

# REDES E SISTEMAS DE TELECOMUNICAÇÕES

RT001

## TEORIA DA INFORMAÇÃO E CODIFICAÇÃO DE CANAL (PARTE 1)

Prof. Dr. Estevan Lopes

Inatel

CAMINHOS  
QUE CONECTAM  
COM O FUTURO

## TRILHA DE CONTEÚDOS



- LIMITES FUNDAMENTAIS NA TEORIA DA INFORMAÇÃO
- CÓDIGOS DE BLOCO LINEARES
- CÓDIGOS REED-SOLOMON
- CÓDIGOS CONVOLUCIONAIS

## REFERÊNCIAS BIBLIOGRÁFICAS

- HAYKIN, SIMON – COMMUNICATIONS SYSTEMS. 4<sup>TH</sup> ED. JOHN WILEY & SONS.
- SKLAR, BERNARD, DIGITAL COMMUNICATIONS: FUNDAMENTALS AND APPLICATIONS. 2 ED. NEW JERSEY: PRENTICE HALL.

## AVALIAÇÃO

- SÉRIE DE EXERCÍCIOS (100 PONTOS)

## OBJETIVOS GERAIS

- CONHECER OS LIMITES IMPOSTOS PELA TEORIA DA INFORMAÇÃO.
- CONHECER OS PRINCIPAIS TIPOS DE CÓDIGOS PARA CORREÇÃO DE ERRO.

# CAPÍTULO 1



O trabalho de **Shannon** (\*1916-†2001) sobre a teoria da informação, publicado em 1948, estabelece a base teórica para que sistemas de comunicações sejam eficientes e confiáveis.

Os principais tópicos abordados são:

- A *Entropia* como uma medida básica de informação.
- O Teorema da *Codificação de Fonte* e algoritmos de compactação de dados.
- O Teorema da *Codificação de Canal* como base para comunicações confiáveis.
- O Teorema da *Capacidade de Informação* como a base no compromisso entre a BW do canal e a RSR.

## 1.1. INTRODUÇÃO

No contexto das comunicações, a teoria da informação fornece uma modelagem matemática que permite responder duas questões fundamentais:

1. Qual é a complexidade irredutível abaixo da qual um sinal não pode ser comprimido?
2. Qual é a máxima taxa de transmissão para uma comunicação confiável em um canal ruidoso?



## 1.2. INCERTEZA, INFORMAÇÃO E ENTROPIA

Suponha que um *experimento probabilístico* envolva a observação da saída de uma fonte de eventos discretos em unidades de intervalo de tempo. Estes eventos podem ser modelados como variáveis discretas aleatórias ( $s_k$ ) que fazem parte de um conjunto ou *alfabeto* ( $\mathbf{S}$ ). Assim,  $\mathbf{S} = \{s_0, s_1, \dots, s_{K-1}\}$  com probabilidades  $P(\mathbf{S} = s_k) = p_k$ , para  $k = 0, 1, \dots, K-1$ , que satisfaz a igualdade

$$\sum_{k=0}^{K-1} p_k = 1$$

Se os símbolos emitidos pela fonte são Estatisticamente Independentes: *fonte discreta sem memória*.

A quantidade de informação produzida pela fonte está associada à *incerteza ou surpresa*. Se não há surpresa não há informação. A quantidade de informação,  $I(s_k)$ , obtida por um evento,  $s_k$ , é definida como

$$I(s_k) = \log_2\left(\frac{1}{p_k}\right)$$



## 1.2. INCERTEZA, INFORMAÇÃO E ENTROPIA

A quantidade de informação apresenta as seguintes propriedades:

1.  $I(s_k) = 0$  para  $p_k = 1$
2.  $I(s_k) \geq 0$  para  $0 \leq p_k \leq 1$
3.  $I(s_k) > I(s_l)$  então  $p_k < p_l$
4.  $I(s_k s_l) = I(s_k) + I(s_l)$

$H(\mathbf{S})$  que é definida como *entropia*. A entropia determina a quantidade média de informação por símbolo (evento) da fonte. Também pode ser entendida como uma medida da *informação* contida numa mensagem.

$$H(\mathbf{S}) = E[I(s_k)]$$

$$H(\mathbf{S}) = \sum_{k=0}^{K-1} p_k \log_2 \left( \frac{1}{p_k} \right)$$



## 1.2. INCERTEZA, INFORMAÇÃO E ENTROPIA

Nos canais de comunicação digital, uma fonte de grande interesse é a fonte binária sem memória. Para essa fonte é interessante o entendimento do comportamento da entropia em função da probabilidade de ocorrência dos eventos "0" e "1". Esse comportamento é mostrado no Exemplo 1.1.

### EXEMPLO 1.1

Considere uma fonte discreta sem memória que emite os símbolos  $s_0 = 0$  e  $s_1 = 1$ , com probabilidades  $p_0$  e  $p_1$ , respectivamente. A entropia desta fonte é

$$H(\mathbf{S}) = \sum_{k=0}^{K-1} p_k \log_2 \left( \frac{1}{p_k} \right) = p_0 \log_2 \left( \frac{1}{p_0} \right) + p_1 \log_2 \left( \frac{1}{p_1} \right)$$

mas  $p_1 = 1 - p_0$ .

$$H(\mathbf{S}) = p_0 \log_2 \left( \frac{1}{p_0} \right) + (1 - p_0) \log_2 \left( \frac{1}{1 - p_0} \right)$$

Consequentemente, a entropia da fonte torna-se

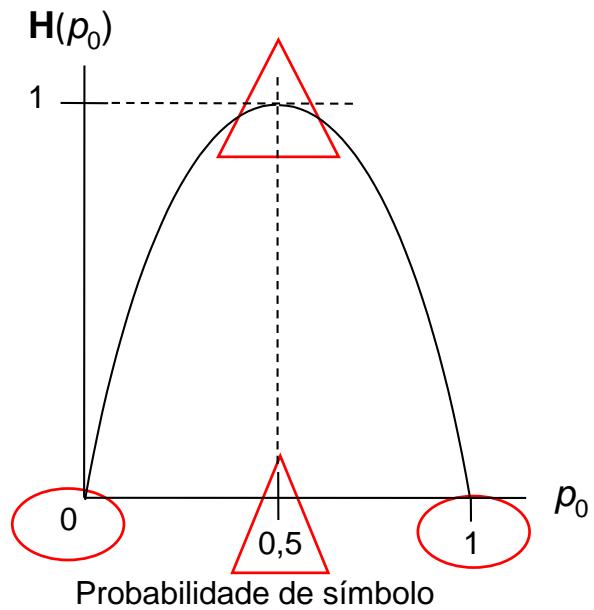
$$H(p_0) = -p_0 \log_2 p_0 - (1 - p_0) \log_2 (1 - p_0)$$



## 1.2. INCERTEZA, INFORMAÇÃO E ENTROPIA

Onde,  $H(p_0)$  é chamada de *função entropia* e

1. Quando  $p_0 = 0$ ,  $H(\mathbf{S}) = 0$ .
2. Quando  $p_1 = 1$ ,  $H(\mathbf{S}) = 0$ .
3. A entropia é máxima quando  $p_1 = p_0 = \frac{1}{2}$ .





### 1.3. TEOREMA DA CODIFICAÇÃO DE FONTE

A codificação de fonte é o processo pelo qual os dados de uma fonte discreta são representados de forma a permitir uma transmissão *eficiente*.

No caso dos sistemas de comunicações digitais é conveniente que dois requisitos funcionais sejam satisfeitos:

1. **As palavras códigos devem estar na forma binária.**
2. **As palavras códigos devem ser inequivocamente decodificáveis.**

**O comprimento médio das palavras códigos de uma codificação de fonte:**

$$\bar{L} = \sum_{k=0}^{K-1} p_k l_k$$

onde  $l_k$  é o número de bits da palavra código correspondente ao símbolo  $k$ .



### 1.3. TEOREMA DA CODIFICAÇÃO DE FONTE

#### PRIMEIRO TEOREMA DE SHANNON: TEOREMA DA CODIFICAÇÃO DE FONTE

Dada uma fonte discreta sem memória de entropia  $H(S)$ , o comprimento médio das palavras códigos para um esquema de codificação livre de distorção é limitado como

$$\bar{L} \geq H(S)$$

De acordo com o Primeiro Teorema de Shannon a entropia representa um limite fundamental do número médio de bits por símbolo necessários para representar uma fonte discreta sem memória. Em outras palavras, um esquema de codificação de fonte pode ser feito de modo que o comprimento médio das palavras códigos seja tão pequeno quanto à entropia, mas nunca menor do que ela.

#### Eficiência de Codificação:

$$\eta = \frac{H(S)}{\bar{L}}$$



## **1.4. COMPACTAÇÃO DE DADOS**

Para uma transmissão eficiente as informações redundantes devem ser removidas do sinal que será transmitido.

Neste texto os termos compactação e compressão apresentam significados distintos.

*Compactação* está associado a um processo onde não há perda de informação.

*Compressão* é usado nos processos em que se admite perda de informação.

Na compactação de dados, o processo de codificação de fonte é limitado, necessariamente, pela entropia da fonte.

### **▫ CÓDIGOS PREFIXOS**

Um código prefixo é um código de fonte com decodificação inequívoca, ou seja, sua decodificação não resulta em ambiguidade.

Um código prefixo é aquele que nenhuma palavra código é prefixo para qualquer outra palavra código.



## 1.4. COMPACTAÇÃO DE DADOS

A partir do apresentado na Tabela 1.1 verifica-se que:

O Código I não é um código prefixo.

O Código II é um código prefixo.

O Código III não é um código prefixo.

Tabela 1.1: Exemplos de códigos de fonte

Símbolo	Probabilidade	Código I	Código II	Código III
$S_0$	0,5	0	0	0
$S_1$	0,25	1	10	01
$S_2$	0,125	00	110	011
$S_3$	0,125	11	111	0111

A decodificação de um código prefixo pode ser feita de acordo com uma árvore de decisão



#### 1.4. COMPACTAÇÃO DE DADOS

Para o código II, a árvore de decisão está apresentada na Figura 1.2, com decodificação inequívoca e *instantânea*.

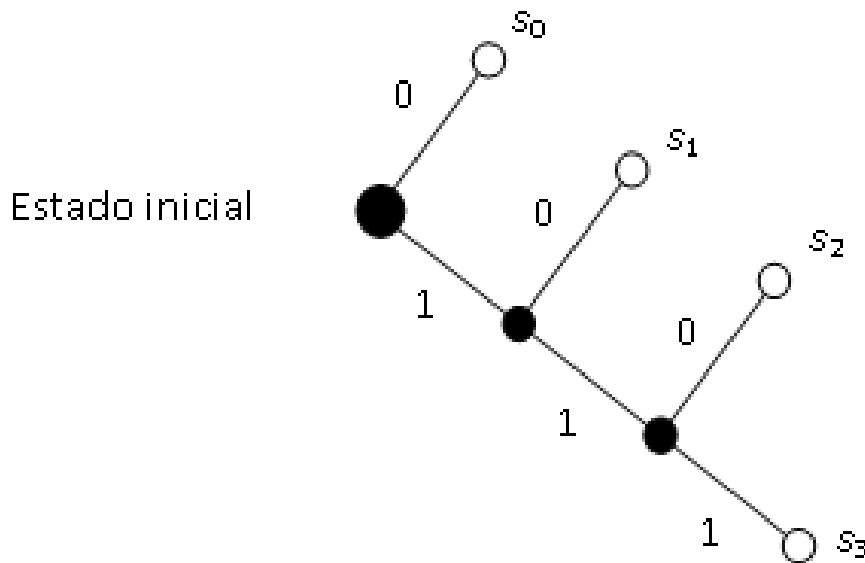


Figura 1.2: Árvore de decodificação para o Código II.

A sequência 1011111000... é inequivocamente descodificável e corresponde correspondente à sequência de símbolos

$s_1 s_3 s_2 s_0 s_0 \dots$



## 1.4. COMPACTAÇÃO DE DADOS

### ❖ ALGORITMO DE HUFFMAN

Propósito:

Atribuir códigos menores para símbolos mais frequentes e códigos maiores para símbolos menos frequentes. O procedimento de Huffman para a criação de um código prefixo consiste de um pequeno conjunto de regras. Essas regras são apresentadas no exemplo apresentado a seguir.

### EXEMPLO 1.2

Suponha uma fonte discreta sem memória  $\mathbf{S} = \{s_0, s_1, s_2, s_3, s_4\}$  com probabilidades

$$P(\mathbf{S} = s_k) = \{0,4; 0,2; 0,2; 0,1; 0,1\}.$$

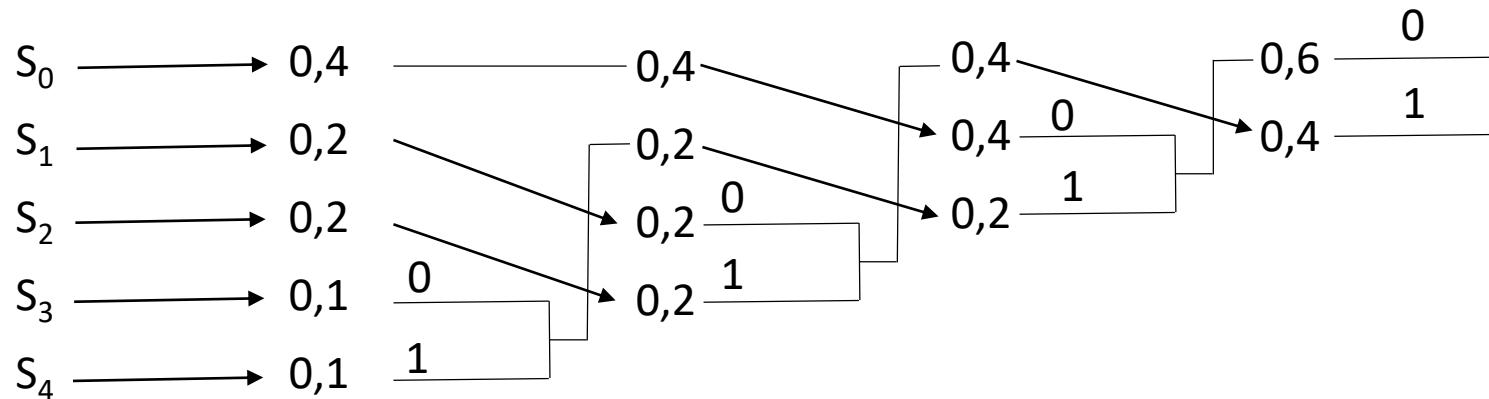
Encontre um código prefixo, usando um conjunto de regras do algoritmo de Huffman, e calcule o comprimento médio das palavras códigos, a entropia, a eficiência de codificação e a árvore de decisão.



## 1.4. COMPACTAÇÃO DE DADOS

### ❖ ALGORITMO DE HUFFMAN

**Símbolos    Probabilidades**



$$\mathbf{S} = \{s_0, s_1, s_2, s_3, s_4\}$$

$$P(\mathbf{S} = s_k) = \{0,4; 0,2; 0,2; 0,1; 0,1\}$$

**CÓDIGO DE HUFFMAN**

$s_0$	00
$s_1$	10
$s_2$	11
$s_3$	010
$s_4$	011



## 1.4. COMPACTAÇÃO DE DADOS

Usando as palavras códigos obtidas e suas probabilidades:

O comprimento médio das palavras códigos obtidos neste exemplo é:

$$\bar{L} = \sum_{k=0}^{K-1} p_k l_k = 0,4(2) + 0,2(2) + 0,2(2) + 0,1(3) + 0,1(3)$$

$$\bar{L} = 2,2 \text{ bits/símbolo}$$

Sendo a entropia determinada por:

$$H(S) = \sum_{k=0}^{K-1} p_k \log_2 \left( \frac{1}{p_k} \right) = 0,4 \log_2 \left( \frac{1}{0,4} \right) + 0,2 \log_2 \left( \frac{1}{0,2} \right) + 0,2 \log_2 \left( \frac{1}{0,2} \right) + \\ + 0,1 \log_2 \left( \frac{1}{0,1} \right) + 0,1 \log_2 \left( \frac{1}{0,1} \right)$$

$$H(S) = 2,12193 \text{ bits/símbolo}$$

$p_k$	$l_k$
$P(s_0) = 0,4$	00
$P(s_1) = 0,2$	10
$P(s_2) = 0,2$	11
$P(s_3) = 0,1$	010
$P(s_4) = 0,1$	011



## 1.4. COMPACTAÇÃO DE DADOS

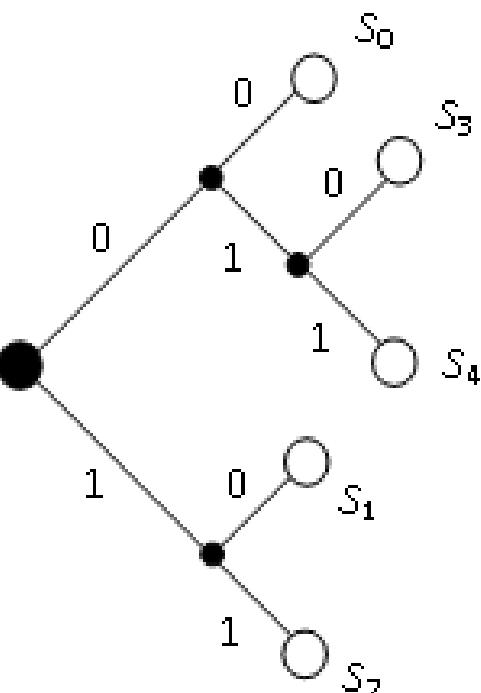
A eficiência obtida nesta codificação é:

$$\bar{L} = 2,2 \text{ bits / símbolo}$$

$$H(S) = 2,12193 \text{ bits / símbolo}$$

$$\eta = \frac{H(S)}{\bar{L}} = \frac{2,12193}{2,2} \Rightarrow \eta = 0,9645$$

A árvore de decisão para o código é:



Conclusão: Este exemplo mostra que o algoritmo de Huffman produz um código prefixo com comprimento médio próximo da entropia e inequivocamente decodificável.



## 1.4. COMPACTAÇÃO DE DADOS

A colocação de símbolos combinados com mesma probabilidade que outro(s), na posição mais alta da coluna, resulta em uma menor variância do comprimento das palavras códigos.

A variância do comprimento das palavras códigos,  $\sigma^2$ , é determinada por

$$\sigma^2 = \sum_{k=0}^{K-1} p_k (l_k - \bar{L})^2$$

Conforme mostrado pelo Primeiro Teorema de Shannon é possível obter um código cujo comprimento médio das palavras códigos seja igual a entropia, ou seja,

$$\bar{L} = H(S)$$

Para esta condição o código prefixo está *casado* com a fonte.



## 1.4. COMPACTAÇÃO DE DADOS

### ❖ CODIFICAÇÃO DE LEMPEL-ZIV

O algoritmo de Huffman necessita do conhecimento prévio da estatística da fonte.

Desvantagem em algumas aplicações com em compactação de texto.

A estatística da fonte para um texto pode mudar de acordo com língua utilizada.

Limitações:

A estatística da fonte para a língua inglesa é diferente da estatística da fonte da língua portuguesa.

Além disso, dentro de uma mesma língua, a estatística da fonte pode variar de acordo com a natureza do texto.

O algoritmo de Lempel-Ziv é adaptativo e não necessita do conhecimento prévio da estatística da fonte.



## 1.4. COMPACTAÇÃO DE DADOS

### ❖ CODIFICAÇÃO DE LEMPEL-ZIV

Para ilustrar este algoritmo considere a seguinte sequência binária:

000101110010100101...

1. Inicialmente os bits 0 e 1 são previamente armazenados nas posições numéricas 0 e 1, conforme apresentado a seguir. Tais posições numéricas compõem o *livro de códigos*.

Posição numérica	0	1	2	3	4	5	6	7	0
Subsequências	0	1							
Blocos codificados	-	-							



## 1.4. COMPACTAÇÃO DE DADOS

### ❖ CODIFICAÇÃO DE LEMPEL-ZIV

2. Em seguida, a subsequência mais curta, ainda não armazenada em nenhuma posição binária, é armazenada na posição 2 pelo codificador. Esta subsequência é **00**. O processo de codificação consiste em transmitir a posição de memória onde se encontra a parte da subsequência **00** já armazenada anteriormente, na forma binária, e o que é *novidade* simplesmente é repetido. Ou seja, da subsequência **00**, o primeiro **0** se encontra na posição numérica **0** (**000** em binário) e o segundo **0** simplesmente é repetido. Veja a seguir.

**00** 0101110010100101...

Posição numérica	0	1	2	3	4	5	6	7	0
Subsequências	0	1	00						
Blocos codificados	-	-	000 0						



## 1.4. COMPACTAÇÃO DE DADOS

### ❖ CODIFICAÇÃO DE LEMPEL-ZIV

Este procedimento é repetido conforme apresentado na sequência a seguir

00**01**01110010100101...

Posição numérica	0	1	2	3	4	5	6	7	0
Subsequências	0	1	00	01					
Blocos codificados	-	-	0000	0001					

0001**011**10010100101...

Posição numérica	0	1	2	3	4	5	6	7	0
Subsequências	0	1	00	01	011				
Blocos codificados	-	-	0000	0001	0111				

0001011**10**010100101...

Posição numérica	0	1	2	3	4	5	6	7	0
Subsequências	0	1	00	01	011	10			
Blocos codificados	-	-	0000	0001	0111	0010			



## 1.4. COMPACTAÇÃO DE DADOS

### ❖ CODIFICAÇÃO DE LEMPEL-ZIV

~~000101110010100101...~~

Posição numérica	0	1	2	3	4	5	6	7	0
Subsequências	0	1	00	01	011	10	010		
Blocos codificados	-	-	0000	0001	0111	0010	0110		

~~000101110010100101...~~

Posição numérica	0	1	2	3	4	5	6	7	0
Subsequências	0	1	00	01	011	10	010	100	
Blocos codificados	-	-	0000	0001	0111	0010	0110	1010	

~~000101110010100101...~~

Posição numérica	-	1	2	3	4	5	6	7	0
Subsequências	-	1	00	01	011	10	010	100	101
Blocos codificados	-	-	0000	0001	0111	0010	0110	1010	1011



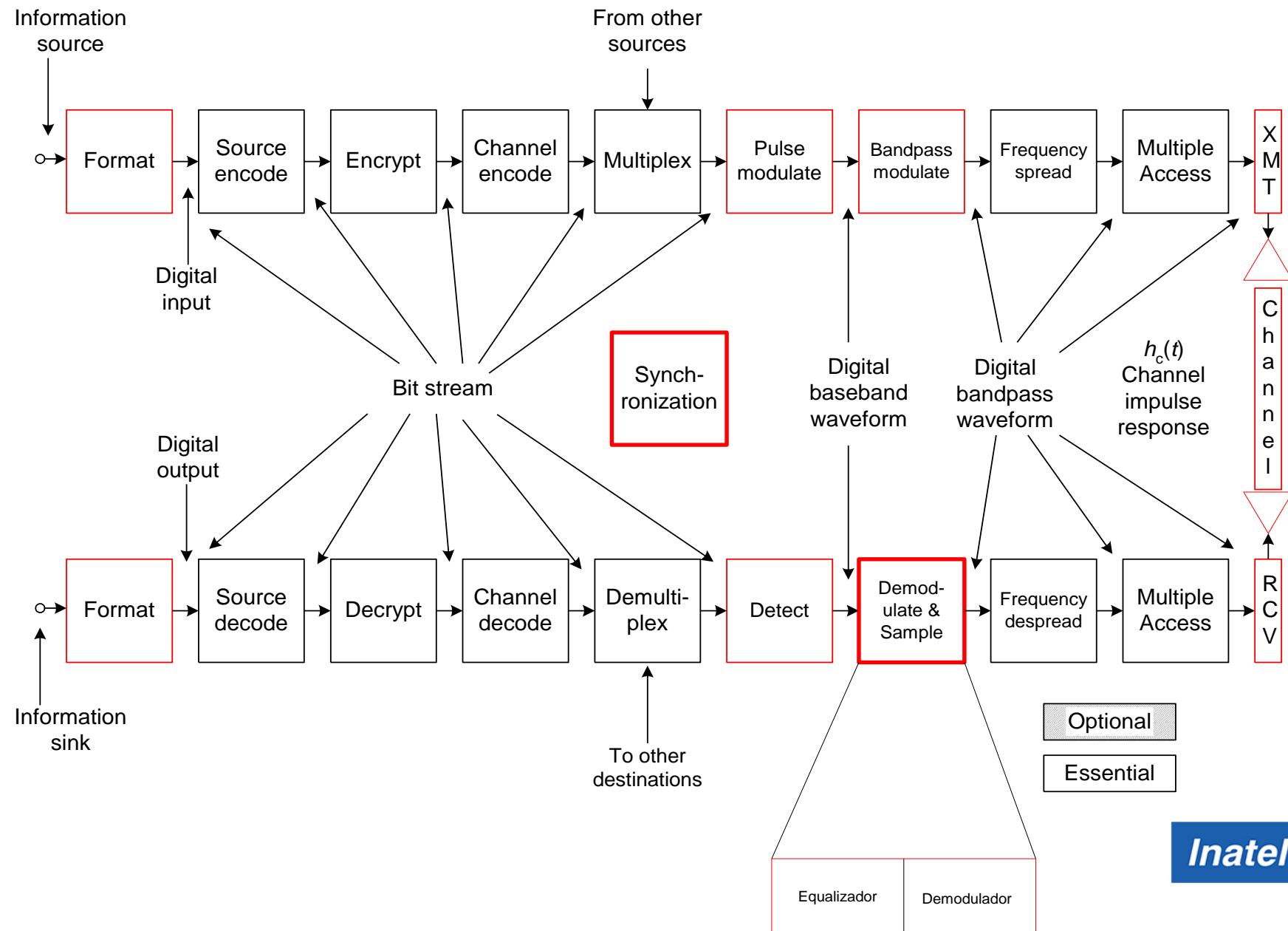
## 1.4. COMPACTAÇÃO DE DADOS

### ❖ CODIFICAÇÃO DE LEMPEL-ZIV

Para a sequência não codificada 00 01 011 10 010 100 101...,

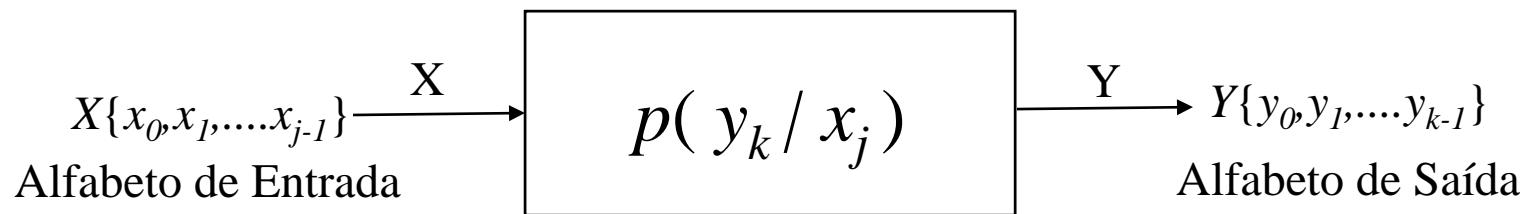
obteve-se a sequência codificada 0000 0001 0111 0010 0110 1010 1011...

Particularmente eficiente para compactação de textos, o algoritmo de Lempel-Ziv pode atingir uma compactação de 55% contra 43% do algoritmo de Huffman, para textos em inglês.





## 1.5. CANAIS DISCRETOS SEM MEMÓRIA



O termo *discreto* significa que os dois alfabetos possuem tamanhos finitos.

O termo *sem memória* significa que o símbolo corrente presente na saída, depende apenas do símbolo corrente presente na entrada e de nenhum outro anterior.

A designação  $p(y_k | x_j)$  representa o conjunto das probabilidades de transição

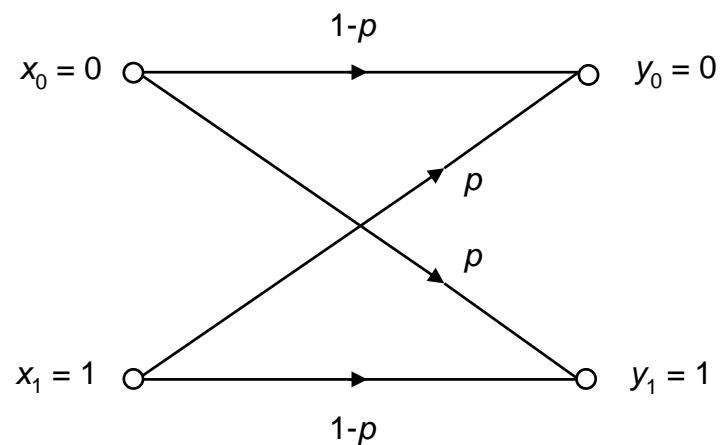
$$p(y_k | x_j) = P(Y = y_k | X = x_j)$$



## 1.5. CANAL DISCRETO SEM MEMÓRIA

### EXEMPLO 1.3

Um canal de grande interesse teórico e prático é o *canal binário simétrico*. Este canal é um caso especial do *canal discreto sem memória* quando  $J = K = 2$ . O alfabeto de entrada do canal possui dois símbolos ( $x_0 = 0, x_1 = 1$ ) e o de saída também dois símbolos ( $y_0 = 0, y_1 = 1$ ). O canal é simétrico porque a probabilidade de se receber **1** quando um **0** é transmitido é a mesma de se receber **0** quando um **1** é transmitido. Esta probabilidade condicional é representada por  $p$ . O diagrama da probabilidade de transição de um canal simétrico binário é mostrado na figura a seguir.



Os termos que compõem a matriz de probabilidade são:

$$p_{10} = P(y = 1 | x = 0)$$

$$p_{01} = P(y = 0 | x = 1)$$

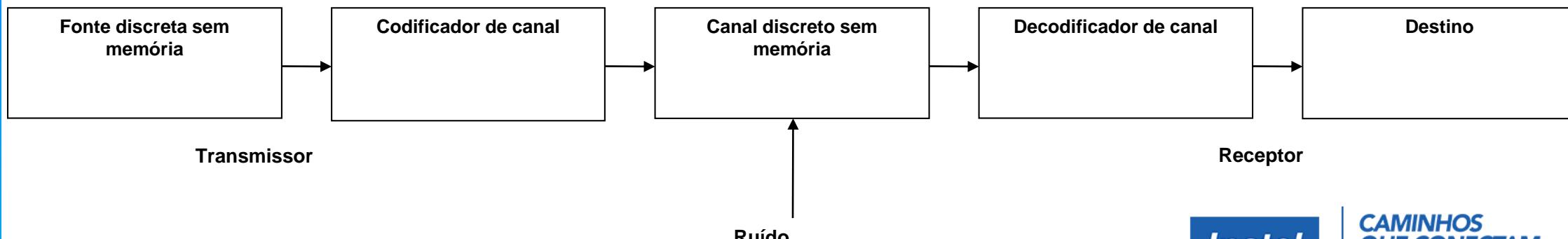
$$p_{10} = p_{01} = p$$



## 1.6. TEOREMA DA CODIFICAÇÃO DE CANAL

A presença de ruído nos canais de comunicações digitais é inevitável e ruído provoca erro. Em canais muito ruidosos a probabilidade de erro pode chegar a  $10^{-1}$  o que significa que em 10 bits transmitidos, 9 são recebidos corretamente. Este nível de confiabilidade é inaceitável para a maior parte das aplicações. A probabilidade de erro aceitável depende da aplicação. Entretanto, uma probabilidade de erro de  $10^{-6}$  ou menor é, frequentemente, um requisito necessário para a maioria das aplicações. Para a obtenção de altos níveis de desempenho, o uso de códigos corretores de erro, ou *codificação de canal* é inevitável.

O objetivo da codificação de canal é aumentar a robustez de um sistema de comunicação digital na presença de ruído e, basicamente, este processo consiste da inserção de redundâncias na sequência binária por um codificador de canal, antes da transmissão. Na recepção, um decodificador de canal verifica a sequência recebida e com o auxílio da redundância introduzida, detecta ou mesmo corrige automaticamente alguns padrões de erro.





## 1.6. TEOREMA DA CODIFICAÇÃO DE CANAL

Quanto maior é a capacidade de correção de erro do código, maior deve ser a quantidade de redundância introduzida e maior é a complexidade de decodificação. Neste ponto uma questão de extrema importância é pode ser colocada da seguinte forma:

*Será que existe um esquema de codificação de canal capaz de tornar a probabilidade de erro de bit arbitrariamente baixa sem que a quantidade de redundância introduzida seja demasiadamente alta?*

Segundo **Shannon**, a resposta é um categórico **sim**, demonstrado por meio do *Teorema da Codificação de Canal*.

Até aqui a variável *tempo* não foi considerada na discussão sobre a capacidade de canal. Admitindo que uma fonte emita símbolos a cada  $T_s$  segundos e que a entropia da fonte,  $H(S)$ , é a medida de bits por símbolo transmitido, então  $H(S)/T_s$  têm a dimensão de *bits por segundo*. Por outro lado,  $C$  é a capacidade de canal em *bits por uso do canal*. Admitindo ainda que o canal possa ser utilizado durante  $T_c$  segundos, então a capacidade de canal dividida pelo tempo de utilização tem a dimensão de *bits por segundo*, que representa a máxima taxa de transferência de informação pelo canal.



## 1.6. TEOREMA DA CODIFICAÇÃO DE CANAL

### SEGUNDO TEOREMA DE SHANNON: TEOREMA DA CODIFICAÇÃO DE CANAL

Suponha uma fonte discreta sem memória com alfabeto  $\mathbf{S}$  e entropia  $H(\mathbf{S})$  bits por símbolo de fonte que produz um símbolo a cada  $T_s$  segundos. Seja um canal discreto sem memória que tem uma capacidade de  $C$  bits por uso do canal e é usado a cada  $T_c$  segundos. Então se

$$\frac{H(\mathbf{S})}{T_s} \leq \frac{C}{T_c}$$

deve existir um esquema de codificação de tal forma que a saída da fonte pode ser transmitida pelo canal com uma taxa de erros arbitrariamente baixa.

O parâmetro  $C/T_c$  é chamado de taxa crítica. Quando a igualdade é obtida diz-se que o sistema está transmitindo na taxa crítica. Inversamente, se

$$\frac{H(\mathbf{S})}{T_s} > \frac{C}{T_c}$$

não é possível transmitir informação pelo canal com taxa arbitrariamente baixa.



## 1.6. TEOREMA DA CODIFICAÇÃO DE CANAL

- O teorema da codificação de canal é considerado o mais importante resultado da teoria da informação.
- Determina a capacidade de canal  $C$  como um *limite fundamental* sobre a taxa na qual uma transmissão pode ser realizada, livre de erros, em um canal discreto sem memória.
- Entretanto duas observações são importantes:
  1. O teorema não mostra como construir bons códigos. Ele deve ser interpretado no sentido de uma *prova de existência*, i.e., desde que a limitação imposta pelo teorema seja satisfeita, então a existência do código é possível.
  2. O teorema não apresenta um resultado preciso para a probabilidade de erro depois da decodificação de canal. Ele indica que a probabilidade de erro tende para zero conforme o comprimento do código aumenta, mas uma vez, desde que a limitação imposta pelo teorema seja satisfeita.



## 1.6. TEOREMA DA CODIFICAÇÃO DE CANAL

### ❖ Aplicação do Teorema da Codificação de Canal em Canais Simétricos Binários

- Considere uma fonte discreta sem memória com símbolos 0's e 1's, com probabilidades iguais, a cada  $T_s$  seg.
- Considere  $H(s) = 1$  bit/símbolo, logo, a fonte emite informação a uma taxa de  $1/T_s$  bits por segundo.
- Considere ainda a existência de um codificador de canal cujo código possui uma taxa decodificação  $r$ .

$$r = \frac{k}{n}$$

$k$  é o número de bits de informação;  $n$  é o número de bits da sequência ou do bloco codificado.

- O codificador de canal produz um símbolo a cada  $T_c$  seg., logo, a taxa de transmissão de símbolos é  $1/T_c$  símbolo(seg). Desta forma, o codificador de canal ocupa um canal simétrico binário a cada  $T_c$  segundos.
- Portanto, a capacidade do canal por unidade de tempo é  $C/T_c$  bits por segundo.
- De acordo com o teorema da codificação de canal, se

$$\frac{H(s)}{T_s} \leq \frac{C}{T_c} \therefore \frac{1}{T_s} \leq \frac{C}{T_c}$$

- A probabilidade de erro pode ser arbitrariamente baixa se usado um esquema de codificação de canal adequado.



## 1.6. TEOREMA DA CODIFICAÇÃO DE CANAL

Como

$$\frac{H(s)}{T_s} \leq \frac{C}{T_c} \therefore \frac{1}{T_s} \leq \frac{C}{T_c} \therefore \frac{T_c}{T_s} \leq C \therefore r = \frac{T_c}{T_s}$$

$$r \leq C$$

para  $r \leq C$ , deve existir um código capaz de permitir uma transmissão com taxa de erro tão baixa quanto se queira.



## 1.6. TEOREMA DA CODIFICAÇÃO DE CANAL

### EXEMPLO 1.4

Seja um canal binário simétrico cuja entrada é equiprovável e a probabilidade de transição do canal (probabilidade de erro) é igual a 0,1. Qual deve ser o maior número de bits de informação para cada bloco de 7 bits de forma ser possível, teoricamente, a obtenção de uma taxa de erro arbitrariamente baixa? Considere que  $C = 1 - H(p)$ .

Do exemplo 1.1 sabe-se que:  $H(p) = -p \log_2 p - (1-p) \log_2 (1-p)$

Como  $p = 0,1$

$$C = 1 - \left( -0,1 \frac{\log 0,1}{\log 2} - 0,9 \frac{\log 0,9}{\log 2} \right)$$

$$C = 0,531$$

Conforme apresentado

$$r = \frac{k}{n} \leq C$$

$$k \leq nC$$

Como  $n = 7$

$$k = \lfloor nC \rfloor = \lfloor 7 \times 0,531 \rfloor = \lfloor 3,72 \rfloor$$

$$k = 3$$



## 1.7. TEOREMA DA CAPACIDADE DE INFORMAÇÃO OU TEOREMA DA CAPACIDADE DE CANAL

### TERCEIRO TEOREMA DE SHANNON: TEOREMA DA CAPACIDADE DE CANAL

A capacidade de informação de um canal contínuo com largura de faixa de  $B$  Hertz, perturbado por ruído Gaussiano branco aditivo de densidade espectral  $N_0/2$ , é dada por

$$C = B \log_2 \left( 1 + \frac{P}{N_0 B} \right)$$

dada em bits por segundo, onde  $P$  é a potência média do sinal transmitido.

- O Teorema da Capacidade de Informação é um dos mais notáveis resultados da Teoria da Informação onde, em uma única expressão, é destacada a interação entre a largura de faixa do canal e a relação sinal/ruído.
- A capacidade do canal varia linearmente com a largura de faixa e logaritmicamente com a relação sinal ruído. Desta forma, *é mais fácil aumentar a capacidade do canal aumentando a sua largura de faixa do que aumentando a relação sinal ruído*.
- O Teorema da Capacidade de Canal define um *limite fundamental* para a taxa de transmissão livre de erros para um canal gaussiano com uma dada relação sinal/ruído e largura de faixa limitada.



## 1.7. TEOREMA DA CAPACIDADE DE INFORMAÇÃO OU TEOREMA DA CAPACIDADE DE CANAL

A capacidade do canal também pode ser dada por

$$C = B \log_2 \left( 1 + \frac{S}{N} \right) \quad \text{bits por segundo}$$

### EXEMPLO 1.5

Considere um canal AWGN limitado em faixa ( $B = 3,4$  kHz) e uma potência de recepção (saída do canal) igual a 1pW. Determine a capacidade do canal, considerando que o mesmo está submetido a uma temperatura equivalente de ruído igual a 290 K. Determine também a relação sinal-ruído em dB.

$$C = B \log_2 \left( 1 + \frac{P}{N_0 B} \right) = B \log_2 \left( 1 + \frac{P}{kTB} \right) = B \log_2 \left( 1 + \frac{S}{N} \right)$$

$$\frac{S}{N} = \frac{P}{kTB} = \frac{1 \times 10^{-12}}{1,38 \times 10^{-23} \times 290 \times 3,4 \times 10^3} = 73,49 \times 10^3$$

$$C = 3,4 \times 10^3 \frac{\log(1 + 73,49 \times 10^3)}{\log 2}$$

$$C = 54,96 \text{ kbit/s}$$

$$\begin{aligned} \left( \frac{S}{N} \right)_{dB} &= 10 \log 73,49 \times 10^3 \\ \left( \frac{S}{N} \right)_{dB} &= 48,66 \text{ dB} \end{aligned}$$



## 1.7. TEOREMA DA CAPACIDADE DE INFORMAÇÃO OU TEOREMA DA CAPACIDADE DE CANAL

CONCLUSÃO:

Deve haver um esquema de codificação de canal, suficientemente complexo, que permita a transmissão a uma taxa de  $\approx 55$  kbit/s em um canal com 3,4 kHz de largura de faixa para uma relação sinal/ruído de  $\approx 49$  dB, com uma taxa de erros de bit arbitrariamente baixa.

## 1.8. IMPLICAÇÕES DO TEOREMA DA CAPACIDADE DE CANAL

Para analisar as implicações do Teorema da Capacidade de informação, admita a existência de um *sistema ideal* que transmite a uma taxa  $R_b$  igual à capacidade de canal  $C$ . Desta forma, a potência média do sinal pode ser escrita como

$$P = E_b C$$

onde  $E_b$  é a energia transmitida por bit. Logo, a expressão

$$C = B \log_2 \left( 1 + \frac{P}{N_0 B} \right)$$
 pode ser reescrita como  $\frac{C}{B} = \log_2 \left( 1 + \frac{E_b}{N_0} \frac{C}{B} \right)$



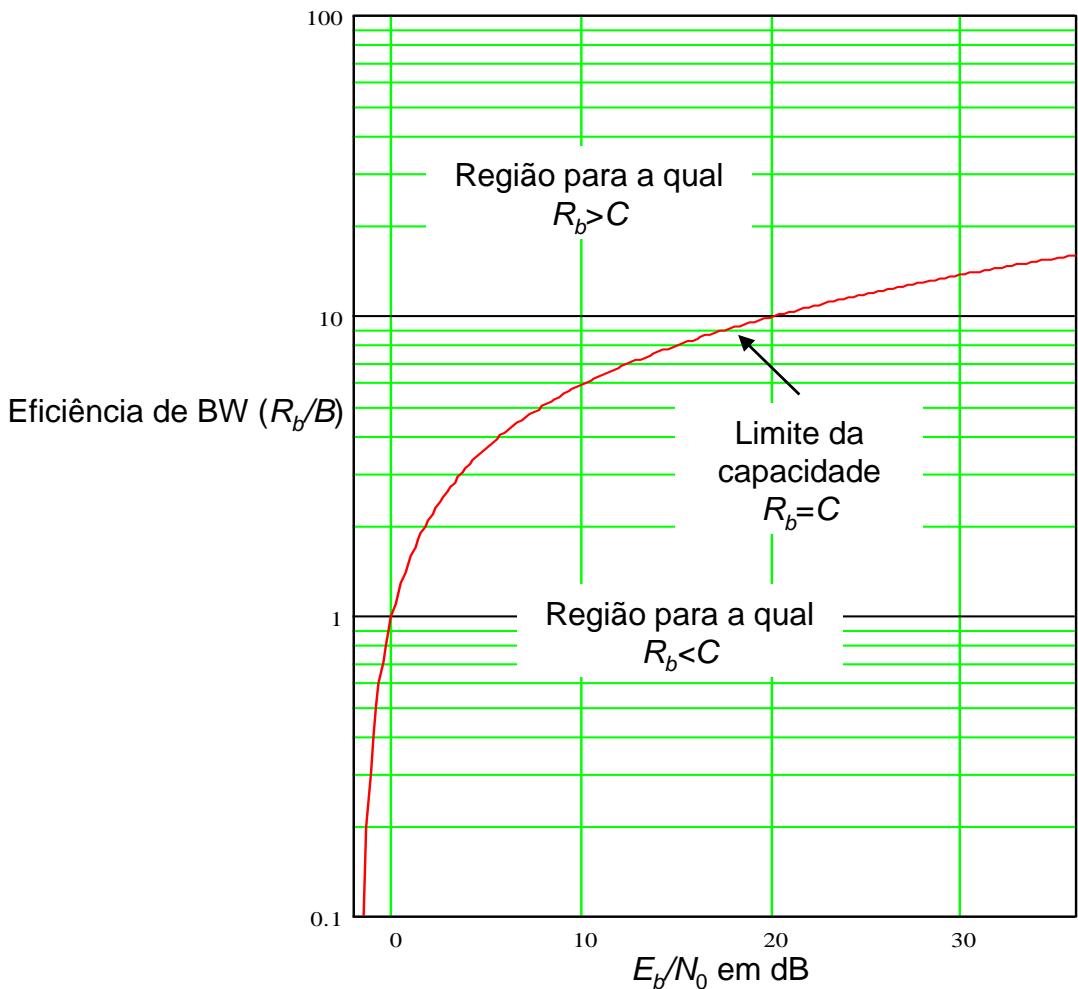
## 1.8. IMPLICAÇÕES DO TEOREMA DA CAPACIDADE DE CANAL

Equivalentemente, a relação entre a energia por bit e a densidade espectral de ruído  $E_b/N_0$  em termos de  $C/B$  para o sistema ideal é

$$\frac{E_b}{N_0} = \frac{2^{C/B} - 1}{C/B}$$

Com a expressão é possível traçar a curva que estabelece a relação entre a *eficiência de largura de faixa*  $R_b/B$  e a relação  $E_b/N_0$ , que é chamado de diagrama da eficiência de largura de faixa. A forma genérica deste diagrama é mostrada na Figura, onde a curva intitulada "*limite da capacidade*" corresponde ao sistema ideal no qual  $R_b = C$ .

O limite é representado por uma fronteira entre a região em que é possível uma transmissão livre de erros ( $R_b < C$ ) e a região em que isso não é possível ( $R_b > C$ ) em um plano da *eficiência de largura de faixa* em função da relação entre a *energia de bit* e a *densidade espectral de ruído*, conforme apresentado a seguir.



# CAPÍTULO 2



## 2.1 INTRODUÇÃO À CODIFICAÇÃO DE CANAL

- A codificação de canal é um processo em que redundâncias são introduzidas antes da transmissão, com o objetivo de permitir que, no receptor, a semelhança entre o sinal que foi transmitido e o sinal que foi reproduzido seja a máxima possível.
- É um processo que permite a redução da  $P_b$  a valores tão baixos quanto possíveis.
- O que se pode obter como *máxima semelhança* ou  *$P_b$  tão baixa quanto possível* com a codificação de canal?
- Essa questão foi parcialmente respondida no capítulo anterior através do Teorema da Codificação de Canal.

*Desde que a taxa de transmissão seja menor do que a capacidade do canal, então existe um esquema de codificação capaz de permitir a obtenção de taxas de erro de bit arbitrariamente baixas.*

- Porém, este teorema não indica o esquema de codificação nem o seu grau de complexidade.
- Portanto, encontrar um esquema de codificação, é necessário conhecimento dos fundamentos de codificação para controle de erro.
- O objetivo deste capítulo é apresentar os fundamentos dos Códigos de Bloco Lineares e seus desempenhos.



## 2.2 CÓDIGOS DE BLOCO: DEFINIÇÕES INICIAIS

- Códigos de blocos se caracterizam pela codificação ser feita sobre blocos de bits ou de símbolos.
- Isso quer dizer que um feixe de bits ou símbolos é segmentado em blocos de  $k$  bits ou símbolos, a partir dos quais são geradas palavras códigos com  $n$  bits ou símbolos.
- Notação: Código de bloco é  $(n, k)$ .
- Por conveniência, a partir deste ponto a notação  $(n, k)$  estará associada à quantidade de bits.
- Quando a notação  $(n, k)$  for usada para representar símbolos, isso será definido explicitamente.
- Se  $k$  bits estão contidos em um bloco de  $n$  bits, então a quantidade de bits de redundância introduzidos no processo de codificação é  $(n - k)$ .



## 2.2 CÓDIGOS DE BLOCO: DEFINIÇÕES INICIAIS

### 2.2.1. TAXA DE CODIFICAÇÃO

- É relação entre o número de bits de informação e o número de bits da palavra código.

$$R_c = \frac{k}{n}$$

- $R_c$  é uma indicação de quantos bits de informação são transmitidos por palavra código.
- Uma vez que  $0 < k \leq n$ , então  $0 < R_c \leq 1$ .
- Entretanto, para que um código produza algum benefício, é necessário que  $k < n$ , ou  $(n - k) > 0$ . Consequentemente,  $0 < R_c < 1$ .

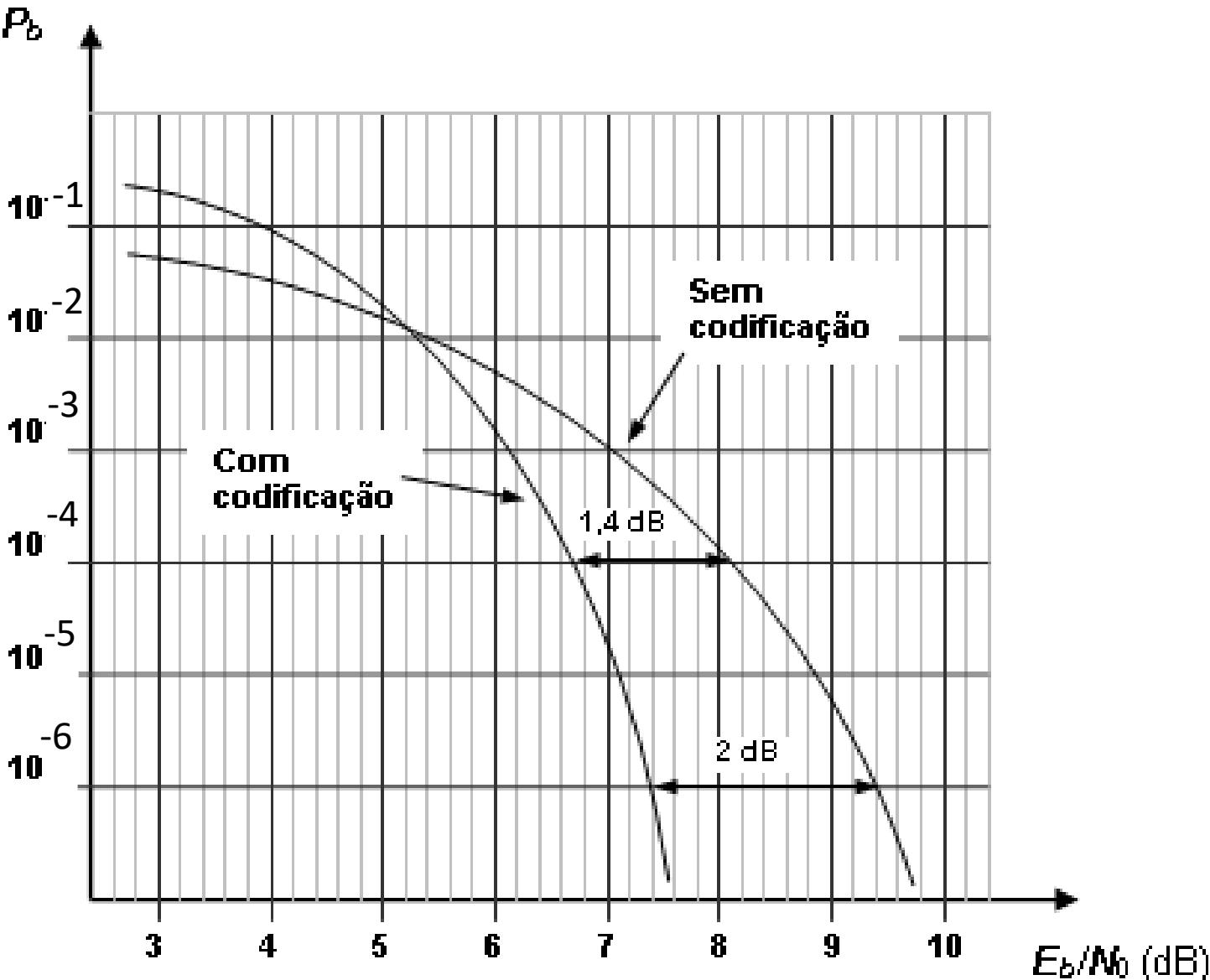


## 2.2 CÓDIGOS DE BLOCO: DEFINIÇÕES INICIAIS

### 2.2.2. GANHO DE CODIFICAÇÃO

Curvas típicas de  $P_b$  versus  $E_b/N_0$  para um sinal com codificação e sem codificação.

$$\left(\frac{E_b}{N_0}\right)_c = R \left(\frac{E_b}{N_0}\right)_{nc}$$





## 2.2 CÓDIGOS DE BLOCO: DEFINIÇÕES INICIAIS

### 2.2.2. GANHO DE CODIFICAÇÃO

De acordo com o comportamento das curvas apresentadas pode-se concluir que:

- 1) Para baixos valores de  $E_b/N_0$  a codificação não apresenta nenhum benefício, ou seja, o ganho de codificação pode ser nulo, para o valor de  $E_b/N_0$  determinado pelo cruzamento das curvas, ou negativo para valores menores.
- 2) Para diferentes valores de  $P_b$  obtém-se diferentes ganhos de codificação. Por exemplo, para  $P_b = 10^{-4}$  o ganho de codificação é igual a 1,4 dB, enquanto para  $P_b = 10^{-6}$  o ganho sobe para 2 dB. Isso demonstra a dependência do ganho de codificação com  $P_b$ .
- 3) A codificação permite obter redução de  $P_b$  com a mesma energia ( $E_b/N_0$  constante) em relação ao sinal sem codificação. Por exemplo, para  $E_b/N_0 = 7$  dB, a  $P_b$  cai de  $10^{-3}$  para um pouco mais que  $10^{-5}$ .

Considerando-se a expansão de largura de faixa provocada pela codificação, pode-se concluir ainda:

- 1) A diminuição de  $P_b$  e/ou  $E_b/N_0$  decorrente da codificação produz uma expansão na largura de faixa.

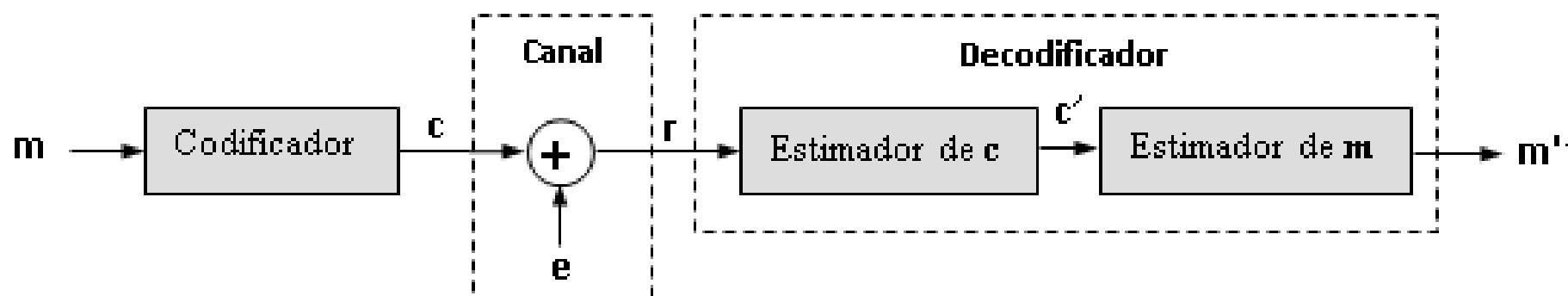


## 2.2 CÓDIGOS DE BLOCO: DEFINIÇÕES INICIAIS

### 2.2.3. VETOR CÓDIGO, VETOR ERRO, VETOR RECEBIDO E VETOR DECODIFICADO

- A codificação por bloco transforma um segmento da mensagem,  $\mathbf{m}$ , com  $k$  bits em uma palavra código ou vetor código,  $\mathbf{c}$ , com  $n$  bits.
- O vetor código é transmitido e pode sofrer alterações devido às degradações impostas pelo meio de transmissão.
- As alterações sofridas pelo vetor código podem ser representadas por meio de um vetor erro,  $\mathbf{e}$ .
- Um vetor código,  $\mathbf{c}$ , somado com um vetor erro,  $\mathbf{e}$ , resulta em um vetor recebido,  $\mathbf{r}$ .

$$\mathbf{r} = \mathbf{c} \oplus \mathbf{e}$$





## 2.2 CÓDIGOS DE BLOCO: DEFINIÇÕES INICIAIS

### 2.2.3. VETOR CÓDIGO, VETOR ERRO, VETOR RECEBIDO E VETOR DECODIFICADO

#### Exemplo 2.1

Seja um código de repetição (5, 1) aplicado ao vetor mensagem  $\mathbf{m} = 1$ . Admitindo que o canal introduza um vetor erro,  $\mathbf{e} = 01010$ , ao vetor código, pede-se determinar todas as transformações vetoriais desde a codificação da mensagem até a obtenção da mensagem estimada na saída do decodificador.

Evidentemente para o código de repetição (5, 1) só existem duas palavras códigos possíveis: 00000 e 11111. A palavra código correspondente à mensagem  $\mathbf{m} = 1$  é

$$\mathbf{c} = 11111$$

O vetor recebido é

$$\mathbf{r} = \mathbf{c} \oplus \mathbf{e} = 11111 \oplus 01010$$

$$\mathbf{r} = 10101$$

Adição
$0 \oplus 0 = 0$
$0 \oplus 1 = 1$
$1 \oplus 0 = 1$
$1 \oplus 1 = 0$

Como um código de repetição pode ser decodificado por lógica majoritária, então a melhor estimativa para o vetor código a partir do vetor recebido é

$$\mathbf{c}' = 11111,$$

que resulta na mensagem estimada

$$\mathbf{m}' = 1.$$



## 2.2 CÓDIGOS DE BLOCO: DEFINIÇÕES INICIAIS

### 2.2.4. PESO DE HAMMING E DISTÂNCIA DE HAMMING

- **Peso de Hamming** de um vetor  $\mathbf{v}$ : Notação é  $w(\mathbf{v})$ ,
- É definido como sendo o número de elementos não zero em  $\mathbf{v}$ .
- Para um vetor binário, o peso de Hamming é igual ao número de dígitos “1” contidos em  $\mathbf{v}$ .

#### EXEMPLO 2.2

Determinar o peso de Hamming do vetor  $\mathbf{v} = 10110$ .

$$w(\mathbf{v}) = 3.$$



## 2.2 CÓDIGOS DE BLOCO: DEFINIÇÕES INICIAIS

### 2.2.4. PESO DE HAMMING E DISTÂNCIA DE HAMMING

- A **distância de Hamming** entre dois vetores códigos  $\mathbf{v}$  e  $\mathbf{x}$ : Notação é  $d(\mathbf{v}, \mathbf{x})$ ,
- É definida como sendo o número de posições em que os dígitos dos dois vetores que são diferentes entre si. Para o caso binário:

$$d(\mathbf{v}, \mathbf{x}) = w(\mathbf{v} \oplus \mathbf{x})$$

#### EXEMPLO 2.3

Determinar a distância de Hamming entre o vetor  $\mathbf{v} = 10110$  e  $\mathbf{x} = 10101$ .

$$d(\mathbf{v}, \mathbf{x}) = w(\mathbf{v} \oplus \mathbf{x}) = w(10110 \oplus 10101) = w(00011)$$

$$d(\mathbf{v}, \mathbf{x}) = 2$$

Adição
$0 \oplus 0 = 0$
$0 \oplus 1 = 1$
$1 \oplus 0 = 1$
$1 \oplus 1 = 0$



## 2.2 CÓDIGOS DE BLOCO: DEFINIÇÕES INICIAIS

### 2.2.5. ESPAÇO VETORIAL E SUBESPAÇO VETORIAL

Considere um conjunto  $V$ , constituídos por  $K$  vetores  $\mathbf{v}_0, \mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{K-1}$ , formado por  $n$  elementos de  $\{0, 1\}$ . Admita que sobre este conjunto sejam definidas duas operações, cujas regras são apresentadas na Tabela. A adição, representada por  $\oplus$ , definida entre os elementos de  $V$  e a multiplicação, representada por  $\bullet$ , entre um elemento de  $\{0, 1\}$  e qualquer vetor de  $V$

Adição	Multiplicação
$0 \oplus 0 = 0$	$0 \bullet 0 = 0$
$0 \oplus 1 = 1$	$0 \bullet 1 = 0$
$1 \oplus 0 = 1$	$1 \bullet 0 = 0$
$1 \oplus 1 = 0$	$1 \bullet 1 = 1$

O conjunto  $V$  é definido como um *espaço vetorial* sobre  $\{0, 1\}$  se as seguintes condições são satisfeitas:



## 2.2 CÓDIGOS DE BLOCO: DEFINIÇÕES INICIAIS

### 2.2.5. ESPAÇO VETORIAL E SUBESPAÇO VETORIAL

- 1) A adição de quaisquer dois vetores de  $V$  resulta em outro vetor em  $V$  (propriedade do fechamento).
- 2) O produto escalar de um elemento de  $\{0, 1\}$  e qualquer vetor de  $V$  resulta em outro vetor em  $V$ .
- 3) A lei distributiva é satisfeita, ou seja, se  $a_1$  e  $a_2$  são escalares de  $\{0, 1\}$  e  $\mathbf{v}_1$  e  $\mathbf{v}_2$  são vetores de  $V$ , então

$$a_1 \bullet (\mathbf{v}_1 \oplus \mathbf{v}_2) = (a_1 \bullet \mathbf{v}_1) \oplus (a_1 \bullet \mathbf{v}_2)$$

$$(a_1 \oplus a_2) \bullet \mathbf{v}_1 = (a_1 \bullet \mathbf{v}_1) \oplus (a_2 \bullet \mathbf{v}_1)$$

- 4) A lei associativa é satisfeita, ou seja, se  $a_1$  e  $a_2$  são escalares de  $\{0, 1\}$  e  $\mathbf{v}_1$  é um vetor de  $V$ , então

$$(a_1 \bullet a_2) \bullet \mathbf{v}_1 = a_1 \bullet (a_2 \bullet \mathbf{v}_1)$$



## 2.2 CÓDIGOS DE BLOCO: DEFINIÇÕES INICIAIS

### 2.2.5. ESPAÇO VETORIAL E SUBESPAÇO VETORIAL

#### EXEMPLO 2.4

Encontrar o espaço vetorial  $V$  composto pelo maior número possível de vetores com cinco elementos de  $\{0, 1\}$ .

O espaço vetorial  $V_5$  é o conjunto de todos os vetores binários com cinco elementos ( $n = 5$ ) apresentados na Tabela.

$V_5$							
00000	00100	01000	01100	10000	10100	11000	11100
00001	00101	01001	01101	10001	10101	11001	11101
00010	00110	01010	01110	10010	10110	11010	11110
00011	00111	01011	01111	10011	10111	11011	11111



## 2.2 CÓDIGOS DE BLOCO: DEFINIÇÕES INICIAIS

### 2.2.5. ESPAÇO VETORIAL E SUBESPAÇO VETORIAL

Um subconjunto  $S$  de um espaço  $V$  é chamado de *subespaço vetorial de  $V$*  se verifica as quatro condições definidas.

#### EXEMPLO 2.5

A partir das propriedades do subespaço vetorial identificar dois subespaços vetoriais de  $V_5$ , apresentado na Tabela a seguir, que contenha:

- 1) 4 vetores
- 2) 8 vetores

Tabela - Subespaços de  $V_5$  com 4 e 8 vetores

$S$ com 4 vetores	$S$ com 8 vetores	
00000	00000	01100
01011	01011	00111
10010	10010	11110
11001	11001	10101

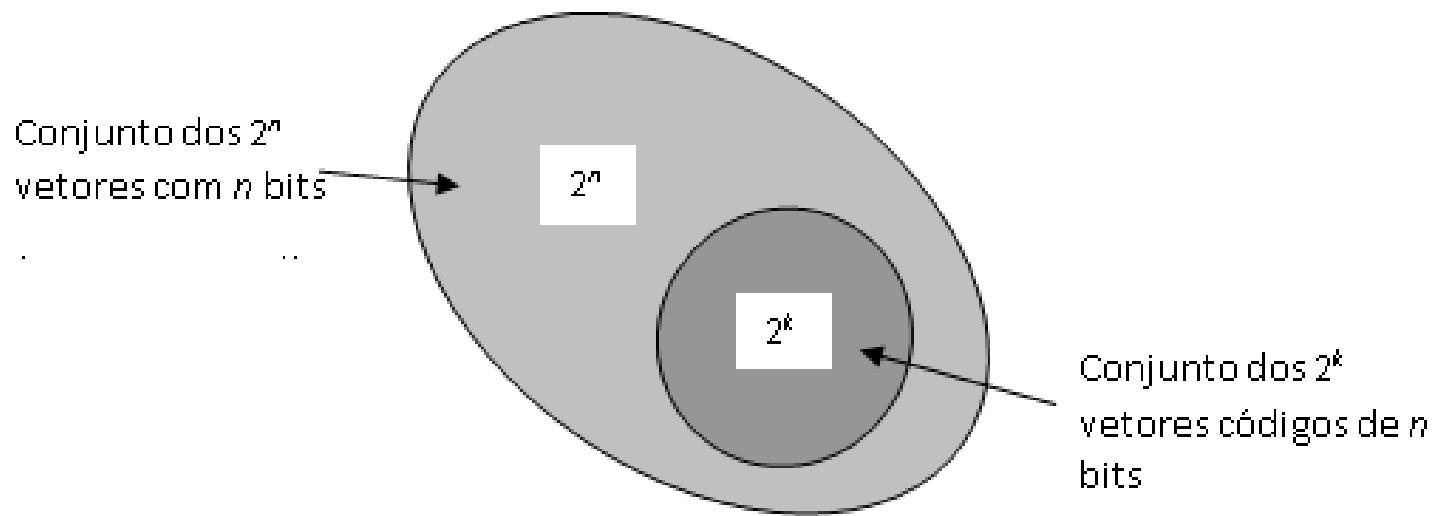
Os subespaços encontrados acima não são únicos. É possível encontrar, em  $V_5$ , outros subespaços contendo 4 e 8 vetores.





### 2.3. CÓDIGOS DE BLOCO LINEARES

- Um *código de bloco linear binário* é um subespaço vetorial com  $2^k$  vetores do espaço vetorial constituído de todos os  $2^n$  vetores com  $n$  elementos de  $\{0, 1\}$ .

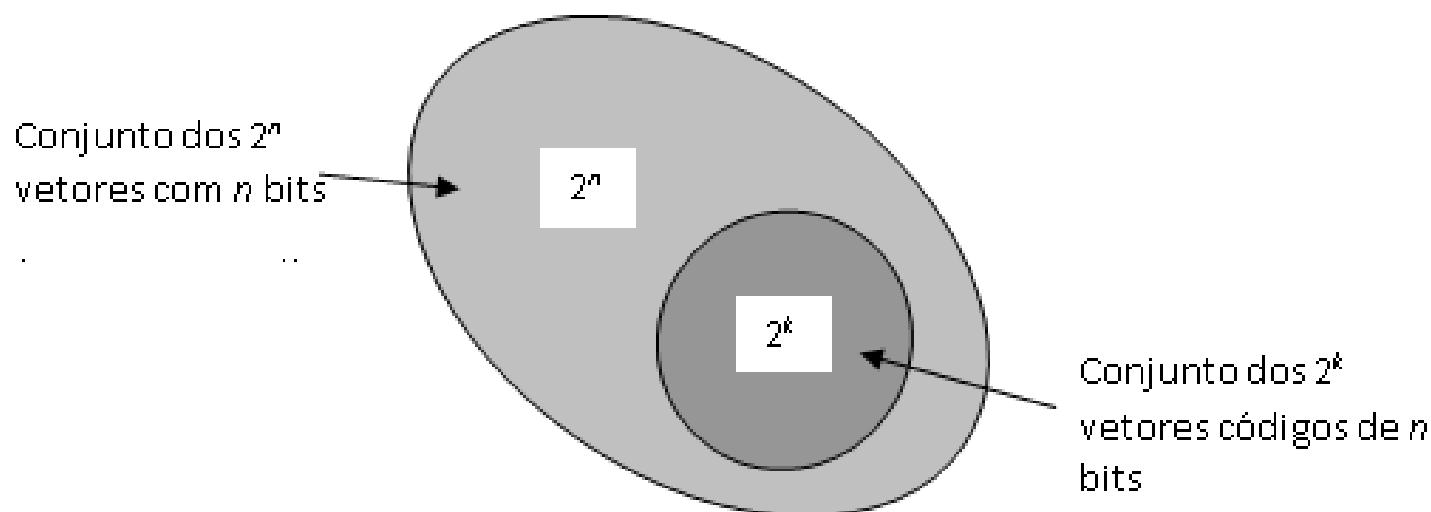


- 1) A soma de duas palavras códigos quaisquer resulta em outra palavra código.
- 2) O vetor nulo ou vetor todo zero é também uma palavra código.



### 2.3. CÓDIGOS DE BLOCO LINEARES

- O subespaço vetorial constitui o conjunto dos vetores códigos ou vetores válidos.
- Qualquer vetor de  $n$  bits que não pertença ao subespaço vetorial está no espaço vetorial, sendo um *vetor não válido*.
- Uma estratégia de **deteção de erros** consiste em verificar se o vetor recebido é um vetor válido ou não válido.
- Se o vetor recebido é um vetor não válido, uma estratégia de **correção de erros** consiste na identificação de qual é o vetor válido que apresenta a menor distância de Hamming em relação ao vetor recebido e elegê-lo como sendo o vetor transmitido.





### 2.3. CÓDIGOS DE BLOCO LINEARES

- Nos códigos de bloco lineares onde o valor de  $k$  é baixo esta tarefa é simples.
- Entretanto, quando  $k$  apresenta valores relativamente altos, esta tarefa pode tornar-se impraticável.

#### EXEMPLO 2.6

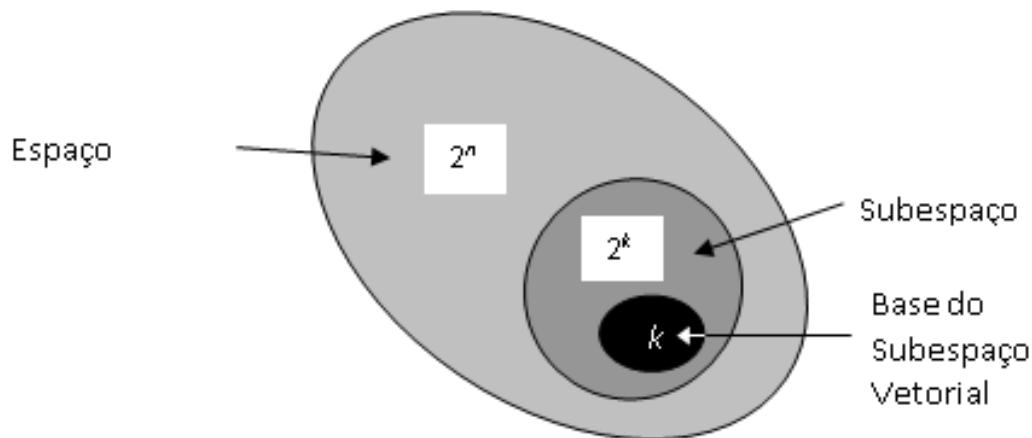
Admitindo a existência de um código de bloco linear (255, 130), pede-se:

- Determinar a quantidade de vetores códigos binários existentes neste código.
  - Determinar a quantidade de vetores não possíveis.
  - Descrever uma estratégia de correção de erros, admitindo que o vetor recebido é um vetor não válido.
- A quantidade de vetores códigos é:  $2^k = 2^{130} \cong 1,36 \times 10^{39}$  vetores válidos.
  - Quando o vetor é não válido, então ele é um dos  $2^n - 2^k$  não válidos:  $2^{255} - 2^{130} \cong 2^{255} \cong 5,7896 \times 10^{76}$  vetores não válidos.
  - Uma estratégia de correção de erros é identificar entre os  $1,36 \times 10^{39}$  vetores válidos qual é o vetor que apresenta a menor distância de Hamming em relação ao vetor não válido recebido!



### 2.3. CÓDIGOS DE BLOCO LINEARES

- Nota-se que um subespaço vetorial é um conjunto de vetores *Linearmente Dependentes* (LD) devido à propriedade do fechamento, i.e., qualquer vetor pode ser obtido pela soma de outros dois vetores do subespaço.
- Uma vez que um subespaço vetorial binário contém  $2^k$  vetores, então deve existir um ou mais subconjuntos com  $k$  vetores ditos *Linearmente Independentes* (LI) cujas combinações lineares produzem todos os outros vetores do subespaço.
- Esses  $k$  vetores linearmente independentes são chamados de *base do subespaço*. Os conceitos de espaço vetorial, subespaço vetorial e base do subespaço estão apresentados, em termos de conjunto, na Figura.





### 2.3. CÓDIGOS DE BLOCO LINEARES

#### EXEMPLO 2.7

A partir do subespaço vetorial com 8 vetores, apresentado na Tabela abaixo, identificar uma base capaz de gerar todos os outros vetores deste subespaço, através de combinações lineares dos vetores desta base.

S com 8 vetores	
00000	01100
01011	00111
10010	11110
11001	10101

Como o objetivo é formar uma base para a geração de um subespaço com  $2^k = 8$  vetores LD o número necessário de vetores na base será  $k = 3$  vetores L.I.. Escolhendo-se arbitrariamente 3 vetores LI entre os 8 vetores da Tabela, pode-se obter:

$$\mathbf{b}_0 = 01011$$

$$\mathbf{b}_1 = 10010$$

$$\mathbf{b}_2 = 01100$$



### 2.3. CÓDIGOS DE BLOCO LINEARES

**Prova:** Com a combinação linear dos vetores encontrados é possível encontrar todos os outros vetores do subespaço, representados a seguir pelos vetores  $\mathbf{v}_j$ , obtidos a partir da operação

$$\mathbf{v}_j = u_0 \bullet \mathbf{b}_0 + u_1 \bullet \mathbf{b}_1 + \dots + u_{k-1} \bullet \mathbf{b}_{k-1}$$

onde  $u_i$  é um elemento de  $\{0, 1\}$  para  $i = 0, 1, 2, \dots, (k - 1)$  e  $j = 0, 1, \dots, (2^k - 1)$ .

$v_0 = 0 \bullet b_0 + 0 \bullet b_1 + 0 \bullet b_2 = 00000$	
$v_1 = 1 \bullet b_0 + 0 \bullet b_1 + 0 \bullet b_2 = 01011$	← Vetor da base
$v_2 = 0 \bullet b_0 + 1 \bullet b_1 + 0 \bullet b_2 = 10010$	← Vetor da base
$v_4 = 0 \bullet b_0 + 0 \bullet b_1 + 1 \bullet b_2 = 01100$	← Vetor da base
$v_3 = 1 \bullet b_0 + 1 \bullet b_1 + 0 \bullet b_2 = 11001$	
$v_5 = 1 \bullet b_0 + 0 \bullet b_1 + 1 \bullet b_2 = 00111$	
$v_6 = 0 \bullet b_0 + 1 \bullet b_1 + 1 \bullet b_2 = 11110$	
$v_7 = 1 \bullet b_0 + 1 \bullet b_1 + 1 \bullet b_2 = 10101$	



## 2.3. CÓDIGOS DE BLOCO LINEARES

### 2.3.1. MATRIZ GERADORA

- Uma matriz geradora,  $\mathbf{G}$ , é aquela que permite obter os vetores códigos,  $\mathbf{c}_j$ , correspondentes às mensagens,  $\mathbf{m}_i$ , a partir do produto interno determinado por

$$\mathbf{c}_j = \mathbf{m}_j \cdot \mathbf{G}$$

- A matriz  $\mathbf{G}$  é uma consequência direta de uma base do subespaço vetorial.
- Elá é uma matriz de dimensões  $k \times n$  que consiste do arranjo formado pelos vetores L.I., ou *vetores geradores*, que compõem uma base do subespaço.

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_{k-1} \end{bmatrix} = \begin{bmatrix} g_{00} & g_{01} & g_{02} & \cdots & g_{0, n-1} \\ g_{10} & g_{11} & g_{12} & \cdots & g_{1, n-1} \\ \vdots & \vdots & \vdots & & \vdots \\ g_{k-1, 0} & g_{k-1, 1} & g_{k-1, 2} & \cdots & g_{k-1, n-1} \end{bmatrix}$$



## 2.3. CÓDIGOS DE BLOCO LINEARES

### 2.3.1. MATRIZ GERADORA

- Onde  $\mathbf{g}_0, \mathbf{g}_1, \dots, \mathbf{g}_{k-1}$ , são os vetores geradores.
- Por conveniência da notação os vetores códigos  $\mathbf{c}_j$  e  $\mathbf{m}_j$  serão representados simplesmente por  $\mathbf{c}$  e  $\mathbf{m}$ , respectivamente.

$$\mathbf{c} = \mathbf{m} \cdot \mathbf{G} = (m_0, m_1, \dots, m_{k-1}) \cdot \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_{k-1} \end{bmatrix} = m_0\mathbf{g}_0 + m_1\mathbf{g}_1 + \dots + m_{k-1}\mathbf{g}_{k-1}.$$

- Observa-se a semelhança com a combinação linear dos elementos dos vetores mensagem com as linhas da matriz geradora que produzem vetores códigos que estão associados inequivocamente aos vetores mensagens que os produziu.



## 2.3. CÓDIGOS DE BLOCO LINEARES

### 2.3.1. MATRIZ GERADORA

#### EXEMPLO 2.8

A partir da base do subespaço vetorial apresentado no Exemplo 2.7, pede-se:

- Construir uma matriz geradora.
- A partir da matriz geradora, construir uma tabela com os vetores mensagens e seus respectivos vetores códigos
- A obtenção de uma matriz geradora a partir do Exemplo 2.7 é direta, pois ela nada mais é do que a base de um subespaço vetorial, logo, utilizando os mesmos vetores  $\mathbf{b}_0 = 01011$ ,  $\mathbf{b}_1 = 10010$  e  $\mathbf{b}_2 = 01100$  para  $\mathbf{g}_0$ ,  $\mathbf{g}_1$  e  $\mathbf{g}_2$ , respectivamente, obtém-se

$$\mathbf{G} = \begin{bmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix}$$



## 2.3. CÓDIGOS DE BLOCO LINEARES

### 2.3.1. MATRIZ GERADORA

b) Como a matriz geradora possui três linhas, os vetores resultantes de todas as combinações lineares serão  $2^k = 2^3 = 8$ , obtidos a partir de todos os vetores mensagens possíveis contendo de três bits. Logo,  $\mathbf{G}$  é a matriz geradora de um código (5, 3).

$m$	$c = m \cdot G$	$c$
000	$c_0 = 0(01011) + 0(10010) + 0(01100)$	00000
100	$c_1 = 1(01011) + 0(10010) + 0(01100)$	01011
010	$c_2 = 0(01011) + 1(10010) + 0(01100)$	10010
110	$c_3 = 1(01011) + 1(10010) + 0(01100)$	11001
001	$c_4 = 0(01011) + 0(10010) + 1(01100)$	01100
101	$c_5 = 1(01011) + 0(10010) + 1(01100)$	00111
011	$c_6 = 0(01011) + 1(10010) + 1(01100)$	11110
111	$c_7 = 1(01011) + 1(10010) + 1(01100)$	10101

$$\begin{array}{ll}
 \left. \begin{array}{l} 1(01011) \\ 0(01100) \\ 0(10010) \end{array} \right\} & \begin{array}{l} (01011) \\ (00000) \\ (00000) \end{array} \\
 \hline
 & 01011
 \end{array}$$
  

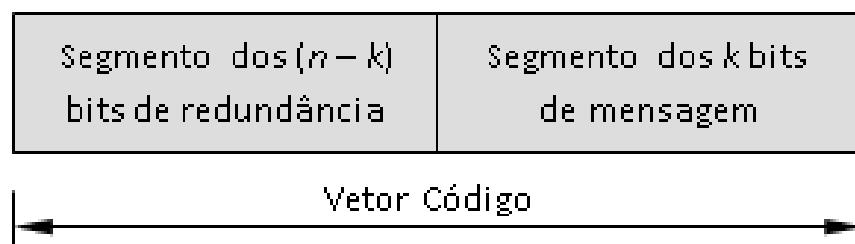
$$\begin{array}{ll}
 \left. \begin{array}{l} 1(01011) \\ 1(10010) \\ 1(01100) \end{array} \right\} & \begin{array}{l} (01011) \\ (10010) \\ (01100) \end{array} \\
 \hline
 & 10101
 \end{array}$$



### 2.3. CÓDIGOS DE BLOCO LINEARES

#### 2.3.2. CODIFICAÇÃO SISTEMÁTICA E NÃO SISTEMÁTICA

- Os vetores códigos apresentados na tabela anterior não apresentam explicitamente a mensagem que o gerou como um segmento do vetor código.
- Essa forma de codificação é chamada de *codificação não sistemática*.
- Uma característica desejável é que o vetor código seja composto por dois segmentos: um segmento composto pelos  $(n - k)$  bits de redundância que permitem a verificação da validade do vetor e outro segmento correspondente aos  $k$  bits da mensagem que gerou os bits de redundância.
- A disposição do segmento redundância e do segmento mensagem é uma questão de convenção. A forma de codificação que permite a obtenção do vetor código nesse formato é chamada de *codificação sistemática*





## 2.3. CÓDIGOS DE BLOCO LINEARES

### 2.3.2. CODIFICAÇÃO SISTEMÁTICA E NÃO SISTEMÁTICA

- Vetores códigos com a convenção mostrada podem ser obtidos a partir de matrizes geradoras com um formato específico.
- Este formato consiste de uma matriz geradora formada por duas outras matrizes:  
uma matriz de paridade com dimensões  $k \times (n - k)$   
uma matriz identidade de dimensões  $k \times k$ .

$$\mathbf{G} = \left[ \mathbf{P}_{k \times (n-k)} \mid \mathbf{I}_{k \times k} \right] = \left[ \begin{array}{cccc|cccc} p_{00} & p_{01} & \cdots & p_{0, n-k-1} & 1 & 0 & \cdots & 0 \\ p_{10} & p_{11} & \cdots & p_{1, n-k-1} & 0 & 1 & & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ p_{k-1, 0} & p_{k-1, 1} & \cdots & p_{k-1, n-k-1} & 0 & 0 & \cdots & 1 \end{array} \right] = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_{k-1} \end{bmatrix}$$



### 2.3. CÓDIGOS DE BLOCO LINEARES

#### 2.3.2. CODIFICAÇÃO SISTEMÁTICA E NÃO SISTEMÁTICA

##### EXEMPLO 2.9

A partir da matriz geradora do código (5, 3) apresentada pelo exemplo 2.8 pede-se:

- Obter uma matriz geradora na forma sistemática.
- Construir uma tabela com os vetores mensagens e seus respectivos vetores códigos.
- Do exemplo 2.8 tem-se:

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \mathbf{g}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix}$$

A matriz na forma sistemática,  $\mathbf{G}'$  correspondente à matriz  $\mathbf{G}$ , é obtida a partir das seguintes operações a partir da matriz  $\mathbf{G}$

$$\mathbf{g}_0' = \mathbf{g}_2 = 01100$$

$$\mathbf{g}_1' = \mathbf{g}_1 = 10010$$

$$\mathbf{g}_2' = \mathbf{g}_0 + \mathbf{g}_1 = 11001$$



## 2.3. CÓDIGOS DE BLOCO LINEARES

### 2.3.2. CODIFICAÇÃO SISTEMÁTICA E NÃO SISTEMÁTICA

#### EXEMPLO 2.9

$$\mathbf{G}' = \begin{bmatrix} \mathbf{g}_0' \\ \mathbf{g}_1' \\ \mathbf{g}_2' \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

b) Repetindo a operação apresentada em

$$\mathbf{c} = \mathbf{m} \cdot \mathbf{G} = (m_0, m_1, \dots, m_{k-1}) \bullet \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_{k-1} \end{bmatrix} = m_0 \mathbf{g}_0 + m_1 \mathbf{g}_1 + \dots + m_{k-1} \mathbf{g}_{k-1}.$$

para a matriz  $\mathbf{G}'$ , obtém-se os vetores códigos apresentados na Tabela.



### 2.3. CÓDIGOS DE BLOCO LINEARES

#### 2.3.2. CODIFICAÇÃO SISTEMÁTICA E NÃO SISTEMÁTICA

##### EXEMPLO 2.9

Tabela - Vetores códigos do código (5, 3) gerados a partir da matriz  $G'$ .

m	$c = m \cdot G'$	c
000	$c_0 = 0(01100) + 0(10010) + 0(11001)$	00000
100	$c_1 = 1(01100) + 0(10010) + 0(11001)$	01100
010	$c_2 = 0(01100) + 1(10010) + 0(11001)$	10010
110	$c_3 = 1(01100) + 1(10010) + 0(11001)$	11110
001	$c_4 = 0(01100) + 0(10010) + 1(11001)$	11001
101	$c_5 = 1(01100) + 0(10010) + 1(11001)$	10101
011	$c_6 = 0(01100) + 1(10010) + 1(11001)$	01011
111	$c_7 = 1(01100) + 1(10010) + 1(11001)$	00111

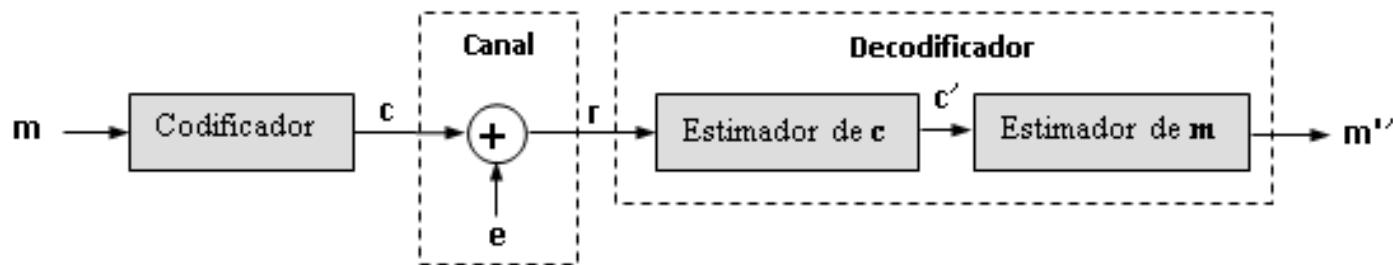
Observe que tanto a matriz  $G$  quanto a matriz  $G'$  geram o mesmo subespaço vetorial. Entretanto, para cada uma das mensagens os vetores códigos gerados são diferentes em cada caso.



## 2.3. CÓDIGOS DE BLOCO LINEARES

### 2.3.3. MATRIZ VERIFICADORA DE PARIDADE

O vetor recebido,  $r$ , pode ser entendido como um vetor código que, ao ser transmitido através de um canal de comunicação, pode ter sofrido uma alteração, consequência da adição de um padrão de erro.



O decodificador em uma abordagem simplista, compara o vetor recebido com todos os vetores códigos.

Entretanto, para valores de  $k$  da ordem de algumas dezenas, esta abordagem pode tornar-se árdua.

Uma forma mais simples para a verificação da validade ou não de um vetor recebido utiliza uma propriedade dos subespaços vetoriais, que pode ser definida da seguinte forma.

*Se um subespaço vetorial,  $S_1$ , pertence a um espaço vetorial,  $V_n$ , composto por todos os vetores de comprimento  $n$ , então deve existir um subespaço vetorial  $S_2$ , que é o espaço nulo ou o espaço dual de  $S_1$ , e que pode ser representado por uma matriz composta por vetores bases linearmente independentes.*



## 2.3. CÓDIGOS DE BLOCO LINEARES

### 2.3.3. MATRIZ VERIFICADORA DE PARIDADE

A matriz geradora do subespaço nulo relativo ao subespaço gerado por  $\mathbf{G}$  é chamada de matriz verificadora de paridade, cuja notação é  $\mathbf{H}$ , e tem dimensões  $(n - k) \times n$ , ou seja,

$$\mathbf{H} = \begin{bmatrix} \mathbf{h}_0 \\ \mathbf{h}_1 \\ \vdots \\ \mathbf{h}_{n-k-1} \end{bmatrix} = \begin{bmatrix} h_{00} & h_{01} & h_{02} & \cdots & h_{0, n-1} \\ h_{10} & h_{11} & h_{12} & \cdots & h_{1, n-1} \\ \vdots & \vdots & \vdots & & \vdots \\ h_{n-k-1, 0} & h_{n-k-1, 1} & h_{n-k-1, 2} & \cdots & h_{n-k-1, n-1} \end{bmatrix}$$

Se um subespaço gerado por  $\mathbf{H}$  é dual ao subespaço gerado por  $\mathbf{G}$  então os vetores de  $\mathbf{G}$  são ortogonais aos vetores de  $\mathbf{H}$ , ou seja

$$\mathbf{G} \cdot \mathbf{H}^T = 0$$

Uma consequência direta da equação acima é que a condição de ortogonalidade de qualquer vetor código,  $\mathbf{c}$ , gerado por  $\mathbf{G}$  em relação ao espaço nulo gerado por  $\mathbf{H}$  é verdadeira, i.e.,

$$\mathbf{c} \cdot \mathbf{H}^T = 0$$



## 2.3. CÓDIGOS DE BLOCO LINEARES

### 2.3.3. MATRIZ VERIFICADORA DE PARIDADE

Quando a matriz **G** está na forma sistemática

$$\mathbf{G} = \left[ \mathbf{P}_{k \times (n-k)} \mid \mathbf{I}_{k \times k} \right]$$

a obtenção da matriz **H** é direta

$$\mathbf{H} = \left[ \mathbf{I}_{(n-k) \times (n-k)} \mid \mathbf{P}^T \right] = \left[ \begin{array}{cccc|cccccc} 1 & 0 & \cdots & 0 & p_{00} & p_{10} & \cdots & p_{k-1,0} \\ 0 & 1 & & 0 & p_{01} & p_{11} & \cdots & p_{k-1,1} \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 & p_{0,n-k-1} & p_{1,n-k-1} & \cdots & p_{k-1,n-k-1} \end{array} \right]$$



## 2.3. CÓDIGOS DE BLOCO LINEARES

### 2.3.3. MATRIZ VERIFICADORA DE PARIDADE

#### EXEMPLO 2.10

A partir da matriz geradora para o código (5, 3), apresentada no exemplo 2.9, pede-se:

- a) Obter a matriz verificadora de paridade  $\mathbf{H}$ .
- b) Verificar a condição de ortogonalidade para o vetor código correspondente ao vetor mensagem  $\mathbf{m} = 101$ .
- a) A partir da matriz geradora na forma sistemática

$$\mathbf{G} = \left[ \mathbf{P}_{k \times (n-k)} \mid \mathbf{I}_{k \times k} \right] = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

obtém-se a matriz  $\mathbf{H}$  na forma

$$\mathbf{H} = \left[ \mathbf{I}_{(n-k) \times (n-k)} \mid \mathbf{P}^T \right] = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \end{bmatrix}$$



## 2.3. CÓDIGOS DE BLOCO LINEARES

### 2.3.3. MATRIZ VERIFICADORA DE PARIDADE

#### EXEMPLO 2.10

b) O vetor código,  $\mathbf{c}$ , correspondente ao vetor mensagem  $\mathbf{m} = 101$  pode ser obtido conforme:

$$\mathbf{c} = \mathbf{m} \cdot \mathbf{G} = (101) \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{bmatrix} = 1(01100) + 0(10010) + 1(11001)$$

$$\mathbf{c} = 10101$$

A condição de ortogonalidade pode ser verificada a partir do resultado do produto interno entre o vetor código,  $\mathbf{c}$ , e a matriz verificadora de paridade transposta  $\mathbf{H}^T$ , conforme mostrado a seguir

$$\mathbf{c} \cdot \mathbf{H}^T = (10101) \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{bmatrix} = 1(10) + 0(01) + 1(01) + 0(10) + 1(11) = 00$$

\* \* \*



## 2.3. CÓDIGOS DE BLOCO LINEARES

### 2.3.4. DISTÂNCIA MÍNIMA DE UM CÓDIGO DE BLOCO LINEAR

Considere o conjunto de distâncias entre todos os pares de vetores código em um espaço  $V_n$ . O menor membro do conjunto é a *distância mínima do código* e é denotado por  $d_{\min}$ .

Mais uma vez a propriedade dos códigos lineares discutida anteriormente permite afirmar que se  $\mathbf{c}_1$  e  $\mathbf{c}_2$  são vetores código, então o vetor  $\mathbf{c}_3$  obtido pela operação  $\mathbf{c}_1 \oplus \mathbf{c}_2$  é também um vetor código. Assim a distância de Hamming entre dois vetores códigos é determinada como sendo

$$d(\mathbf{c}_1, \mathbf{c}_2) = w(\mathbf{c}_1 \oplus \mathbf{c}_2) = w(\mathbf{c}_3)$$

Portanto, não há necessidade de examinarmos as distâncias entre todas as combinações possíveis entre pares de palavras-código, basta que se verifique o peso de cada palavra código, com exceção da palavra toda zero. O menor peso encontrado corresponde à menor distância mínima do código.



## 2.3. CÓDIGOS DE BLOCO LINEARES

### 2.3.4. DISTÂNCIA MÍNIMA DE UM CÓDIGO DE BLOCO LINEAR

#### Exemplo 2.11

Determine a distância mínima do código (6, 3), definida pela matriz geradora na forma sistemática apresentada a seguir.

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Uma vez que este código possui poucas palavras códigos, uma solução é listá-las por meio da operação  $\mathbf{c} = \mathbf{m} \cdot \mathbf{G}$ . O resultado dessa operação para todas as possíveis mensagens com 3 bits está mostrado na tabela a seguir.

Inspeciona-se que as palavras de menor peso são as palavras com peso 3.  
Logo a distância mínima é igual a 3.

m	c
000	000000
100	101100
010	110010
110	011110
001	011001
101	110101
011	101011
111	000111



### 2.3. CÓDIGOS DE BLOCO LINEARES

#### 2.3.4. DISTÂNCIA MÍNIMA DE UM CÓDIGO DE BLOCO LINEAR

##### Exemplo 2.11

Ou alternativamente, inspecionando-se a matriz verificadora de paridade, apresentada abaixo, é fácil verificar que o menor número de colunas que quando somadas resulta em uma coluna toda zero é 3, por exemplo, 1<sup>a</sup> + 2<sup>a</sup> + 5<sup>a</sup> ou 2<sup>a</sup> + 3<sup>a</sup> + 6<sup>a</sup> ou 1<sup>a</sup> + 3<sup>a</sup> + 4<sup>a</sup>.

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}$$

É muito comum um código de bloco linear  $(n, k)$  com distância mínima  $d_{min}$  ser representado pela notação  $(n, k, d_{min})$ .

O código  $(6, 3)$  do Exemplo 2.11 é um código  $(6, 3, 3)$ .

\* \* \*



### 2.3. CÓDIGOS DE BLOCO LINEARES

#### 2.3.5. CAPACIDADE DE CORREÇÃO E DE DETECÇÃO DE ERROS DE UM CÓDIGO DE BLOCO LINEAR

A capacidade de correção de erro, de um código de bloco linear,  $t$ , é

$$t = \left\lfloor \frac{d_{\min} - 1}{2} \right\rfloor$$

onde  $\lfloor x \rfloor$  significa o maior inteiro que não excede o valor de  $x$ .

Um código com  $d_{\min} = 4$ , pode corrigir todos os padrões de 1 erro ( $t = 1$ ).

A capacidade de detecção de erros do código é determinada por

$$e = d_{\min} - 1$$

onde  $e$  é o número de erros detectados.



### 2.3. CÓDIGOS DE BLOCO LINEARES

#### 2.3.6. SÍNDROME DE ERRO

- Seja  $\mathbf{r} = r_1, r_2, \dots, r_n$  um vetor recebido com uma das  $2^n$  palavras de  $n$  bits do espaço vetorial  $V_n$

$$\mathbf{r} = \mathbf{c} + \mathbf{e}$$

onde  $\mathbf{e} = e_1, e_2, \dots, e_n$  é um vetor erro ou padrão de erro introduzido pelo canal.

- *Síndrome de erro* é um vetor com  $n - k$  bits definido, pela operação

$$\mathbf{S} = \mathbf{r} \cdot \mathbf{H}^T$$

Combinando as equações acima tem-se:

$$\mathbf{S} = (\mathbf{c} + \mathbf{e}) \cdot \mathbf{H}^T = \mathbf{c} \cdot \mathbf{H}^T + \mathbf{e} \cdot \mathbf{H}^T$$

mas,

$$\mathbf{c} \cdot \mathbf{H}^T = 0$$

logo,

$$\mathbf{S} = \mathbf{e} \cdot \mathbf{H}^T$$



### 2.3. CÓDIGOS DE BLOCO LINEARES

#### 2.3.6. SÍNDROME DE ERRO

Logo, a síndrome **S** está associada a um padrão de erro. Esta é uma importante propriedade, fundamental para o processo de decodificação, ou seja, cada padrão de erro corrigível deve estar associado a uma síndrome específica.

É importante notar que para isso ocorrer, duas propriedades da matriz verificadora de paridade são necessárias:

*Nenhuma coluna da matriz **H** pode ser toda zero, caso contrário, um erro na posição correspondente à linha toda zero seria indetectável;*

*Todas as colunas de **H** devem ser únicas. Se duas colunas de **H** forem iguais, erros nas posições correspondentes a essas linhas podem ser indistinguíveis.*

*Um código de bloco linear  $(n, k)$  com capacidade de correção de  $t$  erros é capaz de corrigir um total de  $2^{n-k}$  padrões de erros.*

**2.3. CÓDIGOS DE BLOCO LINEARES****2.3.6. SÍNDROME DE ERRO****EXEMPLO 2.12**

Considere o código de bloco linear (6, 3, 3) gerado por.

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Pede-se:

- Determinar a capacidade de correção de erros do código.
- Listar todos os padrões de erros corrigíveis dentro da capacidade de correção de erros do código.
- Listar todas as síndromes de erros associadas aos padrões de erros corrigíveis dentro da capacidade de correção de erros do código.



### 2.3. CÓDIGOS DE BLOCO LINEARES

#### 2.3.6. SÍNDROME DE ERRO

a) A capacidade de correção de erros do código.

Como a distância mínima deste código é  $d_{min} = 3$ , então,

$$t = \left\lfloor \frac{d_{min} - 1}{2} \right\rfloor = \left\lfloor \frac{3-1}{2} \right\rfloor \Rightarrow t = 1$$

b) Padrões de erros corrigíveis dentro da capacidade de correção de erros do código

Como a capacidade de correção de erro é  $t = 1$ , então este código é capaz de corrigir todos os padrões com 1 erro, ou seja,

e (t = 1)
100000
010000
001000
000100
000010
000001



### 2.3. CÓDIGOS DE BLOCO LINEARES

#### 2.3.6. SÍNDROME DE ERRO

c) Síndromes de erros associadas aos padrões de erros corrigíveis dentro da capacidade de correção de erros do código. Da matriz  $\mathbf{H}$  (2.28), obtém-se  $\mathbf{H}^T$ , ou seja,

$$\mathbf{H}^T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$$

$$\mathbf{S} = \mathbf{e} \cdot \mathbf{H}^T \quad S = [1 \ 0 \ 0 \ 0 \ 0 \ 0] \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} = [1 \ 0 \ 0]$$



$\mathbf{e} (t = 1)$	$S$
100000	100
010000	010
001000	001
000100	101
000010	110
000001	011

Para todos os valores de  $\mathbf{e} (t = 1)$  obtém-se:

Padrões de erros corrigíveis e suas respectivas síndromes para o código de bloco linear (6, 3, 3), dados pela matriz  $\mathbf{G}$ .

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$



### 2.3. CÓDIGOS DE BLOCO LINEARES

#### 2.3.7. CORREÇÃO DE ERROS PELA SÍNDROME

Algoritmo de correção de erros pela Síndrome:

1. A partir da capacidade de correção de erros do código, calcula-se a síndrome para todos os padrões de erros corrigíveis:

$$\mathbf{S} = \mathbf{e} \cdot \mathbf{H}^T$$

2. Calcula-se a síndrome de  $\mathbf{r}$ :

$$\mathbf{S} = \mathbf{r} \cdot \mathbf{H}^T$$

3. Localiza-se o padrão de erro correspondente à síndrome calculada.
4. Considerando que o vetor recebido é  $\mathbf{r} = \mathbf{c} + \mathbf{e}$ , então o vetor código será determinado por:

$$\mathbf{c}' = \mathbf{r} + \mathbf{e}$$

- Onde  $\mathbf{c}'$  é a melhor estimativa do vetor código que foi transmitido pelo canal ruidoso.



### 2.3. CÓDIGOS DE BLOCO LINEARES

#### 2.3.7. CORREÇÃO DE ERROS PELA SÍNDROME

##### EXEMPLO 2.13

Suponha que o vetor  $\mathbf{c} = 101011$  do código  $(6, 3, 3)$ , gerado por

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

tenha sido transmitido e corrompido por ruído no canal, de modo que na recepção foi detectado o vetor  $\mathbf{r} = 101010$ . Corrija o erro introduzido pelo canal a partir da associação da síndrome com o padrão de erro mais provável.



### 2.3. CÓDIGOS DE BLOCO LINEARES

#### 2.3.7. CORREÇÃO DE ERROS PELA SÍNDROME

##### EXEMPLO 2.13

A síndrome de erros para o vetor recebido é determinada por meio de

$$\mathbf{S} = \mathbf{r} \cdot \mathbf{H}^T$$

$$\mathbf{S} = \mathbf{r} \cdot \mathbf{H}^T = (101010) \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} = 1(100) + 0(010) + 1(001) + 0(101) + 1(110) + 0(011)$$

$$\mathbf{S} = 100 + 001 + 110 \qquad \Rightarrow \qquad \mathbf{S} = 011$$



### 2.3. CÓDIGOS DE BLOCO LINEARES

#### 2.3.7. CORREÇÃO DE ERROS PELA SÍNDROME

##### EXEMPLO 2.13

As Síndromes e seus respectivos padrões para o código de bloco linear (6, 3, 3) gerado por  $G$  é

$$\mathbf{S} = \mathbf{e} \cdot \mathbf{H}^T$$

S	e ( $t = 1$ )
000	000000
001	001000
010	010000
011	000001
100	100000
101	000100
110	000010

De acordo a Tabela a síndrome calculada corresponde ao padrão de erro  $\mathbf{e} = 000001$ .

Logo, o vetor código mais provável de ter sido o vetor transmitido pode ser determinado por

$$\mathbf{c}' = \mathbf{r} + \mathbf{e} = 101010 + 000001$$

$$\mathbf{c}' = 101011$$

\* \* \*





## 2.3. CÓDIGOS DE BLOCO LINEARES

### 2.3.8. CODIFICADOR PARA CÓDIGOS DE BLOCO SISTEMÁTICOS SIMPLES

- A implementação do circuito de codificação pode ser feita com as equações que determinam as palavras códigos.
- Para um código de bloco linear sistemático, a palavra código pode ser escrita como

$$\mathbf{c} = (p_0, p_1, \dots, p_{n-k-1}, m_0, m_1, \dots, m_{k-1})$$

- Note que a obtenção dos bits de paridade a partir de

$$\mathbf{G} = [\mathbf{P}_{k \times (n-k)} | \mathbf{I}_{k \times k}] = \left[ \begin{array}{cccc|cccc} p_{00} & p_{01} & \cdots & p_{0,n-k-1} & 1 & 0 & \cdots & 0 \\ p_{10} & p_{11} & \cdots & p_{1,n-k-1} & 0 & 1 & & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ p_{k-1,0} & p_{k-1,1} & \cdots & p_{k-1,n-k-1} & 0 & 0 & \cdots & 1 \end{array} \right] = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_{k-1} \end{bmatrix}$$

$$p_0 = m_0 \cdot p_{00} + m_1 \cdot p_{10} + \dots + m_{k-1} \cdot p_{k-1,0}$$

- resume-se às seguintes operações:  $p_1 = m_0 \cdot p_{01} + m_1 \cdot p_{11} + \dots + m_{k-1} \cdot p_{k-1,1}$

⋮

$$p_{n-k-1} = m_0 \cdot p_{0,n-k-1} + m_1 \cdot p_{1,n-k-1} + \dots + m_{k-1} \cdot p_{k-1,n-k-1}$$



### 2.3. CÓDIGOS DE BLOCO LINEARES

#### 2.3.8. CODIFICADOR PARA CÓDIGOS DE BLOCO SISTEMÁTICOS SIMPLES

##### EXEMPLO 2.14

Construa um codificador para o código (6, 3) representado pela sua matriz geradora reproduzida a seguir

$$\mathbf{G} = \left[ \mathbf{P}_{k \times (n-k)} \mid \mathbf{I}_{k \times k} \right] = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

De acordo com as equações de paridade para a matriz geradora são:

$$p_0 = m_0 \cdot p_{00} + m_1 \cdot p_{10} + \dots + m_{k-1} \cdot p_{k-1,0}$$

$$p_1 = m_0 \cdot p_{01} + m_1 \cdot p_{11} + \dots + m_{k-1} \cdot p_{k-1,1}$$

$$\vdots$$

$$p_{n-k-1} = m_0 \cdot p_{0,n-k-1} + m_1 \cdot p_{1,n-k-1} + \dots + m_{k-1} \cdot p_{k-1,n-k-1}$$

$$p_0 = m_0 + m_1$$

$$p_1 = m_1 + m_2$$

$$p_2 = m_0 + m_2$$

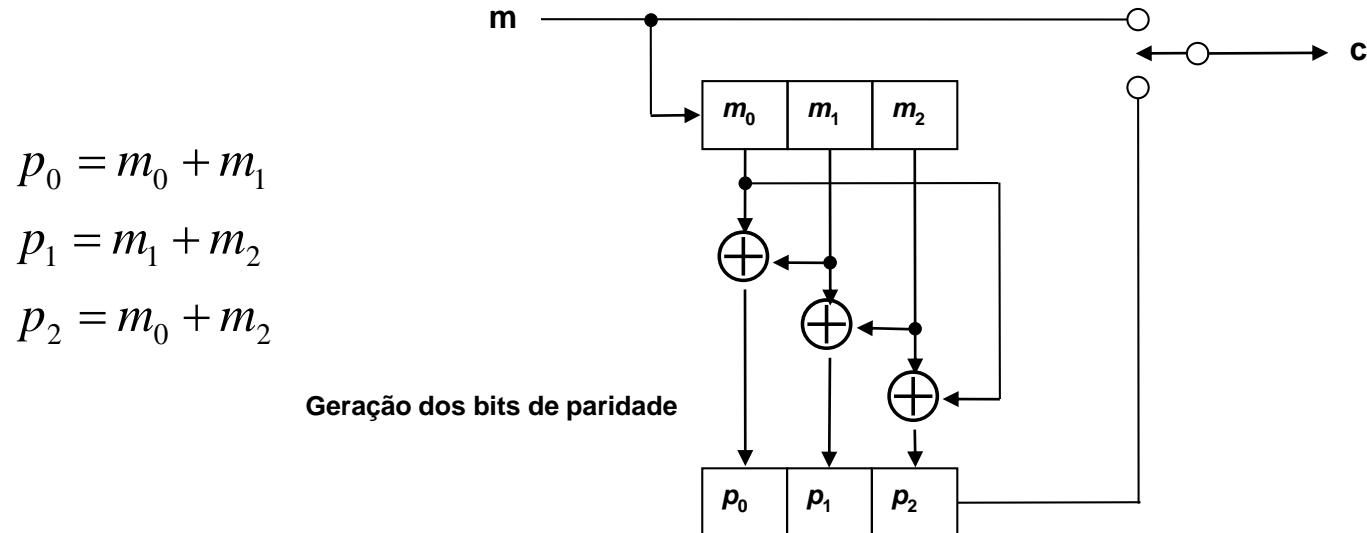


### 2.3. CÓDIGOS DE BLOCO LINEARES

#### 2.3.8. CODIFICADOR PARA CÓDIGOS DE BLOCO SISTEMÁTICOS SIMPLES

##### EXEMPLO 2.14

Logo, um circuito codificador, para o código (6, 3), pode ser implementado conforme apresentado a seguir:



\* \* \*



### 2.3. CÓDIGOS DE BLOCO LINEARES

#### 2.3.9. DECODIFICADOR PARA CÓDIGOS DE BLOCO SISTEMÁTICOS SIMPLES

A implementação segue os seguintes passos:

1. Calculo da síndrome;
2. Localização do padrão de erro;
3. Soma módulo-2 do padrão de erro com o vetor recebido.

O cálculo da síndrome pode ser feito considerando as expressões apresentadas a seguir.

$$\mathbf{S} = \mathbf{r} \cdot \mathbf{H}^T = [r_0, r_1, \dots, r_{n-1}] \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 \\ p_{00} & p_{01} & \cdots & p_{0,n-k-1} \\ p_{10} & p_{11} & \cdots & p_{1,n-k-1} \\ \vdots & \vdots & & \vdots \\ p_{k-1,0} & p_{k-1,1} & \cdots & p_{k-1,n-k-1} \end{bmatrix} \quad \left\{ \begin{array}{l} s_0 = r_0 + r_{n-k} \cdot p_{00} + r_{n-k+1} \cdot p_{10} + \dots + r_{n-1} \cdot p_{k-1,0} \\ s_1 = r_1 + r_{n-k} \cdot p_{01} + r_{n-k+1} \cdot p_{11} + \dots + r_{n-1} \cdot p_{k-1,1} \\ \vdots \\ s_{n-k-1} = r_{n-k-1} + r_{n-k} \cdot p_{0,n-k-1} + r_{n-k+1} \cdot p_{1,n-k-1} + \dots + r_{n-1} \cdot p_{k-1,n-k-1} \end{array} \right.$$

$$\mathbf{S} = \mathbf{r} \cdot \mathbf{H}^T = [s_0, s_1, \dots, s_{n-k-1}]$$



### 2.3. CÓDIGOS DE BLOCO LINEARES

#### 2.3.9. DECODIFICADOR PARA CÓDIGOS DE BLOCO SISTEMÁTICOS SIMPLES

##### EXEMPLO 2.15

Construa um decodificador para o código (6, 3) representado pela sua matriz verificadora de paridade transposta reproduzida a seguir.

$$\mathbf{H}^T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$$

As síndromes a partir da matriz verificadora de paridade acima são:

$$\mathbf{S} = \mathbf{r} \cdot \mathbf{H}^T = [r_0 \ r_1 \ r_2 \ r_3 \ r_4 \ r_5] \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} = [s_0, s_1, s_2] \quad \begin{cases} s_0 = r_0 + r_3 + r_4 \\ s_1 = r_1 + r_4 + r_5 \\ s_2 = r_2 + r_3 + r_5 \end{cases}$$

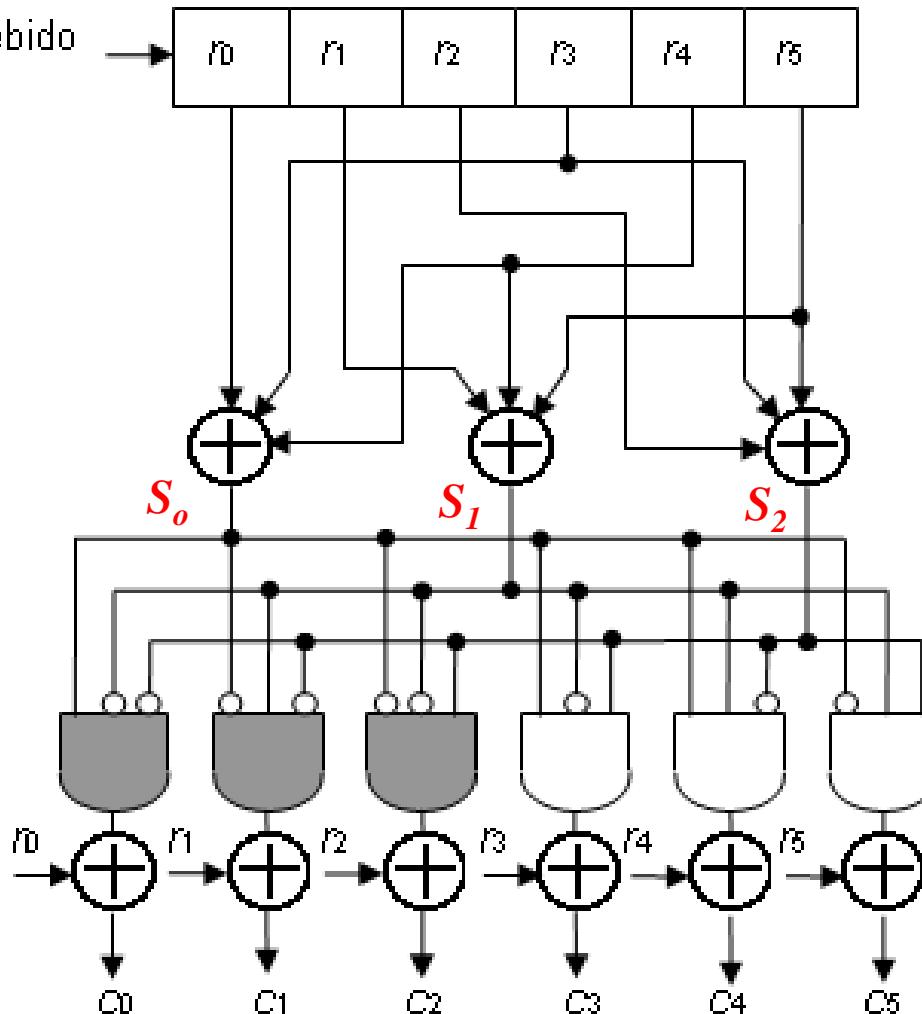


### 2.3. CÓDIGOS DE BLOCO LINEARES

#### 2.3.9. DECODIFICADOR PARA CÓDIGOS DE BLOCO SISTEMÁTICOS SIMPLES

$$\begin{cases} s_0 = r_0 + r_3 + r_4 \\ s_1 = r_1 + r_4 + r_5 \\ s_2 = r_2 + r_3 + r_5 \end{cases}$$

$$\mathbf{H}^T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$$





## 2.4. DESEMPENHO DOS CÓDIGOS DE BLOCO LINEARES

Canal binário simétrico (BSC)

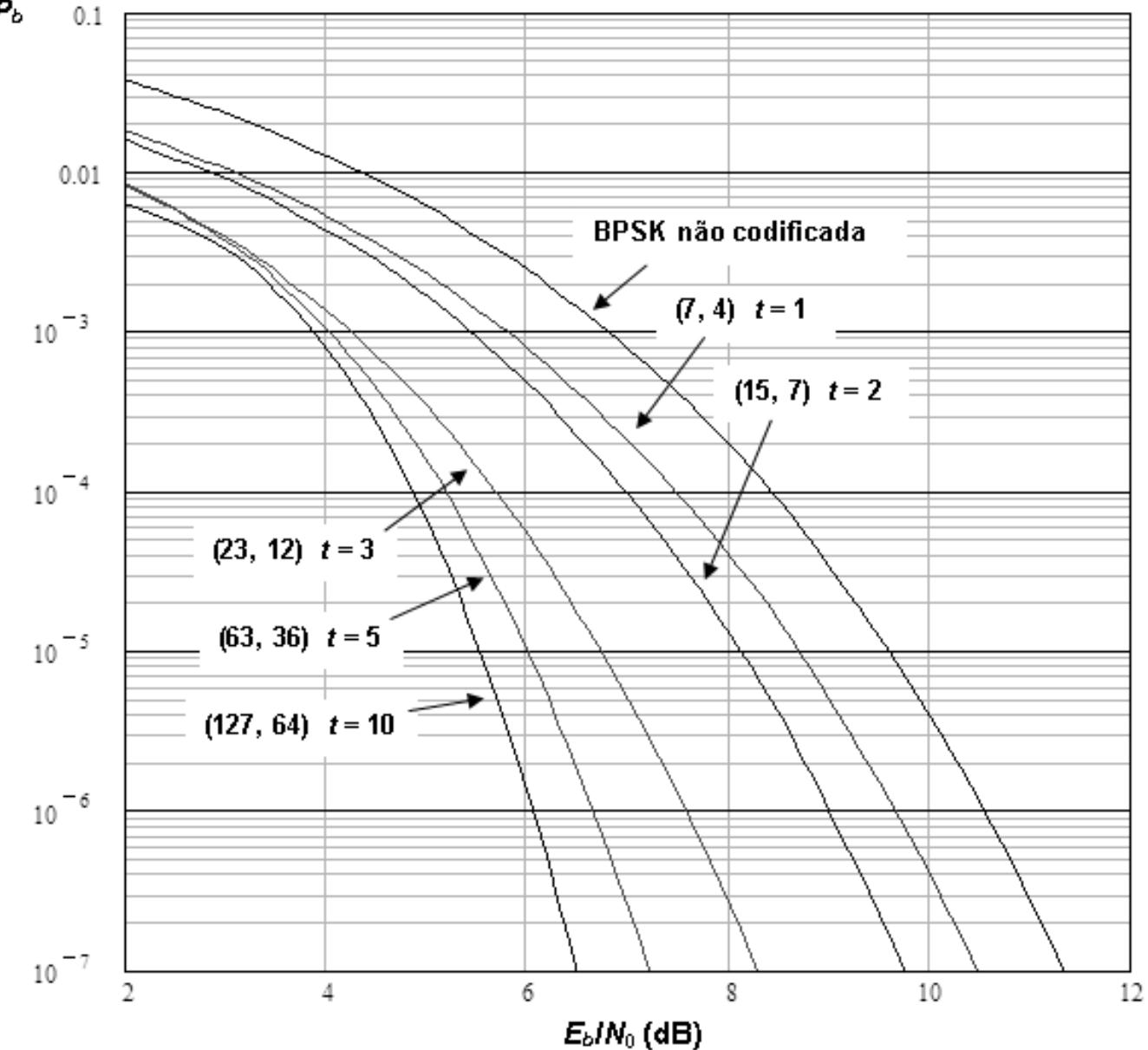
$$P_b \approx \frac{1}{n} \sum_{j=t+1}^n j \binom{n}{j} p^j (1-p)^{n-j}$$

Canal AWGN com modulação BPSK

$$p = \frac{1}{2} erfc \sqrt{R_c \frac{E_b}{N_0}}$$

Taxa de codificação entre 0,47 e 0,57

A capacidade de correção de erros por bloco aumenta com o aumento do comprimento quando a taxa de codificação é mantida constante.





## 2.5. CÓDIGOS CÍCLICOS

### 2.5.1. INTRODUÇÃO AOS CÓDIGOS CÍCLICOS

- São uma subclasse de códigos de bloco lineares.
- São códigos implementáveis com registradores de deslocamento realimentados.
- O cálculo da síndrome também pode ser executado de forma similar, com registradores de deslocamento realimentados.
- Um código linear  $(n, k)$  é chamado de código cíclico se ele pode ser descrito pela propriedade apresentada a seguir.

Se a  $n$ -tupla

$$\mathbf{c} = (c_0, c_1, c_2, \dots, c_{n-1})$$

é um vetor código no subespaço  $S$ , então,

$$\mathbf{c}^{(1)} = (c_{n-1}, c_0, c_1, \dots, c_{n-2})$$

obtido pelo deslocamento correspondente a uma posição de bit, é também um vetor código em  $S$ .



## 2.5. CÓDIGOS CÍCLICOS

### 2.5.1. INTRODUÇÃO AOS CÓDIGOS CÍCLICOS

Expressão Geral:

$$\mathbf{c}^{(i)} = (c_{n-i}, c_{n-i+1}, c_1, \dots, c_{n-1}, c_0, c_1, \dots, c_{n-i-1})$$

obtido pelo deslocamento correspondente a  $i$  posições de bit, é também um vetor código em  $S$ .

Os componentes de um vetor código podem ser tratados como os coeficientes de um polinômio  $c(X)$ .

$$c(X) = c_0 + c_1X + c_2X^2 + \dots + c_{n-1}X^{n-1}$$

Nesta representação a presença ou ausência de cada termo no polinômio indica a presença de um 1 ou 0.



## 2.5. CÓDIGOS CÍCLICOS

### 2.5.1. INTRODUÇÃO AOS CÓDIGOS CÍCLICOS

#### Exemplo 2.16

Seja o vetor código

$$\mathbf{c} = 0 \ 0 \ 0 \ 1 \ 1 \ 0 \ 1$$

de um código de bloco linear  $(7, 4)$ . Sua representação polinomial é

$$c(X) = X^3 + X^4 + X^6$$

ou seja, um polinômio de grau  $n - 1$ . O valor de  $c^{(3)}$ , que também pertence ao mesmo código  $(7, 4)$  é

$$c^{(3)} = 1 \ 0 \ 1 \ 0 \ 0 \ 0 \ 1.$$

\* \* \*



## 2.5. CÓDIGOS CÍCLICOS

### 2.5.1. INTRODUÇÃO AOS CÓDIGOS CÍCLICOS

- Geração dos códigos cíclicos:  
Usa-se um polinômio gerador, da mesma forma que são gerados os códigos de bloco usando uma matriz geradora.
- O polinômio gerador  $g(X)$  para um código cíclico  $(n, k)$  tem a forma

$$g(X) = g_0 + g_1 X + g_2 X^2 + \dots + g_{n-k} X^{n-k}$$

onde  $g_0$  e  $g_{n-k}$  devem ser iguais a 1 e o grau do polinômio gerador deve ser  $n - k$ .

- Um polinômio  $g(X)$  é um polinômio gerador de um código cíclico  $(n, k)$  se, e somente se, ele for um fator de  $X^n + 1$ .



## 2.5. CÓDIGOS CÍCLICOS

### 2.5.1. INTRODUÇÃO AOS CÓDIGOS CÍCLICOS

#### EXEMPLO 2.17

Verifique se o polinômio  $X^3 + X + 1$  gera um código cíclico  $C = (7, 4)$ .

$X^7 + 1$	$X^3 + X + 1$
$(X^7 + X^5 + X^4)$	$X^4 + X^2 + X + 1$
$0 + X^5 + X^4 + 1$	
$(X^5 + X^3 + X^2)$	
$0 + X^4 + X^3 + X^2 + 1$	
$(X^4 + X^2 + X)$	
$0 + X^3 + X + 1$	
$(X^3 + X + 1)$	
0	

Conclusão: O polinômio  $X^3 + X + 1$  gera um código cíclico  $(7, 4)$  e ainda permite-nos concluir que o polinômio  $X^4 + X^2 + X + 1$ , que é o quociente da divisão realizada, gera um código cíclico  $(7, 3)$ , pois

$$\text{Dividendo} = \text{divisor} \times \text{quociente} + \text{resto} \quad \longrightarrow \quad (X^7 + 1) = (X^3 + X + 1)(X^4 + X^2 + X + 1)$$



## 2.5. CÓDIGOS CÍCLICOS

### 2.5.1. INTRODUÇÃO AOS CÓDIGOS CÍCLICOS

- A matriz geradora para um código cíclico gerado pelo polinômio gerador  $g(X)$  pode ser obtida fazendo

$$G = \begin{bmatrix} g_0 & g_1 & g_2 & \cdots & g_{n-k} & 0 & 0 & 0 & \cdots & 0 \\ 0 & g_0 & g_1 & g_2 & \cdots & g_{n-k} & 0 & 0 & \cdots & 0 \\ 0 & 0 & g_0 & g_1 & g_2 & \cdots & g_{n-k} & 0 & \cdots & 0 \\ \vdots & & & & & & \vdots & & & \\ 0 & \cdots & 0 & 0 & 0 & g_0 & g_1 & g_2 & \cdots & g_{n-k} \end{bmatrix}$$

Onde  $g_0, g_1, g_2, \dots, g_{n-k}$  são os termos do polinômio  $g(X) = g_0 + g_1X + g_2X^2 + \dots + g_{n-k}X^{n-k}$ .



## 2.5. CÓDIGOS CÍCLICOS

### 2.5.1. INTRODUÇÃO AOS CÓDIGOS CÍCLICOS

- É possível fazer a codificação de forma sistemática, através de uma matriz geradora  $G'$  obtida a partir da matriz  $G$ .
- Para isso, conforme já visto,  $G'$  deve ter a forma

$$\mathbf{G}' = \begin{bmatrix} \mathbf{P}_{k,(n-k)} & \vdots & \mathbf{I}_k \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1,(n-k)} & 1 & 0 & \cdots & 0 \\ p_{21} & p_{22} & \cdots & p_{2,(n-k)} & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots & \vdots \\ p_{k1} & p_{k2} & \cdots & p_{k,(n-k)} & 0 & 0 & \cdots & 1 \end{bmatrix}$$

- Isso pode ser feito através de operações lineares com as linhas de  $G$  até que  $G'$  tome a forma desejada.



## 2.5. CÓDIGOS CÍCLICOS

### 2.5.1. INTRODUÇÃO AOS CÓDIGOS CÍCLICOS

#### EXEMPLO 2.18

Determine o vetor código de um código cíclico (7, 4), correspondente a mensagem  $m = 1011$ , utilizando a matriz geradora na forma sistemática, obtida a partir do polinômio gerador

$$g(X) = X^3 + X + 1.$$

Do exemplo anterior, a matriz geradora na forma não sistemática é

$$G = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}$$



## 2.5. CÓDIGOS CÍCLICOS

### 2.5.1. INTRODUÇÃO AOS CÓDIGOS CÍCLICOS

$$G = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}$$

Por inspeção verifica-se que a primeira e a segunda linha estão corretamente posicionadas para a obtenção de uma matriz na forma sistemática.

A terceira linha da matriz pode ser obtida através da soma das linhas 1 e 3.

A quarta linha da matriz pode ser obtida somando-se as linhas 1, 2 e 4.

$$G' = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$



## 2.5. CÓDIGOS CÍCLICOS

### 2.5.1. INTRODUÇÃO AOS CÓDIGOS CÍCLICOS

O vetor código na forma sistemática é obtido pela operação  $c = m \cdot G'$ , consequentemente

$$c = m \cdot G' = 1011 \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$c = 1001011$$

\* \* \*

Conclui-se que uma vez obtida a matriz  $G'$  na forma sistemática, a obtenção da matriz verificadora de paridade  $H$ , do código  $C$  gerado por  $G'$  é imediata, pois

$$H = [I_{n-k} \quad : \quad P^T]$$



## 2.5. CÓDIGOS CÍCLICOS

### 2.5.2. CODIFICAÇÃO SISTEMÁTICA DE CÓDIGOS CÍCLICOS COM REGISTRADORES DE DESLOCAMENTO DE $(n-k)$ ESTÁGIOS

Um vetor mensagem pode ser escrito na forma polinomial como

$$m(X) = m_0 + m_1 X + m_2 X^2 + \dots + m_{k-1} X^{k-1}$$

Na forma sistemática, os dígitos de mensagem são apresentados explicitamente como parte do vetor código.

Para que a porção mensagem da palavra código ocupe as posições dos bits mais significativos, podemos fazer um deslocamento dos bits de mensagem para a direita, ficando as  $n - k$  posições mais à esquerda para a parte de paridade, ou seja,

$$X^{n-k} m(X) = m_0 X^{n-k} + m_1 X^{n-k+1} + \dots + m_{k-1} X^{n-1}$$

Dividindo a expressão acima por  $g(X)$ , obtém-se

$$X^{n-k} m(X) = q(X)g(X) + r(X) \quad \text{ou, então}$$

$$r(X) + X^{n-k} m(X) = q(X)g(X) = c(X)$$

O resto  $r(X)$  é a parte de paridade do vetor código e o produto  $X^{n-k} m(X)$  é a parte da mensagem que foi deslocada  $n - k$  bits para a direita.



## 2.5. CÓDIGOS CÍCLICOS

### 2.5.2. CODIFICAÇÃO SISTEMÁTICA DE CÓDIGOS CÍCLICOS COM REGISTRADORES DE DESLOCAMENTO DE $(n-k)$ ESTÁGIOS

#### EXEMPLO 2.19

Determine o vetor código de um código cíclico (7,4), na forma sistemática, para  $m = 1011$ , utilizando o polinômio gerador  $g(X) = X^3 + X + 1$ .

$$m(X) = 1 + X^2 + X^3$$

$$\begin{array}{r} X^6 + X^5 + X^3 \\ \hline X^3 + X^2 + X + 1 \end{array}$$

$$X^{n-k} m(X) = X^3 (1 + X^2 + X^3) = X^3 + X^5 + X^6$$

Dividindo  $X^{n-k} m(X)$  por  $g(X)$  pode-se escrever

$$\begin{array}{cccc|c} X^3 + X^5 + X^6 & = & (1 + X + X^2 + X^3) & (1+X+X^3) & + & 1 \\ X^{n-k} m(X) & & q(X) & g(X) & & \text{resto} \end{array}$$

Finalmente,  $c(X) = r(X) + X^3 m(X) = (1 + X^3 + X^5 + X^6)$

$$c = 1001011$$

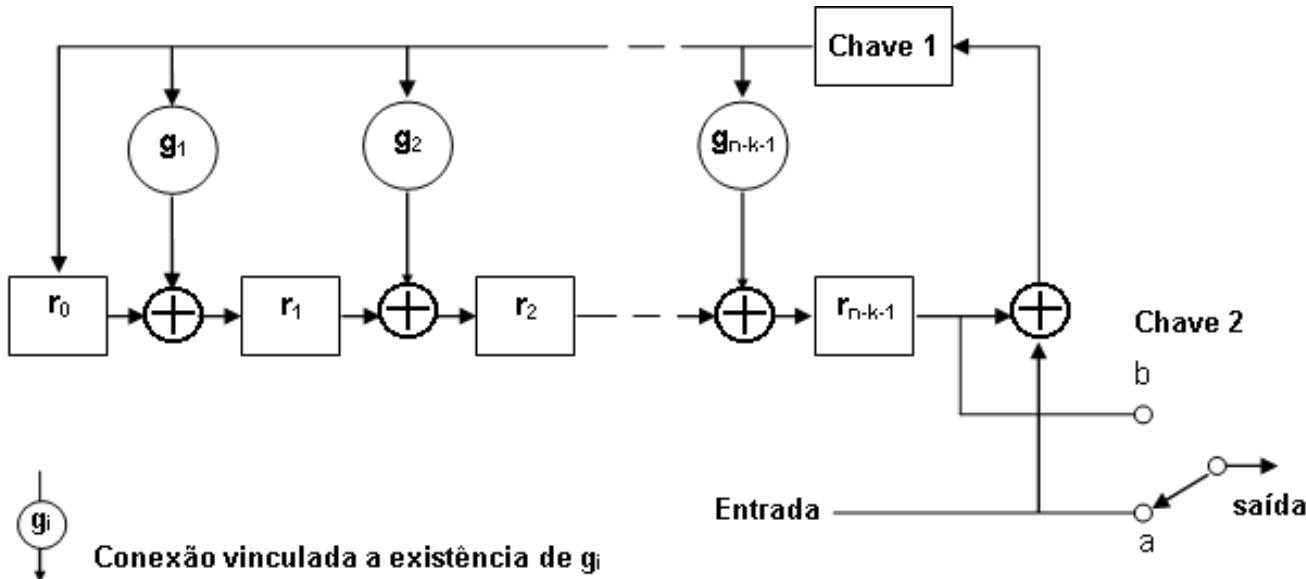
\* \* \*



## 2.5. CÓDIGOS CÍCLICOS

### 2.5.2. CODIFICAÇÃO SISTEMÁTICA DE CÓDIGOS CÍCLICOS COM REGISTRADORES DE DESLOCAMENTO DE $(n-k)$ ESTÁGIOS

O circuito que faz as operações polinomiais apresentadas anteriormente está apresentado na Figura.



Passo 1: A chave 1 permanece fechada, para permitir a entrada dos bits de mensagem no estágio de codificação. A chave 2 permanece na posição (a) para permitir a transmissão dos bits de mensagem diretamente para o registro de saída, durante os primeiros  $k$  deslocamentos.

Passo 2: Após a transmissão dos  $k$  bits de mensagem a chave 1 é aberta (impedindo a realimentação) e a chave 2 é movida para a posição (b).

Passo 3: Os  $(n-k)$  bits de paridade que estão armazenados nos registros de deslocamento são transmitidos, completando a transmissão do polinômio código.

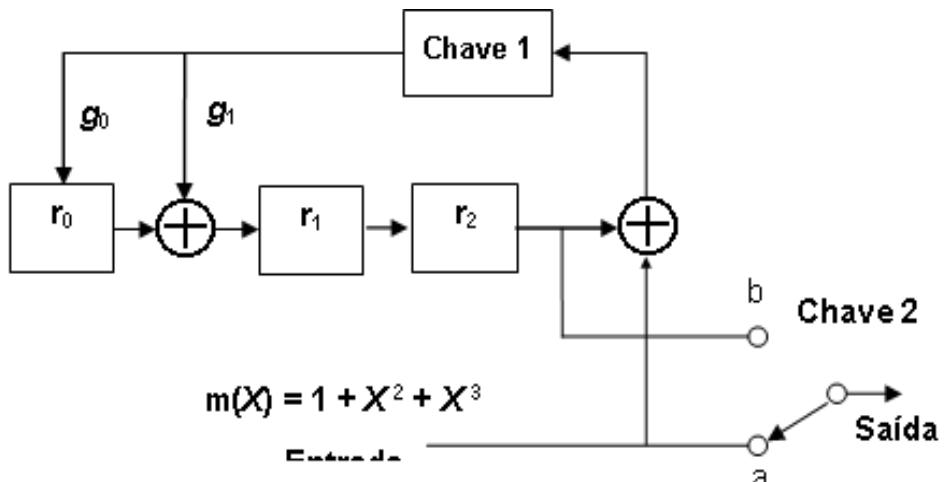


## 2.5. CÓDIGOS CÍCLICOS

### 2.5.2. CODIFICAÇÃO SISTEMÁTICA DE CÓDIGOS CÍCLICOS COM REGISTRADORES DE DESLOCAMENTO DE $(n-k)$ ESTÁGIOS

#### EXEMPLO 2.20

Seja o código (7, 4) cujo polinômio gerador é  $g(X) = 1 + X + X^3$ . Para o vetor mensagem  $m = 1011$ , o polinômio código resultante é  $c(X) = 1 + X^3 + X^5 + X^6$ , que corresponde ao vetor código  $c = 1001011$ . Mostre a formação e transmissão deste vetor código utilizando o circuito da Figura a seguir.



Fila de entrada	Número de deslocamentos	Conteúdo dos registradores	Saída
1011	0	000	-
101	1	110	1
10	2	101	1
1	3	100	0
-	4	100	1

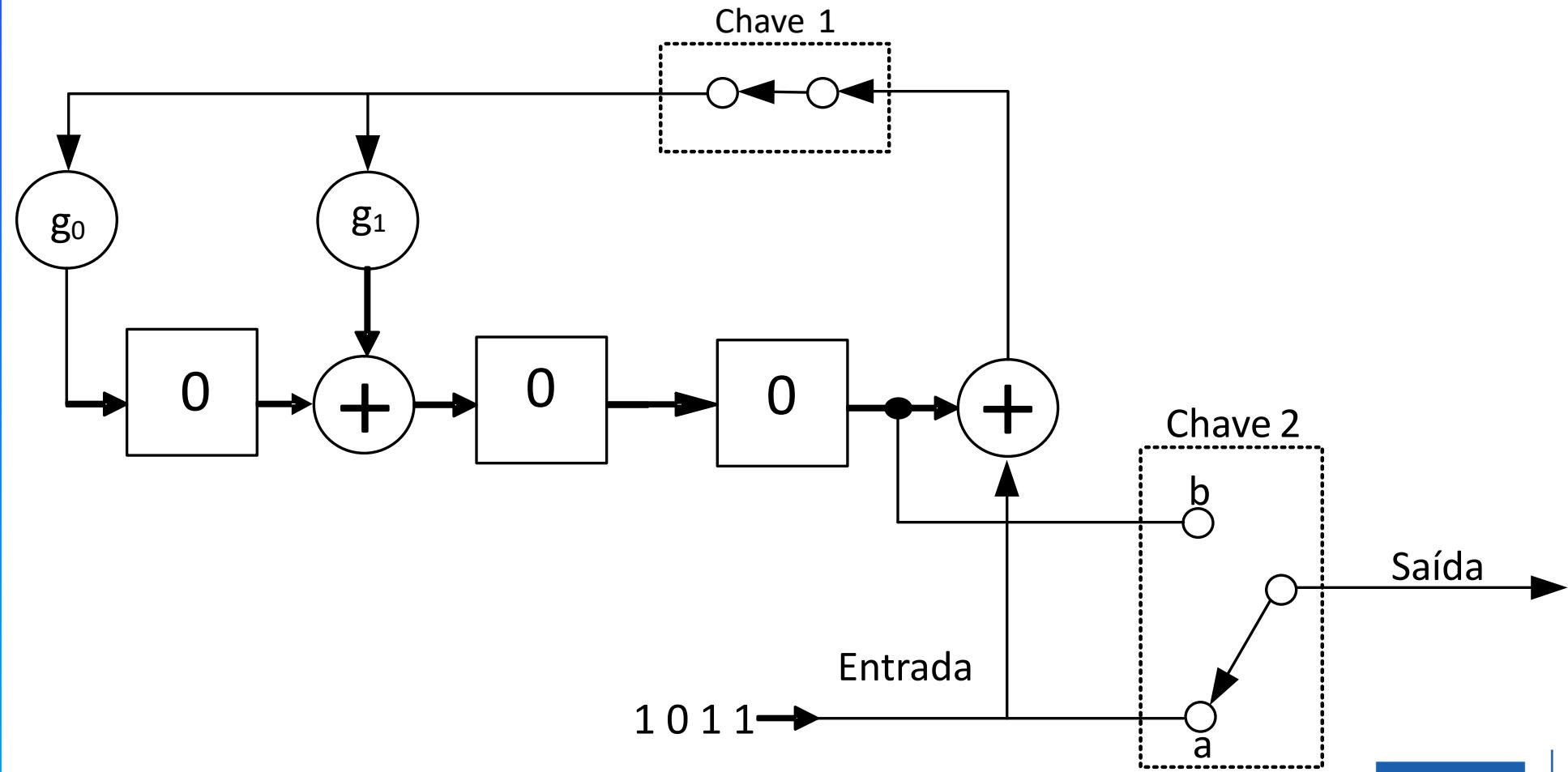
Após os 4 deslocamentos a chave 1 é aberta, a chave 2 passa para a posição b e o conteúdo dos registros (paridade) é transmitido. Logo, o vetor transmitido é:  $c = 1001011$ .

\* \* \*

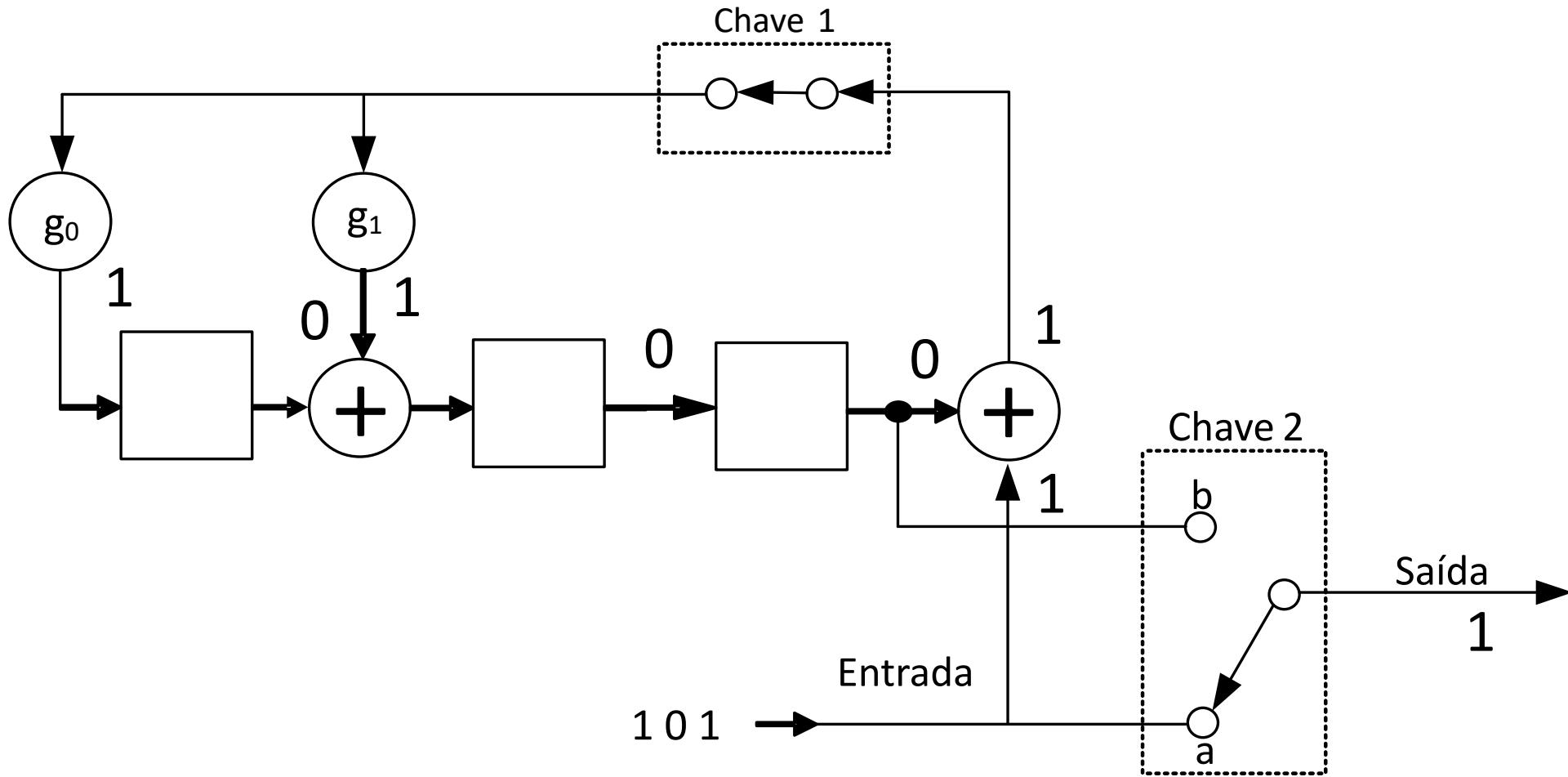
## CODIFICAÇÃO DE CANAL: CÓDIGOS DE BLOCO LINEARES



Fila de entrada	Número de deslocamentos	Conteúdo dos registradores	Saída
1011	0	000	-



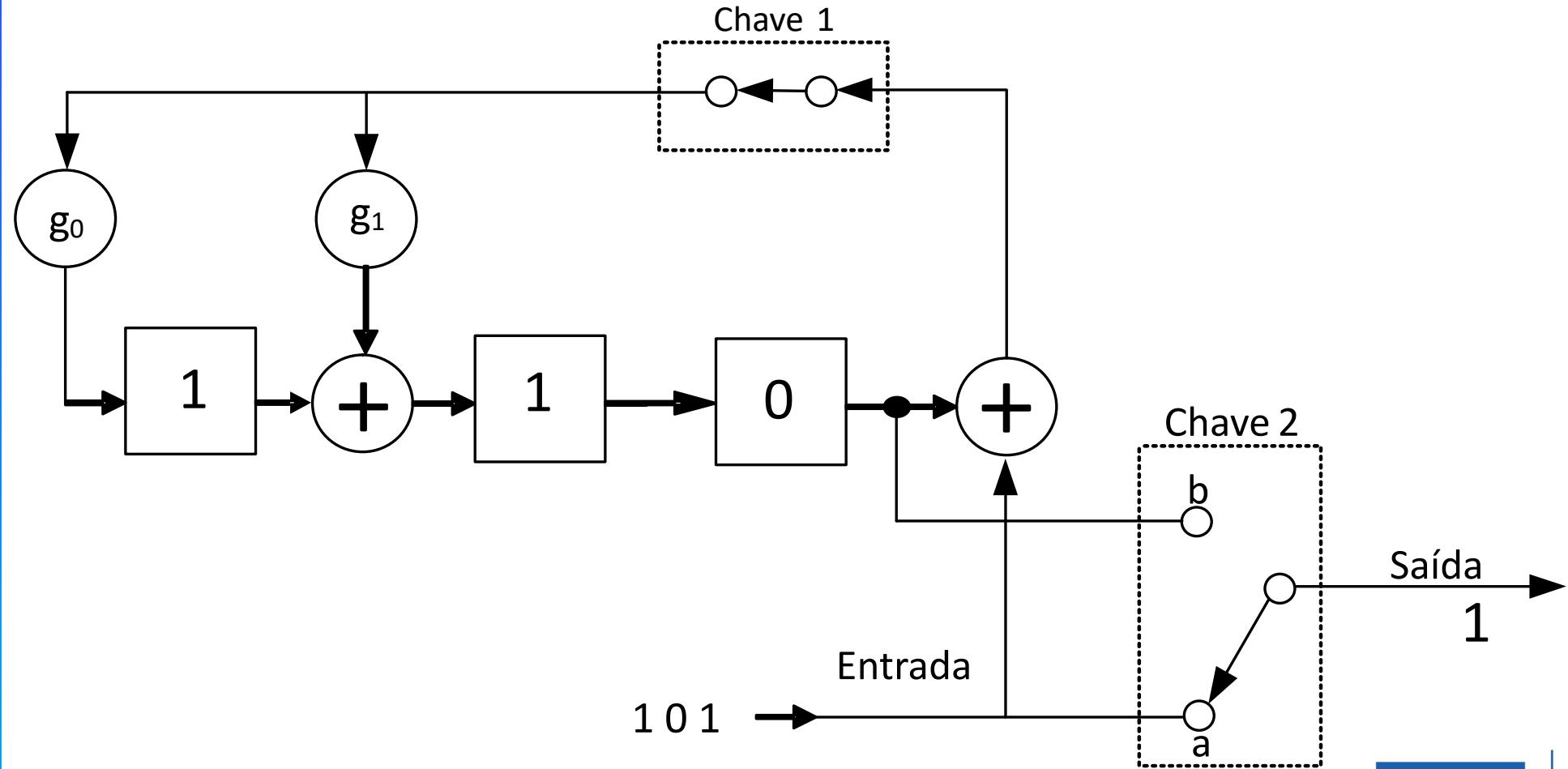
## CODIFICAÇÃO DE CANAL: CÓDIGOS DE BLOCO LINEARES

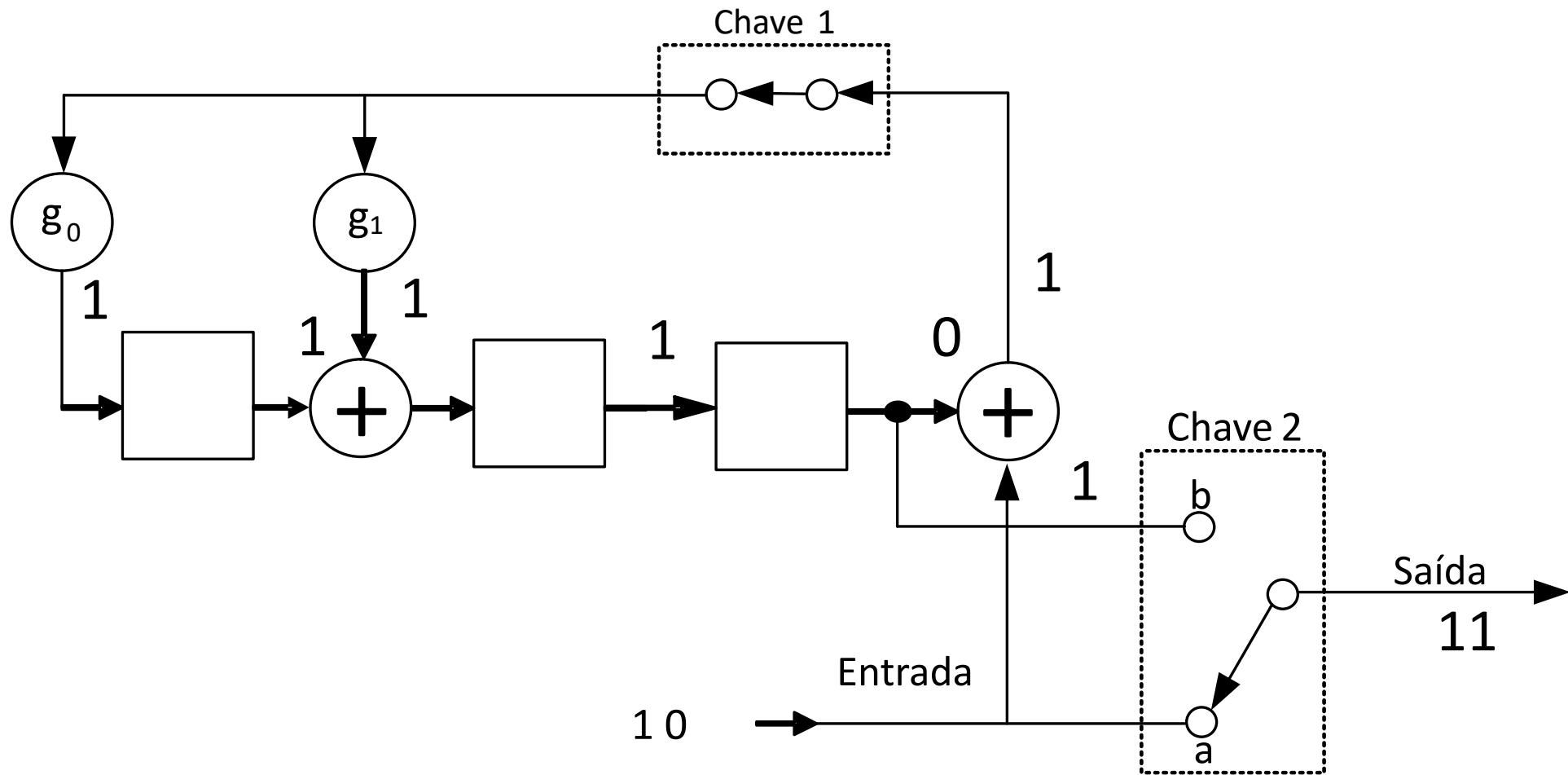


## CODIFICAÇÃO DE CANAL: CÓDIGOS DE BLOCO LINEARES



Fila de entrada	Número de deslocamentos	Conteúdo dos registradores	Saída
101	1	110	1

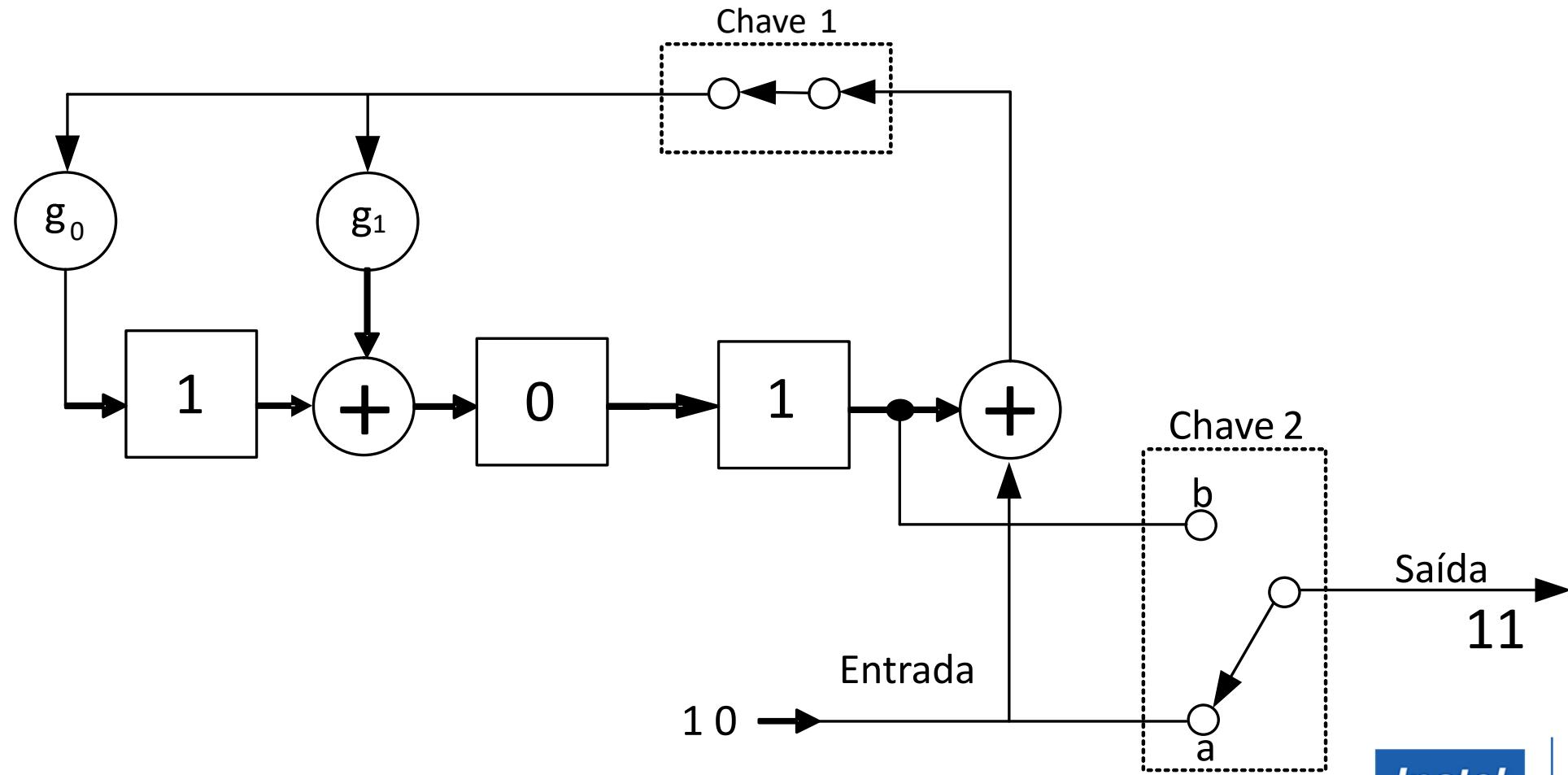


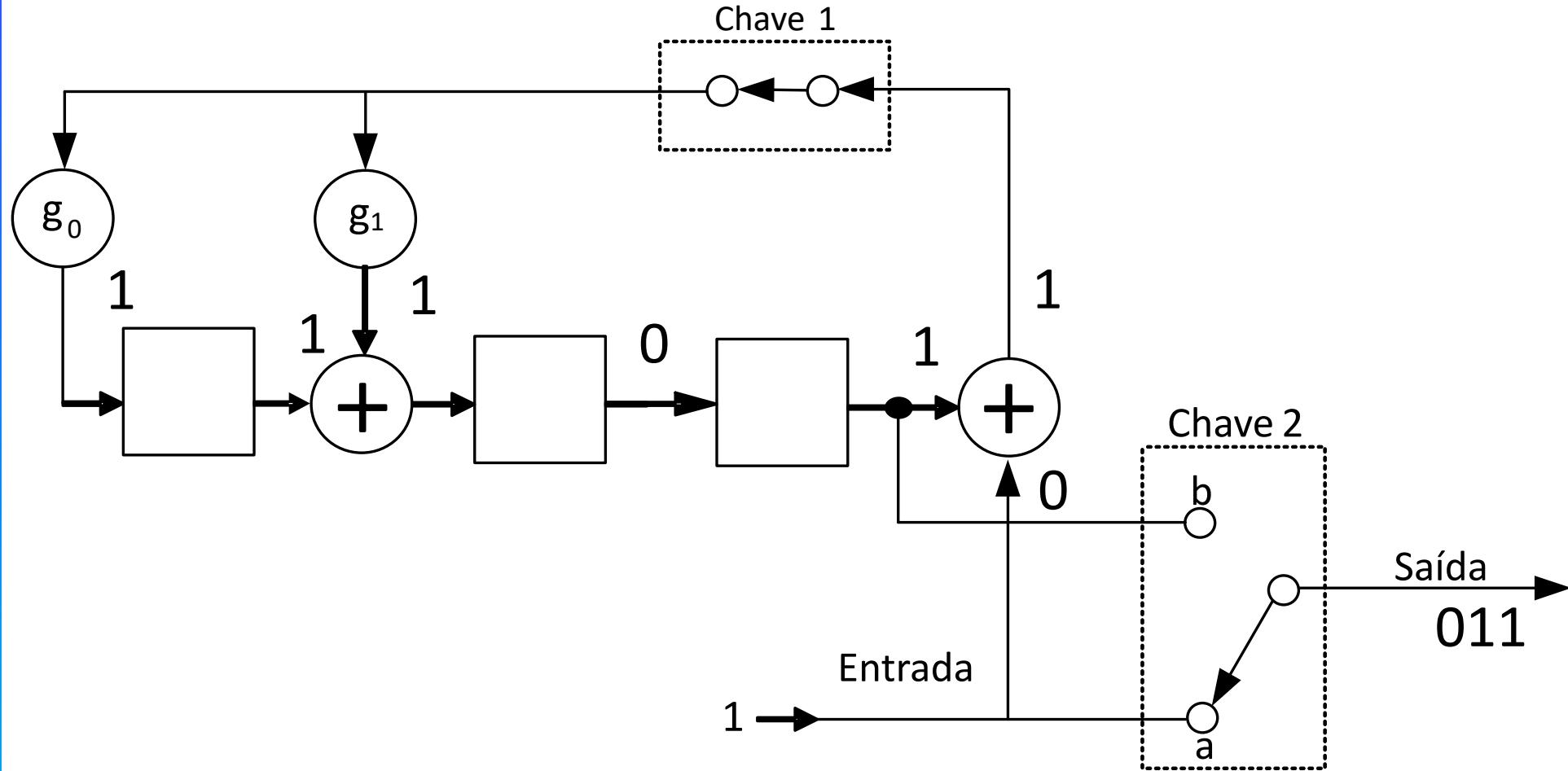


## CODIFICAÇÃO DE CANAL: CÓDIGOS DE BLOCO LINEARES



Fila de entrada	Número de deslocamentos	Conteúdo dos registradores	Saída
10	2	101	11

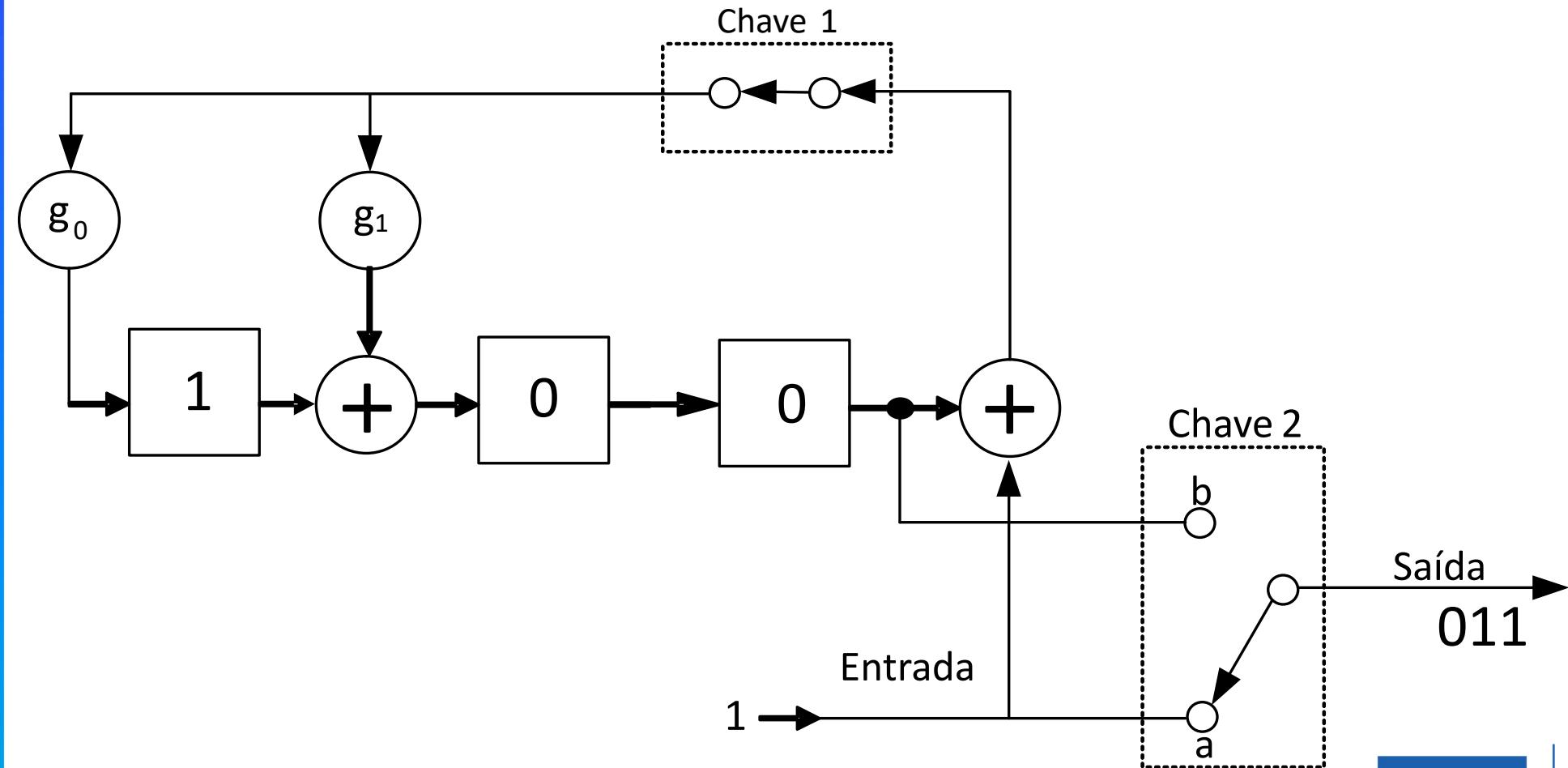




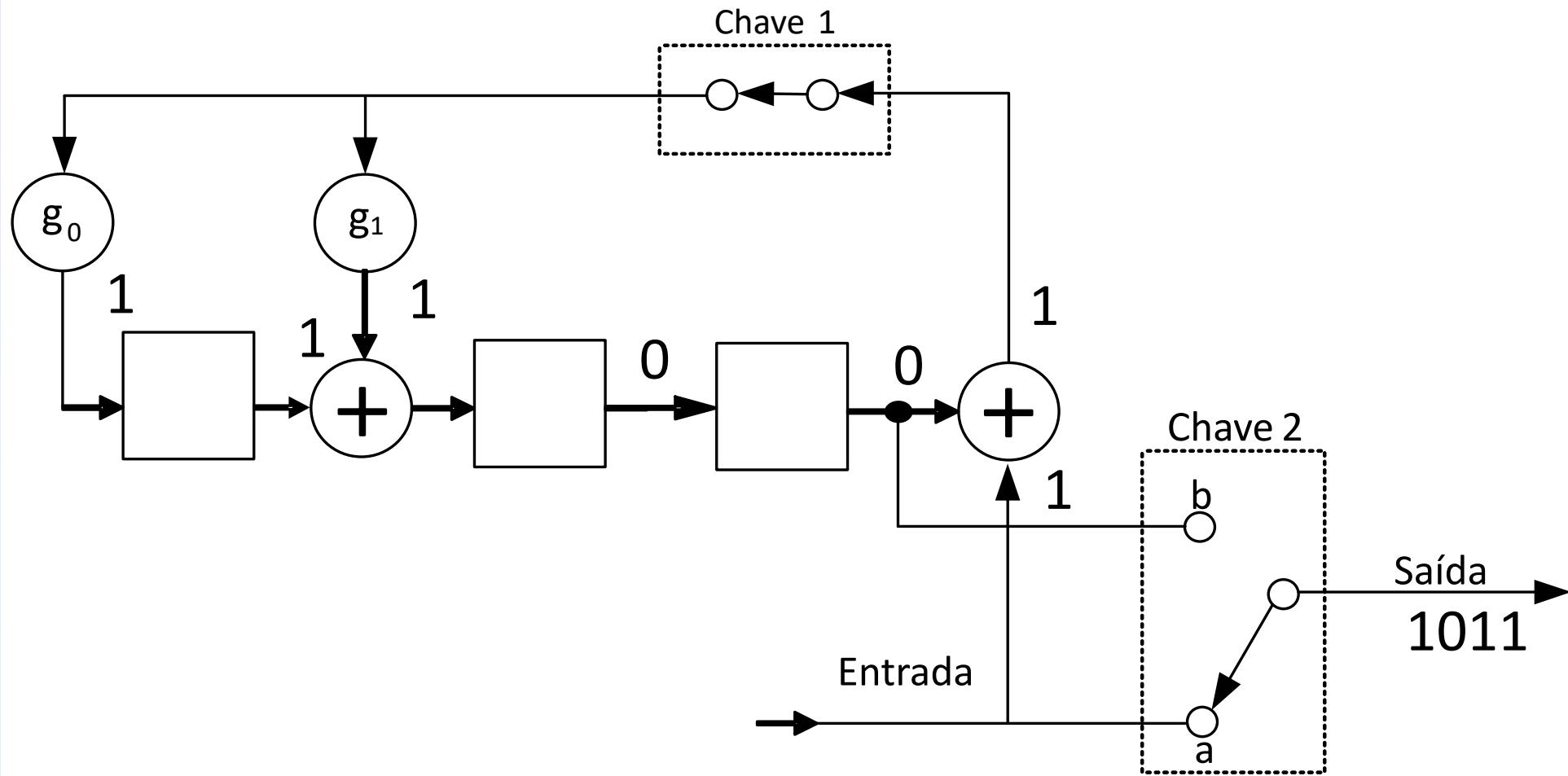
## CODIFICAÇÃO DE CANAL: CÓDIGOS DE BLOCO LINEARES



Fila de entrada	Número de deslocamentos	Conteúdo dos registradores	Saída
1	3	100	011



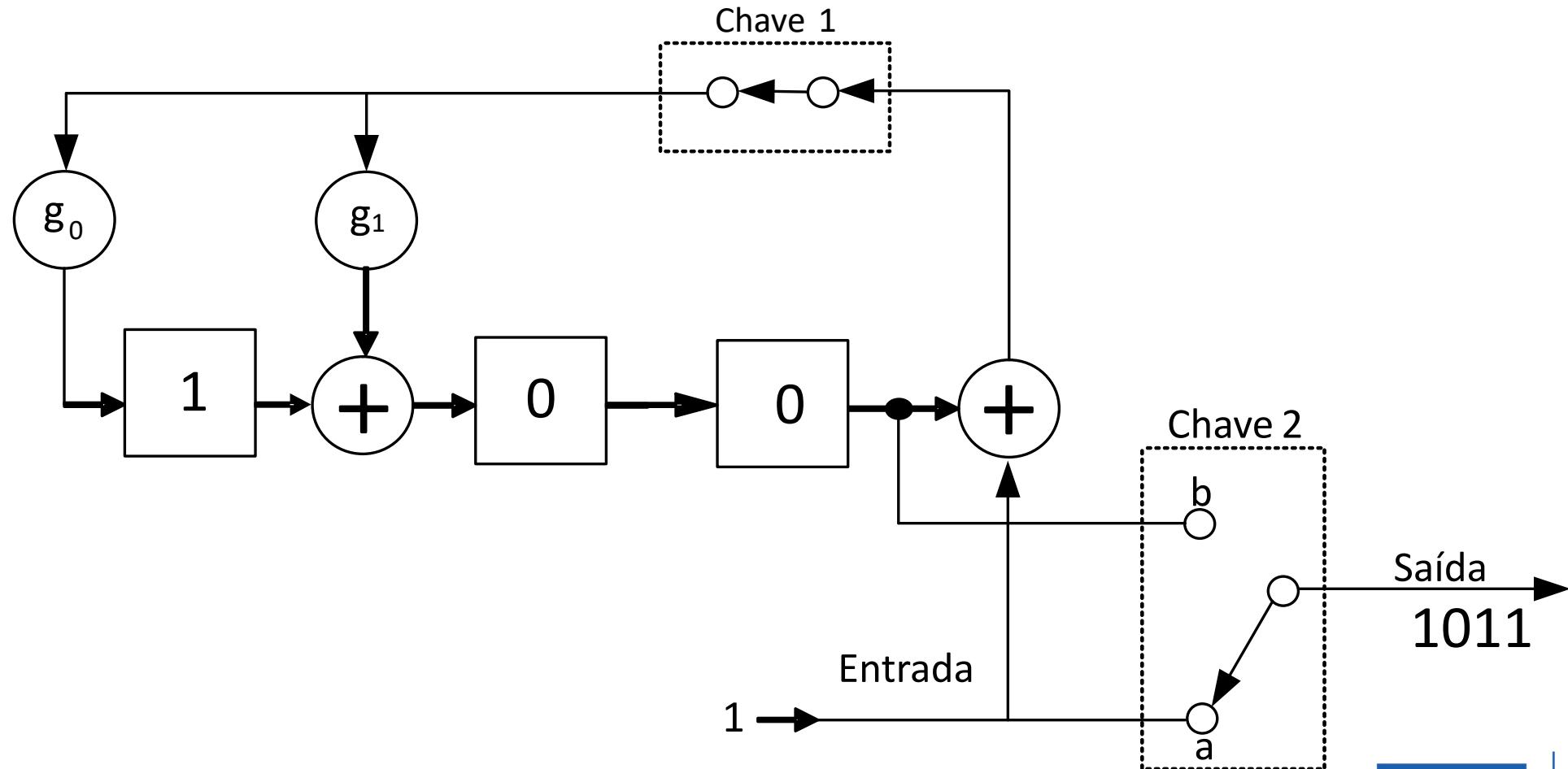
## CODIFICAÇÃO DE CANAL: CÓDIGOS DE BLOCO LINEARES

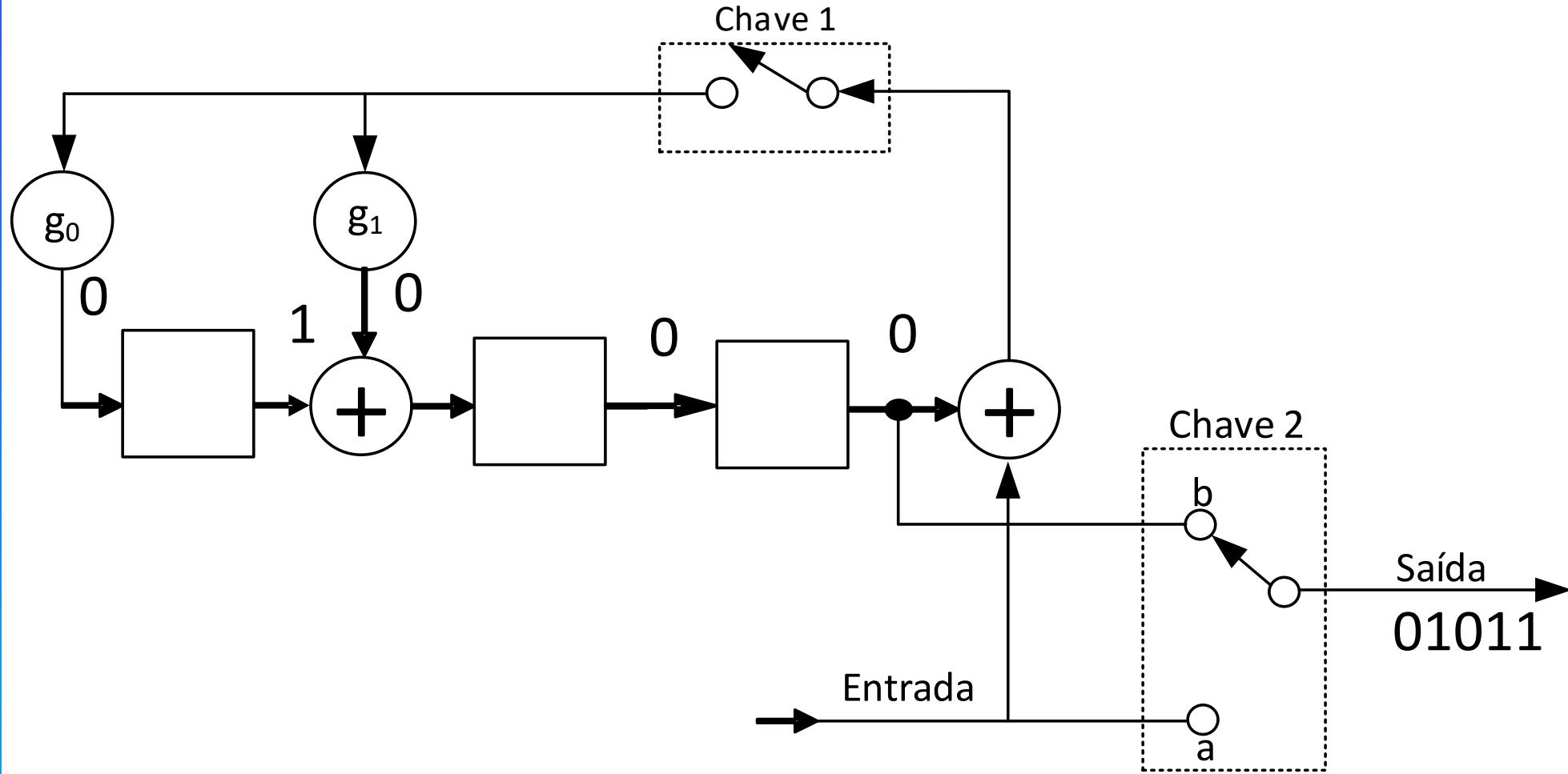


## CODIFICAÇÃO DE CANAL: CÓDIGOS DE BLOCO LINEARES



Fila de entrada	Número de deslocamentos	Conteúdo dos registradores	Saída
-	4	100	1011

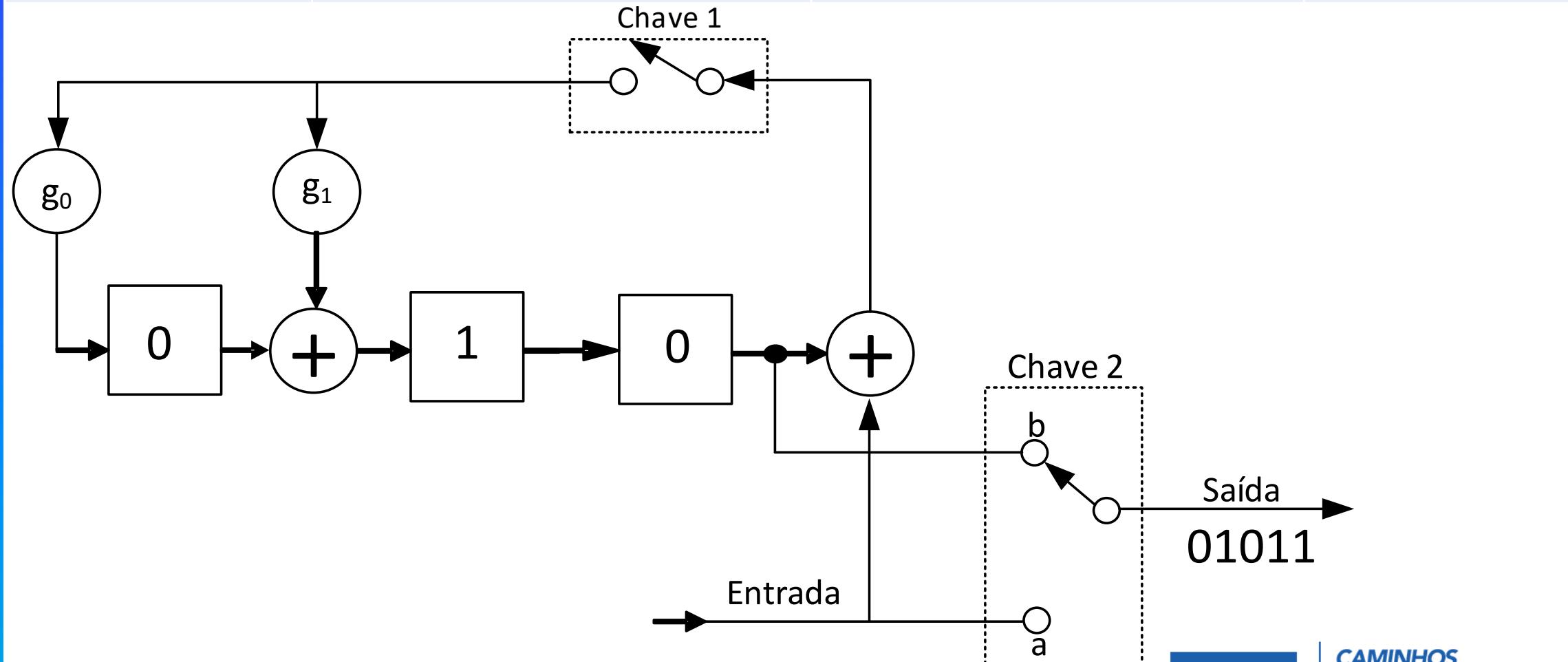




## CODIFICAÇÃO DE CANAL: CÓDIGOS DE BLOCO LINEARES



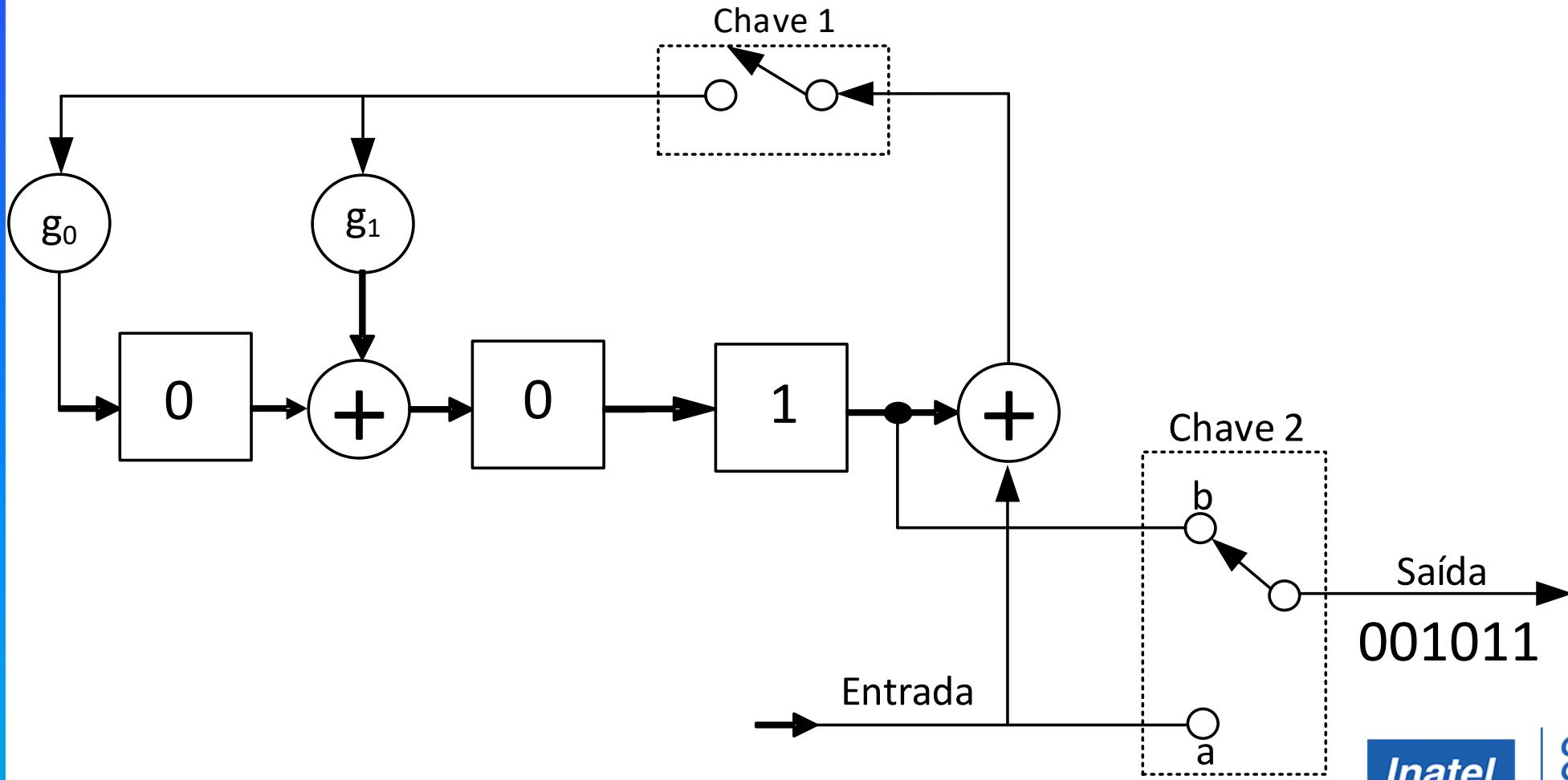
Fila de entrada	Número de deslocamentos	Conteúdo dos registradores	Saída
-	5	010	01011



## CODIFICAÇÃO DE CANAL: CÓDIGOS DE BLOCO LINEARES



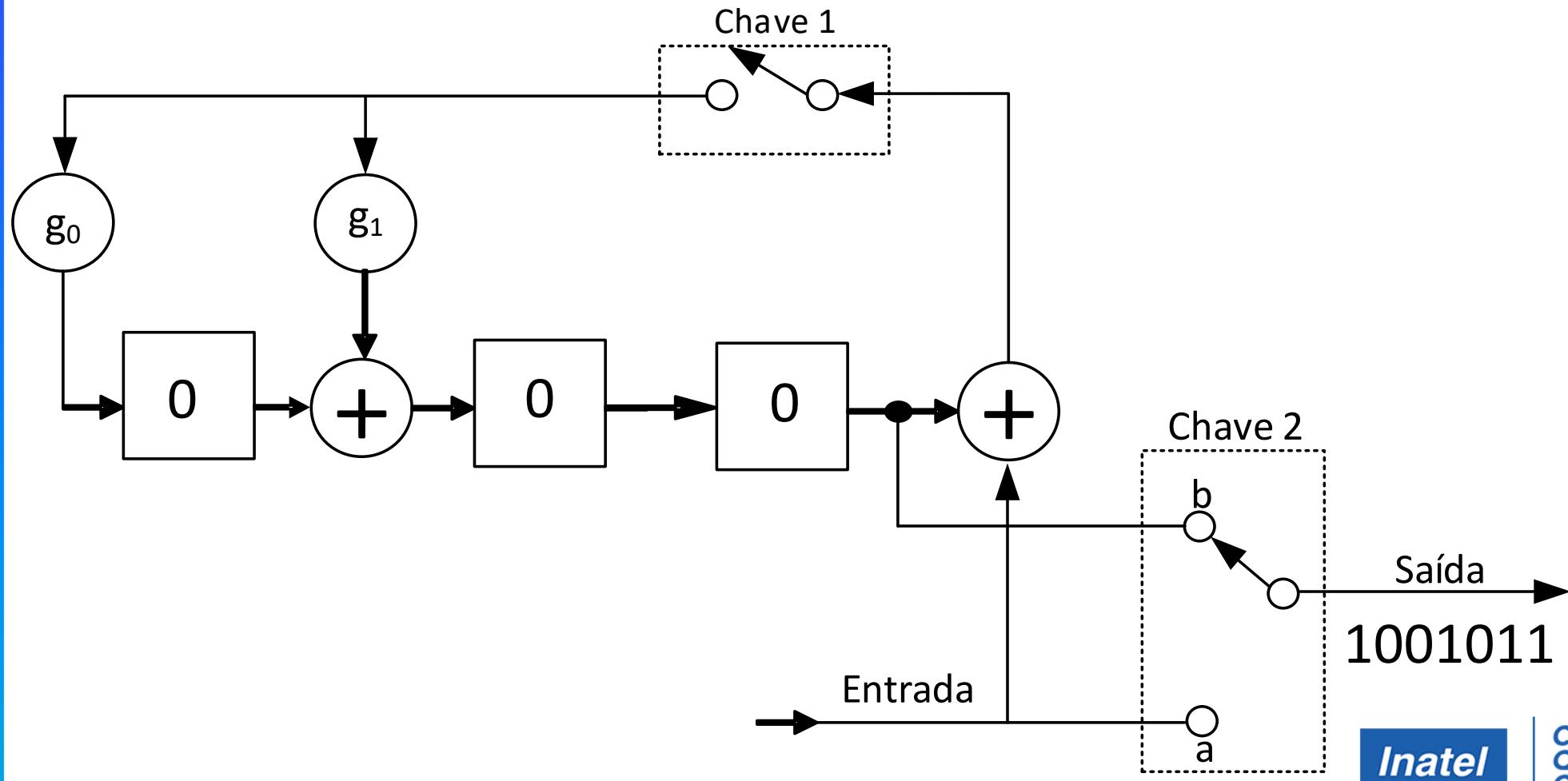
Fila de entrada	Número de deslocamentos	Conteúdo dos registradores	Saída
-	6	001	001011



## CODIFICAÇÃO DE CANAL: CÓDIGOS DE BLOCO LINEARES



Fila de entrada	Número de deslocamentos	Conteúdo dos registradores	Saída
-	7	000	1001011





## 2.5. CÓDIGOS CÍCLICOS

### 2.5.3. DETECÇÃO DE ERROS DE CÓDIGOS CÍCLICOS SISTEMÁTICOS COM REGISTRADORES DE DESLOCAMENTO DE $(n-k)$ ESTÁGIOS

- A função do decodificador é recuperar o vetor código transmitido a partir do vetor recebido pela presença de ruído.

$$r(X) = r_0 + r_1 X + r_2 X^2 + \cdots + r_{n-1} X^{n-1}$$

- O decodificador testa se o vetor recebido é um vetor código, o que equivale a dividir o polinômio recebido pelo polinômio gerador, pois

$$r(X) = q(X)g(X) + s(X)$$

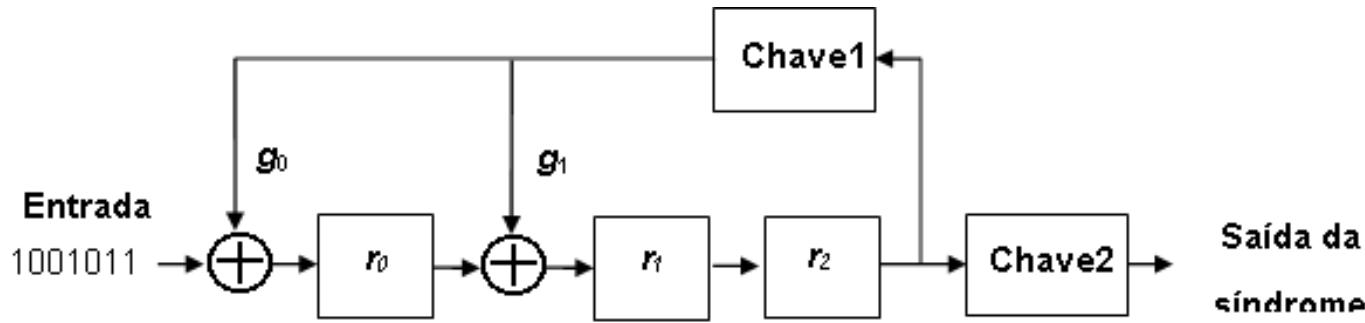
- Se a síndrome for zero, o vetor recebido é aceito como um vetor código, caso contrário, tem-se uma detecção de erro através da síndrome.



## 2.5. CÓDIGOS CÍCLICOS

### 2.5.3. DETECÇÃO DE ERROS DE CÓDIGOS CÍCLICOS SISTEMÁTICOS COM REGISTRADORES DE DESLOCAMENTO DE $(n-k)$ ESTÁGIOS

O circuito para a determinação da síndrome é semelhante ao utilizado para a codificação, conforme mostrado a seguir.



Procedimento:

Passo 1: A chave 1 é inicialmente fechada e a chave 2 aberta. O vetor recebido é deslocado pela entrada dos registradores, cujos estados iniciais são todos zero. Após o vetor recebido estar todo nos registradores, seu conteúdo é a síndrome.

Passo 2: A chave 1 é então aberta e a chave 2 fechada, de forma a permitir que o vetor síndrome possa se deslocado para fora dos registradores.



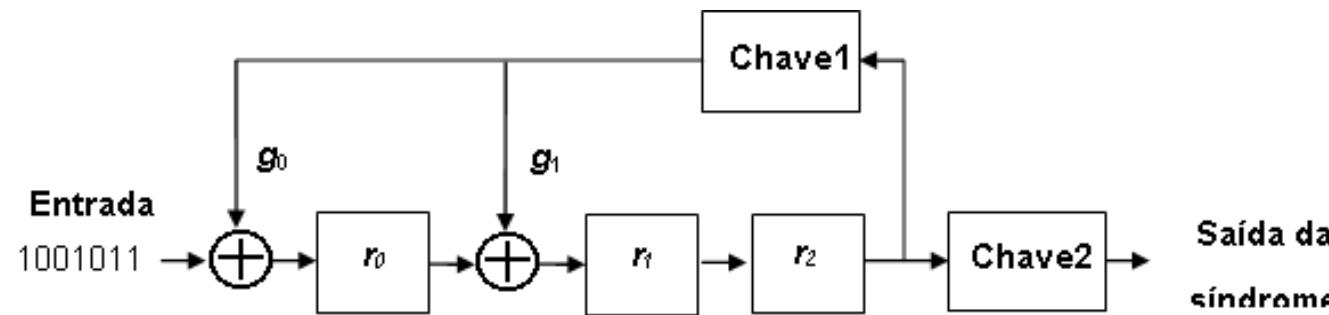
## 2.5. CÓDIGOS CÍCLICOS

### 2.5.3. DETECÇÃO DE ERROS DE CÓDIGOS CÍCLICOS SISTEMÁTICOS COM REGISTRADORES DE DESLOCAMENTO DE $(n-k)$ ESTÁGIOS

#### EXEMPLO 2.21

Determinar a síndrome do vetor  $r = 1001011$ , codificado na forma sistemática a partir do polinômio gerador  $g(X) = 1 + X + X^3$  utilizando registradores de deslocamento. Mostre a formação da síndrome a cada deslocamento.

O conteúdo dos registradores a cada deslocamento é apresentado a seguir.



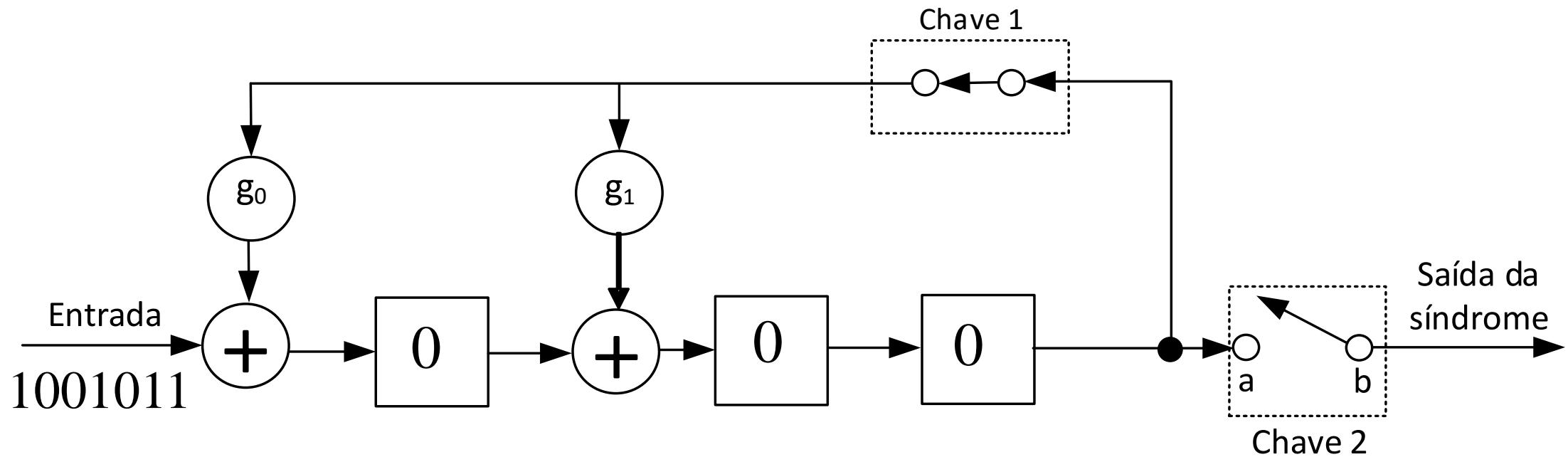
\* \* \*

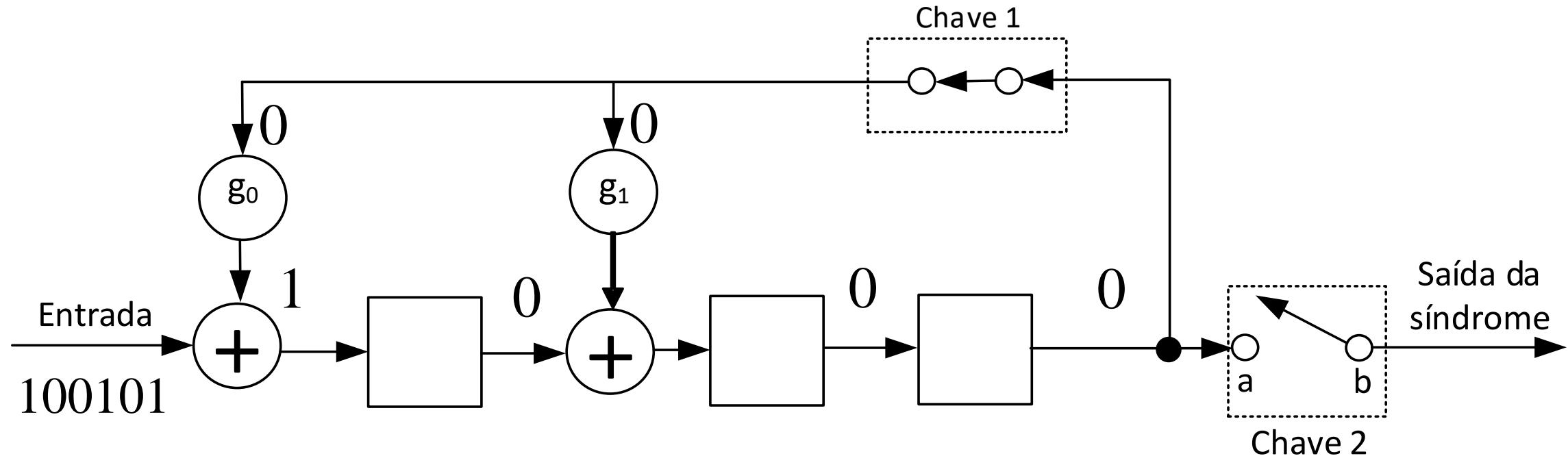
Fila de entrada	Número de deslocamentos	Conteúdo dos registradores
1001011	0	000
100101	1	100
10010	2	110
1001	3	011
100	4	011
10	5	111
1	6	101
-	7	000

## CODIFICAÇÃO DE CANAL: CÓDIGOS DE BLOCO LINEARES



Fila de entrada	Número de deslocamentos	Conteúdo dos registradores
1001011	0	000

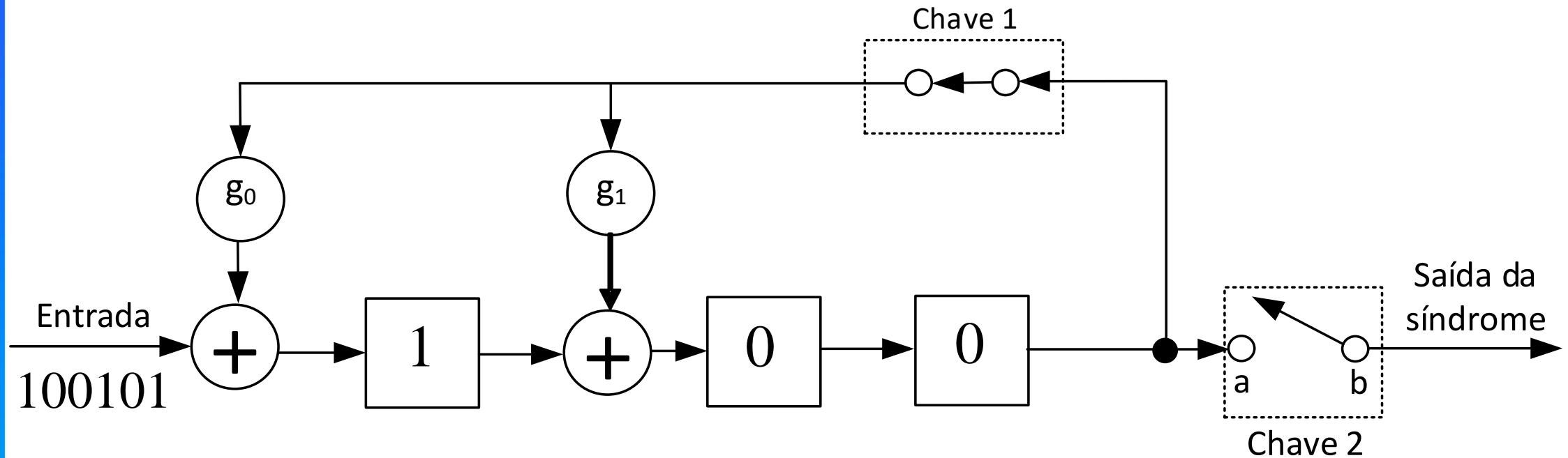


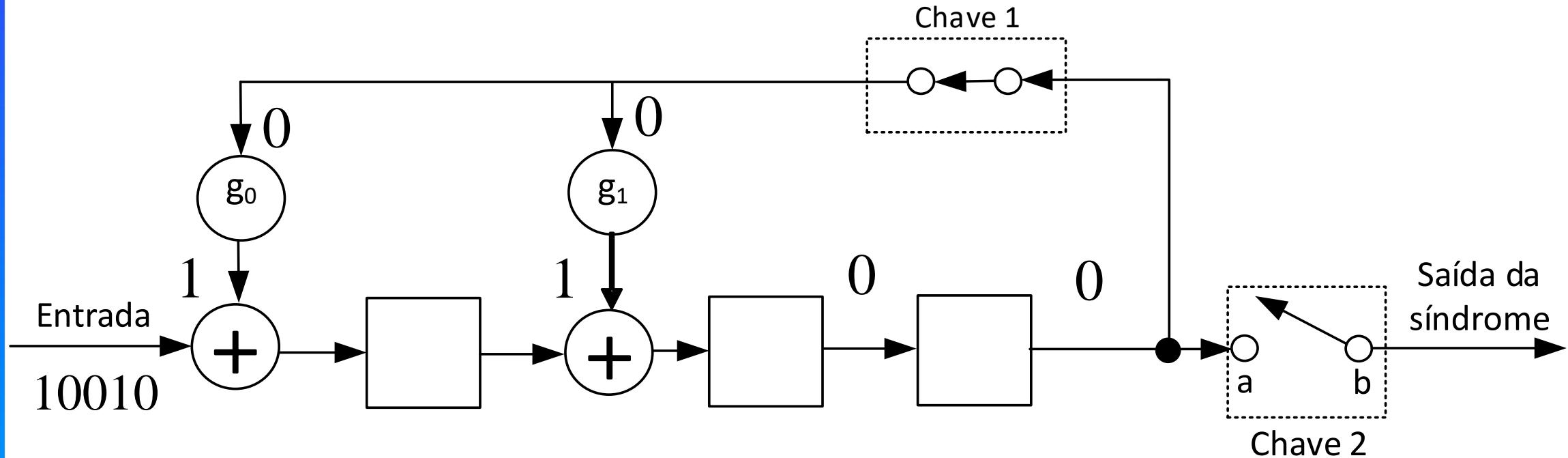


## CODIFICAÇÃO DE CANAL: CÓDIGOS DE BLOCO LINEARES



Fila de entrada	Número de deslocamentos	Conteúdo dos registradores
100101	1	100

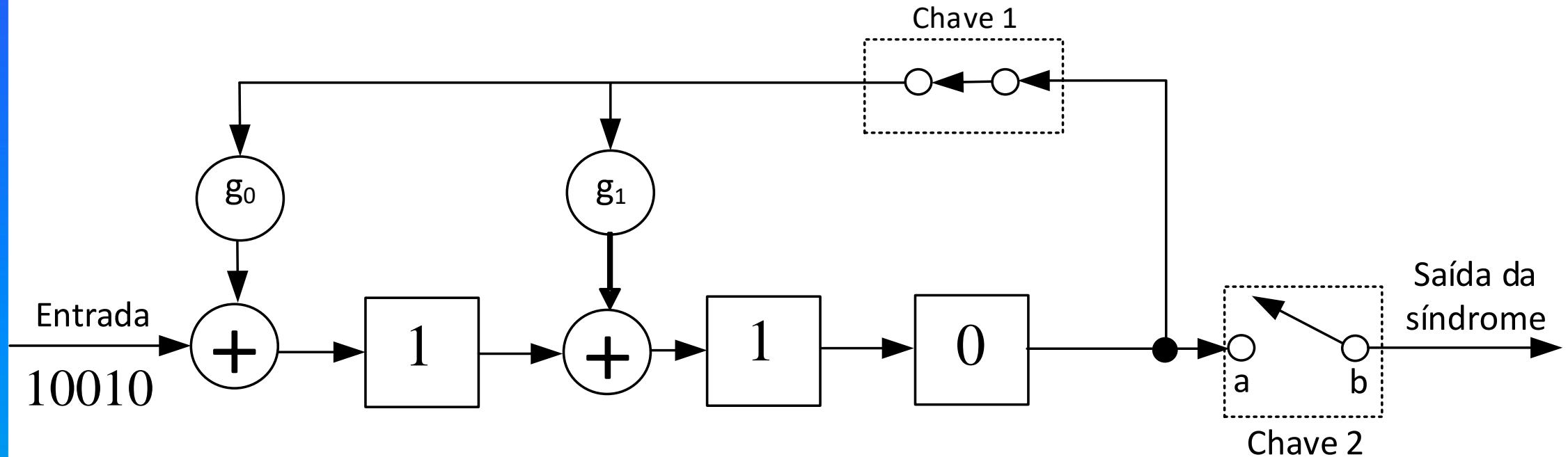


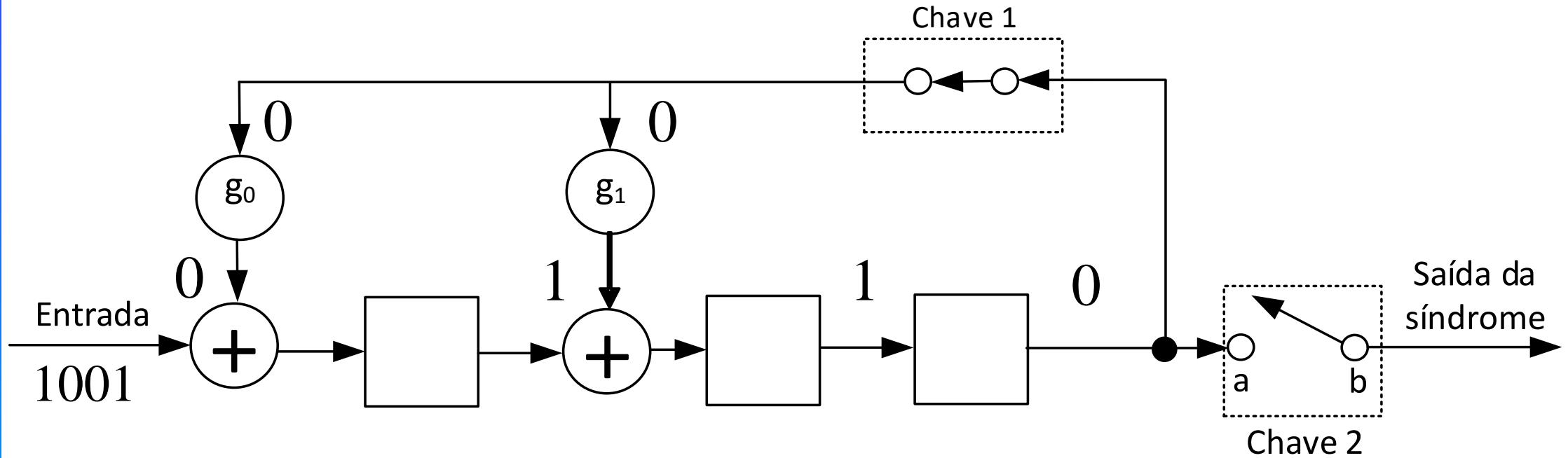


## CODIFICAÇÃO DE CANAL: CÓDIGOS DE BLOCO LINEARES



Fila de entrada	Número de deslocamentos	Conteúdo dos registradores
10010	2	110

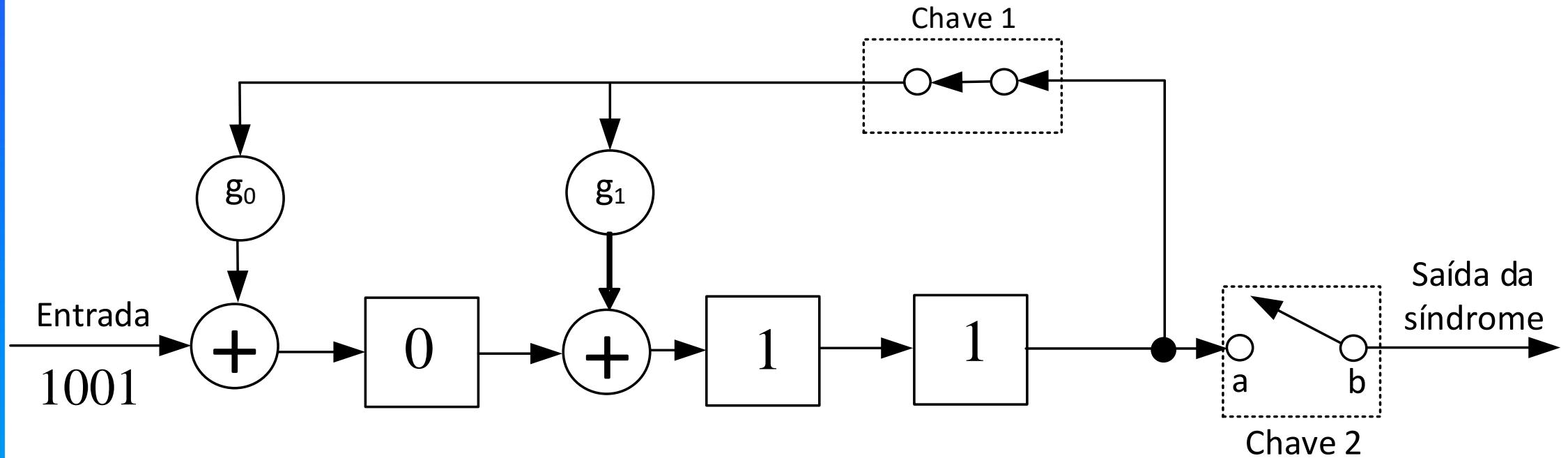


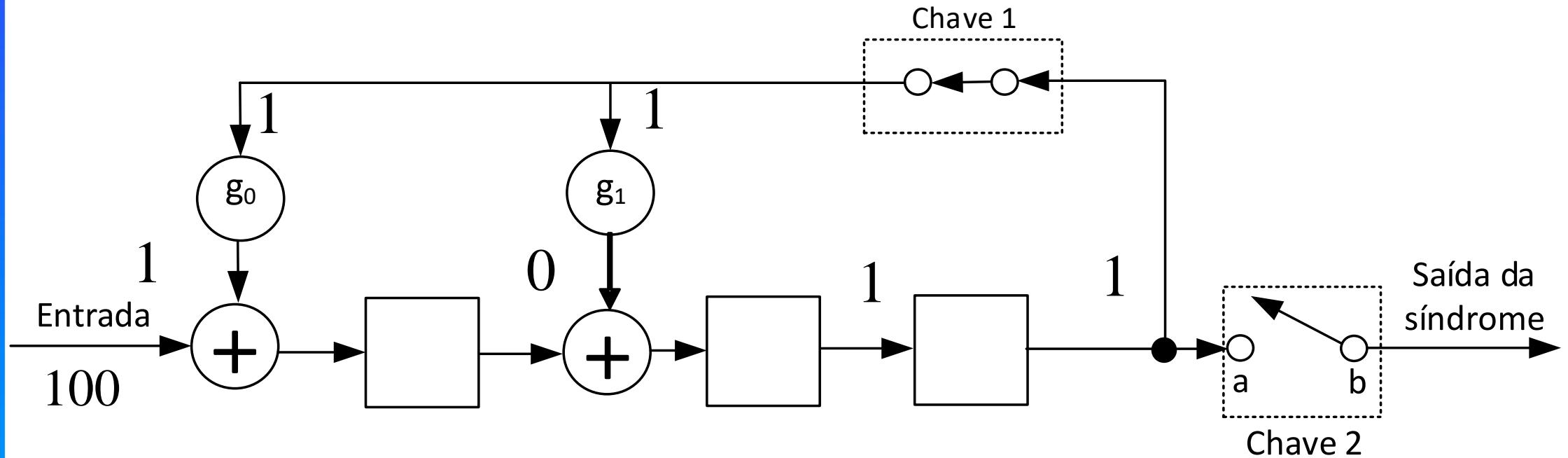


## CODIFICAÇÃO DE CANAL: CÓDIGOS DE BLOCO LINEARES



Fila de entrada	Número de deslocamentos	Conteúdo dos registradores
1001	3	011

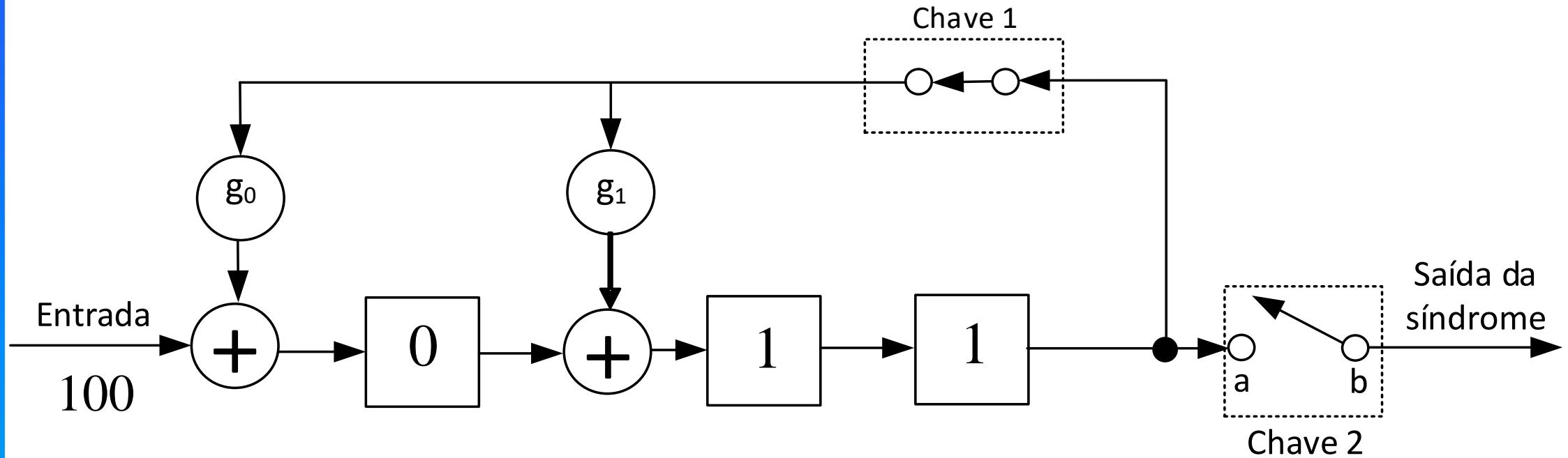


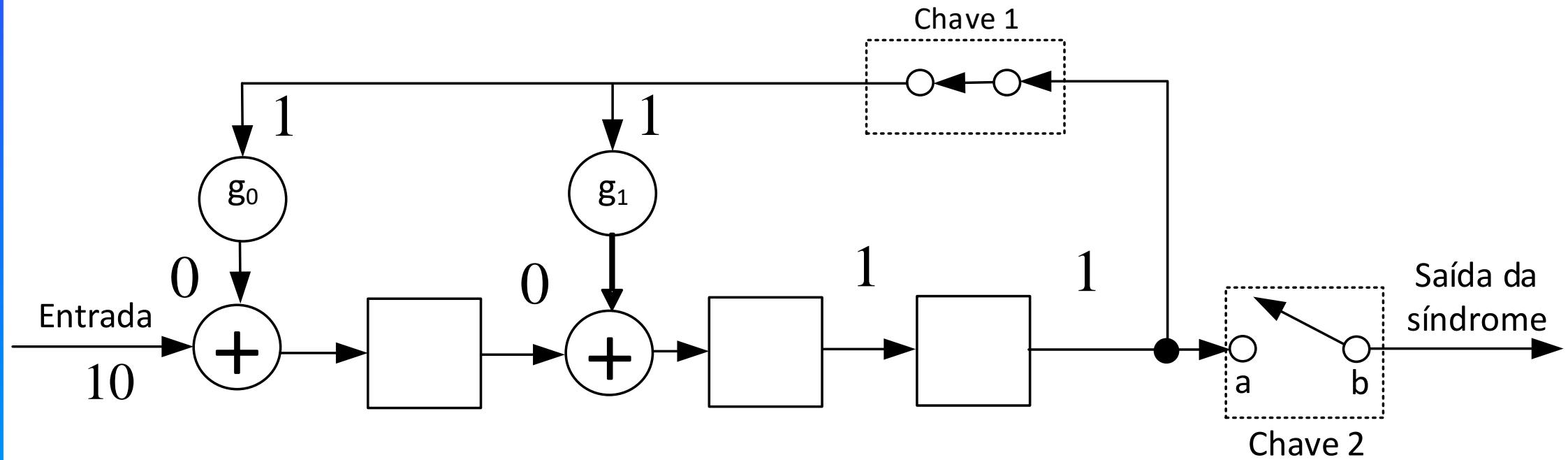


## CODIFICAÇÃO DE CANAL: CÓDIGOS DE BLOCO LINEARES



Fila de entrada	Número de deslocamentos	Conteúdo dos registradores
100	4	011

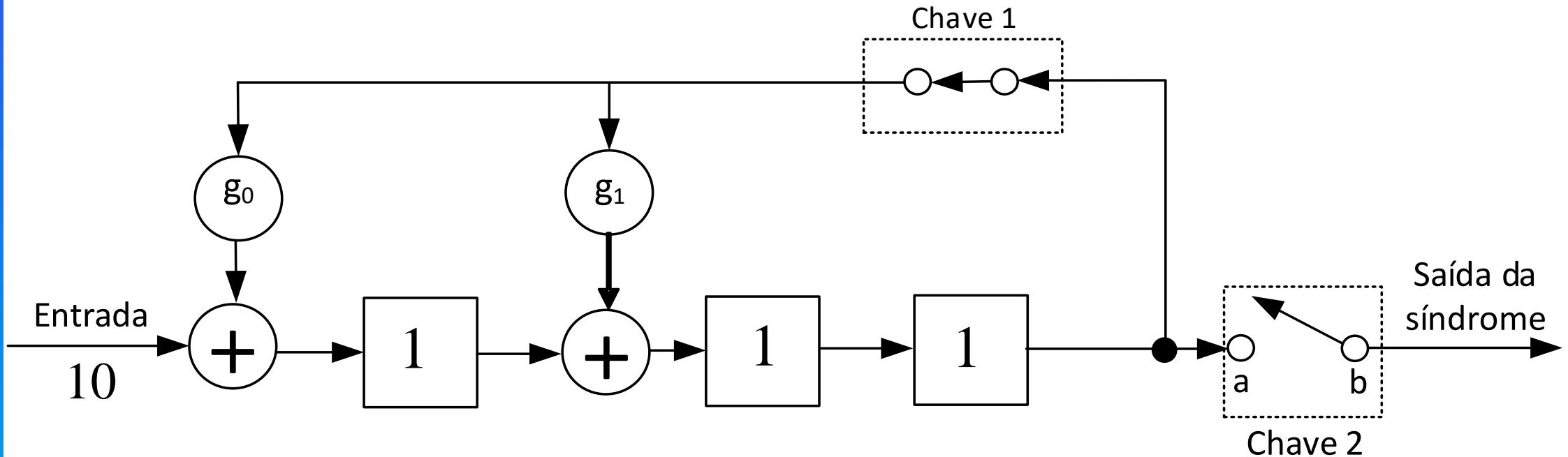


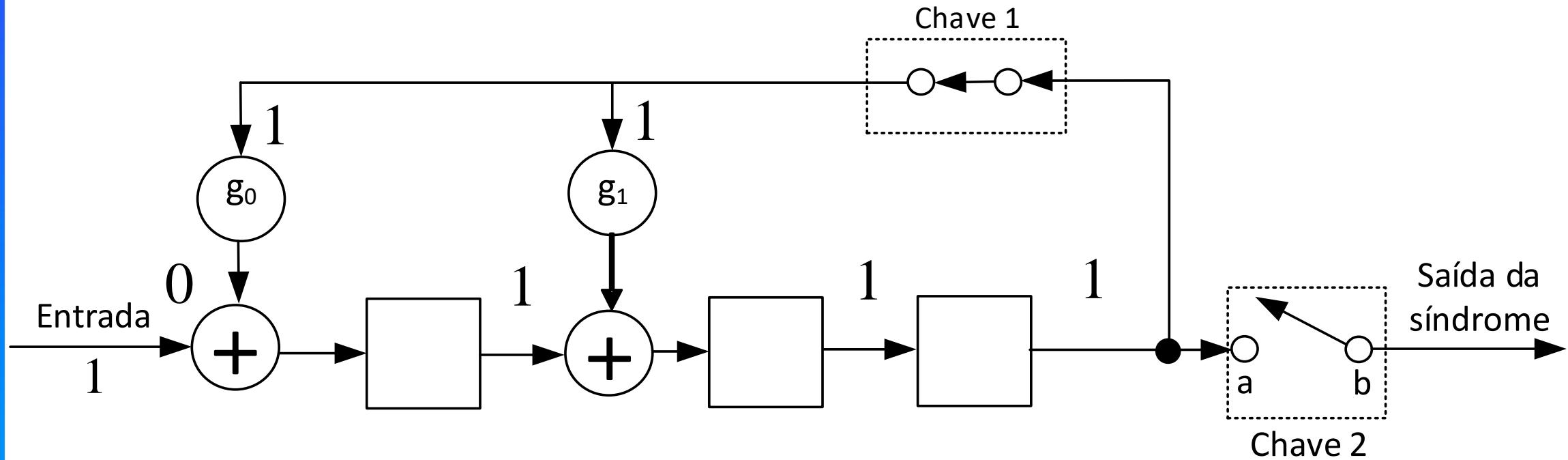


## CODIFICAÇÃO DE CANAL: CÓDIGOS DE BLOCO LINEARES



Fila de entrada	Número de deslocamentos	Conteúdo dos registradores
10	5	111

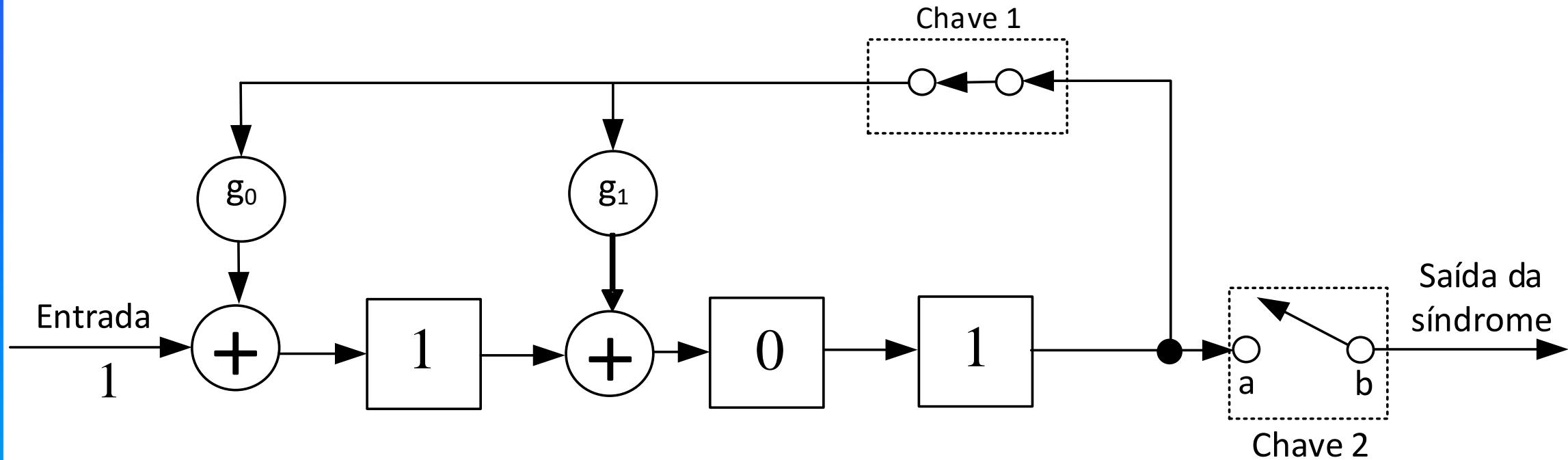


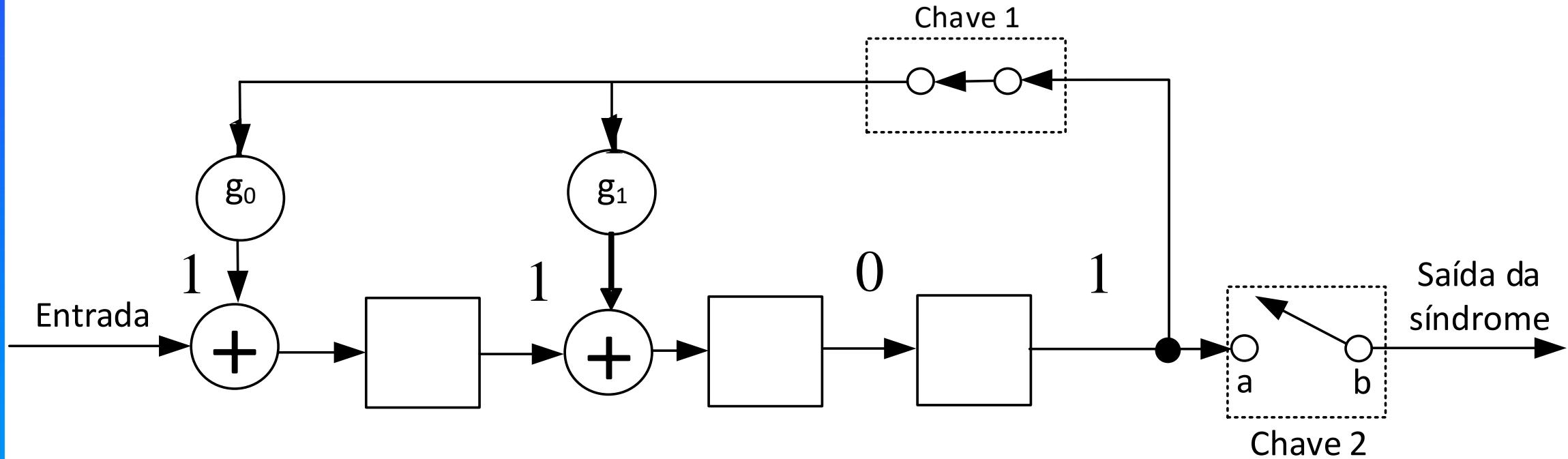


## CODIFICAÇÃO DE CANAL: CÓDIGOS DE BLOCO LINEARES



Fila de entrada	Número de deslocamentos	Conteúdo dos registradores
1	6	101





## CODIFICAÇÃO DE CANAL: CÓDIGOS DE BLOCO LINEARES



Fila de entrada	Número de deslocamentos	Conteúdo dos registradores
-	7	000

