Circuitos Digitais II - 6882

André Barbosa Verona
Nardênio Almeida Martins

Universidade Estadual de Maringá
Departamento de Informática

Bacharelado em Ciência da Computação

Aula de Hoje

Projetos de Circuitos Combinacionais

Circuitos Combinacionais

Projetos de Circuitos Combinacionais

Projeto:

- Projete um circuito para controlar o Sistema de Intercomunicação do prédio da Reitoria da UEM (Universidade Estadual de Morangueira). O sistema deve obedecer a uma ordem de prioridades:

  1º Reitor

  2º Vice-Reitor

  3º Assessor para Assuntos Aleatórios

  4º Secretária

- Caso ocorram duas ou mais chamadas simultaneamente, somente uma chamada será atendida, a de maior prioridade. Faça o diagrama de portas lógicas do circuito e simplifique se possível.

Circuitos Combinacionais

Projetos de Circuitos Combinacionais

Projeto:

Nomenclatura das Entrada:	Convenções:
1º RE	-Presença de Chamada = 1
2º VR	-Ausência de Chamada = 0
3º AS	-Saídas: S_{RE}, S_{VR}, S_{AS}, S_{SE}
4º SE	-Chamada liberada $\Rightarrow$ S=1
	-Chamada bloqueada $\Rightarrow$ S=0

Circuitos Combinacionais

Projetos de Circuitos Combinacionais

Projeto:

RE	VR	AS	SE	S_{RE}	S_{VR}	S_{AS}	S_{SE}
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	0
0	1	0	0	0	1	0	0
0	1	0	1	0	1	0	0
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	0
1	0	0	0	1	0	0	0
1	0	0	1	1	0	0	0
1	0	1	0	1	0	0	0
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	1	0	0	0

Sem chamadas

Libera chamada da Secretária $S_{SE}=\overline{RE}.\overline{VR}.\overline{AS}.SE$

Libera chamada do Assessor $S_{AS}=\overline{RE}.\overline{VR}.AS$

Libera chamada do Vice-Reitor $S_{VR}=\overline{RE}.VR$

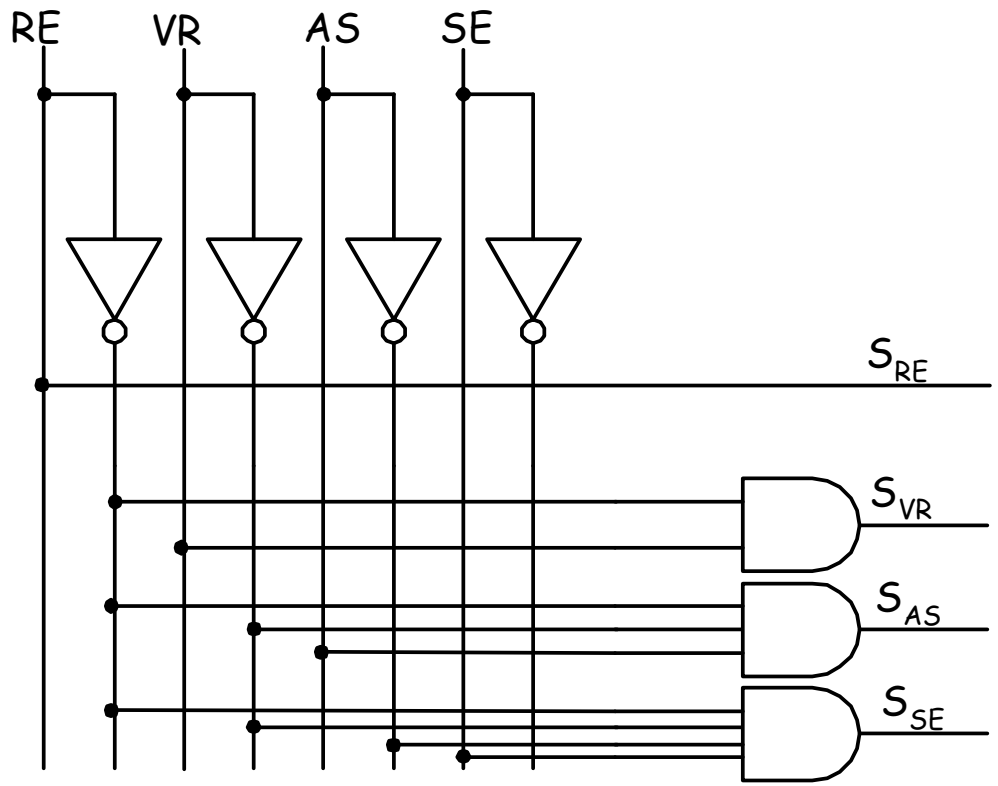
Libera chamada do Reitor $S_{RE}=RE$

Circuitos Combinacionais

Projetos de Circuitos Combinacionais

Projeto:

Circuito de Controle



Características de Projeto

Estrutura básica de um código em VHDL

LIBRARY IEEE; **USE IEEE.STD_LOGIC_1164.all;** **USE IEEE.STD_LOGIC_UNSIGNED.all;**	LIBRARY (PACOTES)
ENTITY exemplo **IS** **PORT** (<descrição dos pinos de I/O>); **END** exemplo;	ENTITY (PINOS DE I/O)
ARCHITECTURE teste **OF** exemplo **IS** **BEGIN** ... **END** teste;	ARCHITECTURE (ARQUITETURA)

VHDL – Comandos Condicionais

Arquitetura Comportamental - Comando Concorrente WHEN ELSE

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

ENTITY lig_tel_when IS
    PORT(ch : IN BIT_VECTOR(3 DOWNTO 0);              -- RE, VR, AS, SE
      saidas : OUT BIT_VECTOR(3 DOWNTO 0));           -- SRE, SVR, SAS, SSE
END lig_tel_when;

ARCHITECTURE teste OF lig_tel_when IS
BEGIN
  saidas <= "0001" WHEN ch = "0001" ELSE
            "0010" WHEN ch = "0010" ELSE
            "0010" WHEN ch = "0011" ELSE
            "0100" WHEN ch = "0100" ELSE
            "0100" WHEN ch = "0101" ELSE
            "0100" WHEN ch = "0110" ELSE
            "0100" WHEN ch = "0111" ELSE
            "1000" WHEN ch = "1000" ELSE
            "1000" WHEN ch = "1001" ELSE
            "1000" WHEN ch = "1010" ELSE
            "1000" WHEN ch = "1011" ELSE
            "1000" WHEN ch = "1100" ELSE
            "1000" WHEN ch = "1101" ELSE
            "1000" WHEN ch = "1110" ELSE
            "1000" WHEN ch = "1111" ELSE
             "0000";
END teste;
```

VHDL – Comandos Condicionais

Arquitetura Comportamental - Comando Concorrente WITH SELECT WHEN

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

ENTITY lig_tel_with IS
    PORT(ch : IN BIT_VECTOR(3 DOWNTO 0);          -- RE, VR, AS, SE
      saidas : OUT BIT_VECTOR(3 DOWNTO 0));       -- SRE, SVR, SAS, SSE
END lig_tel_with;

ARCHITECTURE teste OF lig_tel_with IS
BEGIN
          WITH ch SELECT
                    saidas <= "0000" WHEN "0000",
                              "0001" WHEN "0001",
                              "0010" WHEN "0010",
                               "0010" WHEN "0011",
                              "0100" WHEN "0100",
                              "0100" WHEN "0101",
                               "0100" WHEN "0110",
                               "0100" WHEN "0111",
                               "1000" WHEN "1000",
                               "1000" WHEN "1001",
                               "1000" WHEN "1010",
                               "1000" WHEN "1011",
                               "1000" WHEN "1100",
                               "1000" WHEN "1101",
                               "1000" WHEN "1110",
                               "1000" WHEN "1111";

END teste;
```

VHDL – Comandos Condicionais

Arquitetura Comportamental - Comando Sequencial IF THEN ELSE

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

ENTITY lig_tel_if IS
  PORT(RE, VR, AS, SE : IN BIT;
   SRE, SVR, SAS, SSE : OUT BIT);
END lig_tel_if;

ARCHITECTURE teste OF lig_tel_if IS
BEGIN
  PROCESS(RE, VR, AS, SE)
  BEGIN
        IF RE = '1' THEN
                SRE <= '1';
        ELSE
                SRE <= '0';
        END IF;
        IF RE = '0' AND VR = '1' THEN
                SVR <= '1';
        ELSE
                SVR <= '0';
        END IF;
```

-- continuacao

VHDL – Comandos Condicionais

Arquitetura Comportamental - Comando Sequencial IF THEN ELSE

```
                                                    -- continuacao
        IF RE = '0' AND VR = '0' AND AS = '1' THEN
                SAS <= '1';
        ELSE
                SAS <= '0';
        END IF;
        IF RE = '0' AND VR = '0' AND AS = '0' AND SE = '1' THEN
                SSE <= '1';
        ELSE
                SSE <= '0';
        END IF;
  END PROCESS;
END teste;
```

VHDL – Comandos Condicionais

Arquitetura Comportamental - Comando Sequencial CASE WHEN

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

ENTITY lig_tel_case IS
    PORT(ch : IN BIT_VECTOR(3 DOWNTO 0);                    -- RE, VR, AS, SE
      saidas : OUT BIT_VECTOR(3 DOWNTO 0));                 -- SRE, SVR, SAS, SSE
END lig_tel_case;

ARCHITECTURE teste OF lig_tel_case IS
BEGIN
  PROCESS (ch)
  BEGIN
          CASE ch IS
                    WHEN "0000" => saidas <= "0000";
                    WHEN "0001" => saidas <= "0001";
                    WHEN "0010" => saidas <= "0010";
                    WHEN "0011" => saidas <= "0010";
                    WHEN "0100" => saidas <= "0100";
                    WHEN "0101" => saidas <= "0100";
                    WHEN "0110" => saidas <= "0100";
                    WHEN "0111" => saidas <= "0100";
```

-- continuacao

VHDL – Comandos Condicionais

Arquitetura Comportamental - Comando Sequencial CASE WHEN

```
                                                    -- continuacao
                    WHEN "1000" => saidas <= "1000";
                    WHEN "1001" => saidas <= "1000";
                    WHEN "1010" => saidas <= "1000";
                    WHEN "1011" => saidas <= "1000";
                    WHEN "1100" => saidas <= "1000";
                    WHEN "1101" => saidas <= "1000";
                    WHEN "1110" => saidas <= "1000";
                    WHEN "1111" => saidas <= "1000";
        END CASE;
  END PROCESS;
END teste;
```

VHDL – Comandos Condicionais

Arquitetura Estrutural – Uso de Componentes

Criação do Componente → not_1

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

ENTITY not_1 IS
    PORT (x : IN bit;
          z : OUT bit);
END not_1;

ARCHITECTURE logica1 OF not_1 IS
BEGIN
  z <= not x;
END logica1;
```

VHDL – Comandos Condicionais

Arquitetura Estrutural – Uso de Componentes

Criação do Componente → and_2

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

ENTITY and_2 IS
      PORT (x, y : IN bit;
                z : OUT bit);
END and_2;

ARCHITECTURE logica2 OF and_2 IS
BEGIN
  z <= x and y;
END logica2;
```

VHDL – Comandos Condicionais

Arquitetura Estrutural – Uso de Componentes

Declaração e Instanciação dos Componentes → not_1 e and_2

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

ENTITY lig_tel_est1 IS
PORT (RE, VR, AS, SE : IN bit;
 SRE, SVR, SAS, SSE : OUT bit);
END lig_tel_est1;

ARCHITECTURE teste OF lig_tel_est1 IS

COMPONENT not_1
  PORT(x : IN bit ;
       z : OUT bit) ;
END COMPONENT ;
COMPONENT and_2
  PORT(x : IN bit ;
       y : IN bit ;
       z : OUT bit) ;
END COMPONENT ;

SIGNAL t0, t1, t2, t3, t4 : bit;
```


-- continuacao

VHDL – Comandos Condicionais

Arquitetura Estrutural – Uso de Componentes

Declaração e Instanciação dos Componentes → not_1 e and_2

```
                                                    -- continuacao
BEGIN
    P1: not_1 PORT MAP (RE, t0);
    P2: not_1 PORT MAP (t0, SRE);
    P3: not_1 PORT MAP (VR, t1);
    P4: not_1 PORT MAP (AS, t2);
    P5: and_2 PORT MAP (t0, VR, SVR);
    P6: and_2 PORT MAP (t0, t1, t3);
    P7: and_2 PORT MAP (t3, AS, SAS);
    P8: and_2 PORT MAP (t3, t2, t4);
    P9: and_2 PORT MAP (t4, SE, SSE);
END teste;
```

VHDL – Comandos Condicionais

Arquitetura Estrutural – Uso de Pacotes

Criação do Componente → not_1

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

ENTITY not_1 IS
    PORT (x : IN bit;
          z : OUT bit);
END not_1;

ARCHITECTURE logica1 OF not_1 IS
BEGIN
  z <= not x;
END logica1;
```

VHDL – Comandos Condicionais

Arquitetura Estrutural – Uso de Pacotes

Criação do Componente → and_2

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

ENTITY and_2 IS
      PORT (x, y : IN bit;
                z : OUT bit);
END and_2;

ARCHITECTURE logica2 OF and_2 IS
BEGIN
  z <= x and y;
END logica2;
```

VHDL – Comandos Condicionais

Arquitetura Estrutural – Uso de Pacotes

Criação do Pacote: Declaração dos Componentes → not_1 e and_2

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

PACKAGE lig_tel_est2_package IS

COMPONENT not_1
  PORT(x : IN bit;
       z : OUT bit);
END COMPONENT;
COMPONENT and_2
  PORT(x : IN bit;
       y : IN bit;
       z : OUT bit);
END COMPONENT;

END lig_tel_est2_package;
```

VHDL – Comandos Condicionais

Arquitetura Estrutural – Uso de Pacotes

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
-- LIBRARY work;
-- USE work.all;
USE work.lig_tel_est2_package.all ;

ENTITY lig_tel_est2 IS
PORT (RE, VR, AS, SE : IN bit;
 SRE, SVR, SAS, SSE : OUT bit);
END lig_tel_est2;

ARCHITECTURE teste OF lig_tel_est2 IS

SIGNAL t0, t1, t2, t3, t4: bit;
```

-- continuacao

VHDL – Comandos Condicionais

Arquitetura Estrutural – Uso de Pacotes

```
                                        -- continuacao
BEGIN
    P1: not_1 PORT MAP (RE, t0);
    P2: not_1 PORT MAP (t0, SRE);
    P3: not_1 PORT MAP (VR, t1);
    P4: not_1 PORT MAP (AS, t2);
    P5: and_2 PORT MAP (t0, VR, SVR);
    P6: and_2 PORT MAP (t0, t1, t3);
    P7: and_2 PORT MAP (t3, AS, SAS);
    P8: and_2 PORT MAP (t3, t2, t4);
    P10: and_2 PORT MAP (t4, SE, SSE);
END teste;
```

VHDL – Comandos Condicionais

Arquitetura ou Modelagem Estrutural

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

ENTITY lig_tel_est IS
  PORT(RE, VR, AS, SE : IN BIT;
   SRE, SVR, SAS, SSE : OUT BIT);
END lig_tel_est;

ARCHITECTURE teste OF lig_tel_est IS
SIGNAL t0, t1, t2 : BIT;
BEGIN
  SRE <= RE;
  t0  <= NOT(RE);
  SVR <= t0 AND VR;
  t1  <= NOT(VR);
  SAS <= t0 AND t1 AND AS;
  t2  <= NOT(AS);
  SSE <= t0 AND t1 AND t2 AND SE;
END teste;
```

VHDL – Comandos Condicionais

Arquitetura ou Modelagem por Fluxo de Dados

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

ENTITY lig_tel_fd IS
  PORT(RE, VR, AS, SE : IN BIT;
   SRE, SVR, SAS, SSE : OUT BIT);
END lig_tel_fd;

ARCHITECTURE teste OF lig_tel_fd IS
BEGIN
  SRE <= RE;
  SVR <= NOT(RE) AND VR;
  SAS <= NOT(RE) AND NOT(VR) AND AS;
  SSE <= NOT(RE) AND NOT(VR) AND NOT(AS) AND SE;
END teste;
```