9.6 - Exemplos de Diagramas de Estados e Tabelas de Estados/Saída

Neste item são apresentados diversos exemplos de como obter o diagrama de estados (ou tabela), a partir do enunciado.

Aconselhamos os alunos a não se limitarem a analisar as soluções. Tentem fazer sozinhos e, só depois, comparem o resultado com a solução proposta nesta apostila. Por esse motivo, inicialmente apresentamos todos os enunciados (item 9.6.1) e posteriormente as soluções (item 9.6.2).

Em todos os enunciados:

. CSS - Circuito Seqüencial Síncrono;

. nos exemplos, muitas vezes só é mostrado o instante em que z = 1; onde não aparece nada, considerar z = 0;

. exceto onde explicitamente solicitado, propor solução MEALY;

. em todos os exemplos, os valores das entradas e saídas mostrados são correspondentes à borda de atuação do CLOCK.

9.6.1 - Enunciados

Propor diagramas de estados ou tabelas de estados/saídas para as descrições abaixo:

a) CSS que fornece saída z = 1 toda vez que a entrada x = 1 após três ou mais 0's consecutivos.

Exemplo: x : 1 0 0 1 0 0 0 0 1 .....
z : .....1.....

b) CSS que reconhece qualquer bloco com um número par (par ≥ 2) de zeros consecutivos.

Exemplo: x : 1 0 1 1 0 0 1 0 0 0 0 1 0 .....
z : ..... 1.....1.....

c) CSS que reconhece, continuamente, as seqüências 000 e 010, com superposição.

Exemplo: x : 1 0 0 0 1 0 0 1 0 0 0 1 .....
z : .....1 0 1.....1....1.....

d) CSS que reconhece, continuamente, as seqüências 01 e 1010, com superposição.

Exemplo: x : 1 1 0 1 0 1 0 0 1 0 .....

z : .....1 1 1 1....1 0.....

e) CSS que reconhece, inicialmente, blocos de zeros de comprimento par (par ≥ 2), passando a reconhecer blocos de uns de comprimento ímpar, e assim sucessivamente.

Exemplo: x : 1 1 1 0 0 1 1 1 0 0 0 1 0 0 0 0 1 0 ...
z : 1 1 1 1....

f) CSS que apresenta, na saída z, o valor da soma binária seqüencial das entradas x_1 e x_2.

Exemplo: x_1 : 0 0 1 1 0 1 1 0 1 ...
x_2 : 1 0 1 1 0 0 1 0 0 ... $z = x_1 + x_2$
z : 1 0 0 1 1 1 0 1 1 ...

g) CSS que apresenta, na saída z, o valor da subtração binária seqüencial das entradas x_1 e x_2 ($z = x_1 - x_2$)

Exemplo: x_1 : 1 0 0 1 1 0 1 .....
x_2 : 0 1 0 0 1 1 1 .... $z = x_1 - x_2$
z : 1 1 1 0 0 1 1 ....

h) CSS que apresenta, na saída z, a mesma seqüência de entrada, atrasada 2 unidades de tempo.

Exemplo: x : 1 0 1 0 1 0 1 1 0 1 1 0 ...
z : 0 0 1 0 1 0 1 0 1 1 0 1 1 0
por imposição

i) Mesmo enunciado anterior, adotando modelo MOORE.

j) CSS com uma entrada **x** e uma saída z. A saída z é normalmente zero. Na entrada **x** ocorrem blocos de zeros e blocos de uns. A saída z será 1 ao terminar um bloco cujo comprimento tenha paridade diferente do bloco anterior (Obs.: Paridade é a característica do número ser par ou ímpar).

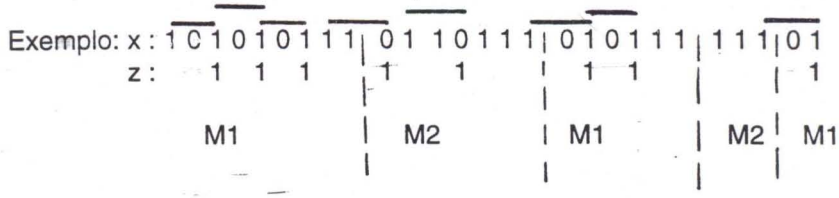
Exemplo: **x** : 0 0 1 1 0 0 0 1 0 0 0 0 1 0....
z : 0 0 0 0 0 0 0 1 0 0 0 0 1 0....

Hipótese: o bloco inicial deve ser comparado com "bloco anterior" de paridade par.

k) CSS com um entrada **x** e uma saída z, que opera em dois modos:

MODO1 : reconhece a seqüência 1 0 1 (inicial)
MODO2 : reconhece a seqüência 1 1 0

A mudança de modo ocorre com o recebimento da seqüência 1 1 1 . Pode haver superposição entre a seqüência que muda de modo e a que é reconhecida.



l) CSS que analisa blocos de 4 bits. Se o bloco representar um número BCD múltiplo de 3, a saída z = 1 coincide com 4º bit; caso contrário, z = 0. Após analisar cada bloco, o circuito volta ao estado inicial.

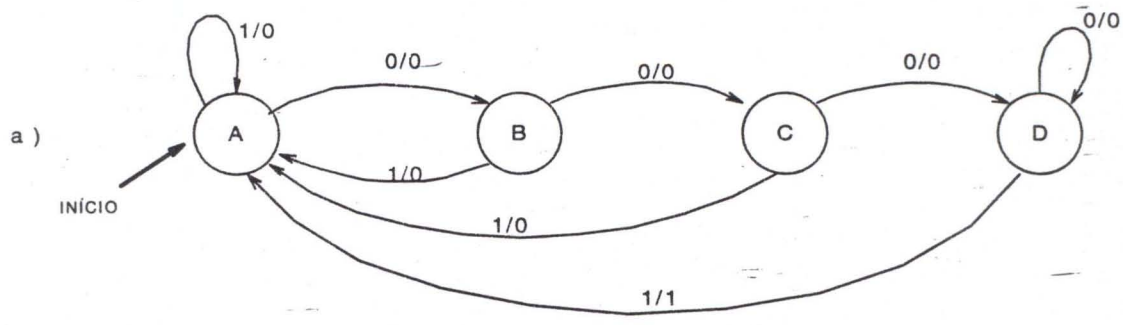
Exemplo: x : 0011 1101 1000 0110...

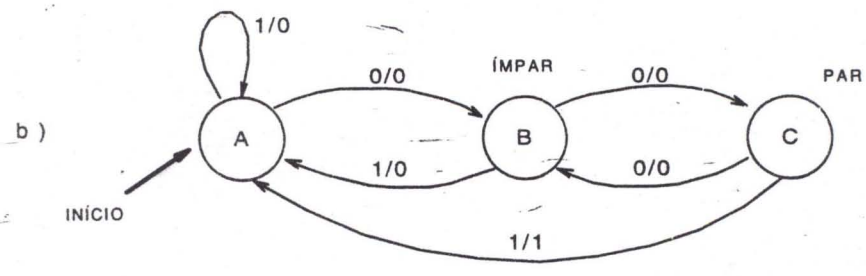
z : 0001 0000 0000 0001...

m) CSS com duas entradas x_1 e x_2 e uma saída z. A saída z é igual a 1 sempre que, num certo instante t, o nº de uns na entrada x e o número de zeros na entrada y forem múltiplos de 3.

Exemplo: x : 1 0 1 1 1 1 0 0 1 1 1 0 0 1...
y : 0 0 0 1 0 1 1 0 0 0 0 1 1 0...
z : 0 0 0 0 0 0 0 0 1 0 0 0 0 1...

9.6.2 - Soluções





c) 000 e 010

Descrição	s (t)	x = 0	x = 1
vazio	$A_\varnothing$	A_0 / 0	$A_\varnothing$
recebeu 0	A_0	A_{00} / 0	A_{01} / 0
recebeu 00	A_{00}	A_{00} / 1	A_{01} / 0
recebeu 01	A_{01}	A_0 / 1	$A_\varnothing$ / 0

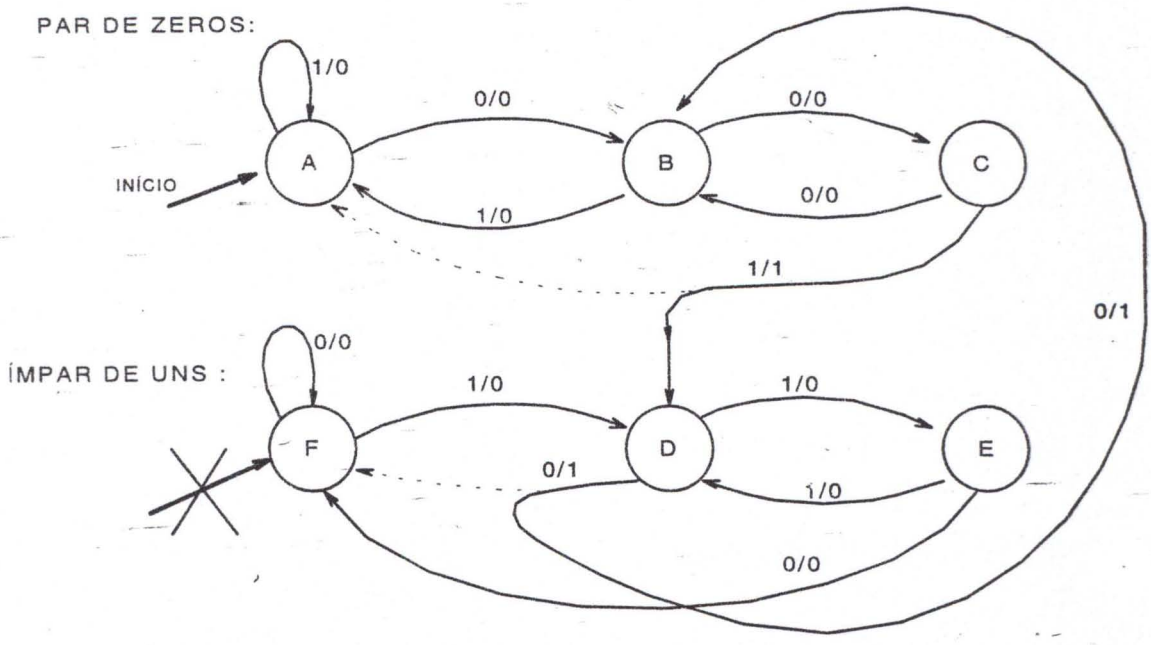
s (t+1) / z (t)

d) 01 e 1010

Descrição	s (t)	x = 0	x = 1
vazio	$A_\varnothing$	A_0 / 0	A_1 / 0
recebeu 0	A_0	A_0 / 0	A_1 / 1
recebeu 1	A_1	A_{10} / 0	A_1 / 0
recebeu 10	A_{10}	A_0 / 0	A_{101} / 1
recebeu 101	A_{101}	A_{10} / 1	A_1 / 0

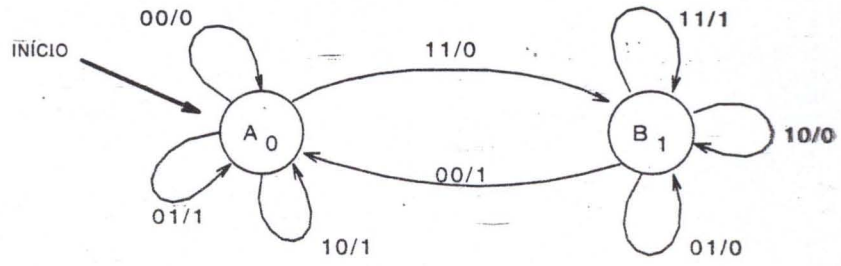
s (t+1) / z (t)

e) Podemos fazer 2 diagramas independentes e depois interconectá-los.

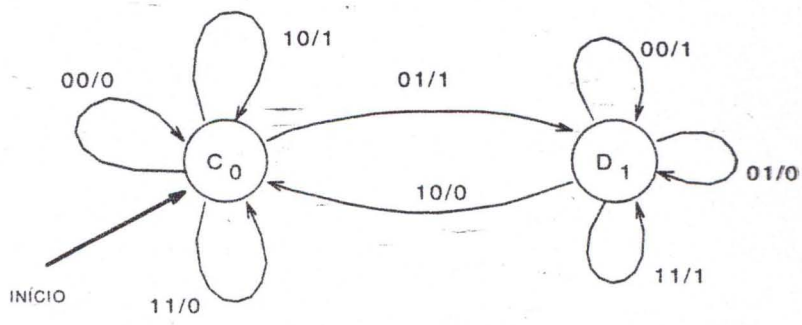


OBS : E = F

f)



g)



h) MEALY:

inicial

Descrição	s	x = 0	x = 1
recebeu 00	A_{00}	A_{00} / 0	A_{01} / 0
recebeu 01	A_{01}	A_{10} / 0	A_{11} / 0
recebeu 10	A_{10}	A_{00} / 1	A_{01} / 1
recebeu 11	A_{11}	A_{10} / 1	A_{11} / 1

s (t+1) / z (t)

i) MOORE

inicial

s	x = 0	x = 1	z(t)
A_{00}	A_{00}	A_{01}	0
A'_{00}	A_{00}	A_{01}	1
A_{01}	A_{10}	A_{11}	0
A'_{01}	A_{10}	A_{11}	1
A_{10}	A'_{00}	A'_{01}	0
A'_{10}	A'_{00}	A'_{01}	1
A_{11}	A'_{10}	A'_{11}	0
A'_{11}	A'_{10}	A'_{11}	1

s (t+1)

j) Cada estado deve conter informações sobre:

. paridade do bloco anterior (I ou P);

. tipo do bloco atual (ZERO ou UM);

. paridade do bloco atual (I_0, I_1, P_0, P_1).

Descrição dos estados por:

PP_0 - Bloco anterior PAR, Bloco atual de ZEROS, Paridade atual PAR
PI_0 - Bloco anterior PAR, Bloco atual de ZEROS, Paridade atual IMPAR
PP_1 - Bloco anterior PAR, Bloco atual de UNS, Paridade atual PAR
PI_1 - Bloco anterior PAR, Bloco atual de UNS, Paridade atual IMPAR
IP_0 - Bloco anterior IMPAR, Bloco atual de ZEROS, Paridade atual PAR
$I I_0$ - Bloco anterior IMPAR, Bloco atual de ZEROS, Paridade atual IMPAR
$I P_1$ - Bloco anterior IMPAR, Bloco atual deUNS, Paridade atual PAR
$I I_1$ - Bloco anterior IMPAR, Bloco atual de UNS, Paridade atual IMPAR

Estado Inicial: $P_\varnothing$

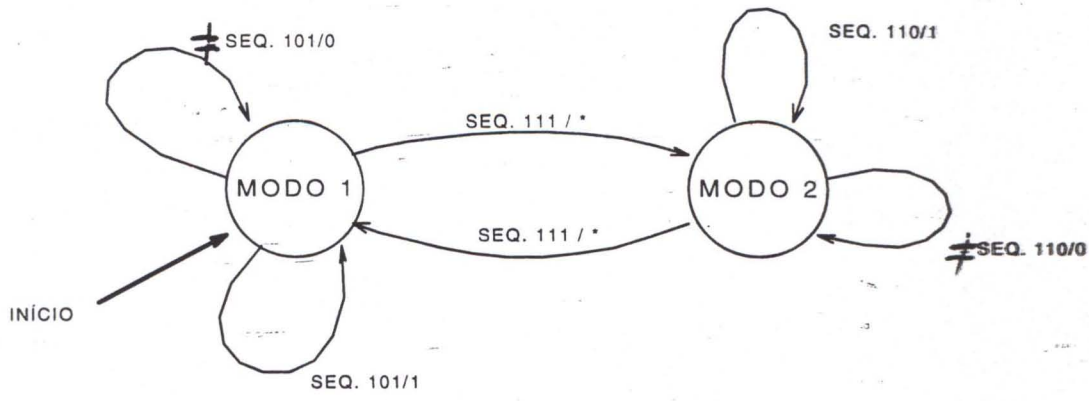
s	x = 0	x = 1
$P_\varnothing$*	PI_0 / 0	PI_1 / 0
PI_0	PP_0 / 0	$I I_1$ / 1
PI_1	$I I_0$ / 1	PP_1 / 0
PP_0*	PI_0 / 0	PI_1 / 0
$I I_1$	$I I_0$ / 1	$I P_1$ / 0
$I I_0$	IP_0 / 0	$I I_1$ / 0
PP_1 *	PI_0 / 0	PI_1 / 0
IP_1	PI_0 / 1	$I I_1$ / 0
IP_0	$I I_0$ / 0	PI_1 / 1

s (t+1) / z (t)

Obs.: Na Tabela, os Estados foram sendo colocados à medida que a Tabela foi sendo construída, a partir do estado inicial.

* equivalente

k) Ha dois modos de operação, resumidos no diagrama abaixo.



*** pelo enunciado, pode haver superposição das seqüências detectadas com a seqüência que muda de modo.**

início

s	x = 0	x = 1	OBS.
A ø	A ø / 0	A_1 / 0	
A_1	A_{10} / 0	A_{11} / 0	
A_{10}	A ø / 0	A_1 / 1	MODO 1
A_{11}	A_{10} / 0	T_{111} / 0	
T_{111}	Bø / 1	T_{1111} / 0	Transição 1 → 2
T_{1111}	Bø / 1	T_{11111} / 0	Pode detectar 110
T_{11111}	Bø / 1	T'_{111} / 0	...
Bø	Bø / 0	B_1 / 0	
B_1	Bø / 0	B_{11} / 0	MODO 2
B_{11}	Bø / 1	T'_{111} / 0	
T'_{111}	A_{10} / 0	T'_{1111} / 0	Transição 2 → 1
T'_{1111}	A_{10} / 0	T'_{11111} / 0	Pode detectar 101
T'_{11111}	A_{10} / 0	T_{111} / 0	

Reduzindo a Tabela

início

s	x = 0	x = 1	OBS.
A ø	A ø / 0	A_1 / 0	
A_1	A_{10} / 0	A_{11} / 0	
A_{10}	A ø / 0	A_1 / 1	MODO 1
A_{11}	A_{10} / 0	T_{111} / 0	
T_{111}	Bø / 1	T_{1111} / 0	1 → 2
T_{1111}	Bø / 1	B_{11} / 0	
Bø	Bø / 0	B_1 / 0	
B_1	Bø / 0	B_{11} / 0	MODO 2
B_{11}	Bø / 1	T'_{111} / 0	
T'_{111}	A_{10} / 0	T'_{1111} / 0	2 → 1
T'_{1111}	A_{10} / 0	A_{11} / 0	

s (t+1) / z (t)

I) Cadeia de 4 bits. Detecta 0, 3, 6, 9. Volta sempre ao estado inicial. (OBS. 12 e 15 **não são** do código BDC)

Dígito	Código
0	0000
3	0011
6	0110
9	1001

os estados devem armazenar os três primeiros bits do código.

início

s	x = 0	x = 1
A	A_0 / 0	A_1 / 0
A_0	A_{00} / 0	A_{01} / 0
A_1	A_{10} / 0	A_{11} / 0
A_{00}	A_{000} / 0	A_{001} / 0
A_{01}	A_{xxx} / 0	A_{011} / 0
A_{10}	A_{100} / 0	A_{xxx} / 0
A_{000}	A / $1^{(0)}$	A / 0
A_{001}	A / 0	A / $1^{(3)}$
A_{011}	A / $1^{(6)}$	A / 0
A_{100}	A_0 / 0	A / $1^{(9)}$
— —	— — —	— — — —
A_{11}	A_{xxx} / 0	A_{xxx} / 0
A_{xxx}	A / 0	A / 0

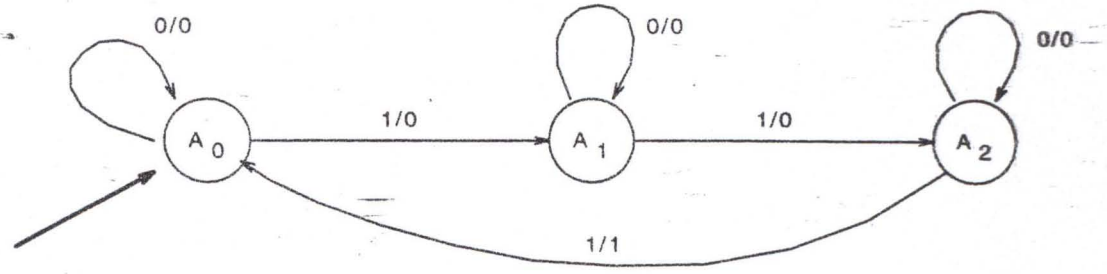
"Lixo". Mas é necessário contar os 4 bits, antes de voltar para A.

s(t+1) / z (t)

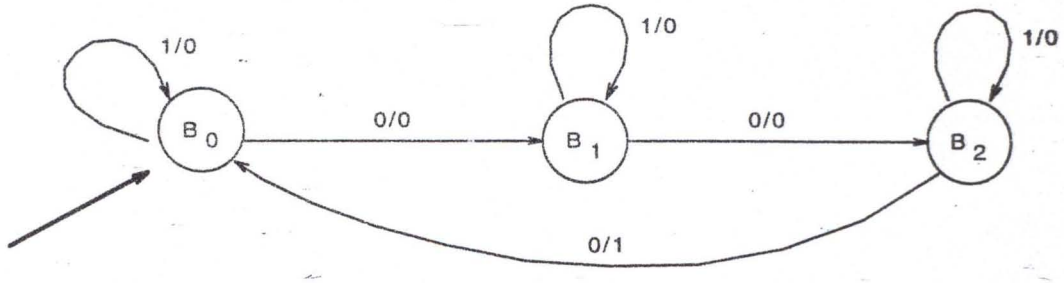
m) z = 1 apenas quando, ao chegar o CLOCK, x = 1 e y = 0 e ocorreu a **multiplicidade** desejada.

Este exemplo pertence à classe de problemas que pode ser resolvida por **composição de** diagramas de estado.

DETECTOR DE Nº MÚLTIPLO DE 3 DE UNS



DETECTOR DE Nº MÚLTIPLO DE 3 DE ZEROS



Podemos construir a Tabela por composição

	xy			
s	**00**	**01**	**11**	**10**
A_0B_0	A_0B_1 / 0	A_0B_0 / 0	A_1B_0 / 0	A_1B_1 / 0
A_0B_1	A_0B_2 / 0	A_0B_1 / 0	A_1B_1 / 0	A_1B_2 / 0
A_0B_2	A_0B_0 / 0	A_0B_2 / 0	A_1B_2 / 0	A_1B_0 / 0
A_1B_0	A_1B_1 / 0	A_1B_0 / 0	A_2B_0 / 0	A_2B_1 / 0
A_1B_1	A_1B_2 / 0	A_1B_1 / 0	A_2B_1 / 0	A_2B_2 / 0
A_1B_2	A_1B_0 / 0	A_1B_2 / 0	A_2B_2 / 0	A_2B_0 / 0
A_2B_0	A_2B_1 / 0	A_2B_0 / 0	A_0B_0 / 0	A_0B_1 / 0
A_2B_1	A_2B_2 / 0	A_2B_1 / 0	A_0B_1 / 0	A_0B_2 / 0
A_2B_2	A_2B_0 / 0	A_2B_2 / 0	A_0B_2 / 0	A_0B_0 / 1

s(t+1) / z (t)

9.7 - Exercícios:

1º) Usando um flip-flop tipo D sensível à borda de subida, sintetizar um circuito que se comporte como um flip-flop J - K sensível à borda de subida, usando a metodologia de síntese de circuitos síncronos.

2º) Projetar um contador decimal síncrono de 4 bits, que forneça nas saídas o código BCD (0 a 9). Utilizar flip-flops tipo J - K sensíveis à ↑.

3º) Projetar, com flip-flops D↑, um contador decimal síncrono de 4 bits, com a seguinte especificação:

- Se a entrada **x** = 0, o comportamento das saídas é o do exercício 2.
- Se a entrada **x** = 1, o contador conta "para baixo", ou seja, a contagem se inicia em 9 e vai decrescendo até 0.

4º) Projetar circuitos seqüenciais síncronos, com flip-flops tipo D↑, descritos pelas seguintes tabelas de estados/saídas:

a)

s \ x	x = 0	x = 1
A	B/0	A/0
B	C/0	A/0
C	B/0	A/1

s (t+1) / z (t)

b)

s \ x	x = 0	x = 1
A	B/0	A/0
B	C/0	D/0
C	C/1	D/0
D	B/1	A/0

s (t+1) / z (t)

c)

s \ x	x = 0	x = 1
A	B/0	C/0
B	B/0	C/1
C	D/0	C/0
D	B/0	E/1
E	D/1	C/0

s (t+1) / z (t)

d)

s \ x_1x_2	00	01	11	10
A_0	A_0 /0	A_0 /1	B_0 /0	A_0 /1
B_0	A_0 /1	B_0 /0	B_0 /1	B_0 /0

s (t+1) / z (t)

e)

s \ x	x = 0	x = 1
A	B / 11	D / 00
B	C / 10	A / 11
C	D / 01	B / 10
D	A / 00	C / 01

s (t+1) / z_1z_2 (t)

5º) Repetir o projeto dos circuitos do exercício 4, utilizando flip-flops J-K sensíveis à **borda**↑.

6º) Transformar as tabelas de estado/saída do exercício 4 em modelo MOORE e sintetizar os respectivos circuitos com flip-flops D sensíveis à ↑.

7º) Transformar as tabelas de estados/saída abaixo nas equivalentes, de acordo com o modelo MEALY.

a)

s^t \ x^t	x = 0	x = 1	z (t)
A	B	E	0
B	C	B	1
C	D	A	0
D	B	E	1
E	F	C	0
F	A	B	1

s (t+1)

b)

s^t \ x^t	x = 0	x = 1	$\bar{z}_1 z_2^t$
A	B	A	0 0
B	C	D	0 1
C	D	B	1 1
D	A	C	1 0

s (t+1)

8º) Minimizar as tabelas de estados/saída apresentadas abaixo.

a)

s \ x	x = 0	x = 1
A	B / 1	H / 1
B	F / 1	D / 1
C	D / 0	E / 1
D	C / 0	F / 1
E	D / 1	C / 1
F	C / 1	C / 1
G	C / 1	D / 1
H	C / 0	A / 1

s(t+1) / z (t)

b)

s \ x	x = 0	x = 1	x = 2	x = 3
A	E / 1	C / 0	B / 1	E / 1
B	C / 0	F / 1	E / 1	B / 0
C	B / 1	A / 0	D / 1	F / 1
D	G / 0	F / 1	E / 1	B / 0
E	C / 0	F / 1	D / 1	E / 0
F	C / 1	F / 1	D / 0	H / 0
G	D / 1	A / 0	B / 1	F / 1
H	B / 1	C / 0	E / 1	F / 1

s(t+1) / z (t)

c)

s (t)	x = 0	x = 1	z(t)
A	B	H	0
B	D	E	0
C	E	A	0
D	B	A	0
E	F	G	0
F	C	G	0
G	B	I	0
H	E	A	1
I	C	G	1

s(t+1)

d)

s(t)	x = a	x = b	x = c	z(t)
1	2	2	5	0
2	1	4	4	1
3	2	2	5	0
4	3	2	2	1
5	6	4	3	0
6	8	9	6	1
7	6	2	8	0
8	4	4	7	0
9	7	9	7	1

s(t+1)

9º) Propor diagramas de estado ou tabelas para as descrições abaixo:

a) CSS que reconhece as seqüências 01, 010, 110 e 11100, com superposição.

b) CSS que fornece saída um sempre que, num determinado instante, o número total de ZEROS já recebidos for igual ao número total de UNS.

c) CSS que reconhece um bloco com um número par de ZEROS, seguido por "111" ou um bloco com um número ~~IMPAR~~ ímpar de UNS, seguido por "00".