um exercício utilizando, como forma de *design entry*, o editor gráfico.

- Inicialmente, é necessário criar, em seu computador, uma pasta para o projeto. Vamos usar o nome “Cap2ex1” para a pasta desse projeto. Essa pasta pode estar dentro de outra pasta conveniente para a organização de seu computador, desde que de acordo com as observações abaixo.

Obs.:

- Use uma pasta exclusiva para cada projeto
- O nome da pasta, bem como todas as outras pastas no caminho, **não pode conter** espaço nem caracteres especiais, tais como “ç” ou letras acentuadas. Para separar palavras, pode ser usado o caractere “_”
- O nome do projeto ou dos arquivos não pode começar com um número

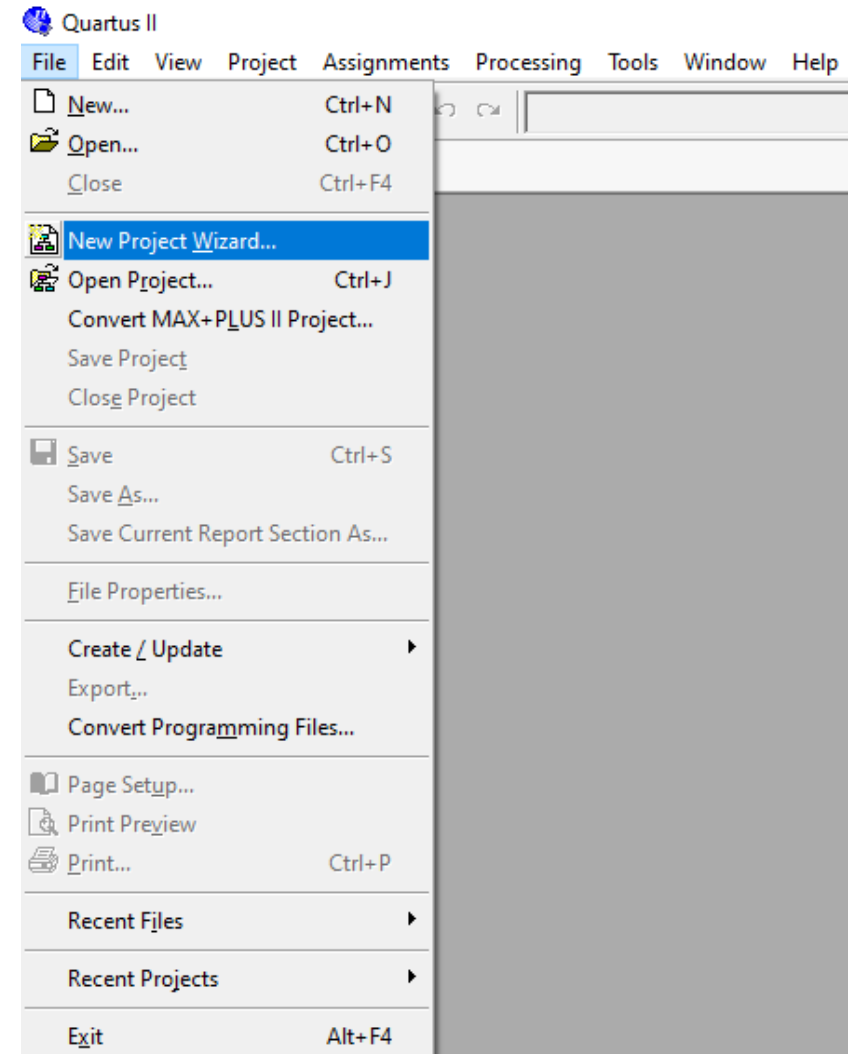
Exercício - criação do projeto

Em seguida, deve ser criado o projeto (*project*):


- No menu “File”, selecionar a opção “New Project Wizard...”

Obs.:

- Inicialmente, o software irá criar um arquivo “Cap2ex1.qpf”, com as informações do projeto.
- Posteriormente serão criados vários outros arquivos na pasta do projeto, com o mesmo nome, porém com diferentes extensões.

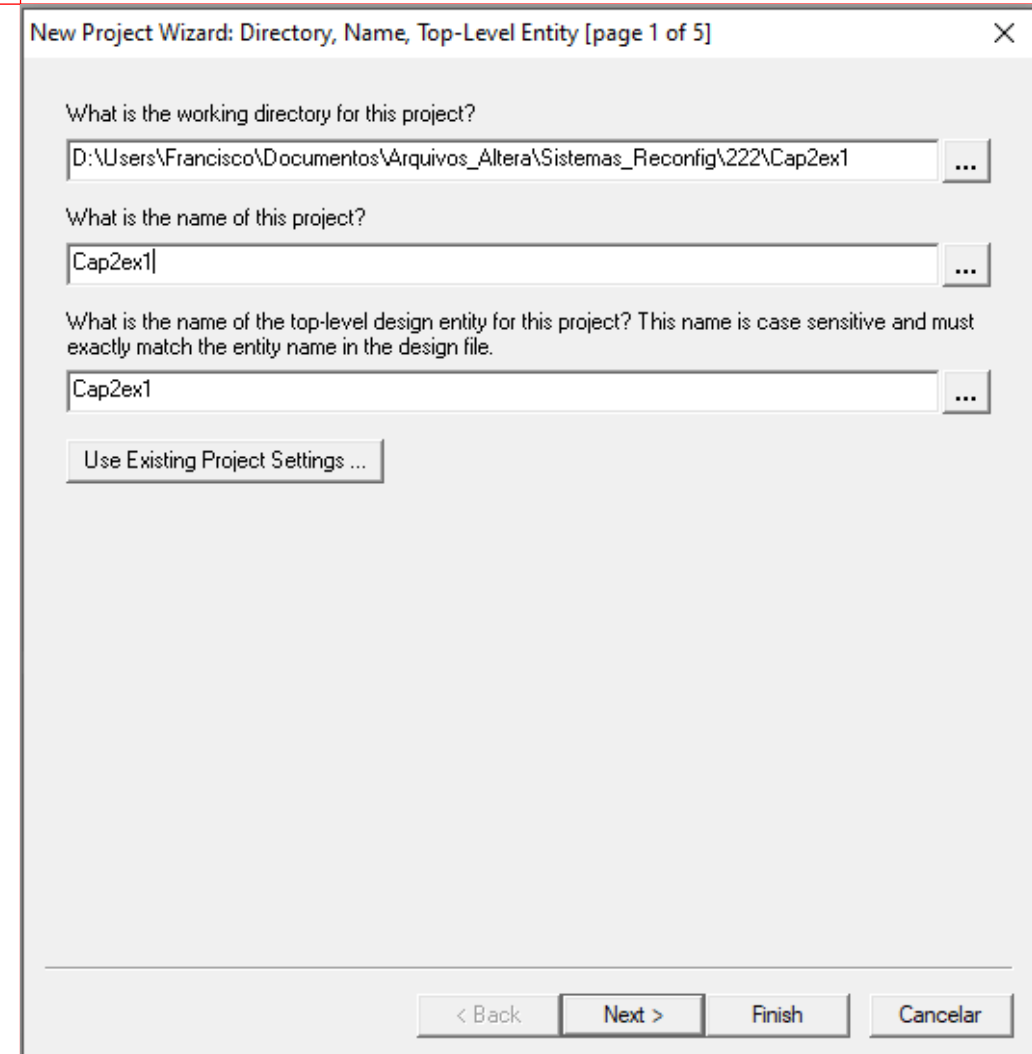


Exercício - criação do projeto

- Na página 1, no primeiro campo, digitar o nome da pasta do projeto ou, usando o botão  apontar para a pasta do projeto;
- no segundo campo, digitar o nome do projeto. Vamos usar também o nome “Cap2ex1” para o projeto;
- o terceiro campo será preenchido automaticamente, com o nome do arquivo topo da hierarquia do projeto.

Obs.:

- Nesse exercício, os nomes da pasta, do projeto e do arquivo são os mesmos. Isso não é obrigatório, mas desejável, pois facilita a organização



New Project Wizard: Directory, Name, Top-Level Entity [page 1 of 5]

What is the working directory for this project?

D:\Users\Francisco\Documentos\Arquivos_Altera\Sistemas_Reconfig\222\Cap2ex1

What is the name of this project?

Cap2ex1

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.

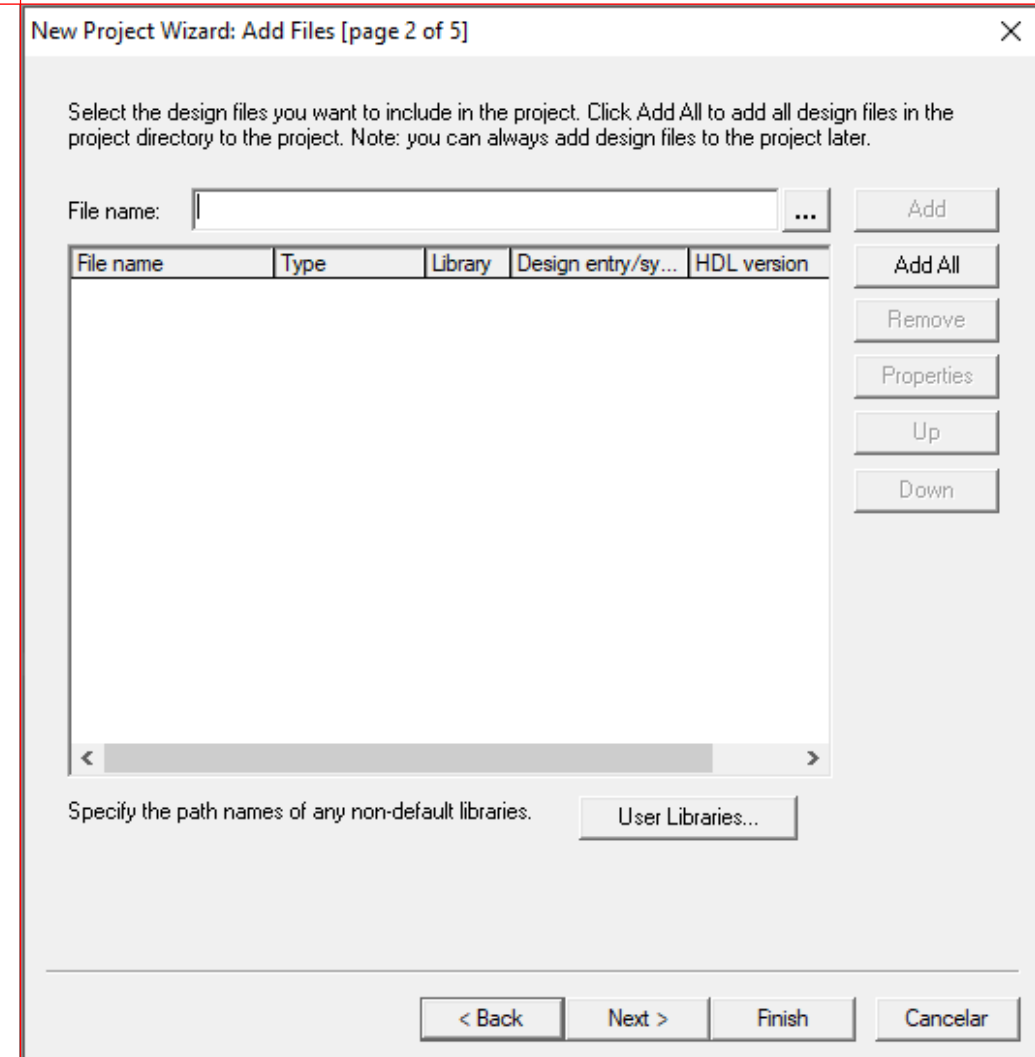
Cap2ex1

Use Existing Project Settings ...

< Back Next > Finish Cancelar

Exercício - criação do projeto

- Na página 2 é solicitado o nome de outros arquivos que fazem parte do projeto. Esse exercício é feito com apenas um arquivo de *design entry*, ou seja, não é um projeto hierárquico. Portanto, clicar em “Next”.



Exercício - criação do projeto

- Na página 3 deve ser configurado o dispositivo alvo para a compilação:
 - Family: Cyclone II ← Device family
 - Package: FBGA
 - Pin count: 672
 - Speed grade: 6
 - Device: EP2C35F672C6 (é o primeiro da lista)

Show in “Avaliable device” list
- Na página 4 não há o que configurar para esse exercício e a página 5 é apenas um resumo. Então, para encerrar, clicar em “Finish”.

Obs.:

- Esse mesmo procedimento deverá ser feito para todos os projetos do semestre.

New Project Wizard: Family & Device Settings [page 3 of 5]

Select the family and device you want to target for compilation.

Device family

Family: Cyclone II

Devices: All

Target device

☐ Auto device selected by the Filter

☒ Specific device selected in 'Available devices' list

Show in 'Available device' list

Package: FBGA

Pin count: 672

Speed grade: 6

☒ Show advanced devices

☐ HardCopy compatible only

Available devices:

Name	Core v...	LEs	User I/...	Memor...	Embed...	PLL
EP2C35F672C6	1.2V	33216	475	483840	70	4
EP2C50F672C6	1.2V	50528	450	594432	172	4
EP2C70F672C6	1.2V	68416	422	1152000	300	4

Companion device

HardCopy:

☒ Limit DSP & RAM to HardCopy device resources


< Back Next > Finish Cancelar

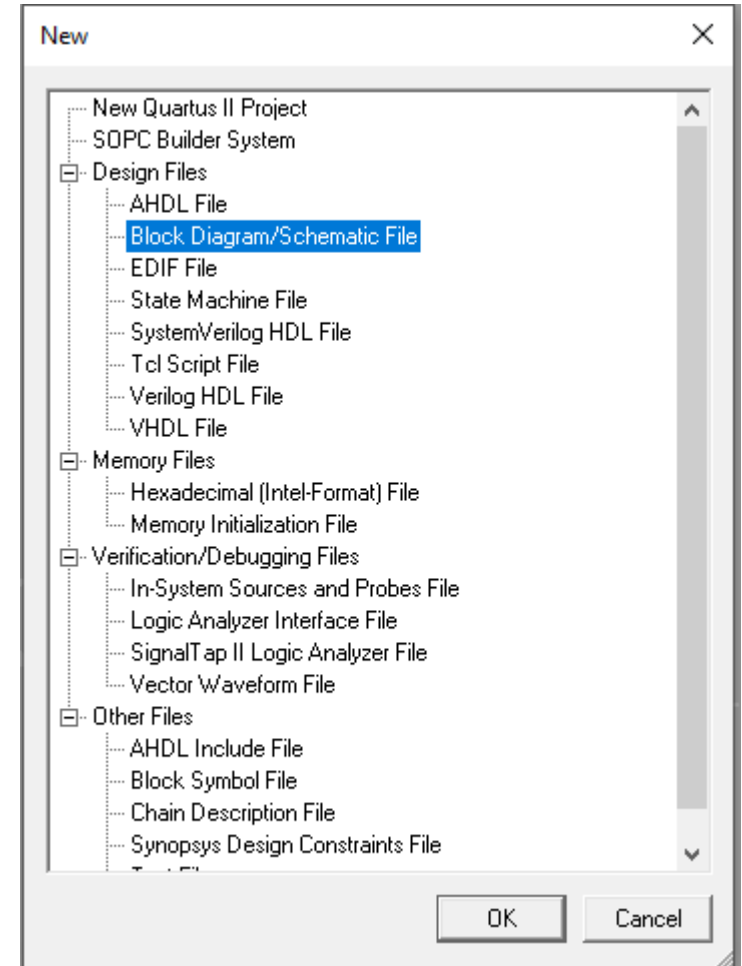
Exercício - criação do projeto

Agora vamos criar o arquivo para a edição do circuito do exercício:

- No menu “File”, escolher a opção “New”;
- na janela aberta, selecionar a opção “Block Diagram/Schematic File”;
- salvar esse arquivo usando o nome “Cap2ex1” (a extensão “.bdf” será adicionada automaticamente)

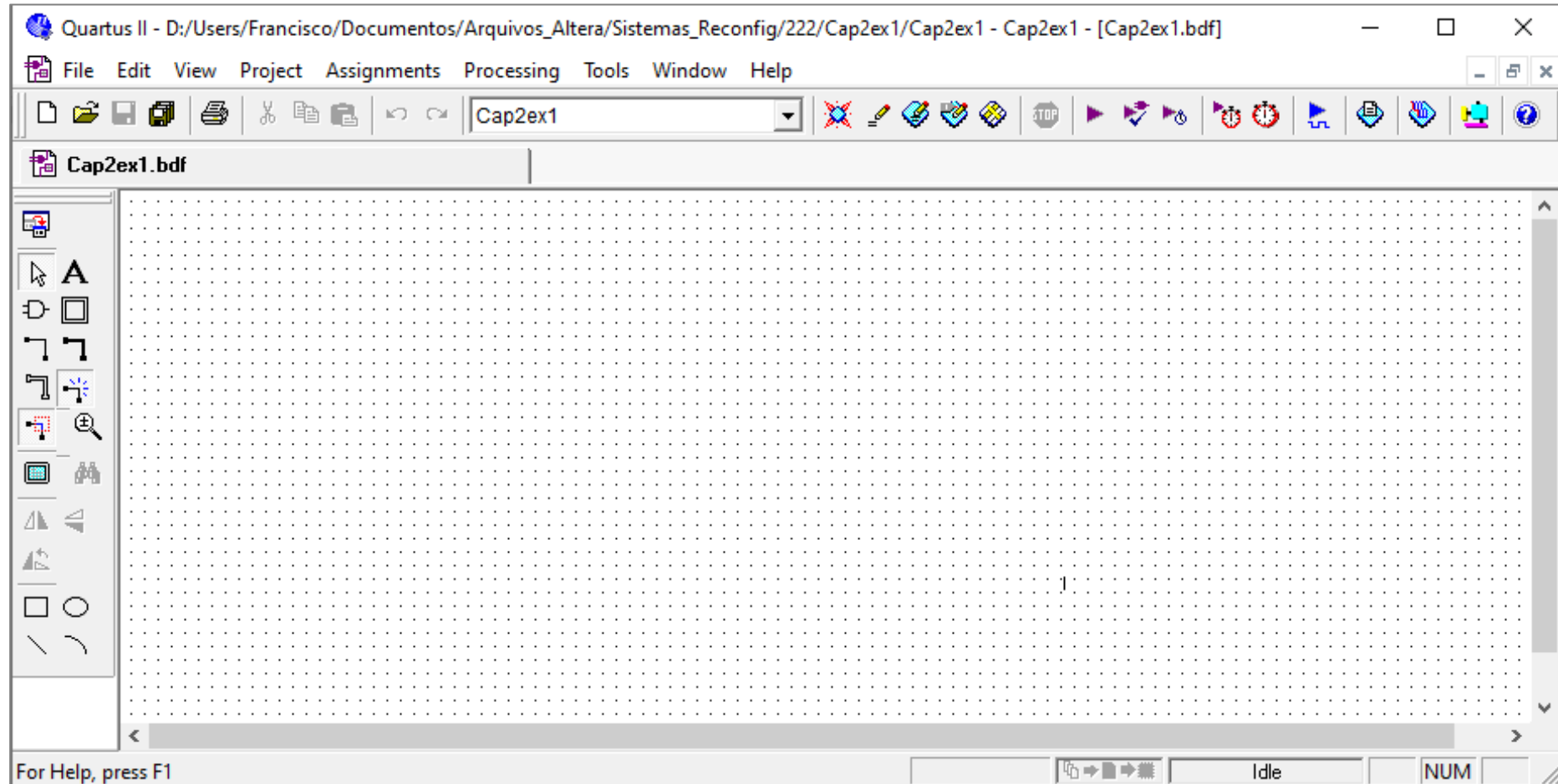
Obs.:

- Para abrir o novo arquivo, pode também ser usado o atalho “Ctrl+N” ou o botão 




Exercício - criação do projeto


Ao lado é mostrada a área de trabalho vazia, com a barra de ferramentas no lado esquerdo

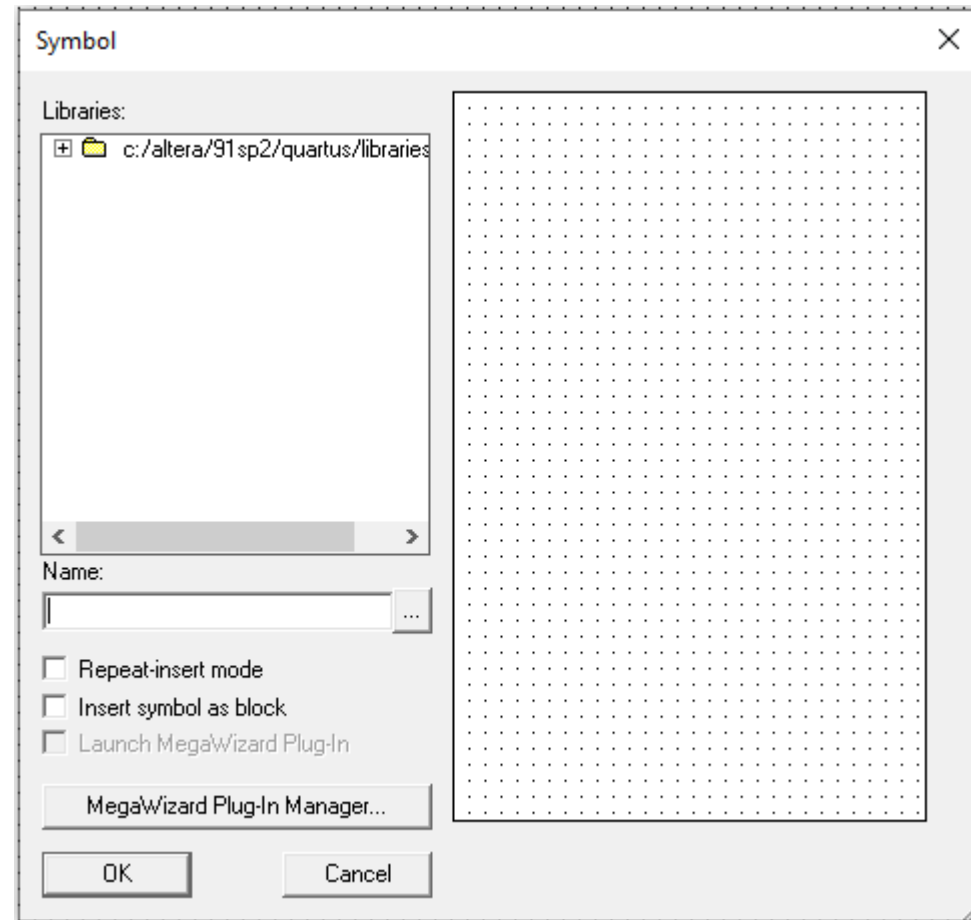


Exercício - edição do diagrama

- Com o apontador do *mouse* em qualquer lugar vazio dentro da área de trabalho, fazer um duplo clique (botão esquerdo do *mouse*) ou, então, clicar no botão  na barra de ferramentas;
- na janela aberta para seleção de símbolos, no campo “Name”, digitar “and2”. Clicar no botão “OK” e posicionar o símbolo na área de trabalho;
- repetir o procedimento acima para “or2” e “xor”.

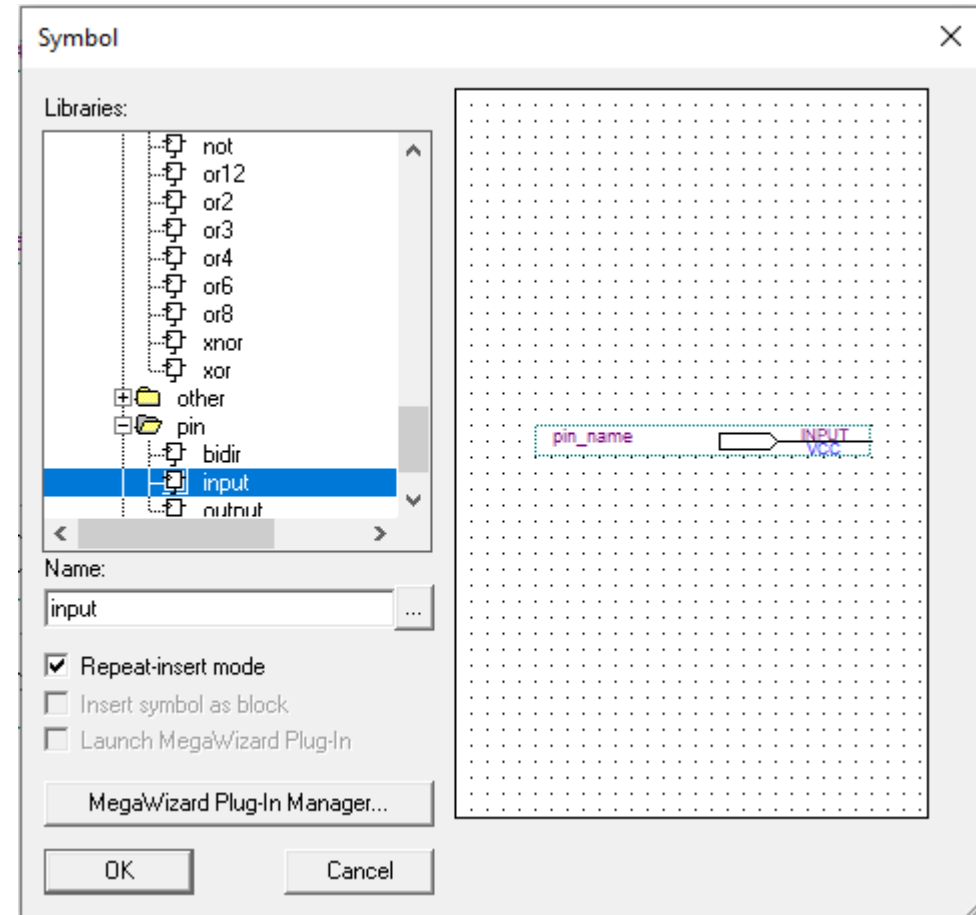
Obs.:

- Usando o botão  é possível procurar o símbolo desejado na biblioteca do *software*.



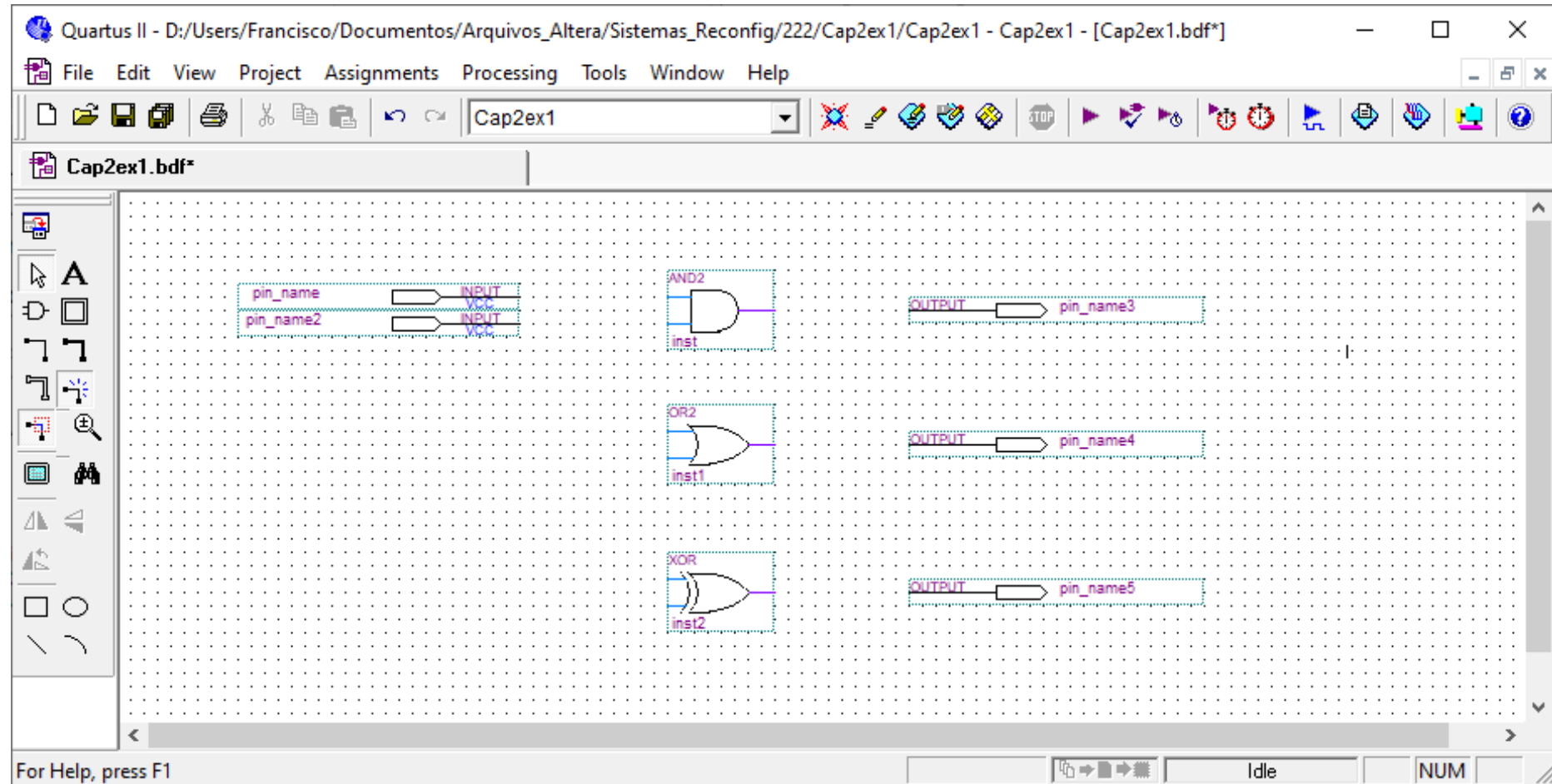
Exercício - edição do diagrama

- Repetir o procedimento anterior para inserir o símbolo “input”. Marcar a opção “Repeat insert mode”, para inserir o símbolo duas vezes. Para sair do modo de repetição, teclar “Esc”. Posicionar os símbolos à esquerda das portas lógicas.
- Fazer o mesmo para inserir três símbolos “output” à direita das portas lógicas



Exercício - edição do diagrama

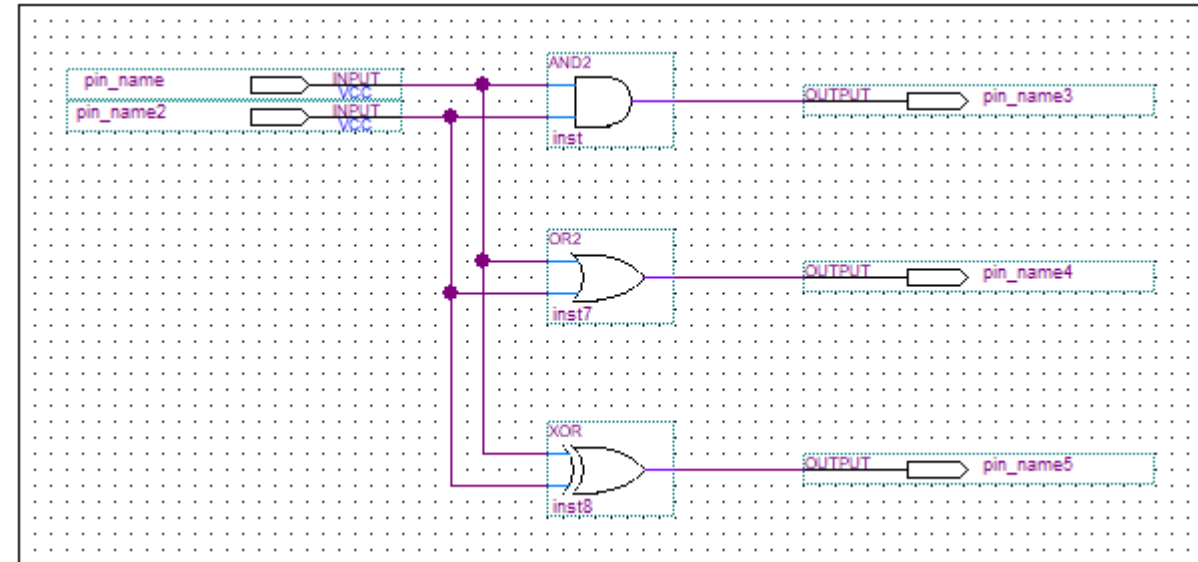
Após a inserção dos símbolos, a área de trabalho deve estar como mostrado ao lado:




Exercício - edição do diagrama

Fazer a conexão entre os símbolos. Para isso:

- Posicionar o apontador do *mouse* no terminal do símbolo. O cursor deverá mudar de seta para uma cruz;
- clicar e arrastar para fazer a ligação;
- repetir até ligar todos os símbolos conforme indicado.




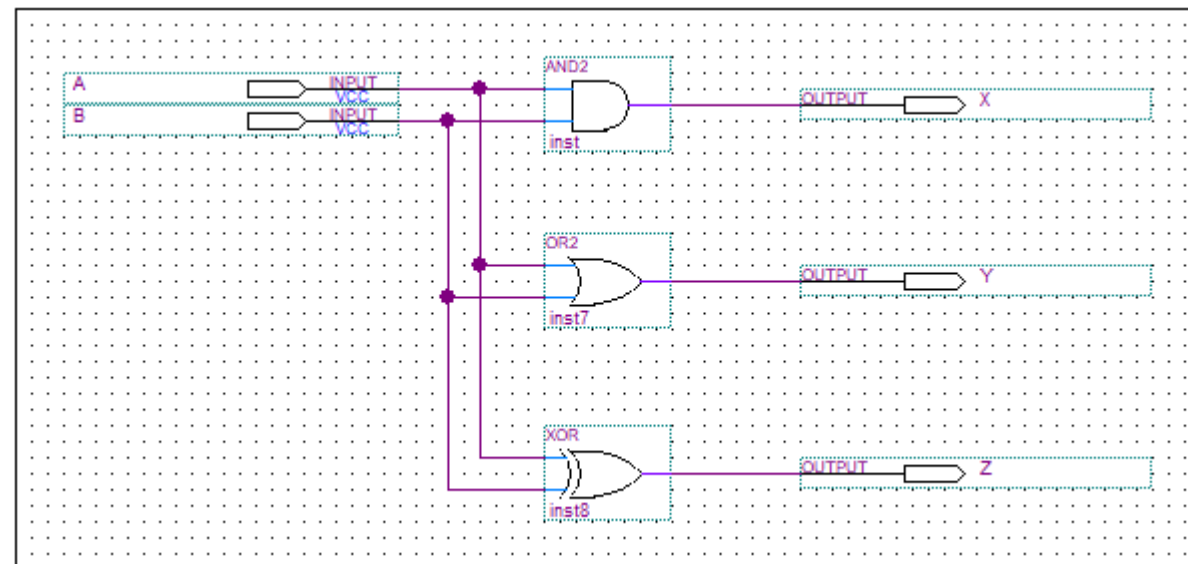
Obs.:

- Um símbolo poderá ser movido depois de feitas as ligações.
- Na barra de ferramentas, se o botão de *rubberbanding*  estiver ativado (para dentro), o símbolo será movido com todas as suas ligações. Caso contrário, o símbolo será movido e as ligações permanecerão no mesmo lugar.

Exercício - edição do diagrama

Vamos renomear as entradas para **A** e **B** e as saídas para **X**, **Y** e **Z**. Para isso:

- Em cada entrada e saída, dar um clique duplo com o mouse onde está escrito “pin_name” e editar para o nome correspondente, conforme mostrado ao lado;
- salvar o arquivo, usando o atalho “**Ctlr+S**” ou clicando no botão  ou, no menu “**File**” escolher a opção “**Save**”.




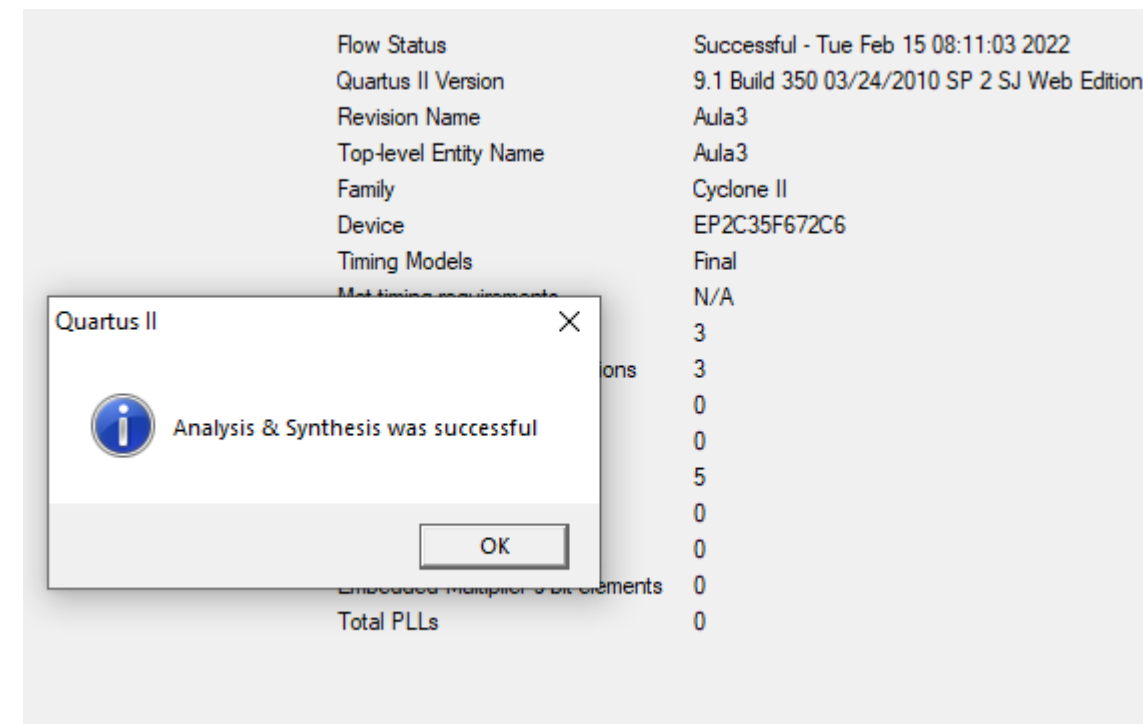
Obs.:

- Os nomes dos pinos são sensíveis a letras maiúsculas ou minúsculas.

Exercício - análise e síntese

Agora que todas as informações do arquivo com o diagrama esquemático já foram inseridas no mesmo, é possível fazer o teste do projeto através da **simulação funcional**. Para isso é necessário, antes, fazer o processo de **análise e síntese**.

- Clicar com o mouse no botão  da barra de comandos superior ou use o atalho “**Ctrl+K**”;
- após algum tempo, o processo de análise e síntese é completado.



Obs.:

- Outra forma de iniciar o processo de análise e síntese é: no menu “Processing” ativar a opção “Start” e depois “Start Analysis & Synthesis”.

Exercício – teste

Fazer um **teste** em um projeto é comparar o **resultado obtido** com o **resultado esperado**.

Em um projeto de lógica combinacional como o desse exemplo, devem ser testadas todas as combinações possíveis das entradas. Como são 2 entradas, devemos testar $2^2 = 4$ combinações.

Essas combinações estão mostradas na tabela ao lado.

- Preencher a tabela com o resultado esperado para esse exercício

Entradas		Saídas		
A	B	$X = A \cdot B$	$Y = A + B$	$Z = A \oplus B$
0	0			
0	1			
1	0			
1	1			

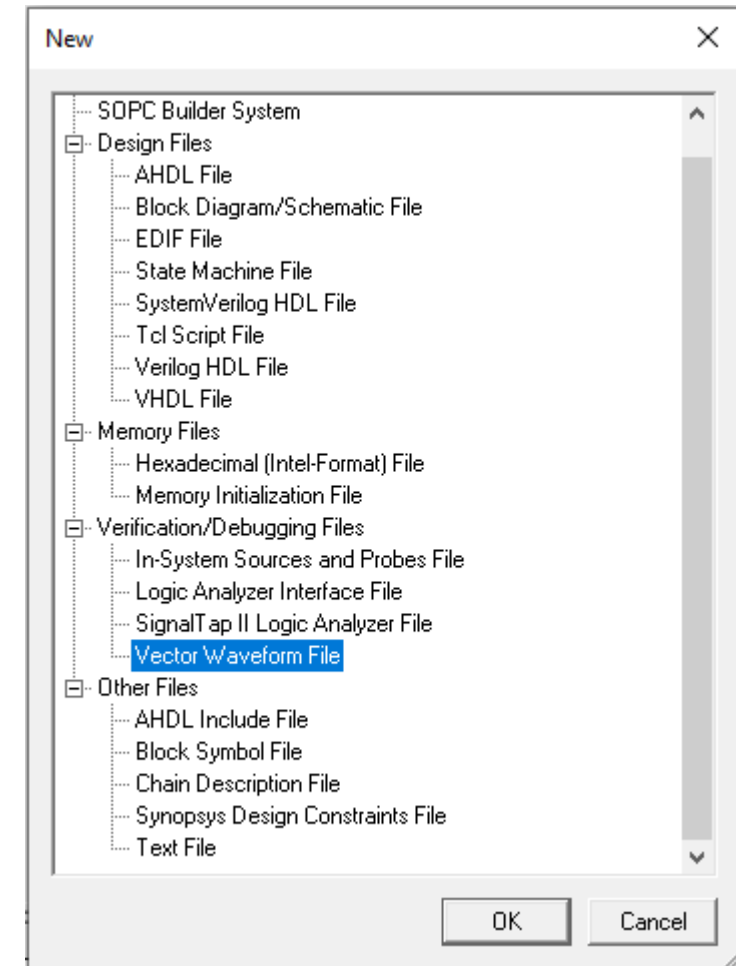
Obs.:

- Um teste onde não se conhece o resultado esperado não é um teste, é um **experimento**.

Exercício – simulação funcional

Agora vamos criar o arquivo para a simulação funcional que, nesse exercício, será feita através de um editor de forma de onda (*waveform*). Nesse arquivo serão programados os estímulos (excitação) para as entradas do projeto e, após o processamento, serão verificados os resultados nas saídas. O teste será concluído com êxito se o resultado obtido for igual ao resultado esperado.

- No menu “**File**”, escolher a opção “**New**”;
- na janela aberta, selecionar a opção “**Vector Waveform File**”;
- salvar esse arquivo usando o nome “**Cap2ex1**” (a extensão “**.vwf**” será adicionada automaticamente). A opção “**Add file to current project**” deverá estar selecionada.

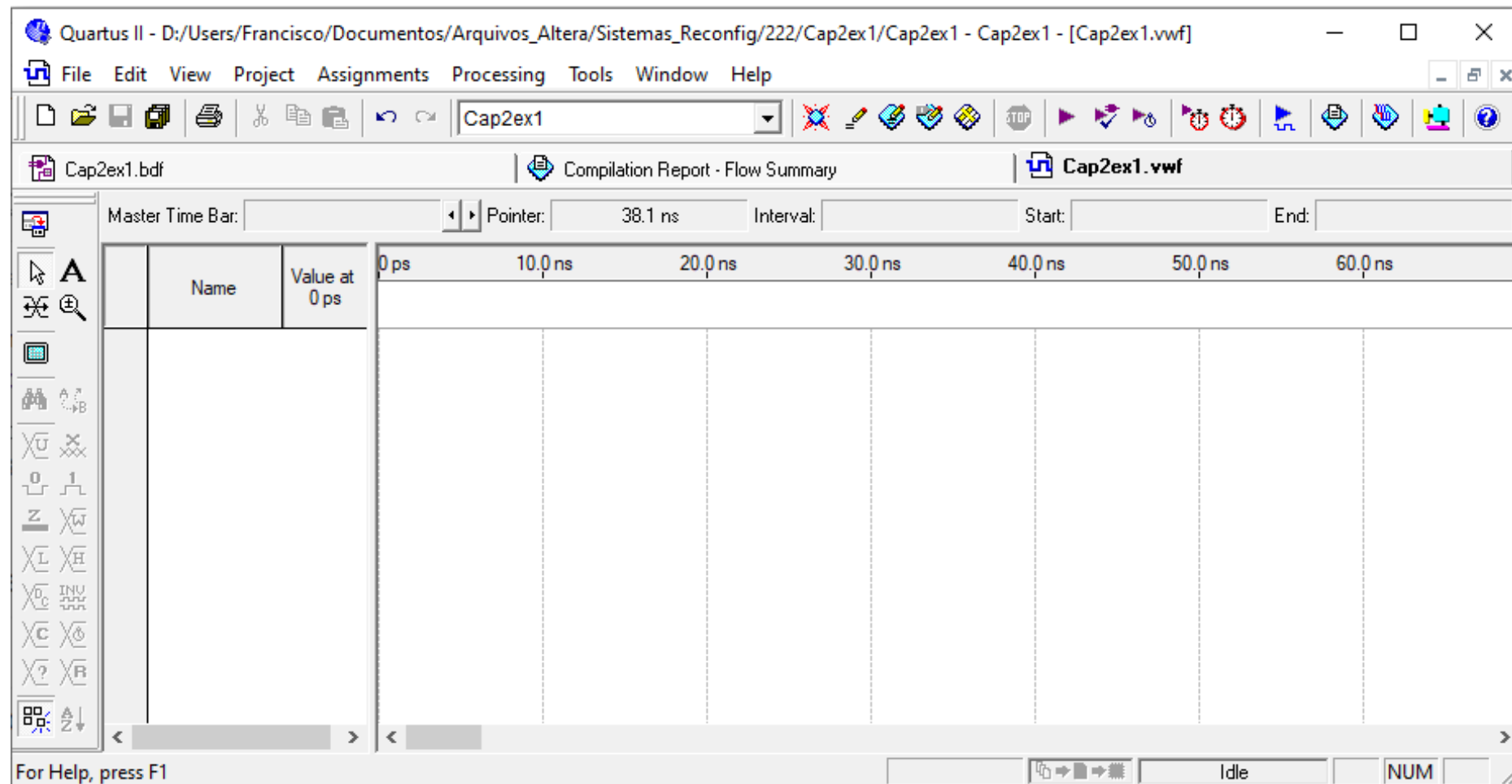


Exercício – simulação funcional

Após a criação do arquivo de simulação, a nova área de trabalho deve estar como mostrado ao lado:

Obs.:

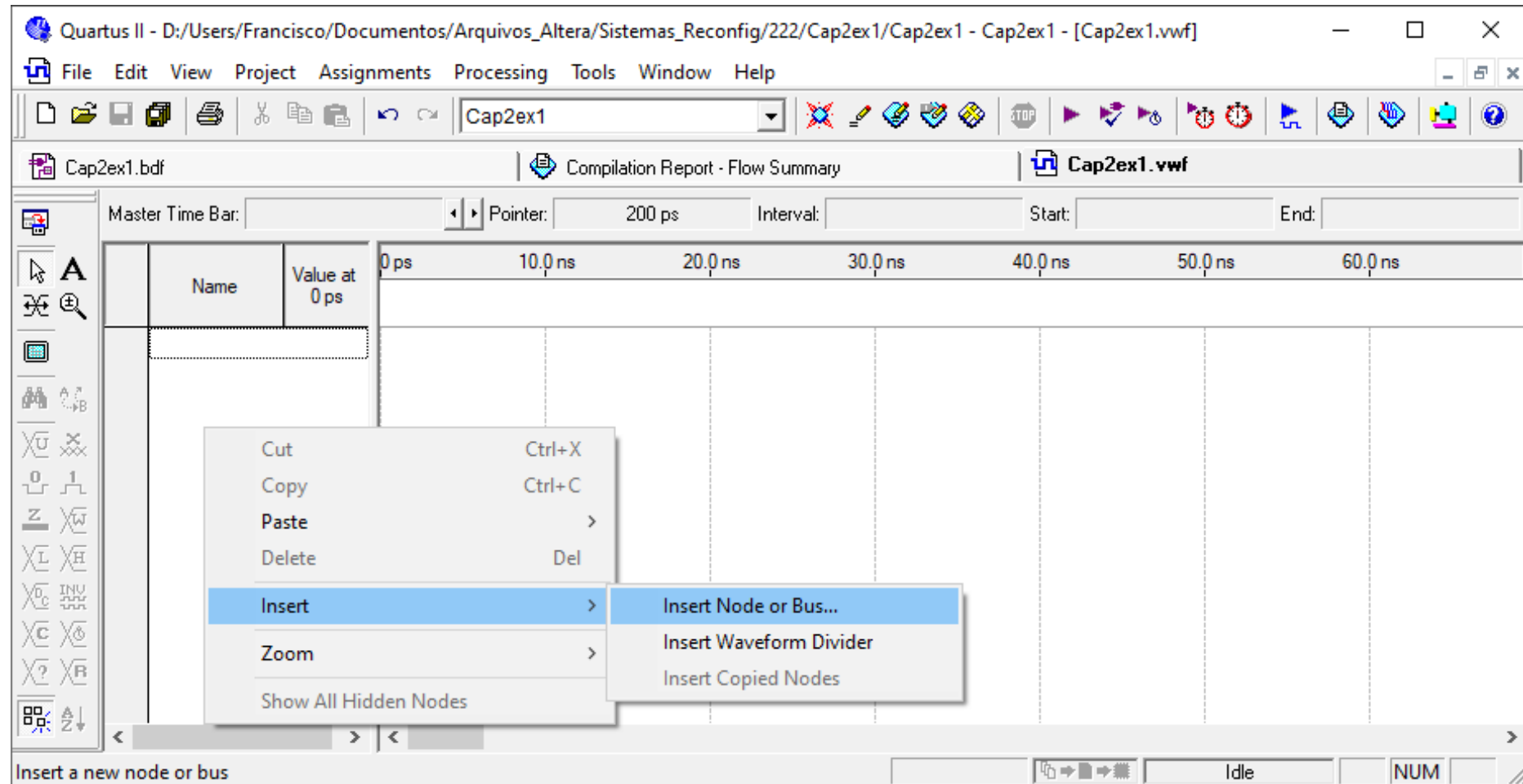
- Para fazer *zoom in/zoom out* na escala de tempo, usar o “Ctrl”+*mouse scroll* ou os atalhos “Ctrl+Space” e “Ctrl+Shift+Space”, respectivamente.



Exercício – simulação funcional

Para inserir os sinais que vão fazer parte da simulação:

- Com o botão direito do *mouse*, clicar na área vazia a esquerda, abaixo de “Name”;
- seleccionar “Insert” e, depois, “Insert Node or Bus”;
- na janela aberta, clicar no botão “Node Finder...”;




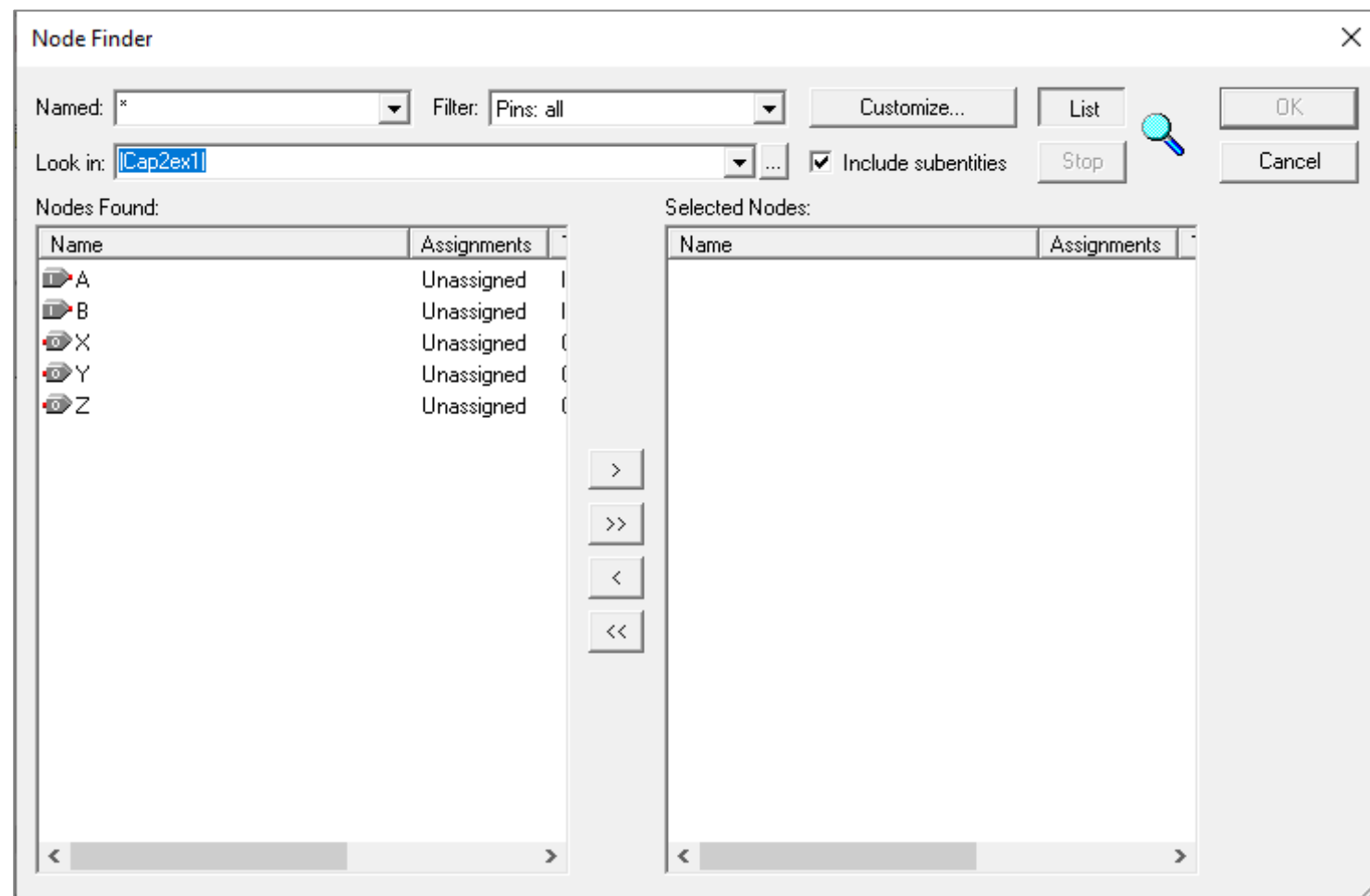
Exercício – simulação funcional

- na nova janela aberta, no campo “Filter”, selecionar a opção “Pins: all”;

- clicar no botão “List”;

A janela da esquerda será preenchida com todos os pinos onde é possível fazer a simulação.

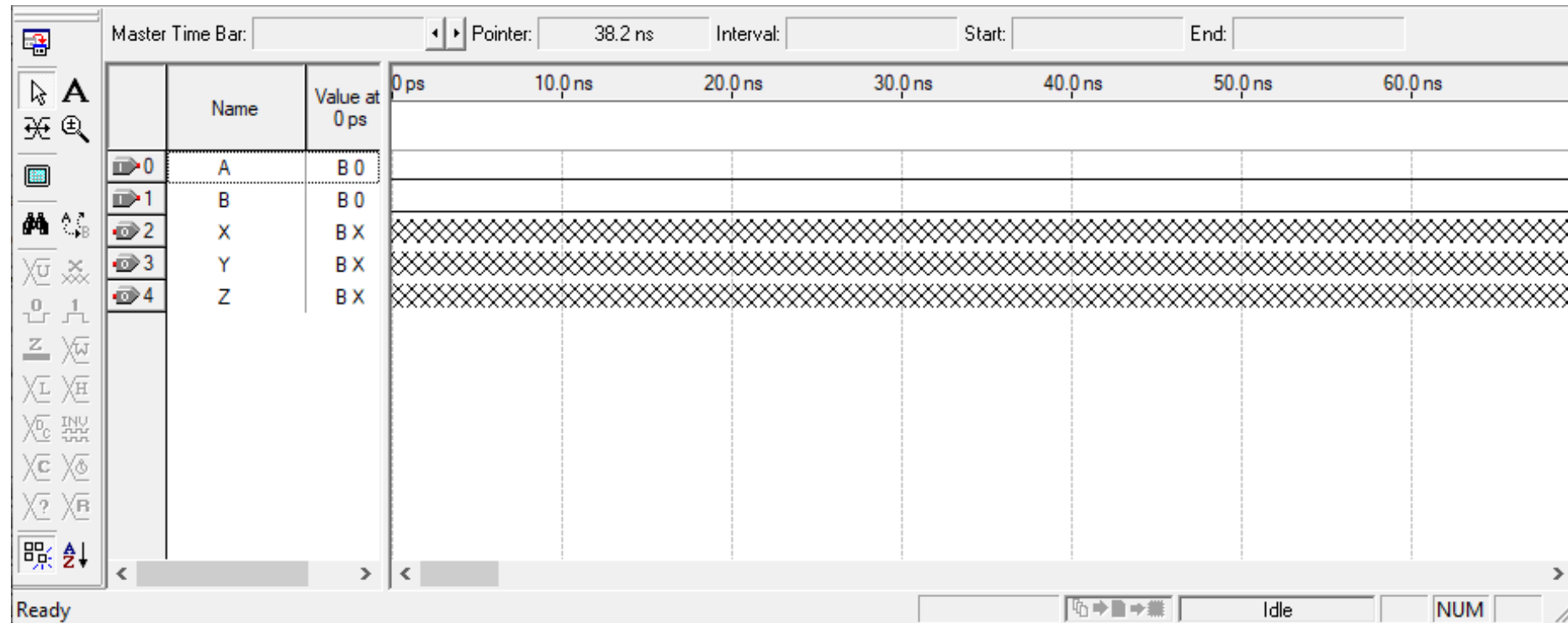
- Clicar no botão  para selecionar todos os sinais (transferir para a janela da direita);
- clicar em “OK” para terminar e, novamente, em “OK” para fechar a outra janela.



Exercício – simulação funcional

Ao lado, é mostrada a área de trabalho do simulador, já com os sinais inseridos.

Nas saídas, é mostrado XXXXXX, indicando nível lógico desconhecido, pois a simulação ainda não foi efetuada.

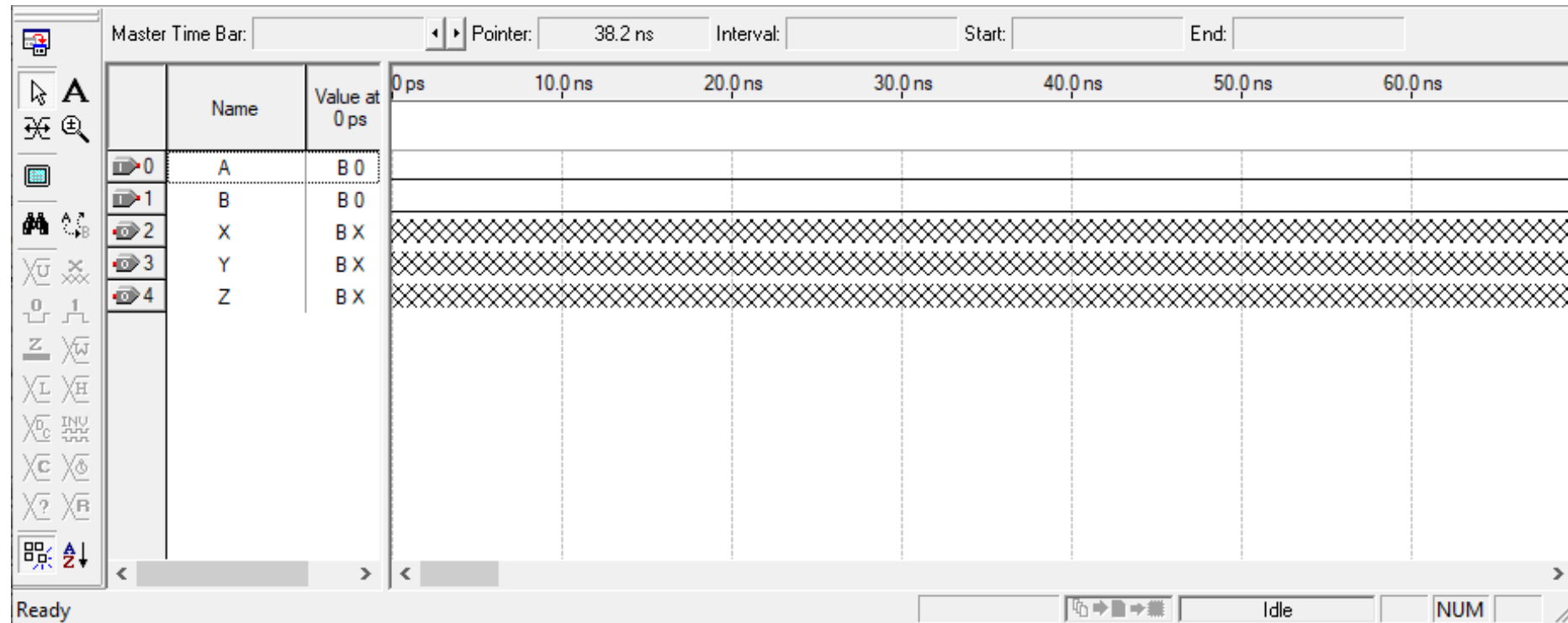


Exercício – simulação funcional

A simulação funcional não leva em consideração os atrasos de propagação internos no circuito integrado. Assim, todas as respostas são ideais (instantâneas).


Portanto, a base de tempo de simulação não interessa.

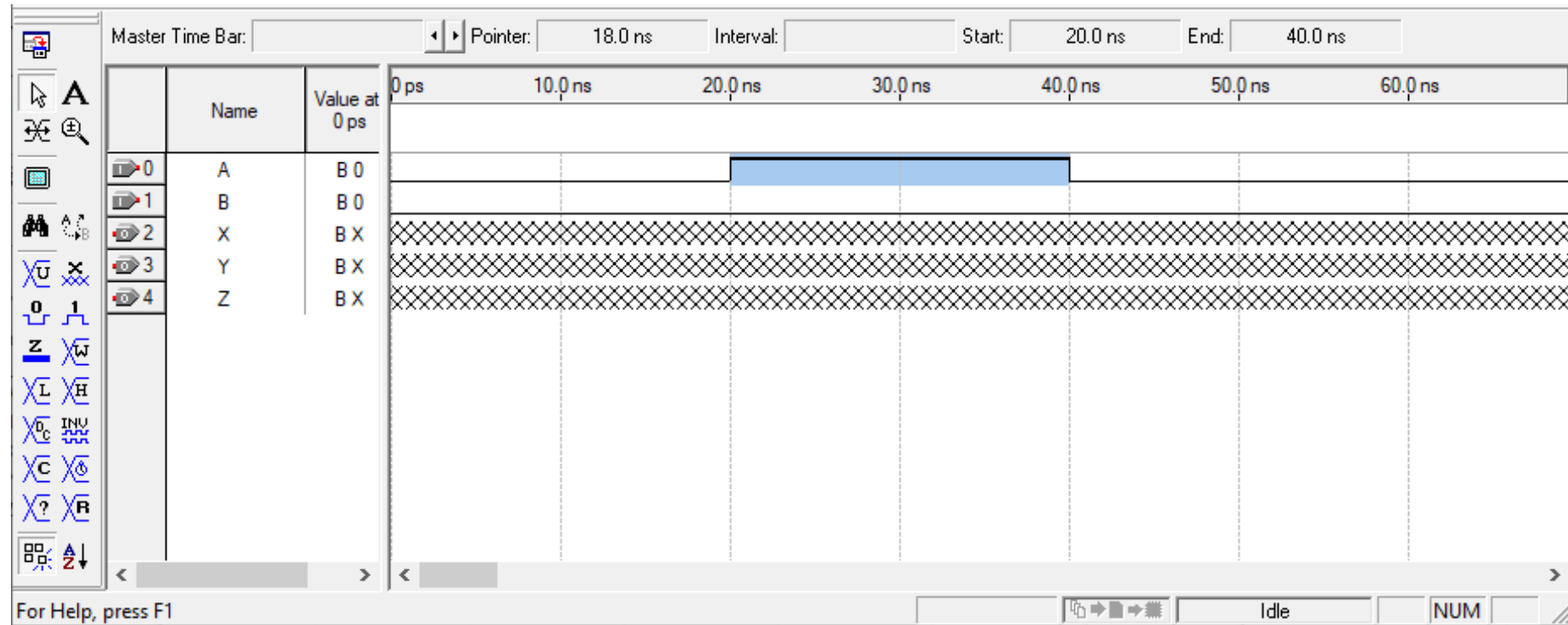
Usaremos uma grade de 10 ns, que é o *default* do simulador.



Exercício – simulação funcional


Vamos agora configurar os estímulos de entrada para a simulação. As entradas **A** e **B** devem ser configuradas para os valores **00**, **01**, **10** e **11**. Para isso:

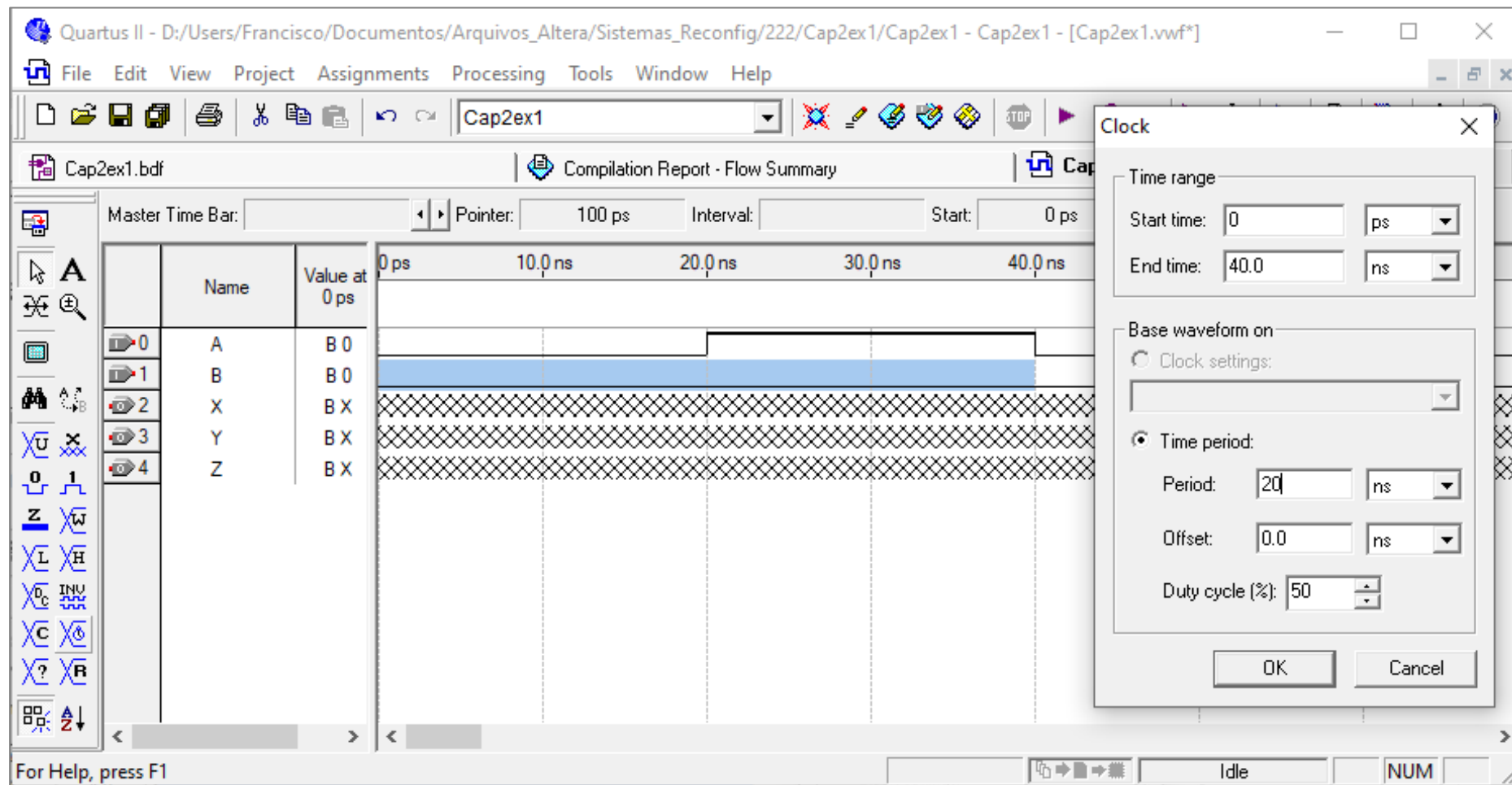
- Com o *mouse*, marcar a área da linha da entrada **A**, entre 20,0 e 40,0 ns;
- clicar no botão  da barra de ferramentas lateral



Exercício – simulação funcional

A entrada B será configurada usando outra ferramenta:

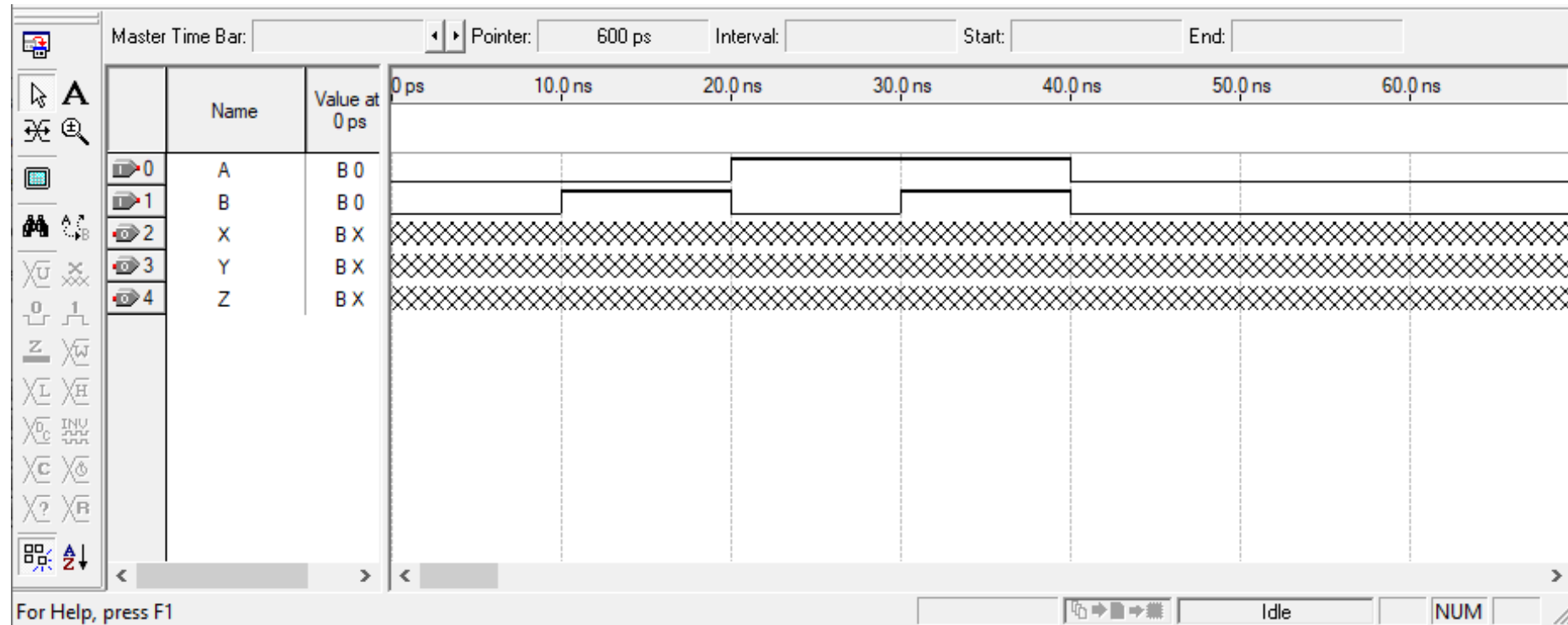
- Com o *mouse*, marcar a área da linha da entrada B, entre 0,0 e 40,0 ns;
- clicar no botão  (*clock*) da barra de ferramentas;
- preencher a janela aberta conforme mostrado ao lado;
- clicar em “OK” para fechar a janela



Exercício – simulação funcional

Agora os estímulos já foram configurados e o arquivo está pronto para a simulação.

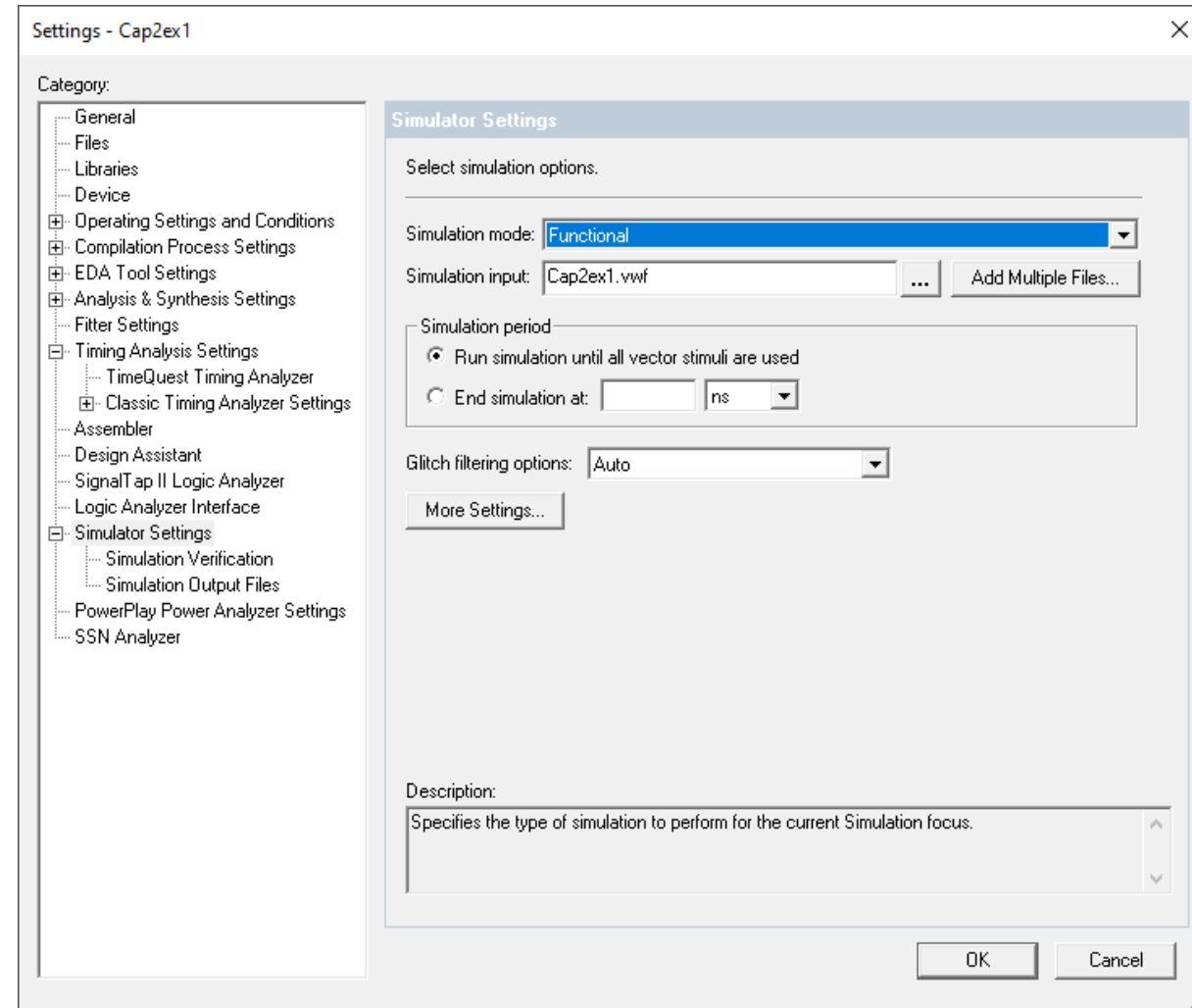
- Salvar o arquivo.



Exercício – simulação funcional

É necessário configurar o software para realizar a **simulação funcional** (o *default* é simulação temporal). Para isso:

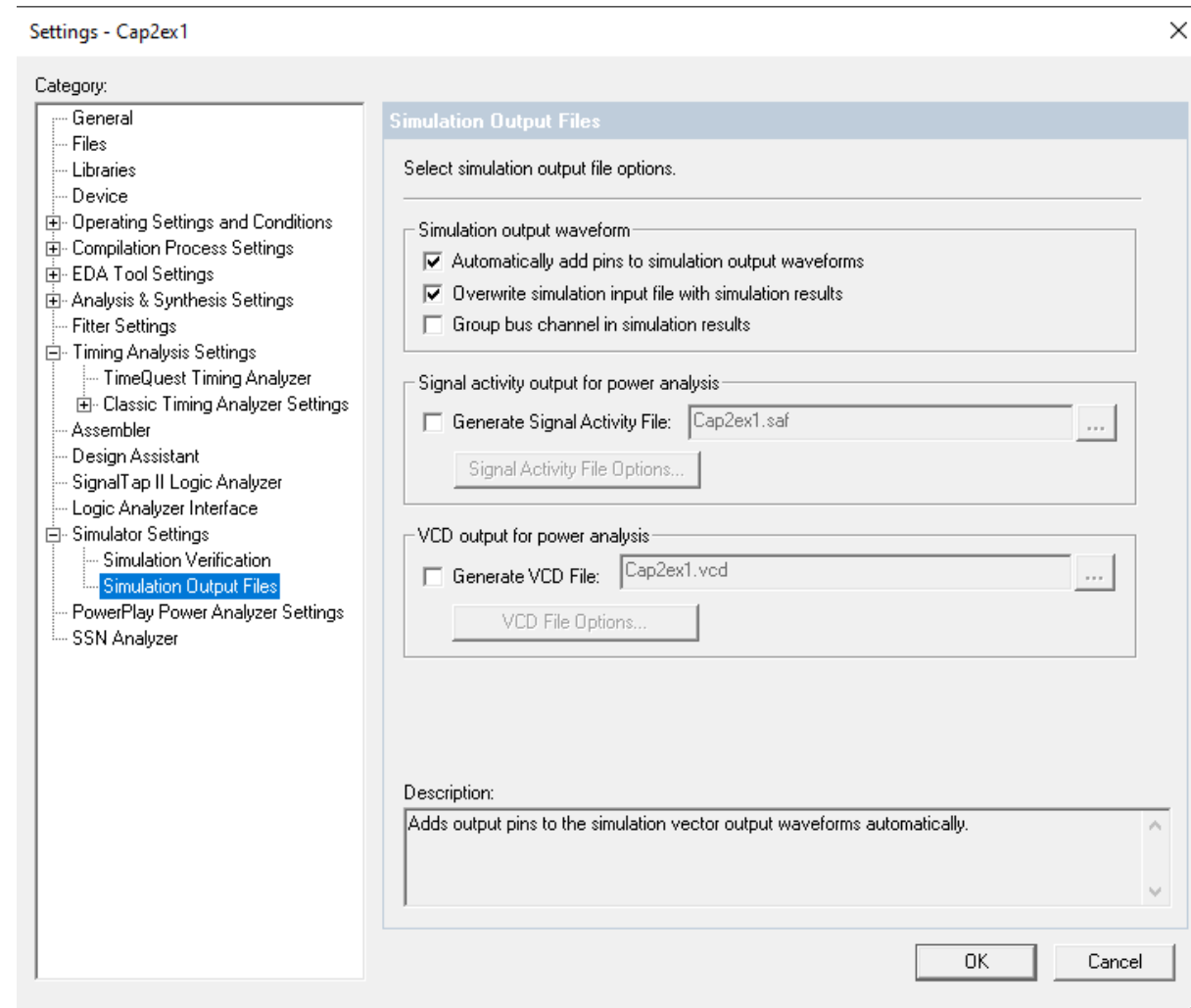
- No menu “**Assignments**”, escolher a opção “**Settings**”, ou use o atalho “**Ctrl+Shift+E**”;
- na janela aberta, escolher a opção “**Simulator Settings**” e, ao lado, no campo “**Simulation mode:**”, selecionar a opção “**Functional**”;



Exercício – simulação funcional

- Expandir a opção “**Simulator Settings**” e escolher a opção “**Simulation output file**”;
- No lado direito, ativar a opção “**Overwrite simulation input file with simulation results**”;
- clicar no botão “**OK**” para finalizar.

Essa última configuração é para que o resultado da simulação seja salvo no arquivo **.vwf**.




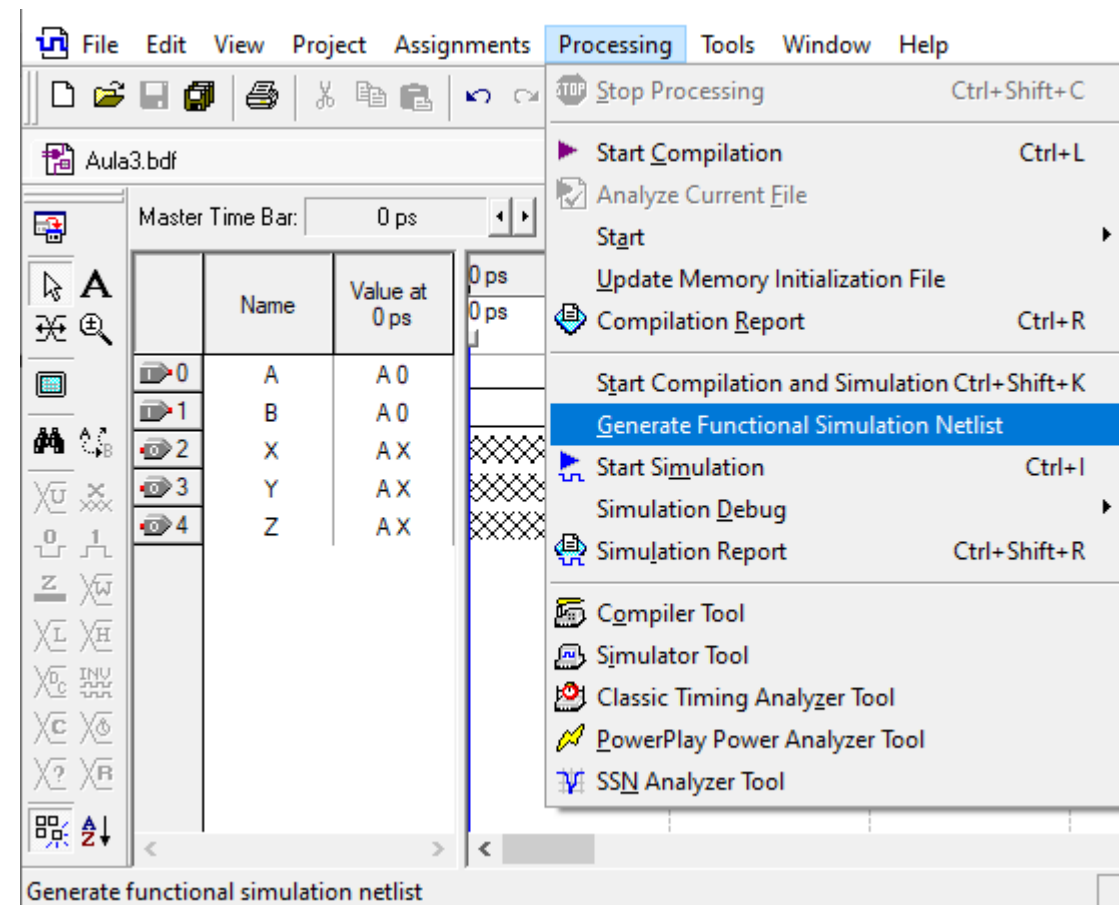
Exercício – simulação funcional

Antes de fazer a simulação, é necessário criar uma lista de ligações (*netlist*) para o circuito a ser simulado:

- No menu “**Processing**” escolher a opção “**Generate Functional Simulation Netlist**”;
- clicar em “**OK**” na janela de confirmação de comando executado.



Agora **está tudo pronto** para iniciarmos a simulação:

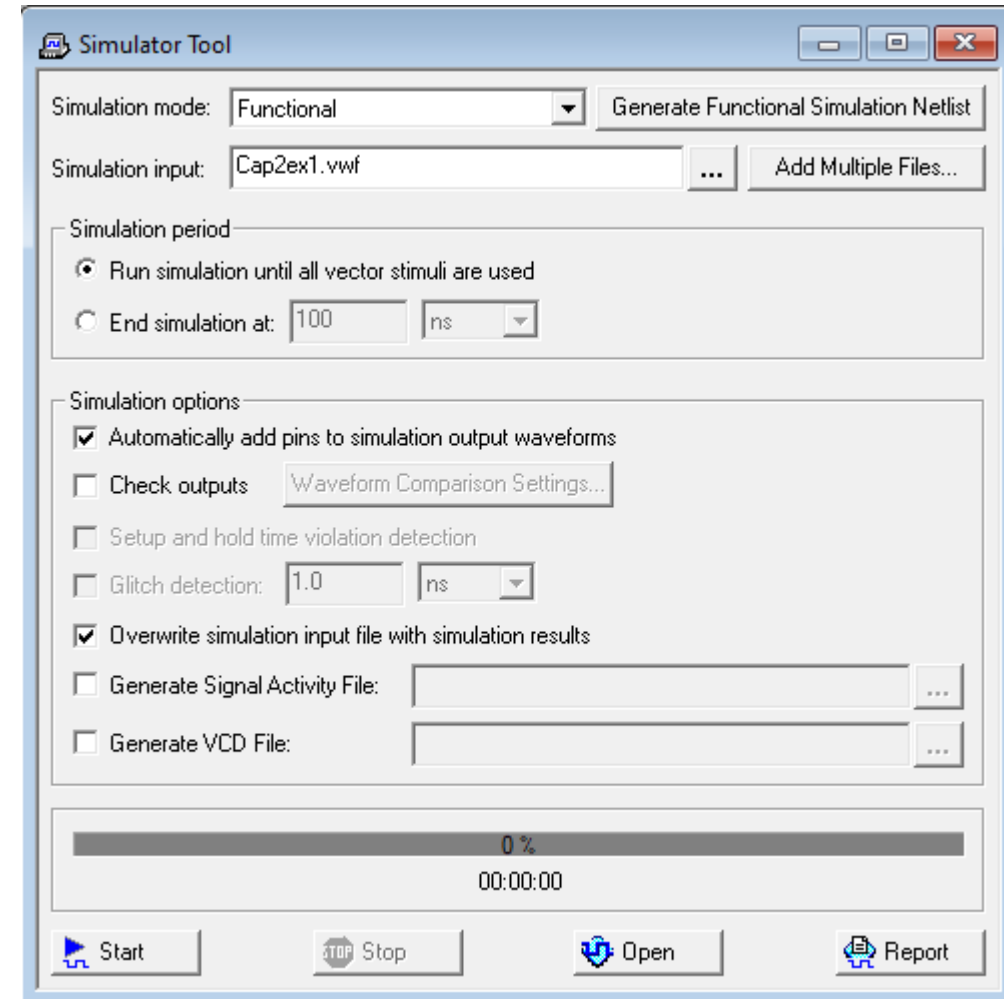
- Clicar no botão  da barra de comandos superior, ou usar o atalho “**Ctrl+I**” ou, no menu “**Processing**”, escolher a opção “**Start Simulation**”;
- após alguns instantes, a simulação está pronta.



Exercício – simulação funcional

Uma **outra maneira** de configurar e iniciar a simulação é:

- No menu “**Processing**”, escolher a opção “**Simulator Tool**”;
- na janela aberta, no campo “**Simulation mode**”, escolher a opção “**Functional**”
- clicar no botão “**Generate Functional Simulation Netlist**”;
- ativar a opção “**Overwrite simulation input file with simulation results**”;
- clicar no botão  **Start** para iniciar a simulação;
- se o arquivo de simulação (.vwf) não estiver aberto, clicar no botão  **Open**



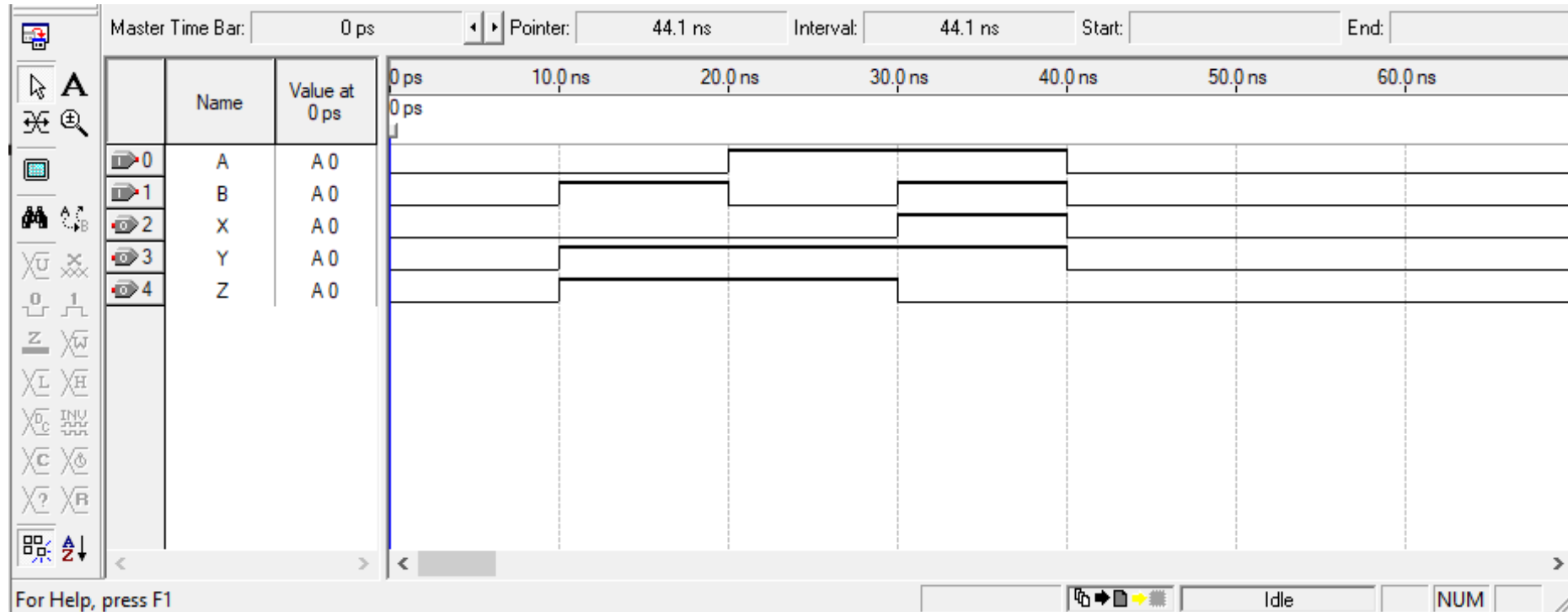
Exercício – simulação funcional

Para visualizar os resultados:

- Clicar na aba do arquivo de simulação
“Cap2ex1.vwf”


Os resultados obtidos correspondem aos resultados esperados?

Comparar com a tabela criada anteriormente


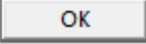


Exercício – teste no módulo

Para testar o projeto no módulo DE2, é necessário compilar o projeto, o que, ao final, irá gerar o arquivo de configuração do FPGA.

- Para compilar, clicar com o mouse no botão  da barra de comandos superior;
- após algum tempo, a compilação é completada.

Flow Status	Successful - Tue Aug 08 09:07:37 2023
Quartus II Version	9.1 Build 350 03/24/2010 SP 2 SJ Web Edition
Revision Name	Cap2ex1
Top-level Entity Name	Cap2ex1
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Met timing requirements	Yes
Total logic elements	3 / 33,216 (< 1 %)
Total combinational functions	3 / 33,216 (< 1 %)
Dedicated logic registers	0 / 33,216 (0 %)
Total registers	0
Total pins	5 / 475 (1 %)
Total virtual pins	0
Total memory bits	0 / 483,840 (0 %)
Embedded Multiplier 9-bit elements	0 / 70 (0 %)
Total PLLs	0 / 4 (0 %)

 Full Compilation was successful (4 warnings)


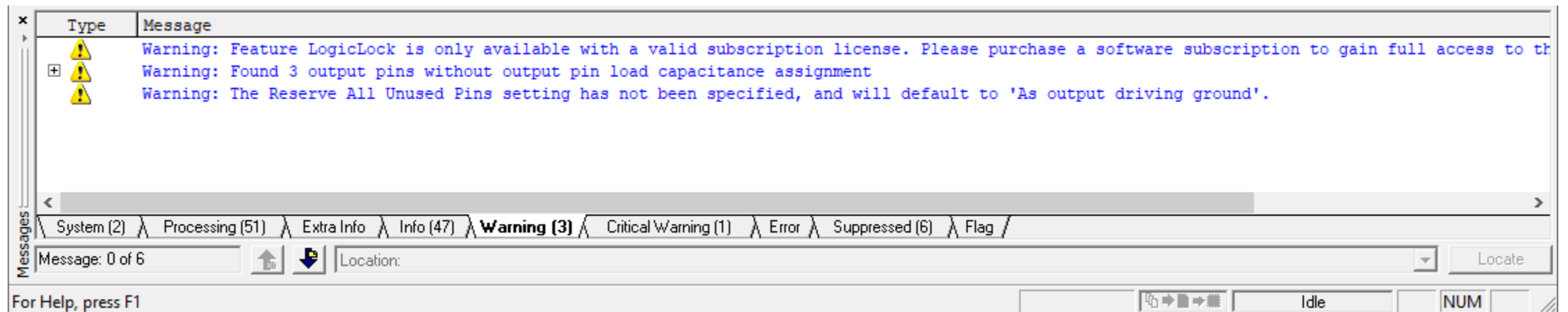
3 warning, 1 critical warning

Obs.:

- Outras formas de iniciar o processo de compilação são: pelo atalho “Ctrl+L” ou no menu “Processing” ativar a opção “Start_Compilation”.

Exercício – teste no módulo

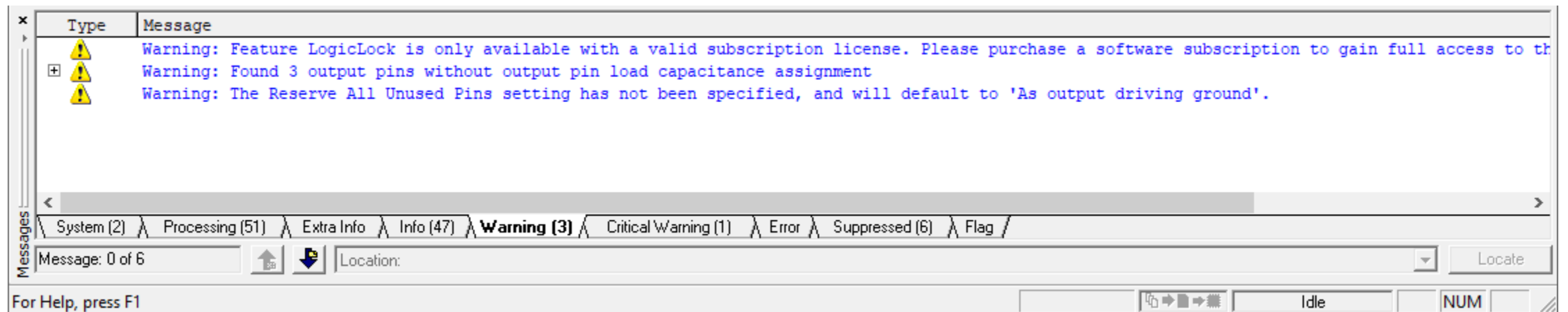
Reparar que não houve erros, mas ocorreram 3 *warnings*:



- O primeiro *warning* “**Feature Logiclock**” é apenas um lembrete de que uma determinada função não está disponível nessa versão gratuita do *software*;
- O segundo *warning* “**Found 3 outputs pins**” alerta para o fato de não terem sido especificadas as capacitâncias equivalentes dos circuitos ligados aos pinos de saída. Essas capacitâncias são usadas na simulação de tempo de atraso de saída e, para esse exercício, não são importantes.

Exercício – teste no módulo

Reparar que não houve erros, mas ocorreram 3 *warnings*:

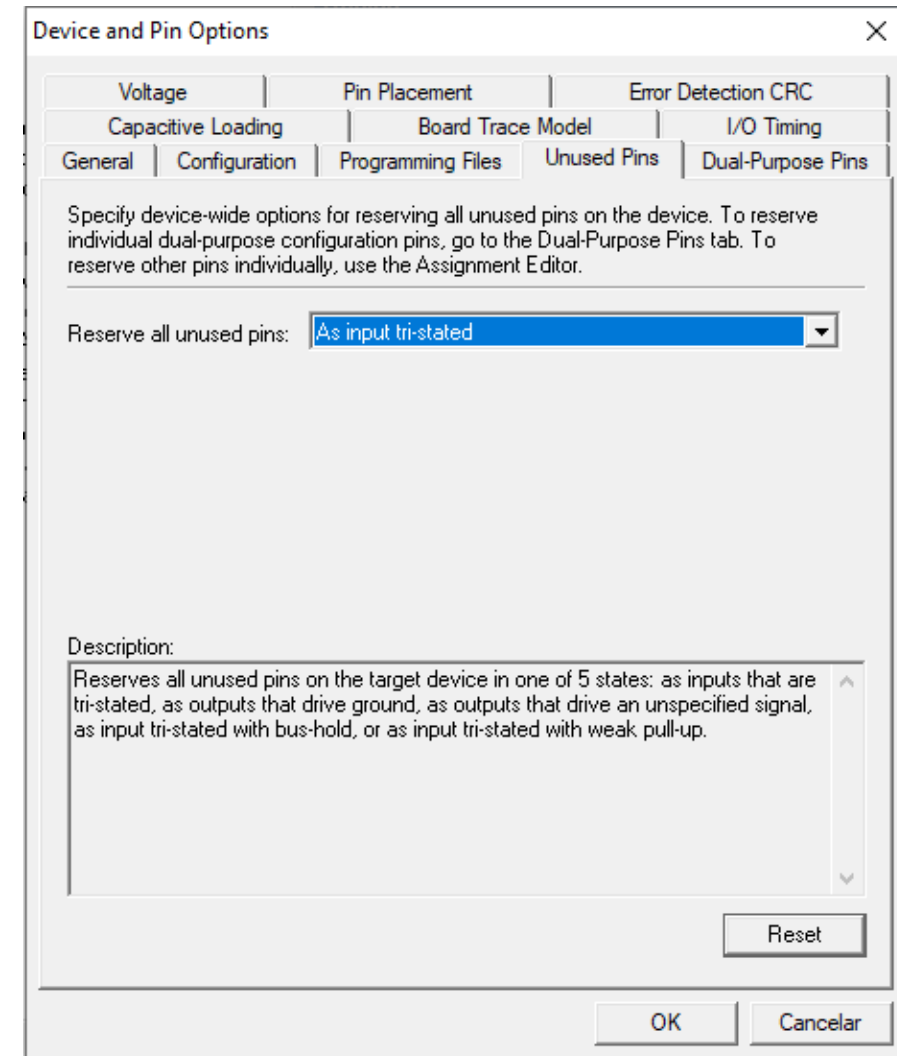


- O terceiro *warning* “*The Reserve All ...*” aponta que não foi feita opção alguma para a especificação do que o compilador deve fazer a respeito dos pinos não usados e que, por padrão, foi ajustada para que todos esses pinos se comportem como saída ligadas ao terra (GND). Para simulação, esse *warning* não é relevante. No entanto, para testar o exercício no módulo DE2, **isso tem de ser mudado**. Lembrar que o módulo possui vários circuitos ligados ao FPGA, que podem ser danificados se ligados ao terra.

Exercício – teste no módulo

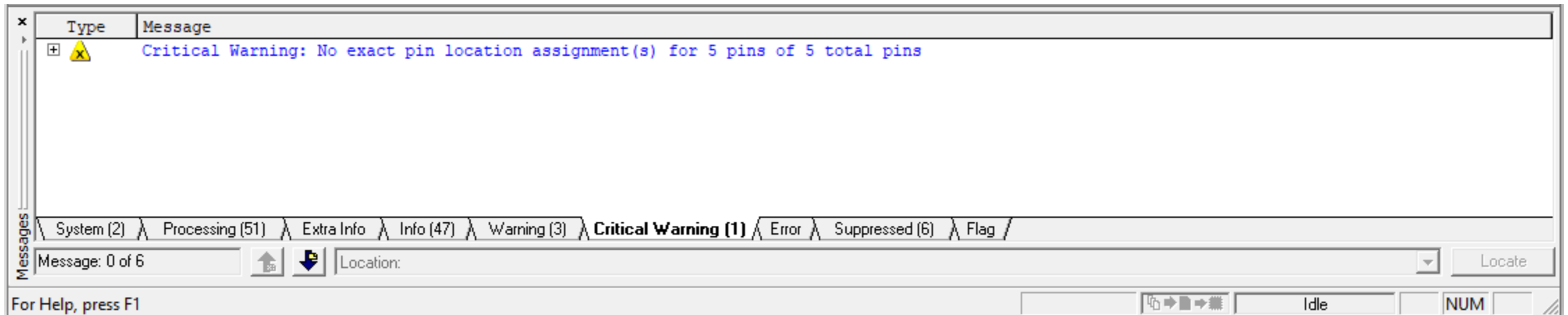
- No *menu* “Assignments” escolher a opção “Device...”;
- na janela aberta, clicar no botão “Device and Pins Options...”, que está no lado direito;
- na nova janela, selecionar a aba “Unused Pins” e em “Reserve all unused pins:” escolher a opção “As input tri-stated”;
- Fechar as janelas com “OK” e depois “OK” novamente.

Dessa forma, os pinos não usados se comportarão como entrada, não gerando conflito com os outros circuitos do módulo.



Exercício – teste no módulo

Ocorreu também 1 *critical warning*:



- O *critical warning* “*No exact pin location ...*” aponta para o fato de que não foi especificado em quais pinos do circuito integrado devem ser ligados os sinais de entrada (A e B) e saída (X, Y, e Z) do exercício. Esses pinos devem ser escolhidos de acordo com o projeto do módulo DE2. Para testar o exercício, serão usadas chaves (entradas) e LEDs (saídas). As informações sobre os pinos correspondentes aos recursos existentes no módulo podem ser encontradas no manual do usuário.

Exercício – teste no módulo

Ao lado, um recorte do manual do usuário do módulo DE2

- Para as entradas, serão usadas as chaves 0 e 1:

Sinal	Chave	Pino
A	SW[0]	PIN_N25
B	SW[1]	PIN_N26

- Para as saídas, serão usados os LEDs vermelhos 0, 1 e 2:

Sinal	LED	Pino
X	LEDR[0]	PIN_AE23
Y	LEDR[1]	PIN_AF23
Z	LEDR[2]	PIN_AB21

Obs.:

- Tabela com os pinos dos LEDs no *slide* seguinte

Exercício – teste no módulo

Ao lado, um recorte do manual do usuário do módulo DE2

- Para as entradas, serão usadas as chaves 0 e 1:

Sinal	Chave	Pino
A	SW[0]	PIN_N25
B	SW[1]	PIN_N26

- Para as saídas, serão usados os LEDs vermelhos 0, 1 e 2:

Sinal	LED	Pino
X	LEDR[0]	PIN_AE23
Y	LEDR[1]	PIN_AF23
Z	LEDR[2]	PIN_AB21

Signal Name	FPGA Pin No.	Description
SW[0]	PIN_N25	Toggle Switch[0]
SW[1]	PIN_N26	Toggle Switch[1]
SW[2]	PIN_P25	Toggle Switch[2]
SW[3]	PIN_AE14	Toggle Switch[3]
SW[4]	PIN_AF14	Toggle Switch[4]

Table 4.1. Pin assignments for the toggle switches.

Signal Name	FPGA Pin No.	Description
LEDR[0]	PIN_AE23	LED Red[0]
LEDR[1]	PIN_AF23	LED Red[1]
LEDR[2]	PIN_AB21	LED Red[2]
LEDR[3]	PIN_AC22	LED Red[3]
LEDR[4]	PIN_AD22	LED Red[4]

Table 4.3. Pin assignments for the LEDs.

Exercício – teste no módulo

Para especificar os pinos:

- No *menu* “Assignments” escolher a opção “Pins”;
- na janela aberta, clicar na região correspondente à linha do “Node Name” A e coluna “Location”, e digitar ou escolher o pino correspondente;
- repetir para os outros os sinais;
- após finalizar a configuração de todos os pinos, fechar a janela (não há necessidade de confirmação).

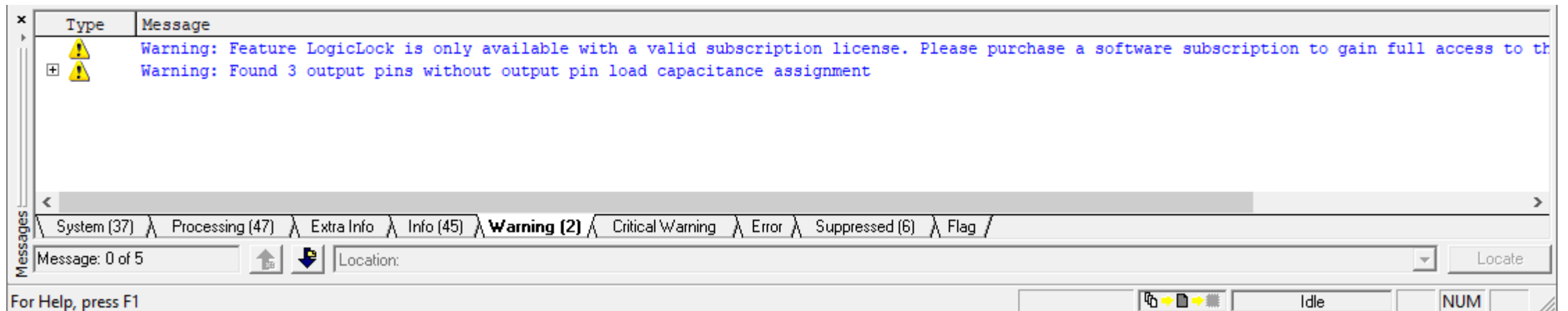
	Node Name	Direction	Location	I/O Bank	VREF Group	I/O Standard	Reserved
1	A	Input	PIN_AE21	7	B7_N0	3.3-V LVTTL (default)	
2	B	Input				3.3-V LVTTL (default)	
3	X	Output				3.3-V LVTTL (default)	
4	Y	Output				3.3-V LVTTL (default)	
5	Z	Output				3.3-V LVTTL (default)	
6	<<new node>>						

Exercício – teste no módulo

Compilar o projeto novamente.


Ainda permanecem 2 *warnings*, porém, como analisado anteriormente, eles não prejudicam o projeto.

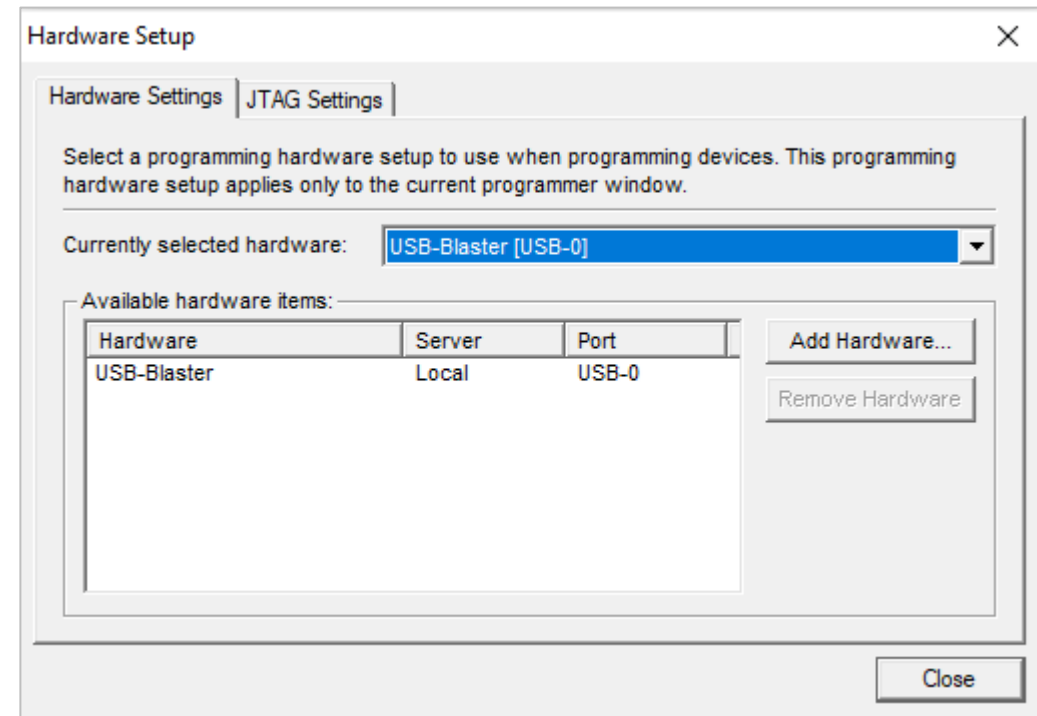
Notar que não há mais *critical warnings*



Exercício – teste no módulo

Programar o dispositivo:

- Clicar no botão  da barra de comandos superior ou, no menu “Tools”, selecionar opção “Programmer”;
- na janela aberta, clicar em “Hardware Setup”;
- selecionar opção “USB-Blaster”;
- clicar em “Close”;



Obs.:

- O módulo DE2 deverá estar ligado ao computador através da interface USB



Fim

Até a próxima aula!