

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

Lógica Reconfigurável

Utilizando o kit de desenvolvimento

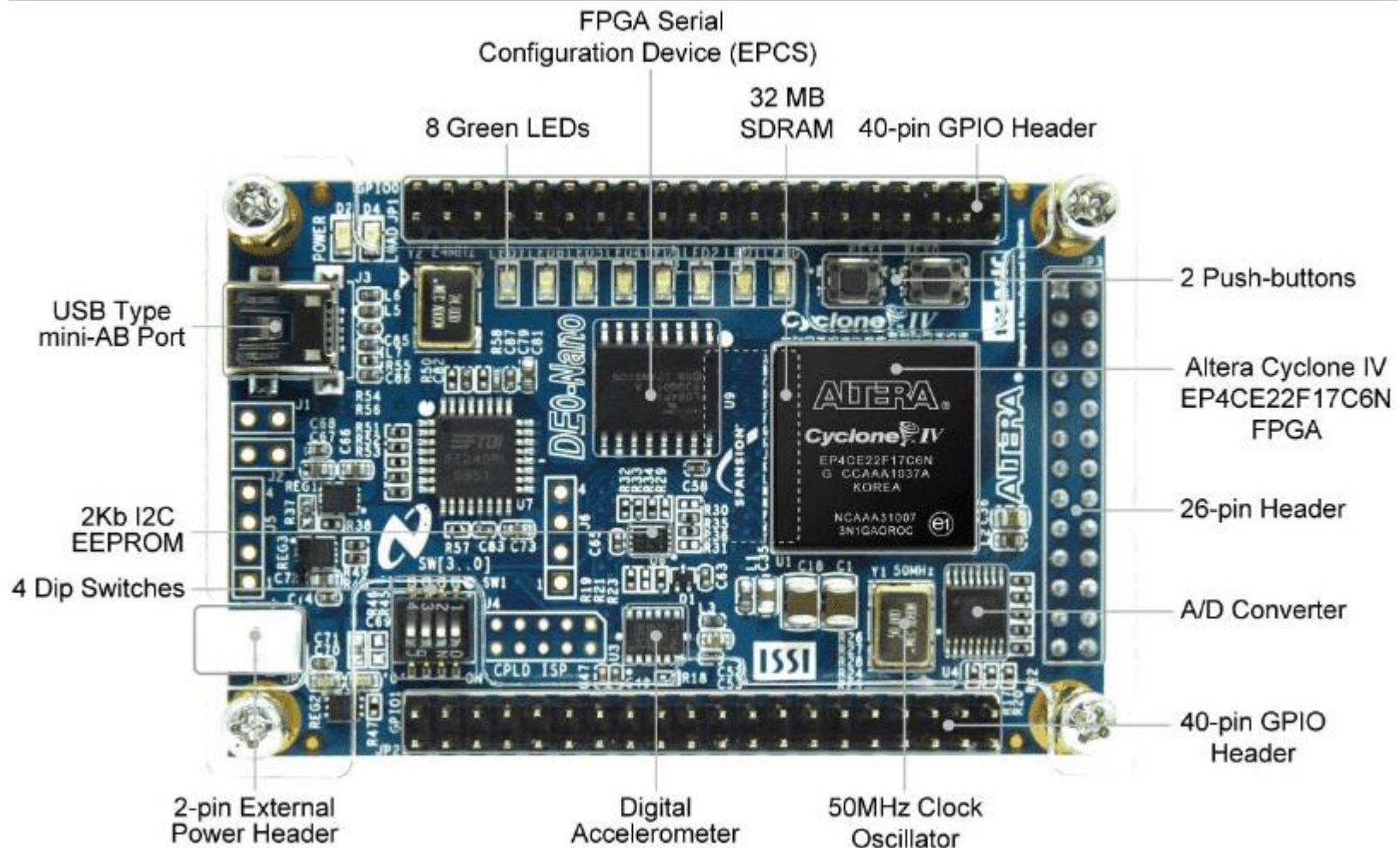
Tiago Piovesan Vendruscolo



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/)

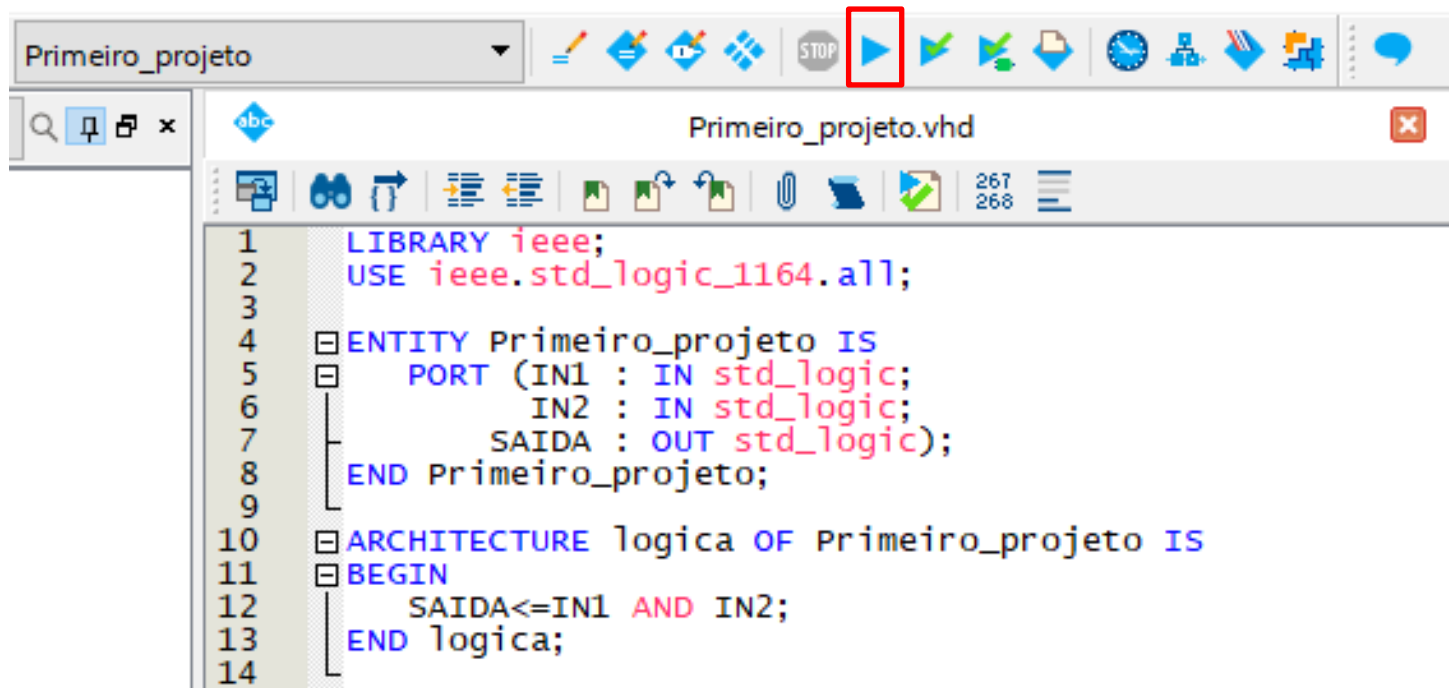
FPGA utilizada nas aulas

DE0 nano – ALTERA

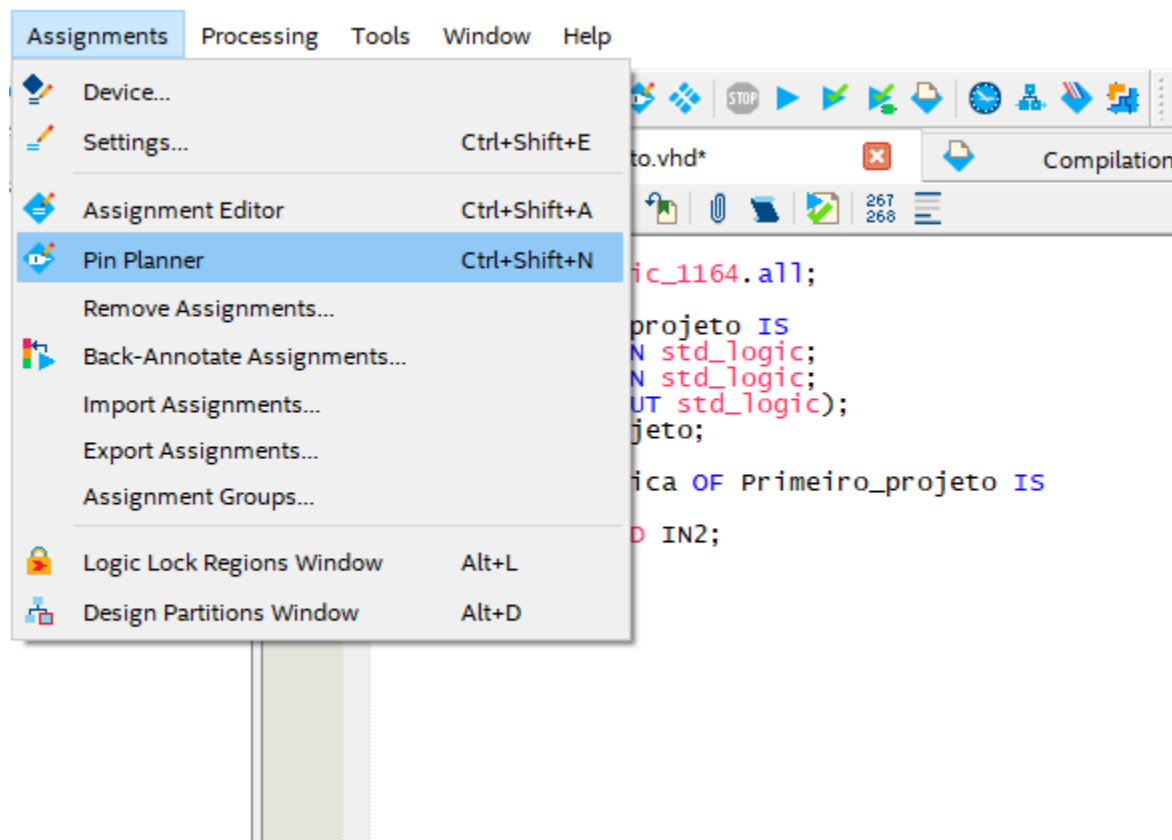


VHDL – Exemplo 1

- Compile o primeiro código feito na aula anterior:



VHDL – Gravando o código na FPGA



VHDL – Gravando o código na FPGA

File Edit View Processing Tools Window Help

Search altera.com

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 IV E - EP4CE22F17C6

Named: * Edit: Filter: Pins: all

Node Name	Direction	Location	I/O Bank	VREF Group	I/O Standard	Reserved
IN1	Input				2.5 V (default)	
IN2	Input				2.5 V (default)	
SAIDA	Output				2.5 V (default)	
<<new node>>						

Moodle - DE0_nano_user_manual

Pag. 14

Signal Name	FPGA Pin No.
LED[0]	PIN_A15
LED[1]	PIN_A13
LED[2]	PIN_B13
LED[3]	PIN_A11

Pag. 15

Signal Name	FPGA Pin No.
DIP Switch[0]	PIN_M1
DIP Switch[1]	PIN_T8
DIP Switch[2]	PIN_B9
DIP Switch[3]	PIN_M15

VHDL – Gravando o código na FPGA

File Edit View Processing Tools Window Help

Search altera.com

Report

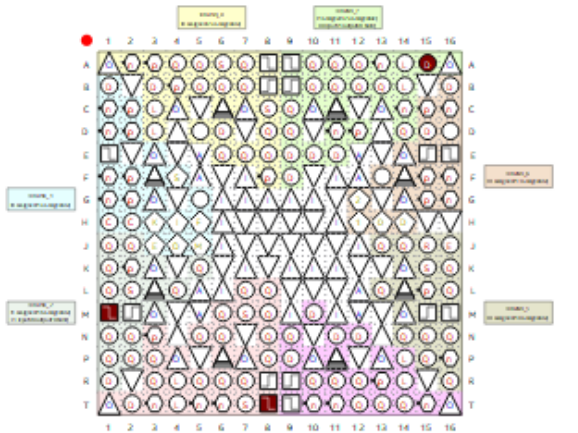
Report not available

Groups Report

Tasks

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

Top View - Wire Bond
Cyclone IV E - EP4CE22F17C6



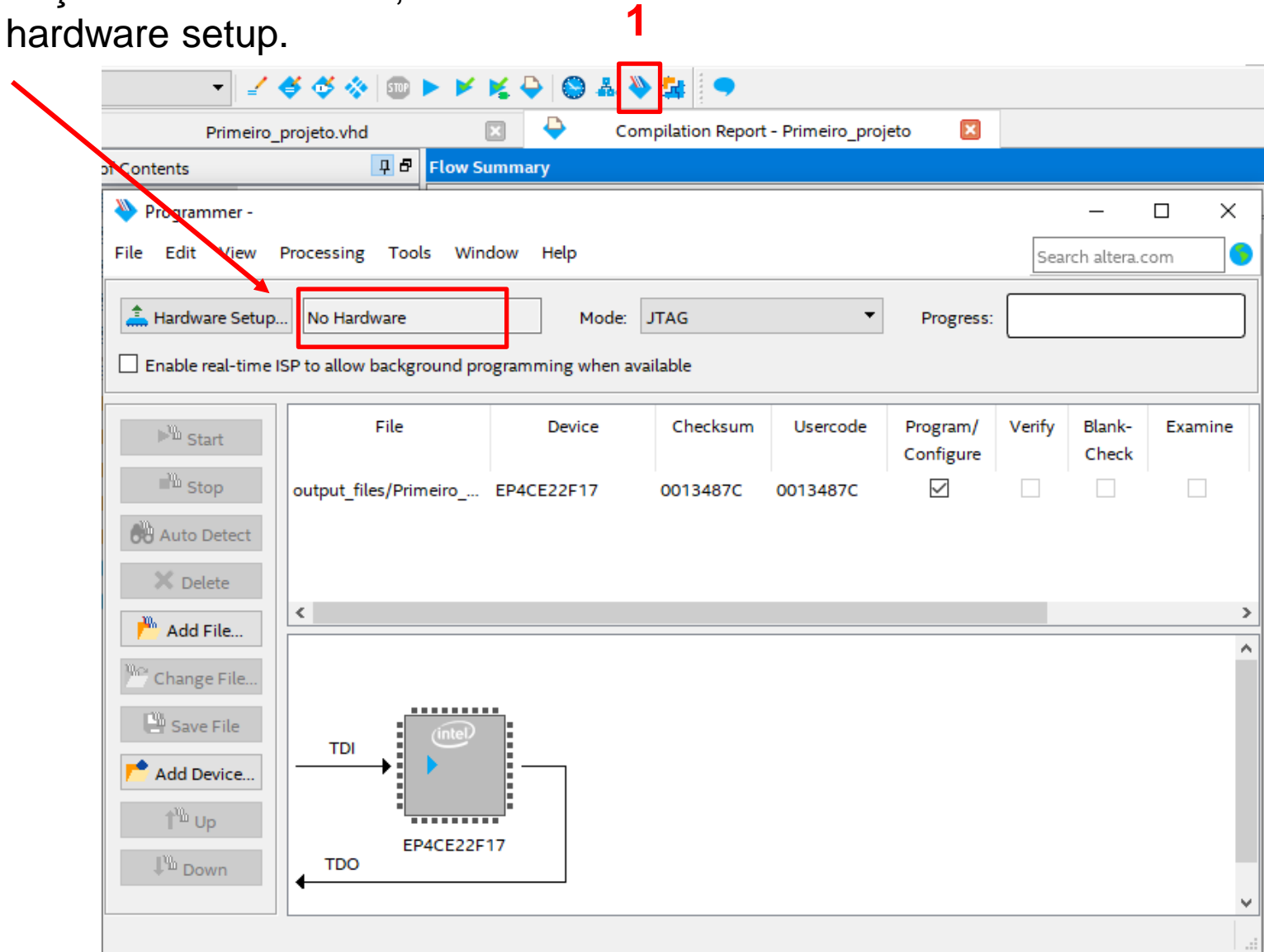
Named: * Edit: Filter: Pins: all

Node Name	Direction	Location	I/O Bank	VREF Group	I/O Standard	Reserved
in IN1	Input	PIN_M1	2	B2_N0	2.5 V (default)	
in IN2	Input	PIN_T8	3	B3_N0	2.5 V (default)	
out SAIDA	Output	PIN_A15	7	B7_N0	2.5 V (default)	
<<new node>>						

Após colocar todos os pinos, feche a janela e compile novamente.

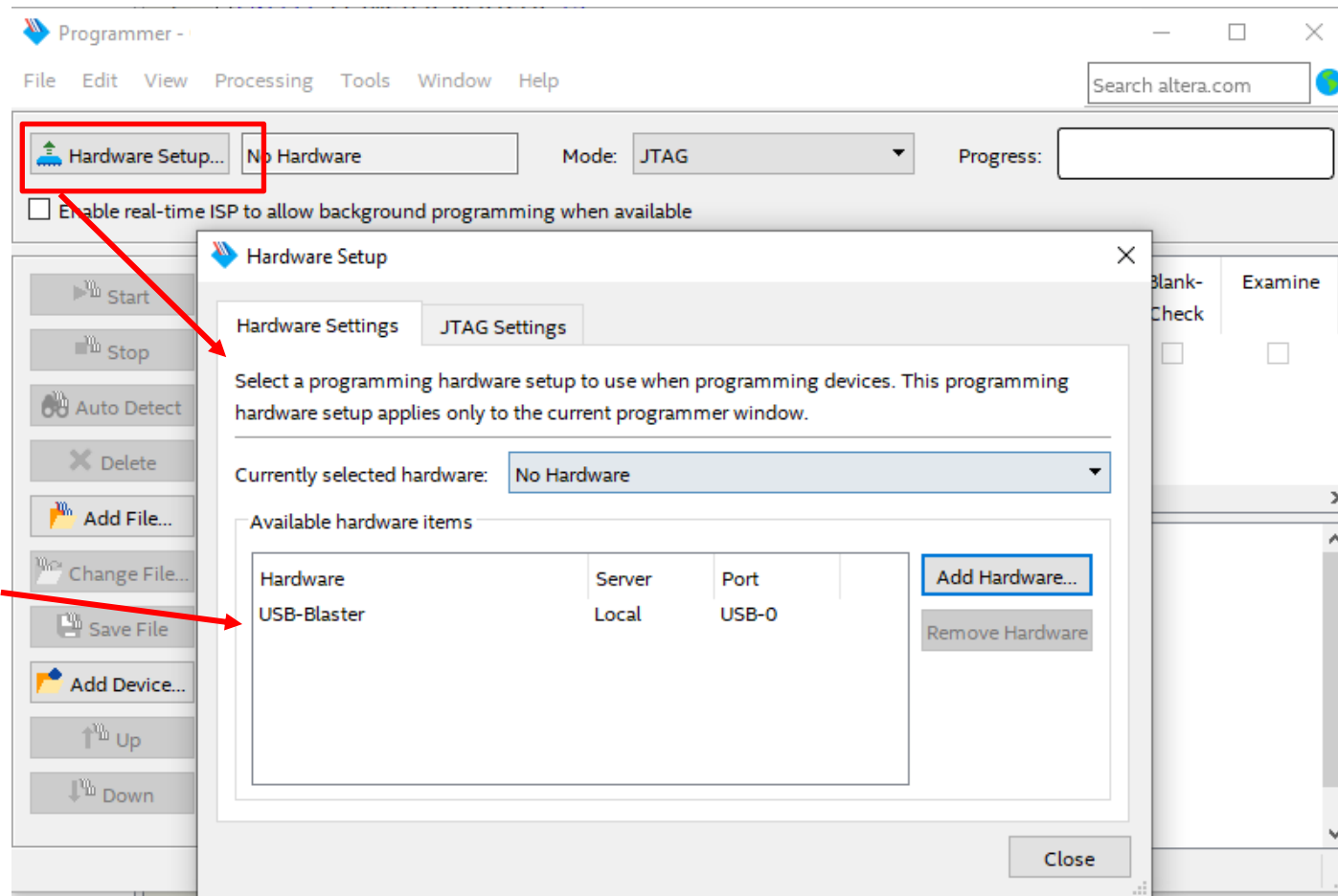
VHDL – Gravando o código na FPGA

Caso apareça “no hardware”,
clique em hardware setup.



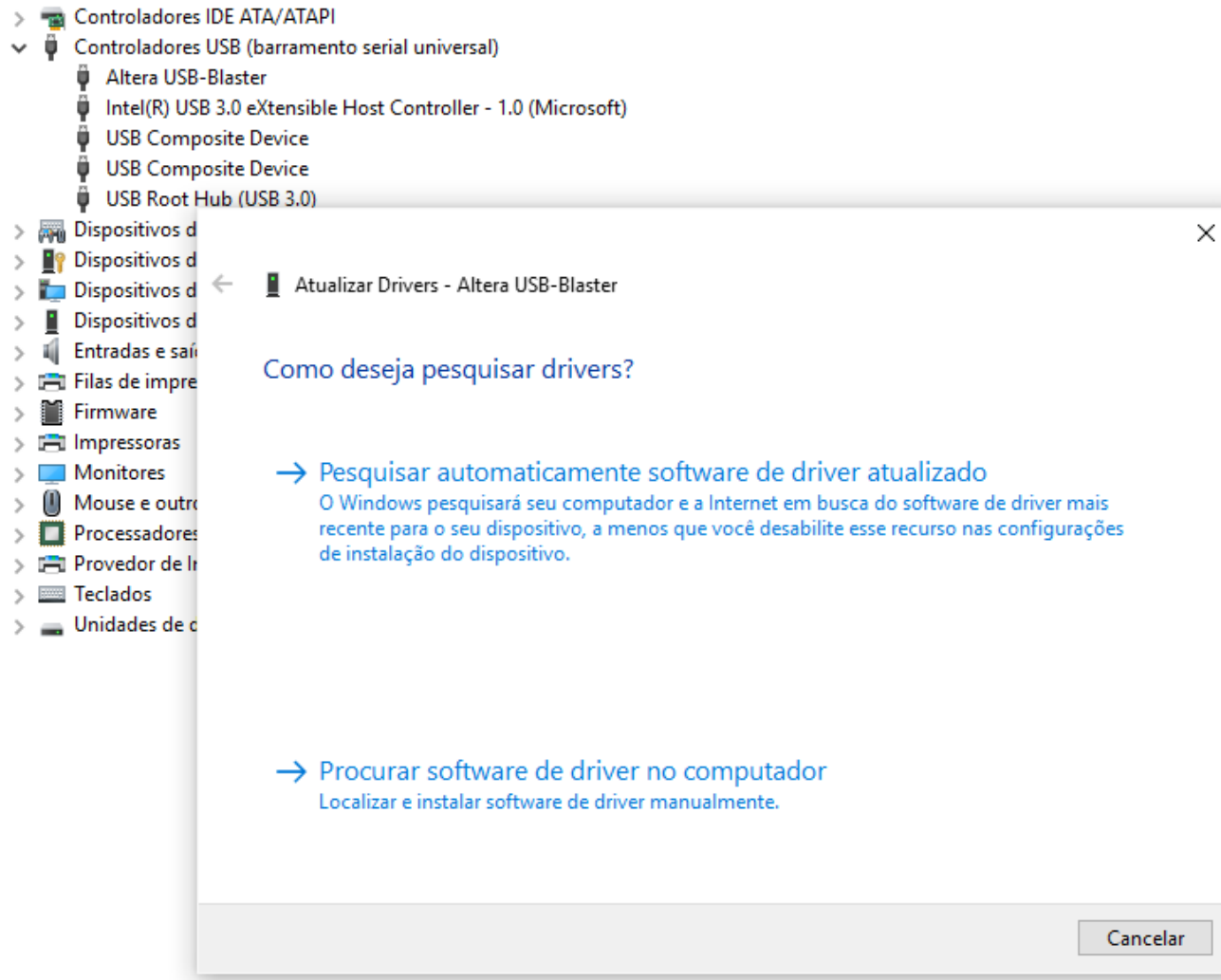
VHDL – Gravando o código na FPGA

Caso apareça “no hardware”,
clique em hardware setup.



Dois cliques em
“USB-Blaster” e
após em “close”.

Instalando o driver Altera USB-Blaster (pode aparecer como dispositivo desconhecido)
Gerenciador de dispositivos > Atualizar driver > Procurar software de driver no PC.



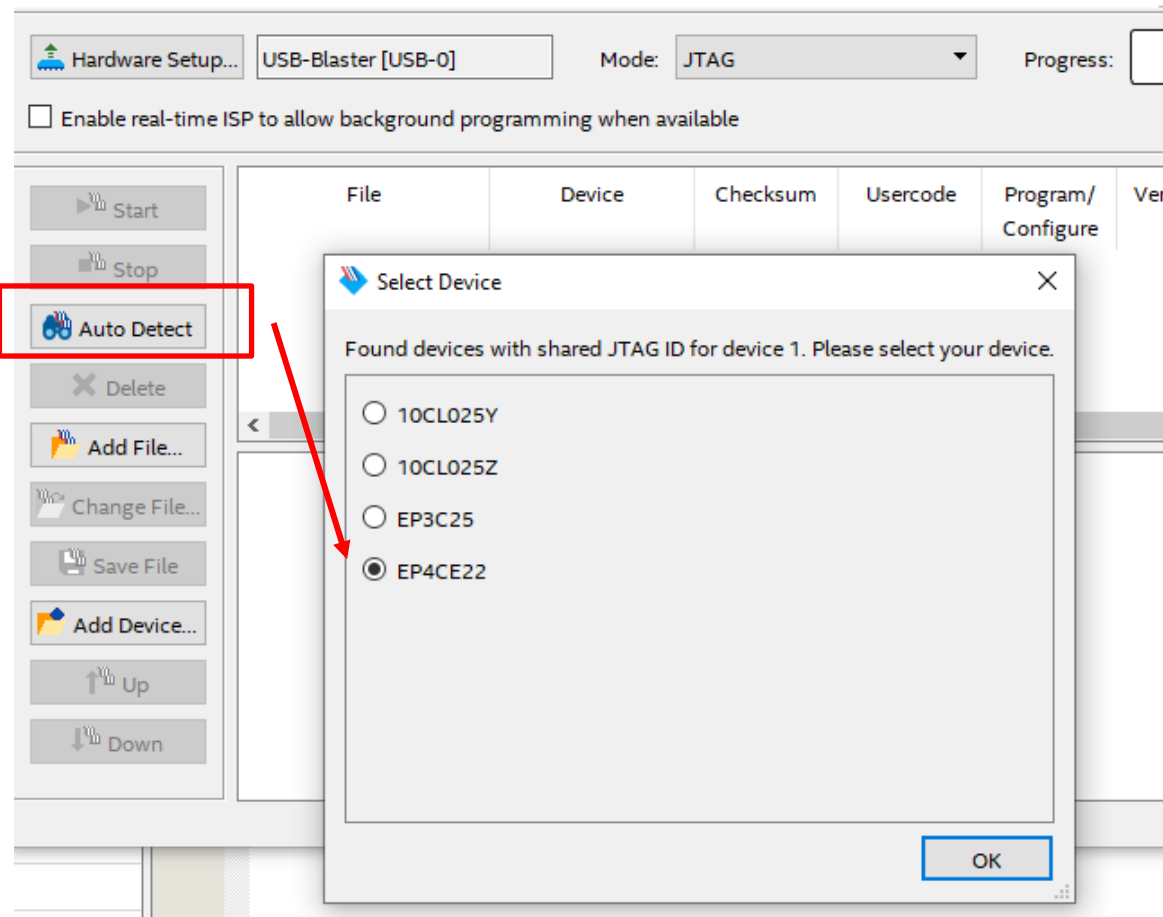
Selecione a pasta drivers no caminho abaixo e instale:

```
C:\intelFPGA_lite\18.1\quartus\drivers\usb-blaster
```

VHDL – Gravando o código na FPGA

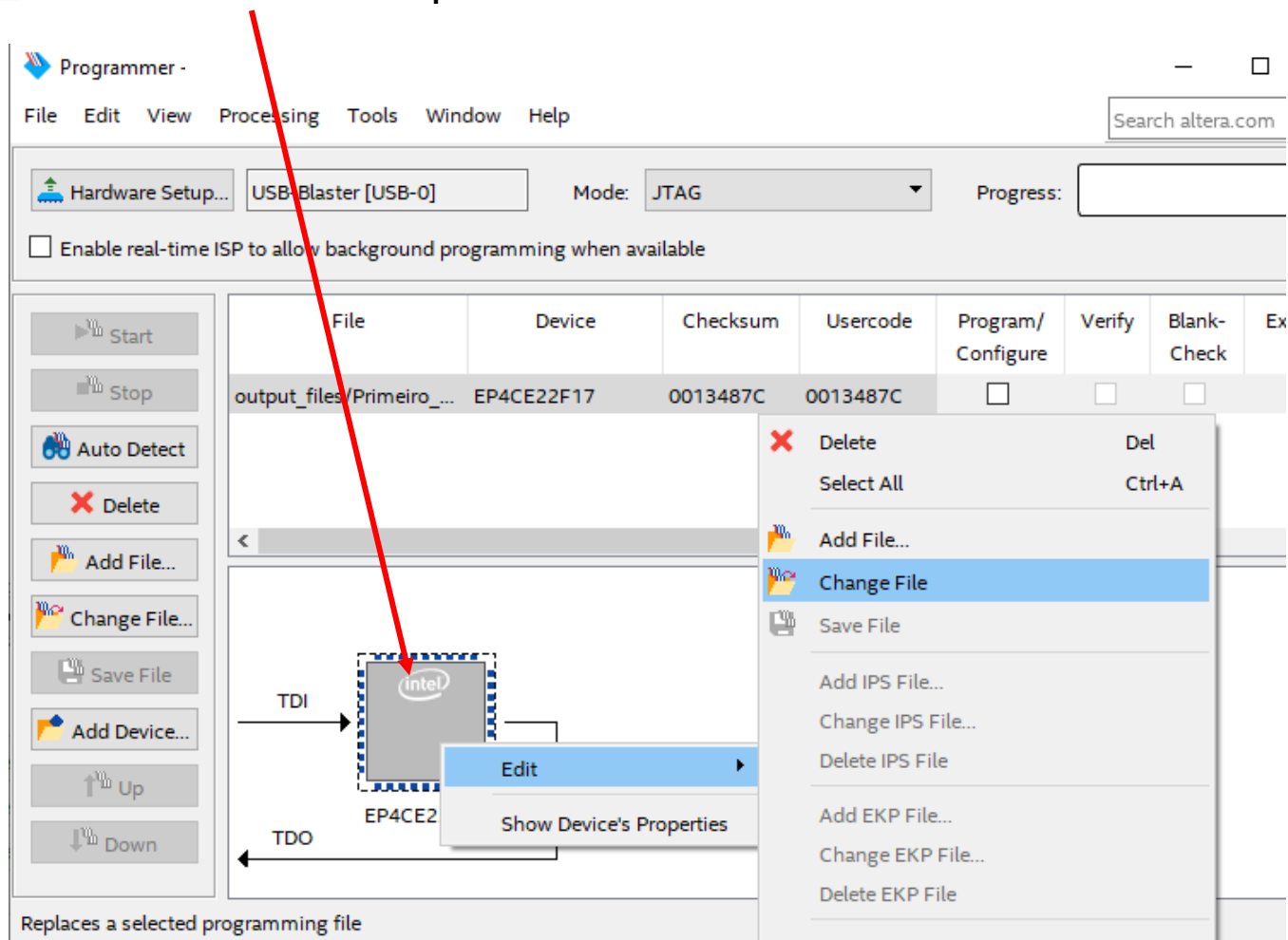
Caso a FPGA não apareça automaticamente:

Clique em “auto detect” – Selecione EP4CE22



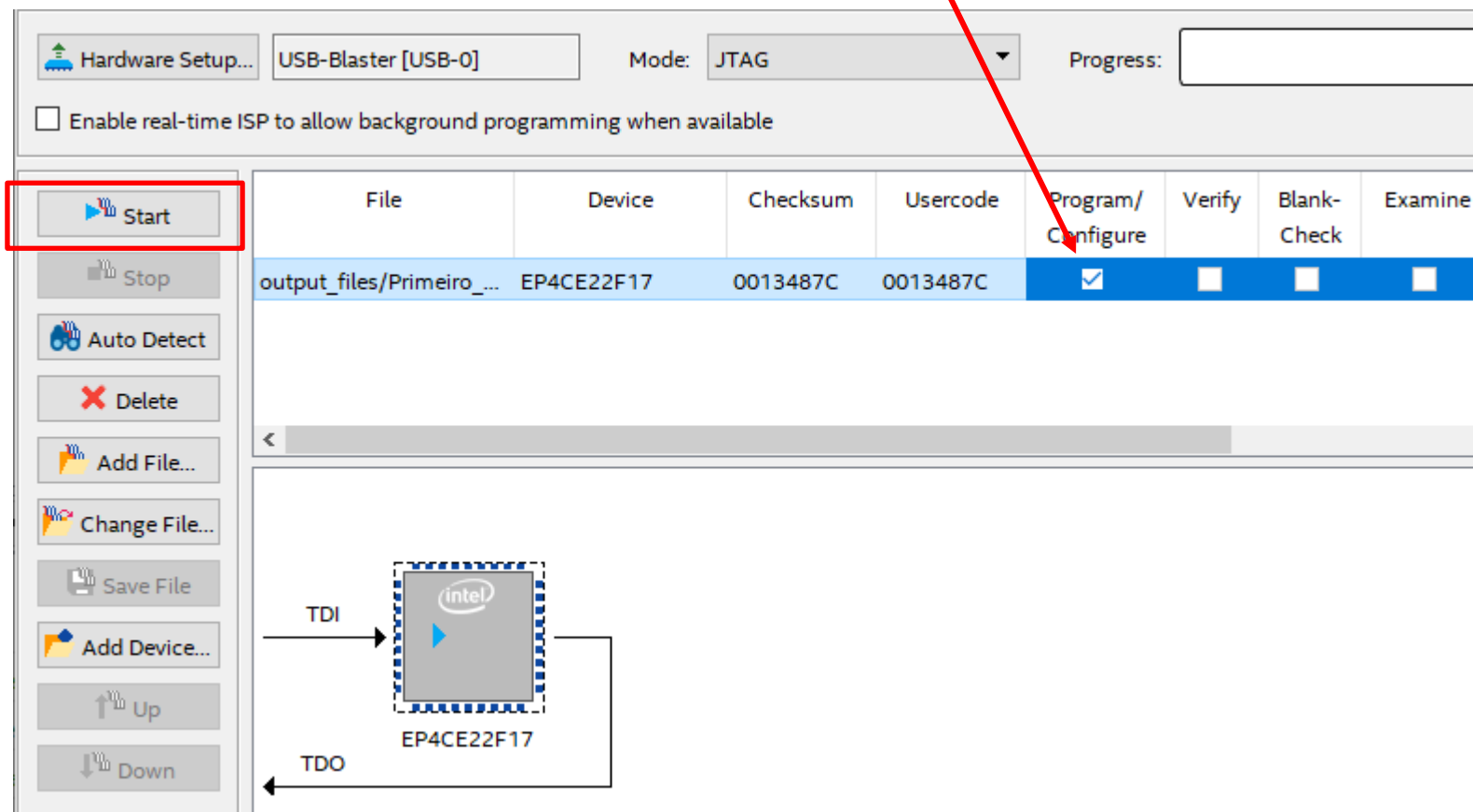
VHDL – Gravando o código na FPGA

Clique com o botão direito em cima do “chip” > Edit > Add file ou change file caso queira mudar o arquivo. Vá até a pasta “output_files” e escolha o arquivo .sof



VHDL – Gravando o código na FPGA

Após selecionar “program/Configure”, o botão “Start” ficará habilitado para realizar a gravação. Verifique o funcionamento no kit.



VHDL – Gravando o código na FPGA

- Refaça o procedimento para o exercício abaixo:

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

ENTITY Primeiro_projeto IS
    PORT (IN1 : IN std_logic;
          IN2 : IN std_logic;
          IN3 : IN std_logic;
          SAIDA1 : OUT std_logic;
          SAIDA2 : OUT std_logic);
END Primeiro_projeto;

ARCHITECTURE logica OF Primeiro_projeto IS
BEGIN
    SAIDA1<= (IN1 AND IN2) OR IN3;
    SAIDA2<= (IN1 XOR IN2) AND NOT(IN3);
END logica;
```

DE0_nano_user_manual

Pag. 14

Signal Name	FPGA Pin No.
LED[0]	PIN_A15
LED[1]	PIN_A13
LED[2]	PIN_B13
LED[3]	PIN_A11

Pag. 15

Signal Name	FPGA Pin No.
DIP Switch[0]	PIN_M1
DIP Switch[1]	PIN_T8
DIP Switch[2]	PIN_B9
DIP Switch[3]	PIN_M15

VHDL – Gravando o código na FPGA

- Utilize DIP Switch [0-2] como entradas e LED [0-1] como saídas.

O “?” é referente à compilação anterior, selecione e delete.

Top View - Wire Bond
Cyclone IV E - EP4CE22F17C6

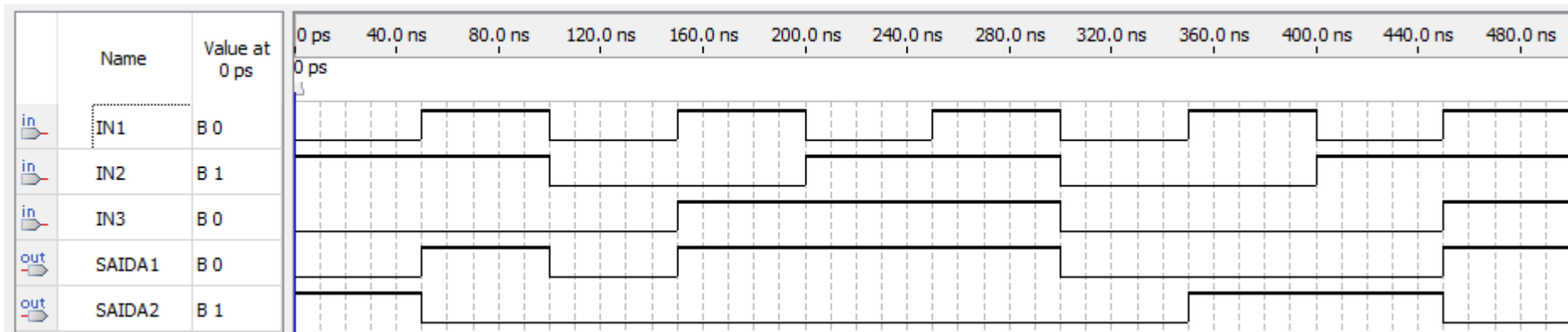
Node Name	Direction	Location	I/O Bank	VREF Group	Fitter Location
in IN1	Input	PIN_M1	2	B2_N0	PIN_M1
in IN2	Input	PIN_T8	3	B3_N0	PIN_T8
in IN3	Input				PIN_L3
out SAIDA1	Output				PIN_L1
out SAIDA2	Output				PIN_N2
? SAIDA	Unknown	PIN_A15	7	B7_N0	
<<new node>>					

VHDL – Gravando o código na FPGA

- Resultado:

Node Name	Direction	Location
in IN1	Input	PIN_M1
in IN2	Input	PIN_T8
in IN3	Input	PIN_B9
out SAIDA1	Output	PIN_A15
out SAIDA2	Output	PIN_B13
<<new node>>		

- Compile novamente, e faça a simulação.



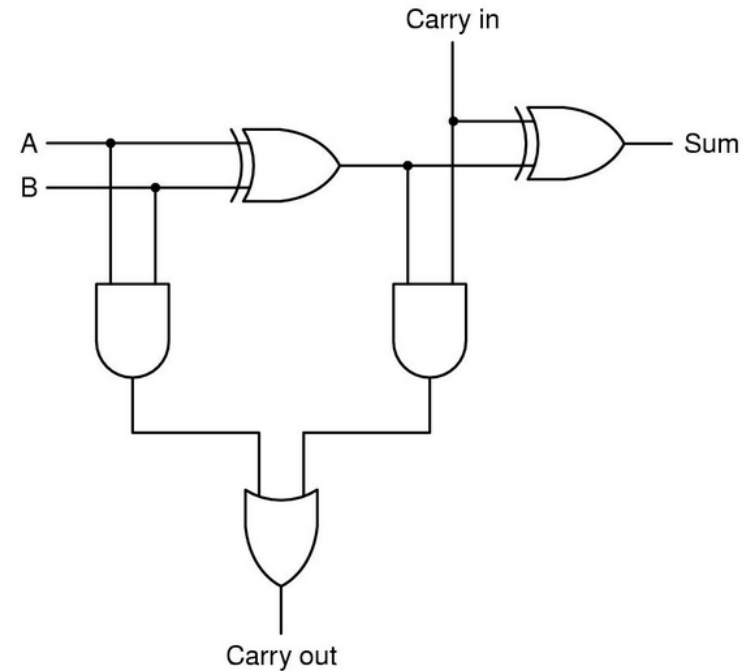
Exercício 1

- Desenvolvendo um somador completo: Crie um novo projeto.
- Escreva em VHDL o código equivalente circuito proposto.
- Utilizar como entrada: A - DIP Switch[0], B - DIP Switch [1] e Cin - DIP Switch [2], e como saída: Sum - LED[0] e Cout - LED[1].

Signal Name	FPGA Pin No.
LED[0]	PIN_A15
LED[1]	PIN_A13
LED[2]	PIN_B13
LED[3]	PIN_A11

Signal Name	FPGA Pin No.
DIP Switch[0]	PIN_M1
DIP Switch[1]	PIN_T8
DIP Switch[2]	PIN_B9
DIP Switch[3]	PIN_M15

A	B	Carry in	Sum	Carry out
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

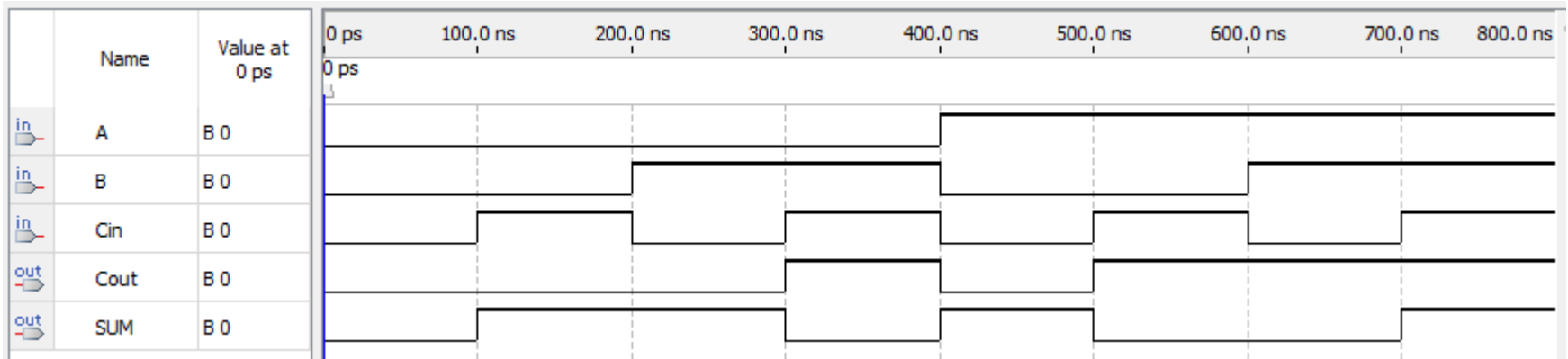


Exercício 1

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

-- ENTITY Somador IS
--   PORT (A, B, Cin : IN std_logic;
--         SUM, Cout : OUT std_logic);
-- END Somador;

-- ARCHITECTURE logica OF Somador IS
-- BEGIN
--   SUM<=(A XOR B) XOR Cin;
--   Cout<=(A AND B) OR ((A XOR B) AND Cin);
-- END logica;
```



Exercício 2

- Utilizando vetores: Crie um novo projeto.
- Faça um software que coloque na saída X o vetor de 2 bits da entrada A.
- Utilizar como entrada: DIP Switch[0] e DIP Switch [1], e como saída: LED[0] e LED[1].
 - Exemplo para 3 bits:
 - STD_LOGIC_VECTOR vetor de bits
 - STD_LOGIC_VECTOR(2 downto 0) – (D2 D1 D0)
 - STD_LOGIC_VECTOR(0 to 2) – (D0 D1 D2)

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

entity vetor is
    Port ( A      : in  STD_LOGIC_VECTOR (1 downto 0);
          X      : out STD_LOGIC_VECTOR (1 downto 0));
end vetor;

architecture funcao of vetor is
begin
    X <= A;
end funcao;
```

Exercício 3

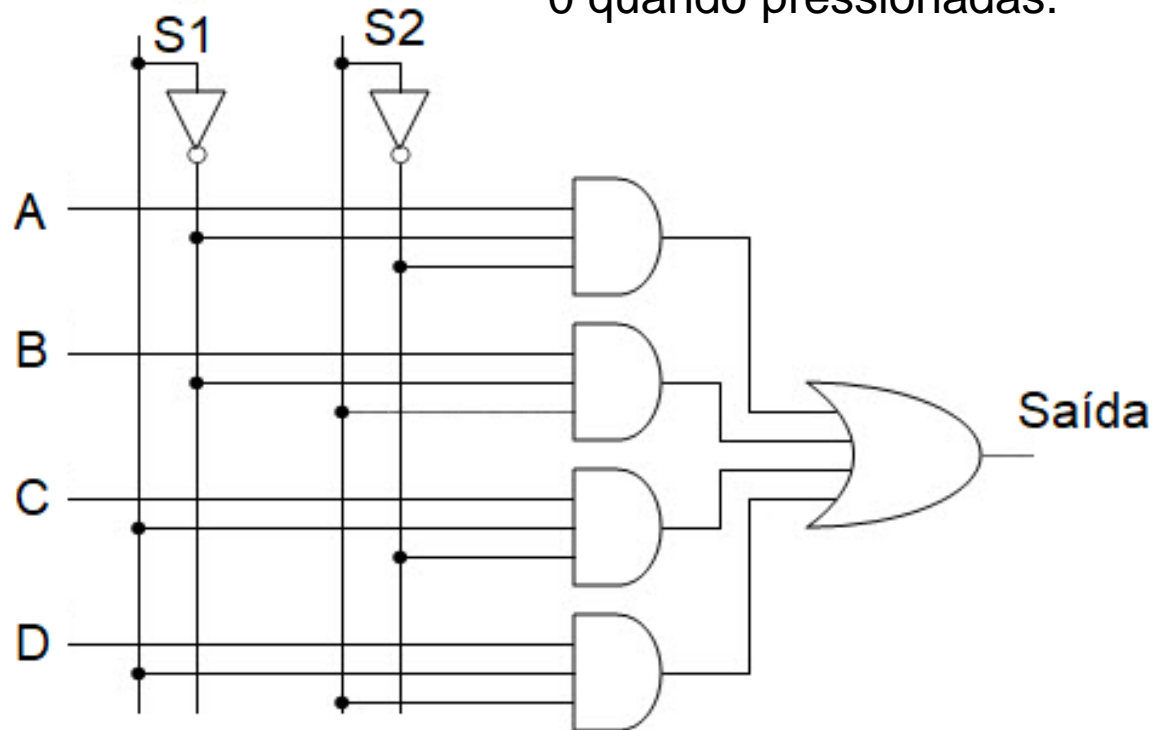
- Refaça o exercício da aula anterior (mux 4:1) usando um vetor na entrada e na seleção: Utilizar como entradas: DIP Switch[0-3], como seleção: KEY [0-1] e como saída: LED[0].

Signal Name	FPGA Pin No.
LED[0]	PIN_A15
LED[1]	PIN_A13
LED[2]	PIN_B13
LED[3]	PIN_A11

Signal Name	FPGA Pin No.
DIP Switch[0]	PIN_M1
DIP Switch[1]	PIN_T8
DIP Switch[2]	PIN_B9
DIP Switch[3]	PIN_M15

Signal Name	FPGA Pin No.
KEY[0]	PIN_J15
KEY[1]	PIN_E1

As KEY's geram nível lógico 0 quando pressionadas.



OBS: Para ler um bit do vetor, utilize "PORT(x)", sendo PORT o nome utilizado para o port, e x a posição do bit no vetor

Exercício 3

- Resposta:

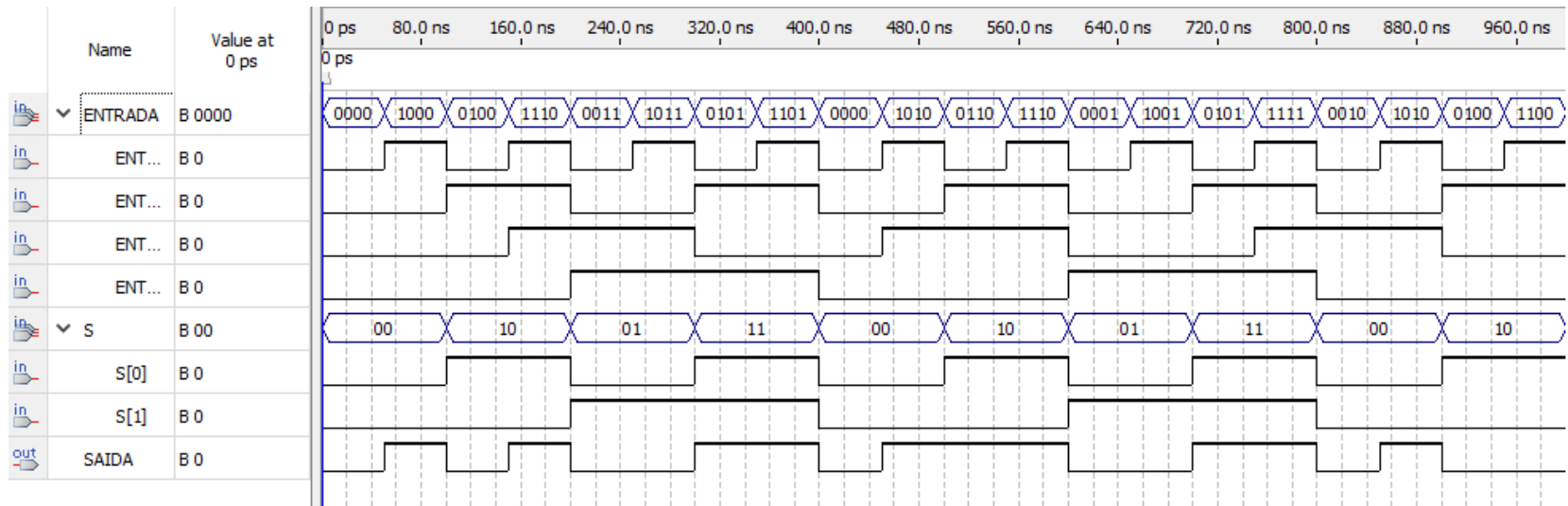
```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY mux4_1_vetor IS
    PORT (ENTRADA : IN std_logic_vector(0 to 3);
          S : IN std_logic_vector(0 to 1);
          SAIDA : OUT std_logic);
END mux4_1_vetor;

ARCHITECTURE logica OF mux4_1_vetor IS
BEGIN
    SAIDA <= ((ENTRADA(0) and not(S(0)) and not(S(1)))
    OR (ENTRADA(1) and not(S(0)) and S(1))
    OR (ENTRADA(2) and S(0) and not(S(1)))
    OR (ENTRADA(3) and S(0) and S(1)));
END logica;
```

Exercício 3

■ Resposta:



- Próxima aula: Gravação no Kit DE1-SoC e Implementação de circuitos com diagrama de blocos.