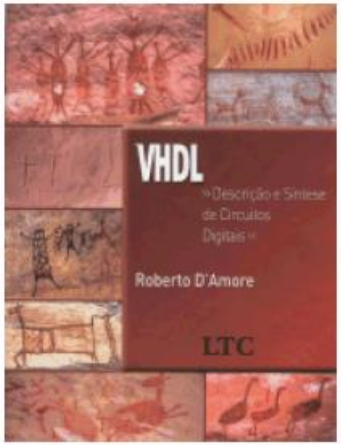
SEL0606 – Laboratório de Sistemas Digitais
Máquinas de Estados em VHDL

Prof. Dr. Maximiliam Luppe

Livro adotado:

VHDL - Descrição e Síntese de Circuitos Digitais
Roberto d'Amore

ISBN 85-216-1452-7
Editora LTC www.ltceditora.com.br



Para informações adicionais consulte: **www.ele.ita.br/~damore/vhdl**

Estratégias de descrição de circuitos síncronos

Tópicos

- Registrador sensível a nível
- Registrador sensível a borda - inicialização síncrona
- Registrador sensível a borda - inicialização assíncrona
- Registrador sensível a borda com habilitação para sinal de relógio
- Máquinas de estado finito
- Contadores
- Cuidados na descrição

Máquinas de estado finito

- **Máquina de estados**:
  - circuito seqüencial que transita numa seqüência pré definida de estados

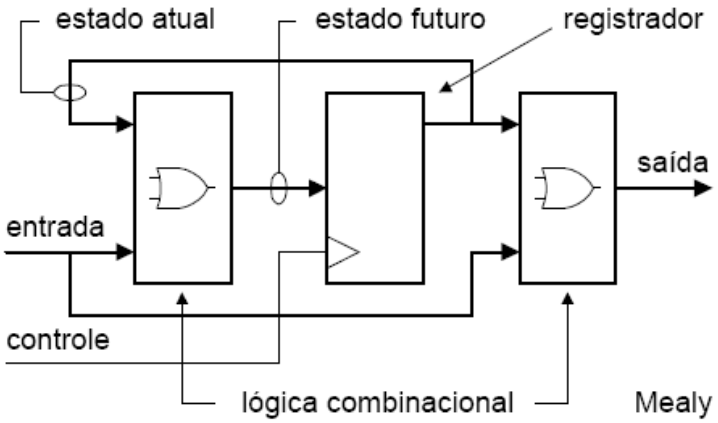
- **Transição entre estados**:
  - comandada por um sinal de controle

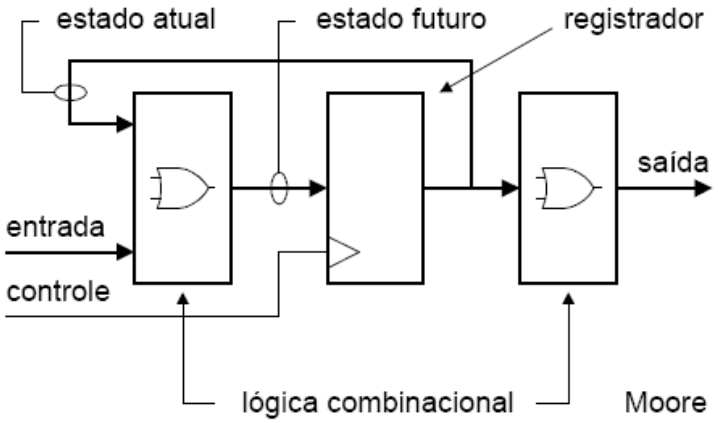
- **Estado atual**:
  - definido por elementos de memória

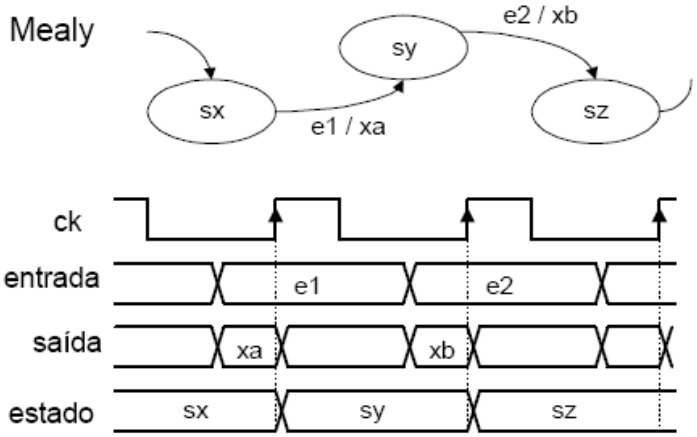
- **Estado futuro**
  - determinado com base no estado atual e a condição das entradas

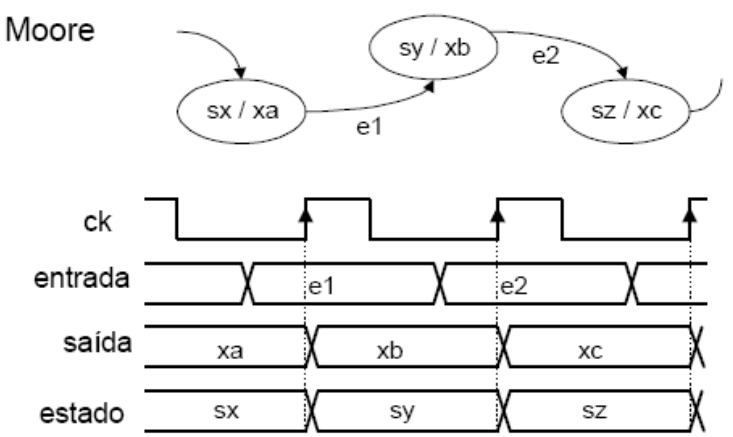
- **2 estilos**:
  - máquina *Mealy*
    - o valor da saída é função do estado atual e da condição das entradas
  - máquina *Moore*
    - o valor da saída depende exclusivamente do estado atual

Máquinas de estado finito









Máquinas de estado finito

- **Inferência de uma máquina de estado** - uma descrição contendo:
  - um objeto que armazena o estado atual (registradores)
  - uma definição da transição de estados (lógica combinacional)
  - uma especificação dos valores de saída (lógica combinacional)

- **Transição entre estados na descrição**:
  - controlada por um sinal de relógio
  - pode conter:
    - condições de inicialização síncronas ou assíncronas

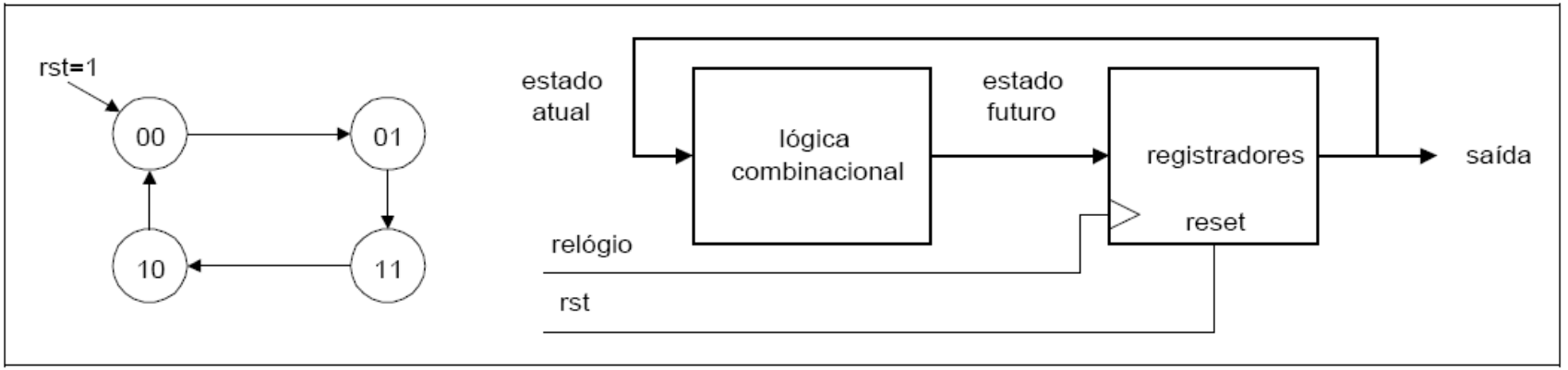
Máquinas de estado finito

- **Construção do tipo IF ELSE detecta**:
  - inicialização rst =1
  - ocorrência de uma borda de subida no sinal de relógio
- **Construção CASE WHEN**:
  - definição das transições de estado
  - força a especificação de todos os estados

```
PROCESS (ck, rst)
 BEGIN
   IF rst = '1' THEN                                    -- estado inicial
     estado <= estado_inicial;
   ELSIF (ck'EVENT and ck ='1') THEN                    -- ciclo de estados
     CASE estado IS                                     --   .
       WHEN estado_inicial => estado <= estado_1;       --   .
       WHEN estado_1       => estado <= estado_2;       --   .
                                                        --   .
       WHEN estado_x       => estado <= estado_final;
     END CASE;
   END IF;
 END PROCESS;
```

Máquinas de estado finito - exemplo

- **Contador**
- **Saída dos registradores corresponde ao valor de saída**
  - código do estado = valor de saída
    - não é necessário decodificar

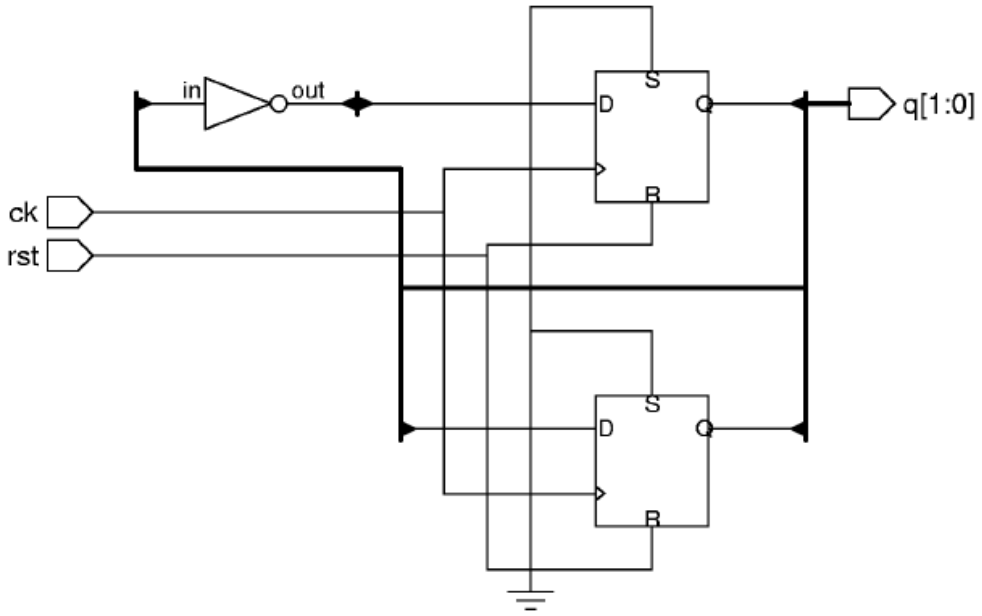


Máquinas de estado finito - descrição

```
 1 ENTITY maq_est1 IS
 2   PORT (ck      :  IN     BIT;                        -- relogio borda subida
 3         rst     :  IN     BIT;                        -- rst=1, q=00
 4         q       :  BUFFER BIT_VECTOR (1 DOWNTO 0)); -- saida
 5 END maq_est1;
 6
 7 ARCHITECTURE teste OF maq_est1 IS
 8
 9 BEGIN
10   abc: PROCESS (ck, rst)
11   BEGIN
12     IF rst = '1' THEN                          -- estado inicial
13       q <= "00";
14     ELSIF (ck'EVENT and ck ='1') THEN          -- ciclo de estados
15       CASE q IS
16         WHEN "00" => q <= "01";
17         WHEN "01" => q <= "11";
18         WHEN "11" => q <= "10";
19         WHEN "10" => q <= "00";
20        END CASE;
21     END IF;
22   END PROCESS abc;
23 END teste;
```

Máquinas de estado finito - contador

- **Circuito gerado nível RTL**
- **Inicialização assíncrona inferida corretamente**:
  - rst =1 leva ao estado 00
    (sinal rst ligado ao *reset* dos flip flops)



Máquinas de estado finito - definição dos estados

- **Valores binários assumidos pelos estados - opções**:
  - especificados na descrição
  - codificados pelo aplicativo de síntese

- **Codificados pelo aplicativo de síntese**:
  - um tipo enumerado composto nomes
    - cada nome representam o estado da máquina
    - resultado: uma leitura mais fácil do código

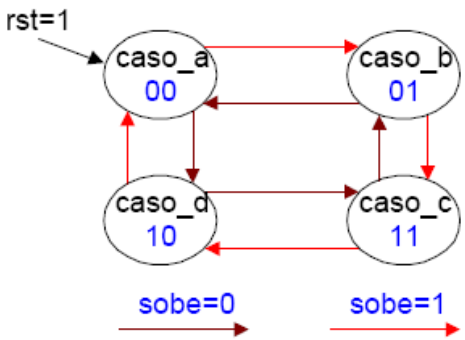
- **Ferramenta de síntese atribui a cada estado um valor binário**

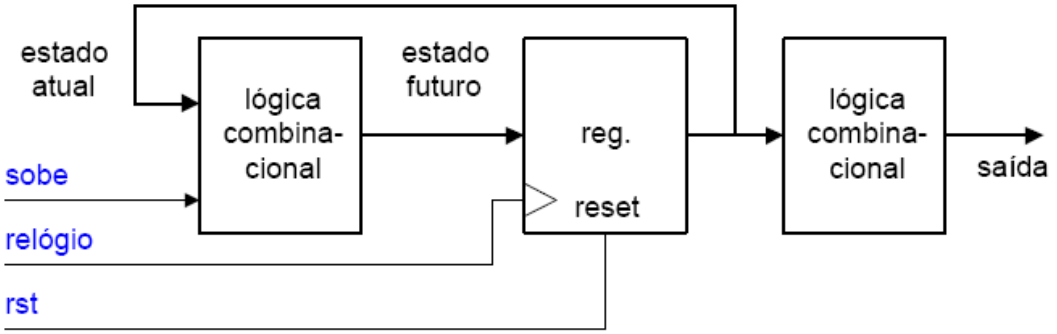
- **Exemplos de esquemas de atribuição**:

estado definido no código	seqüencial	código Gray	único um
reset	000	000	00001
ciclo_rd	001	001	00010
ciclo_wr	010	011	00100
ciclo_int	011	010	01000
ciclo_dma	100	110	10000

Máquinas de estado finito - contador crescente / decrescente

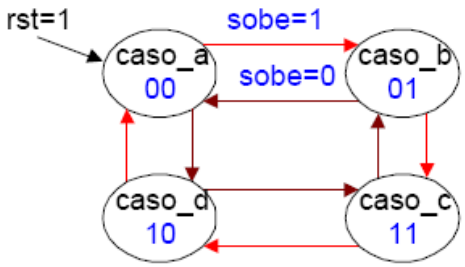
- Evolução dos estados segundo o esquema:





Máquinas de estado finito - contador crescente / decrescente

- **Evolução dos estados**:



- **Tipo enumerado definido na descrição**:

```
TYPE st IS (caso_d, caso_c, caso_b, caso_a); -- novo tipo definido
SIGNAL estado : st;                          -- sinal pode assumir os
                                             -- valores definidos em st
```

- **4 estados definidos** → st: caso_d caso_c caso_b caso_a
- **valores da saída q correspondentes** → q: 10 11 01 00
- **Correlação estado ↔ saída**:
  - pode ser necessário uma decodificação
- **Sinal estado armazena o estado atual da máquina**

Máquinas de estado finito - contador código crescente / decrescente

- **Descrição**: definição tipo enumerado e seqüência de estados

```
 8 ARCHITECTURE teste OF maq_est2 IS
 9   --            q=2     q=3     q=1     q=0
10   TYPE st IS (caso_d, caso_c, caso_b, caso_a);  -- novo tipo definido
11   SIGNAL estado : st;                           -- sinal estado tipo "st"
12 BEGIN
13   abc: PROCESS (ck, iniciar)
14   BEGIN
15     IF iniciar = '1' THEN                 -- estado inicial
16       estado <= caso_a;                   -- q=0
17     ELSIF (ck'EVENT and ck ='1') THEN     -- ciclo de estados
18       CASE estado IS
19         WHEN caso_a =>                              -- q=0
20           IF sobe = '1' THEN estado <= caso_b;  -- q=1
21           ELSE               estado <= caso_d;  -- q=2
22           END IF;
23         WHEN caso_b =>                              -- q=1
24           IF sobe = '1' THEN estado <= caso_c;  -- q=3
25           ELSE               estado <= caso_a;  -- q=0
26           END IF;
27         WHEN caso_c =>                              -- q=3
28           IF sobe = '1' THEN estado <= caso_d;  -- q=2
29           ELSE               estado <= caso_b;  -- q=1
30           END IF;
31         WHEN caso_d =>                              -- q=2
32           IF sobe = '1' THEN estado <= caso_a;  -- q=0
33           ELSE               estado <= caso_c;  -- q=3
34           END IF;
35       END CASE;
36     END IF;
37   END PROCESS abc;
38
```

```
39   WITH estado SELECT   -- decodifica estado
40     q <= "00" WHEN caso_a, -- q=0
41          "01" WHEN caso_b, -- q=1
42          "11" WHEN caso_c, -- q=3
43          "10" WHEN caso_d; -- q=2
44 END teste;
```

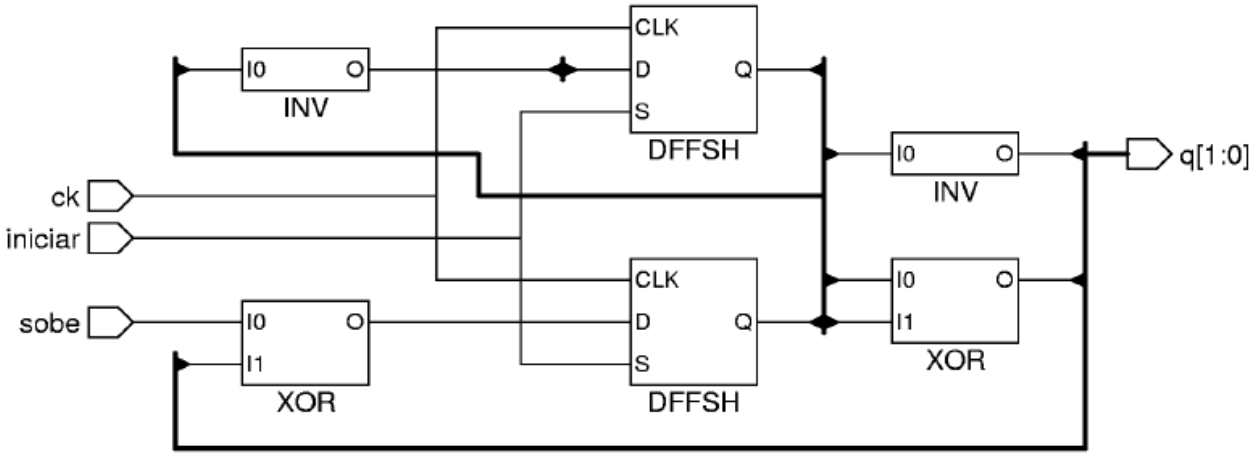
- **Descrição continuação**: decodificação do estado ↑

- **Atribuição de estados binária**
  - Mensagem da ferramenta: codificação dos estados:

```
"/vhdl/maq_est2.vhd",line 10: Info, Enumerated type st with 4 elements encoded as
binary.
Encodings for st values
        value    st[1-0]
========================
         caso_d  00
         caso_c  01
         caso_b  10
         caso_a  11
```

Máquinas de estado finito - contador crescente / decrescente

- **Inicialização assíncrona do estado inicial**: caso_a
- **Sinal iniciar interligado às entradas de *set* assíncronas dos *flip flops***
  - iniciar =1 leva os *flip flops* para o valor 11
    - corresponde ao estado caso_a




```
Encodings for st values
        value     st[1-0]
=======================
          caso_d  00
          caso_c  01
          caso_b  10
          caso_a  11
```

Cuidados na descrição - Ciclo de inicialização

- **Projeto de um circuito síncrono**:
  deve sempre prever um ciclo de inicialização

- **O valor dos registradores após a energização do circuito**:
  desconhecido

- **Falta de um ciclo de inicialização**:
  - comportamento não previsto pela simulação: exemplo
    - iniciar estado diferente
    - iniciar estado não previsto na descrição

- **Portanto**:
  - deve-se incluir de operações de inicialização
    - assíncronas ou síncronas

Cuidados na descrição - Inserção desnecessária de um *latch*

- **Na linguagem VHDL**:
  - a falta de atribuição de um valor em um objeto:
    - implica na manutenção do valor neste objeto
- **Problemas**:
  - transição de estados especificada incompletamente
  - falta de atribuição de um valor a uma saída
    - leva a inserção *latch* desnecessário.

- **Construção CASE WHEN na especificação da transição de estados**
  - todas as condições devem ser obrigatoriamente cobertas

- **Evitar cláusula OTHERS**
  - lógica combinacional extra pode ser gerada
    - necessário detectar os estados restantes

- **Opção**:
  - criar tipo enumerado definindo todos os estados previstos
  - não é necessário uso da clausula OTHERS