

Universidade Tecnológica Federal do Paraná – Toledo
Engenharia da Computação – COENC

Lógica Reconfigurável

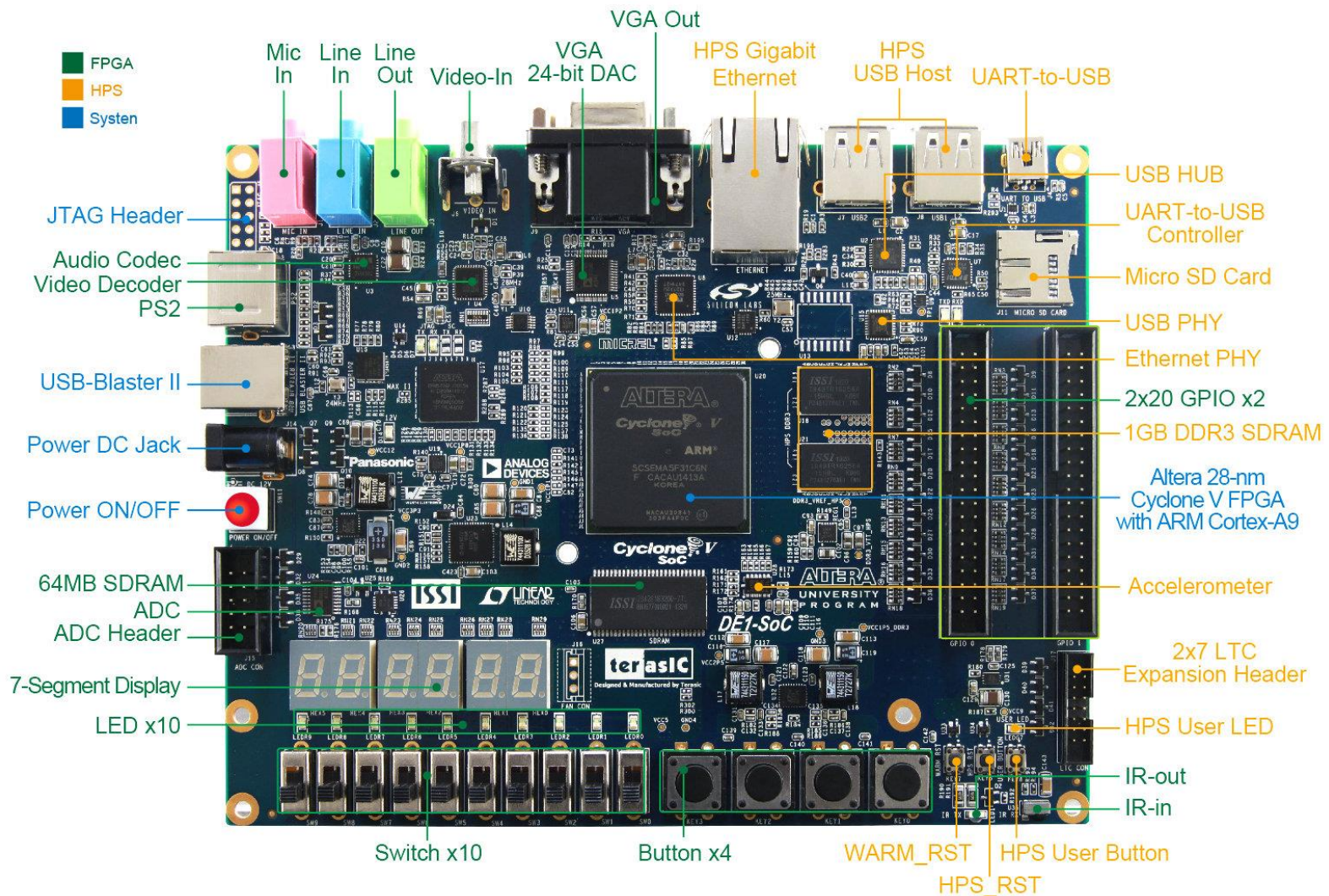
Implementação de circuitos com diagrama de blocos Utilizando o kit de desenvolvimento DE1-SoC

Tiago Piovesan Vendruscolo



Esta licença permite que outros remixem, adaptem e criem a partir do trabalho para fins não comerciais, desde que atribuam o devido crédito aos autores originais. [4.0 international](https://creativecommons.org/licenses/by-nc-nd/4.0/)

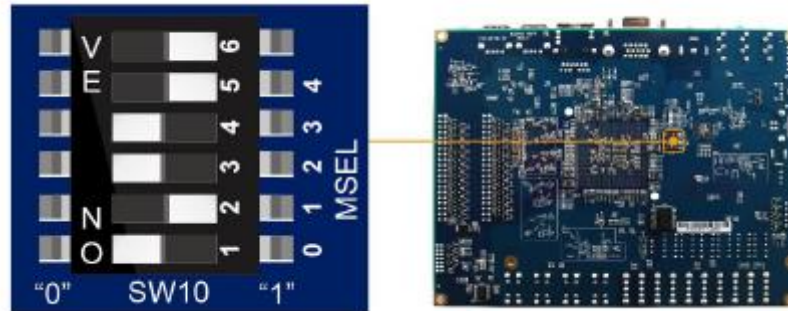
DE1-SoC - ALTERA



FPGA utilizada nas aulas

- FPGA Cyclone V SoC 5CSEMA5F31C6 – 85k Elementos lógicos.
- Dual-core ARM Cortex-A9.
- 6 PLLs.
- 11 Leds, 4 pushbuttons, 10 dip switches e 6 displays 7 segmentos.
- 1 Acelerômetro de três eixos com resolução de 13 bits.
- Conversor A/D de 8 canais e 12-bits com taxa de conversão de até 500 Ksps. Tensão analógica de entrada: 0 ~ 4,096 V.
- 2 conectores com 40 pinos I/O em 3.3 V.
- Memória SDRAM de 64MB – FPGA
- Memória DDR3 de 1 GB – ARM.
- Saída VGA com DAC 24-bit.
- Áudio com CODEC 24-bit (linha de entrada, saída e microfone).
- Comunicação: UART, Ethernet, IR.
- Sistema de clock on-board com um oscilador dedicado de 50 MHz;

- Quartus Prime Lite 18
- FPGA: 5CSEMA5F31C6
- Confira se as chaves de programação estão corretas



- Criem um novo projeto > 5CSEMA5F31C6

Family, Device & Board Settings

Device

Board

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.

Device family

Family: Cyclone V (E/GX/GT/SX/SE/ST)

Device: All

Target device

☐ Auto device selected by the Fitter
☒ Specific device selected in 'Available devices' list
☐ Other: n/a

Show in 'Available devices' list

Package: Any

Pin count: Any

Core speed grade: Any

Name filter:

☒ Show advanced devices

Available devices:

Name	Core Voltage	ALMs	Total I/Os	GPIOs	GXB Channel PMA	GXB Channel P
5CSEMA4U23C8	1.1V	15880	314	314	0	0
5CSEMA4U23I7	1.1V	15880	314	314	0	0
5CSEMA5F31A7	1.1V	32070	457	457	0	0
5CSEMA5F31C6	1.1V	32070	457	457	0	0
5CSEMA5F31C7	1.1V	32070	457	457	0	0
5CSEMA5F31C8	1.1V	32070	457	457	0	0
5CSEMA5F31I7	1.1V	32070	457	457	0	0

<

>

< Back

Next >

Finish

Cancel

Help

Quando for utilizar o Kit DE1-SOC









Family, Device & Board Settings

Device **Board**

Select the board/development kit you want to target for compilation.

Family: Cyclone V Development Kit: Any

Available boards:

	Name	Version	Family	Device	Vendor
	Atlas-SoC (DE0-Nano-SoC)	1.0	Cyclone V	5CSEMA4U23C6	Terasic
	Cyclone V E FPGA Development Kit	1.0	Cyclone V	5CEFA7F31I7	Altera
	Cyclone V GT FPGA Development Kit	1.0	Cyclone V	5CGTFD9E5F35C7	Altera
	Cyclone V SoCKit	1.0	Cyclone V	5CSXFC6D6F31C6	Arrow
	Cyclone V SoC Development Kit	1.0	Cyclone V	5CSXFC6D6F31C6	Altera
	Cyclone V GX Starter Kit	1.0	Cyclone V	5CGXFC5C6F27C7	Terasic
	DE0-CV Development Board	1.0	Cyclone V	5CEBA4F23C7	Terasic
	DE1-SoC Board	1.0	Cyclone V	5CSEMA5F31C6	Altera

☒ Create top-level design file.

Can't find your board? Check the [Design Store](#) for additions and search for baseline under Design Examples.

< Back **Next >** Finish Cancel Help

- Importando assignments pins.
 - *Assignments > import assignments.*
 - Importem o arquivo pin_assignments.qsf
 - *Analise os pinos atribuídos no pin planner.*
- Utilize para os PORTs o mesmo nome usado no manual: “DE1-SoC User manual”, dessa forma, não será necessário fazer a atribuição dos pinos manualmente.

File Edit View Processing Tools Window Help

Report

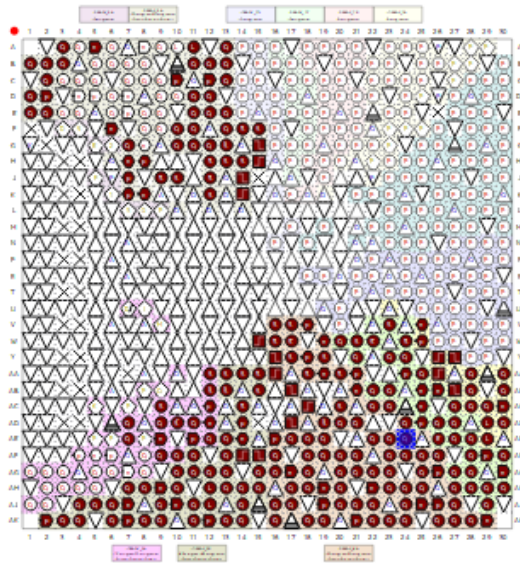
Report not available

Groups Report

Tasks

- Early Pin Planning
 - Early Pin Planning...
 - Run I/O Assignment Analysis...
 - Export Pin Assignments...

Top View - Wire Bond
Cyclone V - 5CSEMA5F31C6



Node Name	Direction	Location	I/O Bank	VREF Group	I/O Standard	Reserved	Current Strength	S
GPIO_1[5]	Unknown	PIN_AE23	4A	B4A_NO	3.3-V LVTTTL		16mA (default)	
GPIO_1[6]	Unknown	PIN_AE24	4A	B4A_NO	3.3-V LVTTTL		16mA (default)	
GPIO_1[7]	Unknown	PIN_AF25	4A	B4A_NO	3.3-V LVTTTL		16mA (default)	
GPIO_1[8]	Unknown	PIN_AF26	4A	B4A_NO	3.3-V LVTTTL		16mA (default)	
GPIO_1[9]	Unknown	PIN_AG25	4A	B4A_NO	3.3-V LVTTTL		16mA (default)	
HEX0[0]	Unknown	PIN_AE26	5A	B5A_NO	3.3-V LVTTTL		16mA (default)	
HEX0[1]	Unknown	PIN_AE27	5A	B5A_NO	3.3-V LVTTTL		16mA (default)	
HEX0[2]	Unknown	PIN_AE28	5A	B5A_NO	3.3-V LVTTTL		16mA (default)	
HEX0[3]	Unknown	PIN_AG27	5A	B5A_NO	3.3-V LVTTTL		16mA (default)	
HEX0[4]	Unknown	PIN_AF28	5A	B5A_NO	3.3-V LVTTTL		16mA (default)	

- Criem um novo .vhd

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY Primeiro_projeto IS
    PORT (IN1 : IN std_logic;
          IN2 : IN std_logic;
          SAIDA : OUT
std_logic);
END Primeiro_projeto;

ARCHITECTURE logica OF Primeiro_projeto
IS
BEGIN
    SAIDA<=IN1 AND IN2;
END logica;
```

- Renomeiem os PORTs de acordo com os nomes utilizados no arquivo pin_assignments.qsf.

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY Primeiro_projeto IS
    PORT (SW : IN std_logic_vector(1 downto 0);
          LEDR : OUT std_logic_vector(0 downto 0));
END Primeiro_projeto;

ARCHITECTURE logica OF Primeiro_projeto IS
BEGIN
    LEDR(0) <= SW(1) AND SW(0);
END logica;
```

Utilizando vetores de 1 bit:

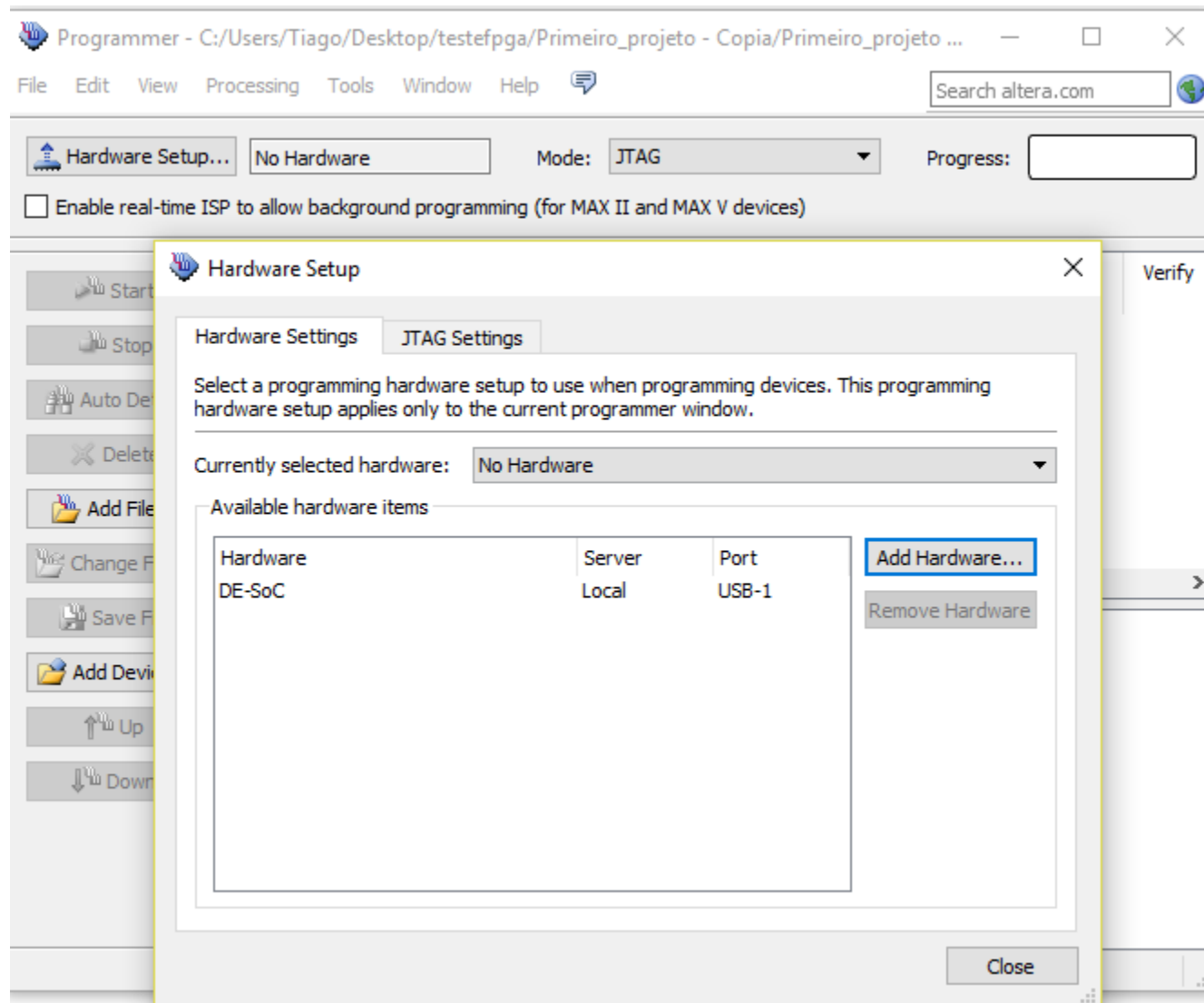
```
ENTITY ex IS
    PORT ( clk1: OUT STD_LOGIC_vector(0 to 0));
END ex;

clk1 <= "1";
clk1(0) <= '1';
```

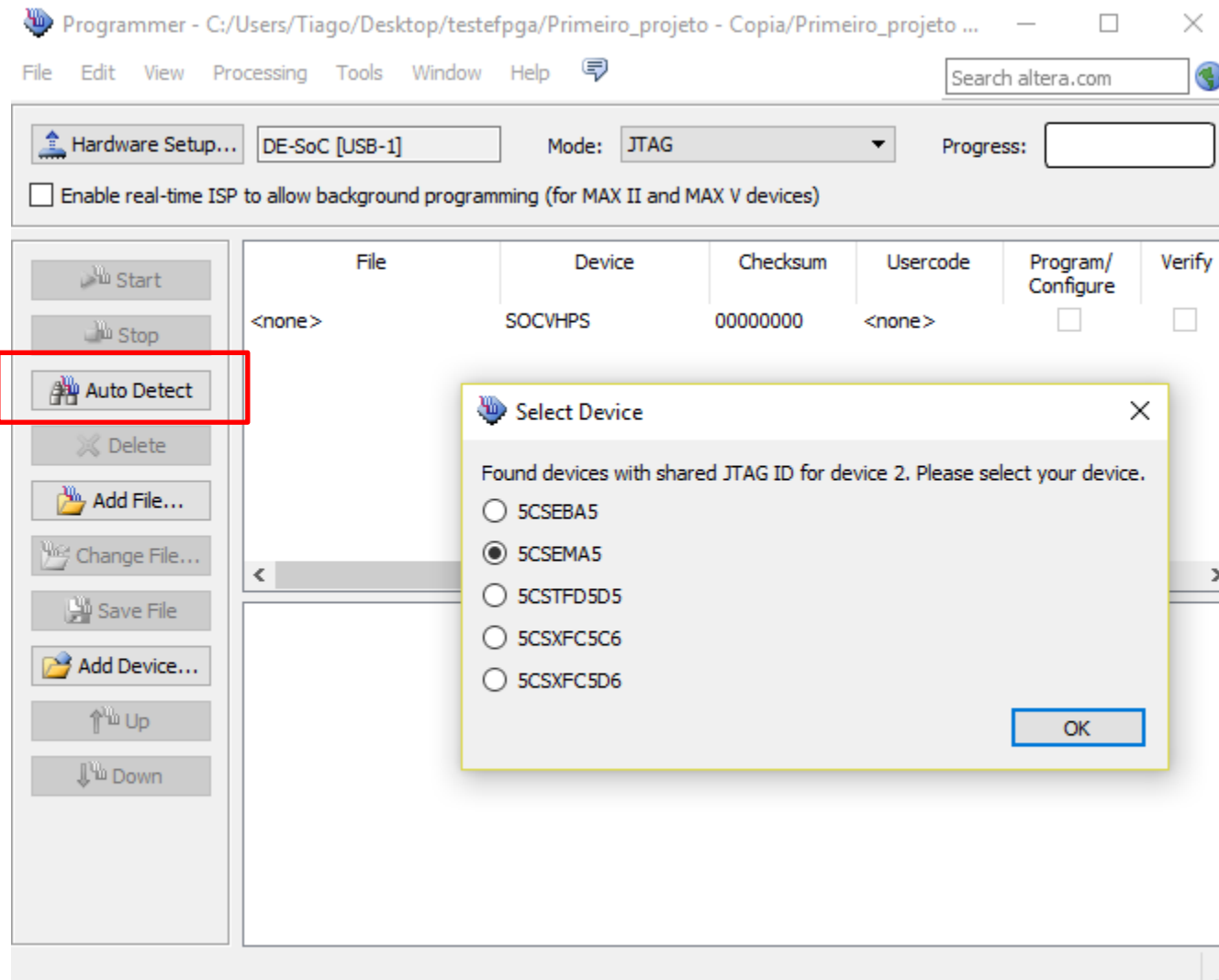
- Caso você queira utilizar apenas a saída GPIO_1[5] (slide 8), você deve fazer:

```
PORT (GPIO_1: OUT STD_LOGIC_VECTOR (5 to 5));
```
- Problema:
 - Ao fazer isso, será criado automaticamente um vetor de 6 bits (0 to 5), no entanto, apenas a posição 5 ficará disponível. Isso desperdiçará recursos.

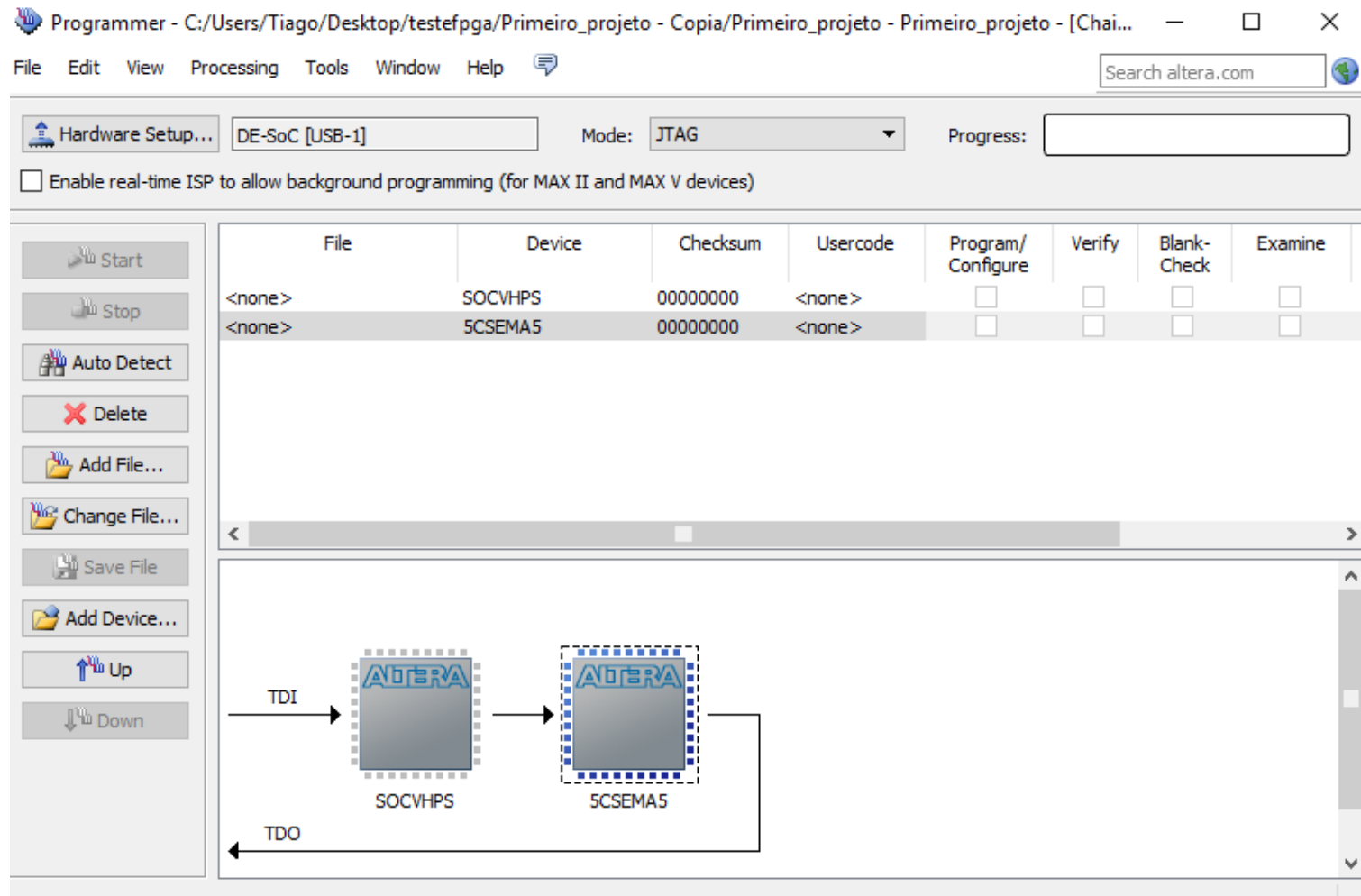
- Gravando na FPGA



Clique em auto detect e escolha 5CSEMA5



Nesse caso aparecerá 2 chips, pois essa placa possui uma FPGA e também um microcontrolador ARM. Selecione a FPGA (5CSEMA5).



Programmer - C:/Users/Tiago/Desktop/testefpga/Primeiro_projeto - Copia/Primeiro_projeto - Primeiro_projeto - [Chai... — □ ×

File Edit View Processing Tools Window Help

Hardware Setup... DE-SoC [USB-1] Mode: JTAG Progress:

☐ Enable real-time ISP to allow background programming (for MAX II and MAX V devices)

File	Device	Checksum	Usercode	Program/Configure	Verify	Blank-Check	Examine
<none>	SOCVHPS	00000000	<none>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<none>	5CSEMA5	00000000	<none>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Start Stop Auto Detect Delete Add File... Change File... Save File Add Device... Up Down

Replaces a selected programming file

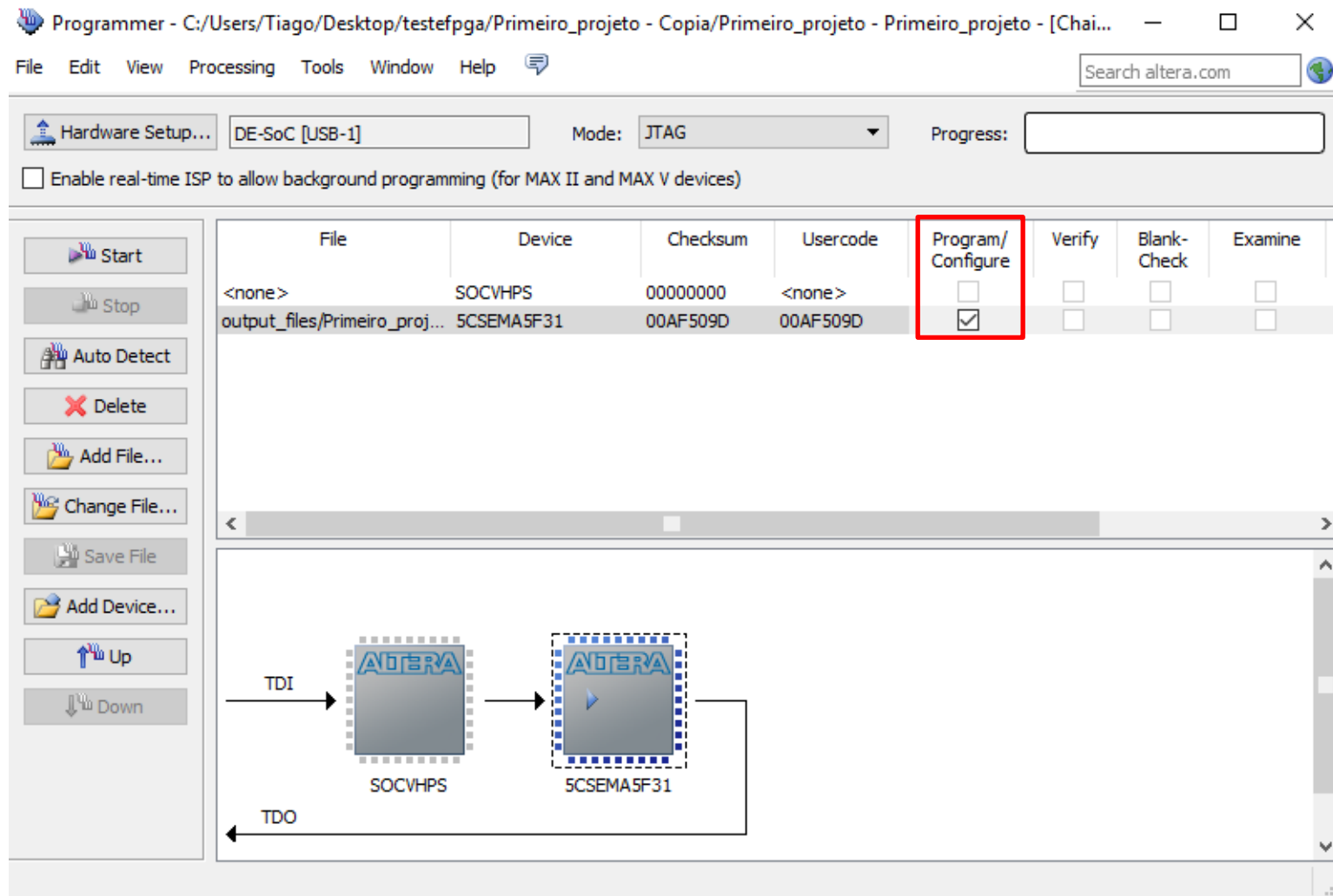
Diagram showing TDI input to SOCVHPS and TDO output from SOCVHPS.

Context menu options:

- Delete (Del)
- Select All (Ctrl+A)
- Add File...
- Change File
- Save File
- Add IPS File...
- Change IPS File...
- Delete IPS File
- Add PR Programming File...
- Change PR Programming File...

Pasta Output_files

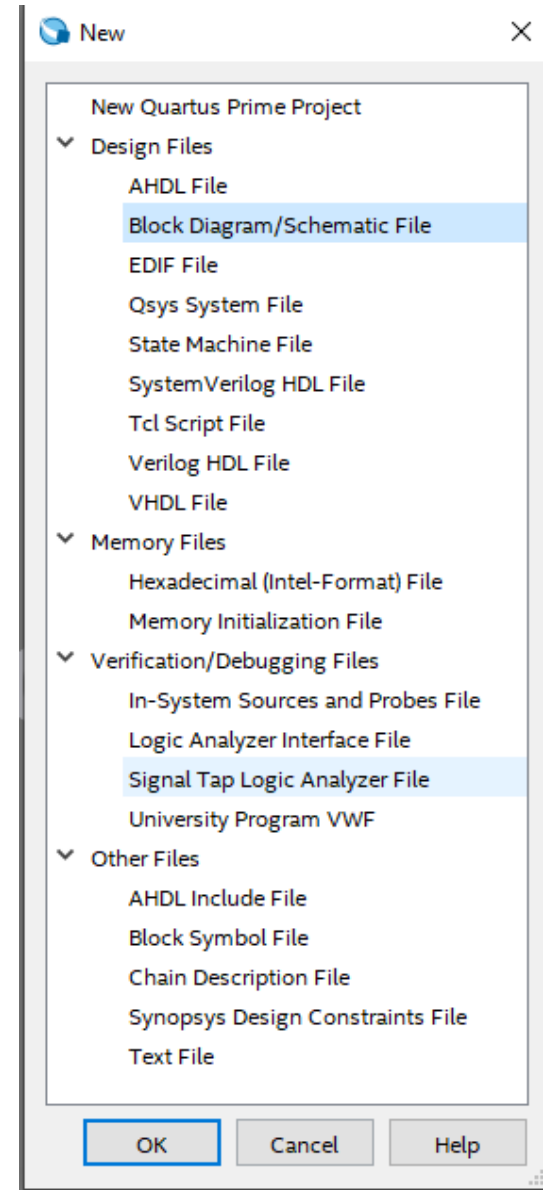
Marque a caixa “Program/Configure” e clique em “Start”, as vezes aparece um erro na barra de “progress”, apenas clique “Start” novamente.



Implementando circuitos com diagrama de blocos

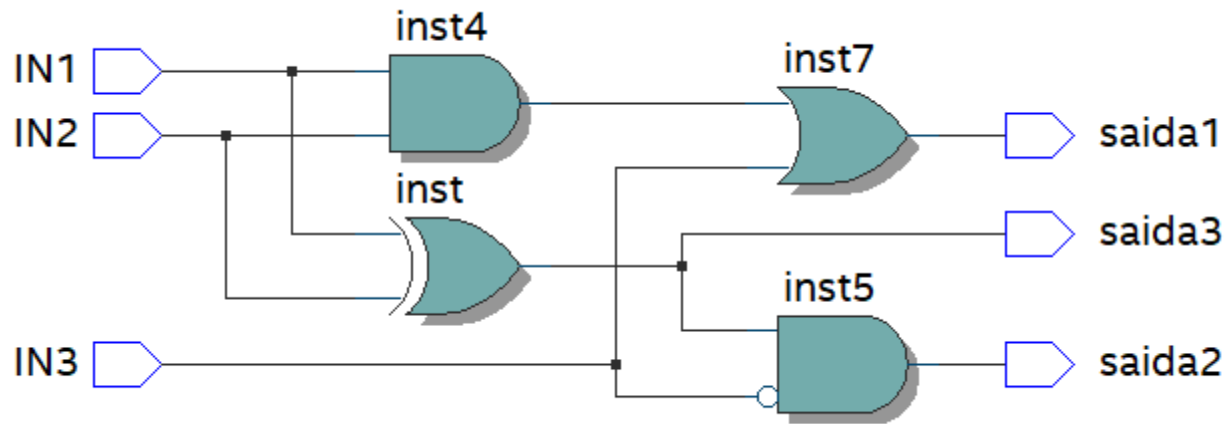
Crie um novo projeto.

New > Block Diagram/Schematic File



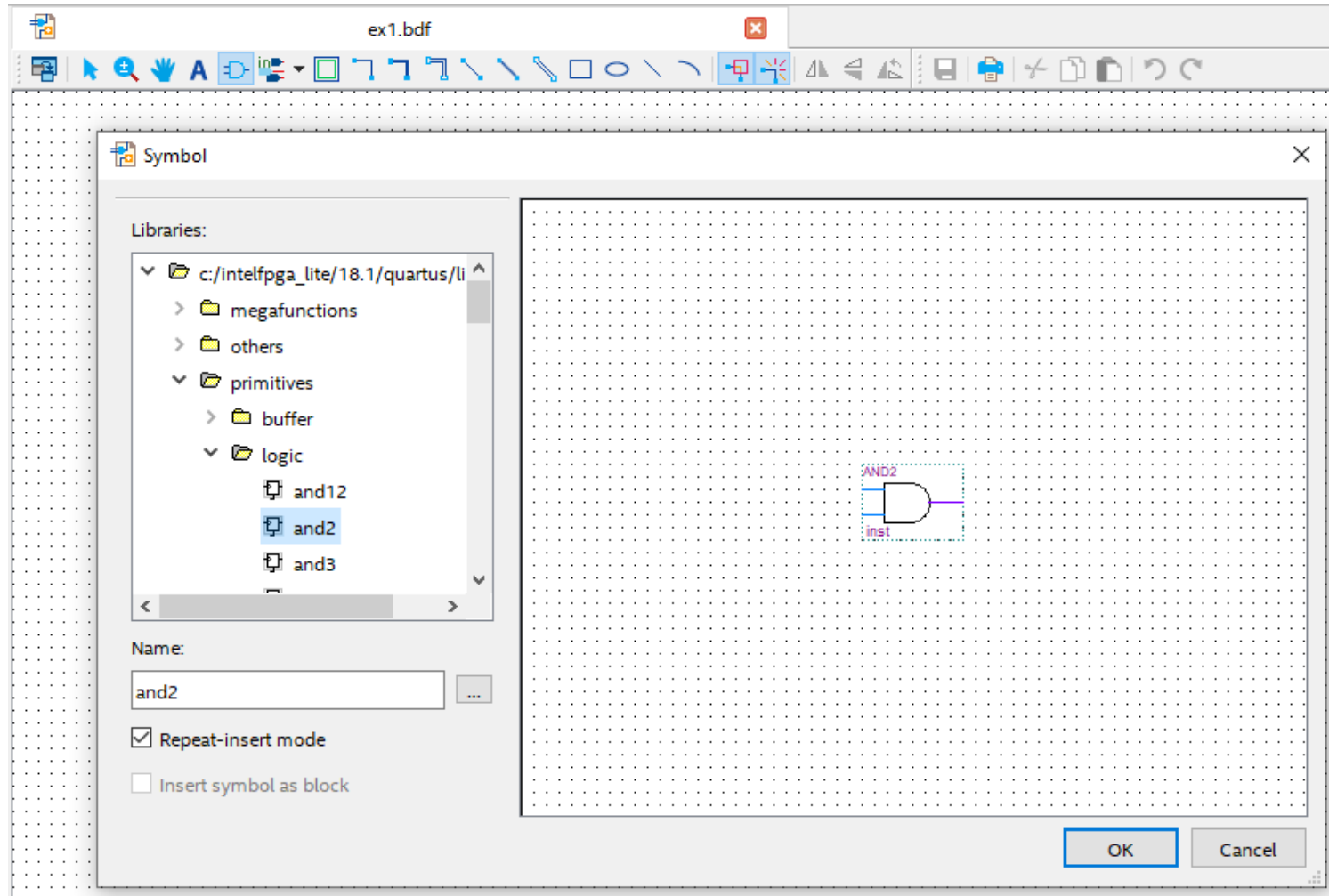
Implementando circuitos com diagrama de blocos

Nessa aula, iremos implementar utilizando diagrama de blocos



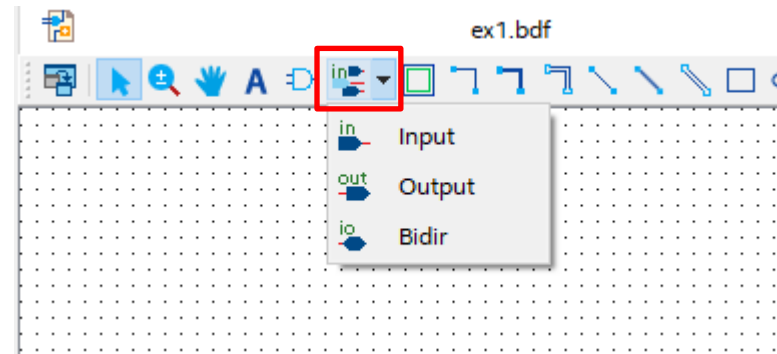
Implementando circuitos com diagrama de blocos

- Clique em Symbol tool, e na janela que abrir, você encontrará as portas lógicas no caminho “.../quartus/libraries/ > primitives > logic”

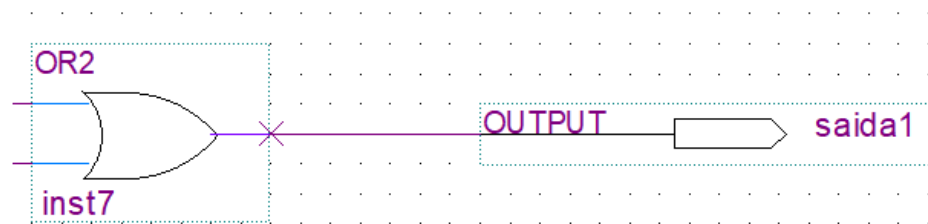


Implementando circuitos com diagrama de blocos

- Os pinos de entrada/saída são encontrados em pin tool, conforme visto abaixo:



Se aparecer o “x” em alguma conexão, é sinal que a conexão está com problema e dará erro de compilação. Delete e faça novamente



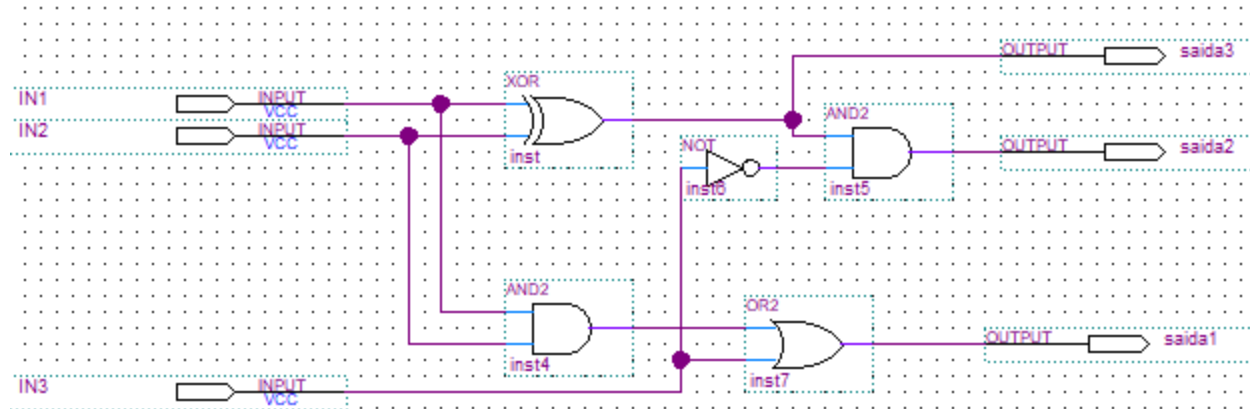
Implementando circuitos com diagrama de blocos

- Também é possível associar os sinais através de nomes, sem ter uma conexão física.



Implementando circuitos com diagrama de blocos

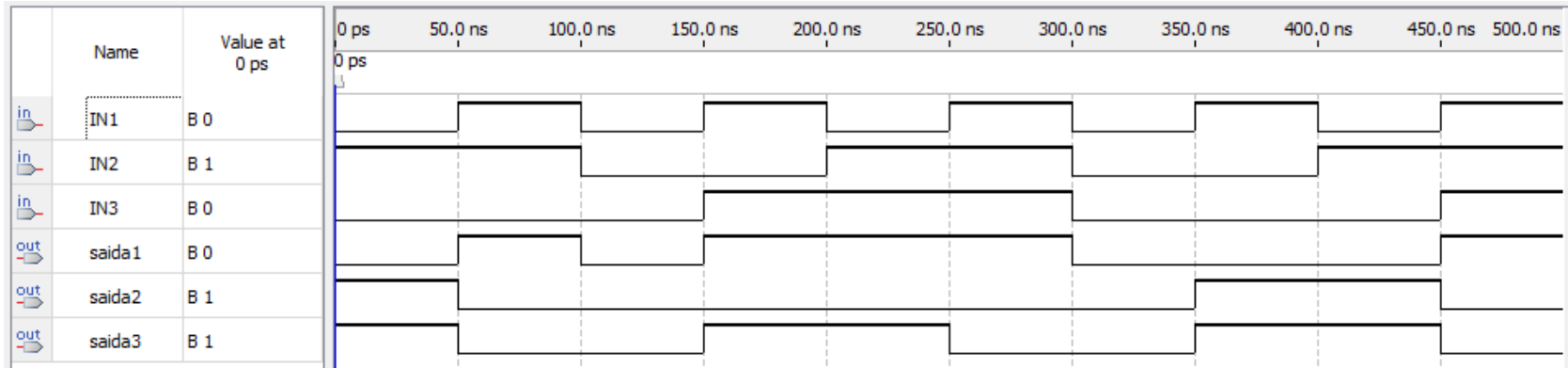
- Salve e compile, faça a simulação.



- Valores para simulação:
- Edit > Set end time : 500 ns.
- Edit > Grid size : 50 ns.
- IN1: Start value: 0; count every: 50 ns;
- IN2: Start value: 1; count every: 100 ns;
- IN3: Start value: 0; count every: 150 ns;

Implementando circuitos com diagrama de blocos

■ Resultado:



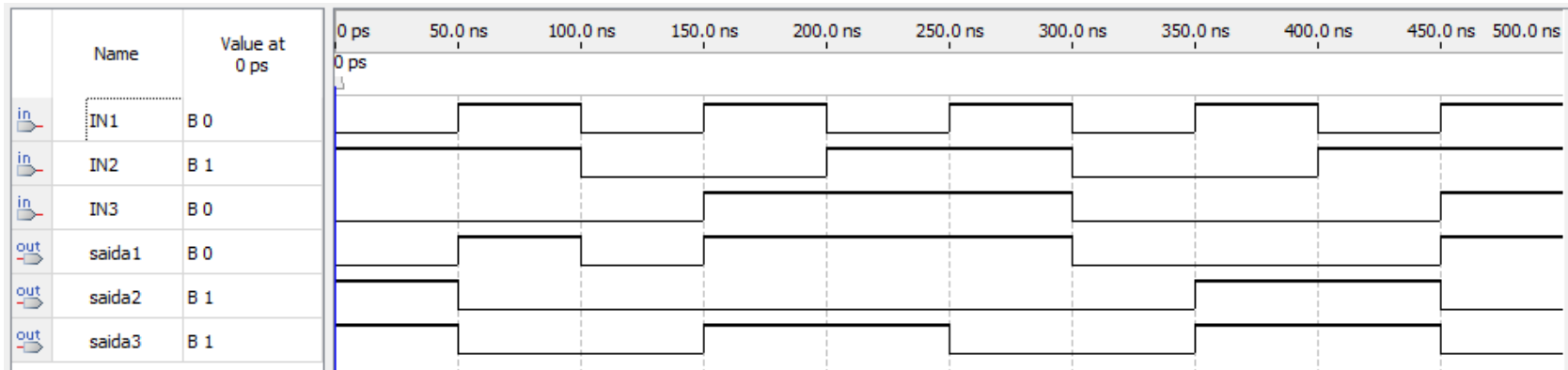
Obs. O Quartus tem alguns “bugs”, caso durante a compilação da simulação aparecer a mensagem:

```
# ** Warning: (vcom-6) -- Waiting for lock by <pc utilizado>
```

Vá até a pasta `/simulation/qsim/work/` dentro do projeto e delete o arquivo “`_lock`” durante a simulação.

Implementando circuitos com diagrama de blocos

- Resultado:



- No Pin Planner, Utilize DIP Switch SW[0-2] como entradas e LEDR [0-2] como saídas. Compile novamente e grave na FPGA.
- Página 24, 25 e 26 do manual.

Table 3-6 Pin Assignment of Slide Switches

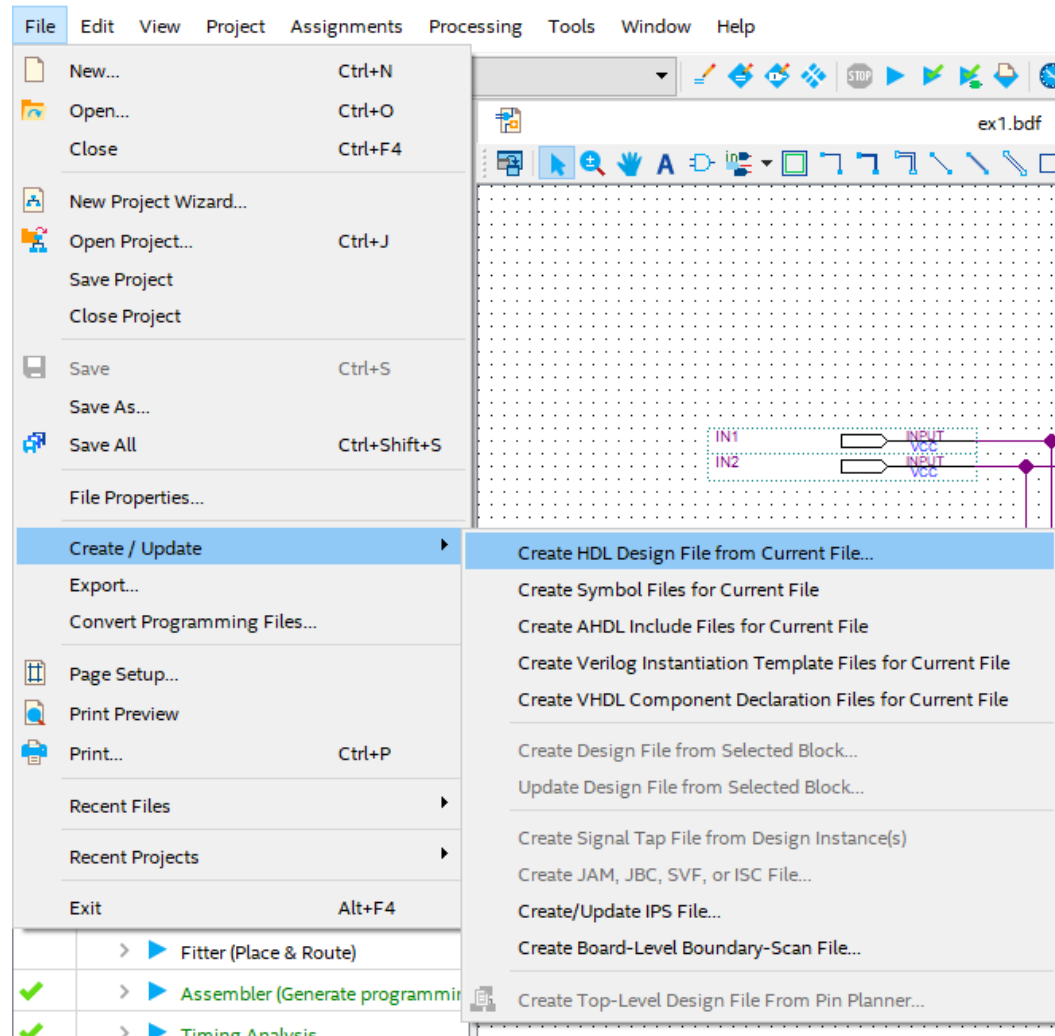
Signal Name	FPGA Pin No.	Description	I/O Standard
SW[0]	PIN_AB12	Slide Switch[0]	3.3V
SW[1]	PIN_AC12	Slide Switch[1]	3.3V
SW[2]	PIN_AF9	Slide Switch[2]	3.3V

Table 3-8 Pin Assignment of LEDs

Signal Name	FPGA Pin No.	Description	I/O Standard
LEDR[0]	PIN_V16	LED [0]	3.3V
LEDR[1]	PIN_W16	LED [1]	3.3V
LEDR[2]	PIN_V17	LED [2]	3.3V

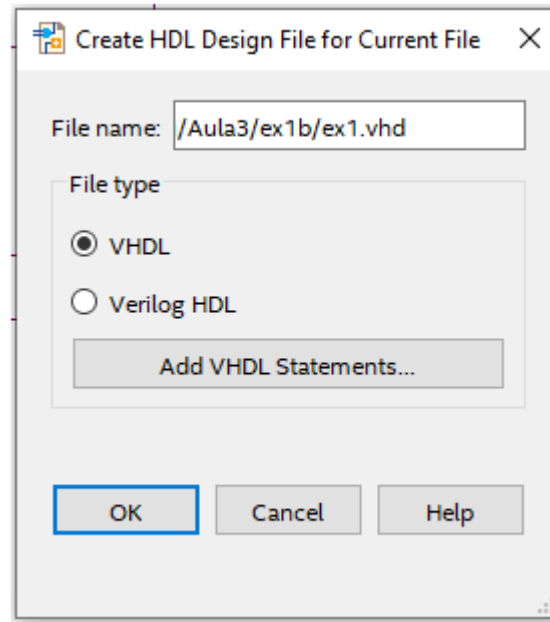
Implementando circuitos com diagrama de blocos

Exercício 1b: Também é possível converter o diagrama de blocos em código VHDL, para isso, selecione o arquivo .bdf e siga os passos:

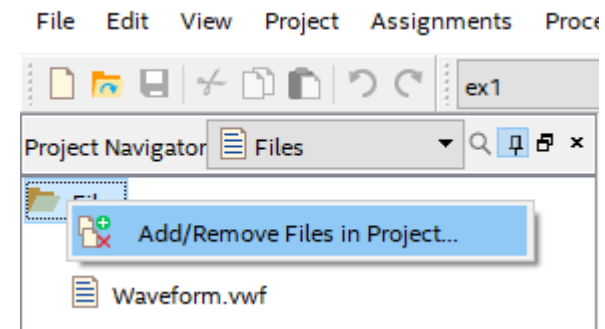


Exercício 1b

- Selecione VHDL e clique em OK. Agora você tem o arquivo .vhd na pasta do projeto.

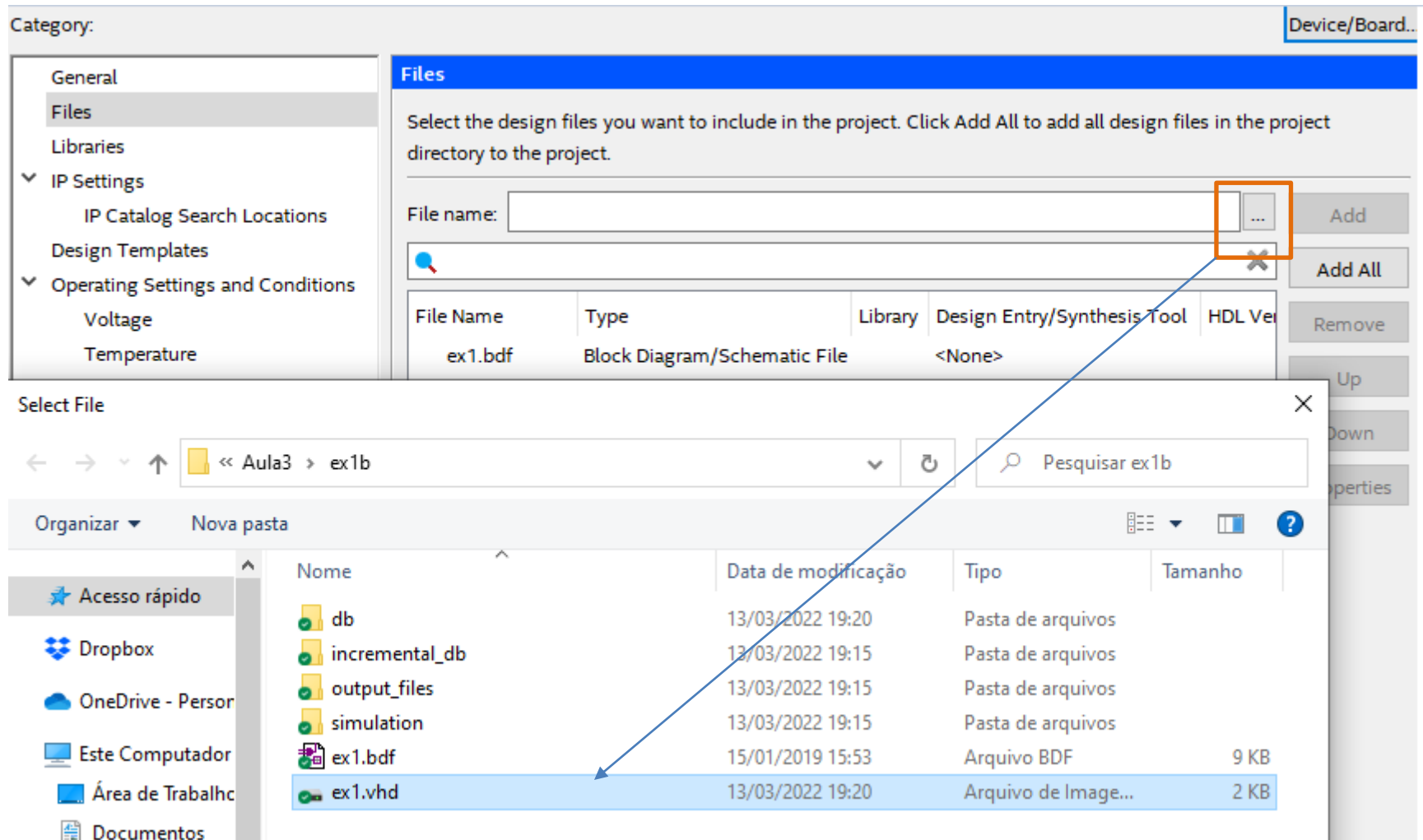


- Para adicioná-lo no Quartus, clique com o botão direito em File, na árvore de arquivos, e clique em Add/remove...



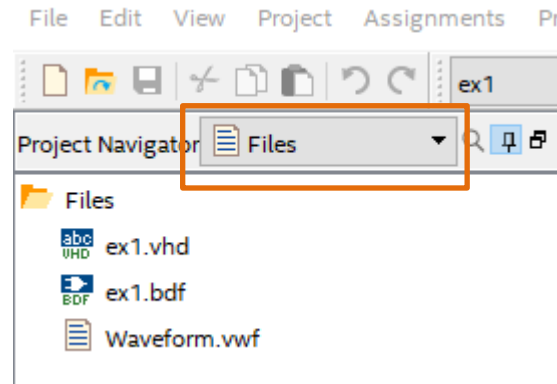
Exercício 1b

- Localize o arquivo e clique em abrir e depois OK.



Exercício 2

- Certifique-se se estar na aba “Files” e dê dois cliques no arquivo .vhd.



- Pronto! Agora você tem acesso ao código VHDL caso queira reaproveitar em outro projeto.

Exercício 2

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

LIBRARY work;

ENTITY ex1 IS
  PORT
  (
    IN1 : IN  STD_LOGIC;
    IN2 : IN  STD_LOGIC;
    IN3 : IN  STD_LOGIC;
    saida3 : OUT STD_LOGIC;
    saida2 : OUT STD_LOGIC;
    saida1 : OUT STD_LOGIC
  );
END ex1;

ARCHITECTURE bdf_type OF ex1 IS

  SIGNAL SYNTHESIZED_WIRE_0 : STD_LOGIC;
  SIGNAL SYNTHESIZED_WIRE_1 : STD_LOGIC;
  SIGNAL SYNTHESIZED_WIRE_2 : STD_LOGIC;

BEGIN
  saida3 <= SYNTHESIZED_WIRE_0;

  SYNTHESIZED_WIRE_0 <= IN1 XOR IN2;

  SYNTHESIZED_WIRE_2 <= IN1 AND IN2;

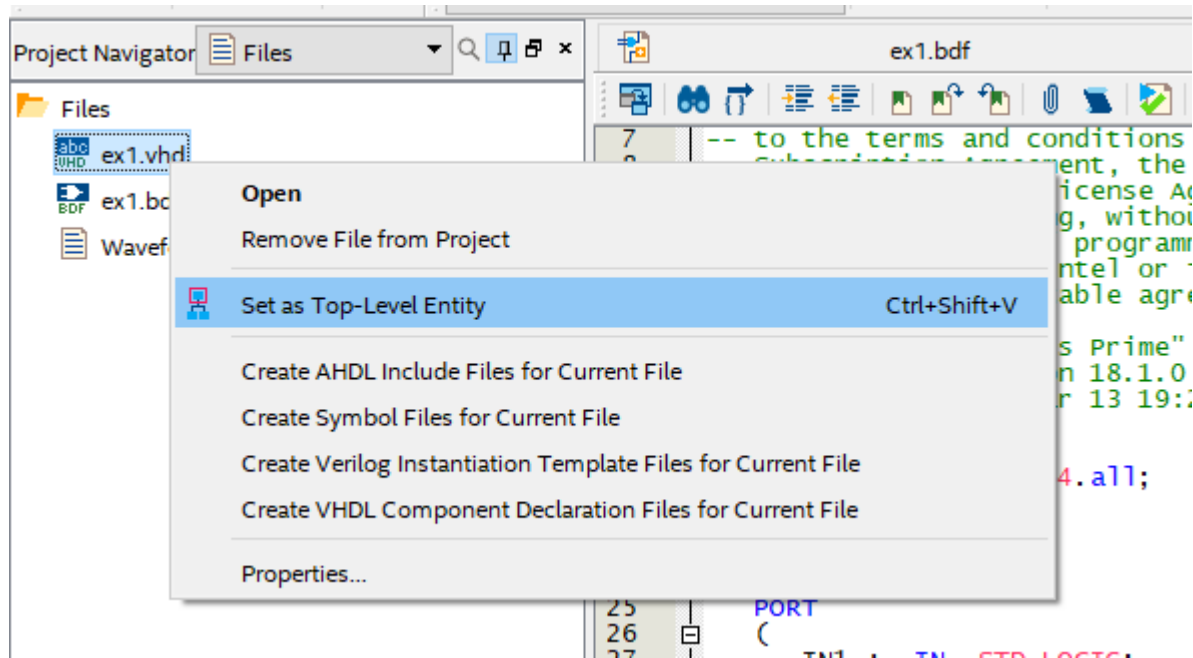
  saida2 <= SYNTHESIZED_WIRE_0 AND SYNTHESIZED_WIRE_1;

  SYNTHESIZED_WIRE_1 <= NOT(IN3);

  saida1 <= IN3 OR SYNTHESIZED_WIRE_2;
END bdf_type;
```

Exercício 1b

- Para compilar esse arquivo, coloque ele como top-level entity (equivalente ao “main”).

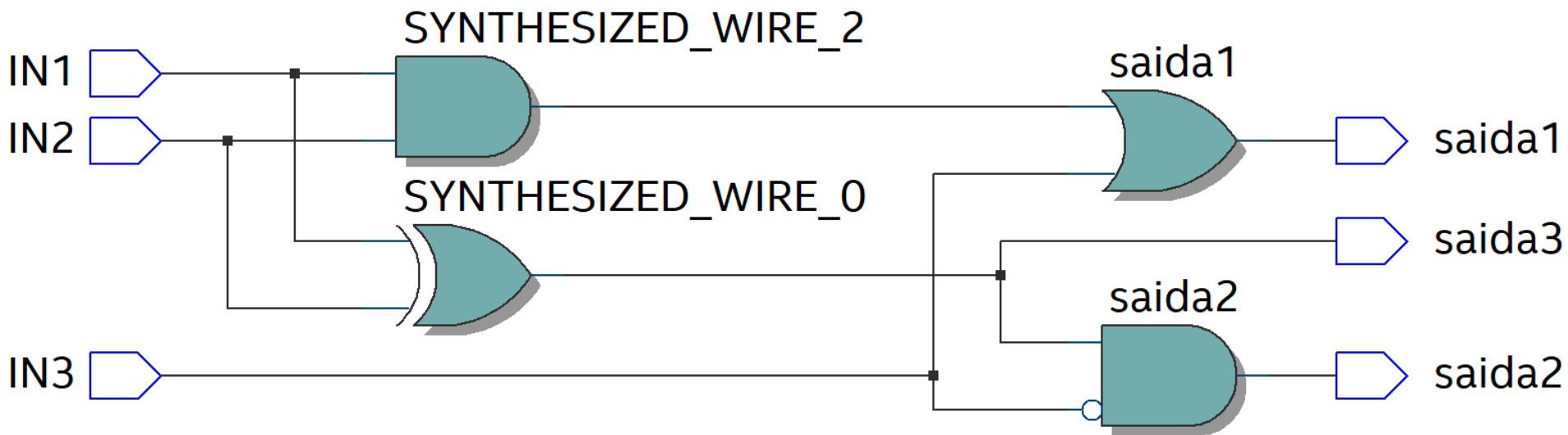


- Você não conseguirá compilar por ter o mesmo nome do .bdf, então tire o .bdf do projeto e compile o .vhd. Após, abra o RTL VIEWER. TOOLS>NETLIST VIEWER> RTL VIEWER

Exercício 1b

- Agora você consegue ver o que são os signal:

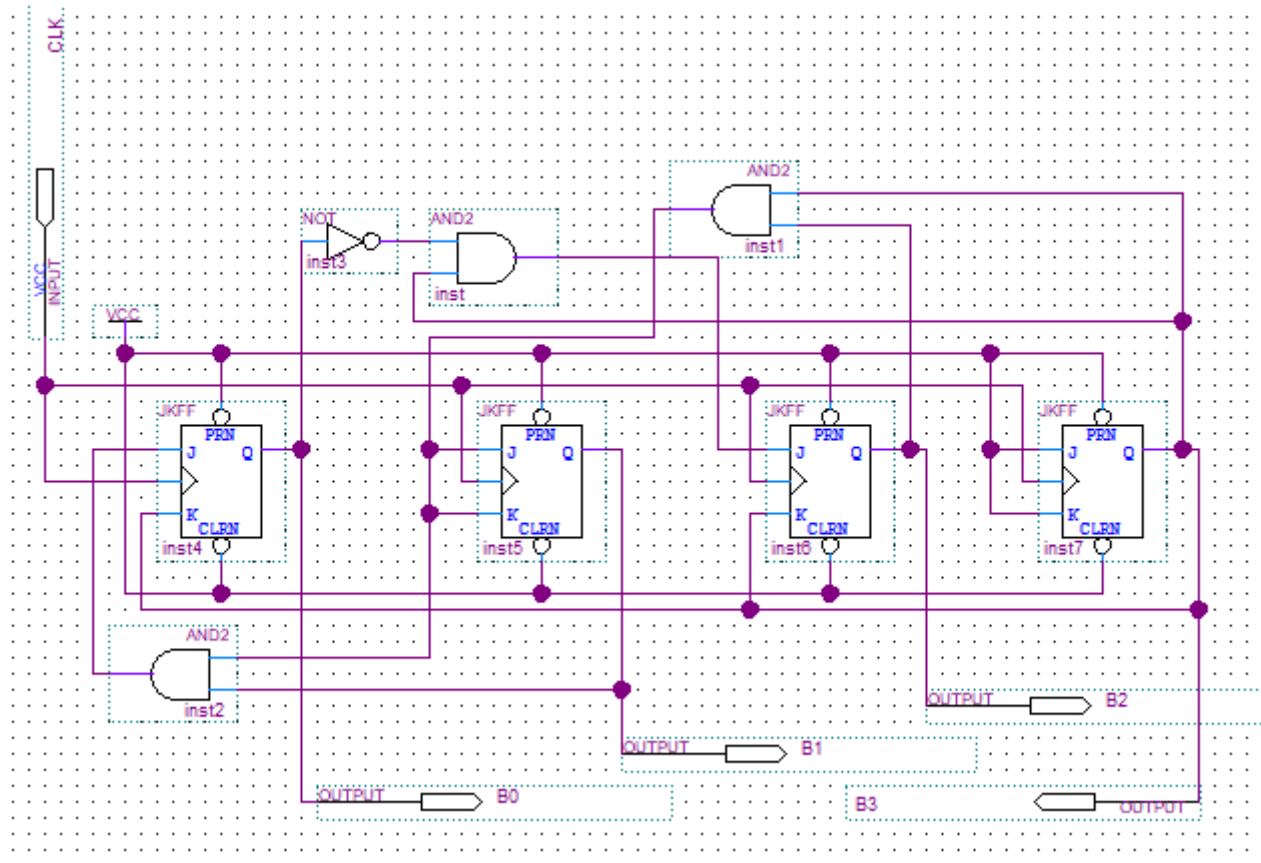
```
SIGNAL SYNTHESIZED_WIRE_0 : STD_LOGIC;  
SIGNAL SYNTHESIZED_WIRE_1 : STD_LOGIC;  
SIGNAL SYNTHESIZED_WIRE_2 : STD_LOGIC;
```



- Signal é um sinal intermediário, como se fosse um condutor físico entre um componente e outro. Nas próximas aulas aprenderemos como utilizá-lo.

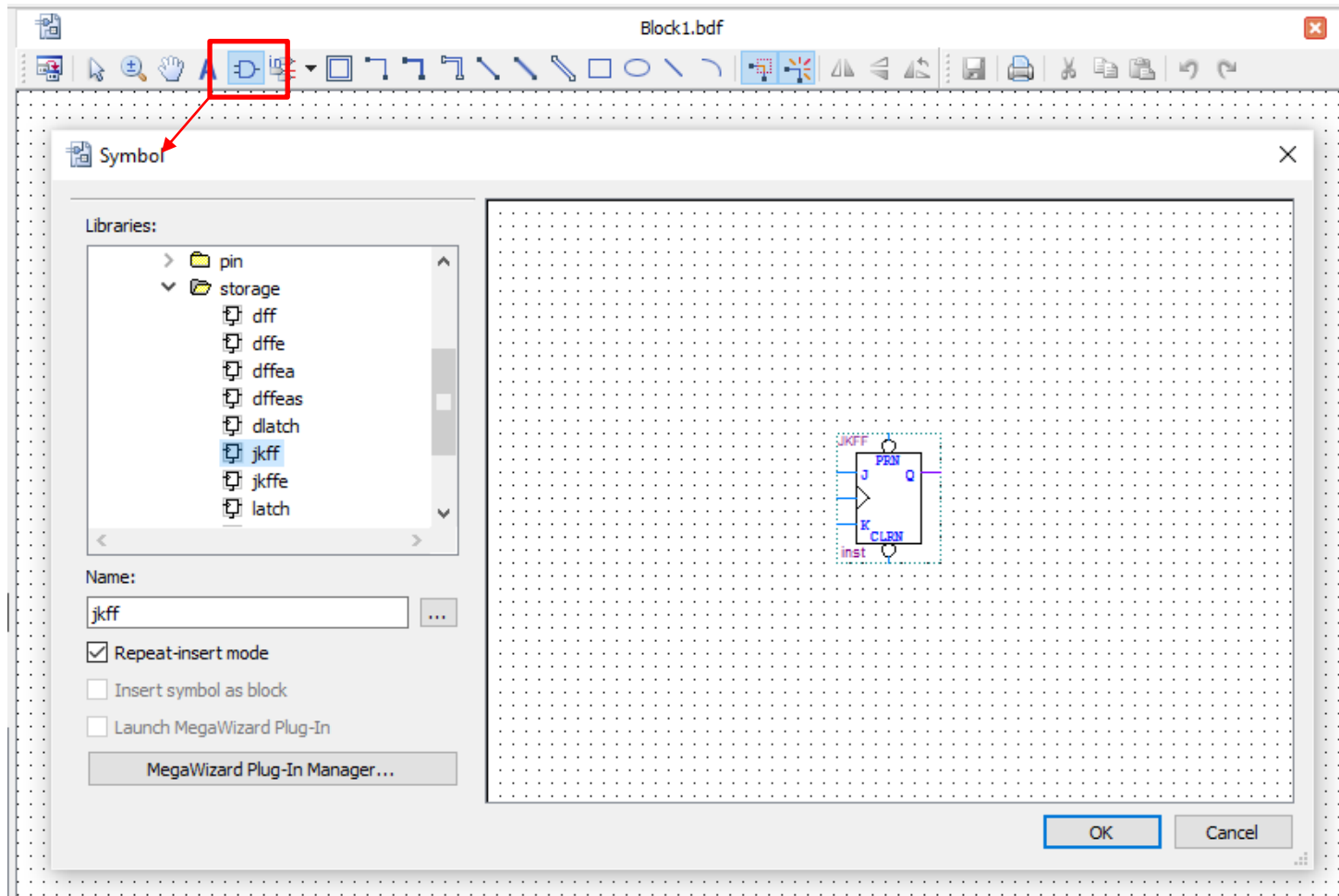
Implementando circuitos com diagrama de blocos

Exercício 2: Implementar um contador de 4 bits: (0 – 9). Compile e simule o circuito.



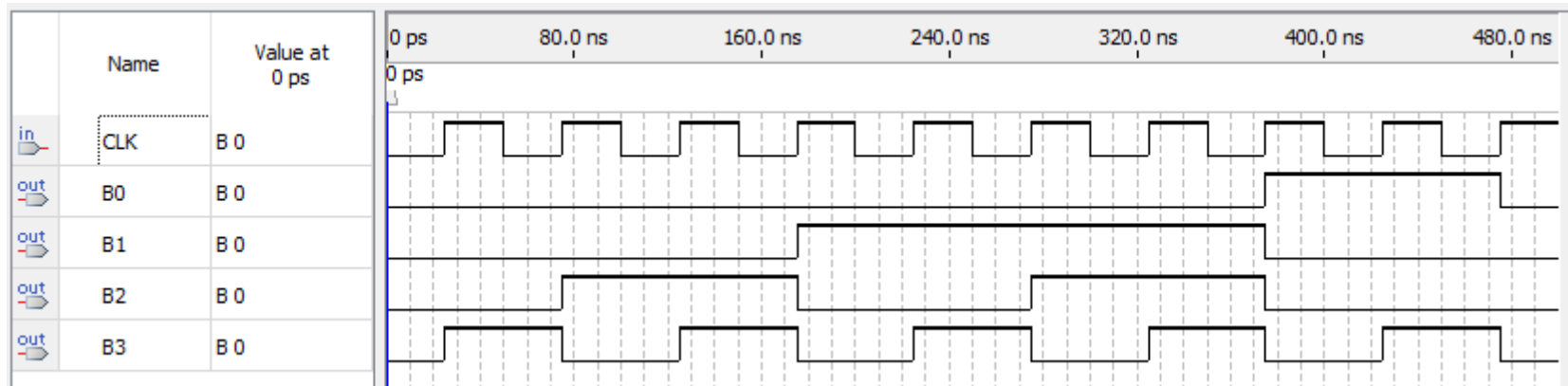
Implementando circuitos com diagrama de blocos

Clique em Symbol tool, e na janela que abrir, para encontrar o Flip-Flop JK, vá até primitives > storage > jkff, após, encontre as portas lógicas.

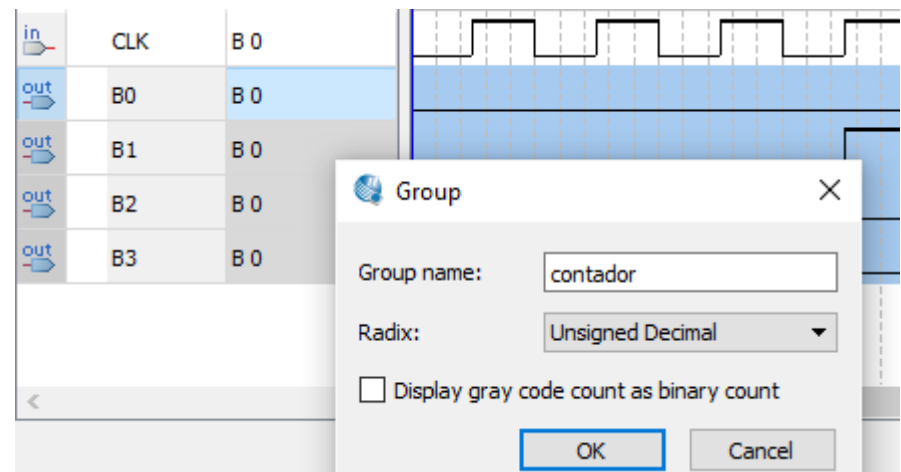


Implementando circuitos com diagrama de blocos

Resultado da simulação:

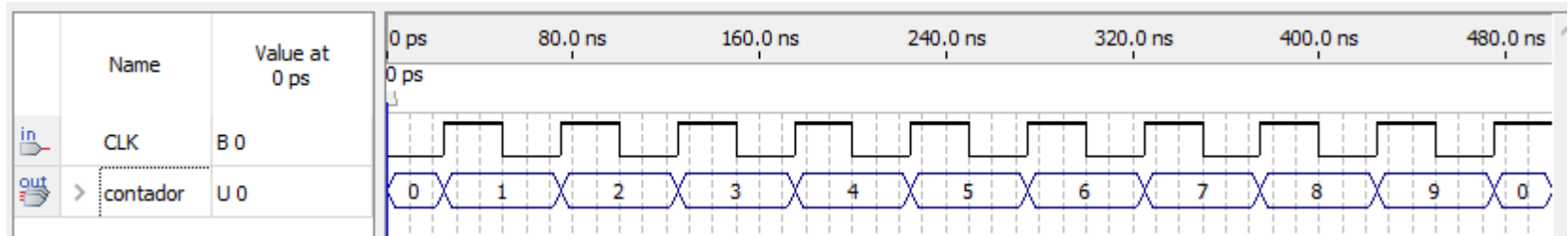


Para facilitar a visualização da contagem, vamos criar um grupo com os bits de saída, e mudar o tipo de dado. Para isso, selecione os 4 bits > botão direito > grouping > group, e mude o tipo de dado de acordo com a imagem abaixo:



Implementando circuitos com diagrama de blocos

Resultado da simulação:



Utilize como entrada de clock a KEY[0], e como saída, os LEDR [0-3].

Table 3-7 Pin Assignment of Push-buttons

Signal Name	FPGA Pin No.	Description	I/O Standard
KEY[0]	PIN_AA14	Push-button[0]	3.3V

Table 3-8 Pin Assignment of LEDs

Signal Name	FPGA Pin No.	Description	I/O Standard
LEDR[0]	PIN_V16	LED [0]	3.3V
LEDR[1]	PIN_W16	LED [1]	3.3V
LEDR[2]	PIN_V17	LED [2]	3.3V
LEDR[3]	PIN_V18	LED [3]	3.3V

Implementando circuitos com diagrama de blocos

- Exercício 3: Faça o mesmo exercício anterior, utilizando o CI contador 7490 e o CI conversor BCD – 7 Segmentos 7447. Grave no KIT utilizando os seguintes pinos:
- Entradas:
 - *KEY[0]* para a entrada de clock
 - *KEY[1]* para o teste dos segmentos
- Saídas:
 - *HEX0[0..6]* para o display 7 segmentos "0".

Table 3-9 Pin Assignment of 7-segment Displays

Signal Name	FPGA Pin No.	Description	I/O Standard
HEX0[0]	PIN_AE26	Seven Segment Digit 0[0]	3.3V
HEX0[1]	PIN_AE27	Seven Segment Digit 0[1]	3.3V
HEX0[2]	PIN_AE28	Seven Segment Digit 0[2]	3.3V
HEX0[3]	PIN_AG27	Seven Segment Digit 0[3]	3.3V
HEX0[4]	PIN_AF28	Seven Segment Digit 0[4]	3.3V
HEX0[5]	PIN_AG28	Seven Segment Digit 0[5]	3.3V
HEX0[6]	PIN_AH28	Seven Segment Digit 0[6]	3.3V

Table 3-7 Pin Assignment of Push-buttons

Signal Name	FPGA Pin No.	Description	I/O Standard
KEY[0]	PIN_AA14	Push-button[0]	3.3V
KEY[1]	PIN_AA15	Push-button[1]	3.3V

CI's com funções vistas nas últimas aulas

Contador de década – 7490

Funções dos pinos

INPUT A - Entrada de clock do contador A (acionada por borda de descida).

INPUT B - Entrada de clock do contador B (acionada por borda de descida).

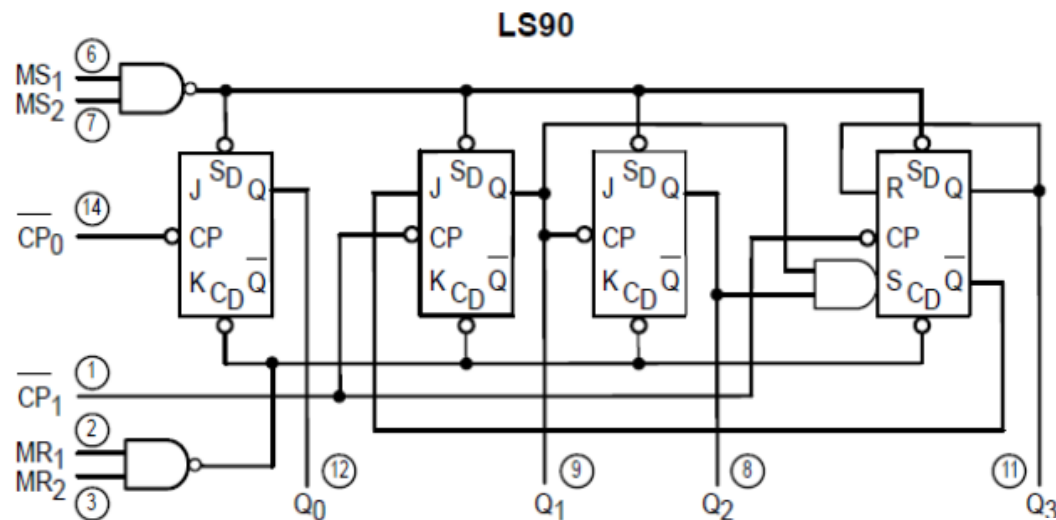
R0(1) , R0(2) - Em nível alto zero (0000) a saída do contador.

R9(1) , R9(2) - Em nível alto leva a nove (1001) a saída do contador.

QA, QB, QC, QD - Saídas do contador.

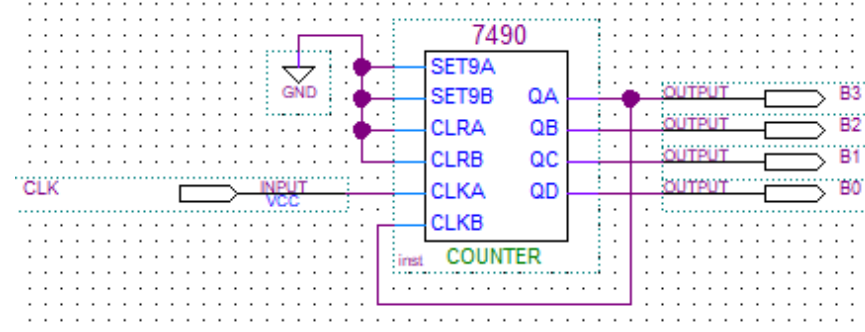
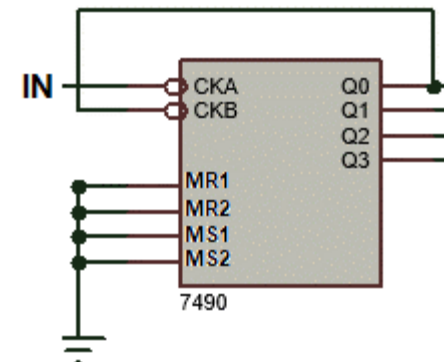
MS = R9

MR = R0

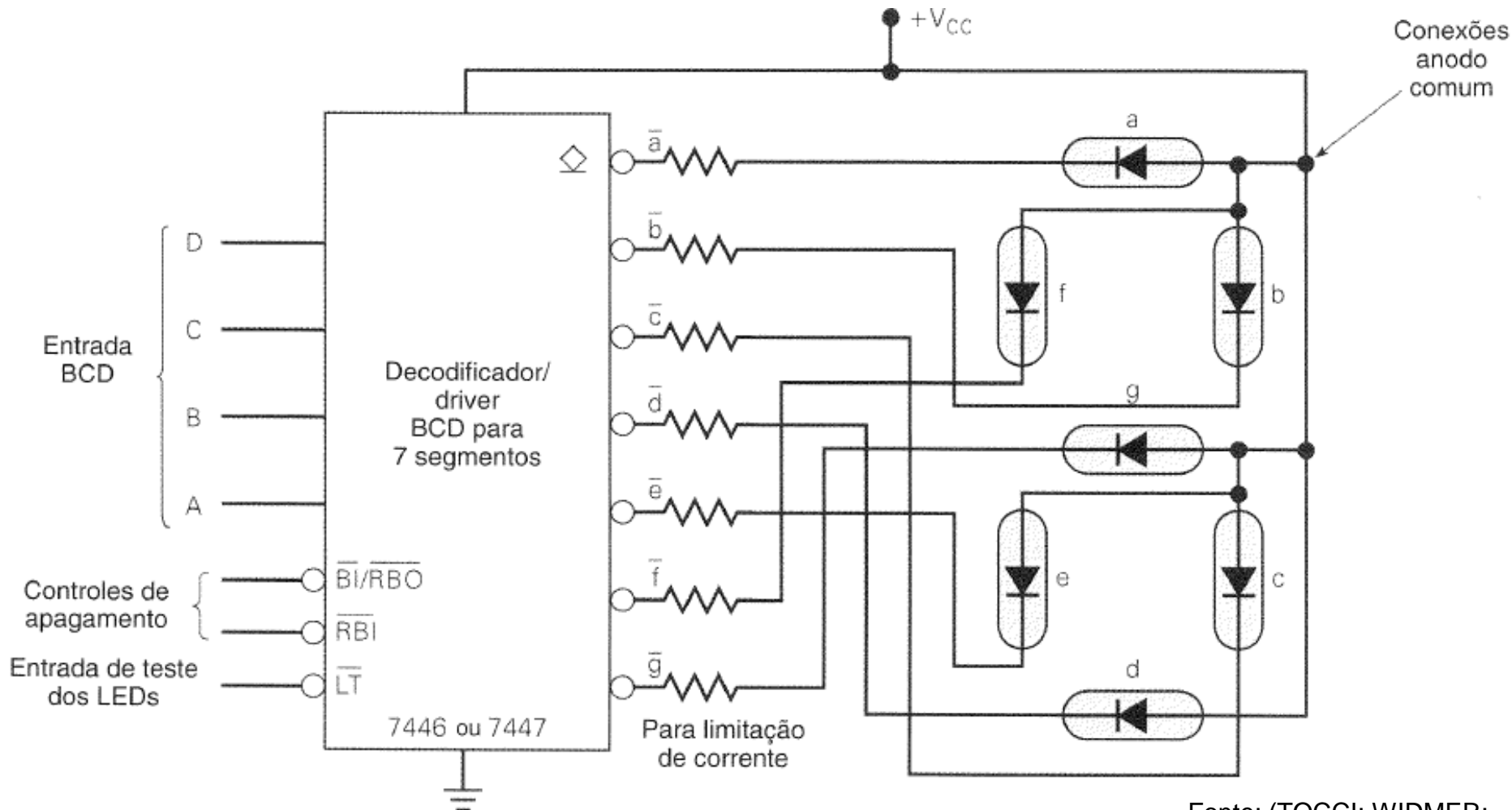


○ = PIN NUMBERS
VCC = PIN 5
GND = PIN 10

SD = PRESET
CD = CLEAR



Codificadores e decodificadores

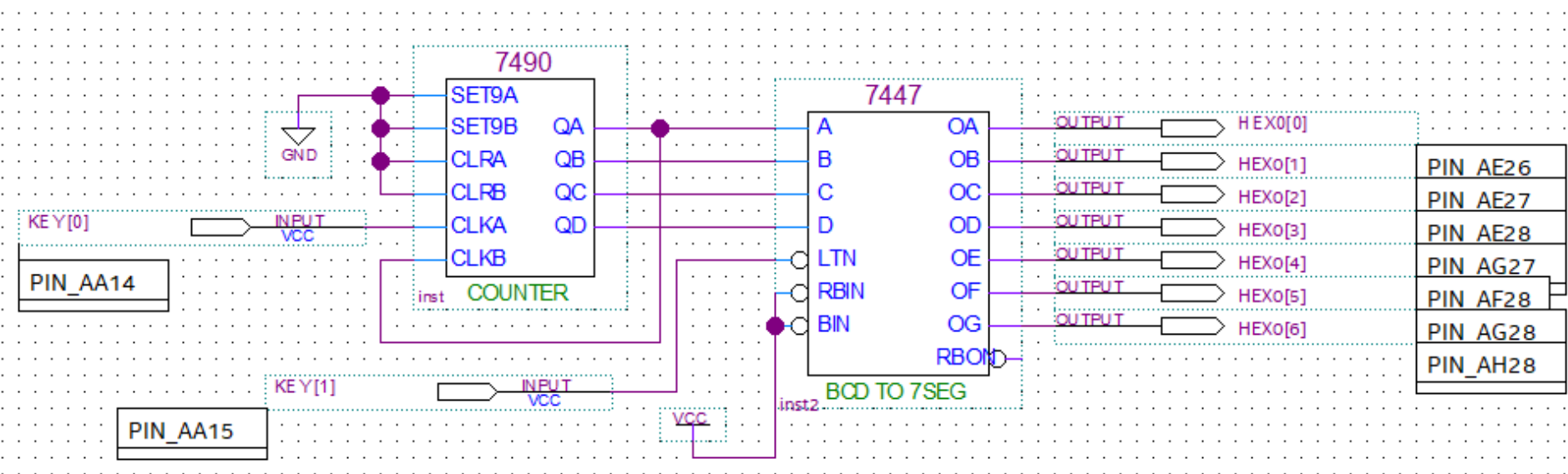


Fonte: (TOCCI; WIDMER; MOSS, 2011)

- LT (lamp test) – acende todos os segmentos.
- RBO – desliga todos os segmentos, também pode ser utilizado para controlar a intensidade dos segmentos.

Implementando circuitos com diagrama de blocos

- Exercício 3: Faça o mesmo exercício anterior, utilizando o CI contador 7490 e o CI conversor BCD – 7 Segmentos 7447.



- Próxima aula: Classes de Objetos, tipos de dados e atributos.