PCS 3115

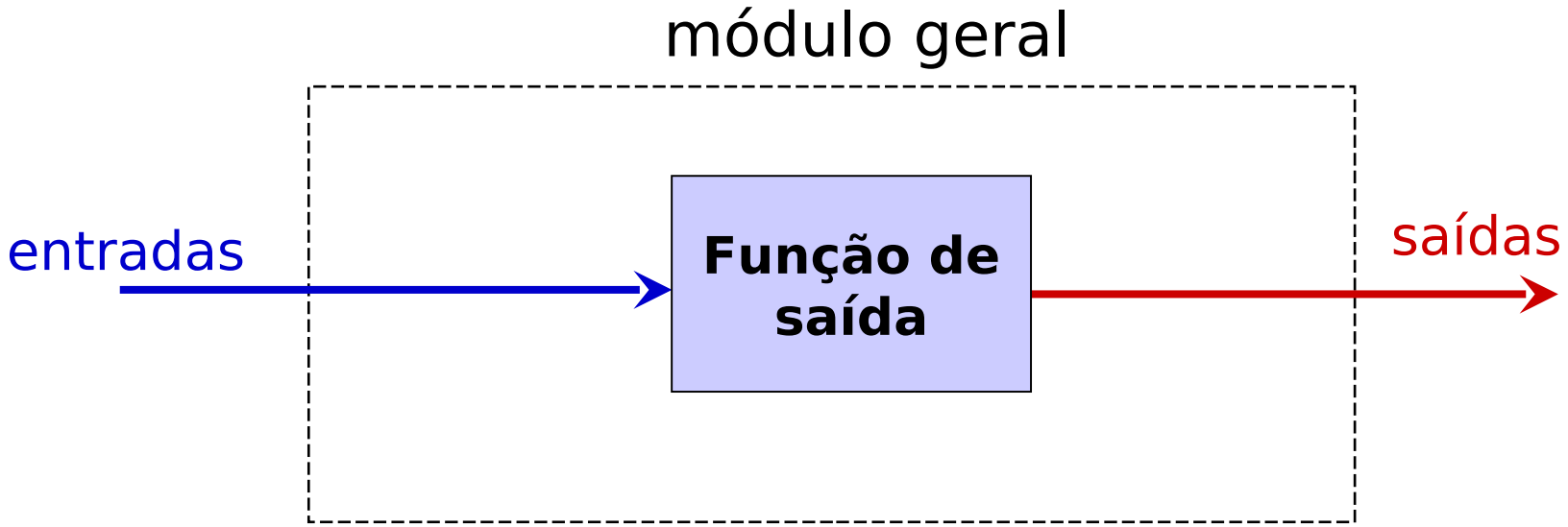
Sistemas Digitais I

Análise de Circuitos Sequenciais

Abril/2020

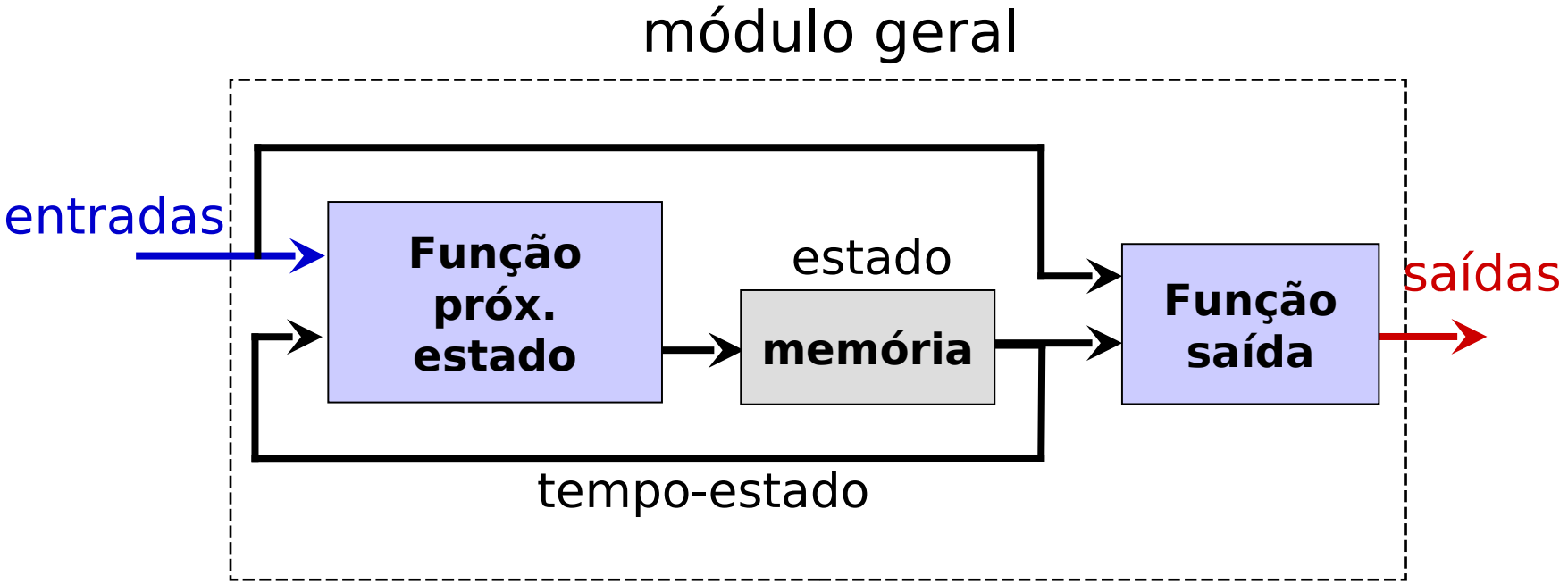
Combinatório vs. Sequencial

- Circuito **combinatório**: saída depende apenas das **entradas**



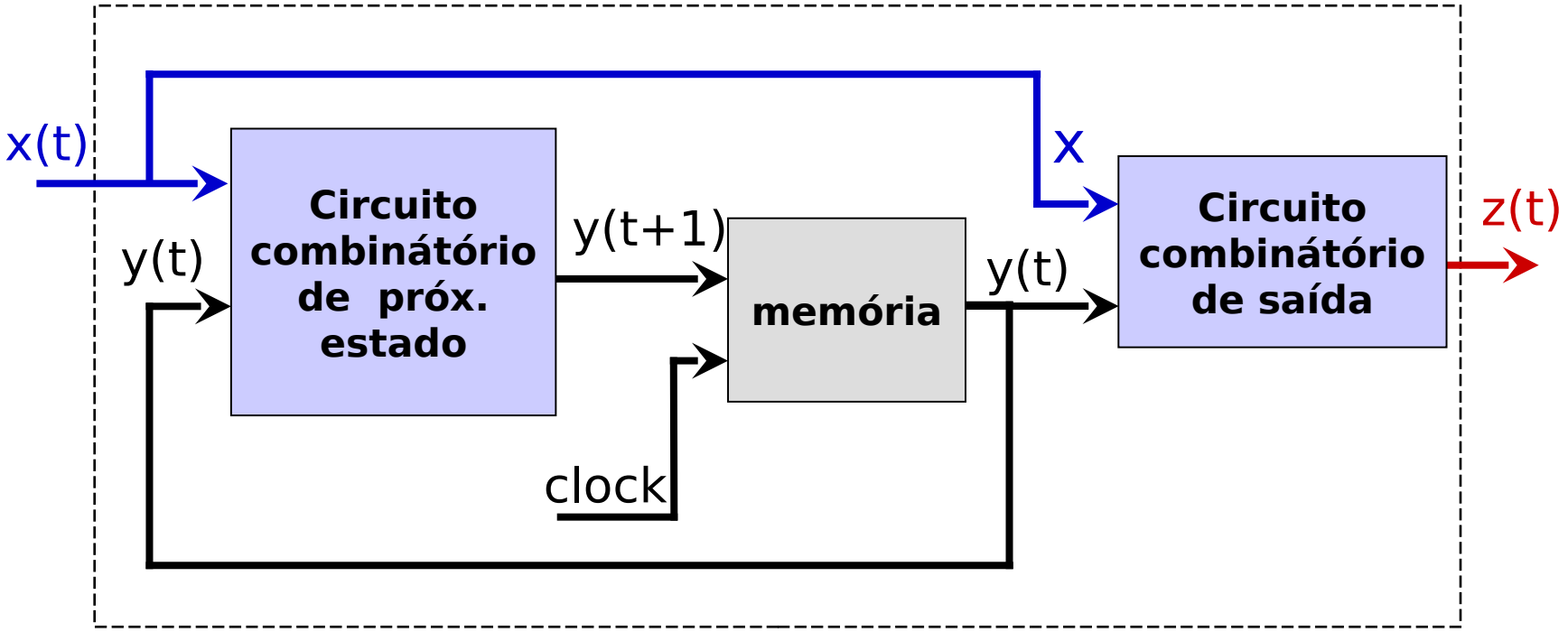
Combinatório vs. Sequencial

- Circuito **sequencial**: saída depende das **entradas** e do **estado atual**
  - "Estado": captura o histórico do circuito (memória)



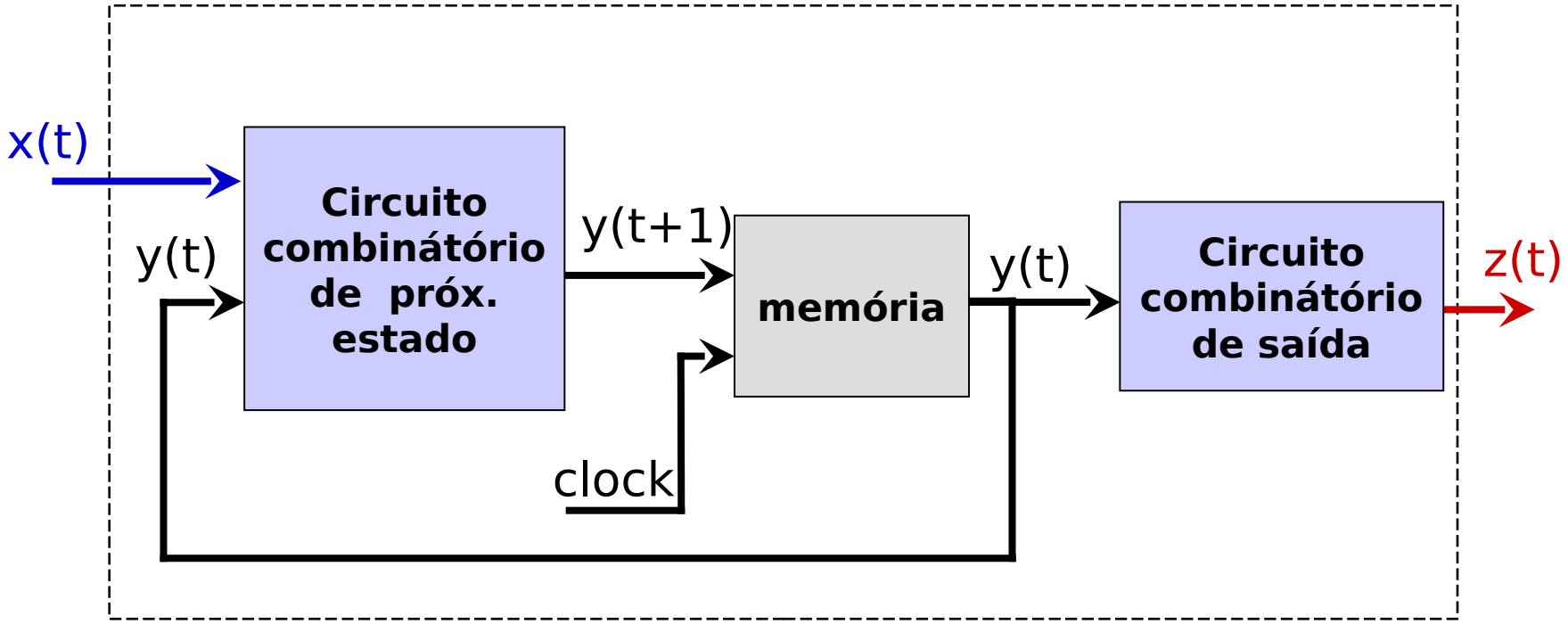
Circuitos sequenciais síncronos: Mealy

- **Saída z(t)**: depende do estado atual y(t) e da entrada atual x(t) ➔ z(t) = f(x(t), y(t))
- **Próximo estado y(t+1)**: idem ➔ y(t+1) = f(x(t), y(t))



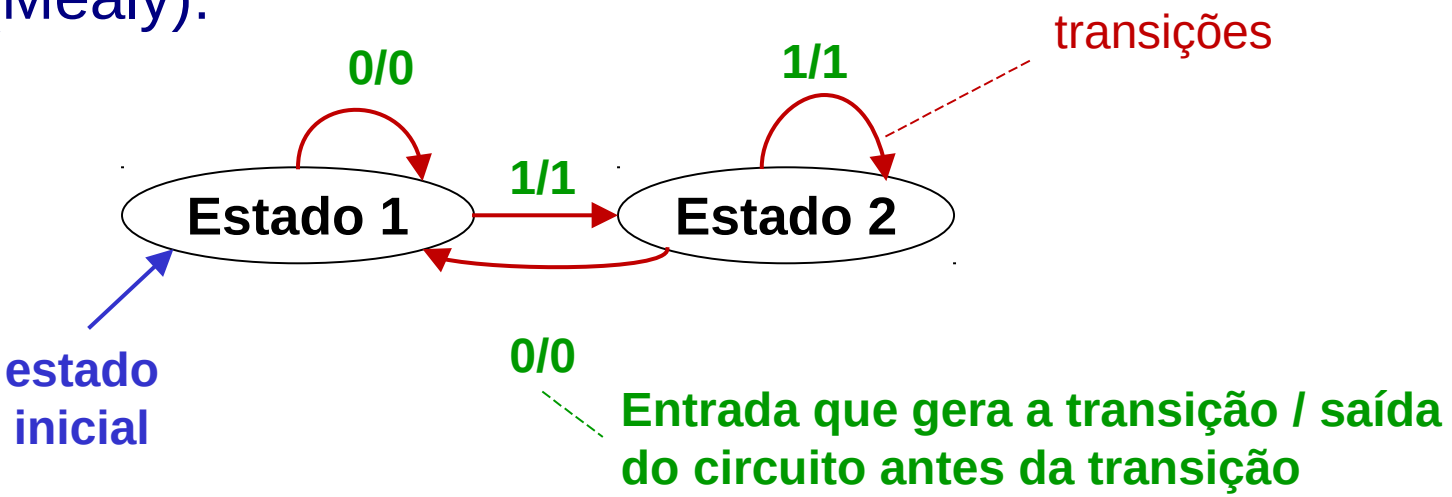
Circuitos sequenciais síncronos: Moore

- **Saída z(t)**: depende apenas do estado atual y(t)
- **Próximo estado y(t+1)**: depende do estado atual y(t) e da entrada atual x(t) ➔ y(t+1) = f(x(t), y(t))



Máquina de Estados Finita (MEF)

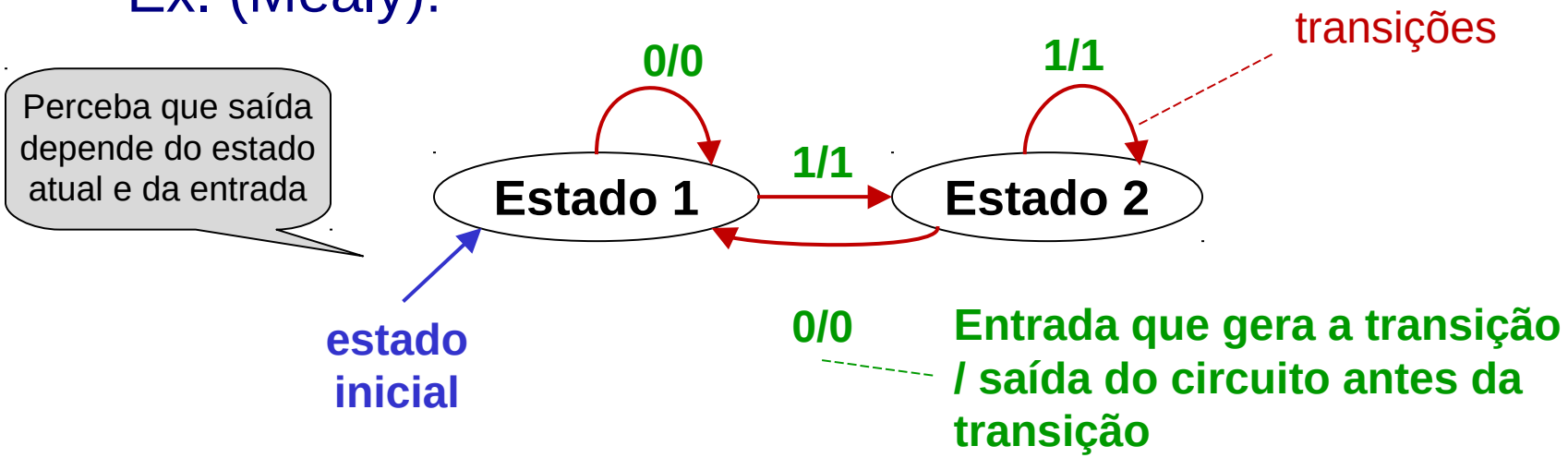
- Modelo usado para representar o comportamento de um circuito digital em função das entradas recebidas
  - Muito útil para representar circuitos sequenciais: captura todos estados e transições possíveis entre eles
- Ex. (Mealy):



- O que faz o circuito correspondente a essa MEF?

Máquina de Estados Finita (MEF)

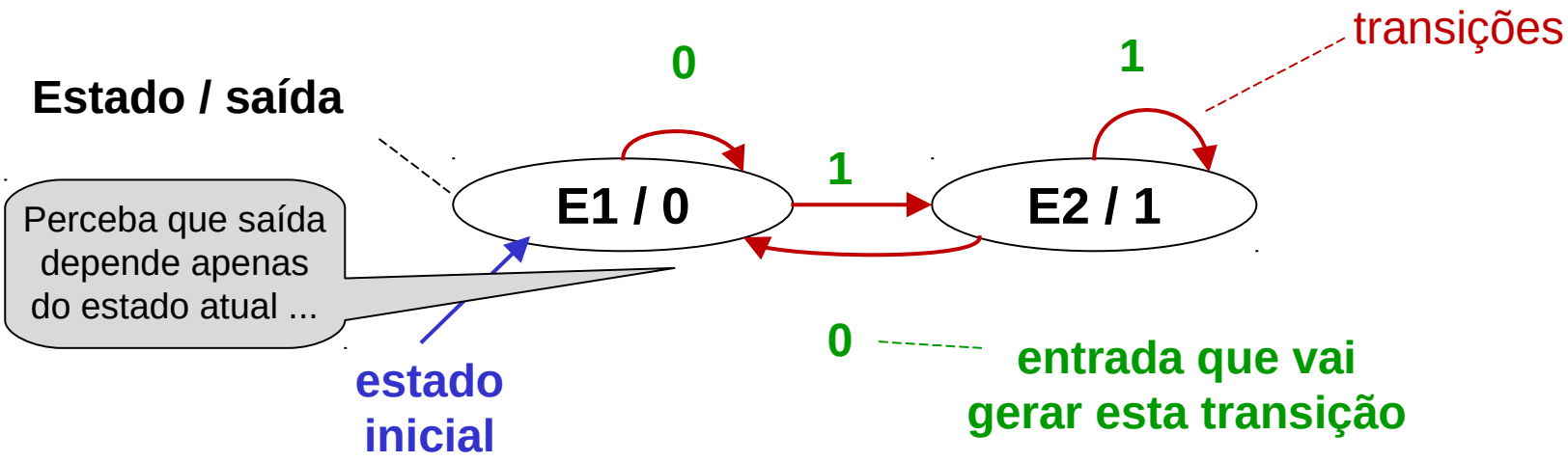
- Ex. (Mealy):



- O que faz o circuito correspondente a essa MEF?
  - Diz se a última entrada recebida foi 0 ou 1
  - Uma possível aplicação: em jogo, entrada corresponde ao comando “mover”; no Estado 1, o avatar não se move os braços, no Estado 2, ele move os braços

Máquina de Estados Finita (MEF)

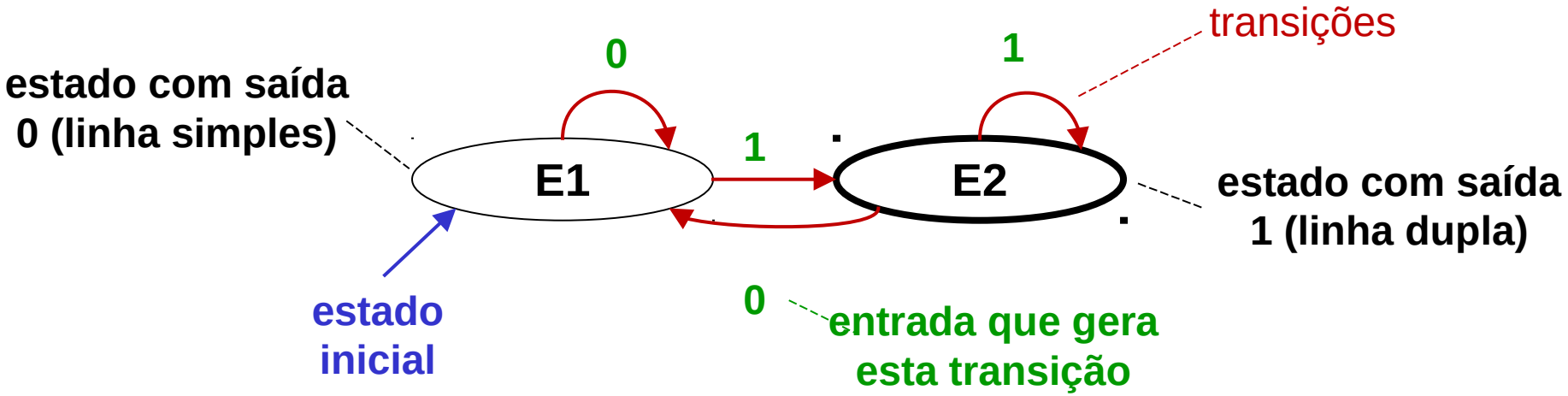
- Ex. (Moore):



- O que faz o circuito correspondente a essa MEF?
  - Diz se a última entrada recebida foi 0 ou 1
  - Uma possível aplicação: em jogo, entrada corresponde ao comando “mover”; no Estado 1, o avatar não se move os braços, no Estado 2, ele move os braços

Máquina de Estados Finita (MEF)

- Ex. (Moore): simplificação quando saída só tem 1 bit...



- O que faz o circuito correspondente a essa MEF?
  - Diz se a última entrada recebida foi 0 ou 1
  - Uma possível aplicação: em jogo, entrada corresponde ao comando “mover”; no Estado 1, o avatar não se move os braços, no Estado 2, ele move os braços

Máquina de Estados Finita (MEF)

- Mais formalmente: uma MEF M = (I, O, S, f, g,σ) consiste em:
  - Um conjunto finito I de símbolos de entrada
  - Um conjunto finito O de símbolos de saída
  - Um conjunto finito S de estados
  - Uma função próximo estado f: $S \times I \rightarrow S$
  - Uma função de saída g: $S \times I \rightarrow O$
  - Um estado inicial $\sigma \in S$

Máquina de Estados Finita (MEF)

- Ex.: M = (*I, O, S, f, g*,σ)
  - Conjunto finito *I* de símbolos de entrada ➔ I={a,b}
  - Conjunto finito *O* de símbolos de saída ➔ O={0,1}
  - Conjunto finito *S* de estados ➔ *S*={σ_0, σ_1}
  - Função próximo estado *f: S* x *I* $\rightarrow$ *S* ➔ vide abaixo
  - Função de saída *g: S* x *I* $\rightarrow$ *O* ➔ vide abaixo
  - Estado inicial $\sigma \in S$ ➔ $\sigma = \sigma_0$

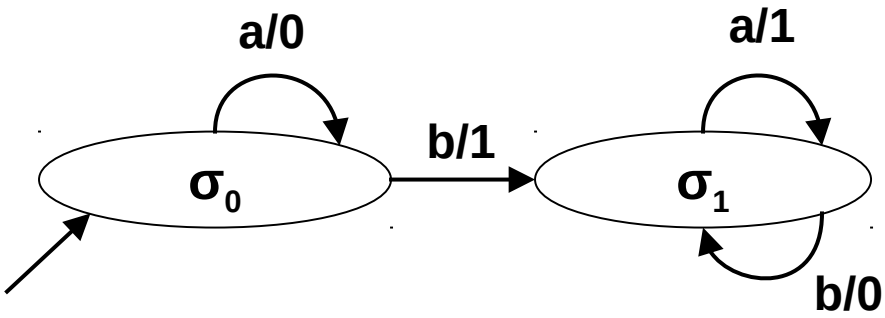
	f		g	
S \ I	a	b	a	b
σ_0	σ_0	σ_1	0	1
σ_1	σ_1	σ_1	1	0

$$f(\sigma_0,\text{a}) = \sigma_0 \quad g(\sigma_0,\text{a}) = 0$$
$$f(\sigma_0,\text{b}) = \sigma_1 \quad g(\sigma_0,\text{b}) = 1$$
$$f(\sigma_1,\text{a}) = \sigma_1 \quad g(\sigma_1,\text{a}) = 1$$
$$f(\sigma_1,\text{b}) = \sigma_1 \quad g(\sigma_1,\text{b}) = 0$$

Máquina de Estados Finita (MEF)

- Ex.: M = (*I, O, S, f, g*,σ) -- Mealy

Saída é 1 se receber ao menos 1 “b”, ou se depois disso a última entrada tiver sido “a” (regex: a*b(a|b*a)*)



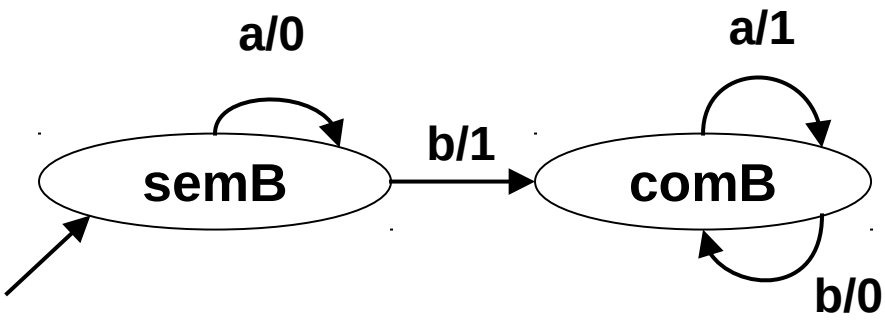
	f		g	
S \ I	a	b	a	b
σ_0	σ_0	σ_1	0	1
σ_1	σ_1	σ_1	1	0

$$f(\sigma_0,a)=\sigma_0 \quad g(\sigma_0,a)=0$$
$$f(\sigma_0,b)=\sigma_1 \quad g(\sigma_0,b)=1$$
$$f(\sigma_1,a)=\sigma_1 \quad g(\sigma_1,a)=1$$
$$f(\sigma_1,b)=\sigma_1 \quad g(\sigma_1,b)=0$$

Máquina de Estados Finita (MEF)

- Ex.: M = (I, O, S, f, g,σ) -- Mealy

Saída é 1 se receber ao menos 1 “b”, ou se depois disso a última entrada tiver sido “a” (regex: a*b(a|b*a)*)



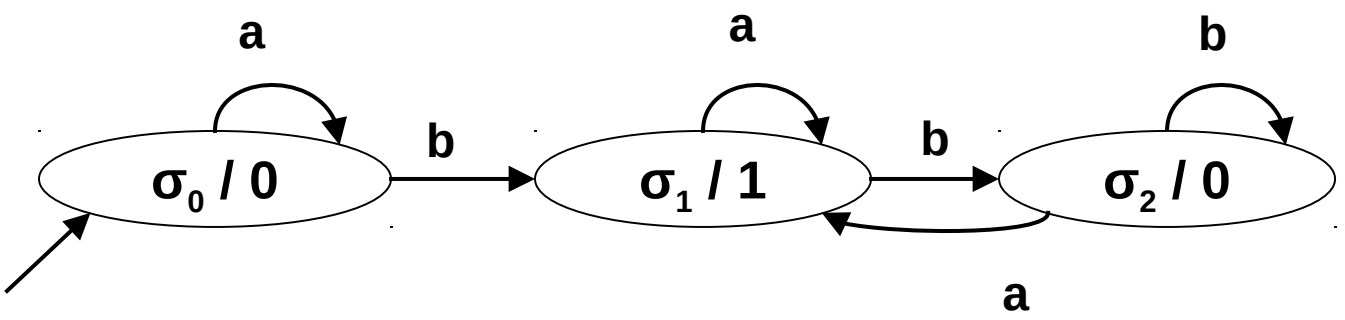
σ_0 = “semB”
σ_1 = “comB”

	f		g	
S \ I	a	b	a	b
σ_0	σ_0	σ_1	0	1
σ_1	σ_1	σ_1	1	0

$$f(\sigma_0,a)=\sigma_0 \quad g(\sigma_0,a)=0$$
$$f(\sigma_0,b)=\sigma_1 \quad g(\sigma_0,b)=1$$
$$f(\sigma_1,a)=\sigma_1 \quad g(\sigma_1,a)=1$$
$$f(\sigma_1,b)=\sigma_1 \quad g(\sigma_1,b)=0$$

Máquina de Estados Finita (MEF)

- Ex.: M = (*I*, *O*, *S*, *f*, *g*,σ) -- Moore



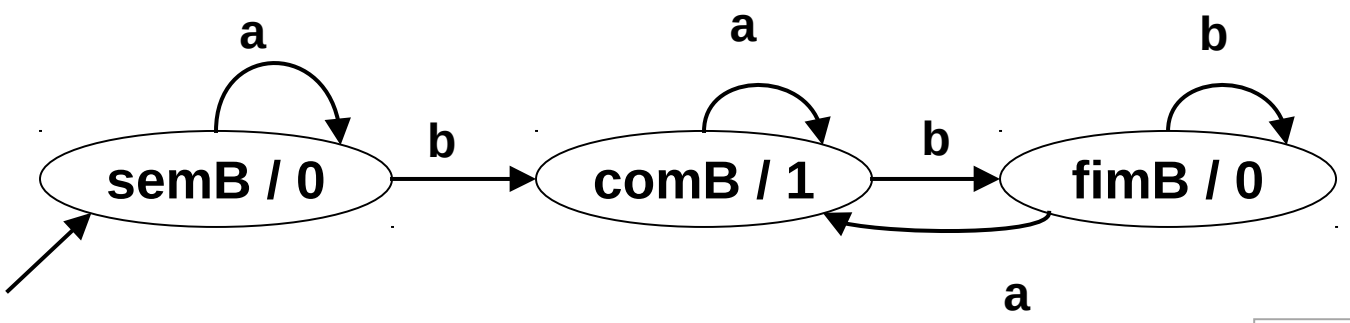
S \ I	f: a	f: b	g
σ_0	σ_0	σ_1	0
σ_1	σ_1	σ_2	1
σ_2	σ_1	σ_2	0

$$f(\sigma_0,\mathrm{a}) = \sigma_0$$
$$f(\sigma_0,\mathrm{b}) = \sigma_1$$
$$f(\sigma_1,\mathrm{a}) = \sigma_1$$
$$f(\sigma_1,\mathrm{b}) = \sigma_2$$
$$f(\sigma_2,\mathrm{a}) = \sigma_1$$
$$f(\sigma_2,\mathrm{b}) = \sigma_2$$

$$g(\sigma_0) = 0 \qquad g(\sigma_1) = 1 \qquad g(\sigma_2) = 0$$

Máquina de Estados Finita (MEF)

- Ex.: M = (*I*, *O*, *S*, *f*, *g*,σ) -- Moore



σ_0 = “semB”
σ_1 = “comB”
σ_2 = “fimB”

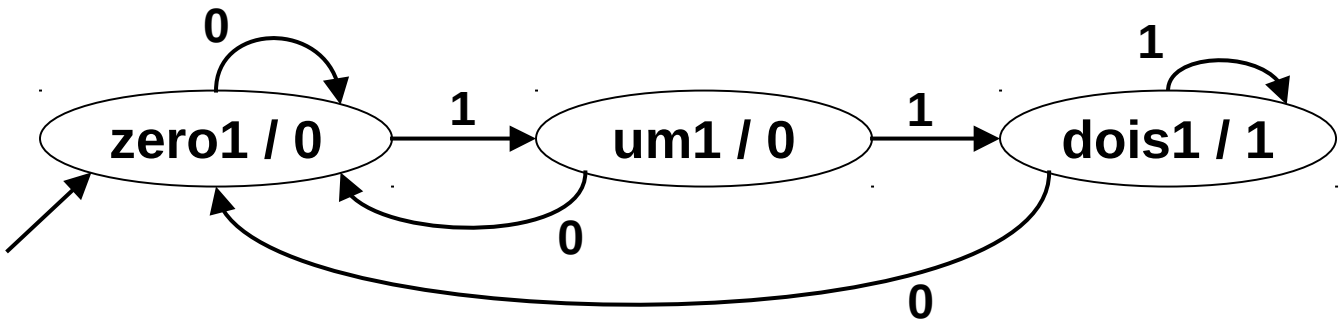
	f		g
S \ I	a	b	
σ_0	σ_0	σ_1	0
σ_1	σ_1	σ_2	1
σ_2	σ_1	σ_2	0

$$f(\sigma_0,\mathrm{a}) = \sigma_0$$
$$f(\sigma_0,\mathrm{b}) = \sigma_1$$
$$f(\sigma_1,\mathrm{a}) = \sigma_1$$
$$f(\sigma_1,\mathrm{b}) = \sigma_2$$
$$f(\sigma_2,\mathrm{a}) = \sigma_1$$
$$f(\sigma_2,\mathrm{b}) = \sigma_2$$

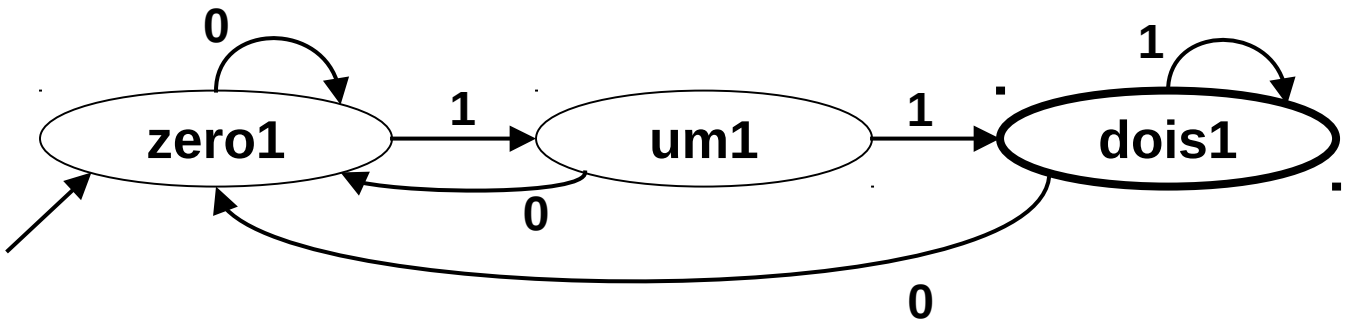
$$g(\sigma_0) = 0$$
$$g(\sigma_1) = 1$$
$$g(\sigma_2) = 0$$

Exercício Resolvido

- Projetar máquina de estados finitos que reconhece se as últimas entradas foram dois ou mais UNS consecutivos. Adotar solução do tipo Moore.



(alternativa: 1 bit de saída apenas

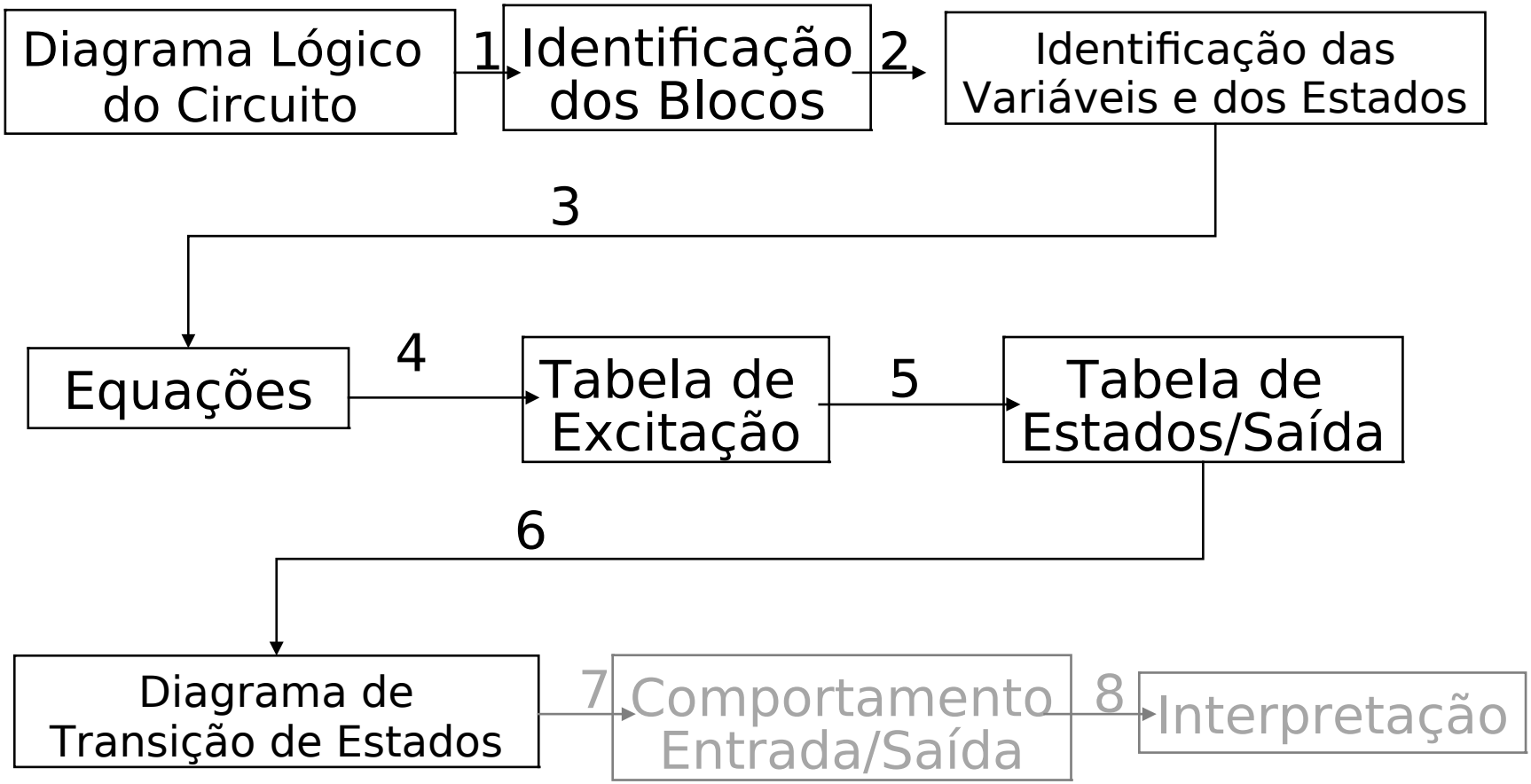


Análise de circuitos sequenciais

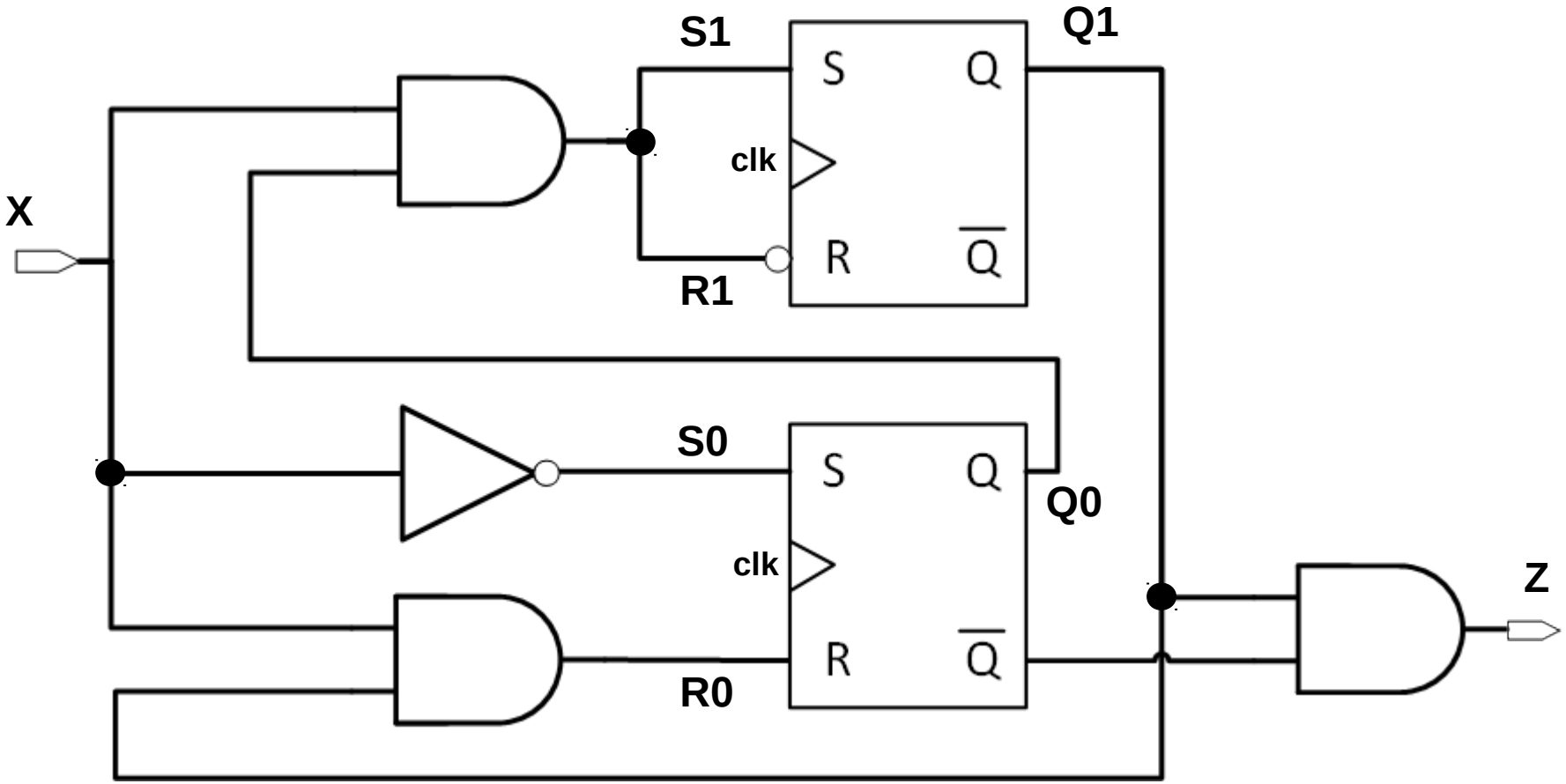
- Dado um circuito sequencial, descrever seu comportamento: determinar a **MEF correspondente**
- Premissas adotadas:
  - A memória é constituída por flip-flops sensíveis a borda.
  - As entradas podem mudar simultaneamente, mas ficam estáveis durante a borda de atuação do clock.
  - A frequência do clock é tal que os sinais internos já estão estabilizados na borda de interesse.
- Pode ser feita para Mealy ou Moore
  - Vamos nos concentrar em **Moore** nos slides a seguir

Análise de circuitos sequenciais

- Passo a passo

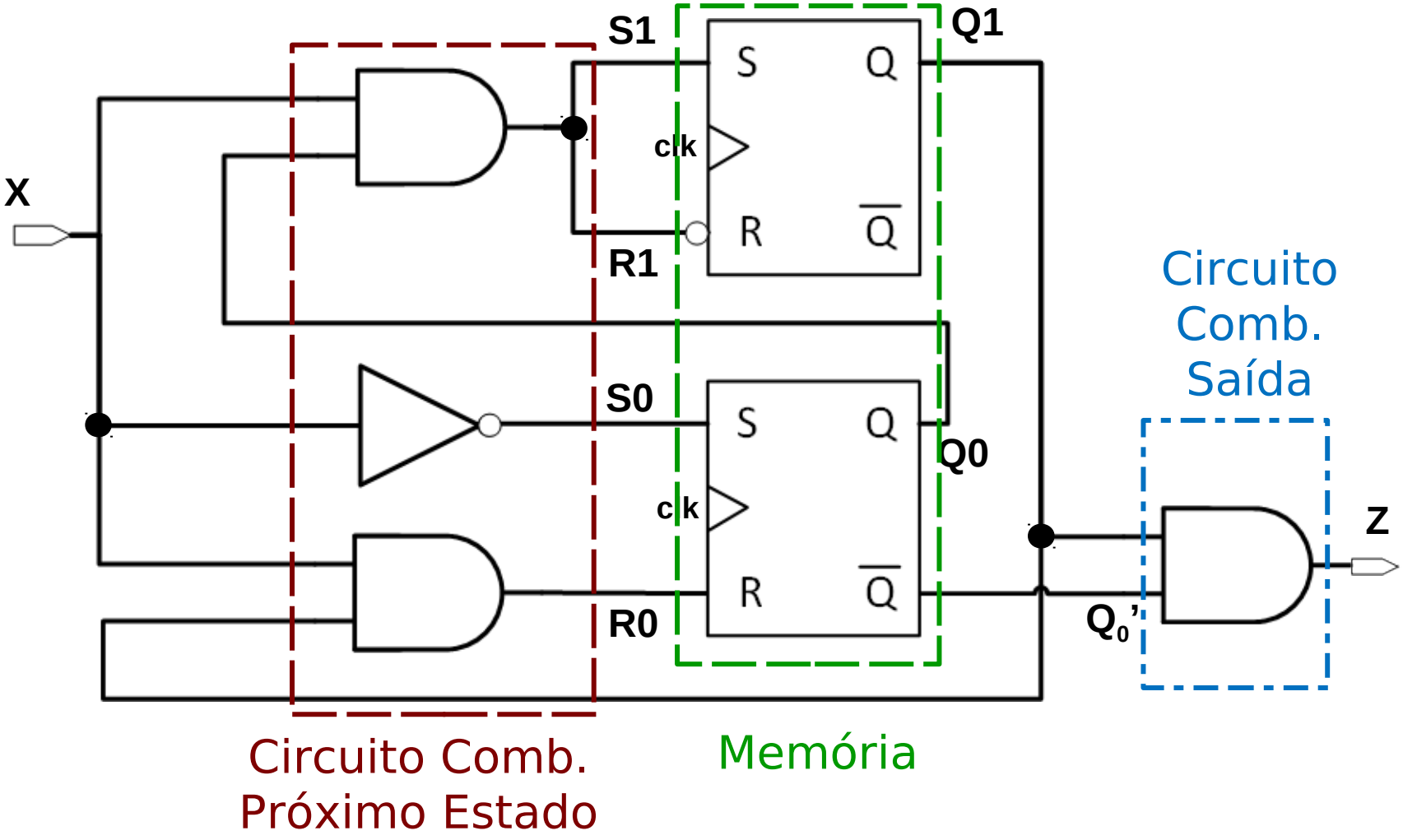


Análise: Exemplo (Moore)



Análise: Exemplo (Moore)

1) Identificação dos blocos



Análise: Exemplo (Moore)

2) Identificação das Variáveis e Estados

- entradas: X
- saídas: Z
- variáveis de excitação: S_1, R_1, S_0, R_0
- variáveis de estado: Q_1, Q_0

Com duas variáveis de estado, obtemos 4 estados, designados por A, B, C, D:

s	Q_1	Q_0
A	0	0
B	0	1
C	1	1
D	1	0

Análise: Exemplo (Moore)

3) Equações

- Variáveis de excitação:
  - $S_1 = X \bullet Q_0$
  - $R_1 = (X \bullet Q_0)'$
  - $S_0 = X'$
  - $R_0 = X \bullet Q_1$

- Saídas
  - $Z = Q_1 \bullet Q_0'$

Análise: Exemplo (Moore)

4) Tabela de Excitação: entradas na memória (R_1S_1 R_0S_0), e saídas correspondentes no instante t

Q_1Q_0 \ X	0	1	Z
0 0	10 01	10 00	0
0 1	10 01	01 00	0
1 1	10 01	01 10	0
1 0	10 01	10 10	1

$(R_1S_1\ R_0S_0)^t$ $(Z)^t$

Análise: Exemplo (Moore)

5) Tabela de Estados/Saídas: próximos estados (Q_1Q_0 no instante t+1) e saídas no instante t

Q_1Q_0 \ X	0	1	Z
0 0	0 1	0 0	0
0 1	0 1	1 1	0
1 1	0 1	1 0	0
1 0	0 1	0 0	1

$(Q_1\ Q_0)^{t+1}$ $(Z)^t$

- Obs.: para flip-flops tipo RS (e qualquer outro, exceto D), as tabelas de excitação e de estados não são iguais.

Análise: Exemplo (Moore)

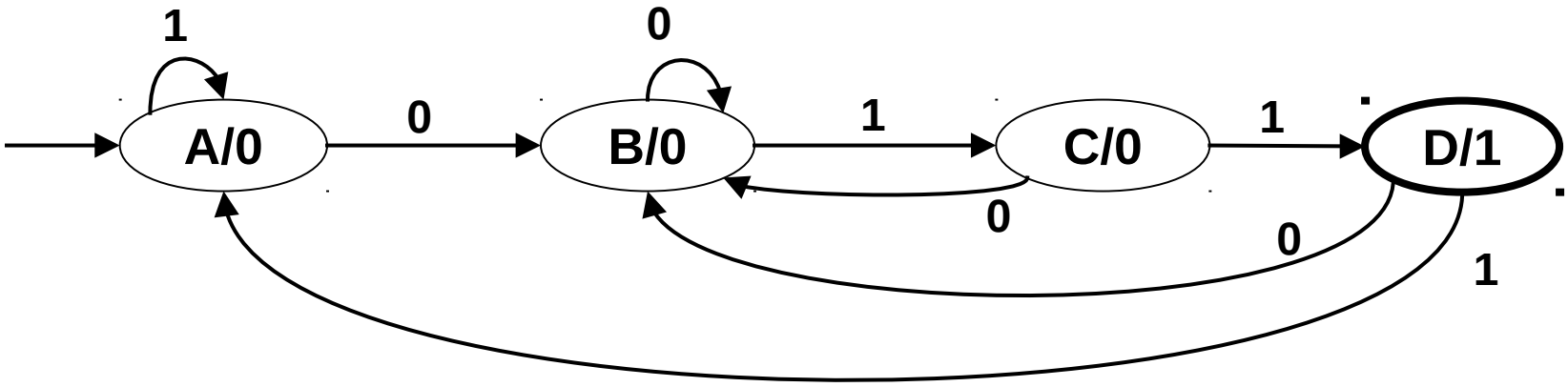
5b) Tabela de Estados/Saídas: agora com nomes

S^t \ X	0	1	Z
A	B	A	0
B	B	C	0
C	B	D	0
D	B	A	1

$(S)^{t+1}$ **$(Z)^t$**

Análise: Exemplo (Moore)

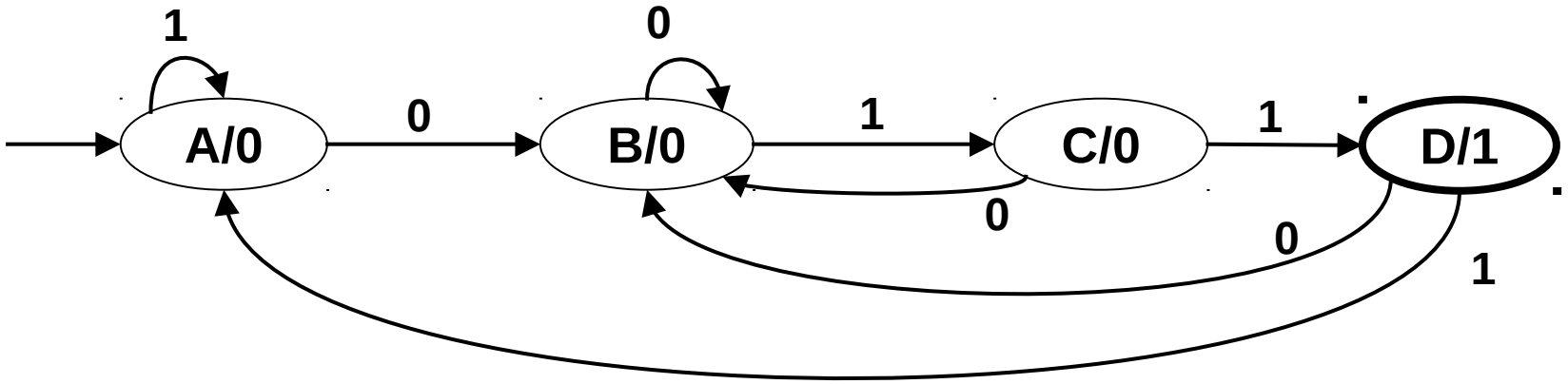
6) Diagrama de Transição de Estados



- Obs.: estado inicial arbitrário (em geral, 00)

Análise: Exemplo (Moore)

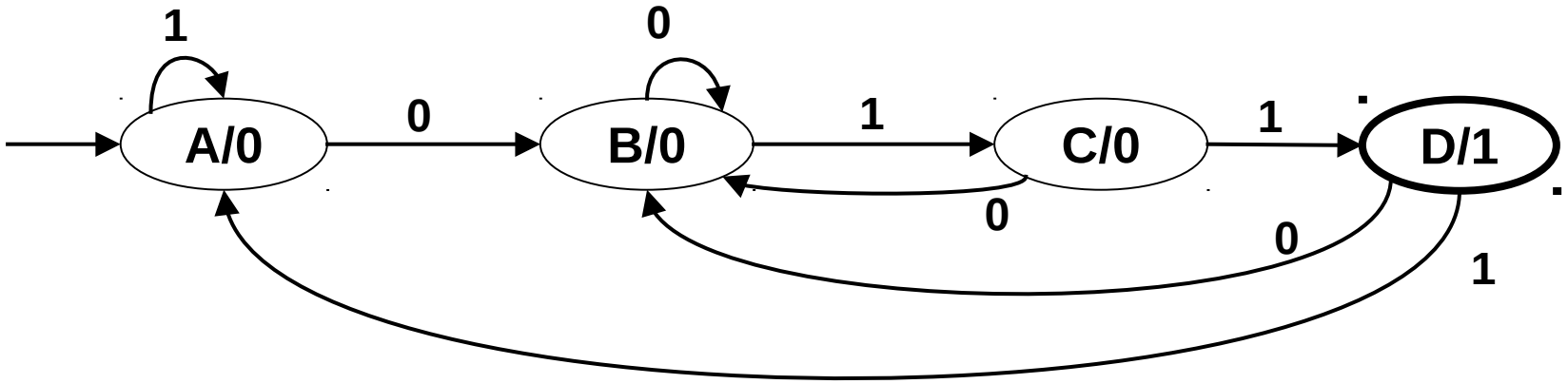
7) Comportamento Entrada/Saída



X_1X_0	--	1	0	0	1	0	1	1	0	1	1	1	0	1	1
Estado	A	A	B	B	C	B	C	D	B	C	D	A	B	C	D
Saída	0	0	0	0	0	0	0	1	0	0	1	0	0	0	1

Análise: Exemplo (Moore)

8) Interpretação

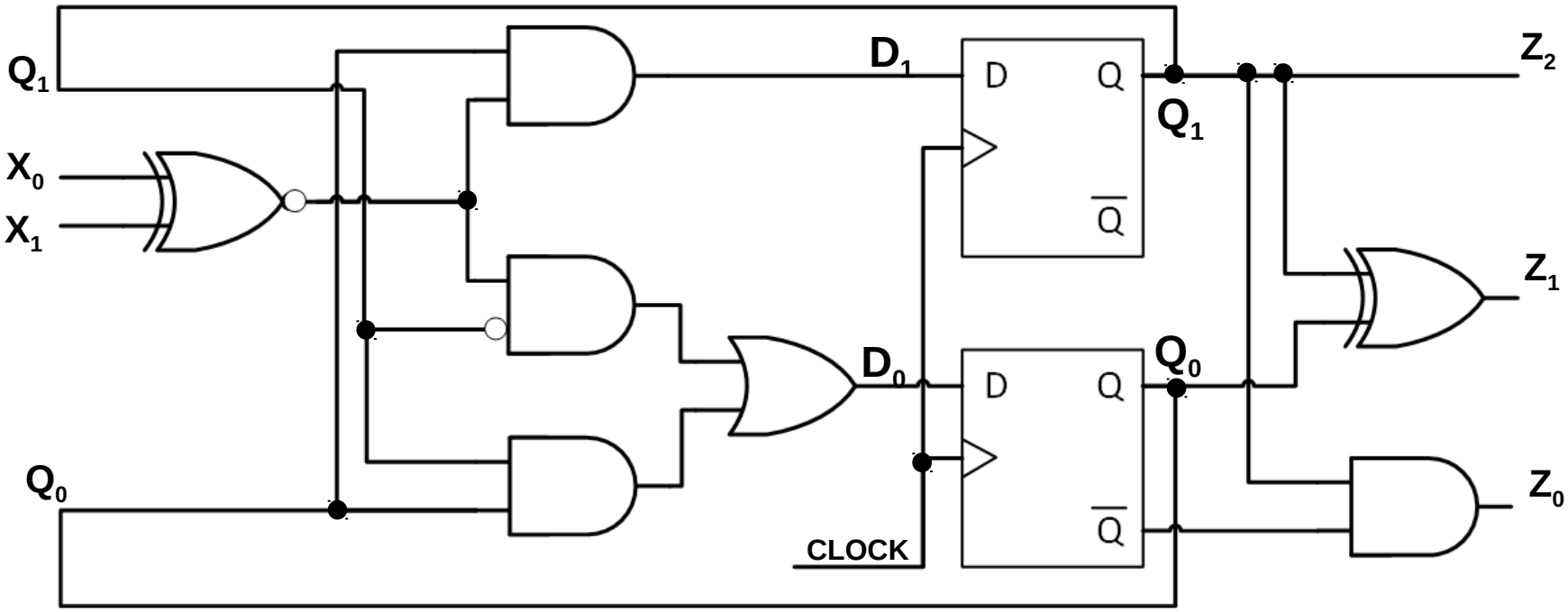


- Supondo estado inicial A:
  - Funciona como um **detector da sequência 011**

APÊNDICE

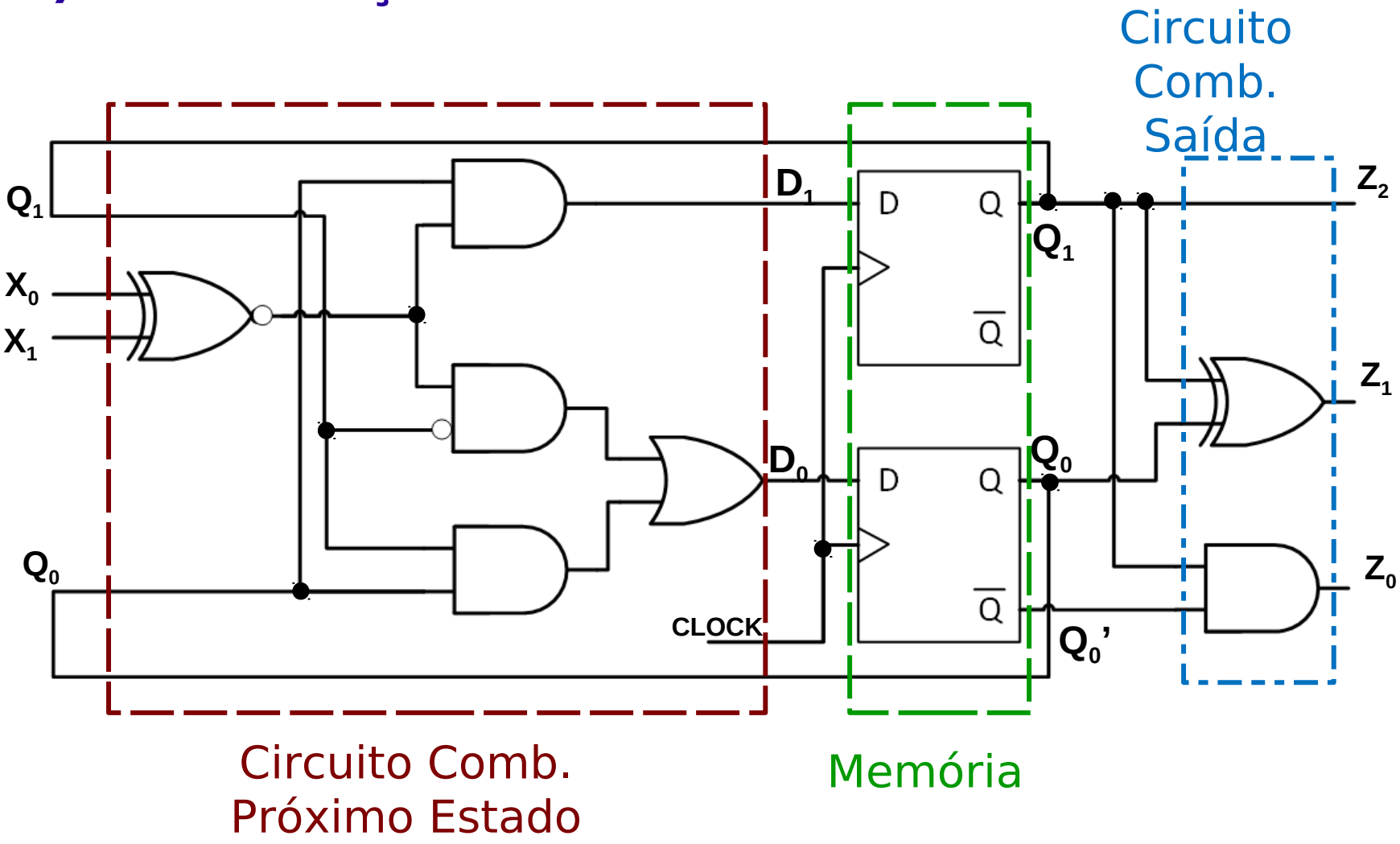
Outro Exemplo de Análise de MEF

Análise: Exemplo (Moore)



Análise: Exemplo (Moore)

1) Identificação dos blocos



Análise: Exemplo (Moore)

2) Identificação das Variáveis e Estados

- entradas: X_1, X_0
- saídas: Z_2, Z_1, Z_0
- variáveis de excitação: D_1, D_0
- variáveis de estado: Q_1, Q_0

Com duas variáveis de estado, obtemos 4 estados, designados por A, B, C, D:

s	Q_1	Q_0
A	0	0
B	0	1
C	1	1
D	1	0

Análise: Exemplo (Moore)

3) Equações

- Variáveis de excitação:
  - $D_1 = (X_1 \oplus X_0)' \bullet Q_0$
  - $D_0 = Q_1 \bullet Q_0 + (X_1 \oplus X_0)' \bullet Q_1'$
- Saídas
  - $Z_2 = Q_1$
  - $Z_1 = Q_1 \oplus Q_0$
  - $Z_0 = Q_1 \bullet Q_0'$

Análise: Exemplo (Moore)

4) Tabela de Excitação: entradas na memória (D_1D_0), e saídas correspondentes no instante t

Q_1Q_0 \ X_1X_0	0 0	0 1	1 1	1 0	$Z_2Z_1Z_0$
0 0	0 1	0 0	0 1	0 0	0 0 0
0 1	1 1	0 0	1 1	0 0	0 1 0
1 1	1 1	0 1	1 1	0 1	1 0 0
1 0	0 0	0 0	0 0	0 0	1 1 1

$(D_1D_0)^t$ $(Z_2Z_1Z_0)^t$

Análise: Exemplo (Moore)

5) Tabela de Estados/Saídas: próximos estados (Q_1Q_0 no instante t+1) e saídas no instante t

Q_1Q_0 \ X_1X_0	0 0	0 1	1 1	1 0	$Z_2Z_1Z_0$
0 0	0 1	0 0	0 1	0 0	0 0 0
0 1	1 1	0 0	1 1	0 0	0 1 0
1 1	1 1	0 1	1 1	0 1	1 0 0
1 0	0 0	0 0	0 0	0 0	1 1 1

$(Q_1Q_0)^{t+1}$ $(Z_2Z_1Z_0)^t$

- Obs.: para flip-flops tipo D, as tabelas de excitação e de estados são iguais. Para outros tipos (e.g., RS), não seria o caso

Análise: Exemplo (Moore)

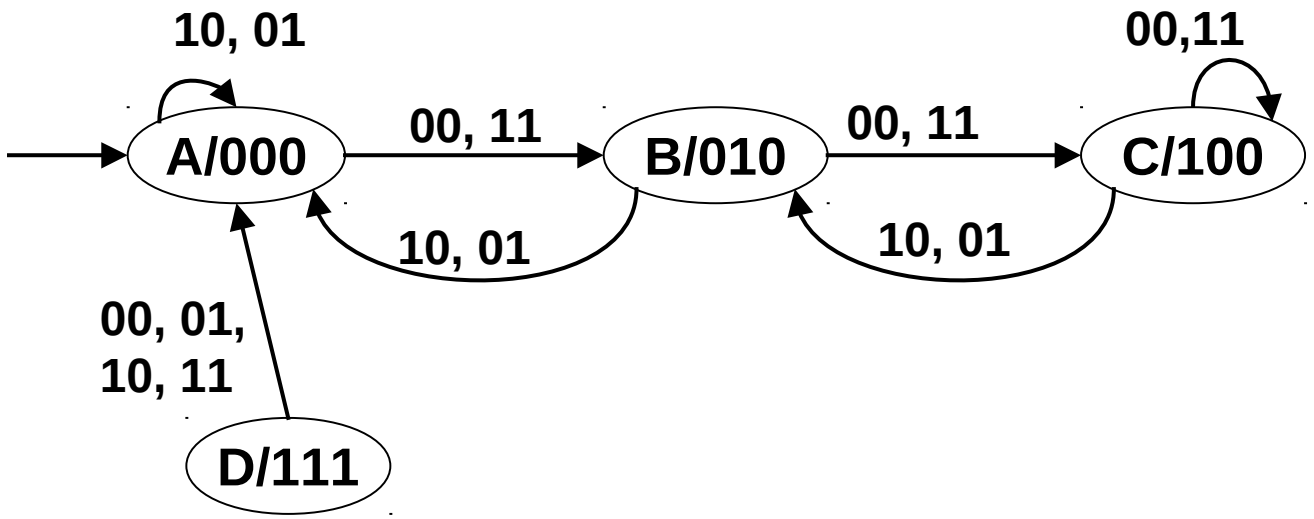
5b) Tabela de Estados/Saídas: agora com nomes

S^t \ X_1X_0	0 0	0 1	1 1	1 0	$Z_2Z_1Z_0$
A	B	A	B	A	0 0 0
B	C	A	C	A	0 1 0
C	C	B	C	B	1 0 0
D	A	A	A	A	1 1 1

$(S)^{t+1}$ **$(Z_2Z_1Z_0)^t$**

Análise: Exemplo (Moore)

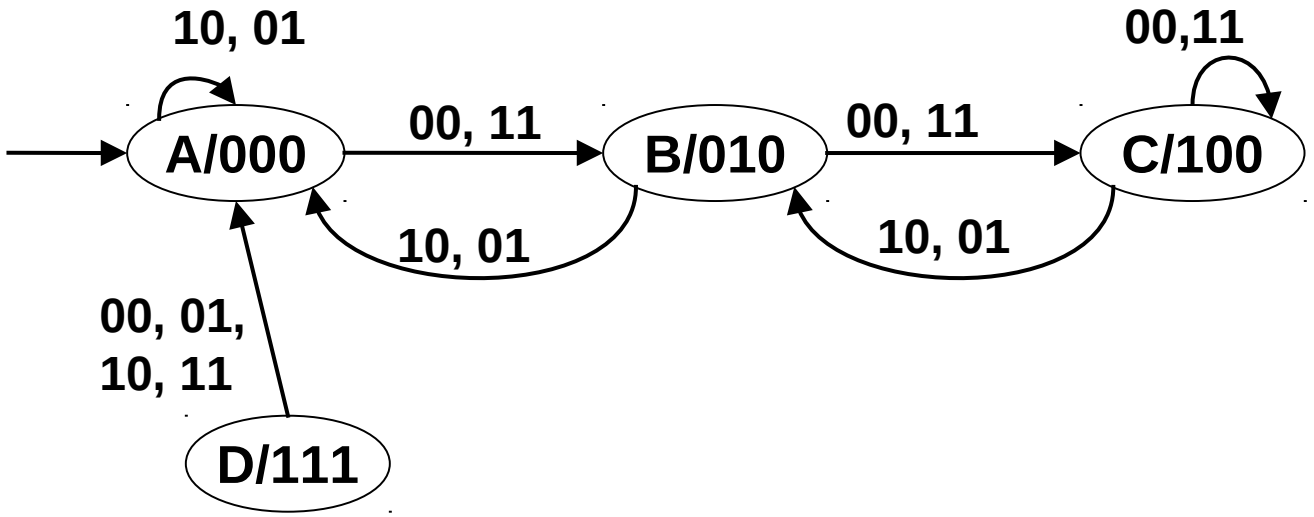
6) Diagrama de Transição de Estados



- Obs.: estado inicial arbitrário (em geral, 00)

Análise: Exemplo (Moore)

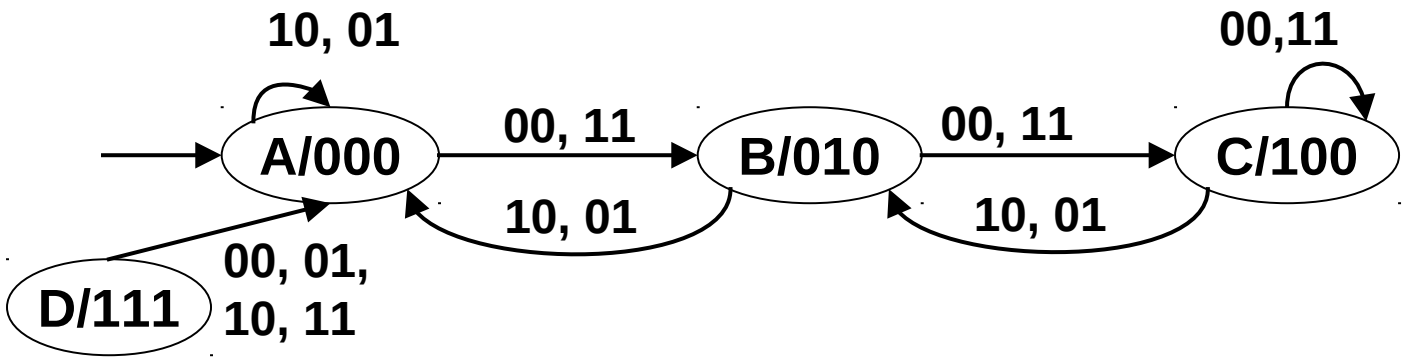
7) Comportamento Entrada/Saída



X_1X_0	--	01	11	00	11	01	00	01	10	01	00	00
Estado	A	A	B	C	C	B	C	B	A	A	B	C
Saída	000	000	010	100	100	010	100	010	000	000	010	100

Análise: Exemplo (Moore)

8) Interpretação



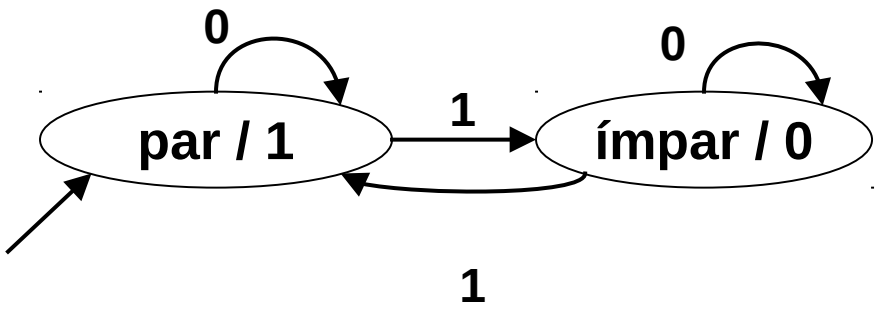
- Supondo estado inicial A:
  - Se entrada são 2 bits diferentes, nada muda
  - Quando entrada são 2 bits iguais, passa para estado que diz “2 bits iguais”; volta para estado anterior se entrarem bits diferentes
  - Se um 2o par de entrada contiver bits iguais, passa p/ estado de “4 bits iguais” e lá fica; volta p/ estado anterior se bits de entrada diferentes
  - ➔ A: “alarme” (discordância); B: “atenção” (houve alguma discordância); Estado C: “normal” (entradas concordam); D: inatingível (volta para A para qualquer entrada)

APÊNDICE

Exercícios -- MEF

Exercício Resolvido

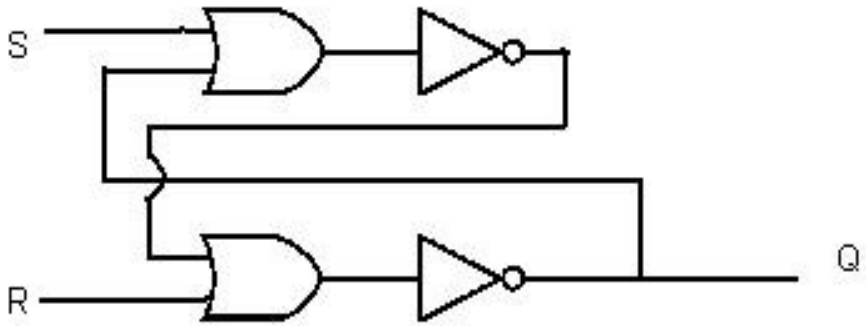
- Projetar máquina de estados finitos que calcula bit de paridade par: saída = 1 se houver número par de 1's na entrada; saída = 0 caso contrário. (modelo de Moore)



Exercício Resolvido

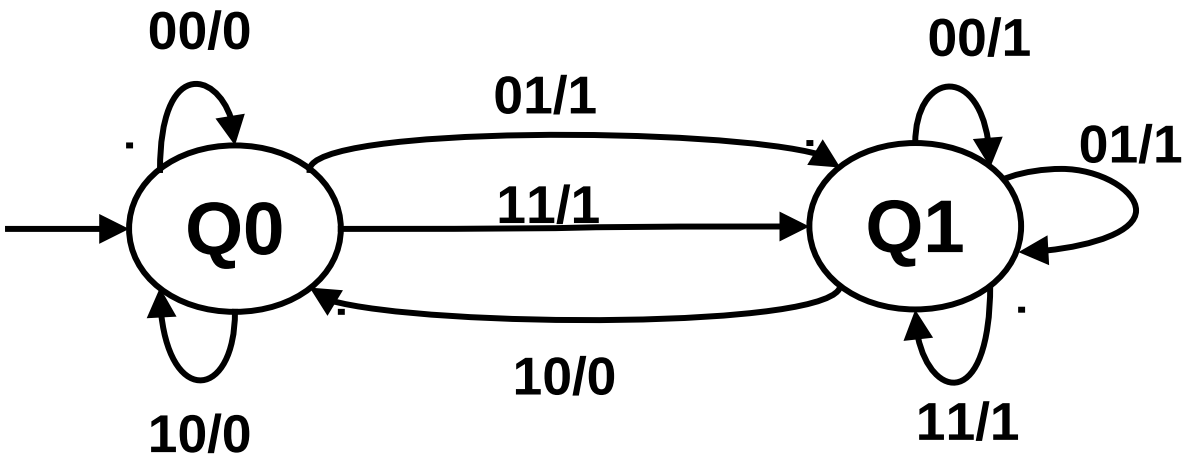
- Projetar um flip-flop RS como uma máquina de estados finitos.

R	S	Q
0	0	mantém valor anterior
0	1	1
1	0	0
1	1	1 (obs.: não permitido, mas pode ocorrer...)



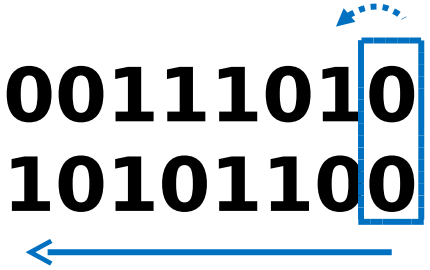
Máquinas de Estado – Introdução

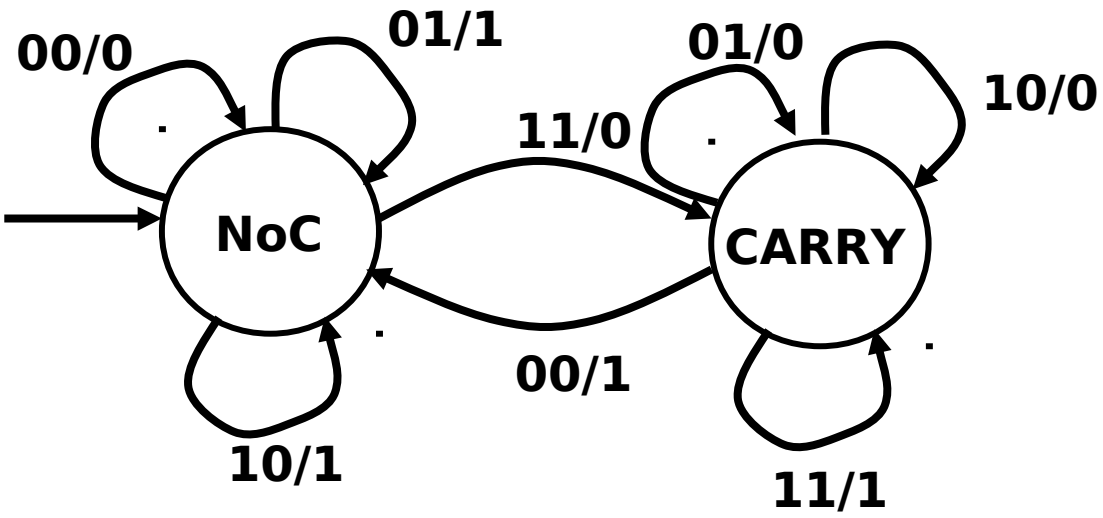
- Entrada: 2 bits (RS)
  - Nota: assumindo que começa com Q=0, mas poderíamos incluir um estado para “indefinido”



Exercício Resolvido

- Projetar máquina de estados finitos que calcula uma soma bit a bit, considerando o vai 1.
  - Entrada: par de bits, considerando “vai 1” do estado anterior
  - Saída: resultado da soma



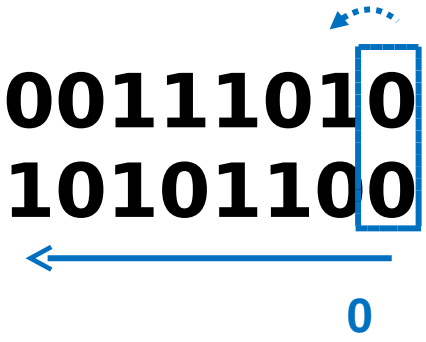


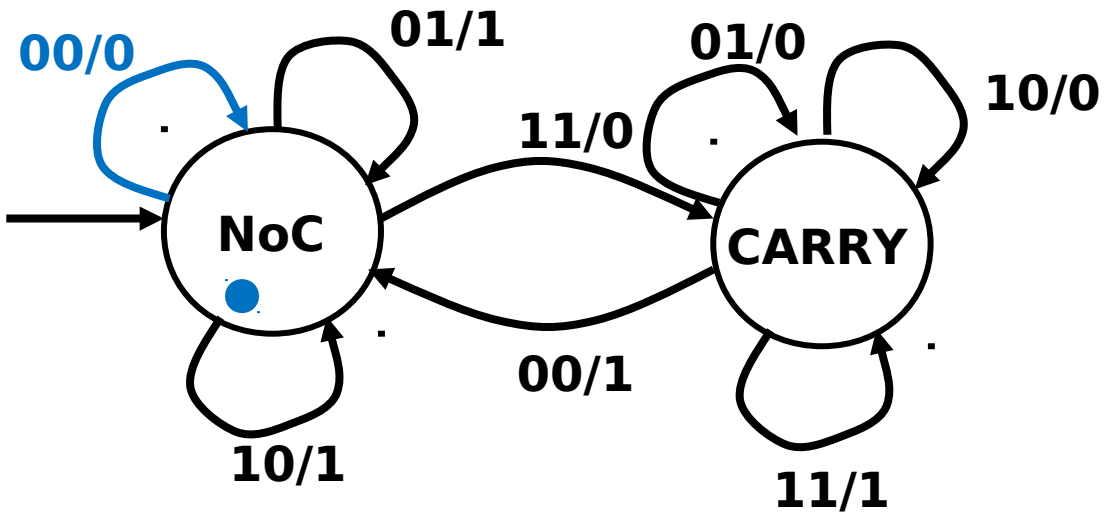
Obs.: Mealy é mais fácil...

Legenda: CARRY = Houve Carry;
NoC = Não houve Carry.

Exercício Resolvido

- Projetar máquina de estados finitos que calcula uma soma bit a bit, considerando o vai 1.
  - Entrada: par de bits, considerando “vai 1” do estado anterior
  - Saída: resultado da soma

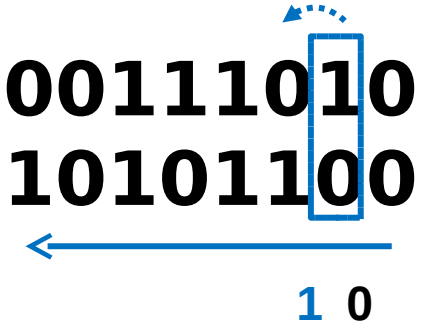


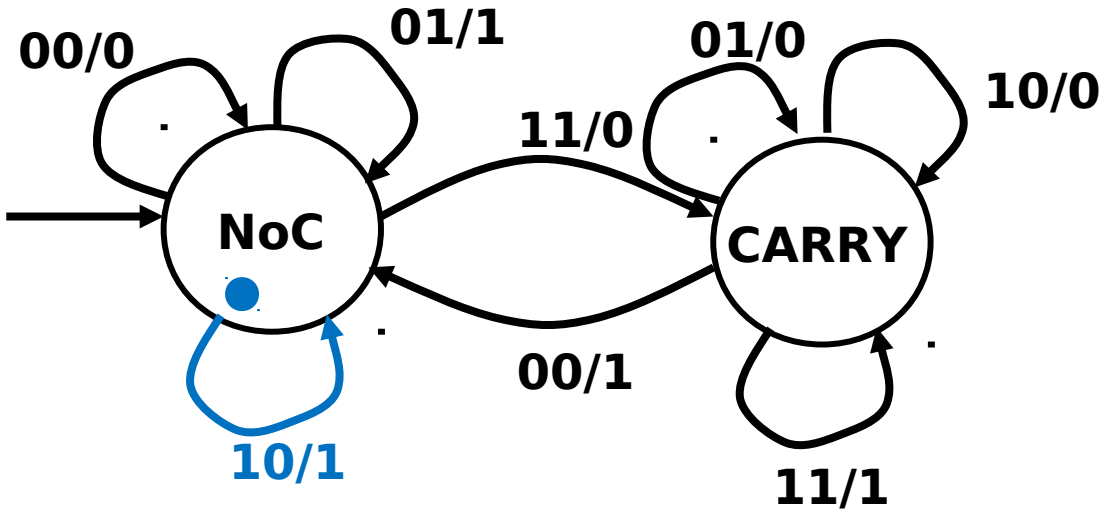


Obs.: Mealy é mais fácil...

Exercício Resolvido

- Projetar máquina de estados finitos que calcula uma soma bit a bit, considerando o vai 1.
  - Entrada: par de bits, considerando “vai 1” do estado anterior
  - Saída: resultado da soma

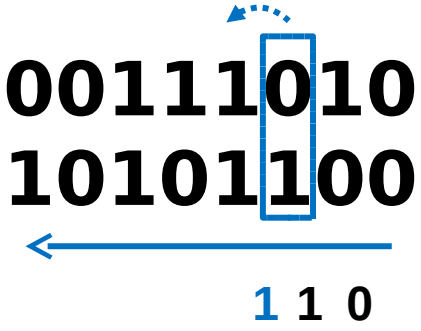


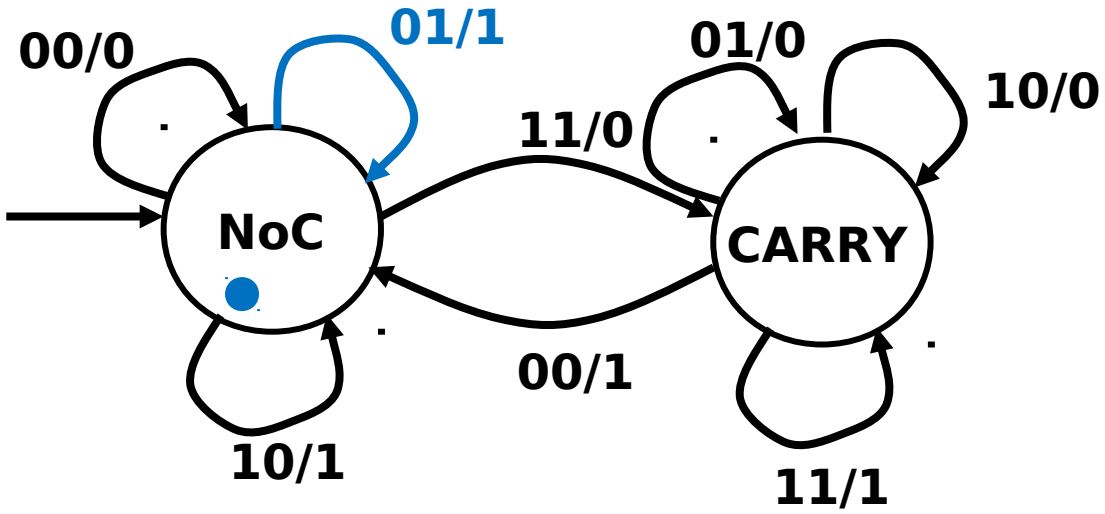


Obs.: Mealy é mais fácil...

Exercício Resolvido

- Projetar máquina de estados finitos que calcula uma soma bit a bit, considerando o vai 1.
  - Entrada: par de bits, considerando “vai 1” do estado anterior
  - Saída: resultado da soma

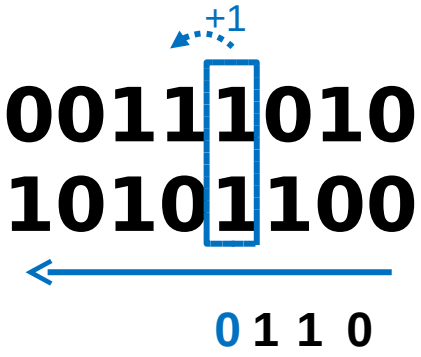


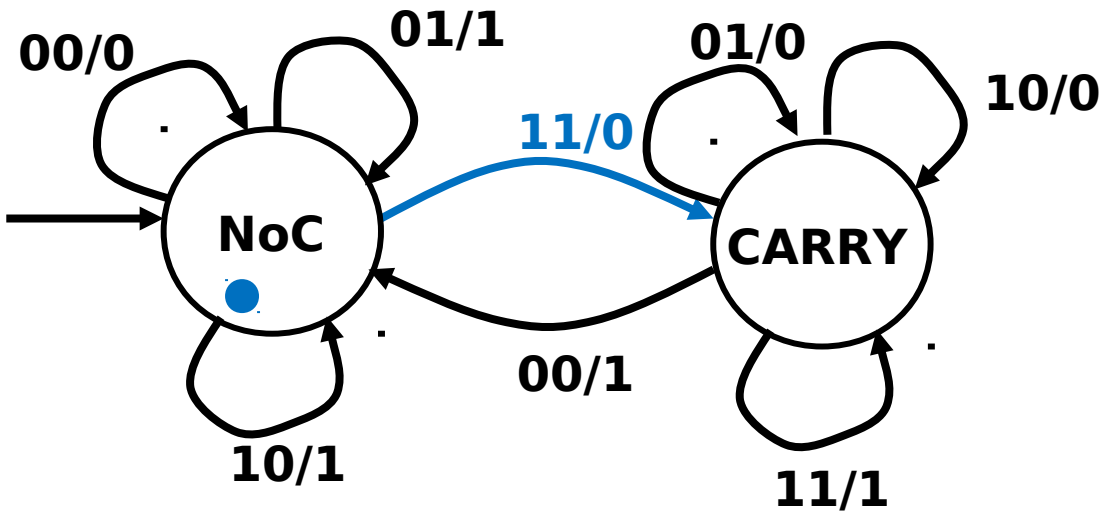


Obs.: Mealy é mais fácil...

Exercício Resolvido

- Projetar máquina de estados finitos que calcula uma soma bit a bit, considerando o vai 1.
  - Entrada: par de bits, considerando “vai 1” do estado anterior
  - Saída: resultado da soma

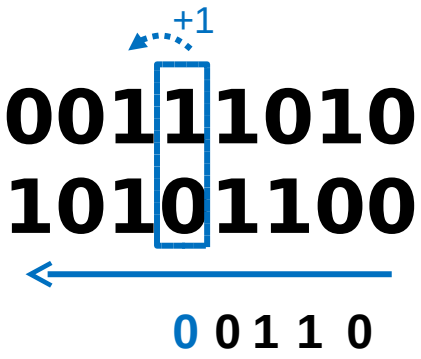


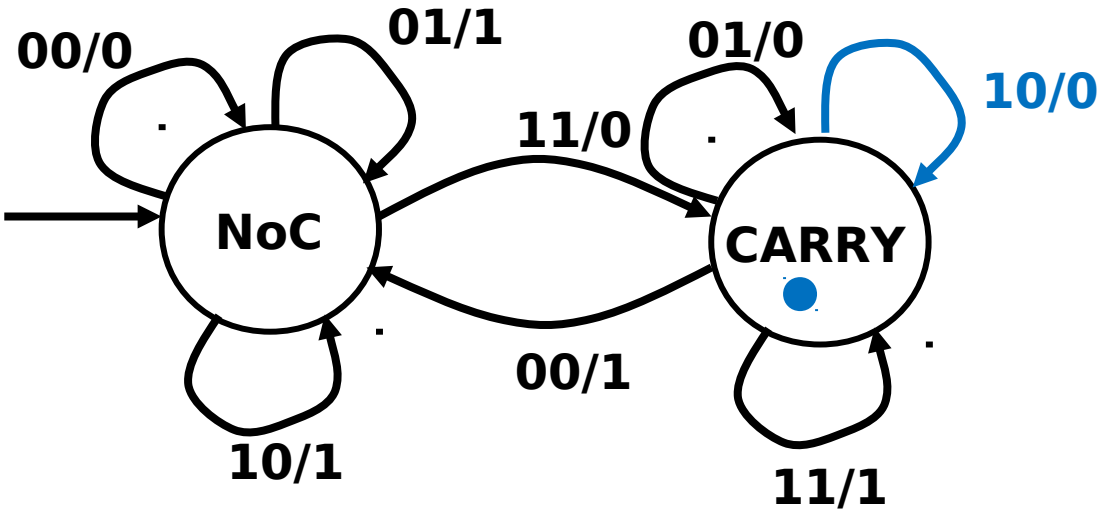


Obs.: Mealy é mais fácil...

Exercício Resolvido

- Projetar máquina de estados finitos que calcula uma soma bit a bit, considerando o vai 1.
  - Entrada: par de bits, considerando “vai 1” do estado anterior
  - Saída: resultado da soma

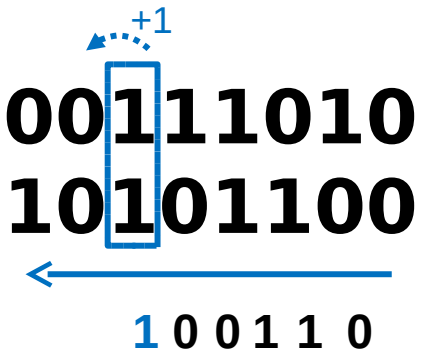


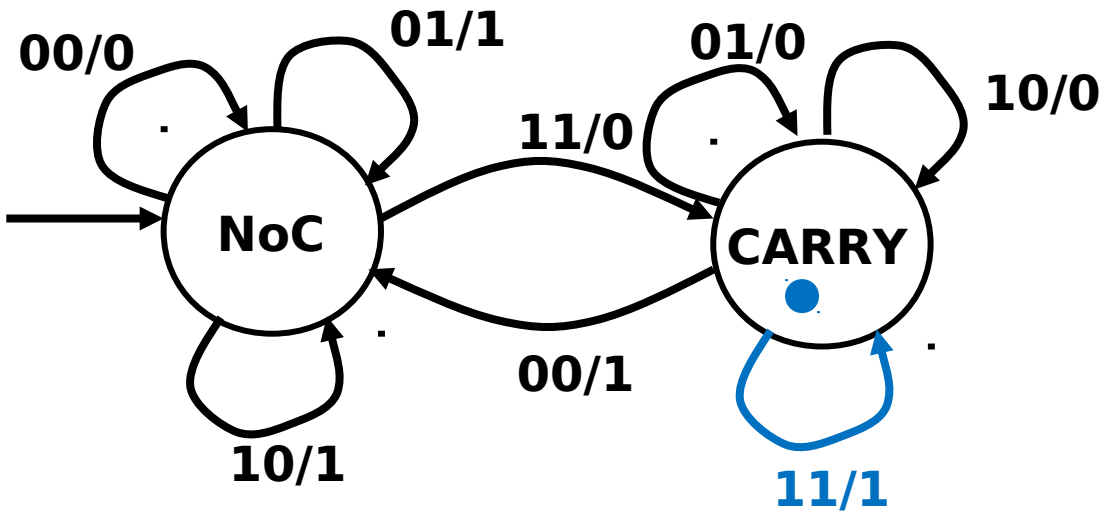


Obs.: Mealy é mais fácil...

Exercício Resolvido

- Projetar máquina de estados finitos que calcula uma soma bit a bit, considerando o vai 1.
  - Entrada: par de bits, considerando “vai 1” do estado anterior
  - Saída: resultado da soma

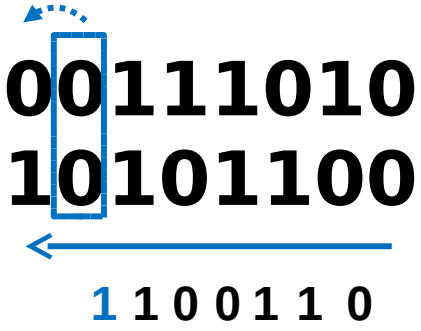


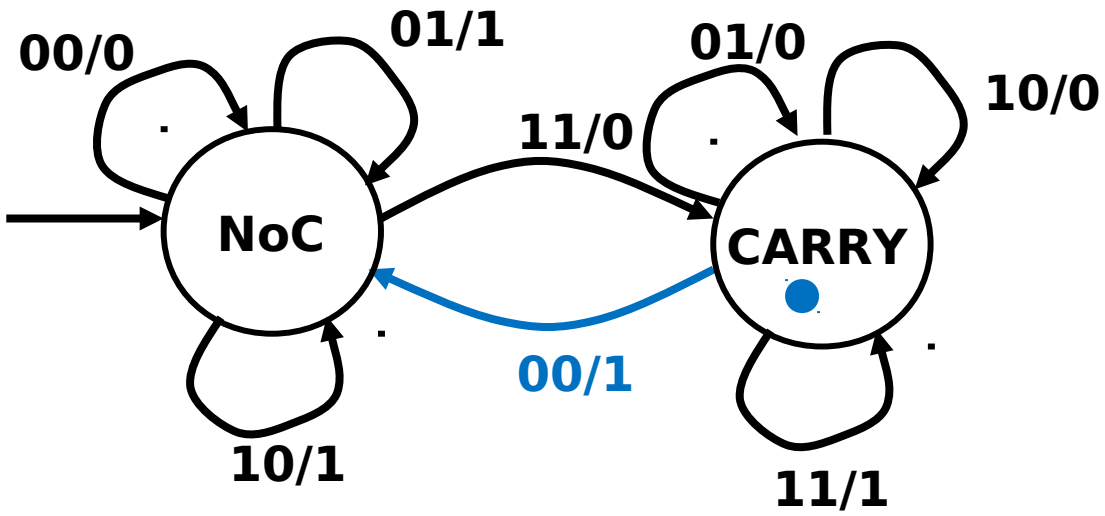


Obs.: Mealy é mais fácil...

Exercício Resolvido

- Projetar máquina de estados finitos que calcula uma soma bit a bit, considerando o vai 1.
  - Entrada: par de bits, considerando “vai 1” do estado anterior
  - Saída: resultado da soma

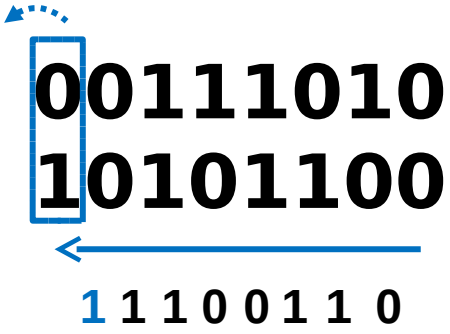


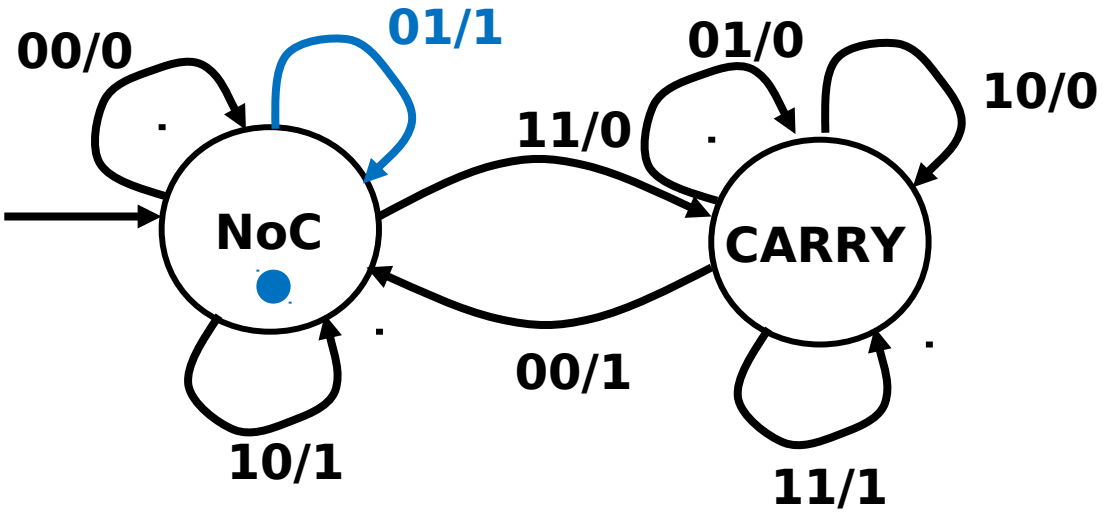


Obs.: Mealy é mais fácil...

Exercício Resolvido

- Projetar máquina de estados finitos que calcula uma soma bit a bit, considerando o vai 1.
  - Entrada: par de bits, considerando “vai 1” do estado anterior
  - Saída: resultado da soma





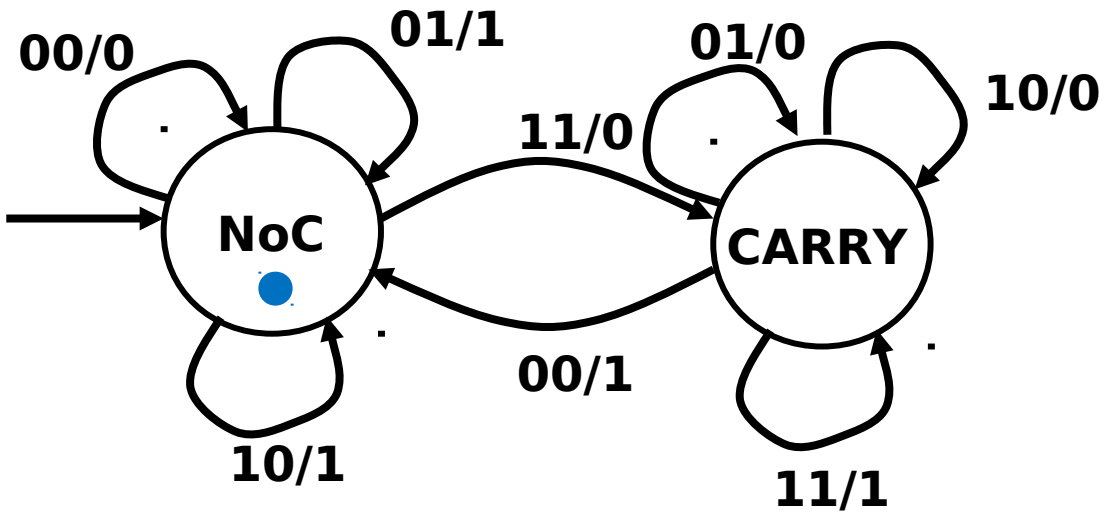
Obs.: Mealy é mais fácil...

Exercício Resolvido

- Projetar máquina de estados finitos que calcula uma soma bit a bit, considerando o vai 1.
  - Entrada: par de bits, considerando “vai 1” do estado anterior
  - Saída: resultado da soma

00111010
10101100

1 1 1 0 0 1 1 0



Obs.: Mealy é mais fácil...

FIM

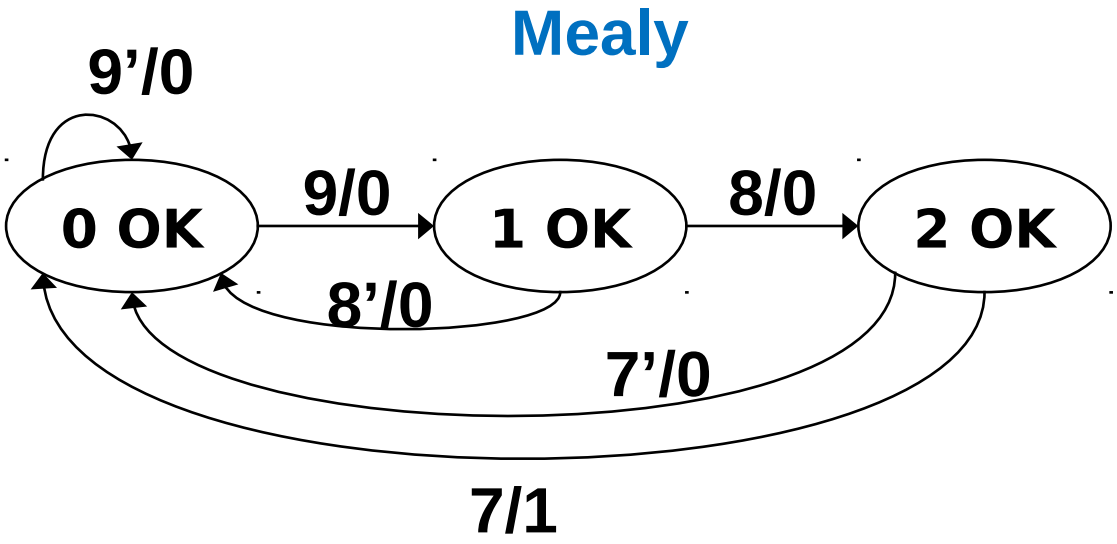
De volta ao “cadeado digital”

- Cadeado v2 (“sequência de dígitos”):
  - Conferir se entrada sequencial de 3 dígitos corresponde a senha.
    - Para facilitar: senha fixa em “987”

- Comecemos com algo simples:
  - Detecção de que “987” foi apertado nessa sequência, em qualquer momento

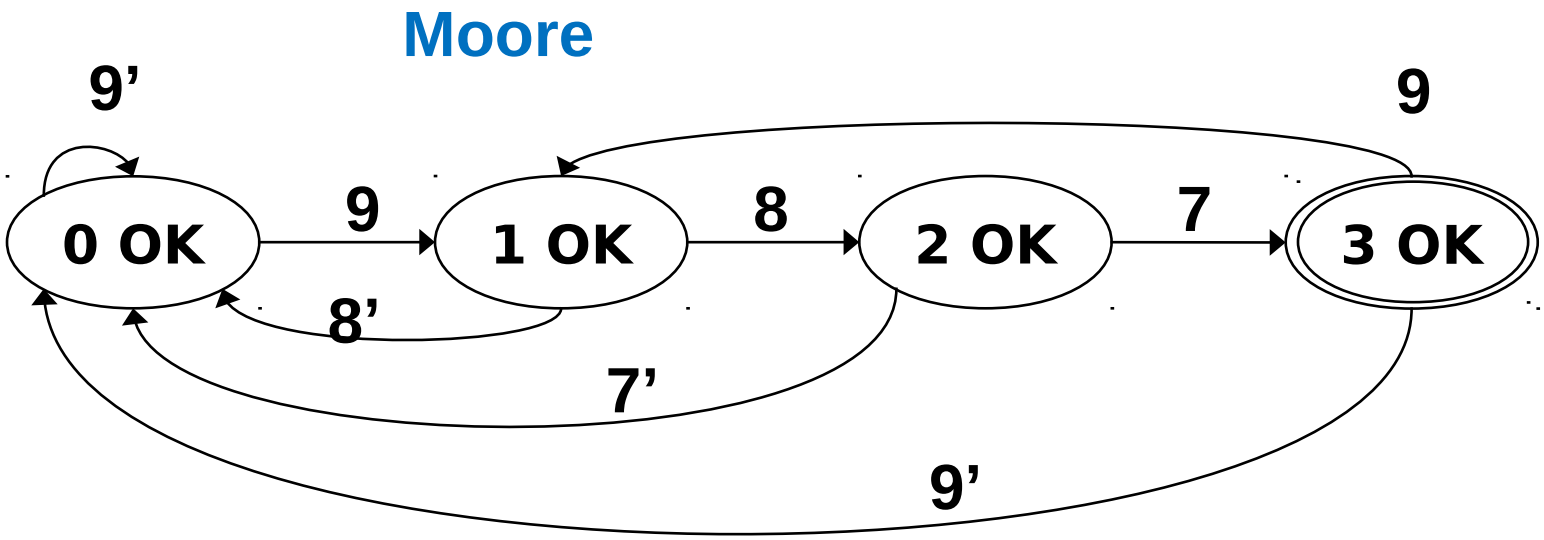
De volta ao “cadeado digital”

- Senha: “987”
  - Saída deve ser 1 apenas nesse caso
  - Notação para facilitar:
    - X’ significa “não é X”



De volta ao “cadeado digital”

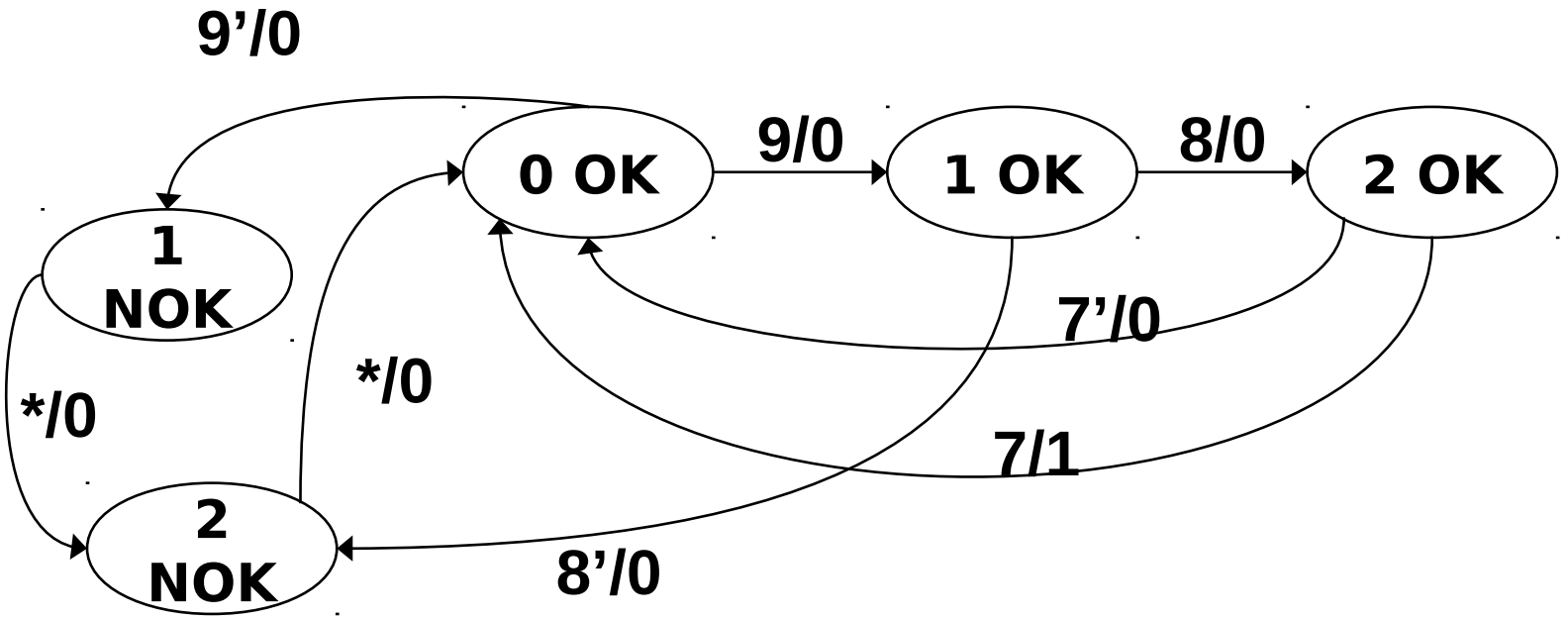
- Senha: “987”
  - Saída deve ser 1 apenas nesse caso
  - Notação para facilitar:
    - X’ significa “não é X”
    - * significa “qualquer valor”



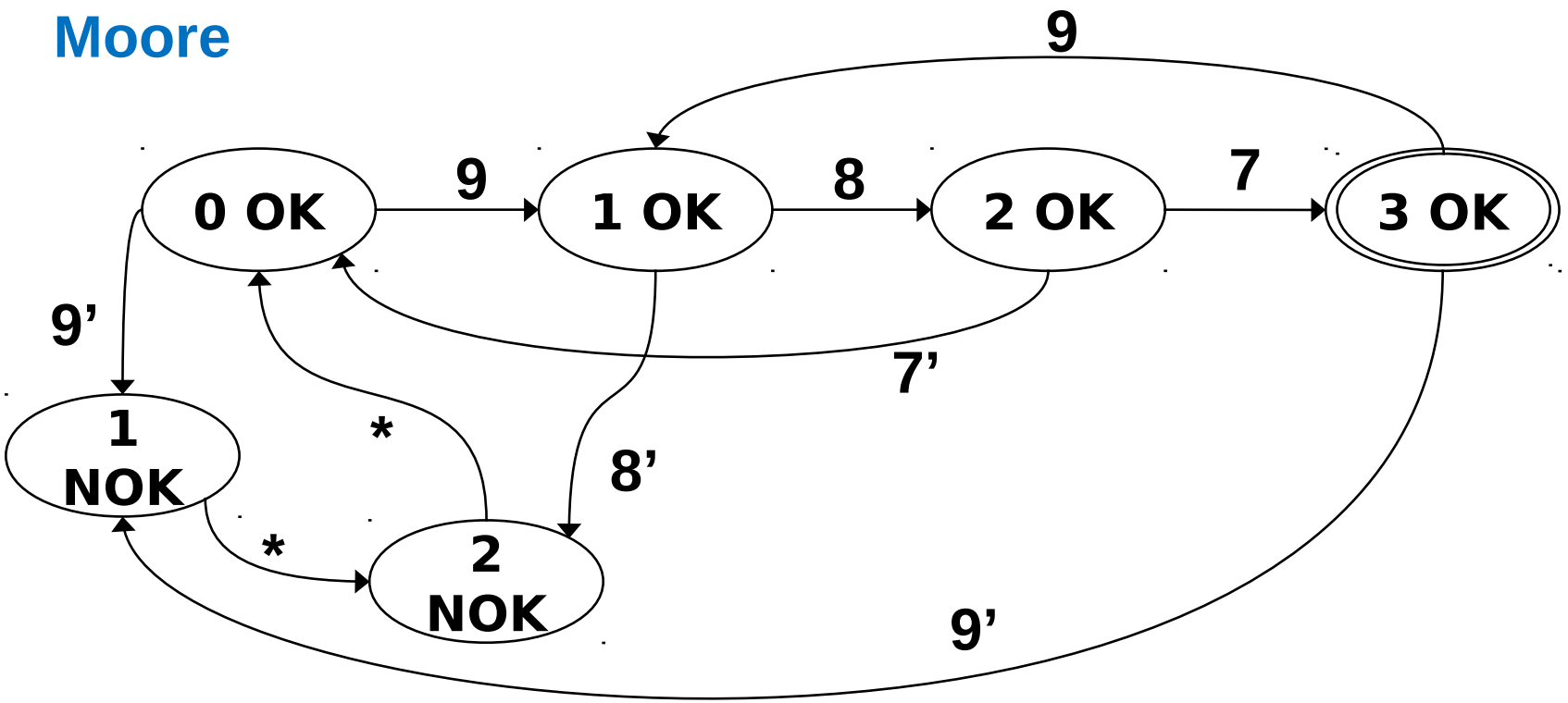
De volta ao “cadeado digital”

- Senha: “987”
  - Melhoria: avalia trincas apenas
    - Ex.: não aceita “1987”, ao contrário de anterior

Mealy



De volta ao “cadeado digital”

- Senha: “987”
  - Melhoria: avalia trincas apenas
    - Ex.: não aceita “1987”, ao contrário de anterior

Moore