

**Curso Superior de Sistemas de
Telecomunicações – Unidade São
José**

**Disciplina: Síntese de Sistemas de
Telecomunicações – 7º Fase**

Bases tecnológicas

- Dispositivos Lógicos Programáveis.
- Introdução à Tecnologia FPGA.
- Introdução aos ambientes de software EDA (Electronic Design Automation).
- Introdução à Linguagem VHDL.
- Aritmética computacional.
- Introdução aos Kits de desenvolvimento.
- Síntese de circuitos baseada em dispositivos lógicos programáveis.

- *Material de apoio fornecido pela Xilinx (Xilinx University Program – XUP). www.xilinx.com*
- *Digital Signal Processing with Field Programmable Gate Arrays; 2.ed; **Uwe Meyer-Baese**; Springer, 2006*
- *Digital Electronics and Design with VHDL; **Volnei A. Pedroni**; Elsevier Science, 2007*
- *Circuit Design with VHDL; **Volnei A. Pedroni**; MIT Press, 2004*
- *Projetando Controladores Digitais com FPGA; **César da Costa**; Novatec, 2006*

Introdução a FPGA

O material a ser apresentado foi elaborado com base nas bibliografias citadas anteriormente.

Aplicações de DSP (Digital Signal Processing)

- ***Processadores DSP***: Nos últimos 20 anos, a maior parte das aplicações DSP foram realizadas por processadores DSP (Texas Instruments, Motorola, Analog Devices...).
- ***ASICs*** (Application Specific Integrated Circuits): São muito utilizados em aplicações específicas de DSP.
- ***FPGA*** (Field Programmable Gate Array): Tecnologia recente para aplicações de DSP de alta velocidade (Xilinx, ALTERA, Atmel...).

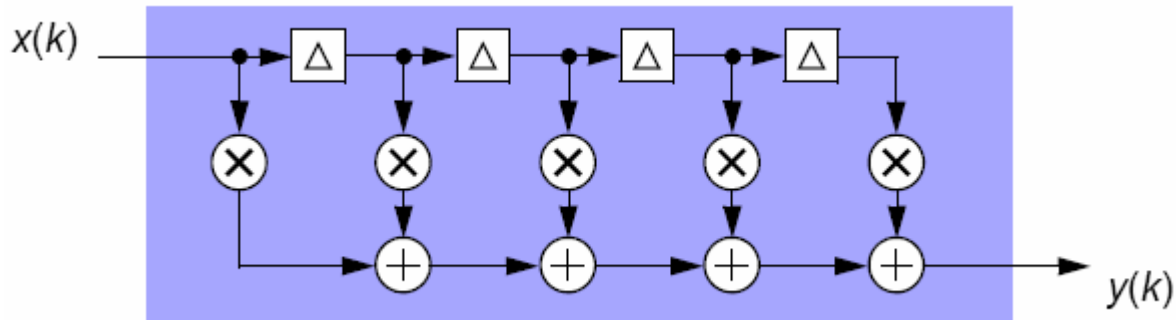
Aplicações de DSP

- A maioria dos algoritmos utilizados nas aplicações de DSP envolvem as operações de multiplicação e soma, conhecidas como operações MAC (Multiplies and accumulates).
- Operações de divisão e raiz quadrada são raras em algoritmos de DSP.
- Normalmente a complexidade de um algoritmo de DSP pode ser medida em termos do número de operações MAC utilizadas.

Exemplo – Operações MAC

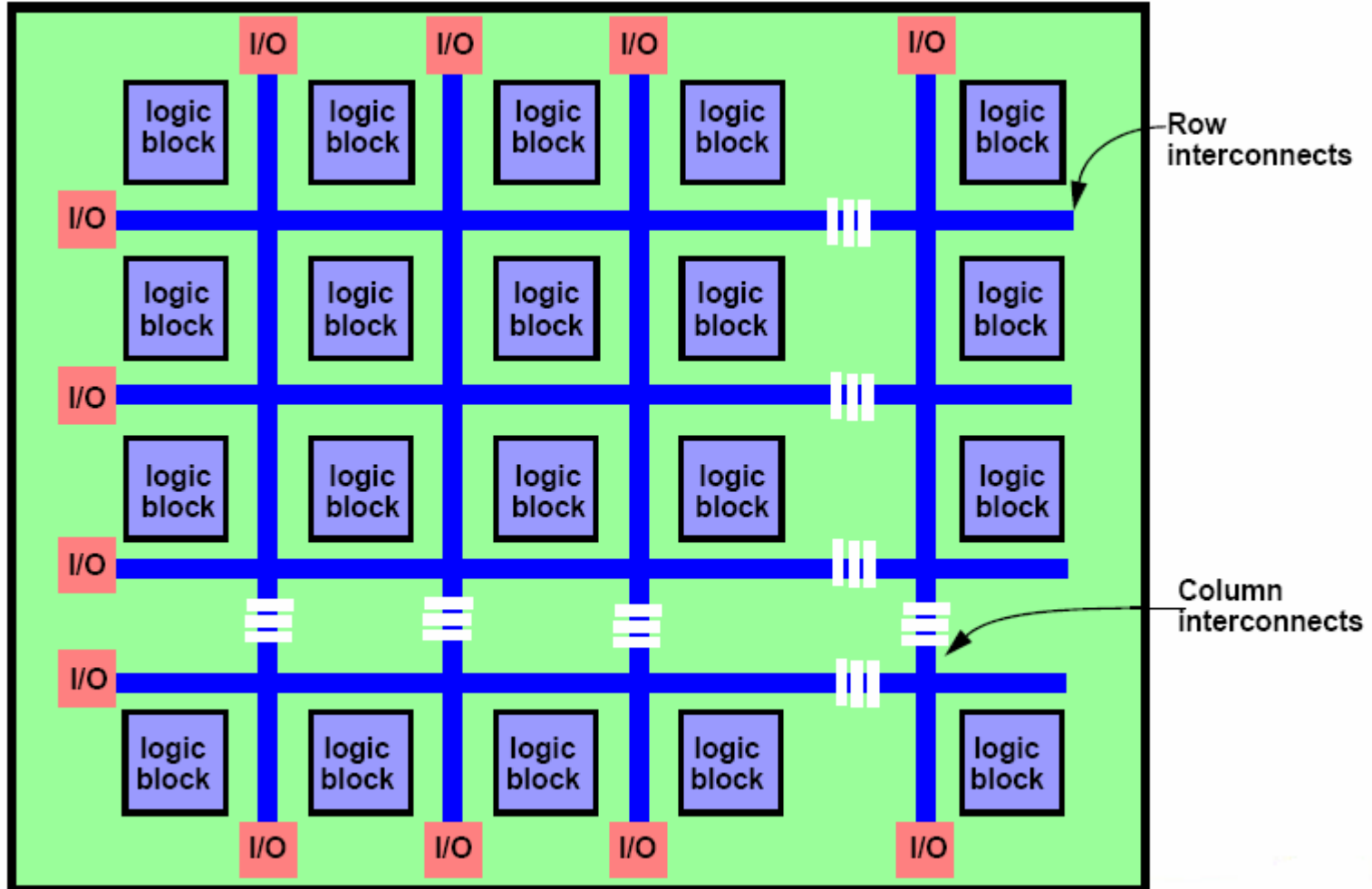
Filtro FIR com 5 coeficientes

$$y(k) = \sum_{n=0}^4 w_n x(k-n)$$



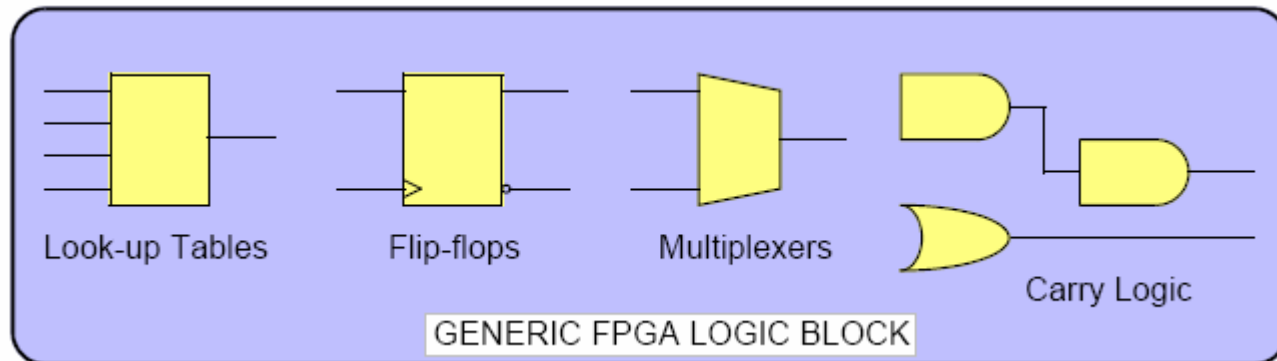
- A FPGA é um conjunto de unidades lógicas idênticas (blocos lógicos) e configuráveis contidas em um único circuito integrado.
- As unidade lógicas são conectadas por uma matriz de trilhas condutoras e por chaves de interconexão.
- As interfaces I/O do FPGA são feitas pelos blocos de I/O do componente.

Estrutura simplificada do FPGA



Bloco Lógico de um FPGA

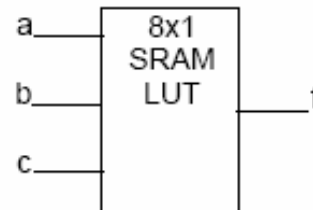
- Os blocos lógicos contêm componentes lógicos que implementam funções lógicas e aritméticas. Para a fabricante Xilinx, os blocos lógicos são chamados de *Slices*.



LUT – Look-Up Table

- São blocos muito utilizados para implementar funções lógicas.
- A forma geral de uma LUT é uma SRAM, que armazena tabelas verdade para funções lógicas de n-entradas.
- Desta forma, as linhas de endereçamento da SRAM funcionam como entrada e a saída fornece o valor da função lógica.

a	b	c	f
0	0	1	1
0	0	0	0
0	1	1	1
0	1	0	0
1	0	1	1
1	0	0	0
1	1	1	1
1	1	0	0



Fluxo de Projeto em FPGA

- Especificação e entrada do projeto.
- Síntese e mapeamento da tecnologia.
- Posicionamento e roteamento.
- Verificação e teste.
- Programação do FPGA.

Especificação e Entrada do Projeto

- Diagramas lógico através de editores gráficos.
- Linguagem de descrição de hardware (Hardware Description Language) através de editores de texto.
- Modelagem de sistemas através de software específico (Ex. System Generator da Xilinx).

Síntese e mapeamento da tecnologia (netlist)

- O processo de síntese otimiza as equações booleanas (otimização lógica independente da tecnologia) geradas pelo projeto, permitindo a redução da área a ser ocupada no FPGA. Os atrasos entre os sinais internos também são reduzidos.
- No mapeamento, o projeto otimizado é adequado à tecnologia empregada no componente a ser utilizado.

Posicionamento e roteamento (Place and Route)

- O posicionamento é a atribuição de componentes disponíveis aos componentes lógicos do projeto.
- Na etapa de roteamento, as conexões entre os componentes são garantidas visando sempre maximizar a velocidade das conexões críticas.

Verificação e teste

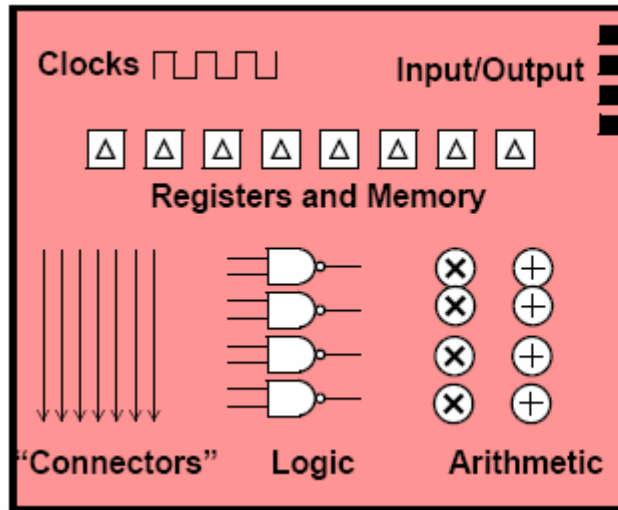
- Nesta etapa são utilizadas algumas ferramentas para simular o funcionamento do projeto.
 - ModelSim
 - Synopsys

- As simulações podem ser feitas para verificar o comportamento do sistema e também para verificar restrições de temporização.

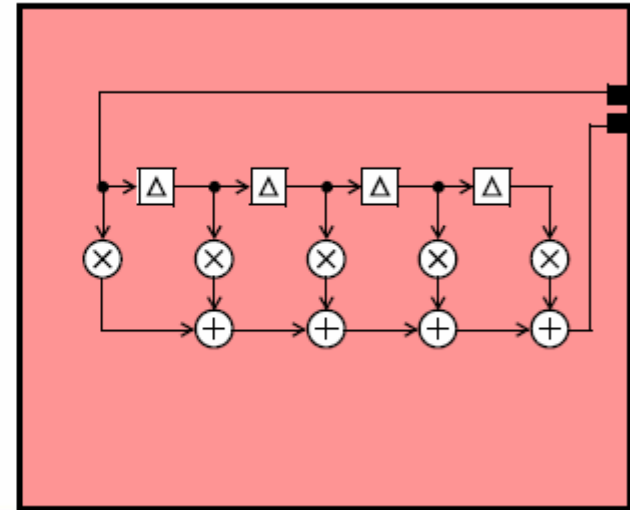
Programação do FPGA

- Nesta etapa, é gerado um arquivo binário para configuração do dispositivo.
- O download pode ser feito através de um cabo USB.

FPGA: uma caixa de blocos de DSP



Design
Verify
Place and Route

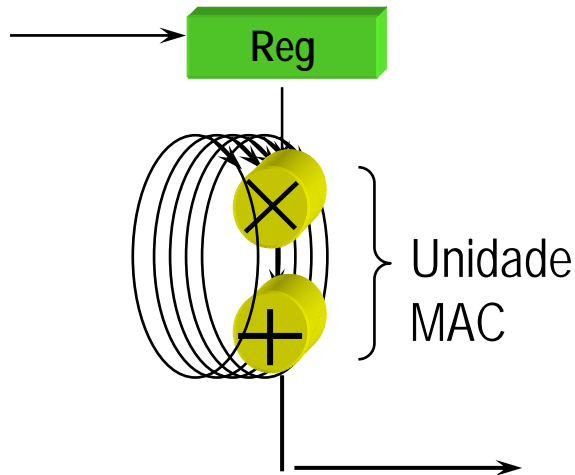


FPGA vs. Processadores DSP

- O FPGA tem a flexibilidade para variar o número de bits de acordo com a aplicação.
- No FPGA o processamento é inerentemente paralelo, para algoritmos seqüenciais um FPGA não é a melhor solução.
- Se uma determinada aplicação pode ser realiza em um processador DSP, provavelmente não é vantagem utilizar um FPGA.

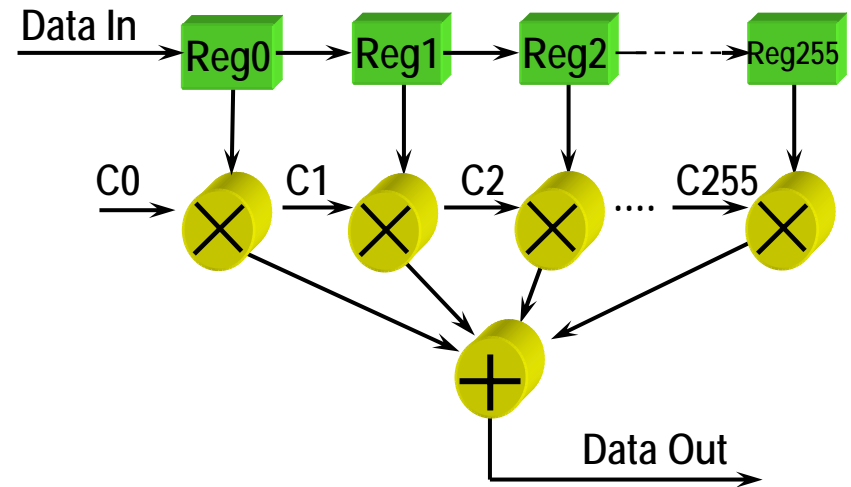
Filtro FIR com 256 coeficientes

DSP Convencional



256 Loops necessários para processar amostras

FPGA



Todas as 256 operações MAC são realizadas em 1 ciclo de relógio (clock)

Ferramentas utilizadas na disciplina

- ISE (Xilinx).
- Quartus (ALTERA).
- Matlab/System Generator (Xilinx).

VHDL

- VHDL (**V**ery **H**igh **S**pecific **I**ntegrated **C**ircuit **H**ardware **D**escription **L**anguage) é uma linguagem de descrição de hardware.
- A linguagem VHDL é um padrão especificado pelo IEEE 1076 e IEEE 1164.
- Por ser um padrão o código desenvolvido independe do fabricante.

VHDL

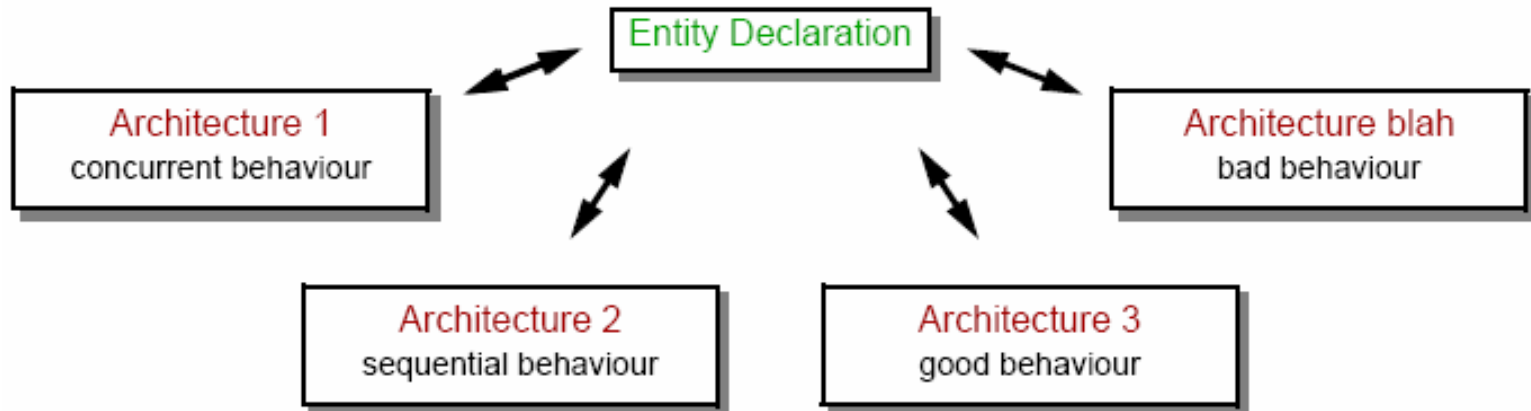
- As duas aplicações principais são em dispositivos lógicos programáveis (CPLD, FPGA) e em ASICs.
- A linguagem (código) VHDL é inerentemente paralela.
- Apenas comandos colocados dentro de *processos*, *funções* ou *procedimentos* são executados sequencialmente.

Unidades básicas do VHDL

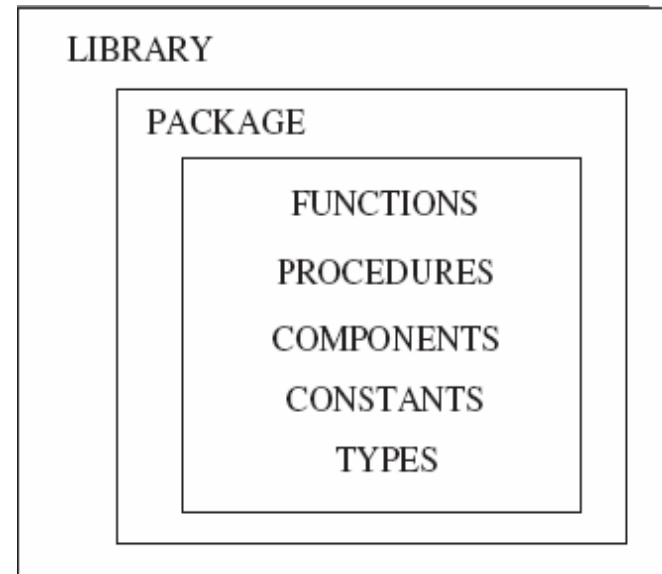
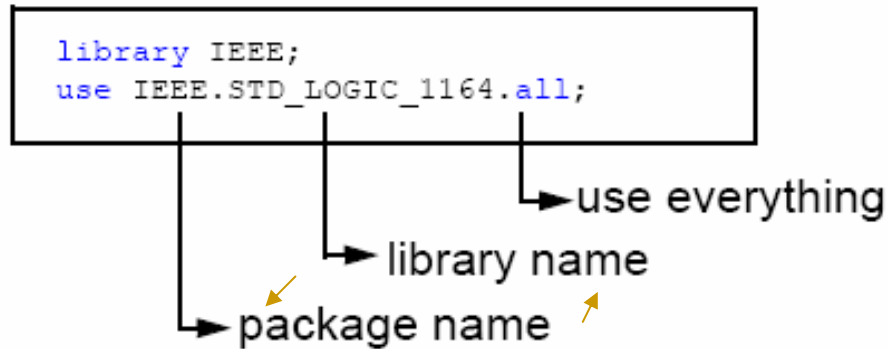
- *Declaração de bibliotecas:* Lista de todas as bibliotecas que podem ser usadas no projeto.
- *Entidade:* Especifica os pinos de I/O do circuito.
- *Arquitetura:* Contém o código VHDL dizendo como o circuito deve funcionar.

VHDL

- Todo o projeto em VHDL deve conter pelo menos um par entidade-arquitetura.
- Um projeto pode ter mais de uma arquitetura, cada uma descrevendo um comportamento para o sistema



Declaração de bibliotecas



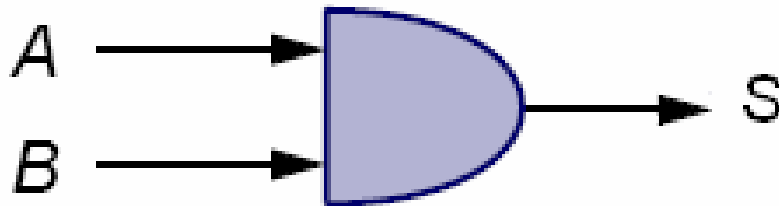
As bibliotecas 'work' e 'std' são incluídas por definição, não necessitando serem citadas no código.

Declaração de Entidade

```
ENTITY entity_name IS
  PORT (
    port_name : signal_mode signal_type;
    port_name : signal_mode signal_type;
    ...);
END entity_name;
```

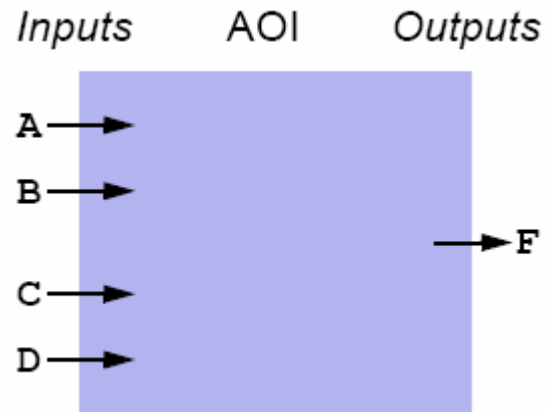
- O modo do sinal pode ser *in* (entrada), *out* (saída), *inout* (bidirecional) e *buffer* (saída e realimentação externa).
- O tipo pode ser *bit*, *std_logic*, *integer*, etc.
- O nome da entidade deve ser o mesmo do arquivo *.vhd*.

Exemplo 1



```
ENTITY PORTA_E IS
  PORT
  (
    A, B: IN BIT;
    S: OUT BIT
  );
END ENTITY PORTA_E;
```

Exemplo 2



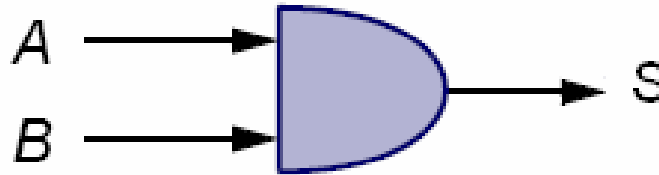
```
entity AOI is
    port (A, B, C, D : in  std_logic;
          F          : out std_logic);
end AOI;
```

Declaração da arquitetura

```
ARCHITECTURE architecture_name OF entity_name IS  
    [declarations]  
BEGIN  
    (code)  
END architecture_name;
```

- A declaração de sinais internos e constantes é feita entre a arquitetura e o início do código.
- O nome da entidade por ser qualquer nome, exceto as palavras reservadas do VHDL.

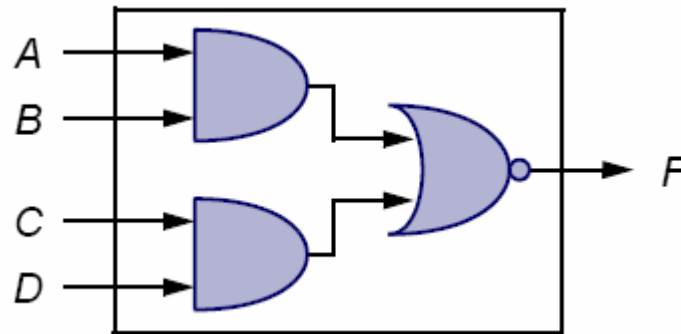
Exemplo 1



```
ARCHITECTURE MINHA_PORTA OF PORTA_E2 IS  
  
BEGIN  
  
    S <= A AND B;  
  
END ARCHITECTURE;
```

Para atribuir valores aos sinais utiliza-se '<=' e para variáveis utiliza-se ':=', neste exemplo o valor de 'A.B' está sendo atribuído ao sinal de saída 'S'.

Exemplo 2



```
architecture V1 of  $\overline{\text{AOI}}$  is
begin
    F <= not ((A and B) or (C and D));
end V1;
```

Exemplo: Porta E

```
ENTITY PORTA_E IS
  PORT
  (
    A, B: IN BIT;
    S: OUT BIT
  );
END ENTITY PORTA_E;
```

```
ARCHITECTURE MINHA_PORTA OF PORTA_E IS
```

```
BEGIN
```

```
    S <= A AND B;
```

```
END ARCHITECTURE;
```



Exemplo: Porta E_OU

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;

-- entity declaration
entity AOI is

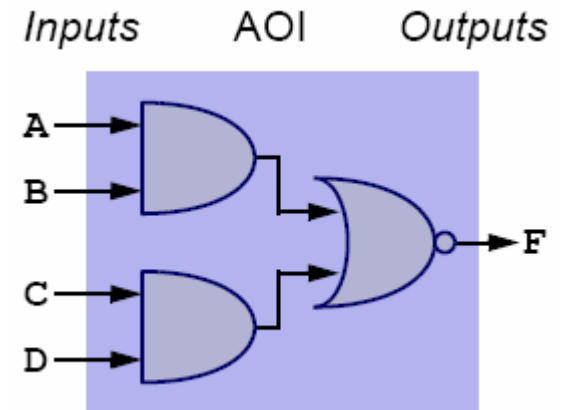
    port (A, B, C, D : in  std_logic;
          F           : out std_logic);

end AOI;

-- architecture body
architecture V1 of AOI is
begin

    F <= not ((A and B) or (C and D));

end V1;
```



Palavras Reservadas do VHDL

ABS	FILE	OF	THEN
ACCESS	FOR	ON	TO
AFTER	FUNCTION	OPEN	TRANSPORT
ALIAS		OR	TYPE
ALL	GENERATE	OTHERS	
AND	GENERIC	OUT	UNAFFECTED
ARCHITECTURE	GROUP		UNITS
ARRAY	GUARDED	PACKAGE	UNTIL
ASSERT		PORT	USE
ATTRIBUTE	IF	POSTPONED	
	IMPURE	PROCEDURE	VARIABLE
BEGIN	IN	PROCESS	
BLOCK	INERTIAL	PURE	WAIT
BODY	INOUT		WHEN
BUFFER	IS	RANGE	WHILE
BUS		RECORD	WITH
	LABEL	REGISTER	
CASE	LIBRARY	REJECT	XNOR
COMPONENT	LINKAGE	REM	XOR
CONFIGURATION	LITERAL	REPORT	
CONSTANT	LOOP	RETURN	
		ROL	
DISCONNECT	MAP	ROR	
DOWNTO	MOD		
		SELECT	
ELSE	NAND	SEVERITY	
ELSIF	NEW	SIGNAL	
END	NEXT	SHARED	
ENTITY	NOR	SLA	
EXIT	NOT	SLL	
	NULL	SRA	
		SRL	
		SUBTYPE	

Introdução ao Quartus II

➤ www.altera.com

Código Concorrente (paralelos) e Seqüencial

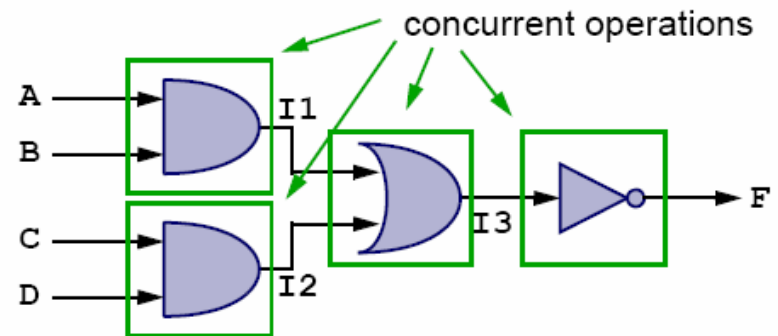
- Os códigos paralelos em geral implementam circuitos lógicos combinacionais, onde a saída do circuito depende apenas da entrada atual
- Os códigos seqüenciais implementam circuitos lógicos seqüenciais, onde a saída do circuito depende de entradas anteriores

Código Concorrente

Não importa a ordem em que os sinais são escritos, eles podem ser executados ao mesmo tempo

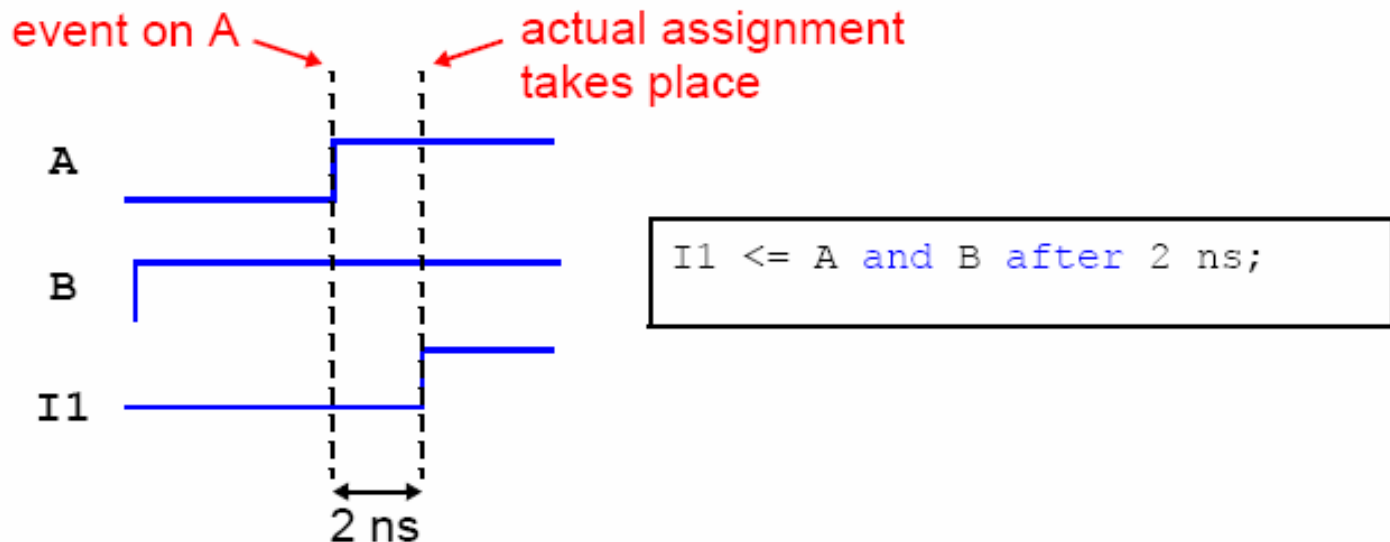
```
-- entity declaration
entity AOI is
    port(A, B, C, D : in  std_logic;
         F           : out std_logic);
end AOI;

-- architecture body
architecture V2 of AOI is
    signal I1, I2, I3 : std_logic;
begin
    -- concurrent assignments
    I1 <= A and B;
    I2 <= C and D;
    I3 <= I1 or I2;
    F  <= not I3;
end V2;
```



Código Concorrente

- A atribuição de um valor a um sinal concorrente é disparada por um **evento** em um determinado sinal.
- O evento neste caso é a **mudança de valor** de um sinal.
- No caso abaixo, um atraso de 2ns na atribuição do valor para I1 é considerado

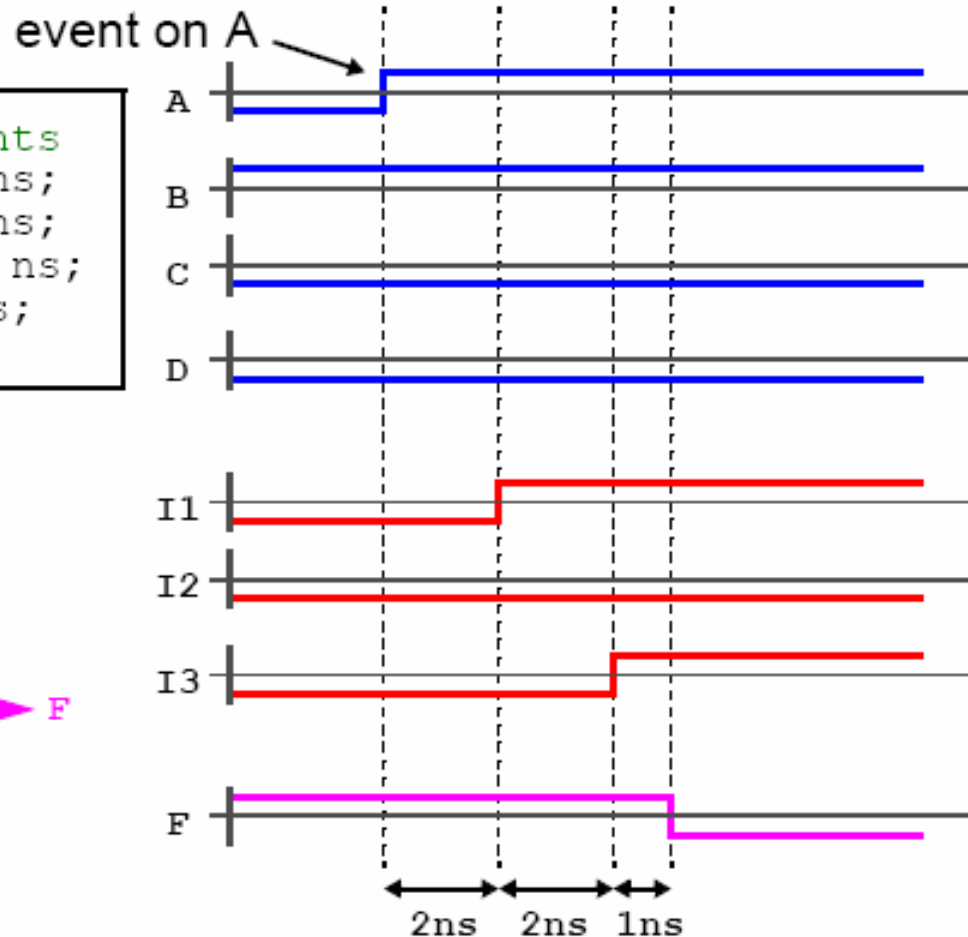
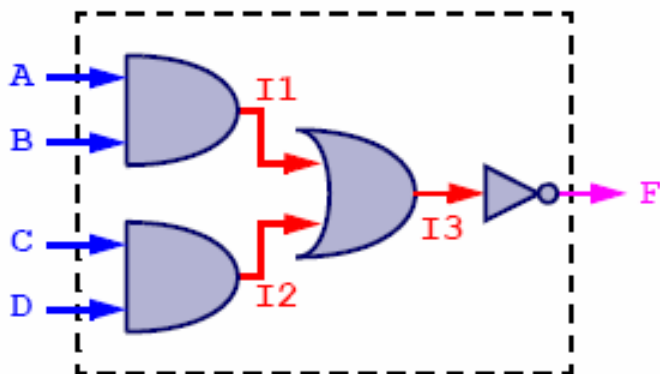


Código Concorrente

O diagrama abaixo mostra a resposta de um circuito ao evento A

```

-- concurrent assignments
I1 <= A and B after 2 ns;
I2 <= C and D after 2 ns;
I3 <= I1 or I2 after 2 ns;
F <= not I3 after 1 ns;
  
```



Exemplo 1: Multiplexador

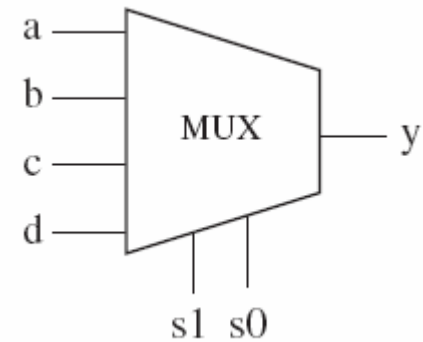
```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

-----

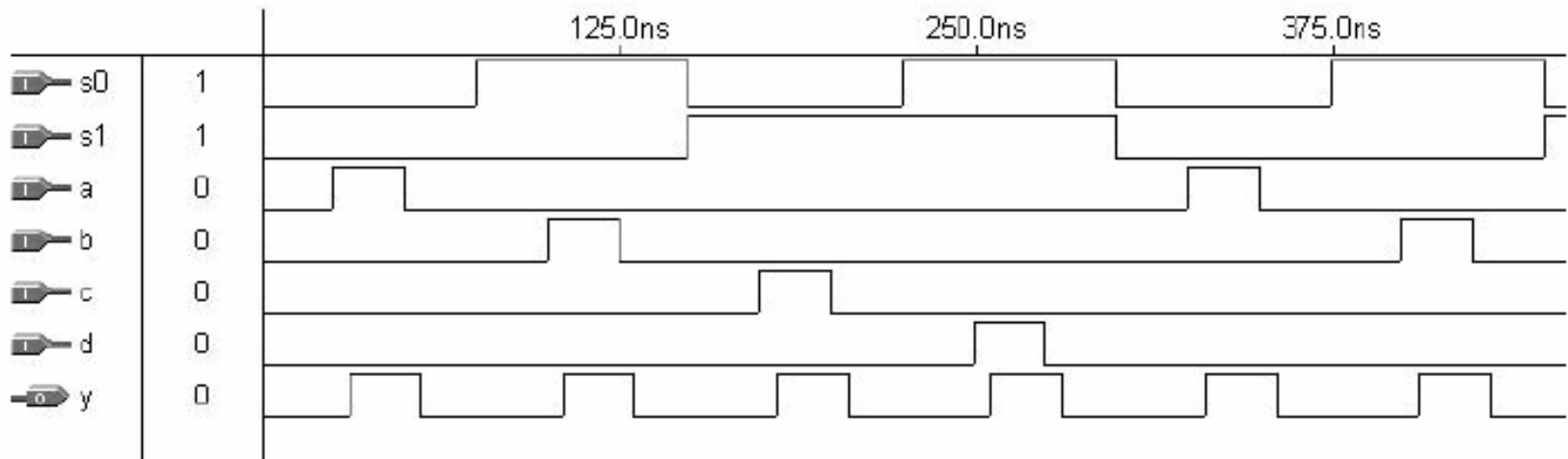
ENTITY mux IS
    PORT ( a, b, c, d, s0, s1: IN STD_LOGIC;
          y: OUT STD_LOGIC);
END mux;

-----

ARCHITECTURE pure_logic OF mux IS
BEGIN
    y <= (a AND NOT s1 AND NOT s0) OR
         (b AND NOT s1 AND s0) OR
         (c AND s1 AND NOT s0) OR
         (d AND s1 AND s0);
END pure_logic;
```

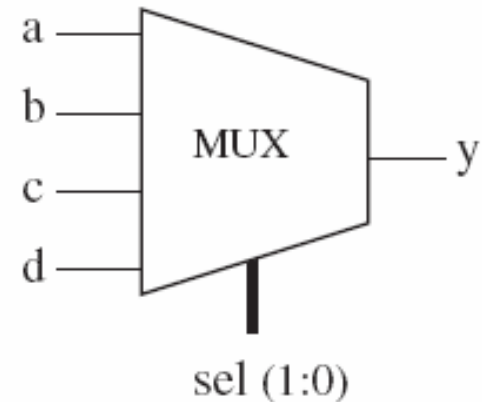


Resultado da simulação



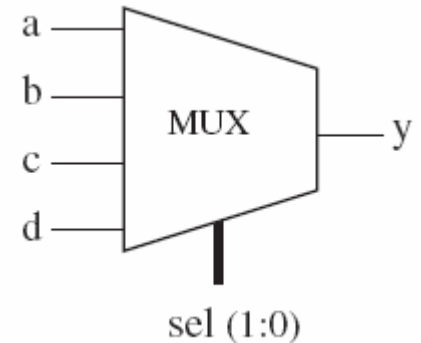
Exemplo 2: Multiplexador (when/else)

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
-----
ENTITY mux IS
    PORT ( a, b, c, d: IN STD_LOGIC;
          sel: IN STD_LOGIC_VECTOR (1 DOWNTO 0);
          y: OUT STD_LOGIC);
END mux;
-----
ARCHITECTURE mux1 OF mux IS
BEGIN
    y <=  a WHEN sel="00" ELSE
         b WHEN sel="01" ELSE
         c WHEN sel="10" ELSE
         d;
END mux1;
```



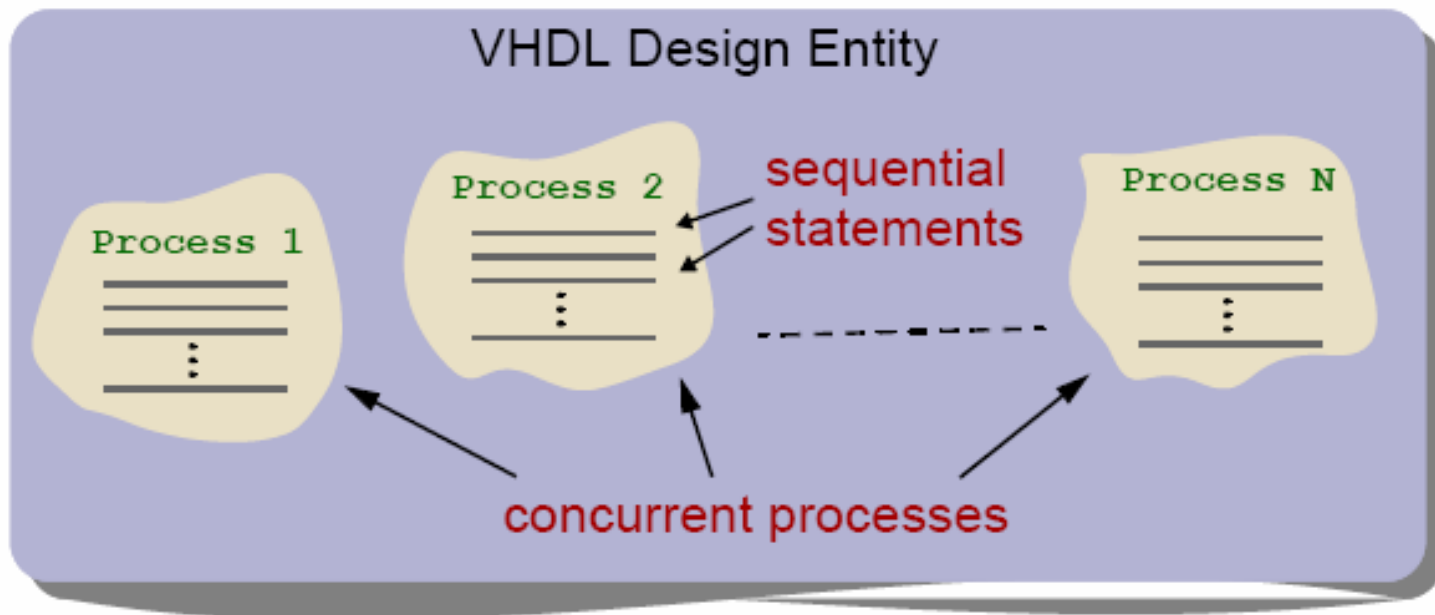
Exemplo 3: Multiplexador (with/select/when)

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
-----
ENTITY mux IS
    PORT ( a, b, c, d: IN STD_LOGIC;
          sel: IN STD_LOGIC_VECTOR (1 DOWNTO 0);
          y: OUT STD_LOGIC);
END mux;
-----
ARCHITECTURE mux2 OF mux IS
BEGIN
    WITH sel SELECT
        y <=  a WHEN "00",      -- notice ", " instead of ";"
             b WHEN "01",
             c WHEN "10",
             d WHEN OTHERS;    -- cannot be "d WHEN "11" "
END mux2;
```



Código Seqüencial - Processos

- Um processo é uma seção seqüencial de um código VHDL.
- As instruções dentro de um processo são executadas em ordem.
- Processos dentro de uma arquitetura são executados em paralelo.

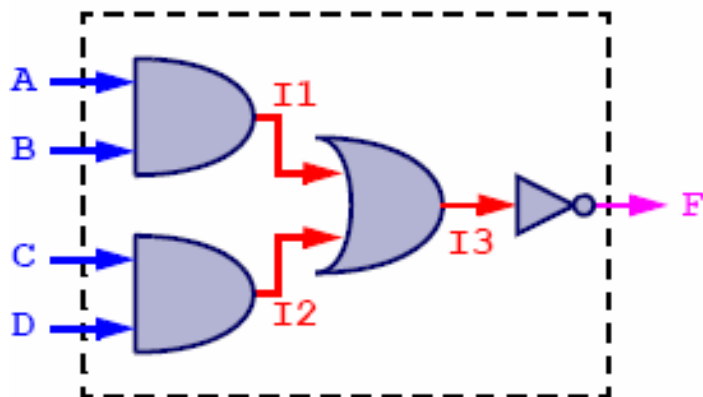


Processos com Lista de Sensibilidade

- A lista de sensibilidade é opcional.
- Processos com lista de sensibilidade apenas são executados quando um evento ocorre.
- O evento é uma mudança do sinal contido na lista de sensibilidade.

```
process (A, B)      -- process with a sensitivity list
begin
    F <= A and B;  -- these statements are executed one ...
    G <= A or B;   -- ... after the other
end process;
```

Vários Processos em uma Arquitetura



```
-- architecture body
architecture V3 of AOI is
    signal I1, I2, I3 : std_logic;
begin

    -- process to model the AND gates
    AND_GATES : process (A, B, C, D)
    begin
        I1 <= A and B after 2 ns;
        I2 <= C and D after 2 ns;
    end process;

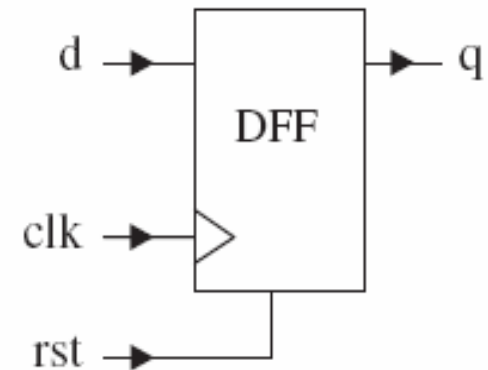
    -- process to model the OR gate
    OR_GATE : process (I1, I2)
    begin
        I3 <= I1 or I2 after 2 ns;
    end process;

    -- process to model the inverter
    NOT_GATE : process (I3)
    begin
        F <= not I3 after 1 ns;
    end process;

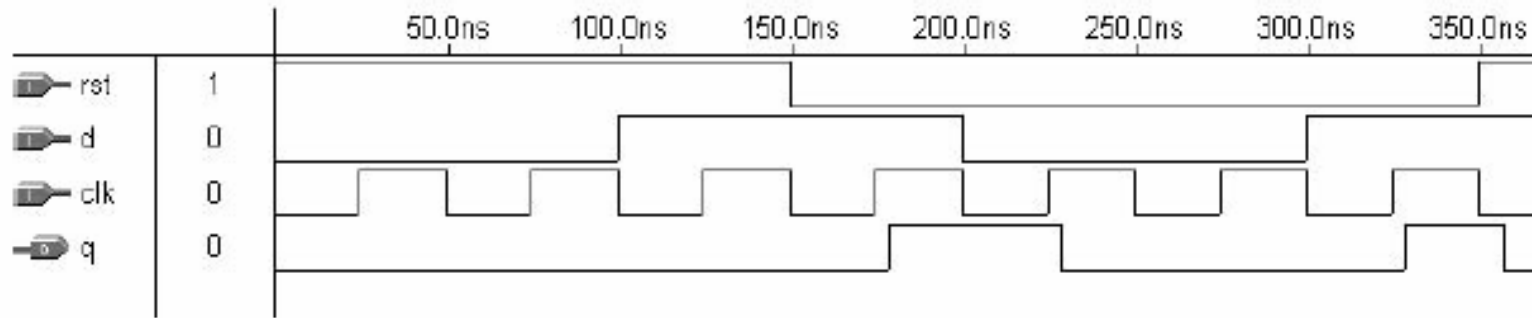
end V3;
```


Exemplo 1: Flip-flop tipo D com reset assíncrono.

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
-----
ENTITY dff IS
    PORT ( d, clk, rst: IN STD_LOGIC;
          q: OUT STD_LOGIC);
END dff;
-----
ARCHITECTURE behavior OF dff IS
BEGIN
    PROCESS (rst, clk)
    BEGIN
        IF (rst='1') THEN
            q <= '0';
        ELSIF (clk'EVENT AND clk='1') THEN
            q <= d;
        END IF;
    END PROCESS;
END behavior;
```

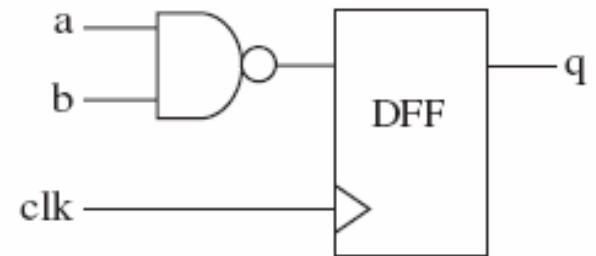


Resultado da simulação

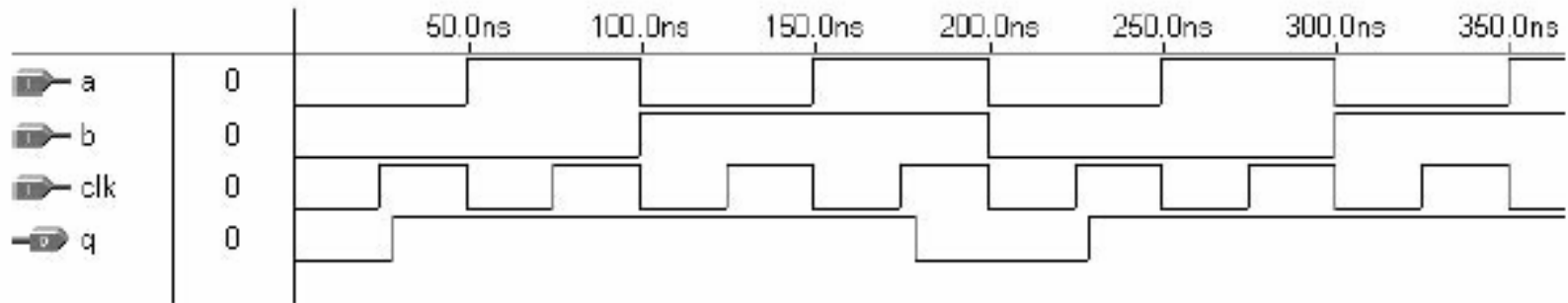


Exemplo 2: Flip-flop D com porta nand

```
ENTITY example IS
    PORT ( a, b, clk: IN BIT;
          q: OUT BIT);
END example;
-----
ARCHITECTURE example OF example IS
    SIGNAL temp : BIT;
BEGIN
    temp <= a NAND b;
    PROCESS (clk)
    BEGIN
        IF (clk'EVENT AND clk='1') THEN q<=temp;
        END IF;
    END PROCESS;
END example;
```



Resultado da simulação



Sinais, Variáveis e Constantes em VHDL

- Sinais: São utilizados para representar fios e barramentos em um circuito
- Variáveis: Similares às variáveis utilizadas em programas de computadores
- Constantes: Valores que não se alteram e que são rotulados para facilitar a programação.

- No caso da variável e do sinal, o valor inicial é opcional.

```
constant NUMBER_OF_BITS: integer := 8;
```

```
variable current_bit: integer := 0;
```

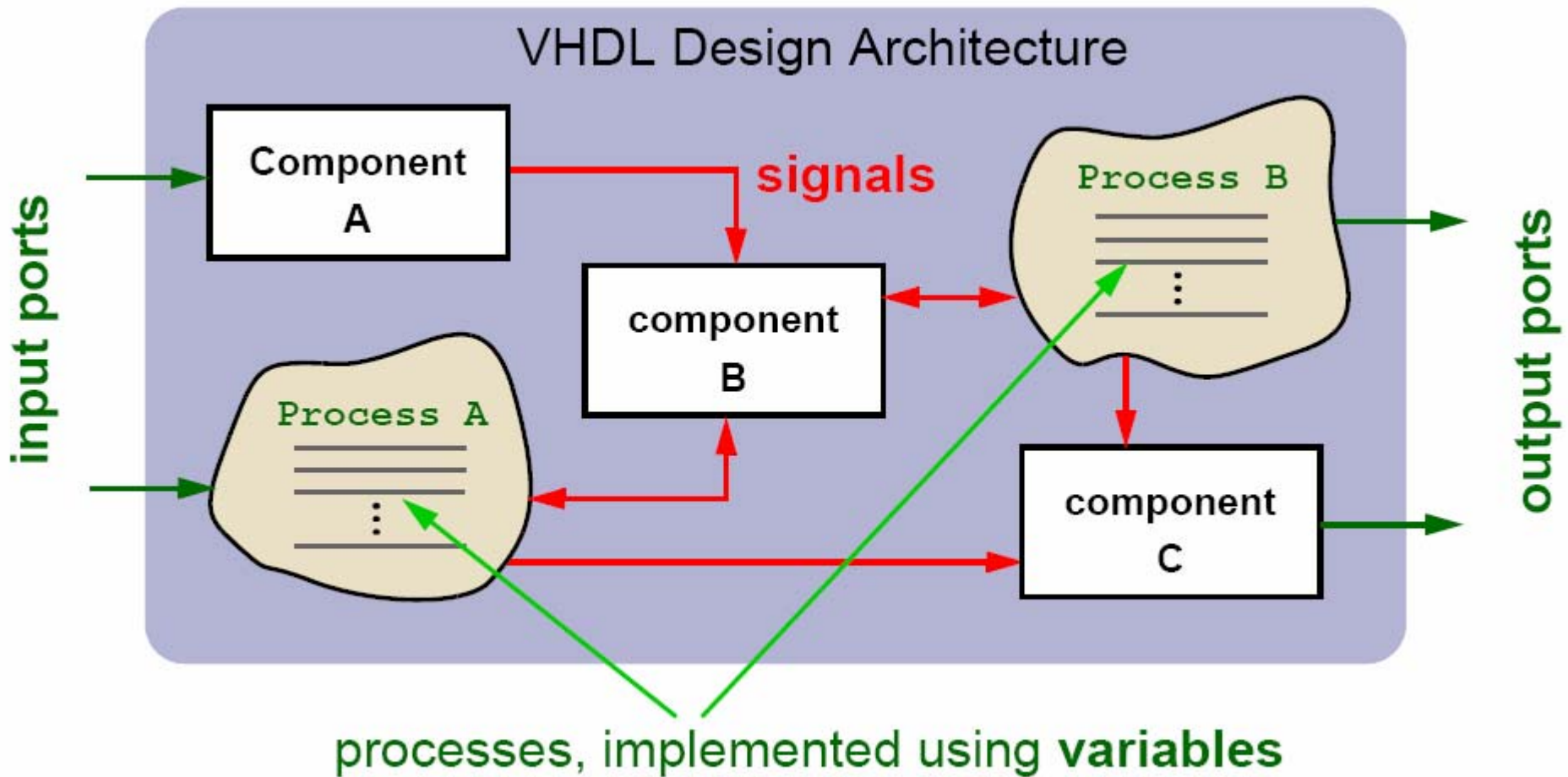
```
signal carry_out: std_logic := '0';
```

Sinais e Variáveis

- Sinais:
 - Podem ser utilizados para a comunicação entre processos.
 - Podem aparecer em listas de sensibilidade de um processo
 - Podem ter atrasos.

- Variáveis
 - Podem ser utilizadas apenas dentro do processo que foram declaradas
 - Não podem aparecer em listas de sensibilidade
 - Não podem ter atrasos

Sinais e Variáveis



Exemplos de Tipos de Sinais: BIT

- BIT (and BIT_VECTOR): 2-level logic ('0', '1').

Examples:

```
SIGNAL x: BIT;
```

```
-- x is declared as a one-digit signal of type BIT.
```

```
SIGNAL y: BIT_VECTOR (3 DOWNTO 0);
```

```
-- y is a 4-bit vector, with the leftmost bit being the MSB.
```

```
SIGNAL w: BIT_VECTOR (0 TO 7);
```

```
-- w is an 8-bit vector, with the rightmost bit being the MSB.
```

```
x <= '1';
```

```
-- x is a single-bit signal (as specified above), whose value is  
-- '1'. Notice that single quotes ( ' ') are used for a single bit.
```

```
y <= "0111";
```

```
-- y is a 4-bit signal (as specified above), whose value is "0111"  
-- (MSB='0'). Notice that double quotes ( " ") are used for  
-- vectors.
```

```
w <= "01110001";
```

```
-- w is an 8-bit signal, whose value is "01110001" (MSB='1').
```

Exemplos de Tipos de Sinais: STD_LOGIC

- STD_LOGIC (and STD_LOGIC_VECTOR): 8-valued logic system introduced in the IEEE 1164 standard.

'X' Forcing Unknown

'0' Forcing Low

'1' Forcing High

'Z' High impedance

'W' Weak unknown

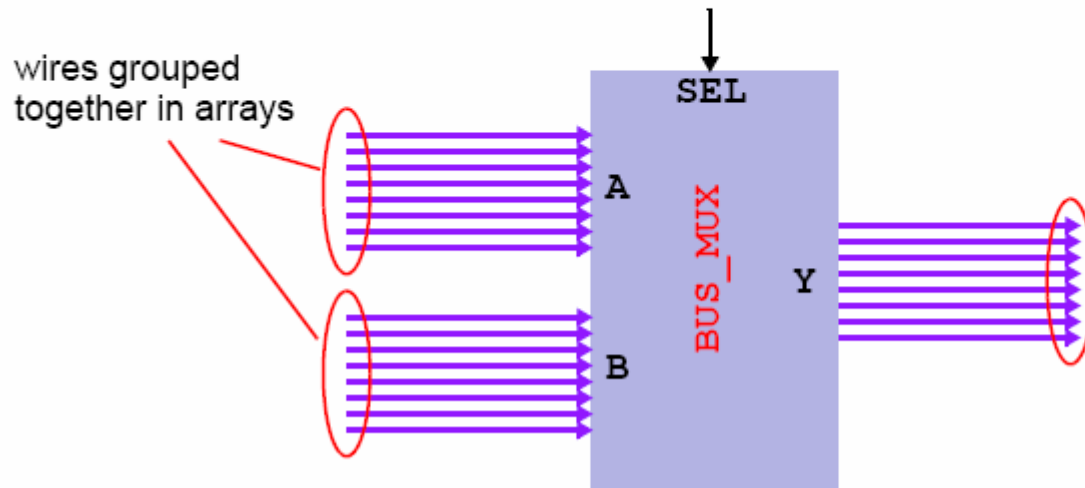
'L' Weak low

'H' Weak high

'-' Don't care

Exemplo

```
entity BUS_MUX is  
    port (A, B : in  std_logic_vector(7 downto 0);  
          SEL  : in  std_logic;  
          Y   : out std_logic_vector(7 downto 0));  
end entity BUS_MUX;
```



Outros Tipos de Dados

- Boolean: Verdadeiro e falso.
- Integer: inteiro de 32 bits [-2.147.483.647 até +2.147.483.647].
- Natural: Inteiros não negativos
- Signed e unsigned: Definidos no pacote *std_logic_arith*. São semelhantes ao STD_LOGIC_VECTOR mas aceitam operações aritméticas.

Exemplos

```
SIGNAL a: BIT;
SIGNAL b: BIT_VECTOR(7 DOWNTO 0);
SIGNAL c: STD_LOGIC;
SIGNAL d: STD_LOGIC_VECTOR(7 DOWNTO 0);
SIGNAL e: INTEGER RANGE 0 TO 255;
...
a <= b(5);      -- legal (same scalar type: BIT)
b(0) <= a;      -- legal (same scalar type: BIT)
c <= d(5);      -- legal (same scalar type: STD_LOGIC)
d(0) <= c;      -- legal (same scalar type: STD_LOGIC)
a <= c;         -- illegal (type mismatch: BIT x STD_LOGIC)
b <= d;         -- illegal (type mismatch: BIT_VECTOR x
                -- STD_LOGIC_VECTOR)
e <= b;         -- illegal (type mismatch: INTEGER x BIT_VECTOR)
e <= d;         -- illegal (type mismatch: INTEGER x
                -- STD_LOGIC_VECTOR)
```

Consultar bibliografia (tipos de dado, operadores, atributos)

- *Digital Electronics and Design with VHDL; Volnei A. Pedroni; Elsevier Science, 2007*
- *Circuit Design with VHDL; Volnei A. Pedroni; MIT Press, 2004*

Exemplo: Contador de um dígito

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
-----
ENTITY counter IS
    PORT (clk : IN STD_LOGIC;
          digit : OUT INTEGER RANGE 0 TO 9);
END counter;
-----
ARCHITECTURE counter OF counter IS
BEGIN
    PROCESS          -- no sensitivity list
        VARIABLE temp : INTEGER RANGE 0 TO 10;
    BEGIN
        WAIT UNTIL (clk'EVENT AND clk='1');
        temp := temp + 1;
        IF (temp=10) THEN temp := 0;
        END IF;
        digit <= temp;
    END PROCESS;
END counter;
```

Exercícios

- Exercício 1: Descreva um código que para cada dois bits de entrada coloque quatro bits na saída.
- Exercício 2: Faça um contador com dois dígitos que conte de 0 a 59.

Exercício 1: Solução 1

- **Solução (Processo implícito): Decodificador**

- LIBRARY IEEE;
- USE IEEE.STD_LOGIC_1164.ALL;

- ENTITY exe_cod_2_4 IS
- PORT
- (
• info: IN STD_LOGIC_VECTOR (1 DOWNTO 0);
- saida_cod: OUT STD_LOGIC_VECTOR (3 DOWNTO 0)
-);
- END ENTITY exe_cod_2_4;

- ARCHITECTURE regra OF exe_cod_2_4 IS

- BEGIN

- saida_cod <= "0001" WHEN info = "00" ELSE
- "0010" WHEN info = "01" ELSE
- "0100" WHEN info = "10" ELSE
- "1000" WHEN info = "11" ELSE
- "0000";

- END ARCHITECTURE;

Exercício 1: Solução 2

- **Solução (Processo Explícito): Decodificador**

- LIBRARY IEEE;
- USE IEEE.STD_LOGIC_1164.ALL;

- ENTITY exe_cod_2_4 IS
- PORT
- (
- info: IN STD_LOGIC_VECTOR (1 DOWNTO 0);
- saida_cod: OUT STD_LOGIC_VECTOR (3 DOWNTO 0)
-);
- END ENTITY exe_cod_2_4;

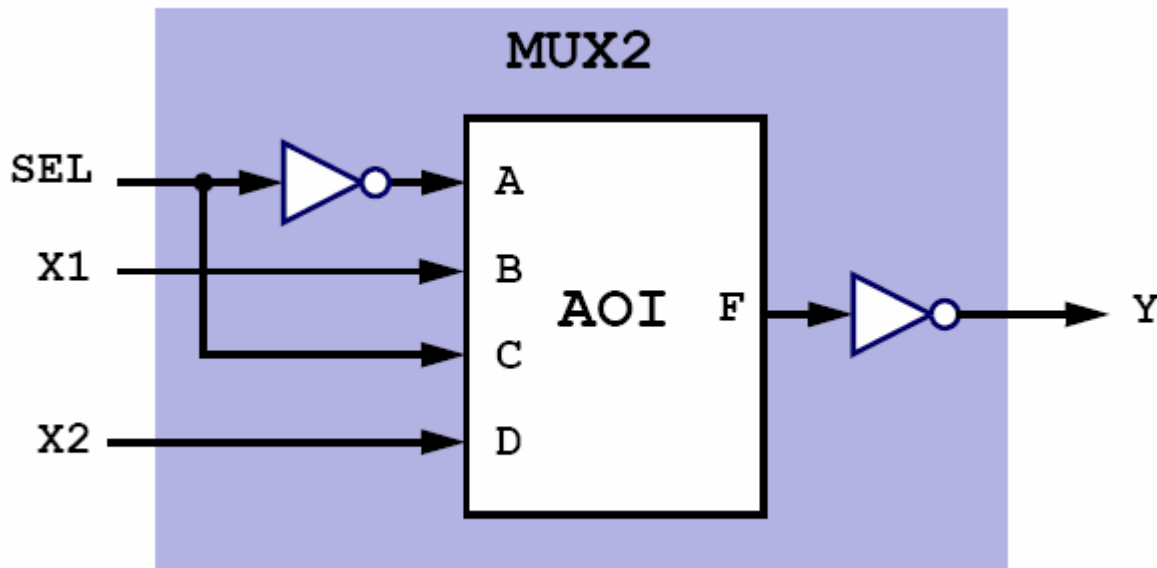
- ARCHITECTURE regra OF exe_cod_2_4 IS
- BEGIN
- PROCESS (info)
- BEGIN
- IF info = "00" THEN
- saida_cod <= "0001";
- ELSIF info = "01" THEN
- saida_cod <= "0010";
- ...
- ELSE
- saida_cod <= "0000";
- END IF
- END PROCESS
- END ARCHITECTURE

Componentes

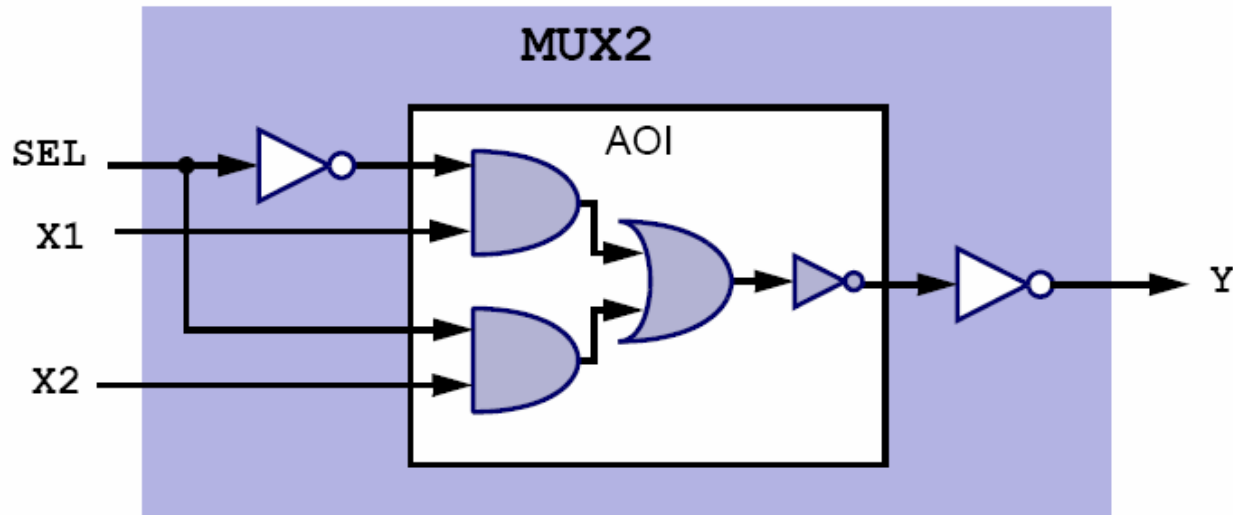
- Uma entidade já projetada que será utilizada para o projeto de uma entidade de nível maior é chamada de *componente*.
- Quando uma componente é utilizada, devemos especificar suas conectividades com outros componente, sinais e portas dentro da entidade.

Exemplo: Multiplexador

- Considerando o projeto MUX2, devemos utilizar a entidade de projeto AOI como uma componente de MUX2.



Entidade do MUX2



```
library IEEE;  
use IEEE.STD_LOGIC_1164.all;  
  
entity MUX2 is  
    port (SEL, X1, X2 : in  std_logic;  
          Y           : out std_logic);  
end entity MUX2;
```

Arquitetura do MUX2

- A palavra-chave *port map* especifica as conectividades do componente.

```
architecture ARCH1 of MUX2 is

    component AOI -- declare component
        port(A, B, C, D : in std_logic;
              F          : out std_logic);
    end component;

    signal SELB, T : std_logic;

begin

    SELB <= not SEL;

    AOI_inst : AOI -- instantiate component
    port map (A => SELB, B => X1, C => SEL, D => X2, F => T);

    Y <= not T;

end ARCH1;
```

Declaração de Vários Componentes

```
architecture GEN_ARCH of GEN_ENT is

    component C1
        -- declare C1 interface here
    end component;

    component C2
        -- declare C2 interface here
    end component;

    signal S1, S2, S3 .....

begin

    -- declare C1 instances
    C1_inst1 : C1 port map ( ..... );
    C1_inst2 : C1 port map ( ..... );
    :
    : : : :
    C1_instN : C1 port map ( ..... );

    -- declare C2 instances
    C2_inst1 : C2 port map ( ..... );
    C2_inst2 : C2 port map ( ..... );
    :
    : : : :
    C2_instM : C2 port map ( ..... );

end GEN_ARCH;
```

Mapeamento das portas

- Sintaxe por associação de nome.

```
[instance_label] : [design entity]  
  port map ( [lp1] => [gp1], [lp2] => [gp2], ..... );
```

- Cada associação é da forma: *porta local* => *porta global*
 - *Porta local*: Portas dos componentes.
 - *Porta global*: Portas/sinais que estão acessíveis dentro da arquitetura do projeto.

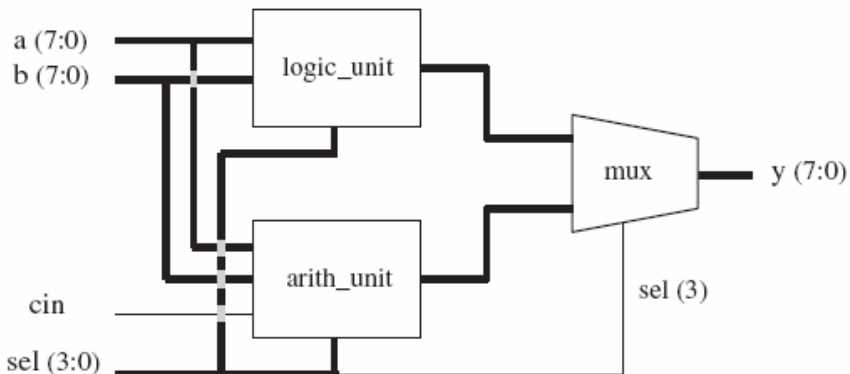
Mapeamento das Portas: Associação por Posição

- Neste tipo de associação a ordem do mapeamento que define as conexões.

```
AOI_inst : AOI          -- positional association  
  port map (SELB, X1, SEL, X2, T);
```

Exercício

- Exercício 1: Faça um projeto de uma unidade lógica aritmética (ULA) conforme a figura abaixo. Utilize o conceito de componentes. Gere um arquivo de formas de onda para simular e verificar o funcionamento do projeto.



sel	Operation	Function	Unit	
0000	$y \leq a$	Transfer a	Arithmetic	
0001	$y \leq a+1$	Increment a		
0010	$y \leq a-1$	Decrement a		
0011	$y \leq b$	Transfer b		
0100	$y \leq b+1$	Increment b		
0101	$y \leq b-1$	Decrement b		
0110	$y \leq a+b$	Add a and b		
0111	$y \leq a+b+cin$	Add a and b with carry		
1000	$y \leq \text{NOT } a$	Complement a		Logic
1001	$y \leq \text{NOT } b$	Complement b		
1010	$y \leq a \text{ AND } b$	AND		
1011	$y \leq a \text{ OR } b$	OR		
1100	$y \leq a \text{ NAND } b$	NAND		
1101	$y \leq a \text{ NOR } b$	NOR		
1110	$y \leq a \text{ XOR } b$	XOR		
1111	$y \leq a \text{ XNOR } b$	XNOR		

Solução: Unidade Aritmética

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

-----

ENTITY uni_arit IS
PORT ( a, b: IN STD_LOGIC_VECTOR (7 DOWNTO 0);
      sel: IN STD_LOGIC_VECTOR (2 DOWNTO 0);
      cin: IN STD_LOGIC;
      x: OUT STD_LOGIC_VECTOR (7 DOWNTO 0));
END uni_arit;

-----

ARCHITECTURE unidade OF uni_arit IS
SIGNAL arith, logic: STD_LOGIC_VECTOR (7 DOWNTO 0);
BEGIN
  WITH sel SELECT
    x <= a  WHEN "000",
        a+1 WHEN "001",
        a-1 WHEN "010",
        b   WHEN "011",
        b+1 WHEN "100",
        b-1 WHEN "101",
        a+b WHEN "110",
        a+b+cin WHEN OTHERS;
END unidade;
```

Solução: Unidade Lógica

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

-----

ENTITY uni_logica IS
PORT ( a, b: IN STD_LOGIC_VECTOR (7 DOWNTO 0);
      sel: IN STD_LOGIC_VECTOR (2 DOWNTO 0);
      x: OUT STD_LOGIC_VECTOR (7 DOWNTO 0));
END uni_logica;

-----

ARCHITECTURE unidade OF uni_logica IS
BEGIN
    WITH sel SELECT
        x <= NOT a    WHEN "000",
           NOT b     WHEN "001",
           a AND b   WHEN "010",
           a OR b    WHEN "011",
           a NAND b  WHEN "100",
           a NOR b   WHEN "101",
           a XOR b   WHEN "110",
           NOT (a XOR b) WHEN OTHERS;
END unidade;
```

Solução: Mux

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

-----

ENTITY mux1 IS
PORT ( a, b: IN STD_LOGIC_VECTOR (7 DOWNTO 0);
      sel: IN STD_LOGIC;
      x: OUT STD_LOGIC_VECTOR (7 DOWNTO 0));
END mux1;

-----

ARCHITECTURE funcionamento OF mux1 IS
BEGIN
    WITH sel SELECT
        x <= a WHEN '0',
        b WHEN OTHERS;
END funcionamento;
```

Solução: Código Principal - ULA

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

-----

ENTITY ula IS
PORT ( a, b: IN STD_LOGIC_VECTOR(7 DOWNTO 0);
      cin: IN STD_LOGIC;
      sel: IN STD_LOGIC_VECTOR(3 DOWNTO 0);
      y: OUT STD_LOGIC_VECTOR(7 DOWNTO 0));
END ula;

-----

ARCHITECTURE funcionamento OF ula IS

-----

COMPONENT uni_arit IS
PORT ( a, b: IN STD_LOGIC_VECTOR(7 DOWNTO 0);
      cin: IN STD_LOGIC;
      sel: IN STD_LOGIC_VECTOR(2 DOWNTO 0);
      x: OUT STD_LOGIC_VECTOR(7 DOWNTO 0));
END COMPONENT;

-----

COMPONENT uni_logica IS
PORT ( a, b: IN STD_LOGIC_VECTOR(7 DOWNTO 0);
      sel: IN STD_LOGIC_VECTOR(2 DOWNTO 0);
      x: OUT STD_LOGIC_VECTOR(7 DOWNTO 0));
END COMPONENT;

-----

COMPONENT mux1 IS
PORT ( a, b: IN STD_LOGIC_VECTOR(7 DOWNTO 0);
      sel: IN STD_LOGIC;
      x: OUT STD_LOGIC_VECTOR(7 DOWNTO 0));
END COMPONENT;

-----

SIGNAL x1, x2: STD_LOGIC_VECTOR(7 DOWNTO 0);

-----

BEGIN
    U1: uni_arit PORT MAP (a, b, cin, sel(2 DOWNTO 0), x1);
    U2: uni_logica PORT MAP (a, b, sel(2 DOWNTO 0), x2);
    U3: mux1 PORT MAP (x1, x2, sel(3), y);
END funcionamento;
```

Funções e Procedimentos

- São pedaços de códigos VHDL seqüenciais que podem ser utilizados ou compartilhados por diferentes projetos.
- Podem ser declarados nos pacotes ou no programa principal.

➤ Corpo da função:

```
FUNCTION function_name [<parameter list>] RETURN data_type IS  
[declarations]  
BEGIN  
(sequential statements)  
END function_name;
```

➤ Exemplo:

```
FUNCTION f1 (a, b: INTEGER; SIGNAL c: STD_LOGIC_VECTOR)  
RETURN BOOLEAN IS  
BEGIN  
(sequential statements)  
END f1;
```


- Função localizada no código principal

```
ARCHITECTURE my_arch OF my_entity IS
-----
    FUNCTION positive_edge(SIGNAL s: STD_LOGIC)
        RETURN BOOLEAN IS
    BEGIN
        RETURN s'EVENT AND s='1';
    END positive_edge;
-----
BEGIN
    PROCESS (clk, rst)
    BEGIN
        IF (rst='1') THEN q <= '0';
        ELSIF positive_edge(clk) THEN q <= d;
        END IF;
    END PROCESS;
END my_arch;
```

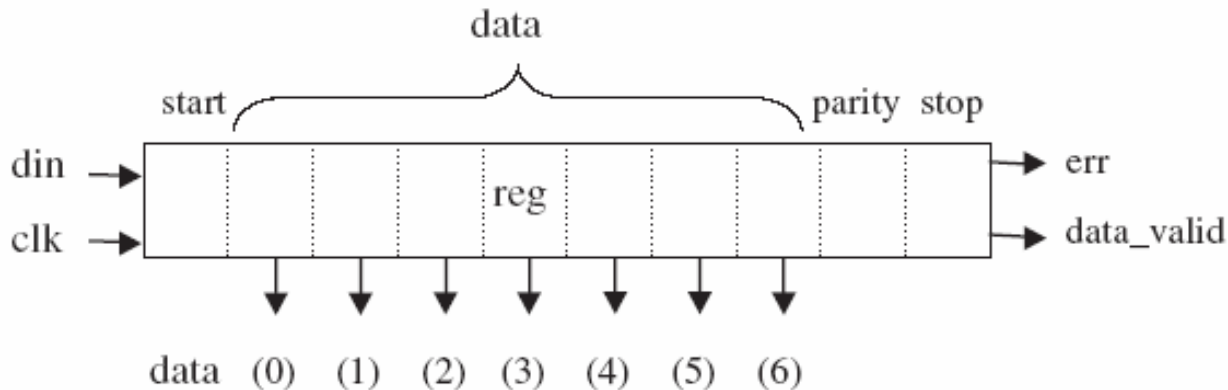
➤ Função localizada em um pacote

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
-----
PACKAGE my_package IS
    FUNCTION positive_edge(SIGNAL s: STD_LOGIC) RETURN BOOLEAN;
END my_package;
-----
PACKAGE BODY my_package IS
    FUNCTION positive_edge(SIGNAL s: STD_LOGIC)
    RETURN BOOLEAN IS
    BEGIN
        RETURN s'EVENT AND s='1';
    END positive_edge;
END my_package;
-----
----- Main code: -----
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE work.my_package.all;
```

- Os procedimentos são muito parecidos com as funções com a diferença que eles podem retornar mais de um valor.
- Exemplo:

```
PROCEDURE my_procedure ( a: IN BIT; SIGNAL b, c: IN BIT;  
                        SIGNAL x: OUT BIT_VECTOR(7 DOWNT0 0);  
                        SIGNAL y: INOUT INTEGER RANGE 0 TO 99) IS  
  
BEGIN  
    ...  
END my_procedure;
```

Exercício: Receptor de Dados Serial

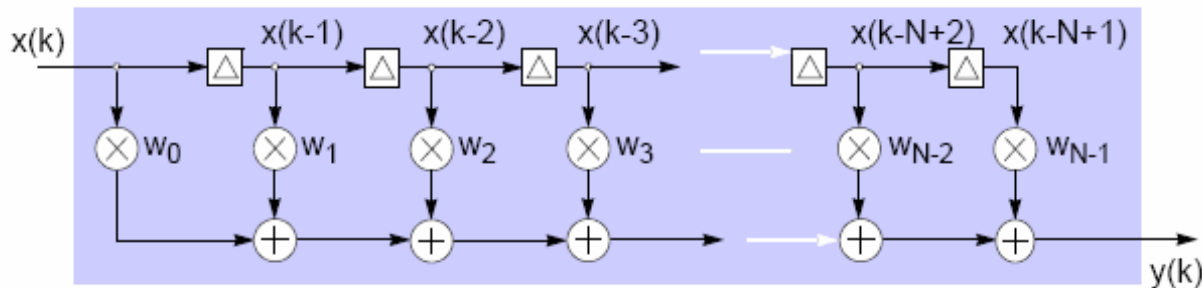


➤ Entidade

```
LIBRARY ieee;  
USE ieee.std_logic_1164.all;  
  
-----  
ENTITY receiver IS  
    PORT ( din, clk, rst: IN BIT;  
          data: OUT BIT_VECTOR (6 DOWNTO 0);  
          err, data_valid: OUT BIT);  
END receiver;  
  
-----
```

```
ARCHITECTURE rtl OF receiver IS
BEGIN
    PROCESS (rst, clk)
        VARIABLE count: INTEGER RANGE 0 TO 10;
        VARIABLE reg: BIT_VECTOR (10 DOWNTO 0);
        VARIABLE temp : BIT;
    BEGIN
        IF (rst='1') THEN
            count:=0;
            reg := (reg'RANGE => '0');
            temp := '0';
            err <= '0';
            data_valid <= '0';
        ELSIF (clk'EVENT AND clk='1') THEN
            IF (reg(0)='0' AND din='1') THEN
                reg(0) := '1';
            ELSIF (reg(0)='1') THEN
                count := count + 1;
                IF (count < 10) THEN
                    reg(count) := din;
                ELSIF (count = 10) THEN
                    temp := (reg(1) XOR reg(2) XOR reg(3) XOR
                        reg(4) XOR reg(5) XOR reg(6) XOR
                        reg(7) XOR reg(8)) OR NOT reg(9);
                    err <= temp;
                    count := 0;
                    reg(0) := din;
                    IF (temp = '0') THEN
                        data_valid <= '1';
                        data <= reg(7 DOWNTO 1);
                    END IF;
                END IF;
            END IF;
        END IF;
    END PROCESS;
END rtl;
```

Exercícios: Filtro FIR 8 taps



$$y(k) = \sum_{n=0}^{N-1} w_n x(k-n)$$

- Coeficientes:

$$[w_0=2^{-3}, w_1=2^{-2}, w_2=2^{-1}, w_3=2^{-0}, w_4=2^{-0}, w_5=2^{-1}, w_6=2^{-2}, w_7=2^{-3}]$$

- Para realizar a simulação, considere que o valor do sinal de entrada seja igual a 100.

Filtro FIR

```
library ieee;
use ieee.std_logic_1164.all;
entity FIR1 is
    port (clk : in std_logic;
          x : in integer range 0 to 255;
          y : out integer range 0 to 511);
end entity FIR1;
architecture arch1 of FIR1 is
begin
    process (clk)
        type RegType is array (7 downto 0) of integer;
        variable Reg: RegType := (others => 0);
    begin
        if (clk'event and clk='1') then
            -- multiply/accumulate (M&C) operation
            y <= Reg(0)/8 + Reg(1)/4 + Reg(2)/2 + Reg(3)
                + Reg(4) + Reg(5)/2 + Reg(6)/4 + Reg(7)/8;
            -- update register values by shifting
            Reg(0) := Reg(1); Reg(1) := Reg(2); Reg(2) := Reg(3); Reg(3) := Reg(4);
            Reg(4) := Reg(5); Reg(5) := Reg(6); Reg(6) := Reg(7); Reg(7) := x;
        end if;
    end process;
end architecture arch1;
```

Implementação de projeto no Quartus

- A sequência de imagens a seguir mostra os passos necessários para implementar um projeto dentro de uma FPGA/CPLD.

Project Navigator

Entity

- MAX7000S: EPM7128SL
- exe_FFD

Device...

- Pins
- Timing Analysis Settings...
- EDA Tool Settings...
- Settings... Ctrl+Shift+E
- Classic Timing Analyzer Wizard...
- Assignment Editor Ctrl+Shift+A
- Pin Planner Ctrl+Shift+N
- Remove Assignments...
- Demote Assignments...
- Back-Annotate Assignments...
- Import Assignments...
- Export Assignments...
- Assignment (Time) Groups...
- Timing Closure Floorplan
- LogicLock Regions Window Alt+L
- Design Partitions Window Alt+D

Exemplo Flip-Flop D

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

ENTITY exe_FFD IS
    PORT
    (
        d, clock, reset: IN STD_LOGIC;
        q: OUT STD_LOGIC
    );
END exe_FFD;

ARCHITECTURE funcionamento OF exe_FFD IS
BEGIN
    PROCESS (reset, clock)
    BEGIN
        IF (reset='1') THEN
            q <= '0';
        ELSIF (clock'EVENT AND clock='1') THEN
            q <= d;
        END IF;
    END PROCESS;
END funcionamento;
```

267
268
ab/

21
22
23
24
25

Hierarchy Files Design Units

Module	Progress %	Time
--------	------------	------

Type Message

System Processing Extra Info Info Warning Critical Warning Error Suppressed Flag

Message: Location: Locate

Project Navigator

Entity	Macros
MAX7000S: EPM7128SLC84-15	
exe_FFD	1

Hierarchy | Files | Design Units

Module	Progress %	Time
--------	------------	------

```
1 -- Exemplo Flip-Flop D
2
3 LIBRARY ieee;
4 USE ieee.std_logic_1164.all;
```

Settings - exe_FFD

Category:

- General
- Files
- Libraries
- Device
- Operating Settings and Conditions
- Compilation Process Settings
- EDA Tool Settings
- Analysis & Synthesis Settings
- Fitter Settings
- Timing Analysis Settings
- Assembler
- Design Assistant
- SignalTap II Logic Analyzer
- Logic Analyzer Interface
- Simulator Settings
- PowerPlay Power Analyzer Settings

Device

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

Family: MAX7000S

Device and Pin Options...

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

Speed grade: Any

Show advanced devices

HardCopy compatible only

Available devices:

Name	Core v...	Macro...
EPM7128SLC84-7	5.0V	128
EPM7128SLC84-10	5.0V	128
EPM7128SLC84-15	5.0V	128
EPM7128SLI84-10	5.0V	128
EPM7128SQC100-6	5.0V	128
EPM7128SQC100-7	5.0V	128
EPM7128SQC100-10	5.0V	128
EPM7128SQC100-15	5.0V	128
EPM7128SQC160-6	5.0V	128
EPM7128SQC160-7	5.0V	128
EPM7128SQC160-10	5.0V	128

Migration compatibility

Migration Devices...

0 migration devices selected

Comparison device

HardCopy II:

Limit DSP & RAM to HardCopy II device resources

OK Cancel

Type	Message
------	---------

System | Processing | Extra Info | Info | Warning | Critical Warning | Error | Suppressed | Flag

Message: Location:



Project Navigator

Entity
MAX7000S: EPM7128SL
exe_FFD

- Device...
- Pins
- Timing Analysis Settings...
- EDA Tool Settings...
- Settings... Ctrl+Shift+E
- Classic Timing Analyzer Wizard...
- Assignment Editor Ctrl+Shift+A
- Pin Planner Ctrl+Shift+N
- Remove Assignments...
- Demote Assignments...
- Back-Annotate Assignments...
- Import Assignments...
- Export Assignments...
- Assignment (Time) Groups...
- Timing Closure Floorplan
- LogicLock Regions Window Alt+L
- Design Partitions Window Alt+D

```
Exemplo Flip-Flop D  
  
LIBRARY ieee;  
USE ieee.std_logic_1164.all;  
  
ENTITY exe_FFD IS  
    PORT  
    (  
        d, clock, reset: IN STD_LOGIC;  
        q: OUT STD_LOGIC  
    );  
END exe_FFD;  
  
ARCHITECTURE funcionamento OF exe_FFD IS  
BEGIN  
    PROCESS (reset, clock)  
    BEGIN  
        IF (reset='1') THEN  
            q <= '0';  
        ELSIF (clock'EVENT AND clock='1') THEN  
            q <= d;  
        END IF;  
    END PROCESS;  
END funcionamento;
```

267
268
ab/
21
22
23
24
25

Hierarchy Files Design Units

Status

Module	Progress %	Time
--------	------------	------

Messages

Type	Message
------	---------

System Processing Extra Info Info Warning Critical Warning Error Suppressed Flag

Message: Location: Locate

Project Navigator

Entity	Maci
MAX7000S: EPM7128SLC84-15	
exe_FFD	1

Hierarchy Files Design Units

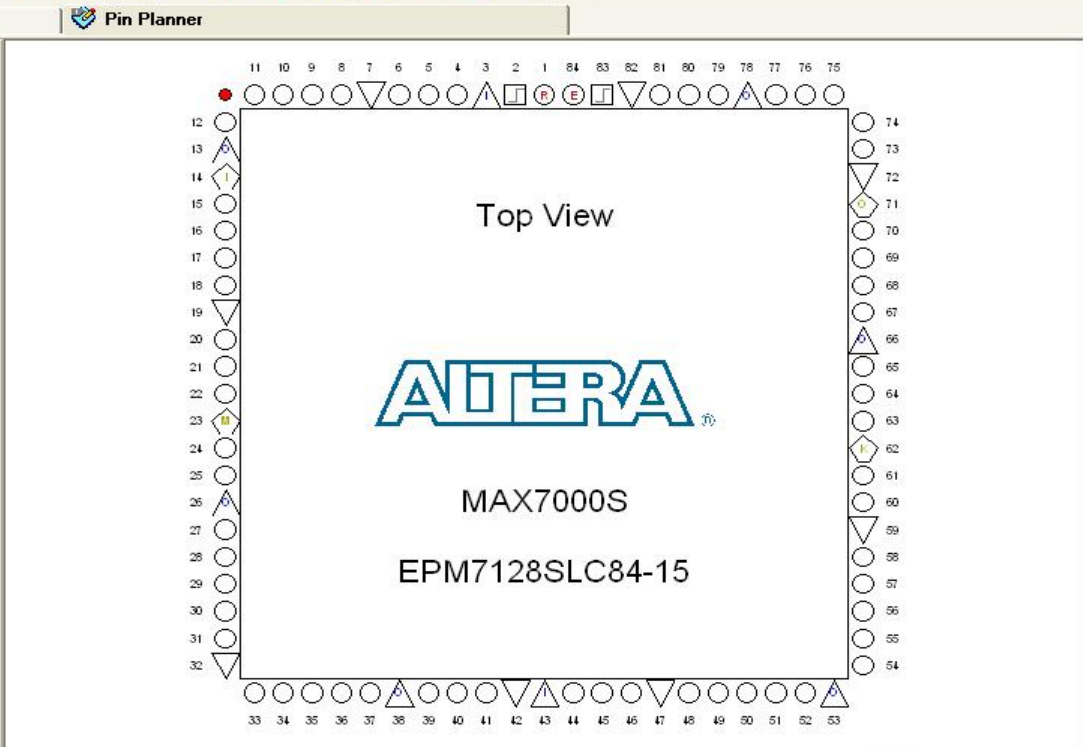
exe_FFD.vhd

Pin Planner

Groups

Named: []

Node Name	Di
<<new node>>	



Status

Module	Progress %	Time

Named: [] Edit: [X] [✓] Filter: Pins: all

	Node Name	Direction	Location	Reserved	Group
1	clock	Input			
2	d	Input			
3	q	Output			
4	reset	Input			
5	<<new node>>				

Messages

Type	Message

System Processing Extra Info Info Warning Critical Warning Error Suppressed Flag

Message: [] Location: [] Locate



Project Navigator

Entity	Maci
MAX7000S: EPM7128SLC84-15	
exe_FFD	1

Hierarchy | Files | Design Units

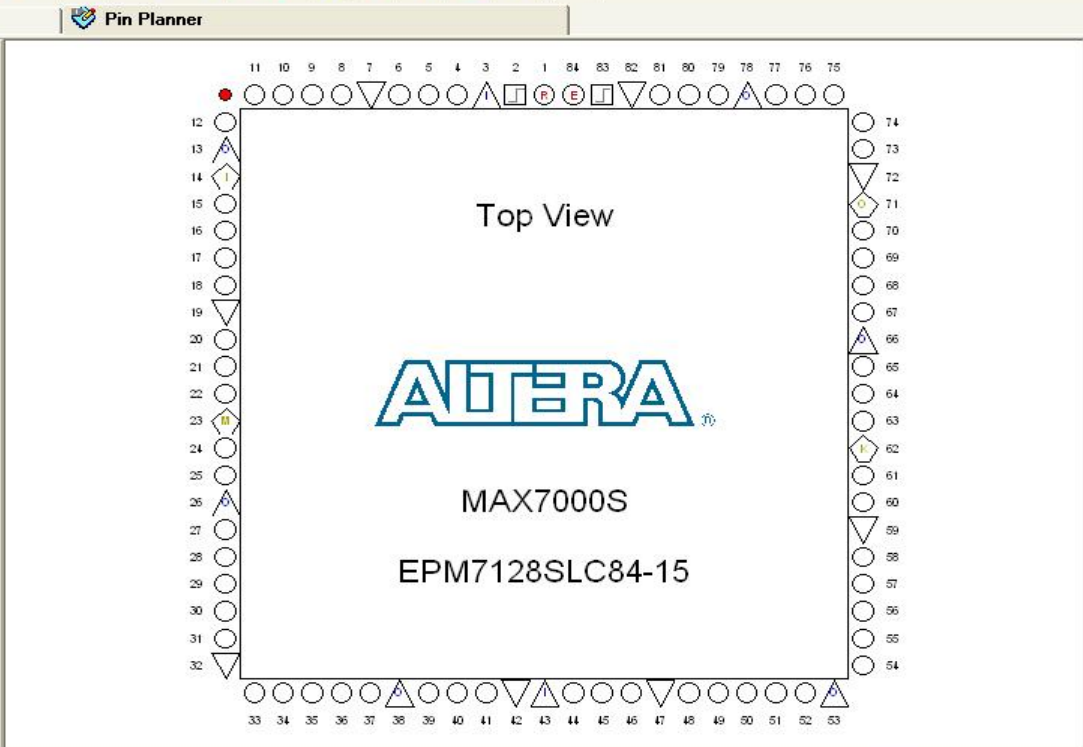
exe_FFD.vhd

Pin Planner

Groups

Named: []

Node Name	Di
<<new node>>	



Status

Module	Progress %	Time

Named: [] Edit: [X] [Y] [Z]

Filter: Pins: all

	Node Name	Direction	Location	Reserved	Group
1	clock	Input			
2	d	Input			
3	q	Output			
4	reset	Input			
5	<<new node>>				

Type	Message

Messages

System | Processing | Extra Info | Info | Warning | Critical Warning | Error | Suppressed | Flag

Message: [] Location: []



Project Navigator

Entity	Mac
MAX7000S: EPM7128SLC84-15	
exe_FFD	1

Hierarchy | Files | Design Units

Status

Module	Progress %	Time
--------	------------	------

Groups

Named: []

Node Name	Di
<<new node>>	

Pin Properties

Pin number: PIN_30

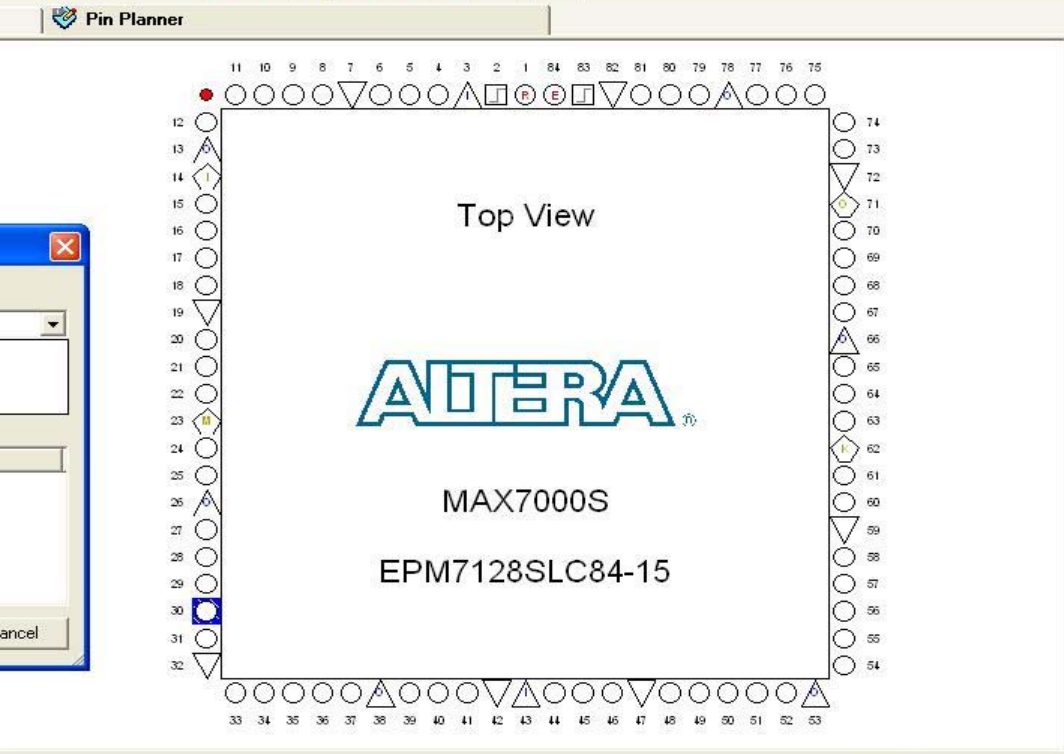
Node name: []

I/O standard: clock, d, q, reset

Reserved:

Name	Value
I/O Bank	
General Function	I/O
Special Function	N/A
Pad ID	29
VREF Pad ID	N/A

OK | Cancel



Named: [] Edit: [] Filter: Pins: all

	Node Name	Direction	Location	Reserved	Group
1	clock	Input			
2	d	Input			
3	q	Output			
4	reset	Input			
5	<<new node>>				

Messages

Type | Message

System | Processing | Extra Info | Info | Warning | Critical Warning | Error | Suppressed | Flag

Message: [] Location: []

File Edit View Project Assignments Processing Tools Window Help

Project Navigator

Entity

- MAX7000S: EPM7128SL
- exe_FFD

Settings... Ctrl+Shift+E

Classic Timing Analyzer Wizard...

Assignment Editor Ctrl+Shift+A

Pin Planner Ctrl+Shift+N

Remove Assignments...

Demote Assignments...

Back-Annotate Assignments...

Import Assignments...

Export Assignments...

Assignment (Time) Groups...

Timing Closure Floorplan

LogicLock Regions Window Alt+L

Design Partitions Window Alt+D

Node Name

<<new node>>

Pin Planner

Top View

ALTERA

MAX7000S

EPM7128SLC84-15

Hierarchy Files Design Units

Status

Module	Progress %	Time
Analysis & Synthesis	100 %	00:00:04

Named: PIN_15 Filter: Pins: all

	Node Name	Direction	Location	Reserved	Group
1	clock	Input	PIN_83		
2	d	Input	PIN_11		
3	q	Output	PIN_12		
4	reset	Input	PIN_15		
5	<<new node>>				

Messages

Type	Message
Info	Running Quartus II Analysis & Synthesis
Info	Command: quartus_map --read_settings_files=on --write_settings_files=off exe_FFD -c exe_FFD
Info	Found 2 quartus units, including 1 entities, in source file exe_FFD.vhd
Info	Elaborating entity "exe_FFD" for the top level hierarchy
Info	Promoted pin-driven signal(s) to global signal
Info	Implemented 5 device resources after synthesis - the final resource count might be different
Info	Quartus II Analysis & Synthesis was successful. 0 errors, 0 warnings

System (2) Processing (8) Extra Info Info (8) Warning Critical Warning Error Suppressed Flag

Message: 0 of 19 Location: Locate

Project Navigator

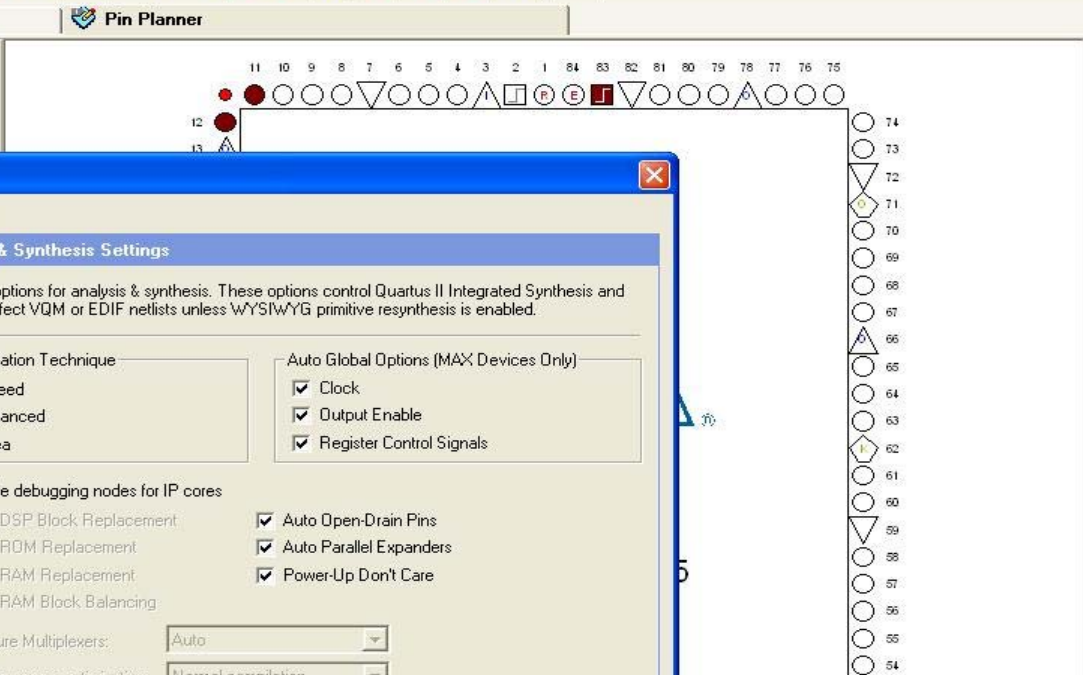
Entity	Maci
MAX7000S: EPM7128SLC84-15	
exe_FFD	1

Hierarchy Files Design Units

Groups

Named: []

Node Name	Di
<<new node>>	



Settings - exe_FFD

Category:

- General
- Files
- Libraries
- Device
- Operating Settings and Conditions
- Compilation Process Settings
- EDA Tool Settings
- Analysis & Synthesis Settings**
 - VHDL Input
 - Verilog HDL Input
 - Default Parameters
 - Synthesis Netlist Optimizations
- Filter Settings
- Timing Analysis Settings
- Assembler
- Design Assistant
- SignalTap II Logic Analyzer
- Logic Analyzer Interface
- Simulator Settings
- PowerPlay Power Analyzer Settings

Analysis & Synthesis Settings

Specify options for analysis & synthesis. These options control Quartus II Integrated Synthesis and do not affect VQM or EDIF netlists unless WYSIWYG primitive resynthesis is enabled.

Optimization Technique

- Speed
- Balanced
- Area

Auto Global Options (MAX Devices Only)

- Clock
- Output Enable
- Register Control Signals

Create debugging nodes for IP cores

- Auto DSP Block Replacement
- Auto ROM Replacement
- Auto RAM Replacement
- Auto RAM Block Balancing
- Auto Open-Drain Pins
- Auto Parallel Expanders
- Power-Up Don't Care

Restructure Multiplexers: [Auto]

PowerPlay power optimization: [Normal compilation]

HDL Message Level: [Level2] [Advanced...]

[More Settings...]

Description:

[]

OK Cancel

Status

Module	Progress %	Time
Analysis & Synthesis	100 %	00:00:04

Messages

Type	Message
Info	Running Quartus II
Info	Command: quartus_map --read_settings_files=on --write_settings_files=off exe_FFD -c exe_FFD
Info	Found 2 design units, including 1 entities, in source file exe_FFD.vhd
Info	Elaborating entity "exe_FFD" for the top level hierarchy
Info	Promoted pin-driven signal(s) to global signal
Info	Implemented 5 device resources after synthesis - the final resource count might be different
Info	Quartus II Analysis & Synthesis was successful. 0 errors, 0 warnings

System (2) Processing (8) Extra Info Info (8) Warning Critical Warning Error Suppressed Flag

Message: 0 of 19 Location: []



Project Navigator

Entity	Maci
MAX7000S: EPM7128SLC84-15	
exe_ffd	1

Hierarchy | Files | Design Units

Status

Module	Progress %	Time
Analysis & Synthesis	100 %	00:00:04

Messages

Type	Message
Info	Running Quartus II
Info	Command: quartus_map --read_settings_files=on --write_settings_files=off exe_ffd -c exe_ffd
Info	Found 2 design units, including 1 entities, in source file exe_ffd.vhd
Info	Elaborating entity "exe_ffd" for the top level hierarchy
Info	Promoted pin-driven signal(s) to global signal
Info	Implemented 5 device resources after synthesis - the final resource count might be different
Info	Quartus II Analysis & Synthesis was successful. 0 errors, 0 warnings

System (2) | Processing (8) | Extra Info | Info (8) | Warning | Critical Warning | Error | Suppressed | Flag

Message: 0 of 19 | Location: | Locate

exe_ffd.vhd

Pin Planner

Groups

Named: []

Node Name	Di
<<new node>>	

Settings - exe_ffd

Category:

- General
- Files
- Libraries
- Device
- Operating Settings and Conditions
- Compilation Process Settings
- EDA Tool Settings
- Analysis & Synthesis Settings
 - VHDL Input
 - Verilog HDL Input
 - Default Parameters
 - Synthesis Netlist Optimizations
- Filter Settings
- Timing Analysis Settings
 - TimeQuest Timing Analyzer
 - Classic Timing Analyzer Settings**
 - Classic Timing Analyzer Report
- Assembler
- Design Assistant
- SignalTap II Logic Analyzer
- Logic Analyzer Interface
- Simulator Settings
 - Simulation Verification
 - Simulation Output Files
- PowerPlay Power Analyzer Settings

Classic Timing Analyzer Settings

Specify settings for the Classic Timing Analyzer. Use the Assignment Editor for individual timing assignments. Note: These settings affect the Classic Timing Analyzer only. To specify TimeQuest Timing Analyzer settings, use the TimeQuest Timing Analyzer (Timing Analysis Settings menu).

Delay requirements

tsu: [] ns [v]
tco: [] ns [v]
tpd: [] ns [v]
th: [] ns [v]

Report minimum timing checks

Minimum delay requirements

Minimum tco: [] ns [v]
Minimum tpd: [] ns [v]

Clock Settings

Default required fmax: [] MHz [v]

Individual Clocks...

More Settings...

Description:

[]

OK Cancel

Project Navigator

Entity	Mac
MAX7000S: EPM7128SLC84-15	
└─> abcd vhd exe_FFD	1

Hierarchy Files Design Units

Analysis & Synthesis 0% 00:00:00

Filter 0% 00:00:00

Assembler 0% 00:00:00

Classic Timing Analyzer 0% 00:00:00

Idle 0% 00:00:00

Start Stop Report

Status

Module	Progress %	Time
Analysis & Synthesis	100 %	00:00:04

Messages

Type	Message
Info	Running Quartus II Analysis & Synthesis
Info	Command: quartus_map --read_settings_files=on --write_settings_files=off exe_FFD -c exe_FFD
Info	Found 2 design units, including 1 entities, in source file exe_FFD.vhd
Info	Elaborating entity "exe_FFD" for the top level hierarchy
Info	Promoted pin-driven signal(s) to global signal
Info	Implemented 5 device resources after synthesis - the final resource count might be different
Info	Quartus II Analysis & Synthesis was successful. 0 errors, 0 warnings

System (2) Processing (8) Extra Info Info (8) Warning Critical Warning Error Suppressed Flag

Message: 0 of 19 Location:

Project Navigator

Entity	Mac
MAX7000S: EPM7128SLC84-15	
exe_FFD	1

Hierarchy | Files | Design Units

Analysis & Synthesis 00:00:03	Fitter 00:00:00	Assembler 00:00:00	Classic Timing Analyzer 00:00:00		
Full Compilation - Analysis & Synthesis 00:00:03					
▶ Start		⏹ Stop		📄 Report	

Status

Module	Progress %	Time
Full Compilation	23 %	00:00
Analysis & Synthesis	96 %	00:00
Fitter	0 %	00:00
Assembler	0 %	00:00
Classic Timing Analyzer	0 %	00:00

Messages

Type	Message
Info	Info: *****
Info	Info: Running Quartus II Analysis & Synthesis
Info	Info: Command: quartus_map --read_settings_files=on --write_settings_files=off exe_FFD -c exe_FFD

System (7) | Processing (3) | Extra Info | Info (3) | Warning | Critical Warning | Error | Suppressed | Flag

Message: 0 of 5 | Location: []

Project Navigator

Entity	Mac
MAX7000S: EPM7128SLC84-15	
abd vhd exe_FFD	1

Hierarchy | Files | Design Units

Status

Module	Progress %	Time
Full Compilation	100 %	00:00
Analysis & Synthesis	100 %	00:00
Filter	100 %	00:00
Assembler	100 %	00:00
Classic Timing Analyzer	100 %	00:00

- EDA Simulation Tool
- Run EDA Timing Analysis Tool
- Launch Design Space Explorer
- TimeQuest Timing Analyzer
- Advisors
- Chip Planner (Floorplan and Chip Editor)
- Netlist Viewers
- SignalTap II Logic Analyzer
- In-System Memory Content Editor
- Logic Analyzer Interface Editor
- In-System Sources and Probes Editor
- SignalProbe Pins...
- Programmer**
- MegaWizard Plug-In Manager...
- SOPC Builder...
- Tcl Scripts...
- Customize...
- Options...
- License Setup...
- Customize Compiler Tool...

Pin Planner | Compiler Tool

Analysis & Synthesis: 100 % 00:00:04

Filter: 100 % 00:00:02

Assembler: 100 % 00:00:02

Classic Timing Analyzer: 100 % 00:00:02

Full Compilation: 100 % 00:00:10

Start | Stop | Report

Messages

Type	Message
Warning	Warning: Found pins functioning as undefined clocks and/or memory enables
Info	Info: No valid register-to-register data paths exist for clock "clock"
Info	Info: tsu for register "q-reg0" (data pin = "d", clock pin = "clock") is 11.000 ns
Info	Info: tco from clock "clock" to destination pin "q" through register "q-reg0" is 8.000 ns
Info	Info: th for register "q-reg0" (data pin = "d", clock pin = "clock") is -3.000 ns
Info	Info: Quartus II Classic Timing Analyzer was successful. 0 errors, 2 warnings
Info	Info: Quartus II Full Compilation was successful. 0 errors, 2 warnings

System (8) | Processing (31) | Extra Info | Info (29) | Warning (2) | Critical Warning | Error | Suppressed | Flag

Message: 0 of 87 | Location:

Project Navigator

Entity	Mac
MAX7000S: EPM7128SLC84-15	
exe_FFD	1

Hierarchy Files Design Units

exe_FFD.vhd | Pin Planner | Compiler Tool | exe_FFD.cdf

Hardware Setup... No Hardware Mode: JTAG Progress: 0%

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

Start	File	Device	Checksum	Usercode	Program/Configure	Verify	Blank-Check	Examine	Security Bit	Erase	ISP CLAMP
Stop	exe_FFD.pof	EPM7128SL84	001E0FF2	0000FFFF	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

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

Status

Module	Progress %	Time
Full Compilation	100 %	00:00
Analysis & Synthesis	100 %	00:00
Filter	100 %	00:00
Assembler	100 %	00:00
Classic Timing Analyzer	100 %	00:00

Messages

Type	Message
Warning	Warning: Found pins functioning as undefined clocks and/or memory enables
Info	Info: No valid register-to-register data paths exist for clock "clock"
Info	Info: tsu for register "q-reg0" (data pin = "d", clock pin = "clock") is 11.000 ns
Info	Info: tco from clock "clock" to destination pin "q" through register "q-reg0" is 8.000 ns
Info	Info: th for register "q-reg0" (data pin = "d", clock pin = "clock") is -3.000 ns
Info	Info: Quartus II Classic Timing Analyzer was successful. 0 errors, 2 warnings
Info	Info: Quartus II Full Compilation was successful. 0 errors, 2 warnings

System (8) Processing (31) Extra Info Info (29) Warning (2) Critical Warning Error Suppressed Flag

Message: 0 of 87 Location:

Project Navigator

Entity	Mac
MAX7000S: EPM7128SLC84-15	
exe_FFD	1

Hierarchy | Files | Design Units

exe_FFD.vhd | Pin Planner | Compiler Tool | exe_FFD.cdf

Hardware Setup... No Hardware Mode: JTAG Progress: 0%

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

File	Device	Checksum	Usercode	Program/Configure	Verify	Blank-Check	Examine	Security Bit	Erase	ISP CLAMP
exe_FFD.pof	EPM7128SL84	001E0FF2	0000FFFF	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<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

Hardware Setup

Hardware Settings | JTAG Settings

Select a programming hardware setup to use when programming devices. This programming hardware setup applies only to the current programmer window.

Currently selected hardware: ByteBlaster [LPT1]

Available hardware items:

Hardware	Server	Port
ByteBlaster	Local	LPT1

Add Hardware...
Remove Hardware

Close

Status

Module	Progress %	Time
Full Compilation	100 %	00:00
Analysis & Synthesis	100 %	00:00
Filter	100 %	00:00
Assembler	100 %	00:00
Classic Timing Analyzer	100 %	00:00

Messages

Type	Message
Warning	Warning: Found pins functioning as undefined clocks and/or memory enables
Info	Info: No valid register-to-register data paths exist for clock "clock"
Info	Info: tsu for register "q-reg0" (data pin = "d", clock pin = "clock") is 11.000 ns
Info	Info: tco from clock "clock" to destination pin "q" through register "q-reg0" is 8.000 ns
Info	Info: th for register "q-reg0" (data pin = "d", clock pin = "clock") is -3.000 ns
Info	Info: Quartus II Classic Timing Analyzer was successful. 0 errors, 2 warnings
Info	Info: Quartus II Full Compilation was successful. 0 errors, 2 warnings

System (8) | Processing (31) | Extra Info | Info (29) | Warning (2) | Critical Warning | Error | Suppressed | Flag

Message: 0 of 87

Location:

Exemplo: MAX II Altera

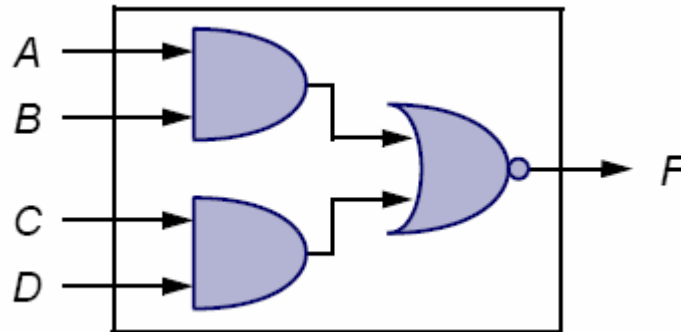
- Carregar arquivo 'Functional_Test.pof'

Introdução ao ISE - Xilinx

➤ <http://www.xilinx.com/>

Utilizando o ISE - Exercício

- Implementar o exemplo abaixo no ISE e gerar um arquivo de forma de onda para testar o funcionamento do projeto.



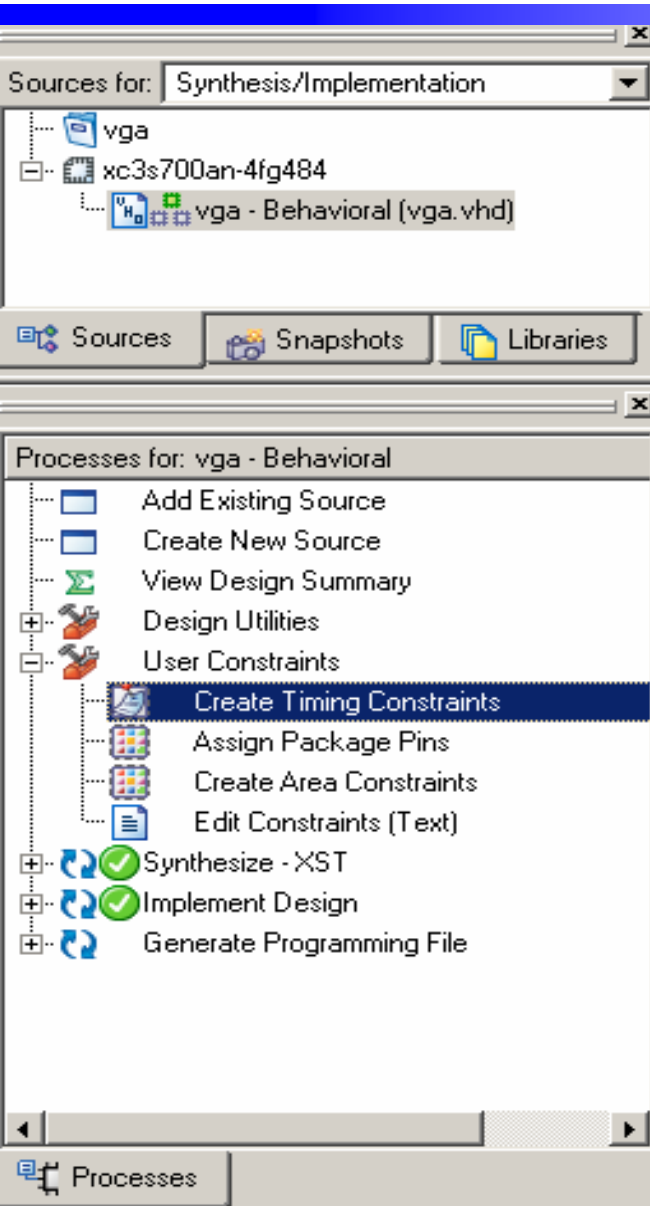
Exercício: Gerado de Padrão VGA

- Analisar o código fornecido.
- Implementar o projeto no kit SPARTAN 3AN da Xilinx.

Implementação do projeto no ISE

- A seqüência de imagens a seguir mostra os passos necessários para implementar um projeto dentro de uma FPGA.

Configurando restrições de tempo



The screenshot shows the Xilinx IDE interface. The top window, titled "Sources for: Synthesis/Implementation", displays a project tree with the following structure:

- vga
 - xc3s700an-4fg484
 - vga - Behavioral (vga.vhd)

Below this window is a toolbar with "Sources", "Snapshots", and "Libraries" buttons. The bottom window, titled "Processes for: vga - Behavioral", shows a list of design processes:

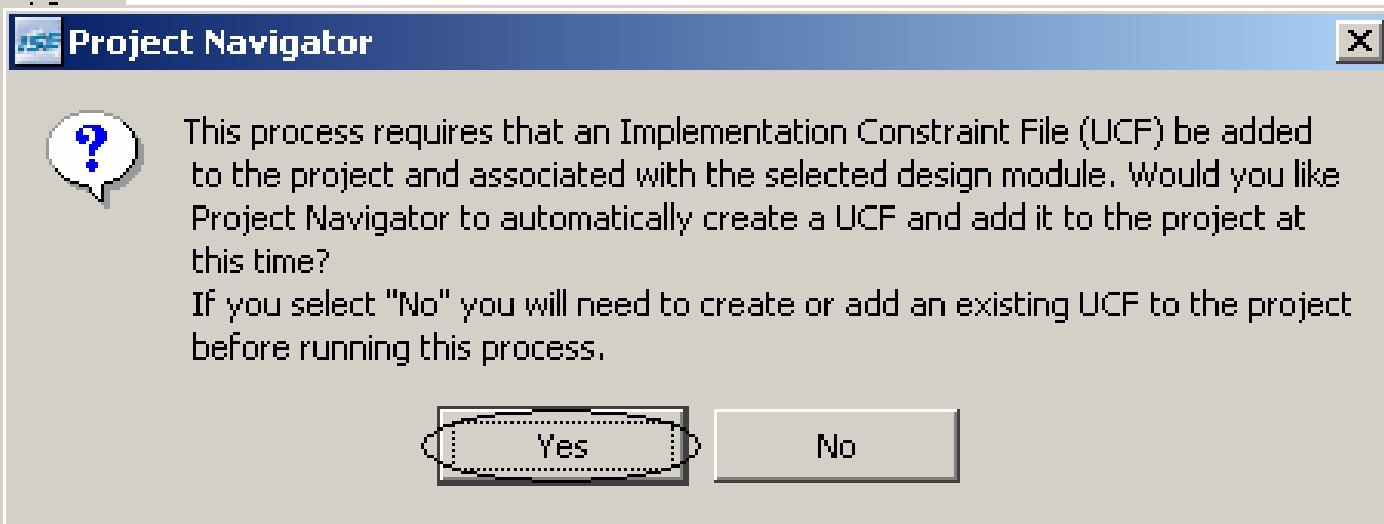
- Add Existing Source
- Create New Source
- View Design Summary
- Design Utilities
- User Constraints
 - Create Timing Constraints** (highlighted)
 - Assign Package Pins
 - Create Area Constraints
 - Edit Constraints (Text)
- Synthesize - XST
- Implement Design
- Generate Programming File

At the bottom of the interface, there are buttons for "Processes", "Design Summary", and "vga.vhd".

```
1 -----
2 -- Company:
3 -- Engineer:
4 --
5 -- Create Date:      15:00:29 04/07/2008
6 -- Design Name:
7 -- Module Name:     vga - Behavioral
8 -- Project Name:
9 -- Target Devices:
10 -- Tool versions:
11 -- Description:
12 --
13 -- Dependencies:
14 --
15 -- Revision:
16 -- Revision 0.01 - File Created
17 -- Additional Comments:
18 --
19 -----
20 library IEEE;
21 use IEEE.STD_LOGIC_1164.ALL;
22 use IEEE.STD_LOGIC_ARITH.ALL;
23 use IEEE.STD_LOGIC_UNSIGNED.ALL;
24
25 ---- Uncomment the following library declaration if in
26 ---- any Xilinx primitives in this code.
27 --library UNISIM;
28 --use UNISIM.VComponents.all;
29
30 entity vga is
31     Port ( clock : in  STD_LOGIC;
```

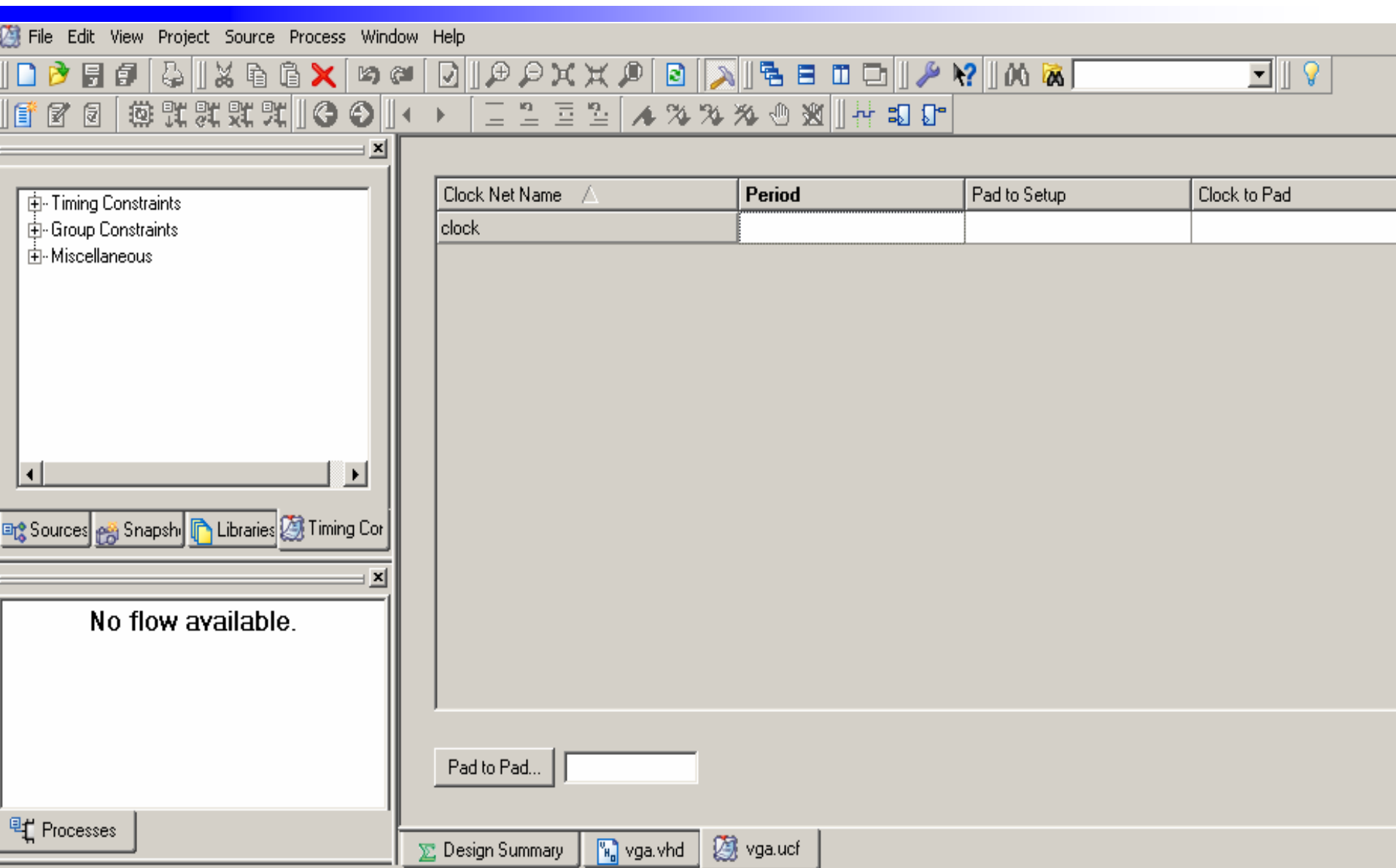
Criando Arquivo de Restrições do Usuário - UCF

```
8  -- Project Name:  
9  -- Target Devices:  
10 -- Tool versions:  
11 -- Description:
```



```
23  use IEEE.STD_LOGIC_UNSIGNED.ALL;  
24  
25  ---- Uncomment the following library declaration if instantiat  
26  ---- any Xilinx primitives in this code.  
27  --library UNISIM;
```

Arquivo .ucf



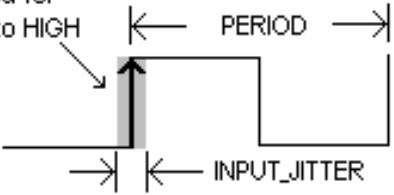
The screenshot shows the Xilinx ISE software interface. The main window displays the UCF editor for a file named 'vga.ucf'. The interface includes a menu bar (File, Edit, View, Project, Source, Process, Window, Help), a toolbar with various editing tools, and a project browser on the left. The project browser shows a tree structure with 'Timing Constraints', 'Group Constraints', and 'Miscellaneous'. Below the project browser, there are tabs for 'Sources', 'Snapshots', 'Libraries', and 'Timing Constraints'. The main editor area contains a table with the following columns: 'Clock Net Name', 'Period', 'Pad to Setup', and 'Clock to Pad'. The table has one row with the value 'clock' in the 'Clock Net Name' column. Below the table, there is a 'Pad to Pad...' button and an empty text input field. At the bottom of the window, there are tabs for 'Design Summary', 'vga.vhd', and 'vga.ucf'. A status bar at the bottom left displays 'No flow available.'

Clock Net Name	Period	Pad to Setup	Clock to Pad
clock			

Configurando período de clock

Clock Period

Initial active edge used for OFFSET value is set to HIGH



OK
Cancel
Help

TIMESPEC Name:
TS_clock

Clock Net Name:
clock

Clock Signal Definition

Specify Time

Time: 20 Units: ns

Start HIGH Start LOW

Time HIGH: 50 Units: %

Relative to other PERIOD TIMESPEC

Reference TIMESPEC:

Multiply by Divide by

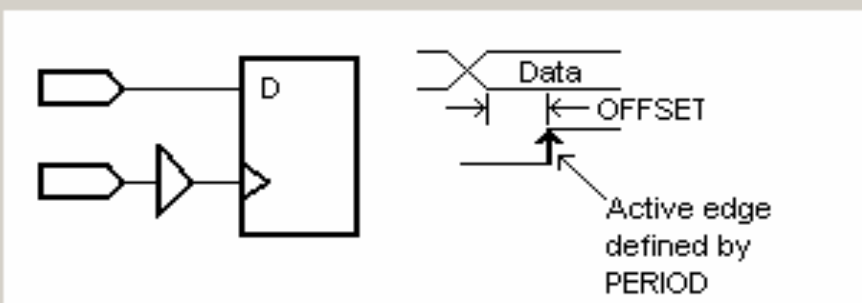
Factors: 1.0

PHASE:

Configurando OFFSET (Intervalo de tempo permitido entre a mudança do sinal e o clock)

Clock Net Name	Period	Pad to Setup	Clock to Pad
clock	20 ns. HIGH 50%		

Pad to Setup



Time Requirement

OFFSET: Units:

Relative to Clock Pad Net:

Relative to Clock Edge:

Comment:

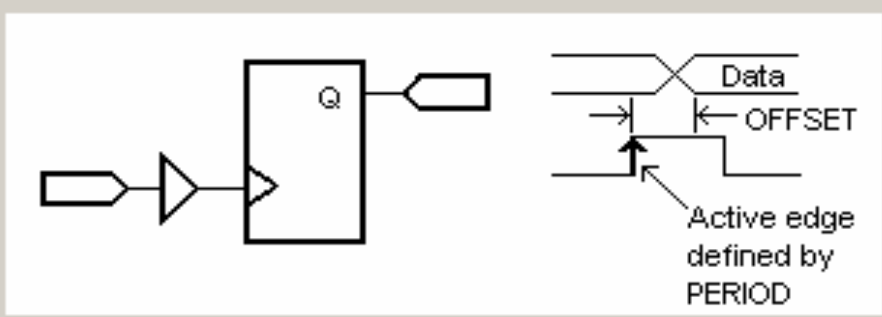
OK
Cancel
Apply
Help

Pad to Pad...

Configurando OFFSET (Intervalo de tempo permitido entre o clock e a mudança do sinal)

Clock Net Name	Period	Pad to Setup	Clock to Pad
clock	20 ns. HIGH 50%	10 ns.	

Clock to Pad
✕



Time Requirement

OFFSET: Units:

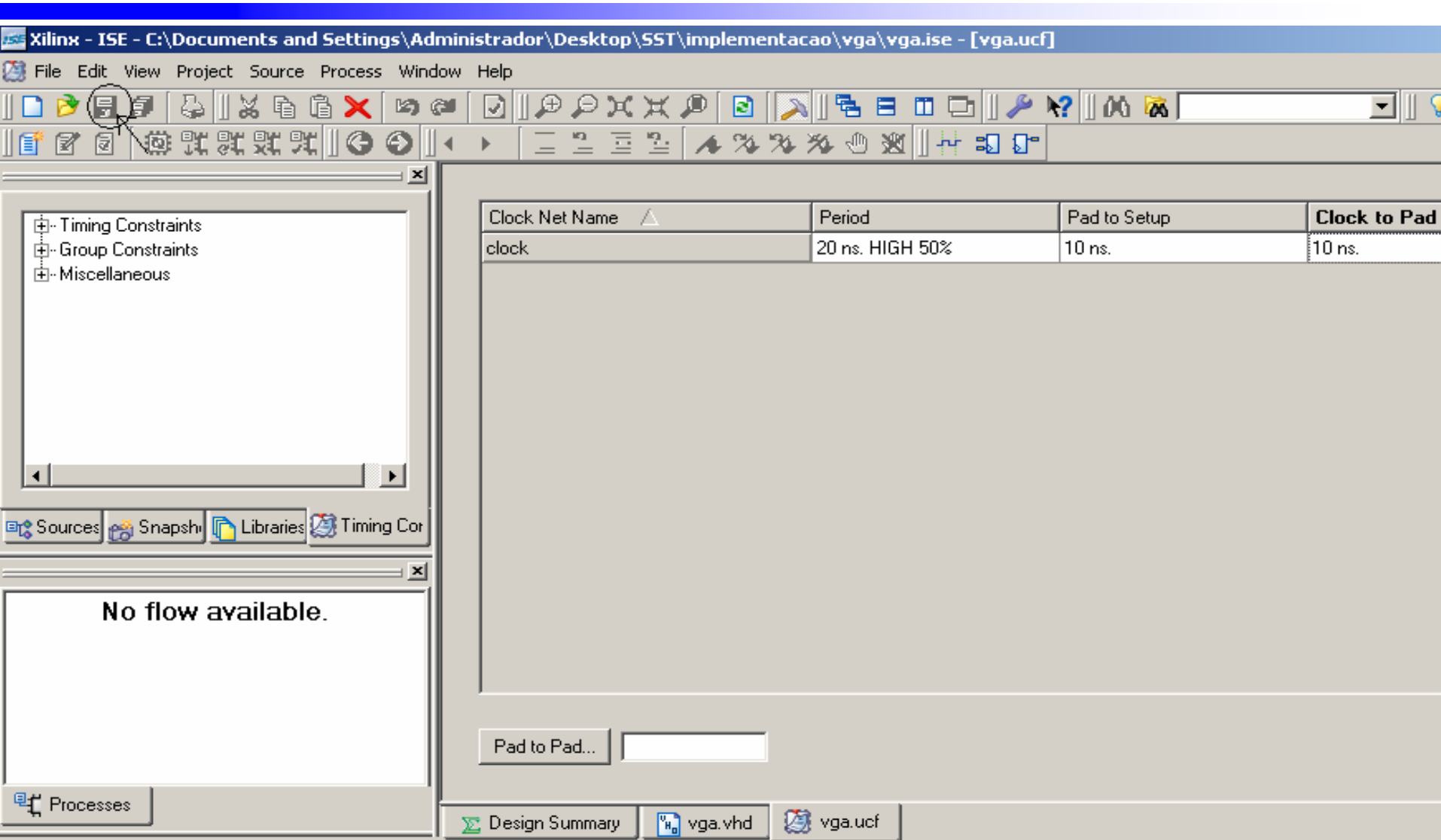
Relative to Clock Pad Net:

Relative to Clock Edge:

Comment:

Pad to Pad...

Salvar arquivo .ucf



Xilinx - ISE - C:\Documents and Settings\Administrador\Desktop\SST\implementacao\vga\vga.ise - [vga.ucf]

File Edit View Project Source Process Window Help

Timing Constraints

- Timing Constraints
- Group Constraints
- Miscellaneous

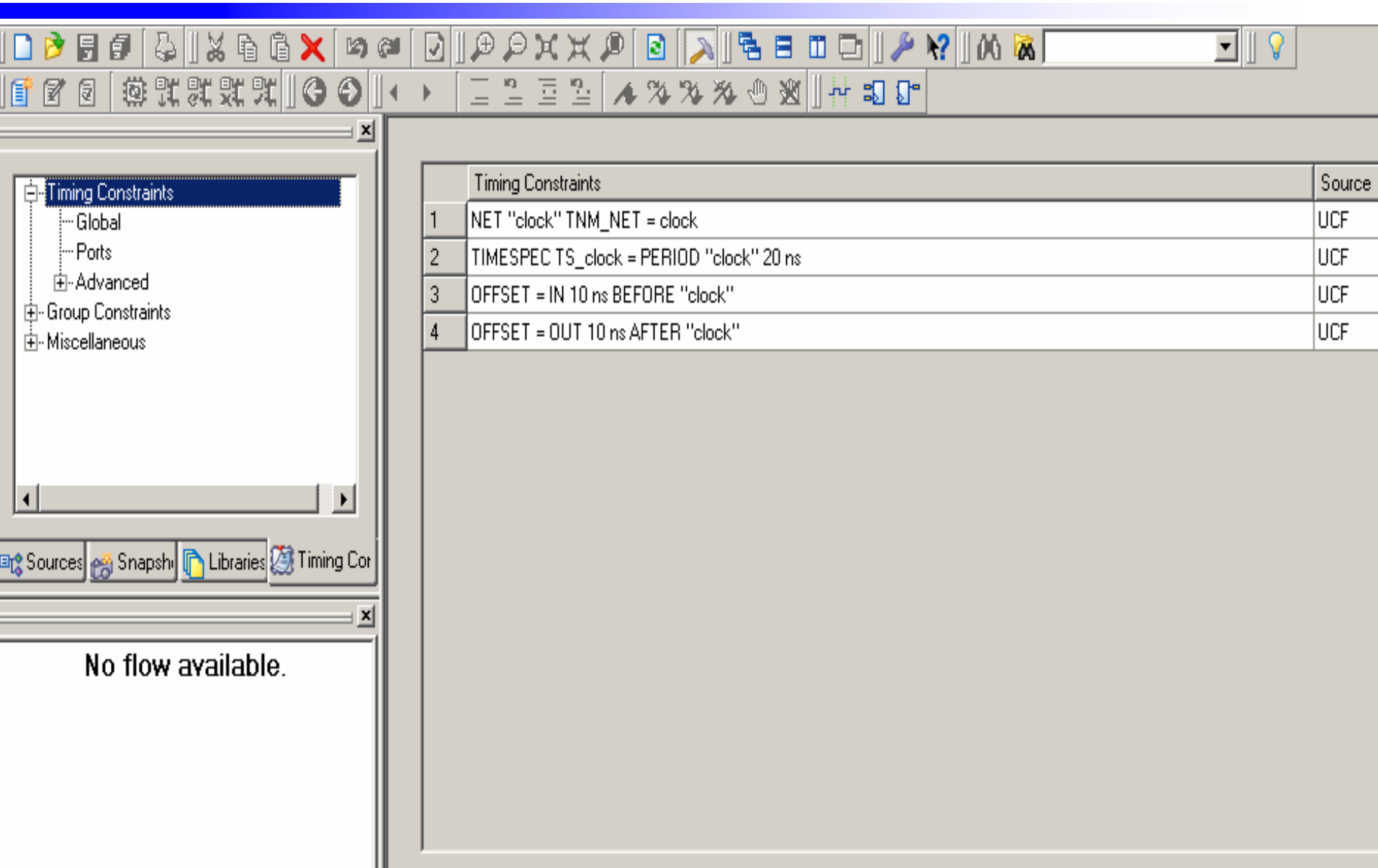
Clock Net Name	Period	Pad to Setup	Clock to Pad
clock	20 ns. HIGH 50%	10 ns.	10 ns.

No flow available.

Pad to Pad...

Design Summary vga.vhd vga.ucf

Visualização das restrições impostas

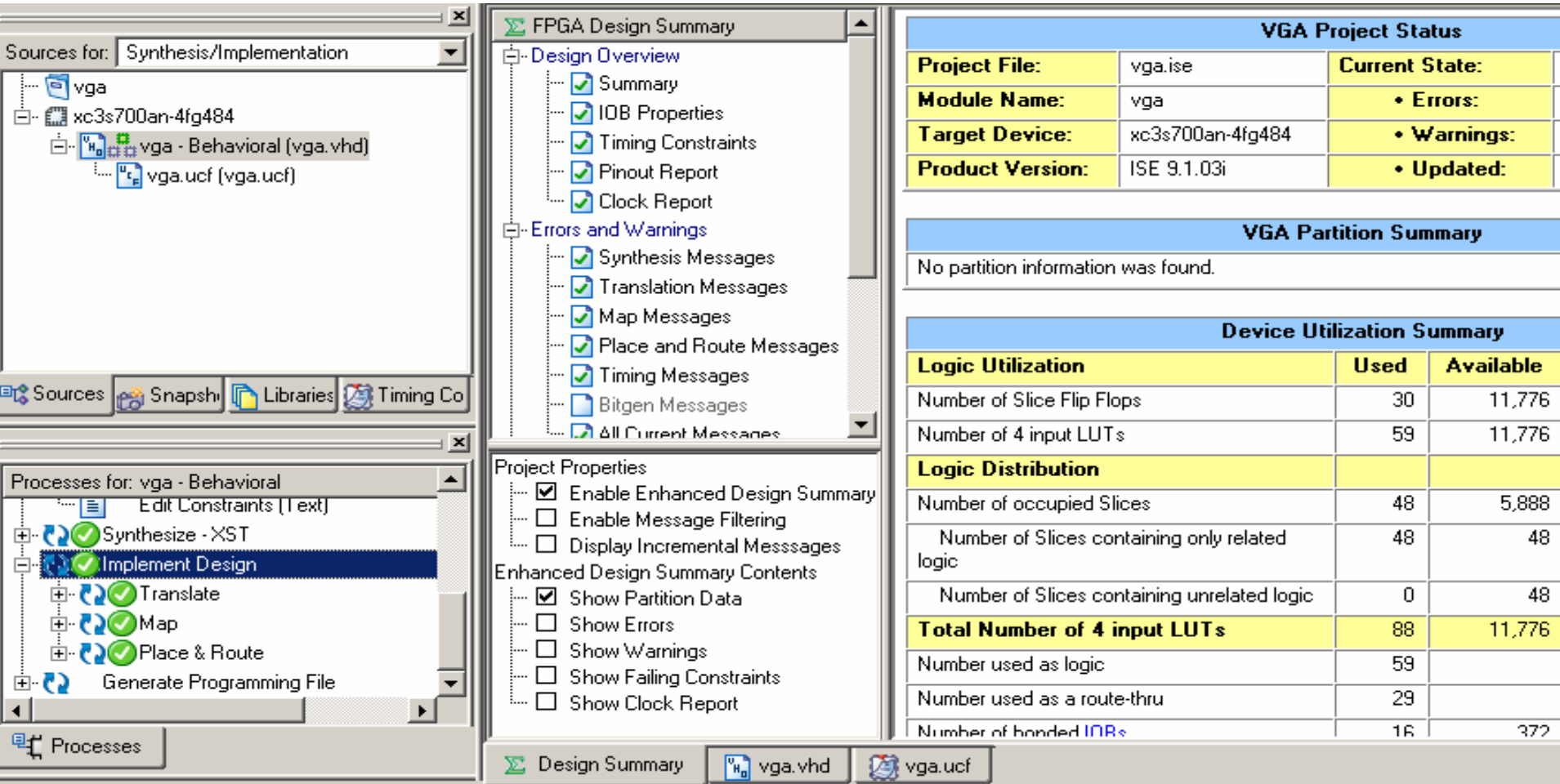


The screenshot displays a software interface for managing timing constraints. On the left, a tree view shows the hierarchy: Timing Constraints (selected), Global, Ports, Advanced, Group Constraints, and Miscellaneous. The main area contains a table of constraints:

	Timing Constraints	Source
1	NET "clock" TNM_NET = clock	UCF
2	TIMESPEC TS_clock = PERIOD "clock" 20 ns	UCF
3	OFFSET = IN 10 ns BEFORE "clock"	UCF
4	OFFSET = OUT 10 ns AFTER "clock"	UCF

At the bottom of the interface, a status bar displays the text: "No flow available."

Implementar o projeto



FPGA Design Summary

Sources for: Synthesis/Implementation

- vga
 - xc3s700an-4fg484
 - vga - Behavioral (vga.vhd)
 - vga.ucf (vga.ucf)

Processes for: vga - Behavioral

- Edit Constraints [Text]
- Synthesize - XST
- Implement Design**
- Translate
- Map
- Place & Route
- Generate Programming File

Design Overview

- Summary
- IOB Properties
- Timing Constraints
- Pinout Report
- Clock Report

Errors and Warnings

- Synthesis Messages
- Translation Messages
- Map Messages
- Place and Route Messages
- Timing Messages
- Bitgen Messages
- All Current Messages

Project Properties

- Enable Enhanced Design Summary
- Enable Message Filtering
- Display Incremental Messages

Enhanced Design Summary Contents

- Show Partition Data
- Show Errors
- Show Warnings
- Show Failing Constraints
- Show Clock Report

VGA Project Status

Project File:	vga.ise	Current State:	
Module Name:	vga	Errors:	
Target Device:	xc3s700an-4fg484	Warnings:	
Product Version:	ISE 9.1.03i	Updated:	

VGA Partition Summary

No partition information was found.

Device Utilization Summary

Logic Utilization	Used	Available
Number of Slice Flip Flops	30	11,776
Number of 4 input LUTs	59	11,776
Logic Distribution		
Number of occupied Slices	48	5,888
Number of Slices containing only related logic	48	48
Number of Slices containing unrelated logic	0	48
Total Number of 4 input LUTs	88	11,776
Number used as logic	59	
Number used as a route-thru	29	
Number of bonded IOBs	16	372

Number of warnings: 0
Total time: 4 secs

Process "Generate Post-Place & Route Static Timing" completed successfully

Verificar se as restrições foram satisfeitas

FPGA Design Summary

- [-] Design Overview
 - Summary
 - IOB Properties
 - Timing Constraints
 - Pinout Report
 - Clock Report
- [-] Errors and Warnings
 - Synthesis Messages
 - Translation Messages
 - Map Messages
 - Place and Route Messages
 - Timing Messages
 - Bitgen Messages
 - All Current Messages
- Project Properties
 - Enable Enhanced Design Summary
 - Enable Message Filtering
 - Display Incremental Messages
- Enhanced Design Summary Contents
 - Show Partition Data
 - Show Errors
 - Show Warnings
 - Show Failing Constraints
 - Show Clock Report

Total Number of 4 input LUTs	88	11,776	1%
Number used as logic	59		
Number used as a route-thru	29		
Number of bonded IOBs	16	372	4%
IOB Flip Flops	2		
Number of GCLKs	1	24	4%
Total equivalent gate count for design	817		
Additional JTAG gate count for IOBs	768		

Performance Summary			
Final Timing Score:	0	Pinout Data:	Pinout F
Routing Results:	All Signals Completely Routed	Clock Data:	Clock R
Timing Constraints:	All Constraints Met		

Detailed Reports				
Report Name	Status	Generated	Errors	Warnings
Synthesis Report	Current	Thu 6. Nov 10:04:04 2008	0	0
Translation Report	Current	Thu 6. Nov 10:45:21 2008	0	0
Map Report	Current	Thu 6. Nov 10:45:30 2008	0	0

Design Summary

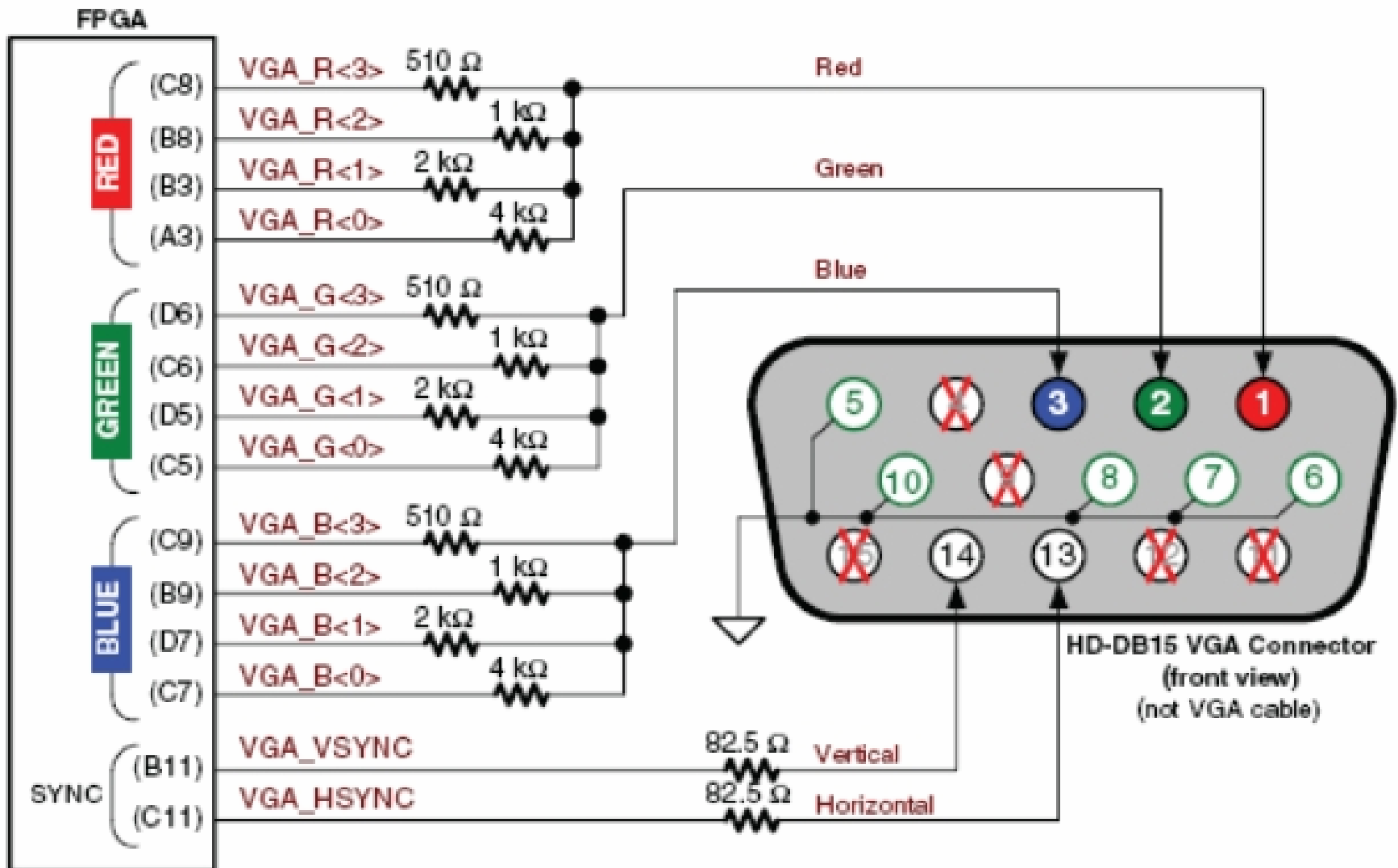
vga.vhd

vga.ucf

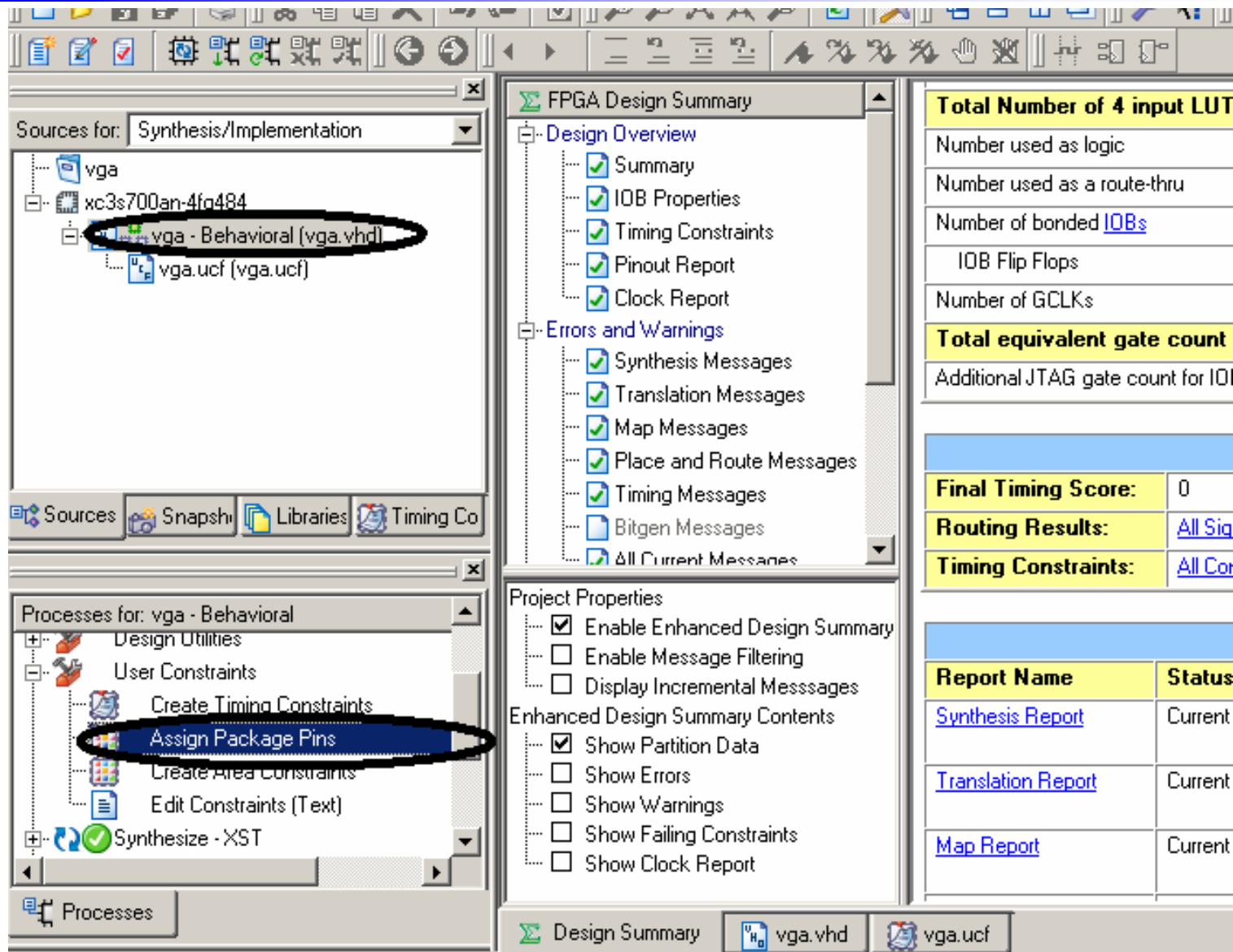
Atribuição da Pinagem

- SPARTAN 3AN
 - Clock de 50 MHz – pino **E12**.
 - Botão de Reset – pino **T15**.

Interface VGA - Pinagem



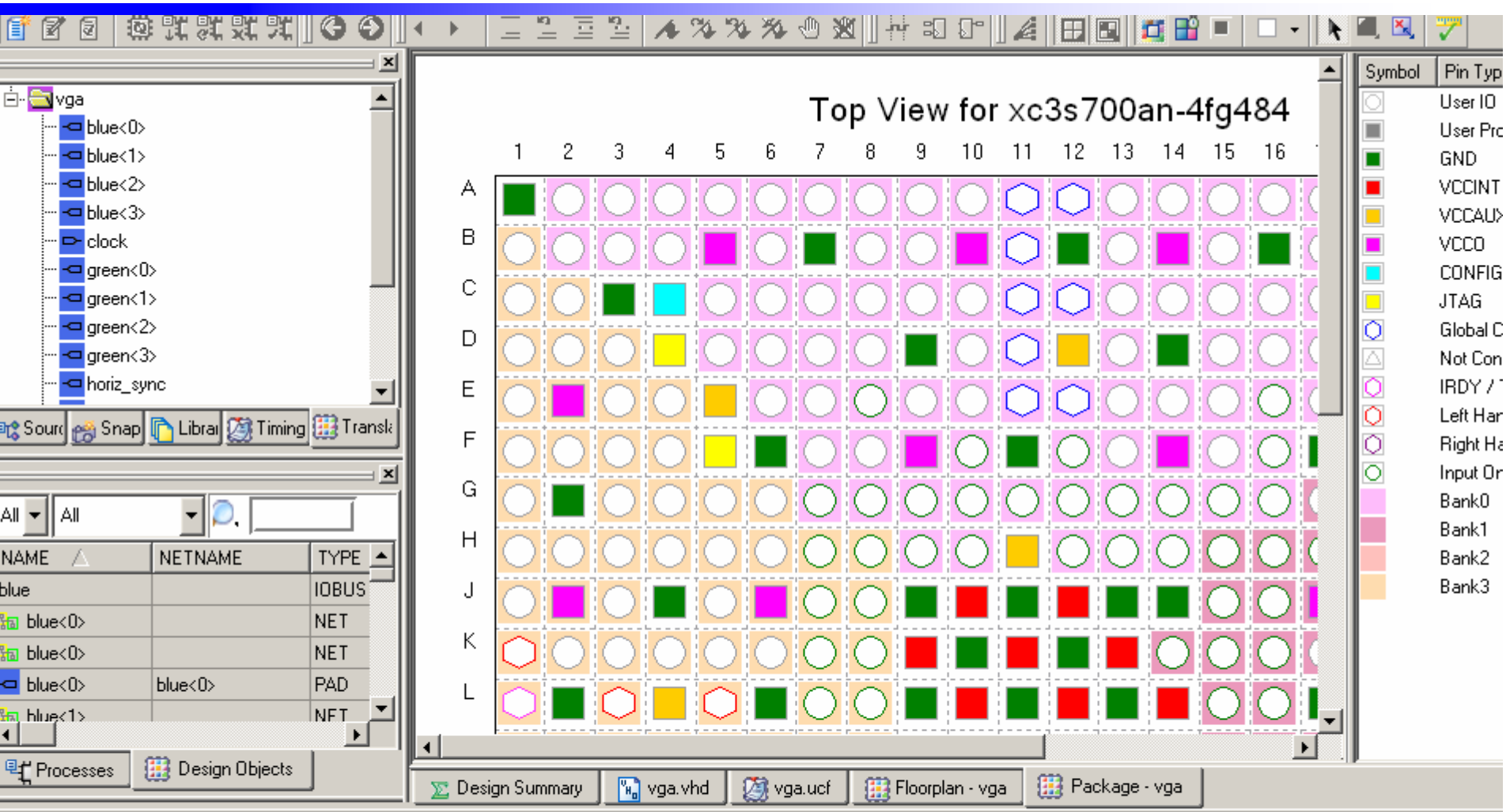
Atribuição da pinagem



The screenshot shows the Xilinx ISE software interface. In the 'Processes for: vga - Behavioral' window, the 'Assign Package Pins' process is highlighted with a red oval. The 'Sources for: Synthesis/Implementation' window shows the project files, with 'vga - Behavioral (vga.vhd)' also highlighted with a red oval. The 'FPGA Design Summary' window is open, displaying the 'Design Overview' and 'Errors and Warnings' sections. The 'Design Overview' section includes: Summary, IOB Properties, Timing Constraints, Pinout Report, Clock Report, Errors and Warnings, Synthesis Messages, Translation Messages, Map Messages, Place and Route Messages, Timing Messages, Bitgen Messages, and All Current Messages. The 'Errors and Warnings' section includes: Enable Enhanced Design Summary, Enable Message Filtering, and Display Incremental Messages. The 'Enhanced Design Summary Contents' section includes: Show Partition Data, Show Errors, Show Warnings, Show Failing Constraints, and Show Clock Report. The 'Total Number of 4 input LUT' section shows: Number used as logic, Number used as a route-thru, Number of bonded IOBs, IOB Flip Flops, and Number of GCLKs. The 'Total equivalent gate count' section shows: Additional JTAG gate count for IOI. The 'Final Timing Score' is 0. The 'Routing Results' are All Sig. The 'Timing Constraints' are All Cor. The 'Report Name' and 'Status' table is as follows:

Report Name	Status
Synthesis Report	Current
Translation Report	Current
Map Report	Current

Package – ‘nome do arquivo’



Top View for xc3s700an-4fg484

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
A	Green	White	White	White	White	White	White	White	White	White	Blue	Blue	White	White	White	White
B	White	White	White	White	Magenta	White	Green	White	White	Magenta	Blue	Green	White	Magenta	White	Green
C	White	White	White	Green	Cyan	White	White	White	White	White	Blue	Blue	White	White	White	White
D	White	White	White	White	Yellow	White	White	White	White	Green	White	Blue	Yellow	White	Green	White
E	White	White	White	White	White	White	White	White	White	White	Blue	Blue	White	White	White	White
F	White	White	White	White	White	White	White	White	White	White	Blue	Blue	White	White	White	White
G	White	Green	White	White	White	White	White	White	White	White	White	White	White	White	White	White
H	White	White	White	White	White	White	White	White	White	White	White	White	White	White	White	White
J	White	White	White	White	White	White	White	White	White	White	White	White	White	White	White	White
K	White	Magenta	White	Green	White	Magenta	White	White	Green	Red	Green	Red	Green	Green	Green	White
L	White	White	White	White	White	White	White	White	White	White	White	White	White	White	White	White

```

Compiling vhdl file "C:/Documents and Settings/Administrador/Desktop/SST/implementacao/vga/vga.vhd" in Library work
Entity <vga> compiled.
Entity <vga> (Architecture <behavioral>) compiled.
Loading device for application Rf_Device from file '3s700a.nph' in environment C:\Xilinx91i.
No DRC error found.
  
```

Configurar a pinagem e salvar o arquivo

Sources for: Synthesis/Implementation

- vga
 - xc3s700an-4fg484
 - vga - Behavioral (vga.vhd)
 - vga.ucf (vga.ucf)

Sources | Snapshots | Libraries | Timing Constraints | Translated Netlist

All | All

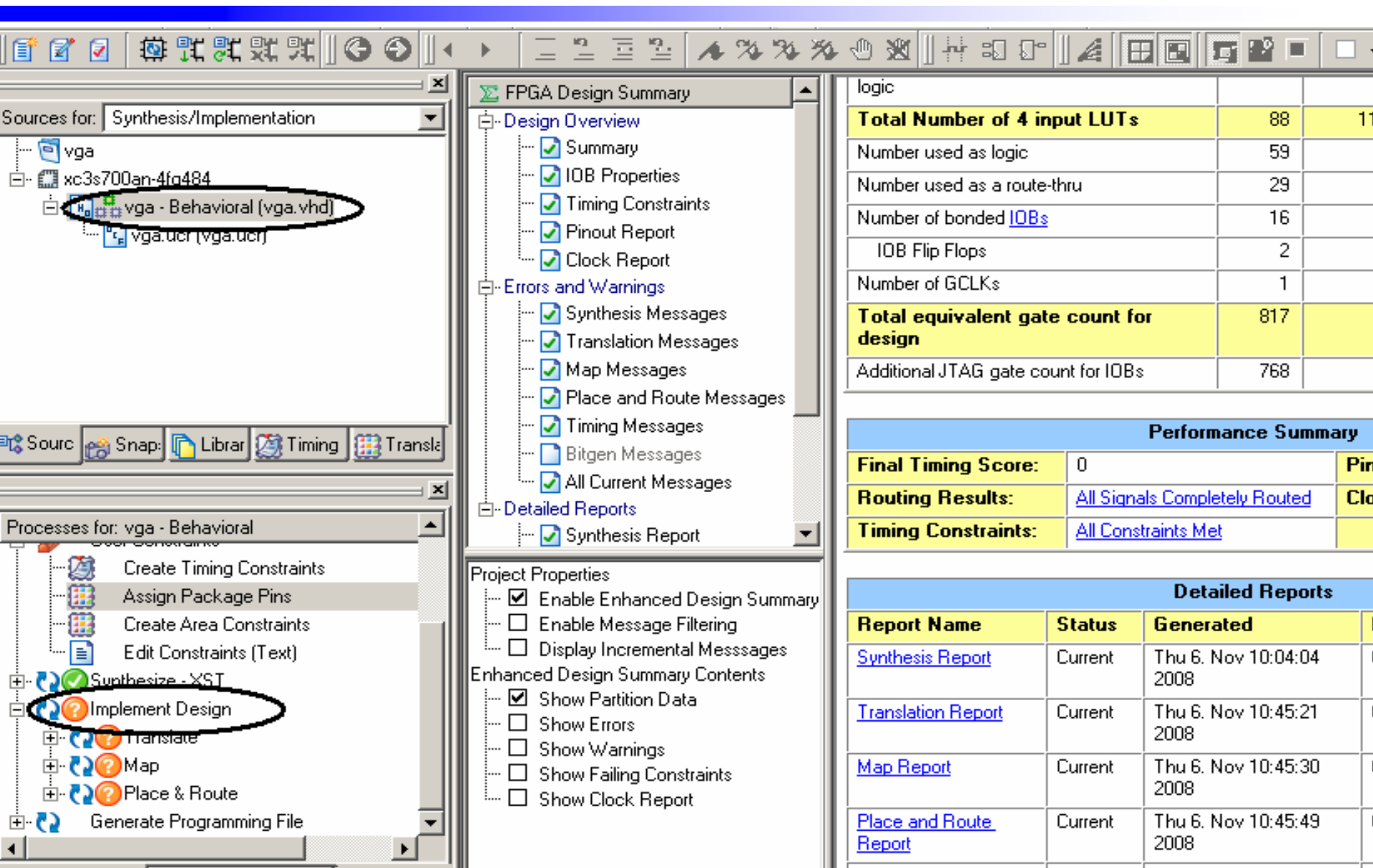
NAME	NETNAME	TYPE	IODIRECTION	LOC	BANK	IOSTANDA
blue<0>	blue<0>	PAD	Output	C7	BANK0	
blue<1>		NET	Output			
blue<1>		NET				
blue<1>	blue<1>	PAD	Output	D7	BANK0	
blue<2>		NET	Output			
blue<2>		NET				
blue<2>	blue<2>	PAD	Output	B9	BANK0	
blue<3>		NET	Output			

Processes | Design Objects

	1	2	3	4	5
A					
B					
C					
D					
E					
F					
G					
H					
J					
K					
L					
M					
N					

Design Su | vga.vhd | vga.ucf

Implementar o projeto novamente



The screenshot shows the Xilinx ISE software interface during the implementation phase. The 'Sources for: Synthesis/Implementation' window displays the project files, with 'vga - Behavioral (vga.vhd)' circled. The 'Processes for: vga - Behavioral' window shows the implementation steps, with 'Implement Design' circled. The 'FPGA Design Summary' window is open, showing the 'Design Overview' and 'Errors and Warnings' sections. The 'Project Properties' window is also visible, showing the 'Enhanced Design Summary Contents' section. The 'Performance Summary' and 'Detailed Reports' tables are displayed on the right side of the interface.

Sources for: Synthesis/Implementation

- vga
- xc3s700an-4fg484
- vga - Behavioral (vga.vhd)**
- vga.ucf (vga.ucf)

Processes for: vga - Behavioral

- Create Timing Constraints
- Assign Package Pins
- Create Area Constraints
- Edit Constraints (Text)
- Synthesize - XST
- Implement Design**
- Translate
- Map
- Place & Route
- Generate Programming File

FPGA Design Summary

- Design Overview**
 - Summary
 - IOB Properties
 - Timing Constraints
 - Pinout Report
 - Clock Report
- Errors and Warnings**
 - Synthesis Messages
 - Translation Messages
 - Map Messages
 - Place and Route Messages
 - Timing Messages
 - Bitgen Messages
 - All Current Messages
- Detailed Reports**
 - Synthesis Report

Project Properties

- Enable Enhanced Design Summary
- Enable Message Filtering
- Display Incremental Messages

Enhanced Design Summary Contents

- Show Partition Data
- Show Errors
- Show Warnings
- Show Failing Constraints
- Show Clock Report

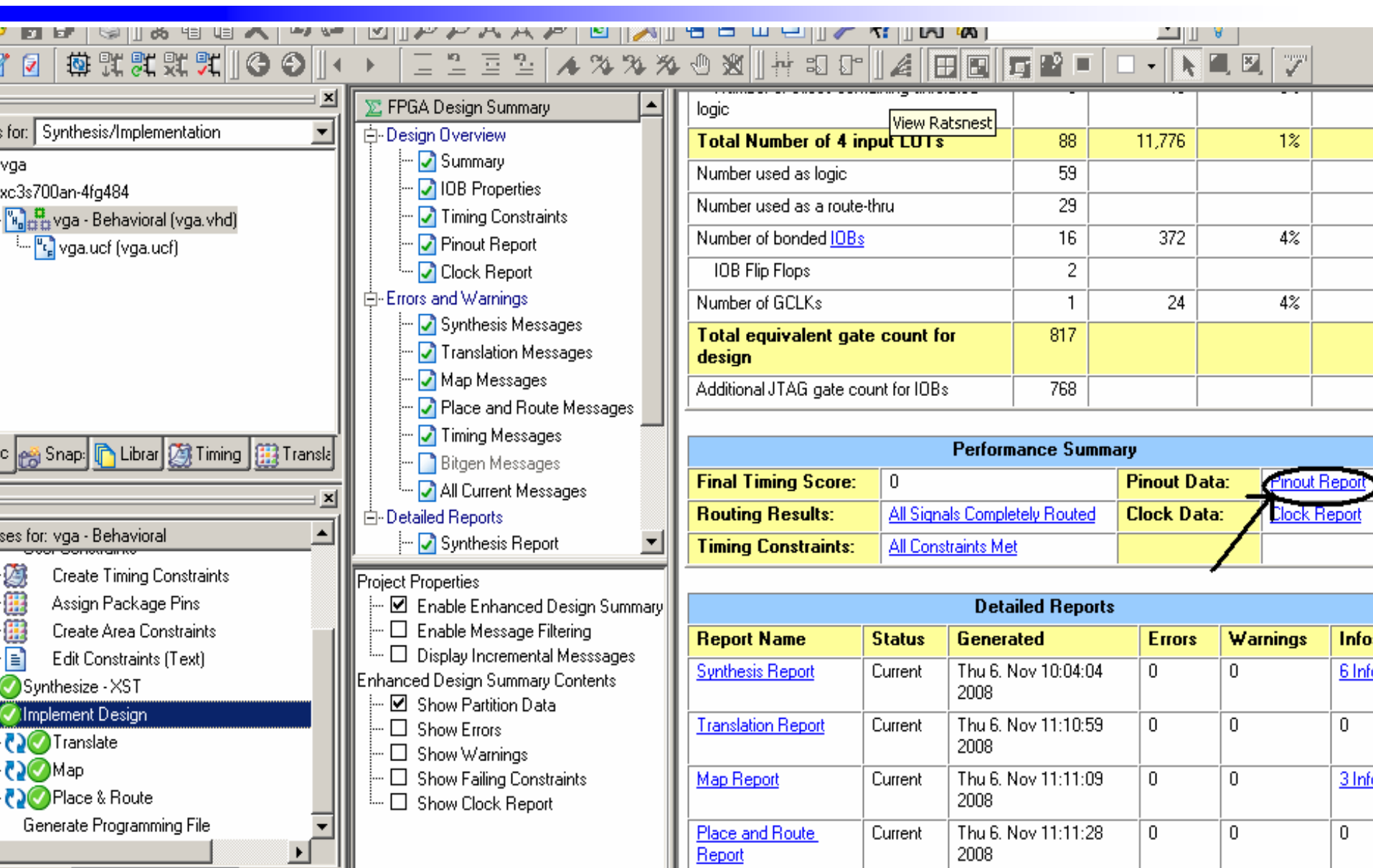
Performance Summary

logic		
Total Number of 4 input LUTs	88	11
Number used as logic	59	
Number used as a route-thru	29	
Number of bonded IOBs	16	
IOB Flip Flops	2	
Number of GCLKs	1	
Total equivalent gate count for design	817	
Additional JTAG gate count for IOBs	768	

Detailed Reports

Report Name	Status	Generated
Synthesis Report	Current	Thu 6. Nov 10:04:04 2008
Translation Report	Current	Thu 6. Nov 10:45:21 2008
Map Report	Current	Thu 6. Nov 10:45:30 2008
Place and Route Report	Current	Thu 6. Nov 10:45:49 2008

Verificar se a pinagem está correta



The screenshot shows the Xilinx ISE software interface. The main window displays the 'FPGA Design Summary' for a project named 'vga'. The 'Design Overview' section is expanded, showing various reports and messages. The 'Performance Summary' section is also visible, providing key metrics for the design.

FPGA Design Summary

- Design Overview
 - Summary
 - I/OB Properties
 - Timing Constraints
 - Pinout Report
 - Clock Report
- Errors and Warnings
 - Synthesis Messages
 - Translation Messages
 - Map Messages
 - Place and Route Messages
 - Timing Messages
 - Bitgen Messages
 - All Current Messages
- Detailed Reports
 - Synthesis Report

Performance Summary

Final Timing Score:	0	Pinout Data:	Pinout Report
Routing Results:	All Signals Completely Routed	Clock Data:	Clock Report
Timing Constraints:	All Constraints Met		

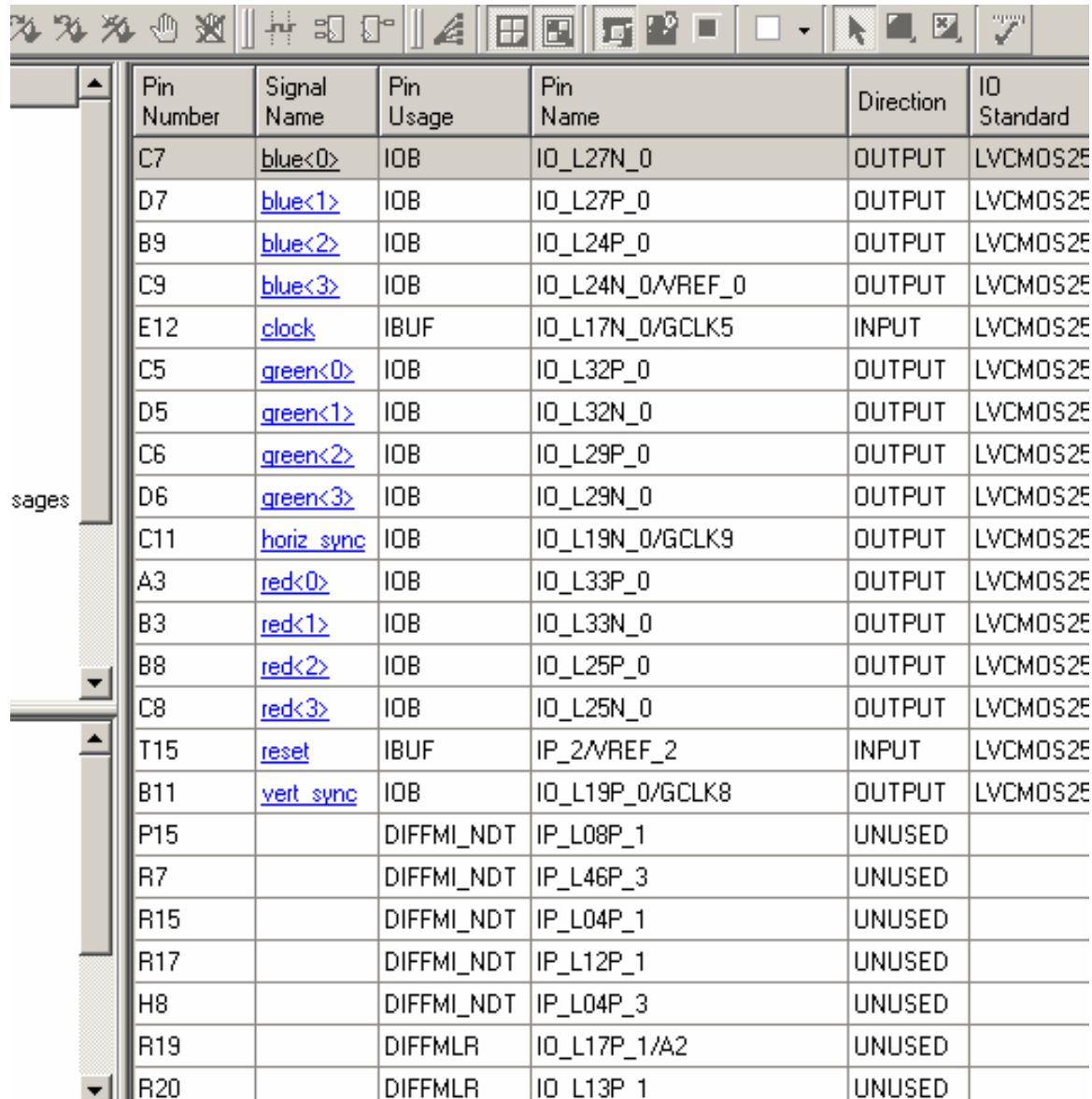
Detailed Reports

Report Name	Status	Generated	Errors	Warnings	Info
Synthesis Report	Current	Thu 6. Nov 10:04:04 2008	0	0	6 Info
Translation Report	Current	Thu 6. Nov 11:10:59 2008	0	0	0
Map Report	Current	Thu 6. Nov 11:11:09 2008	0	0	3 Info
Place and Route Report	Current	Thu 6. Nov 11:11:28 2008	0	0	0

Additional Metrics:

logic			
Total Number of 4 input LUTs	88	11,776	1%
Number used as logic	59		
Number used as a route-thru	29		
Number of bonded IOBs	16	372	4%
IOB Flip Flops	2		
Number of GCLKs	1	24	4%
Total equivalent gate count for design	817		
Additional JTAG gate count for IOBs	768		

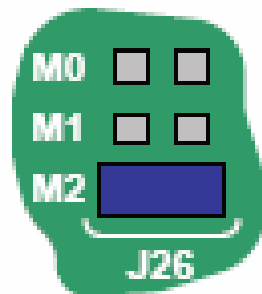
Conferindo pinagem...



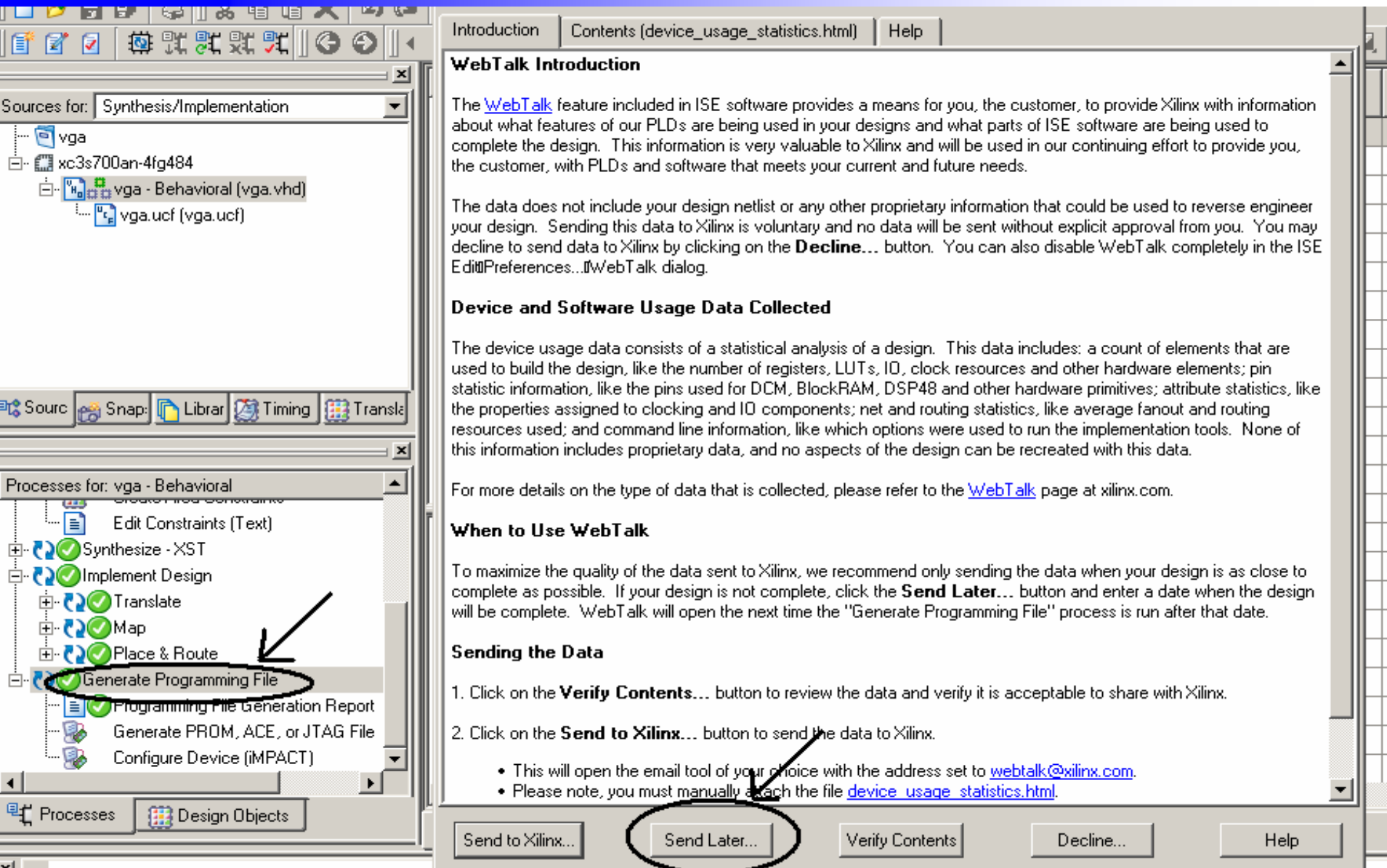
Pin Number	Signal Name	Pin Usage	Pin Name	Direction	IO Standard
C7	blue<0>	IOB	IO_L27N_0	OUTPUT	LVCMS25
D7	blue<1>	IOB	IO_L27P_0	OUTPUT	LVCMS25
B9	blue<2>	IOB	IO_L24P_0	OUTPUT	LVCMS25
C9	blue<3>	IOB	IO_L24N_0/VREF_0	OUTPUT	LVCMS25
E12	clock	IBUF	IO_L17N_0/GCLK5	INPUT	LVCMS25
C5	green<0>	IOB	IO_L32P_0	OUTPUT	LVCMS25
D5	green<1>	IOB	IO_L32N_0	OUTPUT	LVCMS25
C6	green<2>	IOB	IO_L29P_0	OUTPUT	LVCMS25
D6	green<3>	IOB	IO_L29N_0	OUTPUT	LVCMS25
C11	horiz_sync	IOB	IO_L19N_0/GCLK9	OUTPUT	LVCMS25
A3	red<0>	IOB	IO_L33P_0	OUTPUT	LVCMS25
B3	red<1>	IOB	IO_L33N_0	OUTPUT	LVCMS25
B8	red<2>	IOB	IO_L25P_0	OUTPUT	LVCMS25
C8	red<3>	IOB	IO_L25N_0	OUTPUT	LVCMS25
T15	reset	IBUF	IP_2/VREF_2	INPUT	LVCMS25
B11	vert_sync	IOB	IO_L19P_0/GCLK8	OUTPUT	LVCMS25
P15		DIFFMI_NDT	IP_L08P_1	UNUSED	
R7		DIFFMI_NDT	IP_L46P_3	UNUSED	
R15		DIFFMI_NDT	IP_L04P_1	UNUSED	
R17		DIFFMI_NDT	IP_L12P_1	UNUSED	
H8		DIFFMI_NDT	IP_L04P_3	UNUSED	
R19		DIFFMLR	IO_L17P_1/A2	UNUSED	
R20		DIFFMLR	IO_L13P_1	UNUSED	

Configuração da Placa

- Para grava o programa na memória Flash interna do componente Spartan 3AN, diretamente via JTAG USB, o Jumper J26 deve estar configurado da seguinte forma:



Programando o FPGA



The screenshot shows the Xilinx ISE software interface. On the left, the 'Sources for: Synthesis/Implementation' pane shows a project named 'vga' with files 'vga.ucf (vga.ucf)' and 'vga.vhd'. Below it, the 'Processes for: vga - Behavioral' pane shows a list of steps: Edit Constraints (Text), Synthesize - XST, Implement Design, Translate, Map, Place & Route, **Generate Programming File** (circled in red with an arrow pointing to it), Programming File Generation Report, Generate PROM, ACE, or JTAG File, and Configure Device (iMPACT). At the bottom of the interface, the 'Processes' pane shows 'Design Objects'.

The main window displays the 'WebTalk Introduction' dialog. The title bar includes 'Introduction', 'Contents (device_usage_statistics.html)', and 'Help'. The dialog text reads:

WebTalk Introduction

The [WebTalk](#) feature included in ISE software provides a means for you, the customer, to provide Xilinx with information about what features of our PLDs are being used in your designs and what parts of ISE software are being used to complete the design. This information is very valuable to Xilinx and will be used in our continuing effort to provide you, the customer, with PLDs and software that meets your current and future needs.

The data does not include your design netlist or any other proprietary information that could be used to reverse engineer your design. Sending this data to Xilinx is voluntary and no data will be sent without explicit approval from you. You may decline to send data to Xilinx by clicking on the **Decline...** button. You can also disable WebTalk completely in the ISE Edit Preferences.../WebTalk dialog.

Device and Software Usage Data Collected

The device usage data consists of a statistical analysis of a design. This data includes: a count of elements that are used to build the design, like the number of registers, LUTs, IO, clock resources and other hardware elements; pin statistic information, like the pins used for DCM, BlockRAM, DSP48 and other hardware primitives; attribute statistics, like the properties assigned to clocking and IO components; net and routing statistics, like average fanout and routing resources used; and command line information, like which options were used to run the implementation tools. None of this information includes proprietary data, and no aspects of the design can be recreated with this data.

For more details on the type of data that is collected, please refer to the [WebTalk](#) page at xilinx.com.

When to Use WebTalk

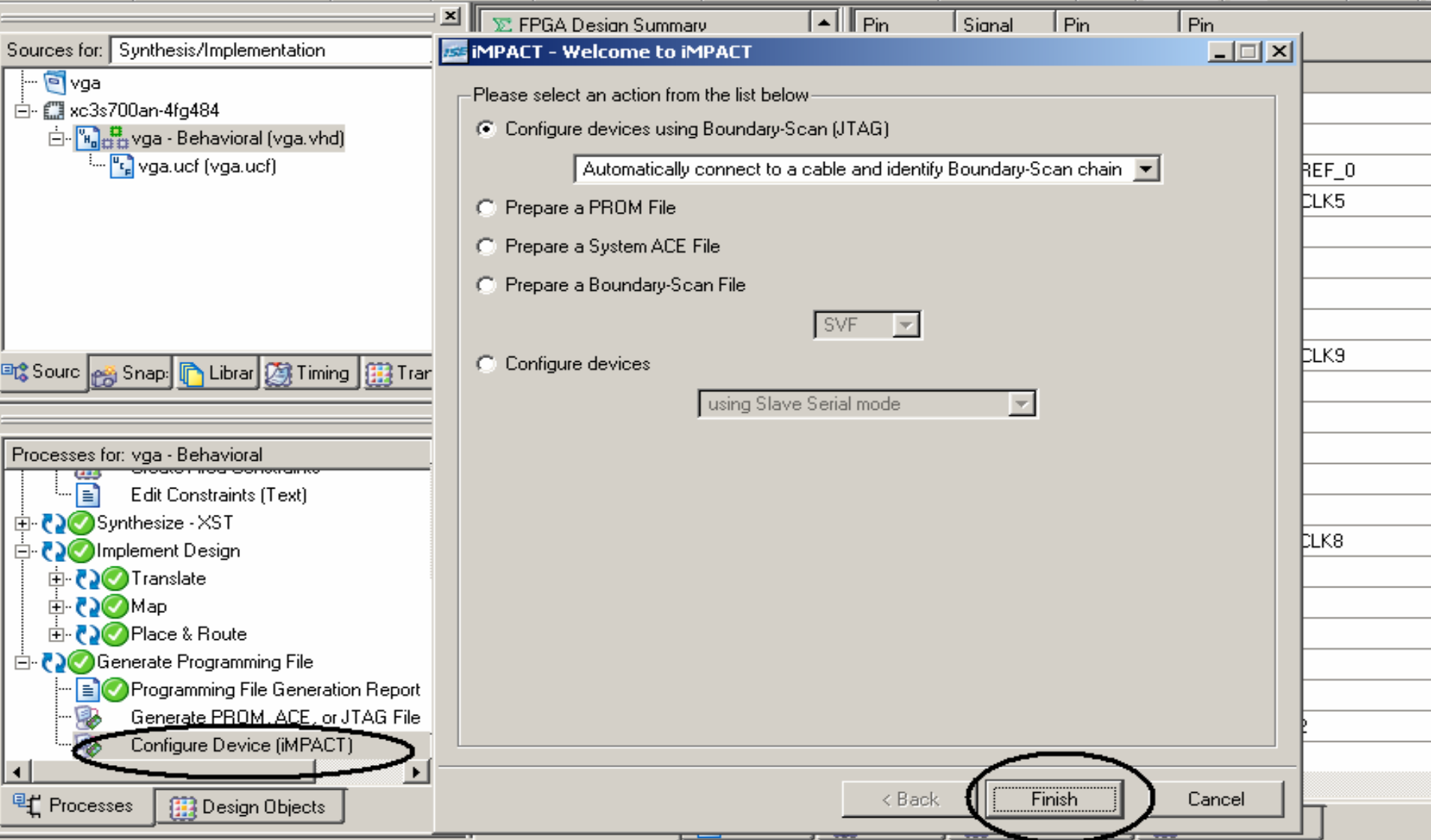
To maximize the quality of the data sent to Xilinx, we recommend only sending the data when your design is as close to complete as possible. If your design is not complete, click the **Send Later...** button and enter a date when the design will be complete. WebTalk will open the next time the "Generate Programming File" process is run after that date.

Sending the Data

1. Click on the **Verify Contents...** button to review the data and verify it is acceptable to share with Xilinx.
2. Click on the **Send to Xilinx...** button to send the data to Xilinx.
 - This will open the email tool of your choice with the address set to webtalk@xilinx.com.
 - Please note, you must manually attach the file [device_usage_statistics.html](#).

At the bottom of the dialog, there are five buttons: 'Send to Xilinx...', 'Send Later...' (circled in red with an arrow pointing to it), 'Verify Contents', 'Decline...', and 'Help'.

Gerando arquivo binário

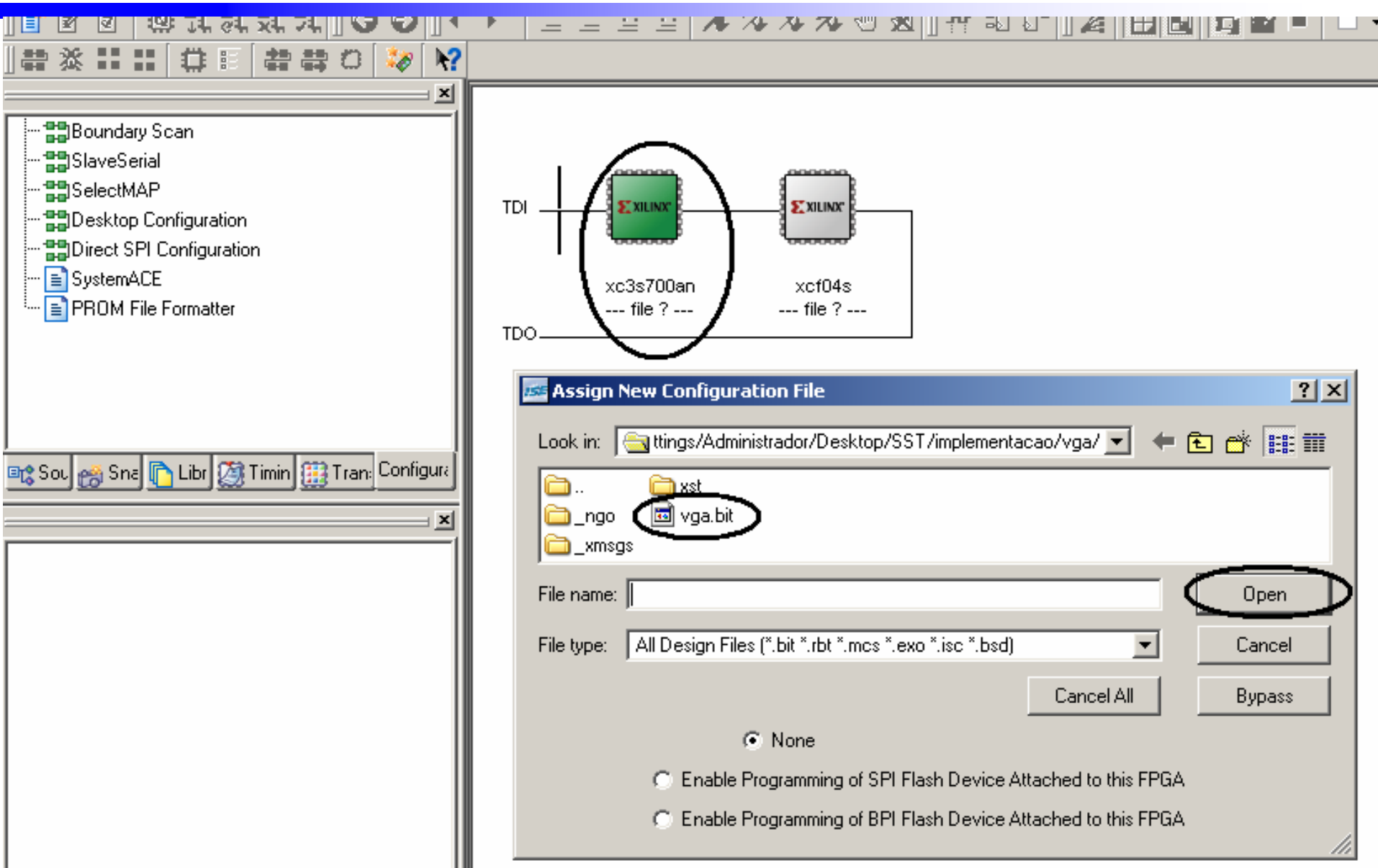


The screenshot displays the Xilinx ISE software interface. On the left, the 'Sources for: Synthesis/Implementation' pane shows a project named 'vga' with files 'vga - Behavioral (vga.vhd)' and 'vga.ucf (vga.ucf)'. Below it, the 'Processes for: vga - Behavioral' pane shows a list of steps: 'Edit Constraints (Text)', 'Synthesize - XST', 'Implement Design', 'Translate', 'Map', 'Place & Route', 'Generate Programming File', 'Programming File Generation Report', 'Generate PROM, ACE, or JTAG File', and 'Configure Device (iMPACT)'. The 'Configure Device (iMPACT)' step is circled in black. The main window shows the 'iMPACT - Welcome to iMPACT' dialog box. It contains the following options:

- Configure devices using Boundary-Scan (JTAG)
 - Automatically connect to a cable and identify Boundary-Scan chain
- Prepare a PROM File
- Prepare a System ACE File
- Prepare a Boundary-Scan File

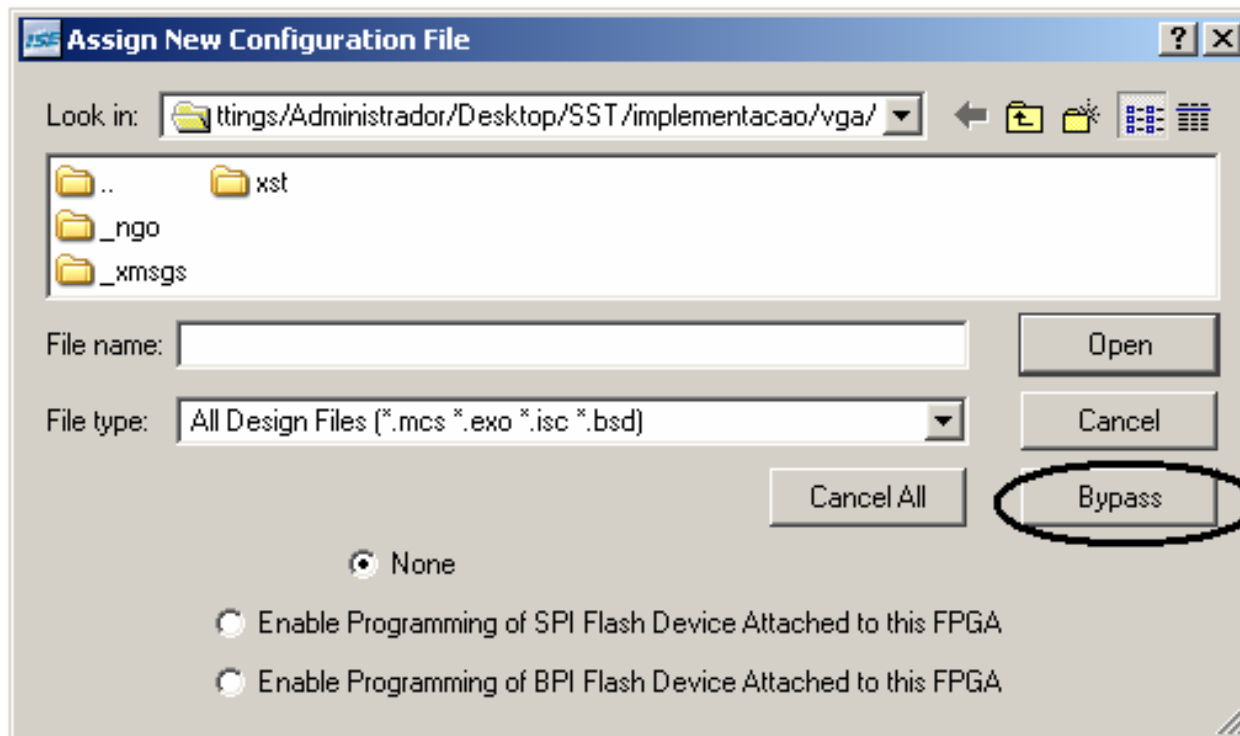
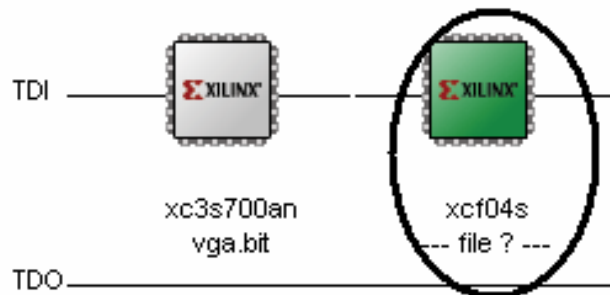
Below these options, there is a dropdown menu set to 'SVF'. At the bottom of the dialog, there is another dropdown menu set to 'using Slave Serial mode'. The 'Finish' button is circled in black.

Carregando arquivo .bit no FPGA

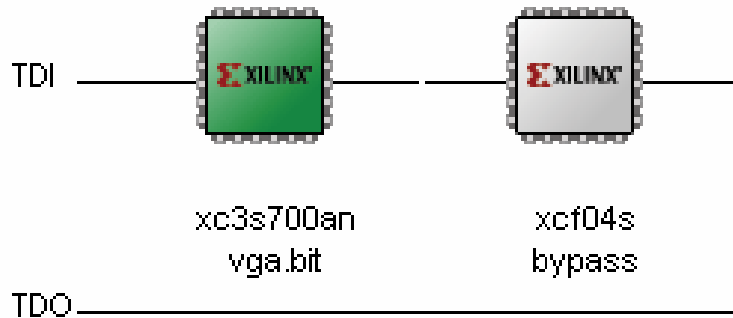


The screenshot displays the Xilinx ISE software interface. On the left, a project tree lists various configuration options: Boundary Scan, SlaveSerial, SelectMAP, Desktop Configuration, Direct SPI Configuration, SystemACE, and PROM File Formatter. The main workspace shows a schematic diagram of two Xilinx FPGAs connected in series. The first device is labeled 'xc3s700an' and the second is 'xcf04s'. Both are currently set to '--- file ? ---'. A black oval highlights the 'xc3s700an' device. Below the schematic, an 'Assign New Configuration File' dialog box is open. The 'Look in:' field shows the path 'ttings/Administrador/Desktop/SST/implementacao/vga/'. The file list contains folders for '..', '_ngo', and '_xmsgs', and a file named 'vga.bit' which is circled in black. The 'File name:' field is empty, and the 'File type:' dropdown is set to 'All Design Files (*.bit *.rft *.mcs *.exo *.isc *.bsd)'. The 'Open' button is circled in black. At the bottom of the dialog, there are radio buttons for 'None', 'Enable Programming of SPI Flash Device Attached to this FPGA', and 'Enable Programming of BPI Flash Device Attached to this FPGA'.

Bypass na memória xcf04s

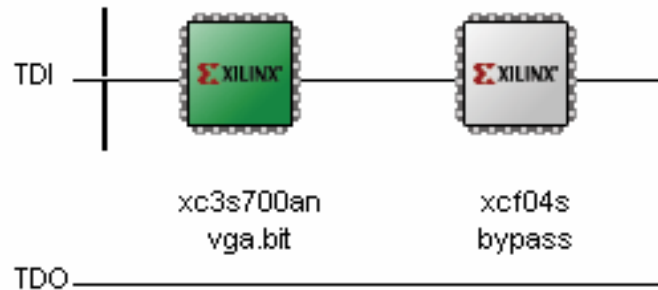


Programando o componente



- Program...
- Verify
- Erase...
- Blank Check
- Get Device ID
- Assign New Configuration File...

FPGA programado!



Program Succeeded