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 $S_{VR}=\overline{RE}.VR$

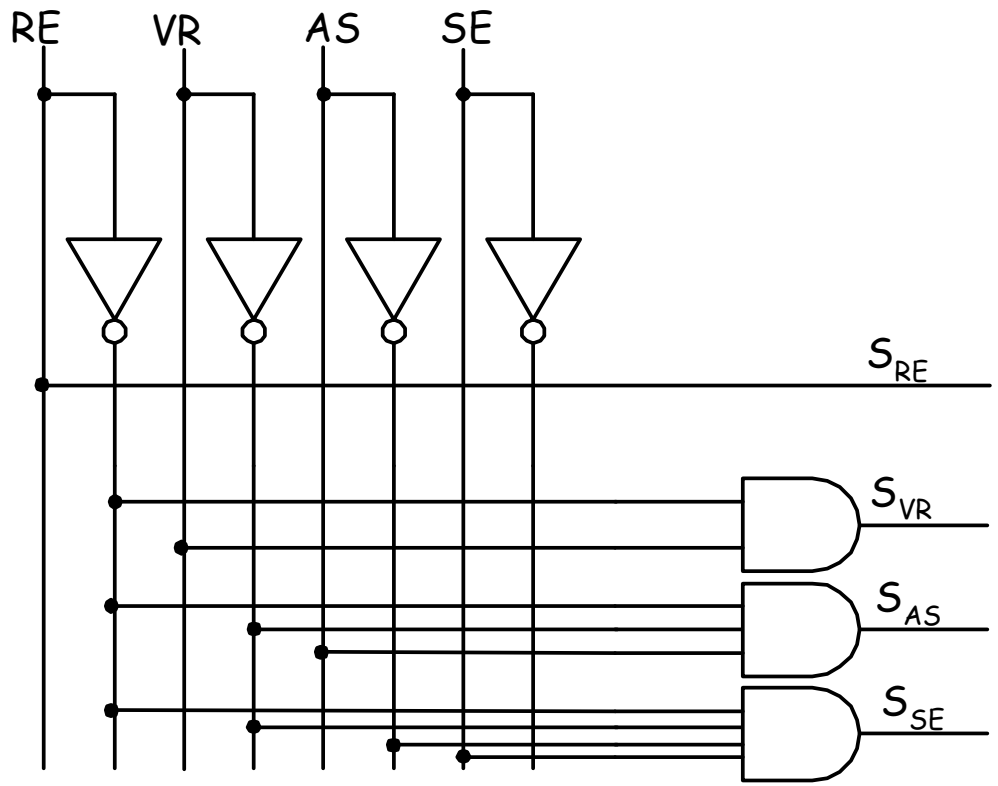
Libera chamada do Reitor $S_{RE}=RE$

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

WHEN ELSE → Arquitetura ou Modelagem por Fluxo de Dados

É um comando concorrente

Restrição de uso dentro de procedimentos, funções e processos

Transfere o valor de uma expressão para um sinal destino caso uma determinada condição seja satisfeita

Sintaxe

```
sinal_destino <= expressao_1 WHEN condiçao_1 ELSE
                 expressao_2 WHEN condiçao_2 ELSE
                 expressao_3 WHEN condiçao_3 ELSE
                 expressao_4;
```

Nota: O comando condicional WHEN ELSE é útil para expressar funções lógicas em forma de tabela verdade

VHDL – Comandos Condicionais

WITH SELECT WHEN → Arquitetura ou Modelagem por Fluxo de Dados

É um comando concorrente.

Transfere um valor a um sinal de destino segundo uma relação de opções.

Todas as condições de seleção devem ser consideradas e elas devem ser mutuamente exclusivas.

A lista de opções nesta construção não contém uma prioridade.

Sintaxe:

```
WITH expressao_de_escolha SELECT                    -- expressao_de_escolha =
  sinal_destino <= expressao_a WHEN condicao_1,     -- condicao_1
                   expressao_b WHEN condicao_2,     -- condicao_2
                   expressao_c WHEN condicao_3 | condicao_4, -- condicao_3 ou condicao_4
                   expressao_d WHEN condicao_5 TO condicao_9, -- condicao_5 ate condicao_9
                   expressao_e WHEN OTHERS;         -- condicoes restantes
```

NOTA: O delimitador | equivale a uma operação OU entre as condições de escolha. As palavras reservadas TO e DOWNTO servem para delimitar uma faixa de condições. A palavra reservada OTHERS na última condição serve para agrupar as condições não-relacionadas na lista.

Características de Projeto

Arquitetura ou Modelagem Comportamental

Declaração de um processo

```
. . .
ARCHITECTURE nome_da_arquitetura OF nome_da_entidade IS
BEGIN
        PROCESS (lista_de_sensibilidade)
        BEGIN
                . . .
                comandos;
                . . .
        END PROCESS;
END nome_da_arquitetura;
```

VHDL – Comandos Condicionais

IF THEN ELSE → Arquitetura ou Modelagem Comportamental

É um comando sequencial

Utilizado na descrição comportamental de componentes → Utilizado em procedimentos, funções e processos.

Transfere o valor de uma expressão para um sinal destino caso uma determinada condição seja satisfeita

Sintaxe →

```
IF condicao_1 THEN
        comando_sequencial;
ELSIF condicao_2 THEN                    -- Clausula ELSIF opcional
        comando_sequencial;
ELSIF condicao_3 THEN
        comando_sequencial;
ELSE                                     -- Clausula ELSE opcional
        comando_sequencial;
END IF;
```

VHDL – Comandos Condicionais

CASE WHEN → Arquitetura ou Modelagem Comportamental

É um comando sequencial com uso dentro de procedimentos, funções e processos.

Permite a definição de várias condições em um componente.

Neste comando, as comparações sempre são feitas em torno de um único objeto ou expressão, e será o valor desse objeto ou determinada condição que indicará quais comandos serão executados.

Sintaxe:

```
CASE expressao_de_escolha IS                          -- expressao_de_escolha =
  WHEN condicao_1                  => comando_a;          -- condicao_1
  WHEN condicao_2                  => comando_b; comando_c;-- condicao_2
  WHEN condicao_3 | condicao_4     => comando_d;          -- condicao_3 ou condicao_4
  WHEN condicao_5 TO condicao_9    => comando_d;          -- condicao_5 ate condicao_9
  WHEN OTHERS                      => comando_e; comando_f;-- condicoes restantes
END CASE;
```

NOTA: O delimitador | equivale a uma operação OU entre as condições de escolha. As palavras reservadas TO e DOWNTO servem para delimitar uma faixa de condições. A palavra reservada OTHERS na última condição serve para agrupar as condições não-relacionadas na lista.

Características de Projeto

Arquitetura ou Modelagem Estrutural usando Componente

Declaração e instanciação de um componente

```
. . .
ARCHITECTURE nome_da_arquitetura OF nome_da_entidade IS
COMPONENT nome_do_componente                    -- declaracao do componente
        PORT (lista_de_porta_de_interface);
END nome_do_componente;
SIGNAL  lista_de_sinal : tipo_de_dado;
BEGIN
        nome_da_instanciacao : nome_componente PORT MAP
        (lista_de_associacao_de_porta);         -- instanciacao do componente
END nome_da_arquitetura;
```

Nota: A declaração do componente é similar à declaração de entidade.

Características de Projeto

Arquitetura ou Modelagem Estrutural usando Pacote

Criação de um pacote

```
. . .
PACKAGE nome_do_pacote IS
COMPONENT nome_do_componente                    -- declaracao do componente
        PORT (lista_de_porta_de_interface);
END nome_do_componente;
END nome_do_pacote;
. . .
```

Nota: A declaração do componente é similar à declaração de entidade.

Características de Projeto

Arquitetura ou Modelagem Estrutural usando Pacote

Declaração de biblioteca de trabalho e de pacote, bem como de instanciação de um componente

```
LIBRARY work;                          -- declaracao da biblioteca de trabalho e do pacote
-- USE work.all;
USE work.nome_do_pacote.all ;
. . .
ARCHITECTURE nome_da_arquitetura OF nome_da_entidade IS
SIGNAL  lista_de_sinal : tipo_de_dado;
BEGIN
        nome_da_instanciacao : nome_componente PORT MAP
        (lista_de_associacao_de_porta);          -- instanciacao do componente
END nome_da_arquitetura;
```

Resumo da Aula de Hoje

Tópicos mais importantes:

- **Projetos de Circuitos Combinacionais**