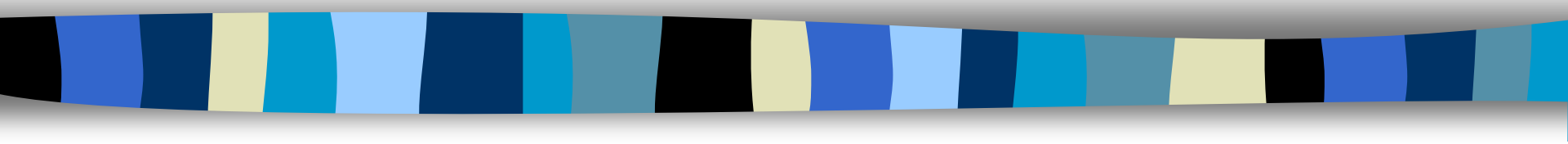
Máquinas de Estados Finitos



Aula 19

Prof. Abel Guilhermino

Definição

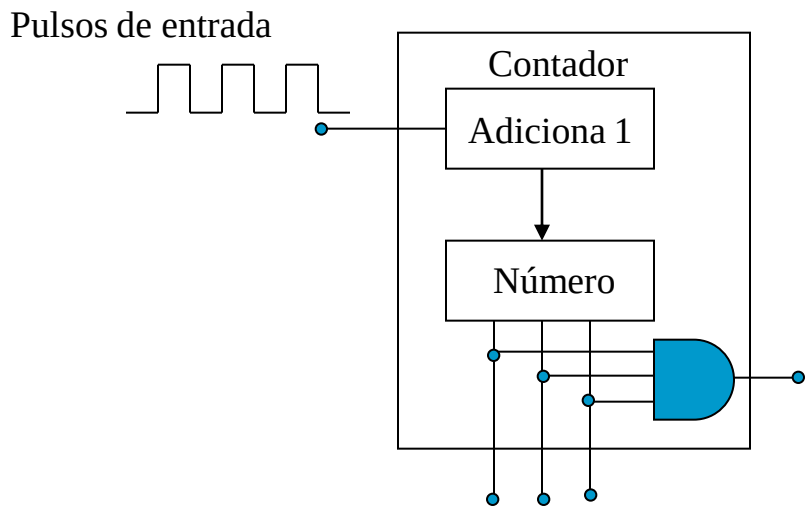
- Um sistema seqüencial deve ter a capacidade de capturar a influência de todas as entradas passadas sobre as saídas atuais e futuras.
- Os valores das seqüências de entrada podem ser agrupados em um número finito de **classes** de tal maneira que todas as funções no tempo que têm o mesmo efeito sobre a saída no tempo $t \geq t_1$ são incluídas na mesma classe.
- Como resultado, a determinação de z(t) não precisa da seqüência de entrada inteira x(0,t) porque basta saber a classe à qual a função pertence.
- A classe é mantida em uma variável auxiliar s chamada **estado**, a qual também é uma função do tempo.
- Uma vez que o número de classes (isto é, o número de estados do sistema) é finito, os sistemas são chamados de sistemas de estados finitos ou **máquinas de estados finitos**.

Representação FSM

- O estado de um sistema seqüencial usa três variáveis no tempo:
  - A entrada
  - O estado
  - A saída
- Além disso há duas funções:
  - Função transição de estado
    - Ou apenas funções de transição, a qual produz o próximo estado, como uma função da entrada atual x(t) e do estado atual s(t).
  - Função de saída
    - A qual produz a saída atual z(t) como uma função da entrada atual e do estado atual.

Representação FSM

- Para melhor explicar representação das máquinas de estados finitos, partimos de um exemplo de uma lógica seqüencial de um contador.



- A unidade é projetada para contar pulsos à medida que eles chegam na entrada X.
- O número de pulsos é indicado na palavra binária de 3 bits $n_2n_1n_0$.
- Palavra limitada à faixa 0-7.
- Saída R definida por:
  - R=0 (contador na faixa 0-6)
  - R=1 (contador em 7)

Representação FSM

- Essas definições caracterizam o ciclo do valor de $n_2n_1n_0$ de acordo com a seqüência:
  - $000 \rightarrow 001 \rightarrow 010 \rightarrow 011 \rightarrow 100 \rightarrow 101 \rightarrow 110 \rightarrow 111 \rightarrow 000..$
- Terminologia:
  - Definimos cada possível valor da palavra $n_2n_1n_0$ como sendo um estado distinto da máquina.
  - Isto significa que existem 8 possíveis estados nos quais o circuito pode estar.
  - Para descrever a operação do circuito, utilizamos desenhos chamados diagrama de estados.

Diagrama de estados

- É uma forma de representação através de desenhos usados para descrever a operação de um circuito.
- Os diagramas de estados mostram todos os estados individuais da máquina e as possíveis seqüências de mudança do circuito de um estado para o outro.
- O elemento básico consiste de um círculo que mostra o estado, com setas entrando/saindo para mostrar as transições para e do estado, respectivamente.

Diagrama de estados

- Na notação do desenho, utilizamos X/R para denotar o valor da entrada X responsável pela transição na direção mostrada e o valor da saída R quando a transição ocorre.
- Cada círculo deve ter linhas que mostram o que acontece para cada possível valor de X.

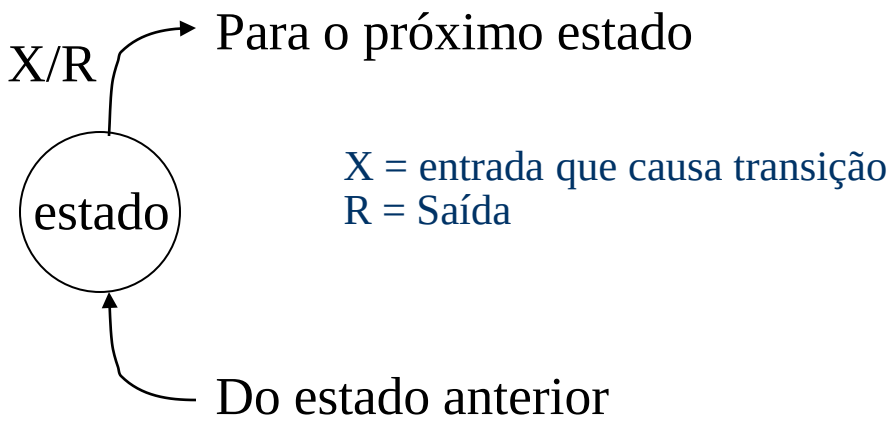
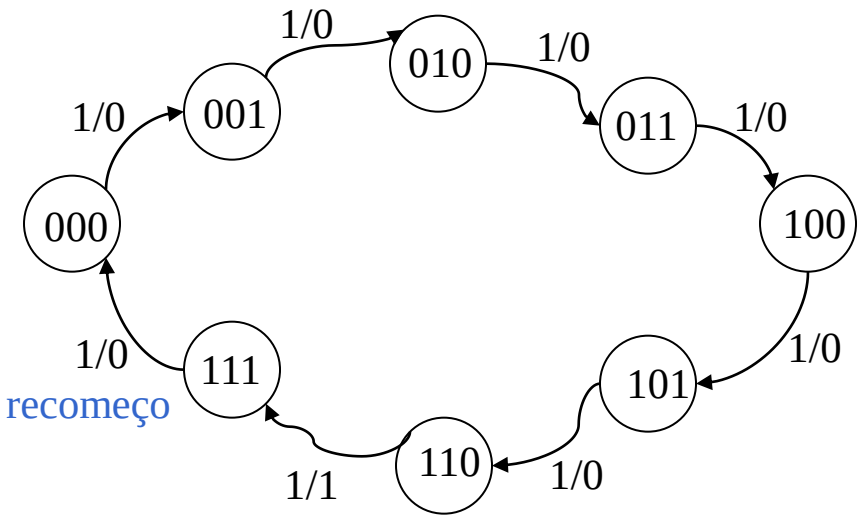
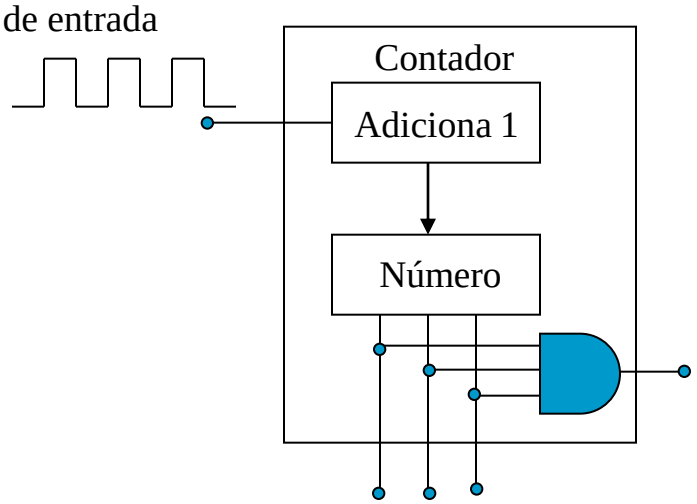


Diagrama de estados

- Diagrama de estados para o contador
  - Como a seqüência é bem-definida, um pulso de entrada de X=1 causa uma transição para o próximo estado.
  - A saída R é 0, a menos que o contador atinja o valor de $n_2n_1n_0$=111, resultando em R=1.

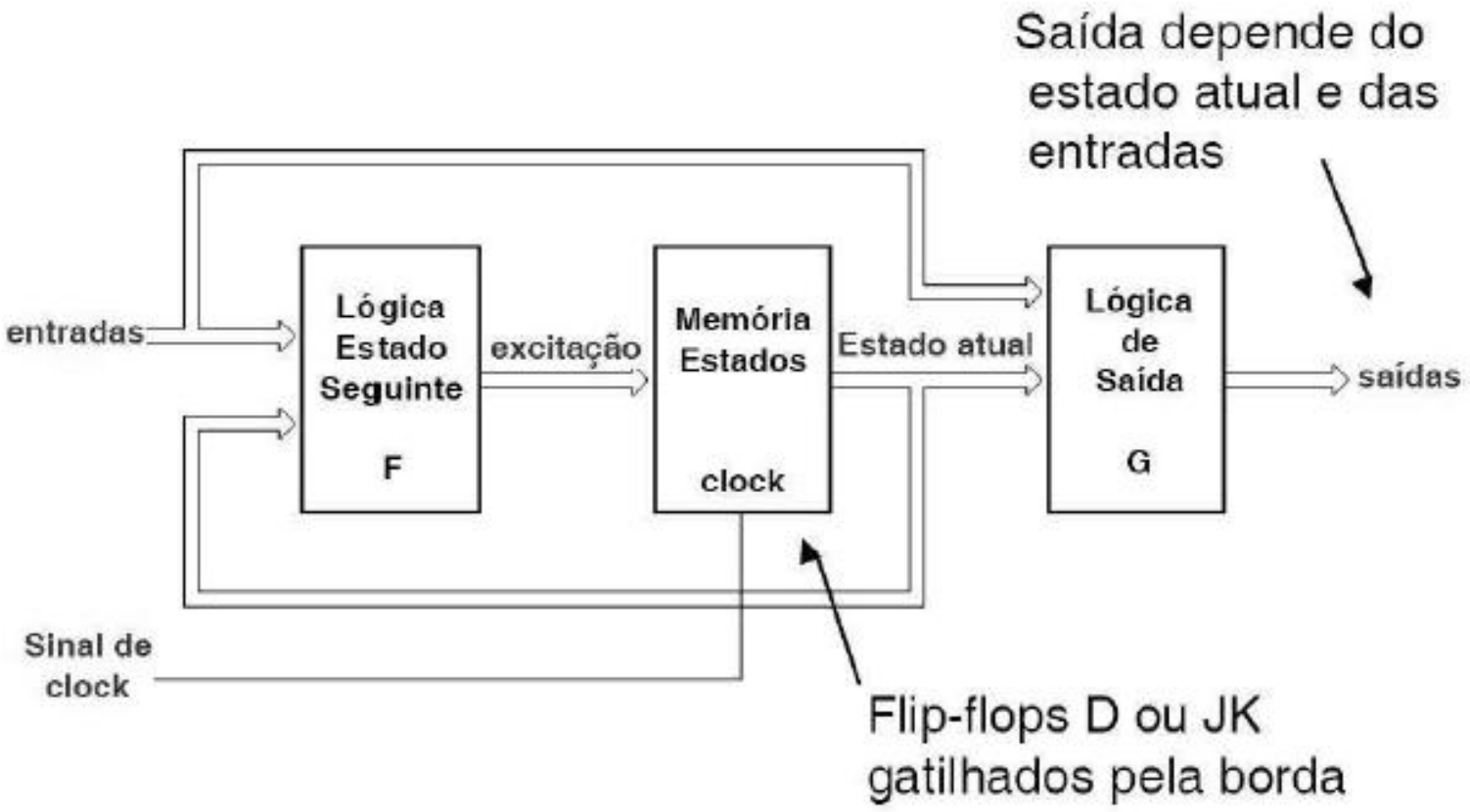




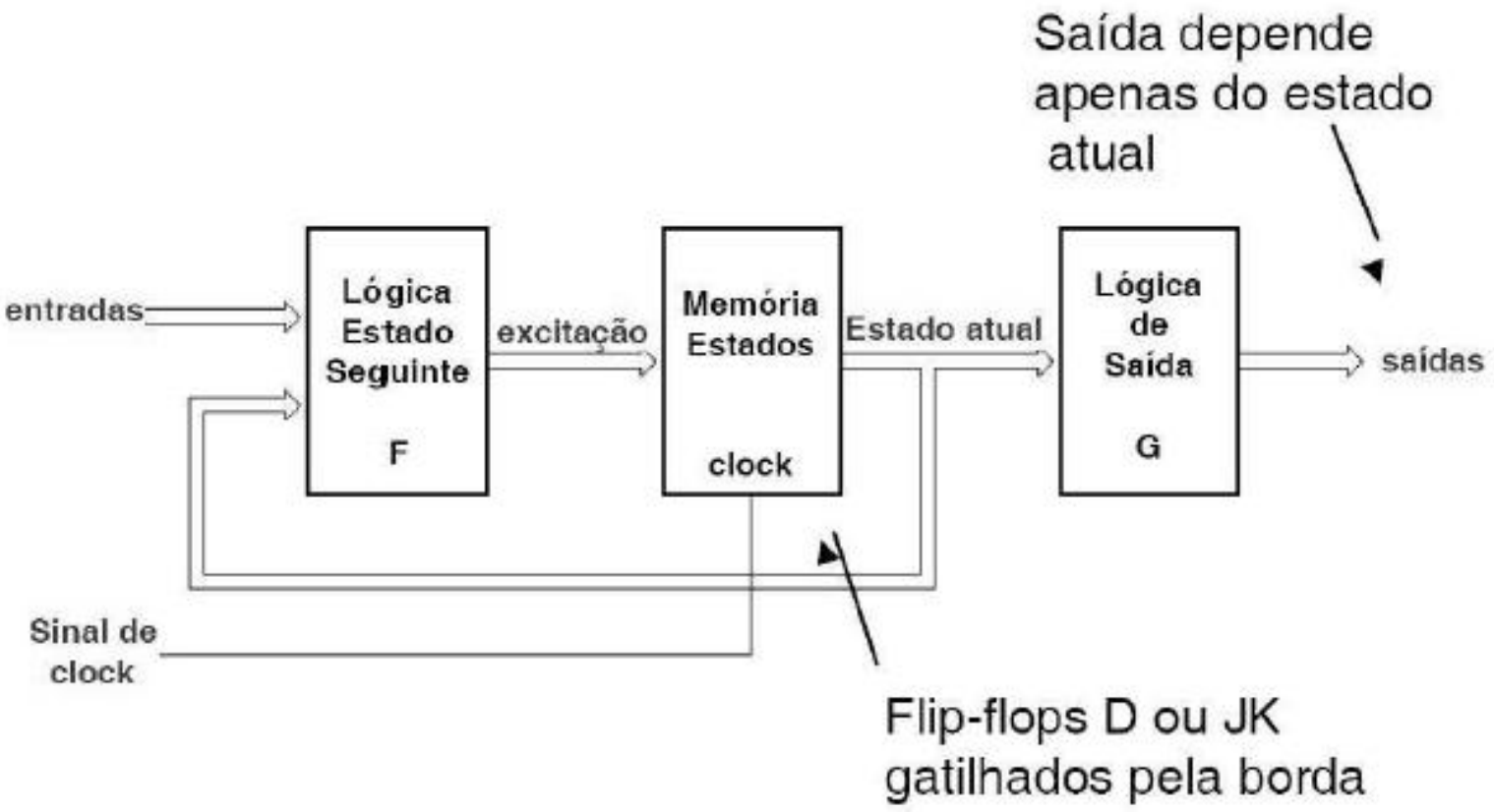
Máquinas de Estados

- É um circuito sequencial que transita numa sequência pré-definida de estados
- A transição entre estados é comandada por uma sinal de controle.
- O estado atual é definido por um elemento de memória, e o estado futuro é determinado com base no estado atual e na condição das entradas.

Máquina de Estados de Mealy



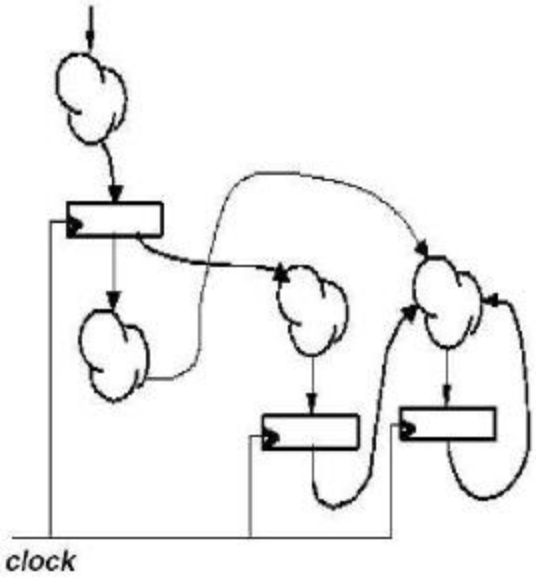
Máquina de Estados de Moore



Máquina de Moore x Mealy

- Moore
  - Saídas dependem apenas do estado atual
  - As saídas apenas são escritas quando os estados variam (transições de estados são síncronos)
- Mealy
  - As saídas dependem de ambos:entradas e estados
  - Quando a entrada muda, as saídas são atualizadas imediatamente, sem esperar por clock

Circuito Síncrono (Clock único)



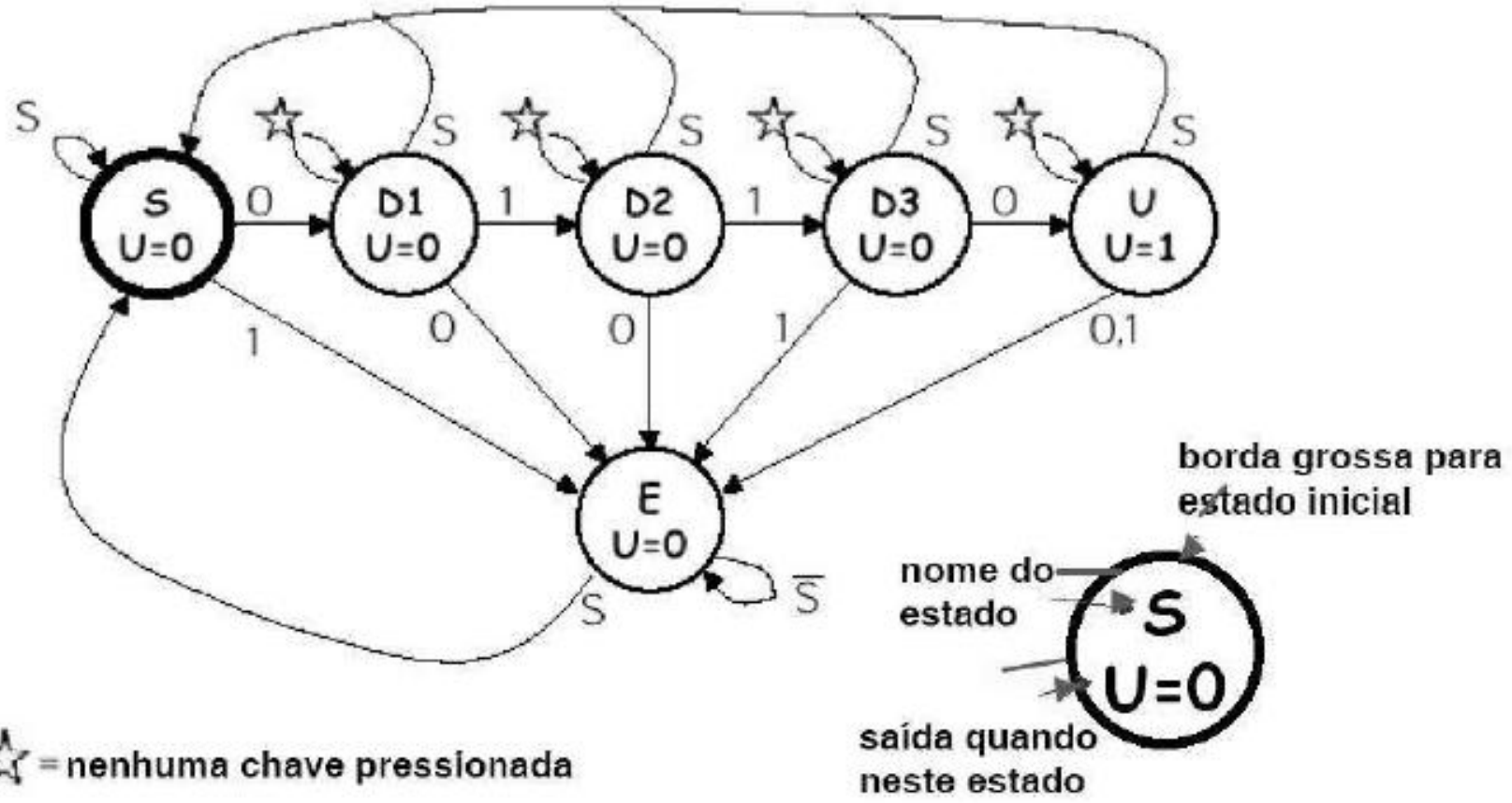
- Receita para circuitos sequenciais robustos:
  - As saídas dos circuitos combinacionais são analisadas apenas durante a subida de clock
  - O período de clock deve ser maior do que qualquer atraso combinacional
  - Os estados devem mudar apenas depois que todas as transições lógicas forem finalizadas

Lógica Sequencial

- A lógica sequencial é usada quando precisamos organizar a solução através de uma sequência de passos:
  - Exemplo: Como abrir uma fechadura com uma combinação digital. A fechadura possui três botões – “start”, “O”, “1”.
    - Passo 1: pressione o botão “Start”
    - Passo 2: pressiono o botão “O”
    - Passo 3: pressiono o botão “1”
    - Passo 4: pressiono o botão “1”
    - Passo 5: pressiono o botão “O”

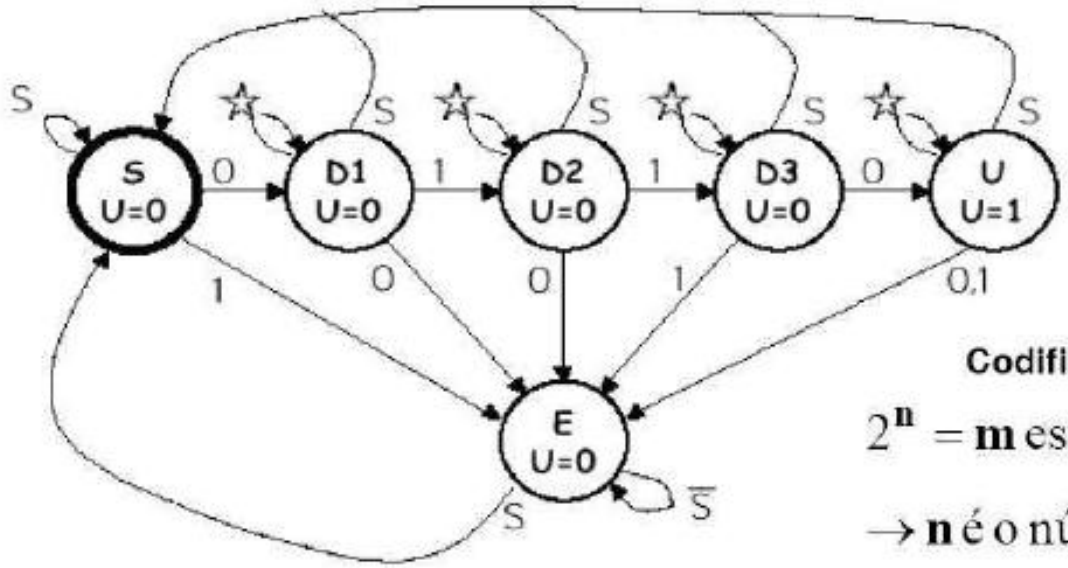
Implementação Máquina de Moore

Diagrama de Estado



obs. verificar que sempre que a chave START é pressionada, a seqüência volta para o estado inicial S.

Diagrama de estados Máquina de Moore



Codificador binário

$2^n = m$ estados possíveis

$\rightarrow n$ é o número de bits

6 estados diferentes : codificados por 3 bits

Arcos saindo de um estado devem ser:

(1) ***mutuamente exclusivos***

não pode ter duas opções para um mesmo valor de entrada

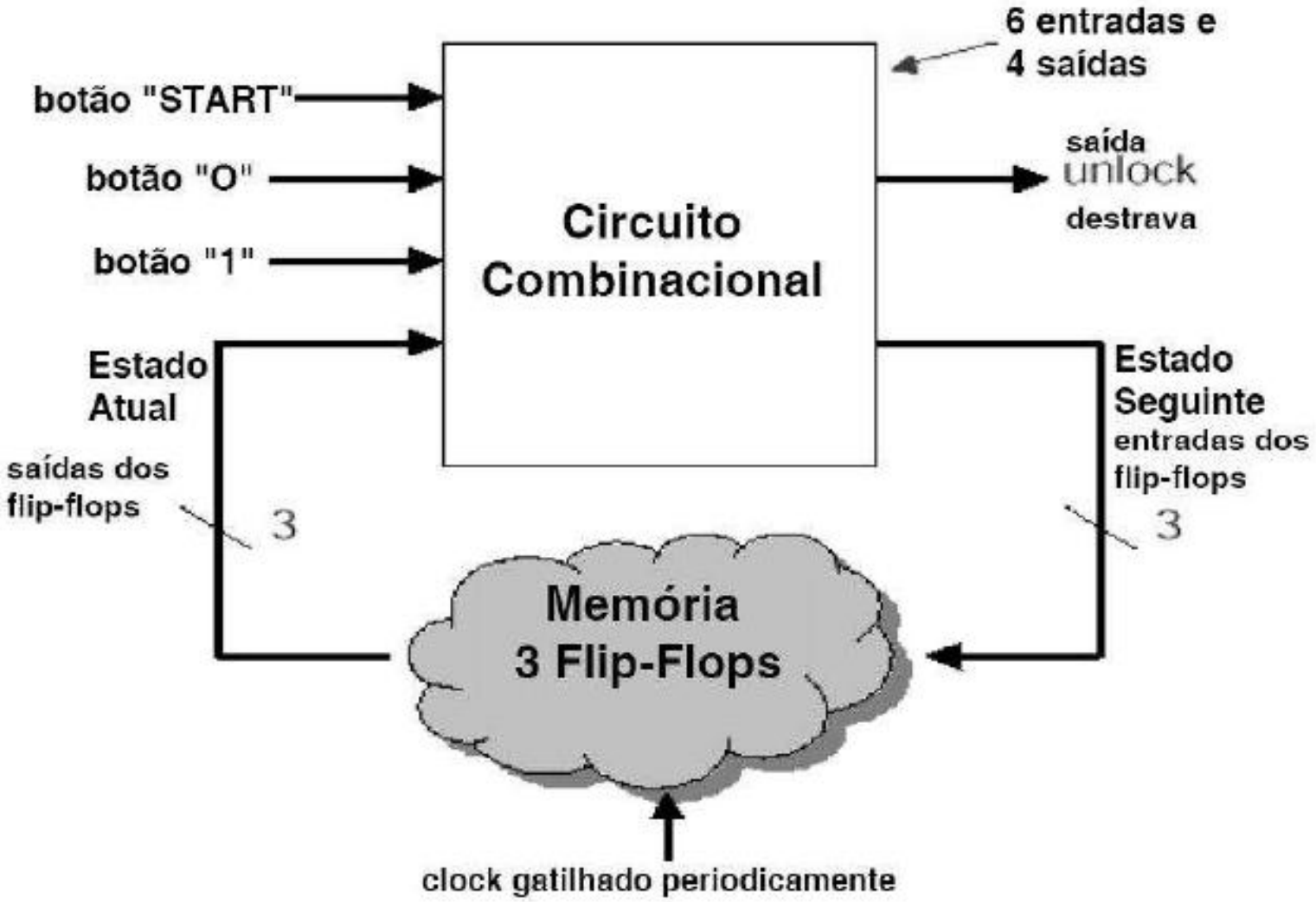
(2) ***coletivamente exaustivo***

todo estado deve especificar o que acontece para cada entrada possível. "*Nada acontece*" significa que o arco deve voltar para o mesmo estado.

Tabela de Transição de Saída Máquina de Moore

Estado atual	$Q_0\ Q_1\ Q_2$	B"start"	B"1"	B"0"	Estado seguinte	$(Q_0\ Q_1\ Q_2)^+$	saída U unlock
---		1	---	---	start	000	0
S start	000	0	0	1	digit1	001	0
S start	000	0	1	0	error	101	0
S start	000	0	0	0	start	000	0
D1 digit1	001	0	1	0	digit2	010	0
D1 digit1	001	0	0	1	error	101	0
D1 digit1	001	0	0	0	digit1	001	0
D2 digit2	010	0	1	0	digit3	011	0
D2 digit2	010	0	0	1	error	101	0
D2 digit2	010	0	0	0	digit2	010	0
D3 digit3	011	0	0	1	unlock	100	0
D3 digit3	011	0	1	0	error	101	0
D3 digit3	011	0	0	0	digit3	011	0
U unlock	100	0	1	0	error	101	1
U unlock	100	0	0	1	error	101	1
U unlock	100	0	0	0	unlock	100	1
E error	101	0	---	---	error	101	0

Implementação do Hardware Máquina de Moore



Mapas de Karnaugh

- Quando o botão "START" for 1, o circuito irá permanecer ou voltar para o estado inicial '000'. Portanto, podemos ligar o botão "START" no botão "RESET" dos flip-flops.

- Usando flip-flop tipo D - equação característica → $(Q_0\ Q_1\ Q_2)^* = (D_0\ D_1\ D_2)$

D_0 $Q_0 = 0$

Q_1Q_2 \ B1 B0	00	01	11	10
00	0	0	X	1
01	0	1	X	0
11	0	1	X	1
10	0	1	X	0

B1 (colunas) — Q_1 (linhas) — Q_2 — B0

$Q_0 = 1$

Q_1Q_2 \ B1 B0	00	01	11	10
00	1	1	X	1
01	1	1	1	1
11	X	X	X	X
10	X	X	X	X

B1 (colunas) — Q_1 (linhas) — Q_2 — B0

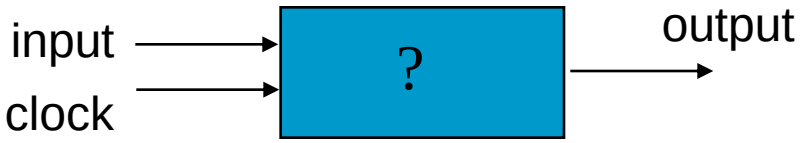
$$Q_0^* = D_0 = B1(\overline{Q_1 Q_2} + Q_1 Q_2) + B0(Q_2 + Q_1) + Q_0$$

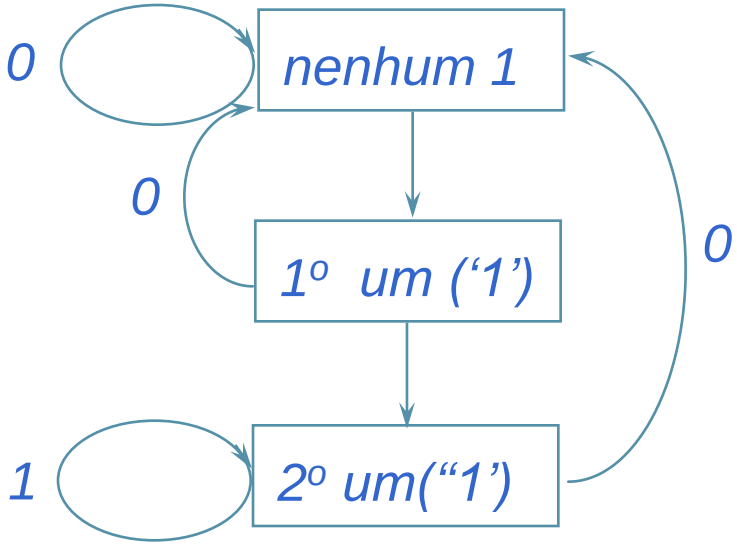
Equação de excitação ou transição

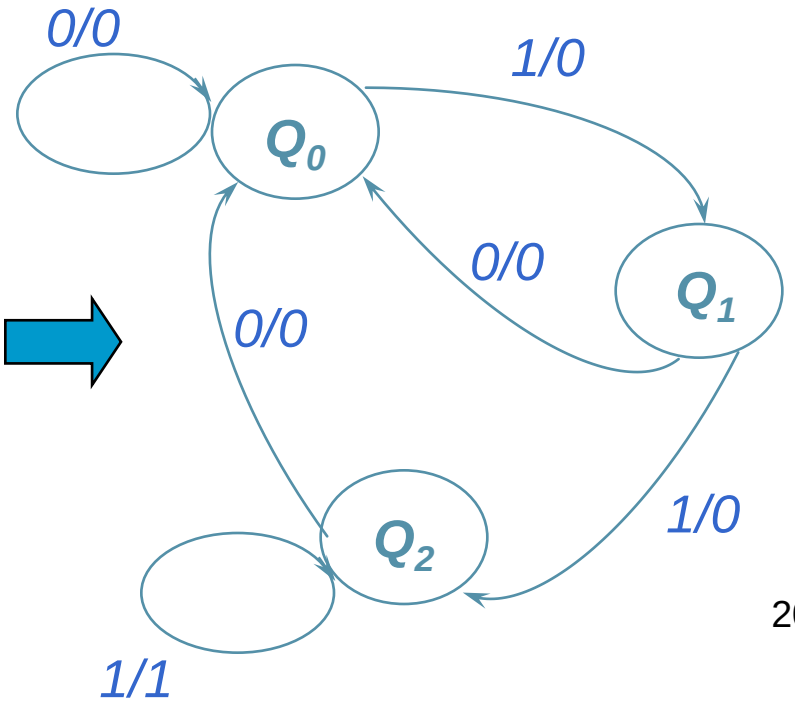
saída U = unlock $= Q_0\,\overline{Q_2}$

Diagrama de estado – Exemplo (Máquina Mealy)

- Exemplo:
- Projete um circuito que gere saída ‘1’ quando for observado 3 uns ‘1’s consecutivos na entrada. Nos demais casos a saída deve ser zero (‘0’).





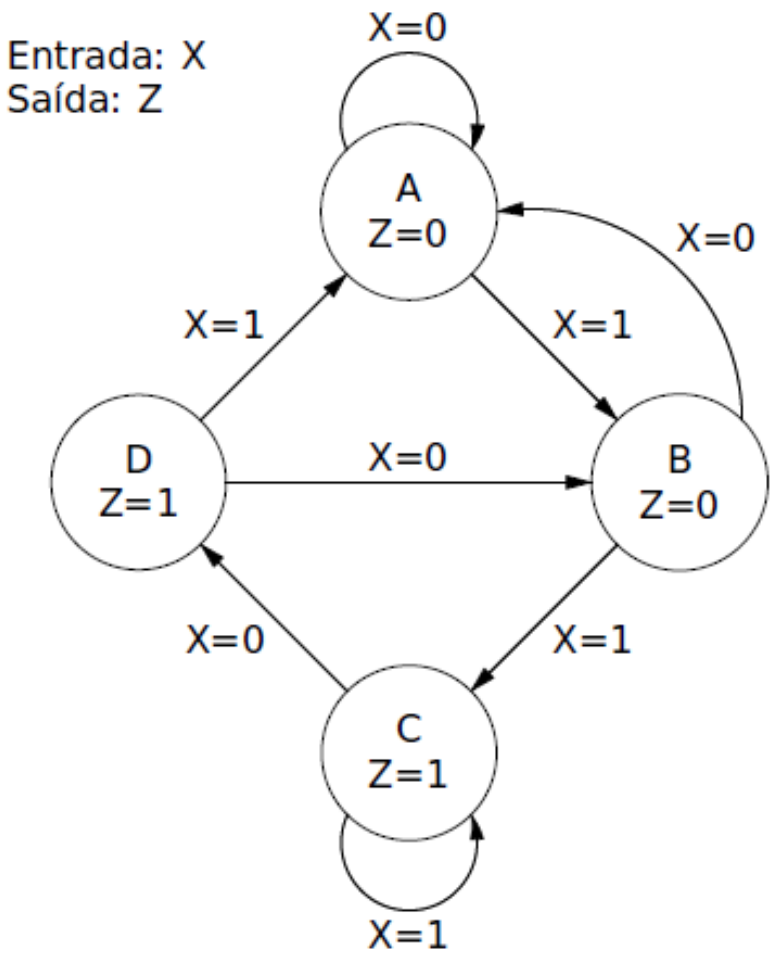


Máquina de Estado Finito Síncrono (Verilog)

```
module mef (clk, in_data, out_data);
input clk;
input [N:0] in_data; output [M:0] out_data;
reg [M:0] out_data;
reg [S:0] estado;

always @(negedge clk)
    case (estado)
        0 : begin output <= ......;   estado <= ......;  end
        1 : begin output <= ......;   estado <= ......;  end
         ......
        default : begin output <= ......;   estado <= ......;  end
     endcase
endmodule
```

Máquina de Moore

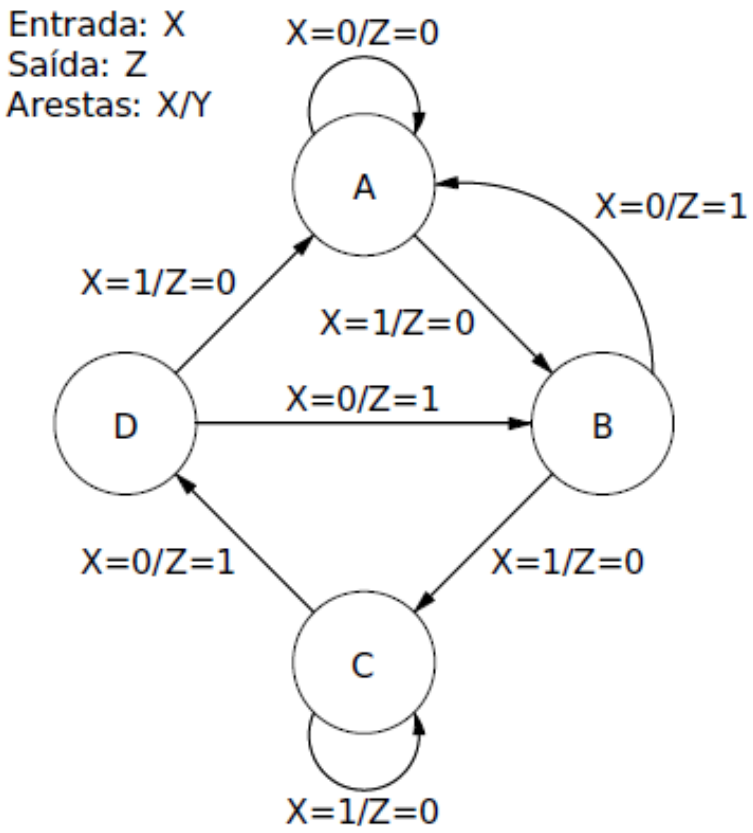


Sinal de *reset*: → A.

```
module moore(clk, X, Z, rst);
input clk, X, rst;
output Z; reg Z;
parameter[1:0] A=2'b00, B=2'b01,
               C=2'b10, D=2'b11;
reg [1:0] estado, prox_estado;
always @(posedge clk)
  if (rst) estado <= A;
  else estado <= prox_estado;

always @(estado or X) begin
  Z = (estado==C || estado==D);
  case(estado)
    A: if (X) prox_estado = B;
       else   prox_estado = A;
    B: if (X) prox_estado = C;
       else   prox_estado = A;
    C: if (X) prox_estado = C;
       else   prox_estado = D;
    D: if (X) prox_estado = A;
       else   prox_estado = B;
  endcase
end
endmodule
```

Máquina de Mealy




```
module mealy(clk, X, Z, rst);
input clk, X, rst;
output Z; reg Z;
parameter[1:0] A=2'b00, ...
reg [1:0] estado, prox_estado;
always @(posedge clk)
  if (rst) estado <= A;
  else estado <= prox_estado;
always @(estado or X)
  case(estado)
    A: if (X) begin
         prox_estado = B; Z=1'b0;
       end
       else begin
         prox_estado = A; Z=1'b0;
       end
    B: if (X) begin
         prox_estado = C; Z=1'b0;
       end
       else begin
         prox_estado = A; Z=1'b1;
       end
    C: ...   D: ...
  endcase
endmodule
```