

ORGANIZAÇÃO E ARQUITETURA DE COMPUTADORES I

Circuitos Combinacionais Básicos

Descrição VHDL

prof. Dr. César Augusto M. Marcon
prof. Dr. Edson Ifarraguirre Moreno

Circuitos combinacionais básicos e representação em VHDL

- **Exemplos de circuitos combinacionais**

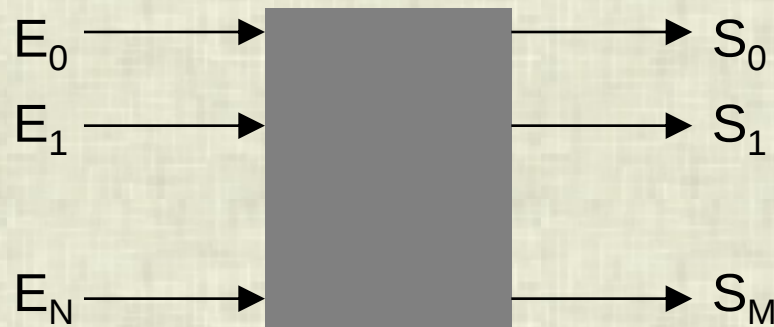
- **Decodificador / Codificador**
- **Comparadores**
- **Geradores de paridade**
- **Multiplexador / Demultiplexador**
- **Somador / Subtrator**
- **ULA**
- **Multiplicadores / Divisores**
- **PLAs**

- **Exemplos de circuitos seqüenciais**

- **Multiplicadores / Divisores**
- **Registradores (deslocamento, carga paralela, acumulador, serial-paralelo)**
- **Contadores (binário, BCD, Johnson, Gray / up, down, up-down)**
- **Máquina de Estados**
- **Geradores de clock**
- **Seqüenciadores**

Codificador

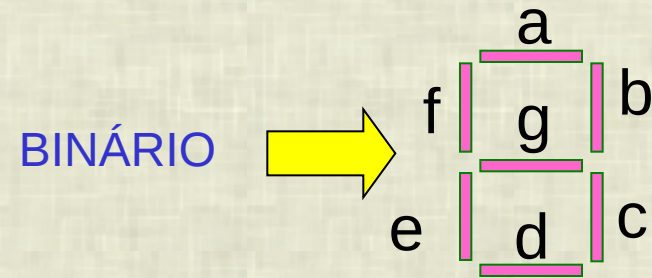
- Codificador é um circuito que mapeia um conjunto de entradas em um conjunto de saídas segundo uma função de codificação
 - **Em outras palavras, é um circuito que transforma uma informação de um formato para outro**
- Um codificador é normalmente implementado de forma combinacional
- Implementação deste em VHDL pode ser realizada com o comando with-select
- Representação gráfica de codificador genérico



Codificador

- O exemplo abaixo ilustra um codificador BCD (binary-coded decimal) para sete segmentos
 - Entrada em 4 bits (vetor Ent(3 downto 0))
 - Saída em 7 bits (vetor S(6 downto 0))

```
with Ent select
  S <="0000001" when "0000",
    "1001111"  when "0001",
    "0010010"  when "0010",
    "0000110"  when "0011",
    "1001100"  when "0100",
    "0100100"  when "0101",
    "0100000"  when "0110",
    "0001111"  when "0111",
    "0000000"  when "1000",
    "0001100"  when "1001",
    "1111110"  when others;
```



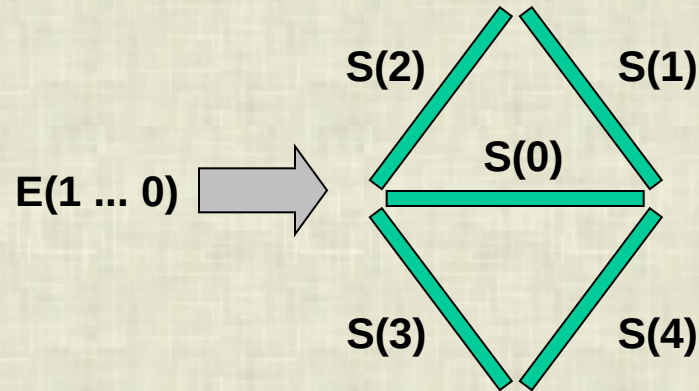
EXERCÍCIOS:

- Descubra qual é a codificação do display de 7 segmentos, que relaciona as posições de S com os segmentos a,b,c,d,e,f,g
- Diga se os segmentos acendem com 0 ou 1
- Faça agora um codificador para mostrar todos os números hexadecimais

Codificador

EXERCÍCIOS:

- Faça a codificação do display de elevador ilustrado abaixo



- Este tem como entrada um vetor de 2 bits que recebe a seguinte codificação

parado	00	-
subindo	01	/\
descendo	10	V
estragado	11	todos segmentos acesos

Codificador com Prioridade

- Codificador com prioridade é um circuito que determina entradas preferenciais
- Conforme a ordem de preferência das entradas, no caso de haver mais de uma entrada ativa (por exemplo em 1), a entrada de maior prioridade é que determinará a codificação
- Exemplo:
 - A prioridade máxima é de S(2) e a mínima de S(0)

```
Y <= "11" when S(2) = '1' else  
     "10" when S(1) = '1' else  
     "01" when S(0) = '1' else  
     "00";
```



Importante haver condição *default* em atribuições e estruturas similares:
NÃO HAVENDO ESTA CONDIÇÃO IMPLICA EM HAVER MEMORIZAÇÃO DO SINAL - diferente de software! (warning *latch infered*)

Exercício:

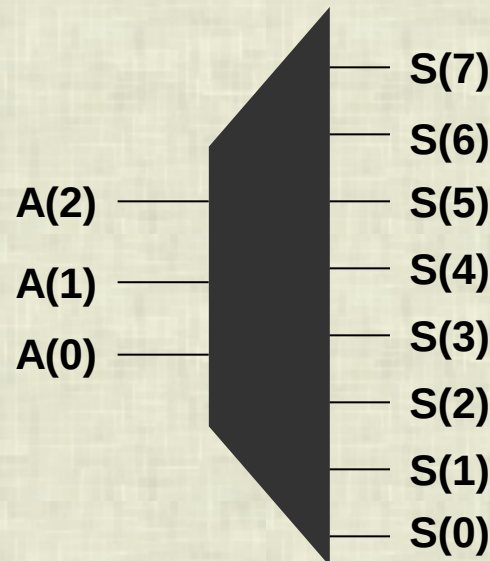
Desenhe o diagrama de blocos deste circuito

Decodificador

- Decodificador é igualmente como um codificador, um circuito que mapeia entradas em saídas segundo uma função de codificação
- Genericamente, um circuito é chamado de decodificador, quando a função de transformação que ele realiza é inversa a de um circuito codificador
- Um caso especial de decodificador é o binário. Este relaciona n entradas em 2^n saídas. Normalmente utilizado para endereçamento de memórias RAM
- Exemplo:
 - Decodificador de 3 entradas ($A(2...0)$) e 2^3 (8) saídas ($S(7...0)$)

```
with A select
```

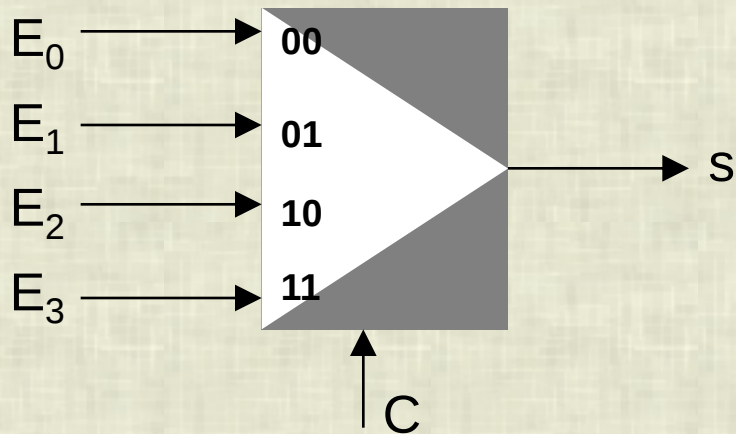
```
S <= "00000001" when "000",  
     "00000010" when "001",  
     "00000100" when "010",  
     "00001000" when "011",  
     "00010000" when "100",  
     "00100000" when "101",  
     "01000000" when "110",  
     "10000000" when "111";
```



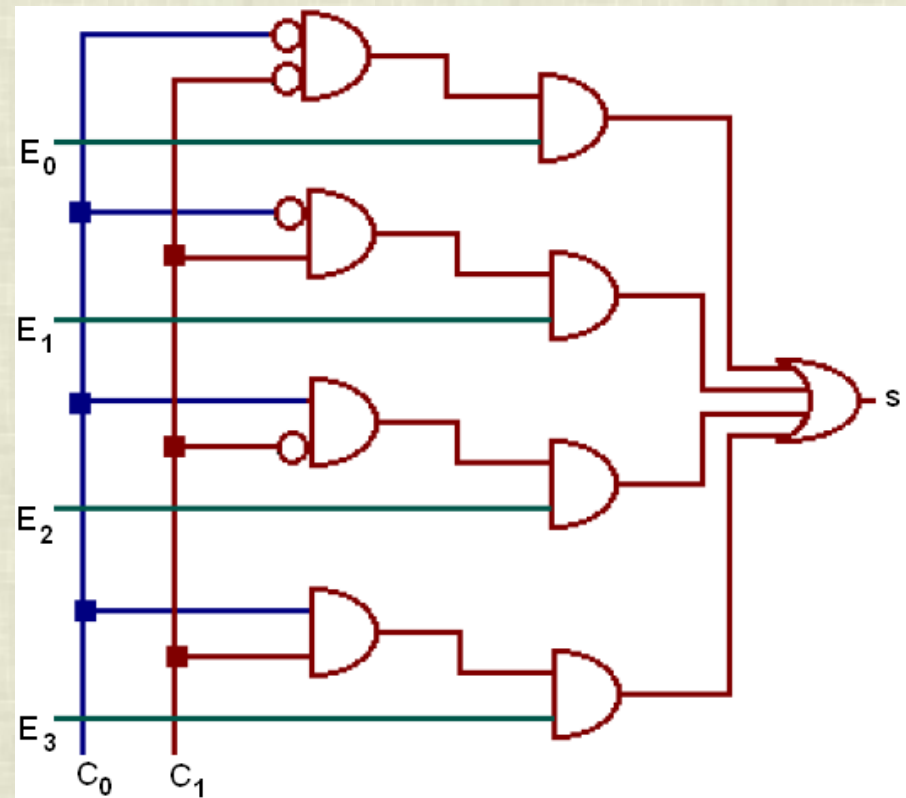
Multiplexador (Mux)

- É um circuito que permite selecionar mais de uma entrada em uma mesma saída, conforme um sinal de seleção
- Exemplo de um multiplexador 4x1

Representação



Implementação com portas lógicas



Multiplexador

Multiplexadores podem ser implementados com diversos comandos:

```
s <= (E(0) and (not C(0) and not C(1) )) or
      (E(1) and (not C(0) and C(1) ))      or
      (E(2) and (C(0) and not C(1) ))      or
      (E(3) and (C(0) and C(1) ));
```

with C select

```
s <= E(0) when "00",
      E(1) when "01",
      E(2) when "10",
      E(3) when others;
```

```
s <= E(0) when C(0) = 0 and C(1) = 0 else 'Z';
s <= E(1) when C(0) = 0 and C(1) = 1 else 'Z';
s <= E(2) when C(0) = 1 and C(1) = 0 else 'Z';
s <= E(3) when C(0) = 1 and C(1) = 1 else 'Z';
```

process(E, C)

begin

case sel is

when "00" =>

s <= E(0);

when "01" =>

s <= E(1);

when "10" =>

s <= E(2);

when others =>

s <= E(3);

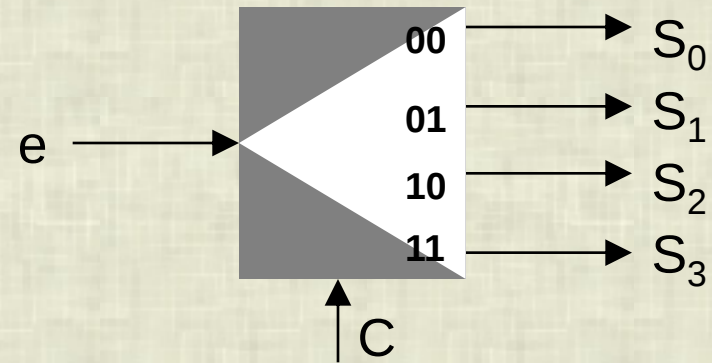
end case;

end process;

Demultiplexador (Demux)

– É um circuito que opera de forma inversa ao multiplexador. Ou seja, recebe uma entrada e distribui esta em uma de várias saídas conforme um sinal de seleção

– Exemplo de um multiplexador 1x4



– Igualmente ao multiplexador, um demultiplexador pode ser implementado com diversos comandos. O mais comum, é o que segue:

S(0) <= e when C = "00" else 'Z';

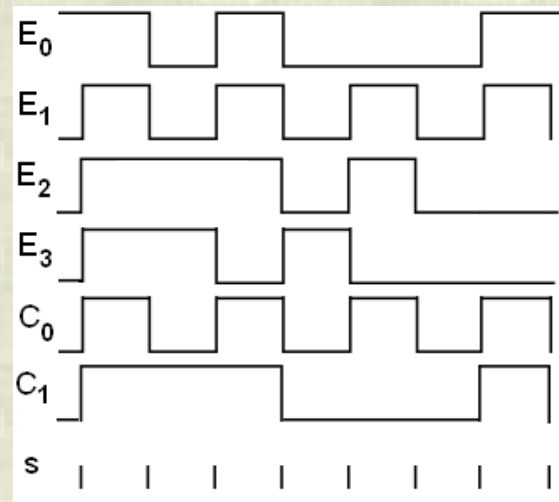
S(1) <= e when C = "01" else 'Z';

S(2) <= e when C = "10" else 'Z';

S(3) <= e when C = "11" else 'Z';

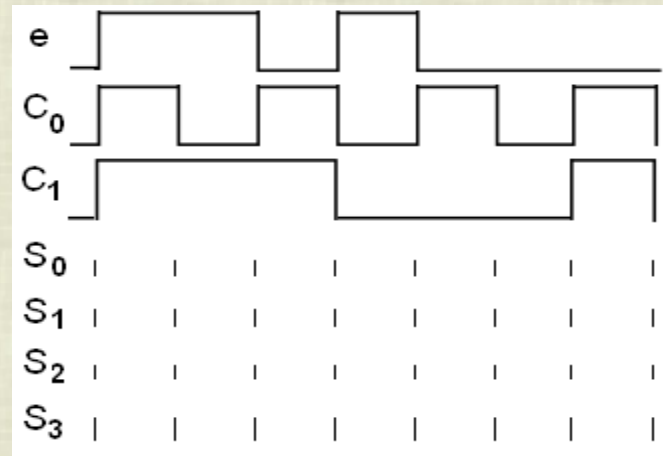
Exercícios

1. Uma fábrica produz porcas e parafusos. O presidente da fábrica necessita de um sistema automático para separar as peças em lotes. Sempre que na esteira houver somente porcas, as mesmas devem ser enviadas para um caixa A. Se na esteira houver somente parafusos, os mesmos devem ser enviados para uma caixa B. Caso existam na esteira porcas e parafusos, os mesmos devem ser enviados para uma outra caixa C. Projete este sistema utilizando um circuito adequado (codificador, decodificador, multiplexador ou demultiplexador)
2. A figura abaixo mostra as formas de ondas um mux 4x1 (E: entrada, C: controle, s: saída). Desenhe as formas de onda de s



Exercícios

3. A figura abaixo mostra as formas de ondas um demux 1x4 (E: entrada, C: controle, S: saídas). Desenhe as formas de onda de S para o caso do demux ser implementado com tri-states e para o caso de ser implementado com portas lógicas



4. Implemente um multiplexador 4x1, utilizando apenas multiplexadores 2x1
5. Implemente um demultiplexador 1x4 utilizando um demultiplexador 1x8
6. Implemente a expressão booleana a seguir usando um mux 8x1
- $$S \leq (A \text{ and not } B) \text{ or } (A \text{ and not } (B \text{ and } C));$$

Exercícios

7. Implemente a expressão booleana a seguir utilizando um mux 4x1 e uma porta NAND

$S \leq (A \text{ and not } B \text{ and not } C) \text{ or } (A \text{ and not } B \text{ and } C) \text{ or } (A \text{ and } B \text{ and not } C);$

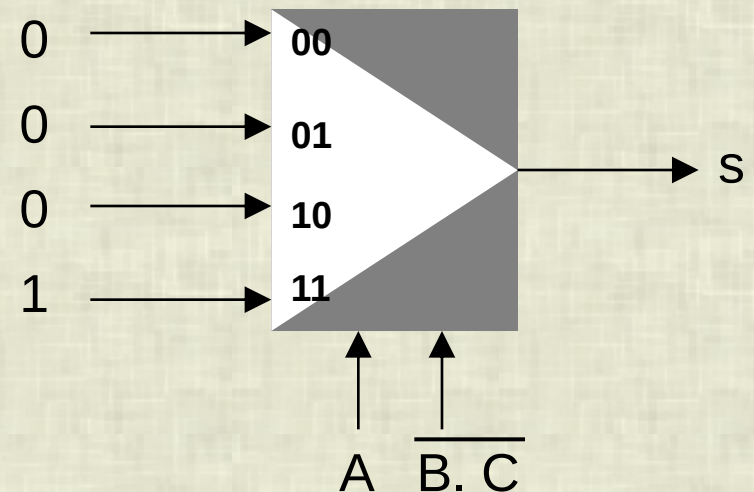
Solução de Exercícios

7. Implemente a expressão booleana a seguir utilizando um mux 4x1 e uma porta NAND

$$S \leq (A \text{ and not } B \text{ and not } C) \text{ or } (A \text{ and not } B \text{ and } C) \text{ or } (A \text{ and } B \text{ and not } C);$$

Simplificando: $S \leq A \text{ and not } (B \text{ and } C);$

A	B	C	A	not(B and C)	S
0	0	0	0	1	0
0	0	1	0	1	0
0	1	0	0	1	0
0	1	1	0	0	0
1	0	0	1	1	1
1	0	1	1	1	1
1	1	0	1	1	1
1	1	1	1	0	0



Exercícios

8. Implemente a expressão booleana a seguir utilizando dois mux 4x1 e um porta AND e um INVERSOR

$S1 \leq \text{not } X \text{ or } (\text{not } Y \text{ and } Z) \text{ or } \text{not } (X \text{ and } Y);$

$S2 \leq (X \text{ or } Y) \text{ and } \text{not } (X \text{ or } Z);$

Solução de Exercícios

8. Implemente a expressão booleana a seguir utilizando dois mux 4x1 e um porta AND e um INVERSOR

$$S1 \leq \text{not } X \text{ or } (\text{not } Y \text{ and } Z) \text{ or } \text{not } (X \text{ and } Y);$$

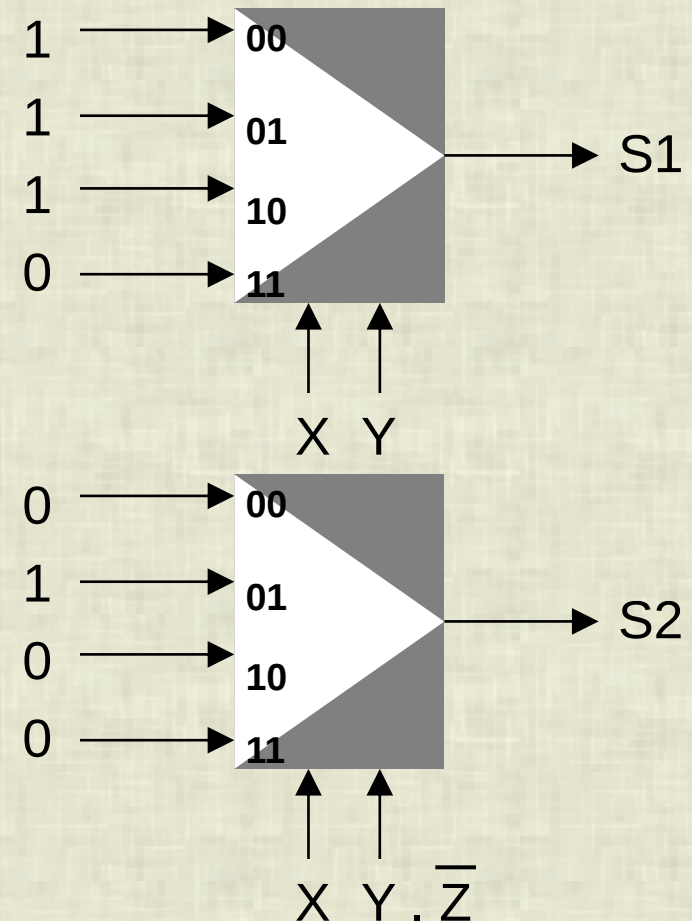
$$S2 \leq (X \text{ or } Y) \text{ and } \text{not } (X \text{ or } Z);$$

Simplificando:

$$S1 \leq \text{not } X \text{ or } \text{not } Y;$$

$$S2 \leq \text{not } X \text{ and } Y \text{ and } \text{not } Z$$

X	Y	Z	X	Y and not Z	S2
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	1	1
0	1	1	0	0	0
1	0	0	1	0	0
1	0	1	1	0	0
1	1	0	1	1	0
1	1	1	1	0	0



Exercícios

9. Faça um decodificador em portas lógicas que implemente a tabela verdade abaixo

A	B	C	S_1	S_0
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	1	0
1	0	1	1	1
1	1	0	0	1
1	1	1	1	1

10. Apresente em portas lógicas um demultiplexador 1x4
11. Faça o mesmo exercício acima, agora considerando o uso de tri-states