Análise e Projeto de Circuitos Combinacionais e Sequenciais

Referência bibliográfica:

- *Digital Design: Principles and Practices* - Wakerly
- Elementos de Eletrônica Digital – Idoeta e Capuano
- Introduction to Switching Theory – Hill e Peterson
- *Logic and Computer Design Fundamentals* – Mano e Kime
- Sistemas Digitais – Tocci e Widmer
- Digital Design – Vahid, F.

Circuito Lógico Combinacional

Circuito Lógico Combinacional é aquele em que as saídas dependem apenas das combinações das variáveis de entrada

Ex.:Somador, Comparador, Decodificador ...

Um Circuito Combinacional pode ser especificado por uma **Tabela da Verdade** que liste todos os valores de **Saída** para cada combinação das variáveis de **Entrada**

Um Circuito Combinacional pode também ser especificado por *m* **Funções Booleanas**, uma para cada variável de saída

Procedimento de Análise de Circuito Combinacional

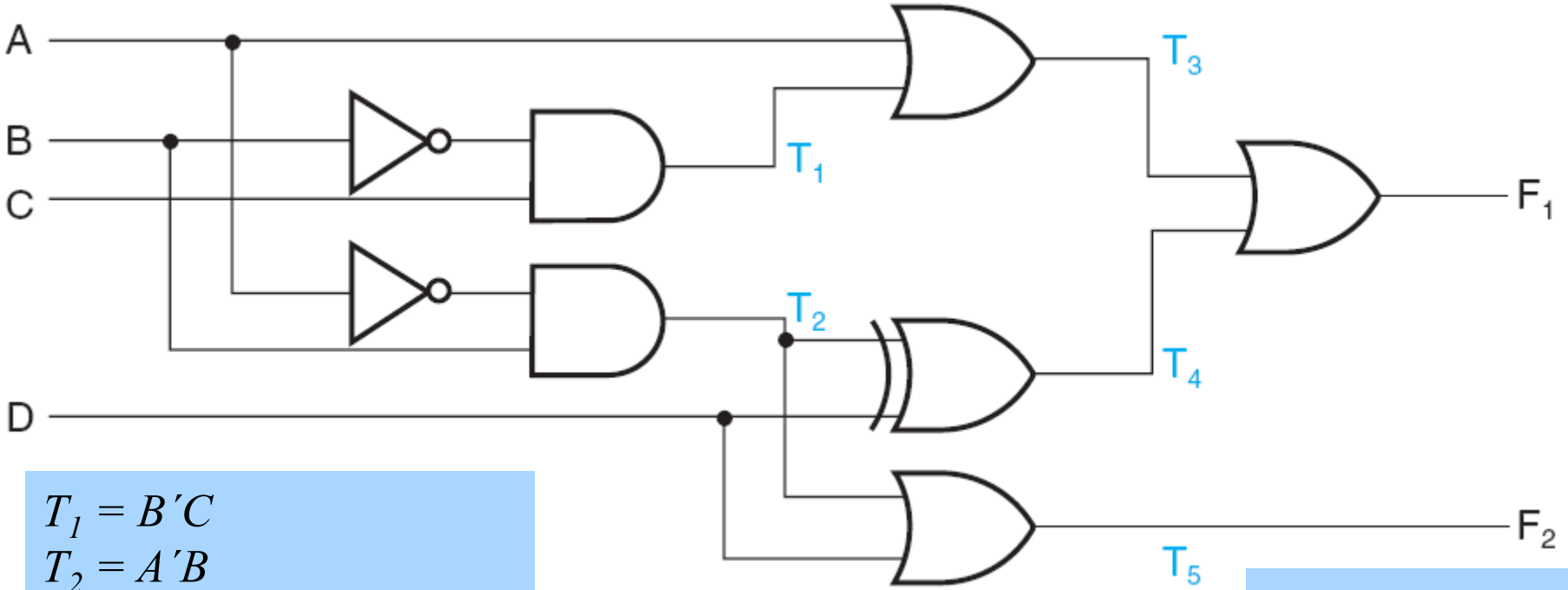
A **Análise** de um **Circuito Combinacional** consiste em determinar a **Função** que o circuito implementa

Através da Análise de um **Diagrama do Circuito** obtém-se um conjunto de **Funções Booleanas** ou uma **Tabela da Verdade**

Com uma descrição da **Função Lógica** é possível determinar o **comportamento do circuito**, **manipular a descrição algébrica** para sugerir diferentes estruturas de circuito etc.

Exemplo de Análise para o Diagrama Lógico

Para obter as **Funções Booleanas** de um Diagrama Lógico identifique as saídas das portas lógicas próximas às entradas e continue em direção às saídas do diagrama



$T_1 = B'C$
$T_2 = A'B$
$T_3 = A + T_1 = A + B'C$

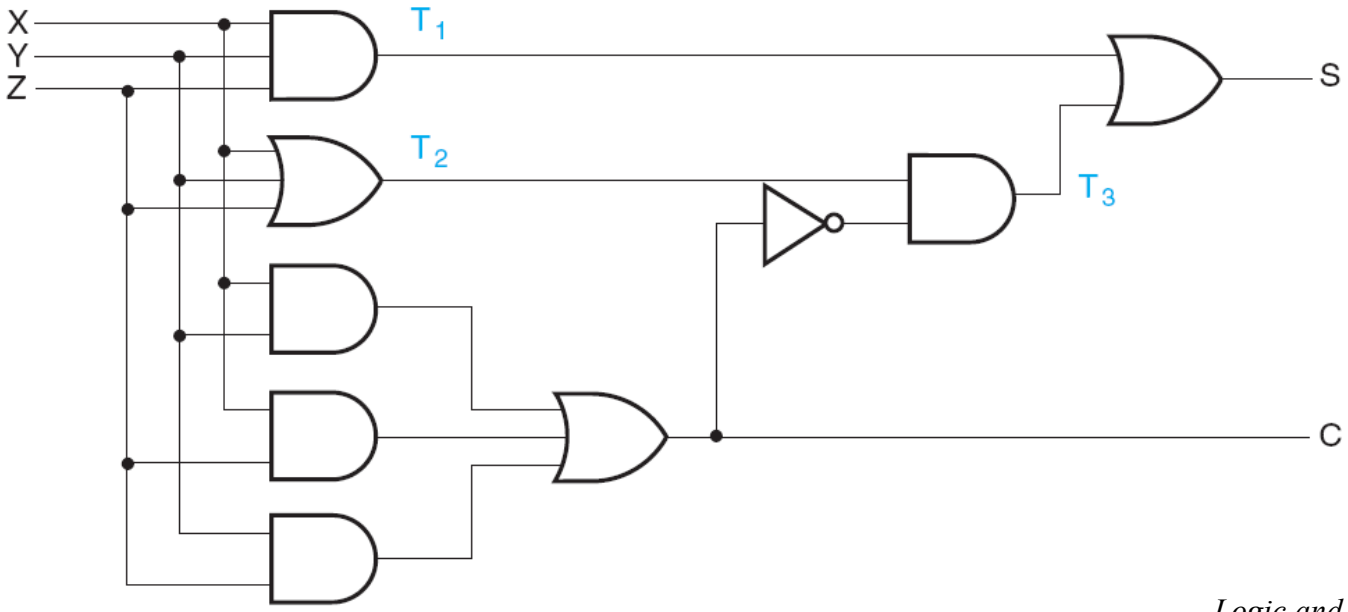
$F_2 = T_5 = A'B + D$

Diagrama Lógico para um Somador Binário

Para obter a **Tabela da Verdade** de um Diagrama Lógico, divida o circuito em pequenos blocos e obtenha a Tabela da Verdade para cada um desses blocos. Repita o procedimento em direção às saídas

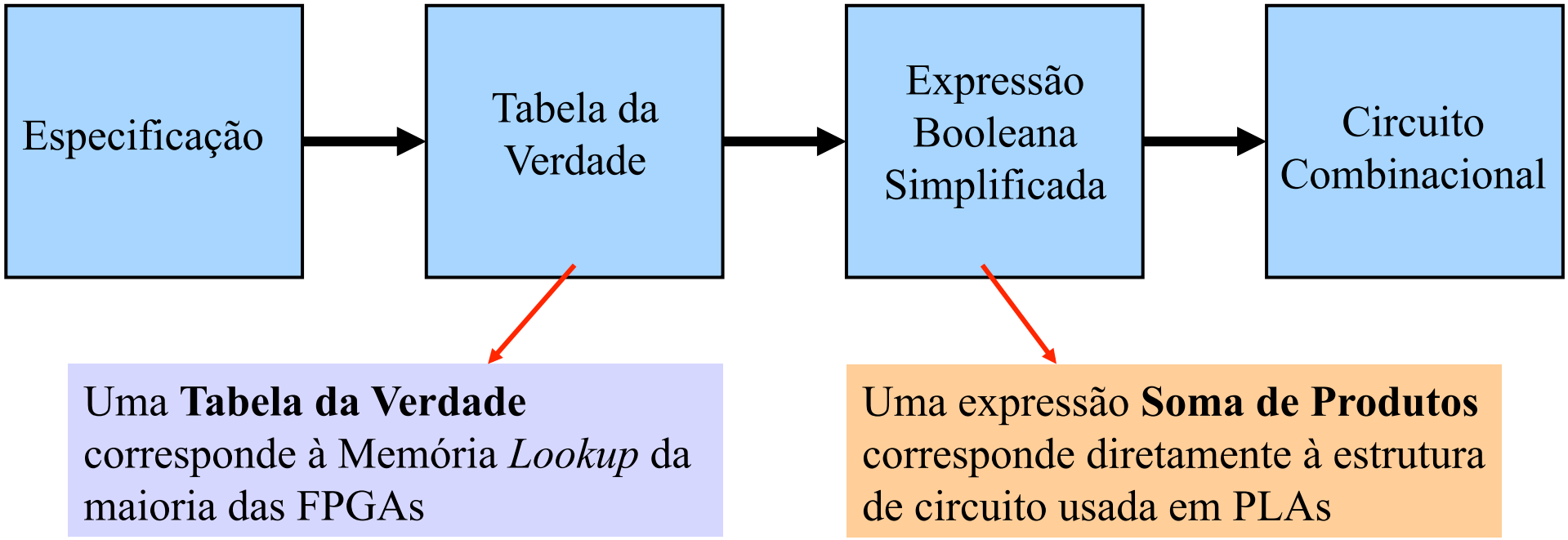
Tabela da Verdade para o Somador Binário

X	Y	Z	C	$\overline{C}$	T_1	T_2	T_3	S
0	0	0	0	1	0	0	0	0
0	0	1	0	1	0	1	1	1
0	1	0	0	1	0	1	1	1
0	1	1	1	0	0	1	0	0
1	0	0	0	1	0	1	1	1
1	0	1	1	0	0	1	0	0
1	1	0	1	0	0	1	0	0
1	1	1	1	0	1	1	0	1



Procedimento de Projeto de Circuito Combinacional

Para obter um **Circuito Combinacional** a partir de uma **Especificação**, primeiramente constrói-se a **Tabela da Verdade**, depois obtêm-se a **Expressão Booleana Simplificada** e o **Circuito Final**



Procedimento de Projeto de Circuito Combinacional

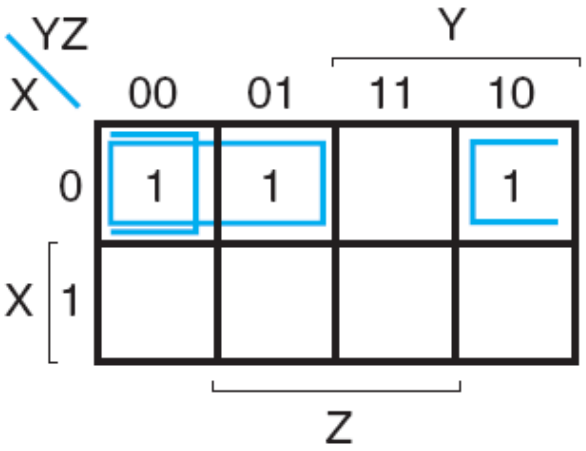
O **Projeto** de um **Circuito Combinacional** parte de uma **Especificação** do problema, passa pela **Tabela da Verdade**, para chegar a um **Diagrama Lógico** ou a um conjunto de **Equações Booleanas**

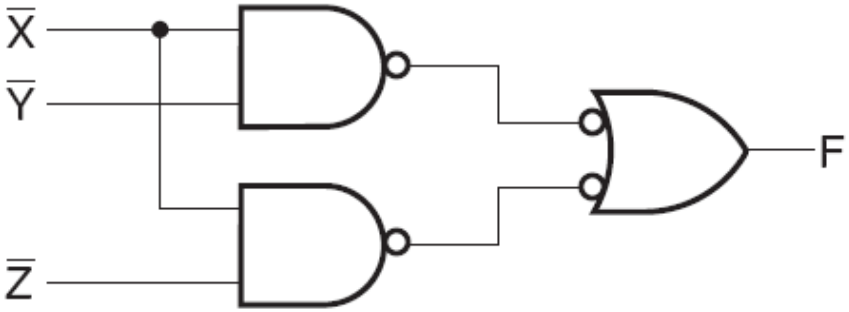
Especificação: Projete um **Circuito Combinacional** com três entradas e uma saída, que deve ser 1 quando o valor binário das entradas for menor que 011 (3) e, caso contrário, saída 0. Use portas NAND.

Tabela da Verdade

X	Y	Z	F
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

$F = X'Y' + X'Z'$

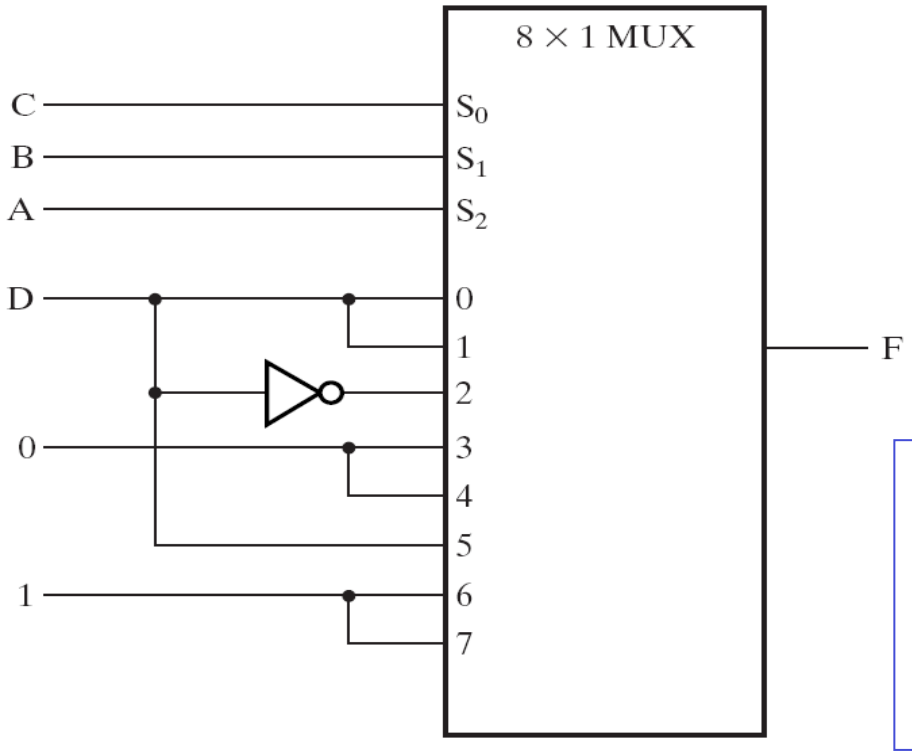




Implementação de Função Booleana

Considere a seguinte função $F(A,B,C,D) = \Sigma m(1,3,4,11,12,13,14,15)$

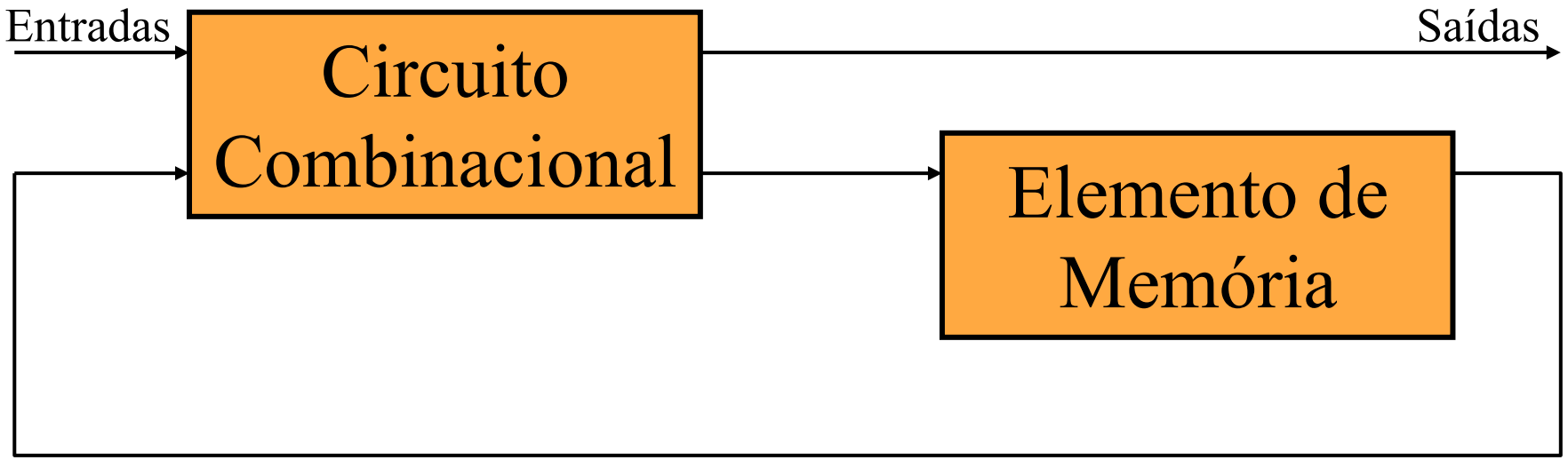
A	B	C	D	F	
0	0	0	0	0	F = D
0	0	0	1	1	
0	0	1	0	0	F = D
0	0	1	1	1	
0	1	0	0	1	F = $\overline{D}$
0	1	0	1	0	
0	1	1	0	0	F = 0
0	1	1	1	0	
1	0	0	0	0	F = 0
1	0	0	1	0	
1	0	1	0	0	F = D
1	0	1	1	1	
1	1	0	0	1	F = 1
1	1	0	1	1	
1	1	1	0	1	F = 1
1	1	1	1	1	



Qualquer função booleana de *n* variáveis pode ser implementada com um MUX de (*n-1*) Entradas de Seleção

Circuitos Lógicos Sequenciais

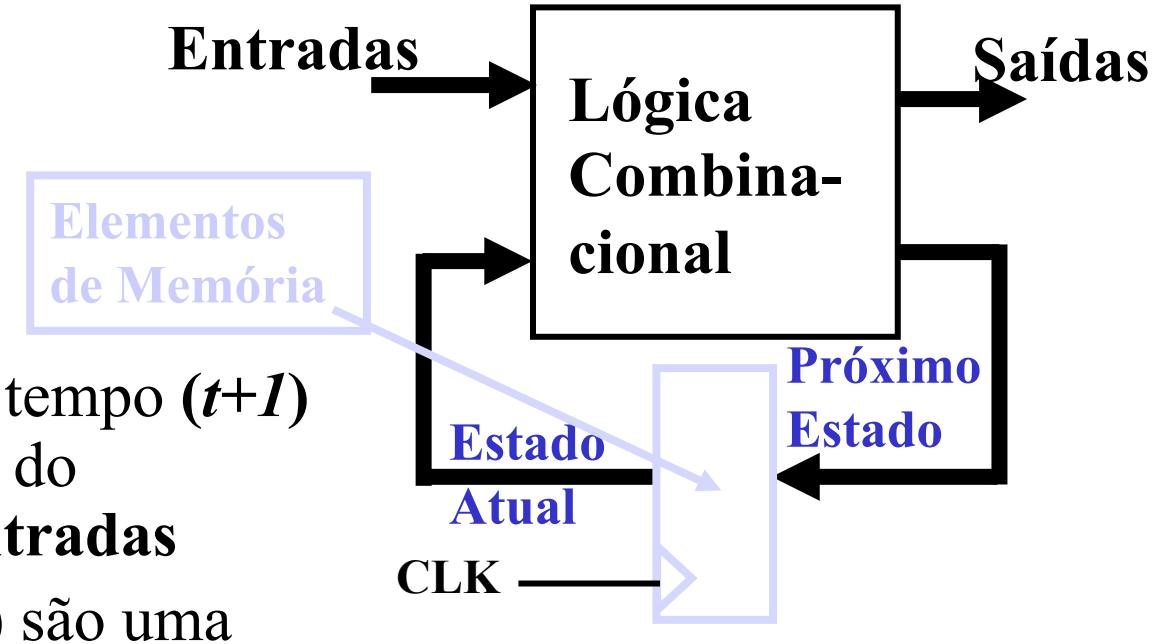
Circuito Lógico Sequencial é aquele que possui algum Elemento de Memória



A **Análise** de um **Circuito Sequencial** consiste em obter uma descrição que demonstre a **sequência no tempo** de entradas, saídas e estados

Análise de Circuitos Sequenciais

- Modelo Geral
  - O **Estado Atual** no tempo (***t***) está salvo num conjunto de *flip-flops*
  - O **Próximo Estado** no tempo (***t*+*1***) é uma função booleana do **Estado Atual** e das **Entradas**
  - As **Saídas** no tempo (*t*) são uma função booleana do **Estado (*t*)** (ou seja, **Estado Atual**) e (às vezes) das **Entradas (*t*)**



Exemplo 1

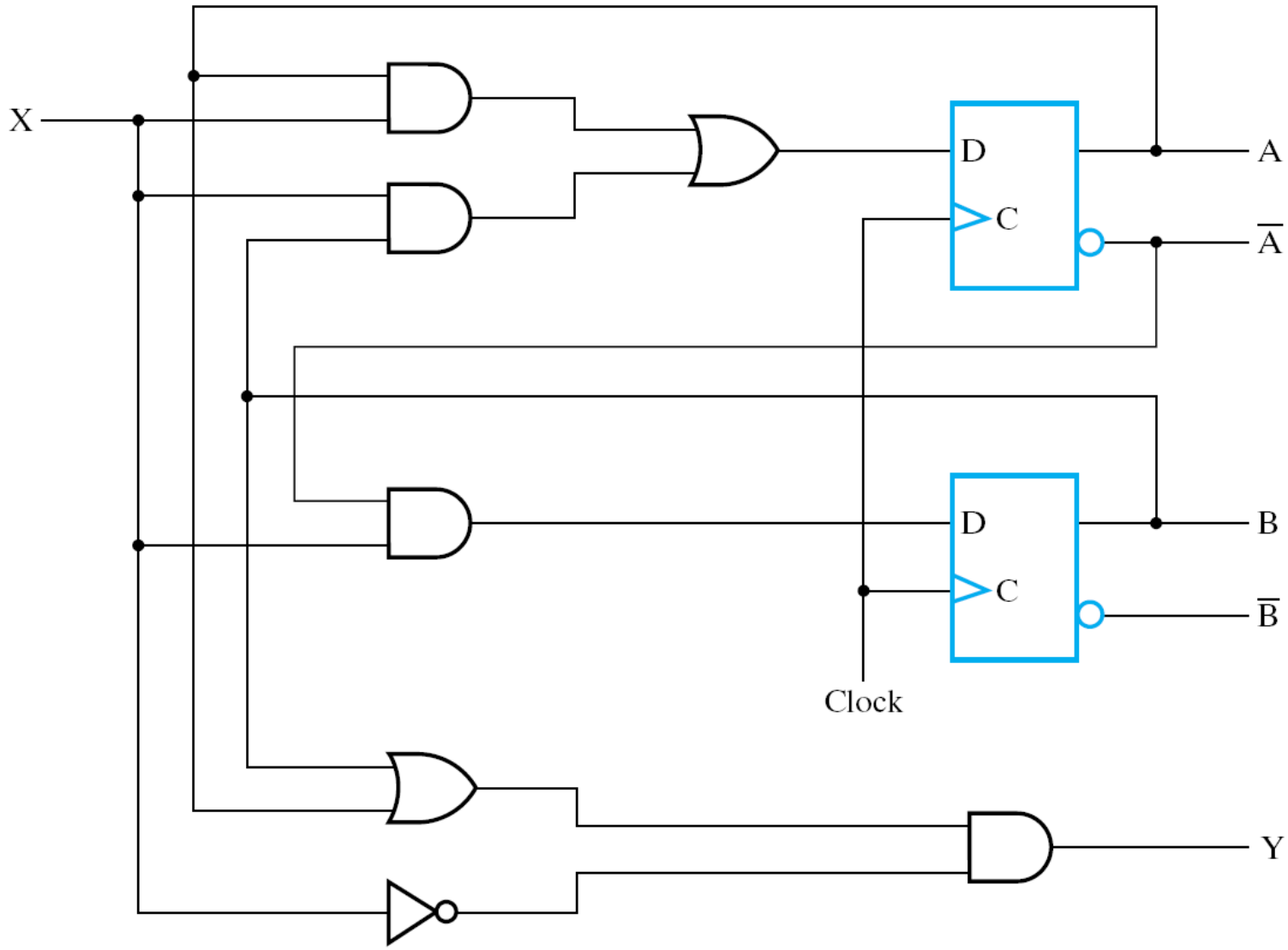
Entrada: X(t)

Saída: Y(t)

Estado Atual: A(t) e B(t)

Qual é a **Função de Saída**?

Qual é a **Função do Próximo Estado**?



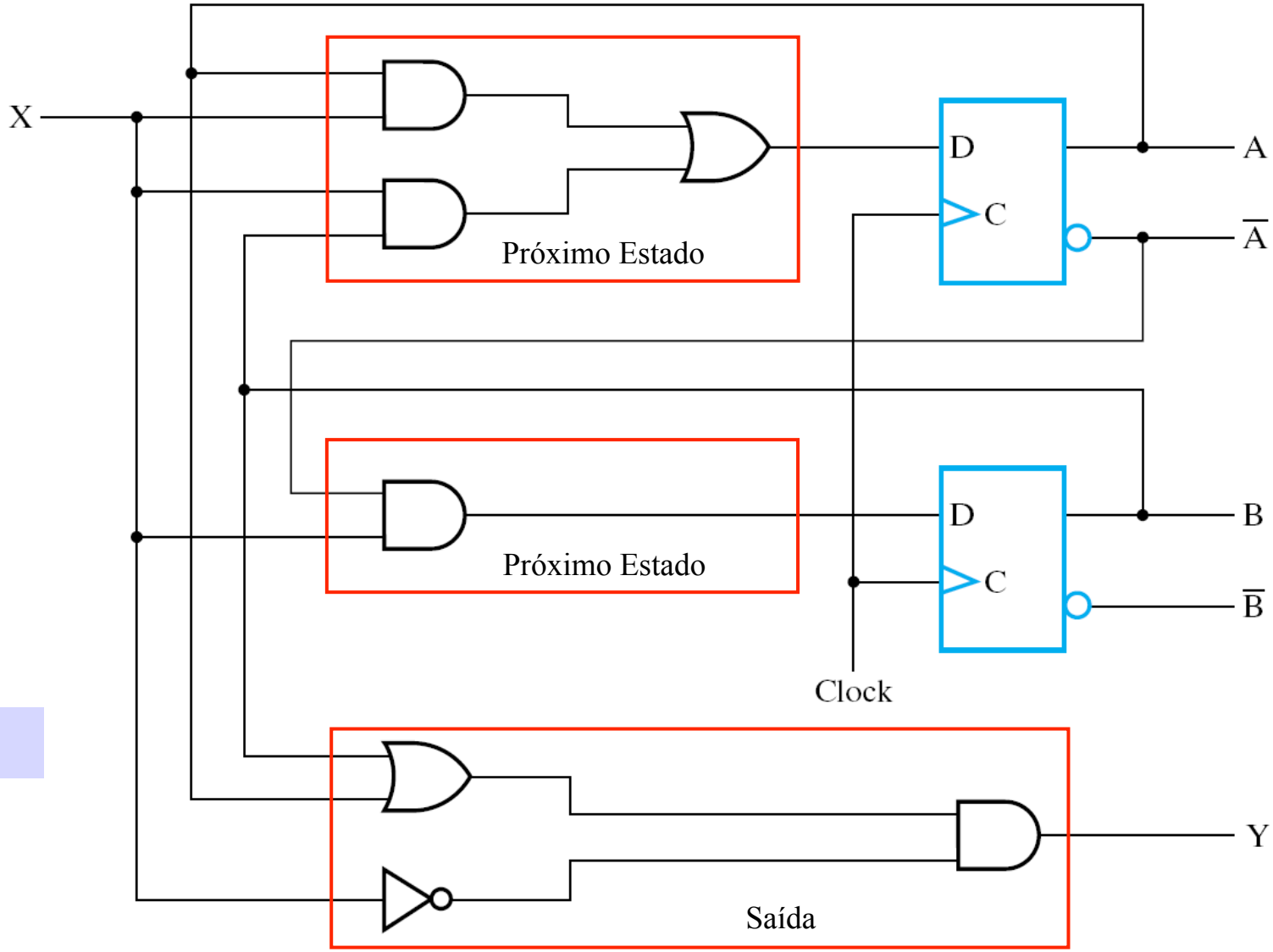
Resolução para o Exemplo 1

Equações Booleanas para as funções:

$A(t+1) = A(t)X(t) + B(t)X(t)$

$B(t+1)=A'(t)X(t)$

$Y(t)=X'(t)(A(t) + B(t))$



Características de uma Tabela de Estados

Tabela de Estados: uma tabela com múltiplas variáveis contendo as seguintes quatro seções:

- **Estado Atual** – os valores das variáveis de estado para cada estado permitido
- **Entrada** – as combinações de entradas permitidas
- **Próximo Estado** – o valor do estado no tempo ($t+1$) baseado no *Estado Atual* (t) e na *Entrada* (t)
- **Saída** – o valor da saída como uma função do *Estado Atual* (t) e (às vezes) da *Entrada* (t)

Fazendo uma analogia com uma **Tabela da Verdade**:

- as entradas são a ***Entrada*** e o ***Estado Atual***
- e as saídas são a ***Saída*** e o ***Próximo Estado***

Tabela de Estados para o Exemplo 1

Estado Atual		Entrada	Próximo Estado		Saída
A	**B**	**X**	**A**	**B**	**Y**
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	0	0
1	1	0	0	0	1
1	1	1	1	0	0

Note que, para toda combinação possível de **Entrada**, cada **Estado Atual** é repetido, formando uma combinação unidimensional

A **Tabela de Estados** pode ser obtida a partir das equações do **Próximo Estado** e da **Saída**

$A(t+1) = A(t)X(t)+B(t)X(t);$ $B(t+1) = A'(t)X(t);$ $Y(t) = X'(t)(A(t)+B(t))$

Tabela Bidimensional de Estados para o Exemplo 1

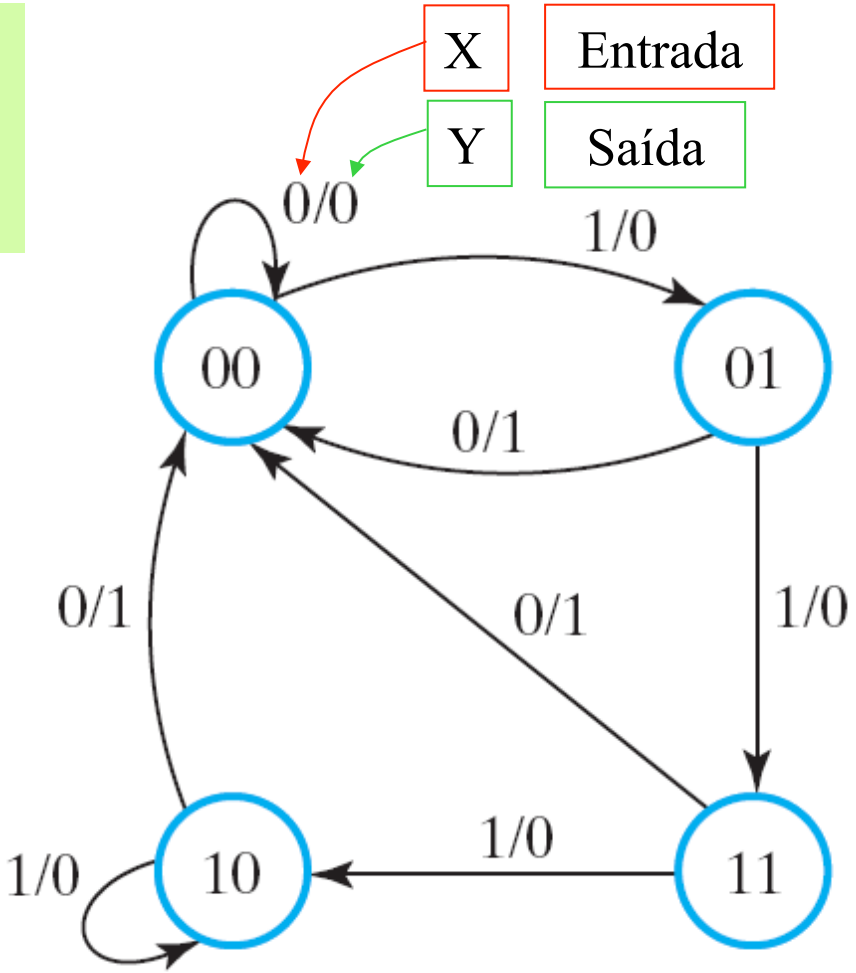
Estado Atual		Próximo Estado				Saída	
		X = 0		X= 1		X = 0	X = 1
A	B	A	B	A	B	Y	Y
0	0	0	0	0	1	0	0
0	1	0	0	1	1	1	0
1	0	0	0	1	0	1	0
1	1	0	0	1	0	1	0

Nesta tabela, os valores do **Estado Atual** ficam nas colunas da esquerda, enquanto que os valores da **Entrada** estão na linha superior, formando uma tabela bidimensional, que se casa bem com um **Mapa de Karnaugh** do tipo A\BX (2x4) ou AB\X (4x2)

Diagrama de Estados para o Exemplo 1

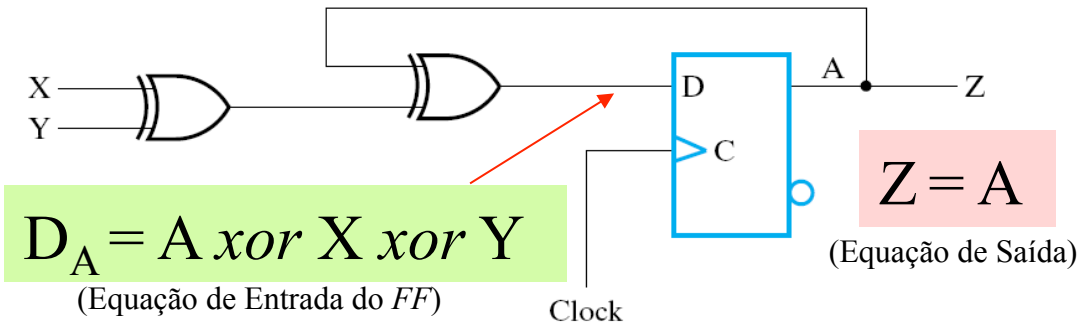
As informações de uma Tabela de Estados podem ser representadas graficamente na forma de um **Diagrama de Estados**

- Os números binários dentro dos círculos identificam o **Estado** dos *FFs*
- Para cada transição de estados há uma seta partindo do **Estado Atual** em direção ao **Próximo Estado**
- Os rótulos sobre as setas indicam o valor da **Entrada** que produziu a transição (ou não) e a **Saída** correspondente (=> Entrada/Saída)



Circuitos sequenciais em que as **Saídas** dependem das **Entradas**, bem como do **Estado Atual**, são conhecidos como ***Máquinas de Mealy***

Diagrama de Estados para o Exemplo 2

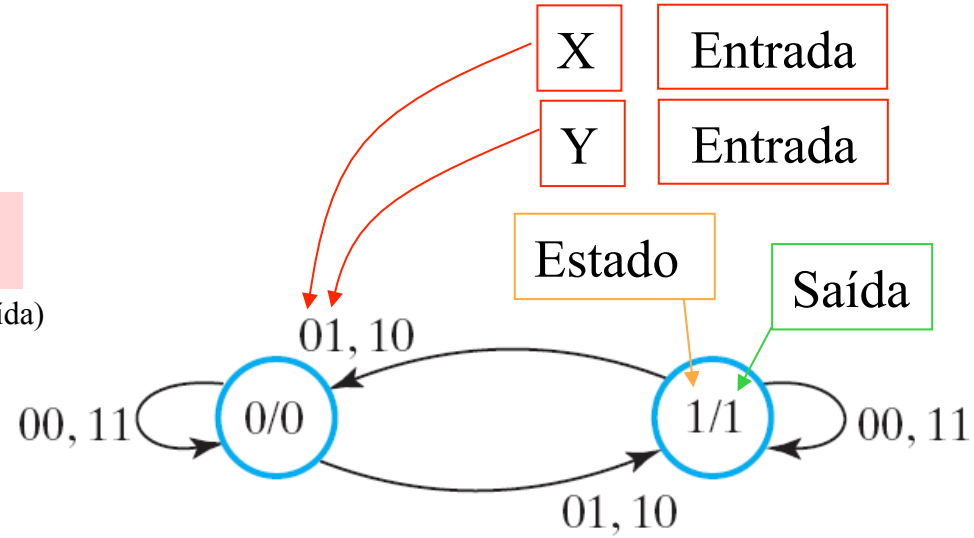


$D_A = A\ xor\ X\ xor\ Y$

(Equação de Entrada do *FF*)

$Z = A$

(Equação de Saída)



Estado Atual	Entradas		Próximo Estado	Saída
A	X	Y	A	Z
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	0
1	0	0	1	1
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Neste caso, a **Saída Z** é uma cópia do **Estado Atual A**

O diagrama acima mostra que enquanto as entradas X e Y tiverem o mesmo valor (00 ou 11), o circuito permanece no mesmo estado

Circuitos sequenciais em que as **Saídas** dependem apenas do **Estado Atual** são conhecidos como ***Máquinas de Moore***

Máquina de Estados Finitos

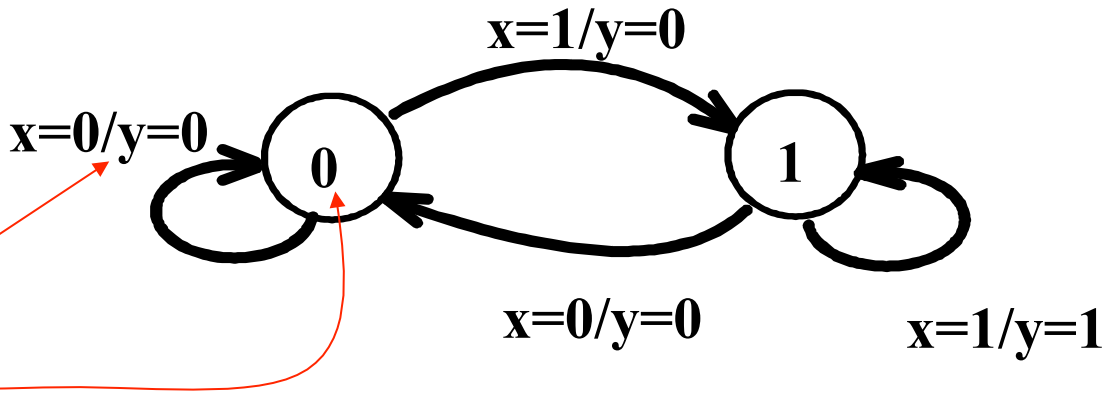
Circuitos Sequenciais ou Máquinas Sequenciais são também conhecidas como **Máquinas de Estados Finitos** – MEF ou FSM (*Finite State Machines*). Há dois modelos formais:

- **Modelo de Moore**
  - Formalizada por E. F. Moore
  - Saídas são funções **APENAS** dos estados, e
  - Normalmente são especificadas no círculo do estado

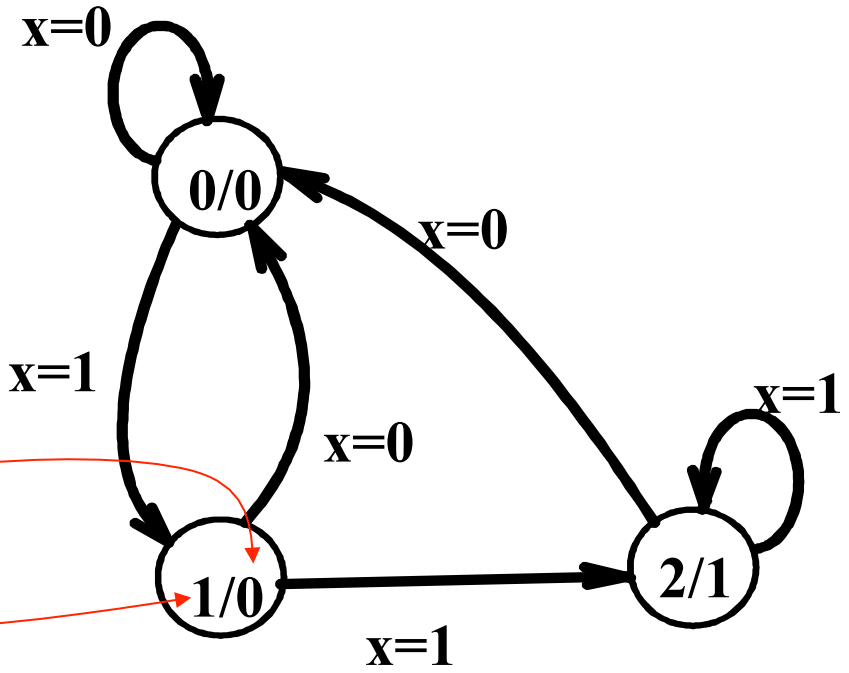
- **Modelo de Mealy**
  - Formalizada por G. Mealy
  - Saídas são funções das entradas **E** dos estados, e
  - Normalmente são especificadas nos arcos de transição dos estados

Comparação entre Modelos de MEFs

- O **Diagrama de Estados** do **Modelo de Mealy** mapeia as entradas e o estado para as saídas



- O **Diagrama de Estados** do **Modelo de Moore** mapeia os estados para as saídas



Exemplo de Tabelas Moore e Mealy

- A **Tabela de Estados** do **Modelo de Mealy** mapeia as **entradas e** o **estado** para as **saídas**

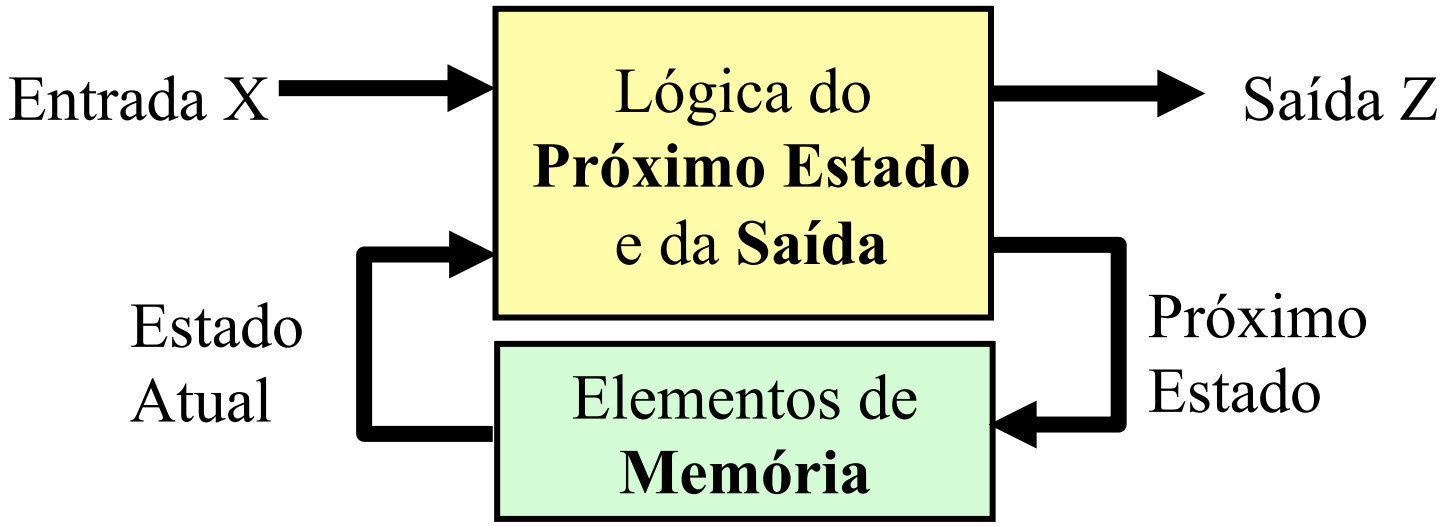
Estado Atual	Próximo Estado		Saída	
	x=0	x=1	x=0	x=1
0	0	1	0	0
1	0	1	0	1

- A **Tabela de Estados** do **Modelo de Moore** mapeia o **estado** para as **saídas**

Estado Atual	Próximo Estado		Saída
	x=0	x=1	
0	0	1	0
1	0	2	0
2	0	2	1

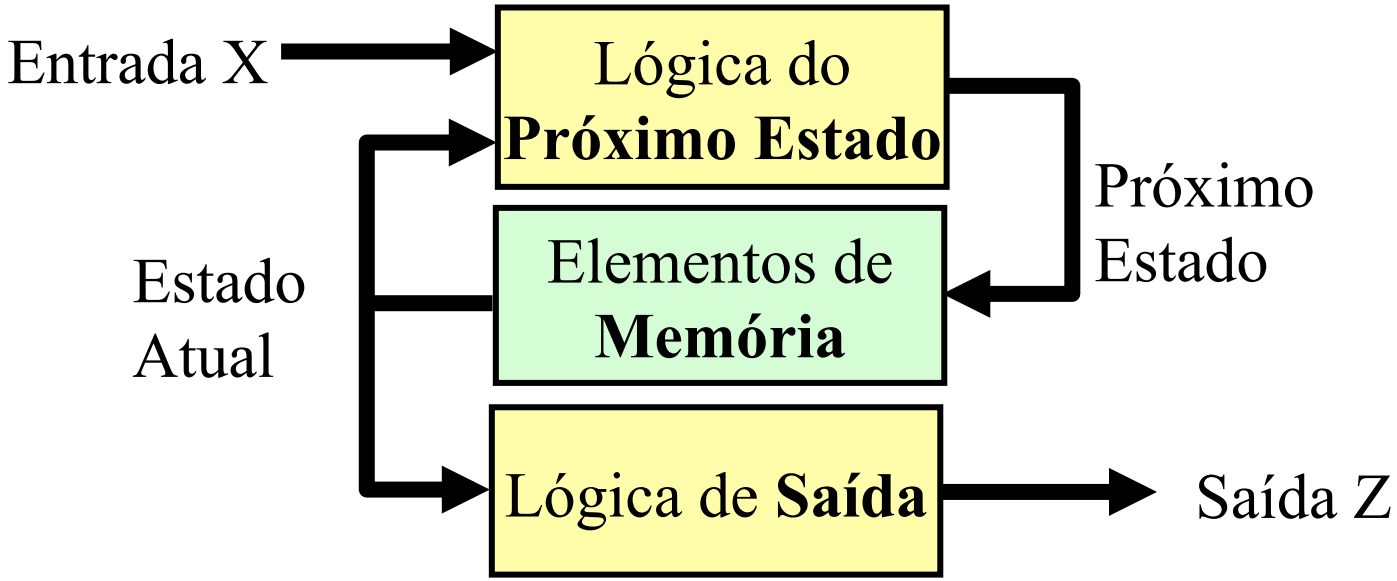
Circuito Sequencial Tipo Mealy

- Circuito sequencial tipo Mealy
  - A **Saída** é função do **Estado Atual** e da **Entrada**



Circuito Sequencial Tipo Moore

- A **Saída** depende apenas do **Estado Atual**
  - A Saída não depende [diretamente] da Entrada
- A Lógica Combinacional é dividida em duas partes:
  - A lógica do Próximo Estado depende da Entrada e do Estado Atual
  - A lógica de Saída depende apenas do Estado Atual



Projeto de Circuito Sequencial

- O projeto de **Circuitos Sequenciais com *clock*** começa com um conjunto de **Especificações** e culmina com um **Diagrama Lógico** ou uma lista de **Funções Booleanas**
- Em contraste com um **Circuito Combinacional**, que é totalmente especificado por uma **Tabela da Verdade**, um **Circuito Sequencial** precisa de uma **Tabela de Estados** para sua especificação
- Por isso, o primeiro passo no projeto de um **Circuito Sequencial** é obter uma **Tabela de Estados** ou um **Diagrama de Estados**

Procedimentos para um Projeto

1. Obter um **Diagrama de Estados** ou uma **Tabela de Estados** da **Especificação** ou **Descrição do Problema**
2. Se existir somente um **Diagrama de Estados** do passo 1, obter uma **Tabela de Estados**
3. Atribuir **Códigos Binários** aos Estados
4. Deduzir as **Equações de Entrada** dos *FFs* a partir de cada caso de **Próximo Estado** da **Tabela de Estados**
5. Deduzir as **Equações de Saída** a partir de cada caso de **Saída** da **Tabela de Estados**
6. Simplificar as **Equações de Entrada** dos *FFs* e as **Equações de Saída**
7. Desenhar o **Diagrama Lógico** com ***FF* tipo *D*** e **Portas Combinacionais**

Exemplo: Reconhecedor de Sequência

1. Um **Reconhecedor de Sequência** é um circuito sequencial que produz um valor de saída distinto sempre que determinado padrão de **Símbolos de Entrada** ocorrer em sequência, i. e., ele ***reconhece*** a ocorrência de um **Padrão de Entrada** (*1s* e *0s*)
2. Primeiramente será apresentado um procedimento específico para **Reconhecedores de Sequências** visando converter uma **Formulação do Problema** em **Diagrama de Estados**
3. A seguir, o **Diagrama de Estados** vai ser convertido numa **Tabela de Estados** a partir da qual o **Circuito Sequencial** será projetado

Procedimento para Reconhecedor de Estados

Para obter um **Diagrama de Estados** de um **Reconhecedor de Sequência**:

- comece com um **estado inicial** em que **NENHUMA** parte da **sequência desejada** tenha ocorrido (normalmente estado "***reset***")
- acrescente um estado que reconheça a ocorrência do primeiro símbolo
- acrescente estados que reconheçam a ocorrência sucessiva dos outros símbolos
- o **estado final** representa a ocorrência da **sequência desejada** (possivelmente menos o valor final de entrada)
- acrescente **arcos de transições de estados** especificando o que ocorre quando um símbolo **NÃO** pertencente à **sequência desejada** aparecer
- acrescente outros arcos correspondentes a entradas não pertencentes à sequência desejada, mas que levam a estados representando uma **subsequência de entrada**

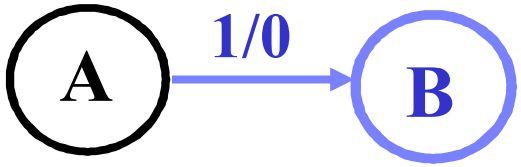
- O último passo é necessário porque o circuito deve reconhecer a **sequência de entrada** ***independentemente de onde ela ocorre dentro de uma sequência geral aplicada desde o reset***

Exemplo de Reconhecedor de Sequência

- Exemplo: **Reconhecer a sequência 1101**
  - note que a sequência **1111101** contém a sequência **1101** e que "**11**" é uma subsequência da sequência desejada
- Assim, a máquina sequencial precisa *lembrar* quando os últimos dois *uns* ocorrerem, mesmo que precedidos por outros símbolos não desejados
- Também a sequência **1101101** contém **1101** tanto na subsequência inicial como na final com uma sobreposição, ou seja, **1101101** ou **1101101**
- E o **1** no meio, **1101101**, está em ambas subsequências
- A sequência **1101** deve ser reconhecida toda vez que ela ocorrer numa sequência de entrada

Exemplo: Reconhecer 1101

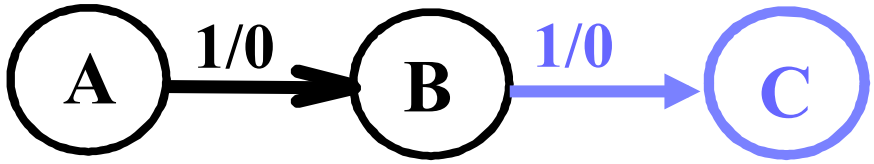
- Defina os estados para a sequência a ser reconhecida:
  - assumindo que ela começa com o primeiro símbolo,
  - continua através de cada símbolo na sequência a ser reconhecida, e
  - gera saída 1 para sinalizar a ocorrência de uma sequência completa,
  - com saída 0, caso contrário
- Começando no estado inicial (arbitrariamente denominado "A"):
  - Acrescente um estado que reconheça o primeiro "1"



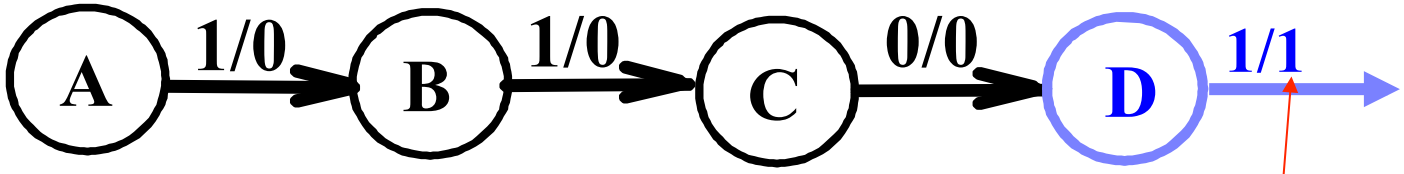
  - O estado "A" é o estado inicial, e o estado "B" é o estado que representa o fato de que o “primeiro” “1” na subsequência de entrada ocorreu. O símbolo de saída "0" significa que a sequência desejada ainda não se completou

Exemplo: Reconhecer 1101 (continuação)

- Após outro 1, obtém-se:
  - C é o estado obtido quando a sequência de entrada possui dois "1"s.

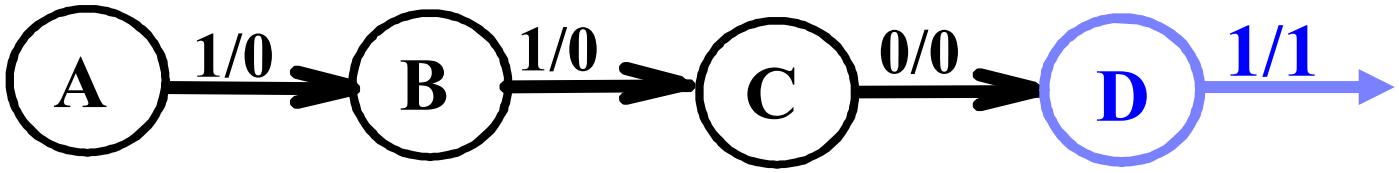


- Finalmente, depois de 110 seguido de um 1, obtém-se:

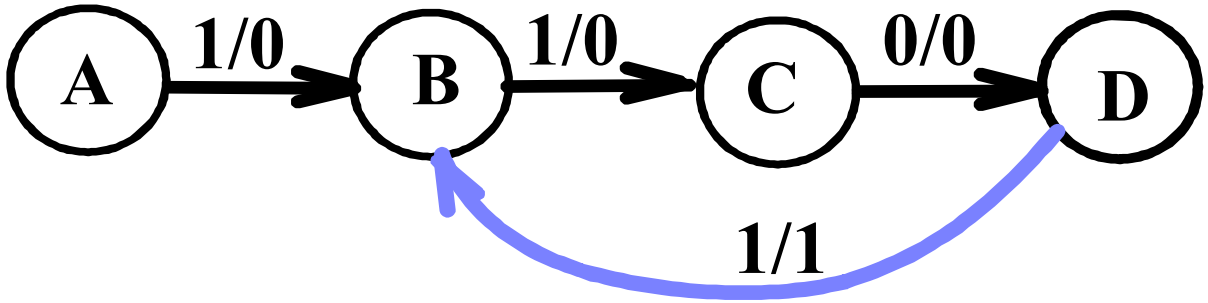


  - Os arcos de transição são usados para denotar a função de saída (*Modelo de Mealy*)
  - A saída 1 no arco associado ao estado D significa que a sequência foi reconhecida
  - Para qual estado deveria o arco a partir de D ir? Lembre-se: 1101<u>1101</u> ?
  - Note que D é o último estado, mas a saída 1 ocorre devido à entrada aplicada em D. Isto ocorre quando se assume um *Modelo de Mealy*

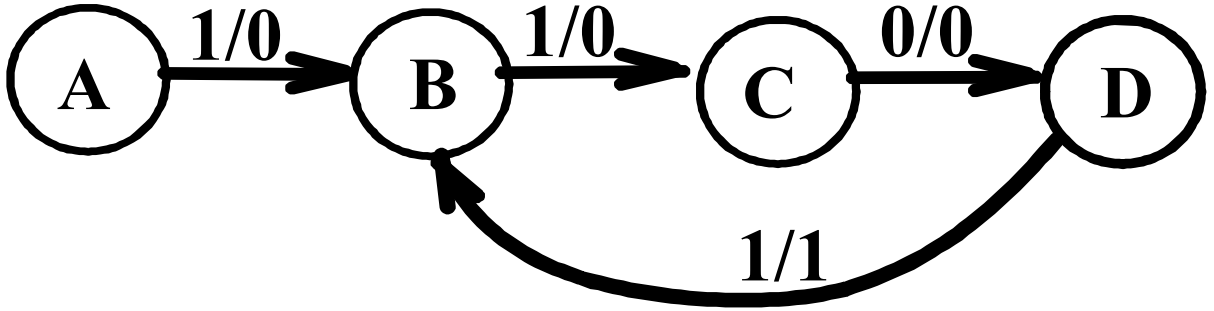
Exemplo: Reconhecer 1101 (continuação)



- O último 1 na sequência reconhecida 1101 é claramente uma subsequência inicial de 1101. Mas esse 1 vem depois de um 0 que, nesta ordem (01), não é a subsequência inicial de 1101. Assim, ele deve representar o *mesmo estado obtido a partir do estado inicial depois do primeiro 1 (estado B)*. Portanto, obtém-se:



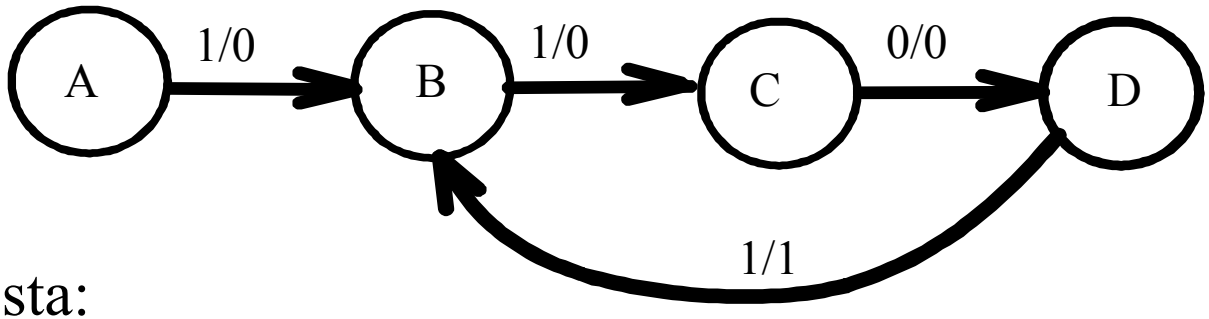
Exemplo: Reconhecer 1101 (continuação)



- Os estados têm os seguintes significados abstratos:
  - A: Nenhuma subsequência da sequência desejada ocorreu
  - B: A subsequência 1 ocorreu
  - C: A subsequência 11 ocorreu
  - D: A subsequência 110 ocorreu
  - O 1/1 no arco partindo de D para B significa que o último 1 ocorreu e, assim, a sequência foi reconhecida

Exemplo: Reconhecer 1101 (continuação)

- Outros arcos devem ser acrescentados a cada estado para as entradas não consideradas ainda. Que arcos estão faltando?



- Resposta:

Arco "0" partindo de A

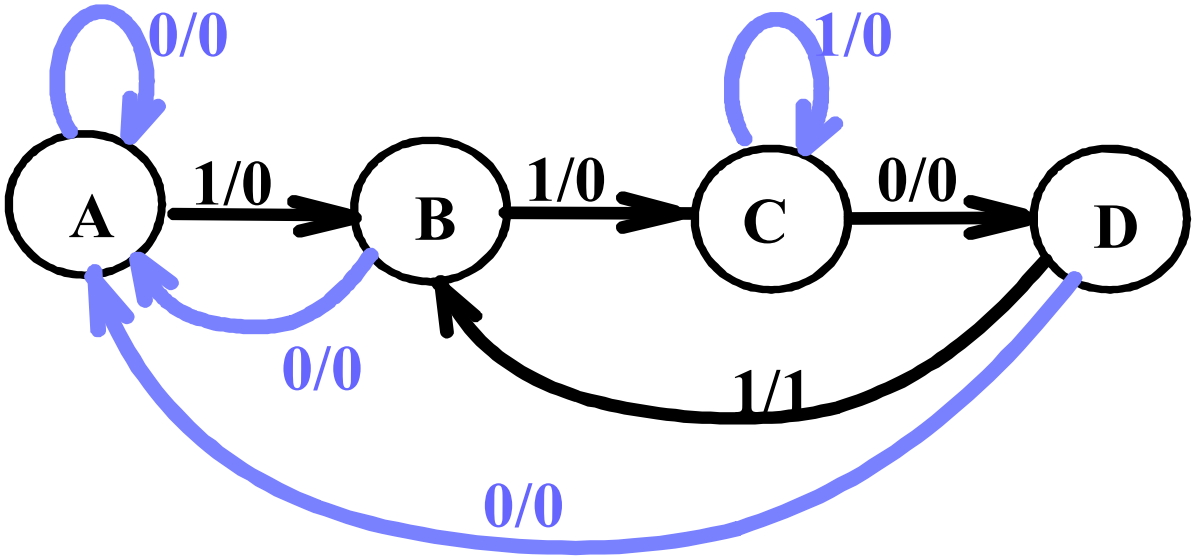
Arco "0" partindo de B

Arco "1" partindo de C

Arco "0" partindo de D

Exemplo: Reconhecer 1101 (continuação)

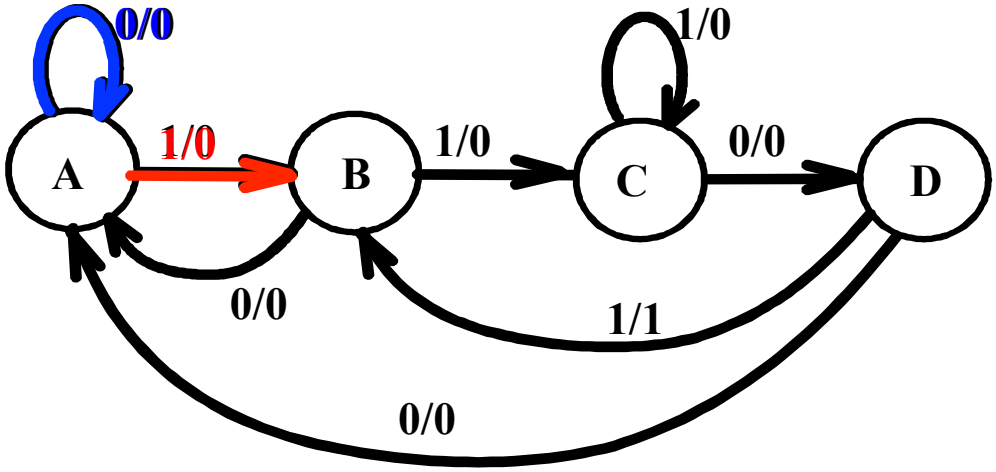
- Os arcos de transição de estados devem representar a ocorrência de uma subsequência de entrada. Assim, obtém-se:



- Note que o arco 1 do estado C para ele mesmo implica que já ocorreram *dois ou mais 1s*.

Formulação: Encontrar a Tabela de Estados

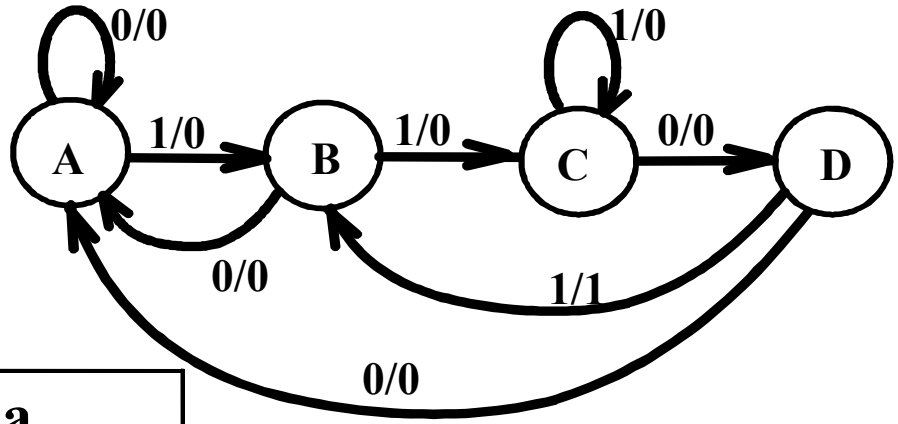
- Do **Diagrama de Estados**, é possível preencher a **Tabela de Estados**
- Há quatro estados, uma entrada e uma saída. Será escolhida a forma com quatro linhas, uma para cada Estado Atual
- Partindo do estado A, as transições de entrada 0 e 1 são preenchidas juntamente com as respectivas saídas



Estado Atual	Próximo Estado x=0	Próximo Estado x=1	Saída x=0	Saída x=1
A	A	B	0	0
B				
C				
D				

Formulação: Encontrar a Tabela de Estados

- Com o **Diagrama de Estados**, a **Tabela de Estados** é completada



Estado Atual	Próximo Estado x=0	Próximo Estado x=1	Saída x=0	Saída x=1
A	A	B	0	0
B	A	C	0	0
C	D	C	0	0
D	A	B	0	1

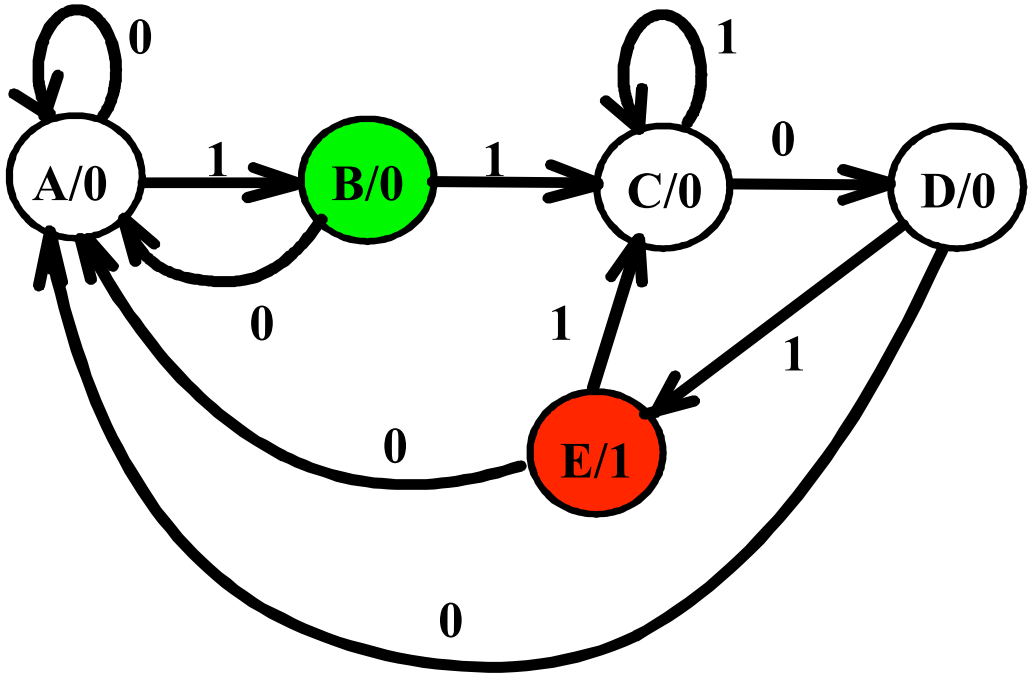
- Como seriam o **Diagrama de Estados** e a **Tabela de Estados** para o ***Modelo de Moore***?

Exemplo: *Modelo Moore* para a Sequência 1101

- Para o *Modelo de Moore*, as saídas estão associadas aos estados
- É necessário acrescentar um estado "E" com o valor de saída 1 para o último 1 da sequência de entrada desejada
  - Este novo estado E, embora similar a B, deve gerar uma saída 1
  - Portanto, E é um estado com um comportamento diferente de B
- O *Modelo de Moore* para um Reconhecedor de Sequência normalmente possui *mais estados* que o *Modelo de Mealy* equivalente

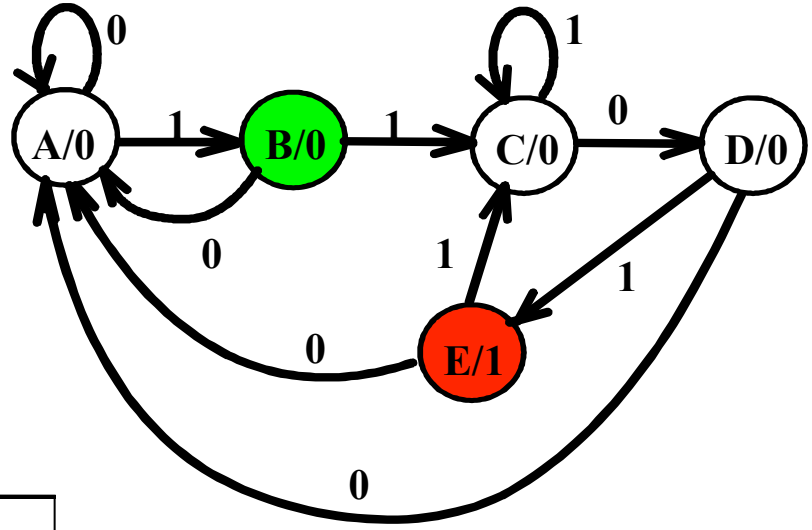
Exemplo: Modelo Moore (continuação)

- Para o *Modelo de Moore*, as saídas são marcadas nos estados
- Agora os arcos mostram apenas as transições de estados
- Acrescentar um novo estado **E** para produzir uma saída 1
- Note que o novo estado, **E**, apresenta o mesmo comportamento do estado **B** para o **Próximo Estado**. Mas ele apresenta uma saída diferente para o **Estado Atual**. Portanto, estes estados representam uma *abstração diferente* da história de entradas



Exemplo: Modelo Moore (continuação)

- A **Tabela de Estados** é mostrada abaixo
- Ajuda mnemônica para lembrar o número maior de estados no *Modelo de Moore*: "*Moore is More*".



Estado Atual	Próximo Estado x=0	Próximo Estado x=1	Saída Z
A	A	B	0
B	A	C	0
C	D	C	0
D	A	E	0
E	A	C	1

Atribuição de Códigos Binários aos Estados

Tabela de Estados Original da Máquina de Mealy para o Reconhecedor de Sequência 1101

Present State	Next State		Output Z	
	X = 0	X = 1	X = 0	X = 1
A	A	B	0	0
B	A	C	0	0
C	D	C	0	0
D	A	B	0	1

Tabela de Estados com Atribuição de Códigos Binários da Máquina de Mealy para o Reconhecedor de Sequência 1101

Present State	Next State		Output *Z*	
AB	X = 0	X = 1	X = 0	X = 1
00	00	01	0	0
01	00	11	0	0
11	10	11	0	0
10	00	01	0	1

D_A

Atribuição de Códigos Binários aos Estados

Tabela de Estados Original da Máquina de Mealy para o Reconhecedor de Sequência 1101

Present State	Next State		Output Z	
	X = 0	X = 1	X = 0	X = 1
A	A	B	0	0
B	A	C	0	0
C	D	C	0	0
D	A	B	0	1

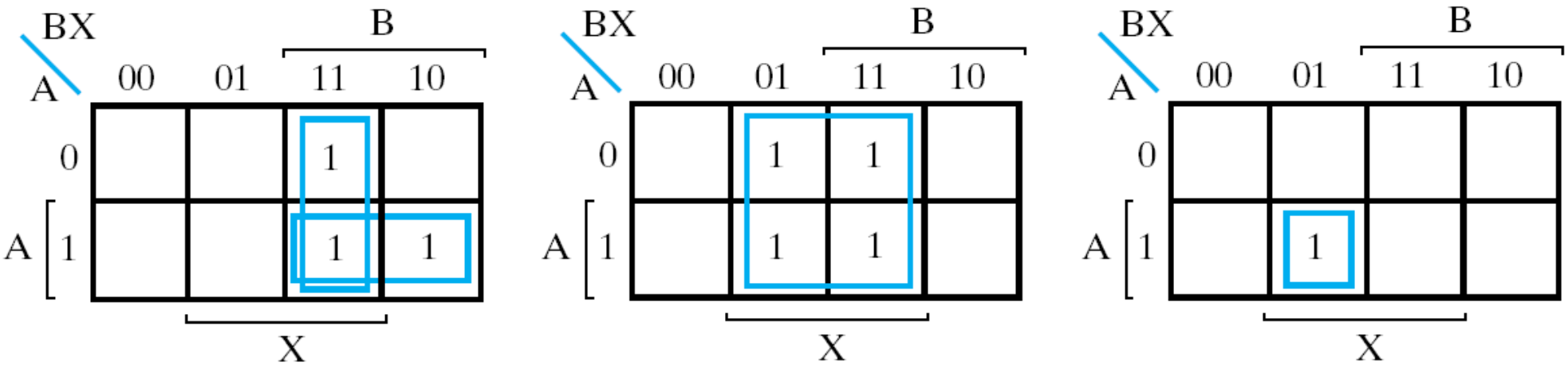
Tabela de Estados com Atribuição de Códigos Binários da Máquina de Mealy para o Reconhecedor de Sequência 1101

Present State	Next State		Output *Z*	
AB	X = 0	X = 1	X = 0	X = 1
00	00	01	0	0
01	00	11	0	0
11	10	11	0	0
10	00	01	0	1

D_B

Equações de Entradas dos FF e Saída

Considerando-se que serão usados FF tipo D, os Diagramas de Karnaugh são elaborados para a obtenção das duas Equações de Entrada dos FFs e a Equação de Saída do Reconhecedor

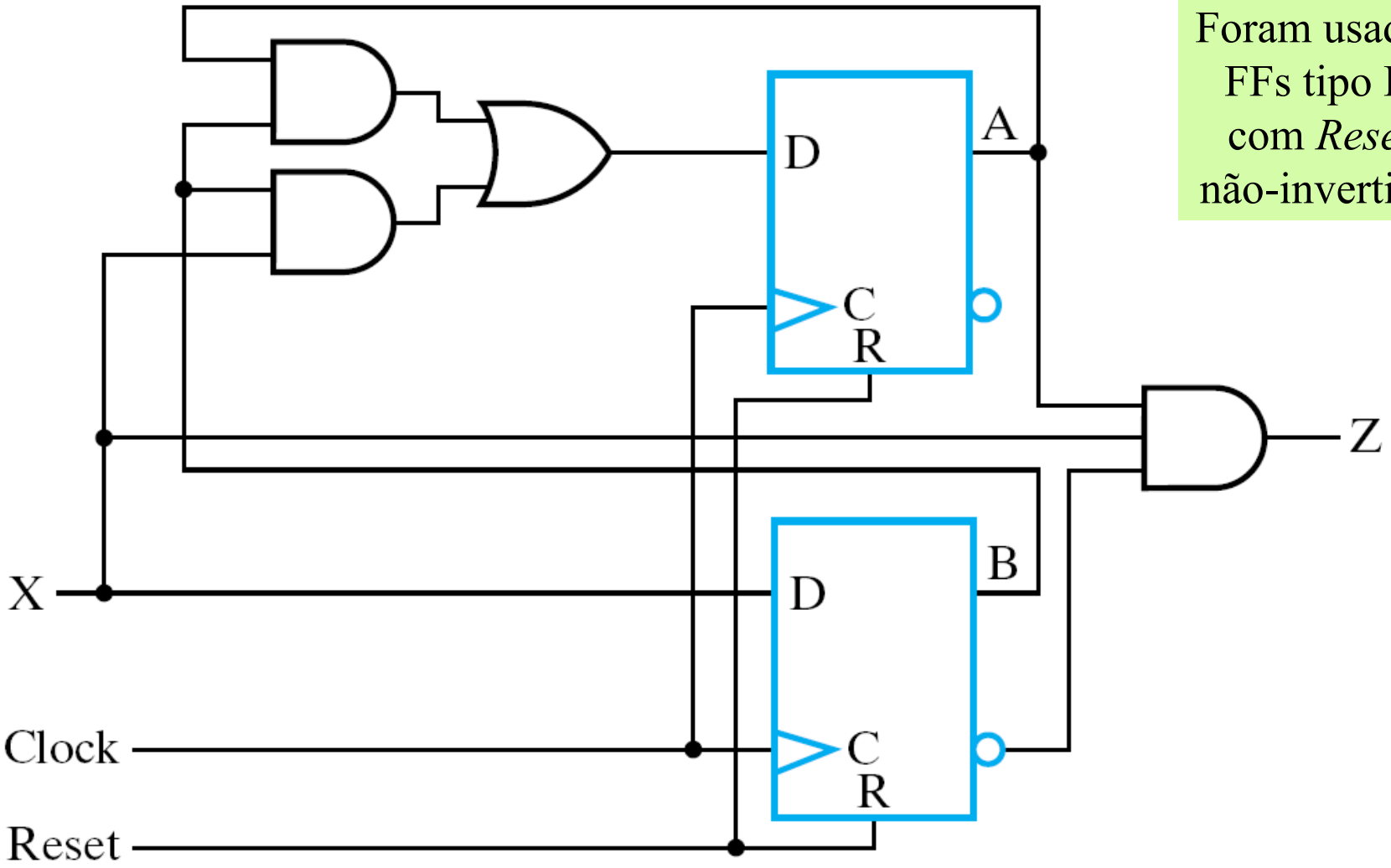


$D_A = AB + BX$

$D_B = X$

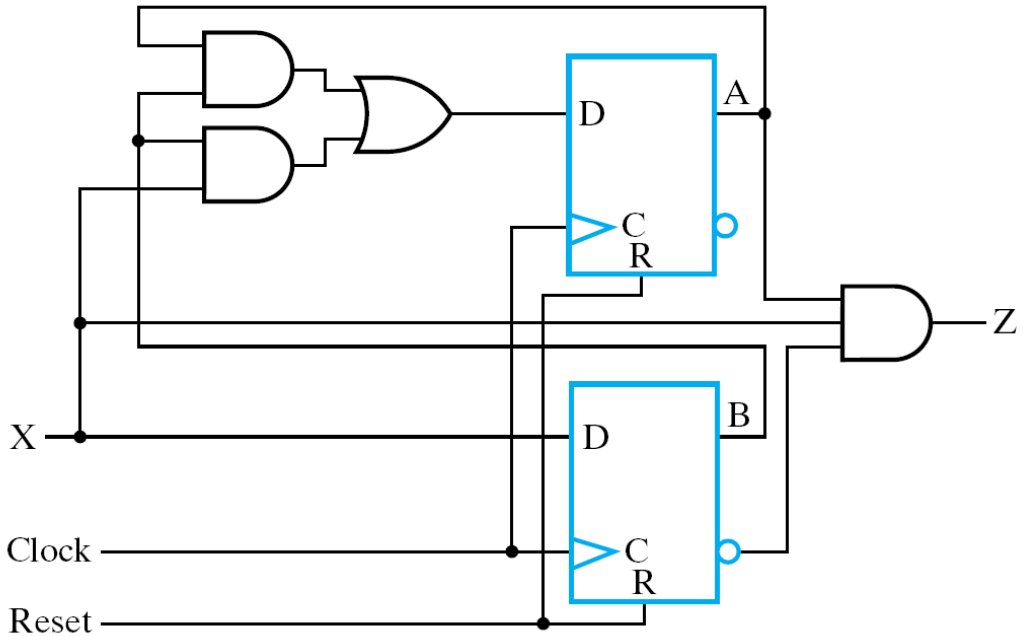
$Z = A\bar{B}X$

Diagrama Lógico do Reconhecedor 1101



Foram usados FFs tipo D com *Reset* não-invertido

Diagrama Lógico do Reconhecedor 1101

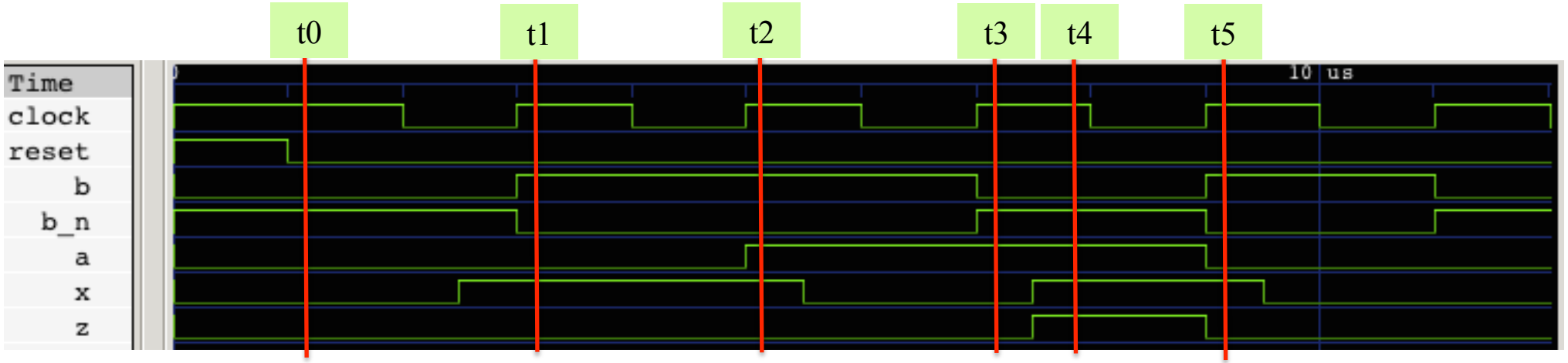


Em t0, X=0, mas em t1 e t2, X=1;

Em t3, X=0 e tanto A quanto B_n têm valor 1;

Como esta é uma máquina de Mealy, em t4, quando X assume o valor 1, a saída Z também torna-se 1 (Z=AB_nX), antes mesmo da borda de subida do Clock!

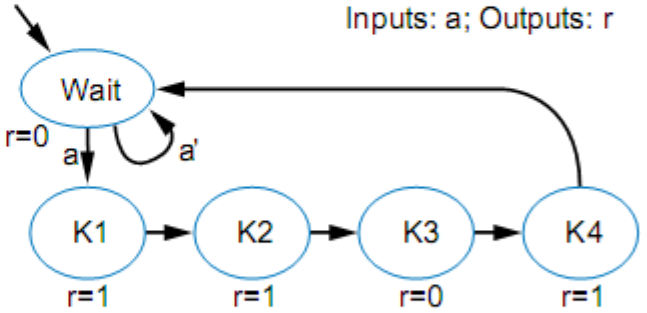
Em t5, embora X=1, Z=0 porque A e B_n foram para 0!



Exemplo de FSM: Chave de Carro Segura

FSM Example: Secure Car Key

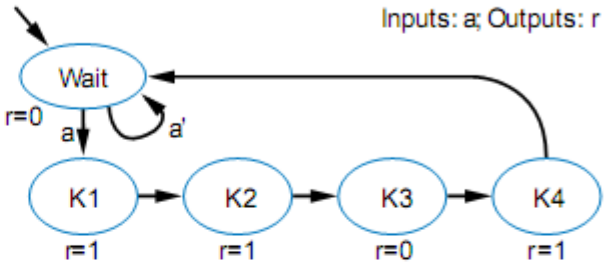
- Many new car keys include tiny computer chip
  - When car starts, car's computer (under engine hood) requests identifier from key
  - Key transmits identifier
    - If not, computer shuts off car
- FSM
  - Wait until computer requests ID (a=1)
  - Transmit ID (in this case, 1101)

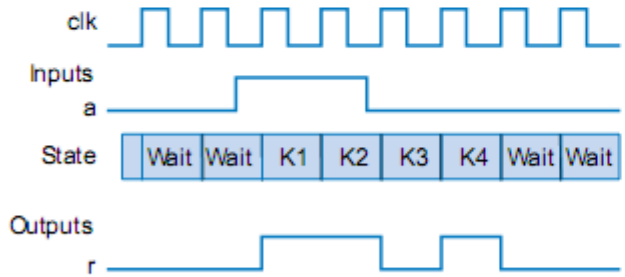


FSM: Chave de Carro Segura (cont.)

FSM Example: Secure Car Key (cont.)

- Nice feature of FSM
  - Can evaluate output behavior for different input sequence
  - Timing diagrams show states and output values for different input waveforms





Q: Determine states and r value for given input waveform: