

Sistemas de Codificação

Circuitos Digitais I

Prof. Fernando Passold

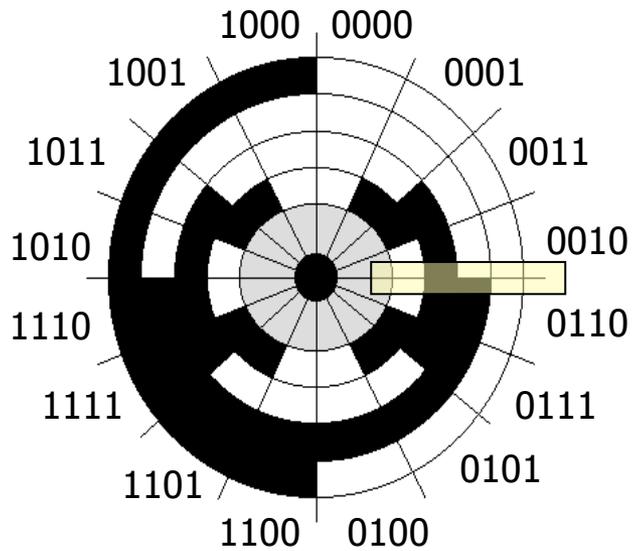
Ascii.cpp (gera tabela ASCII):

```
// Tabela ASCII
// Fernando Passsold, 05/09/2001

#include <stdio.h>
// #include <stdlib.h>
void main() {
    int codigo = 32, coluna=1, linha=1, key, aux;

    for (codigo=32; codigo<256; codigo++) {
        printf("%3d: ",codigo);
        fputc(char(codigo));
        coluna++;
        if (coluna<11){
            printf(" ");
        }
        else {
            printf("\n");
            coluna=1;
            linha++;
            if (linha>24){
                // espera uma tecla ser apertada
                while ( (key = getchar()) != '\n' )
                    printf("%c",key);
                linha=1;
            }
        }
    }
}
```

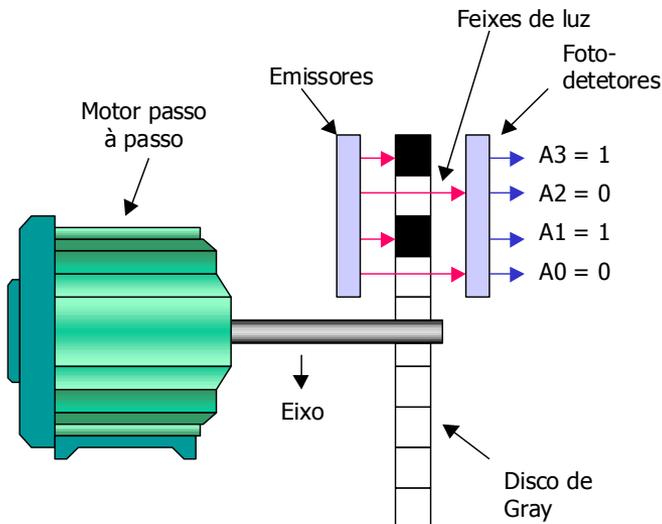
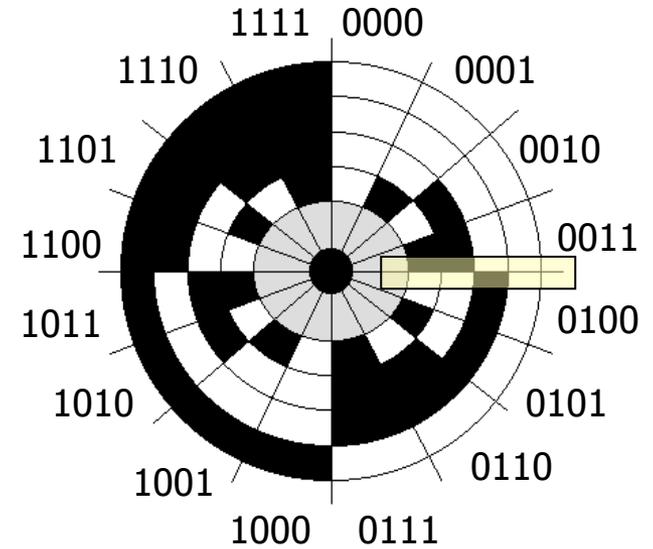
Disco Gray:



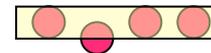
Repare a seqüência:

Ref	Gray	Dec	Bin
0	0000	0	0000
1	0001	1	0001
2	0011	3	0010
3	0010	2	0011
4	0110	6	0100
5	0111	7	0101
6	0101	5	0110
7	0100	6	0111
8	1100	12	1000
9	1101	13	1001
10	1111	15	1010
11	1110	14	1011
12	1010	10	1100
13	1011	11	1101
14	1001	9	1110
15	1000	8	1111

Disco Binário:



Note que no código Gray: entre uma variação e outra de código, somente um bit se altera. Evita erros de leitura se alguns dos foto-detetores se encontra desalinhado frente à seus "companheiros":



Encoders Relativos

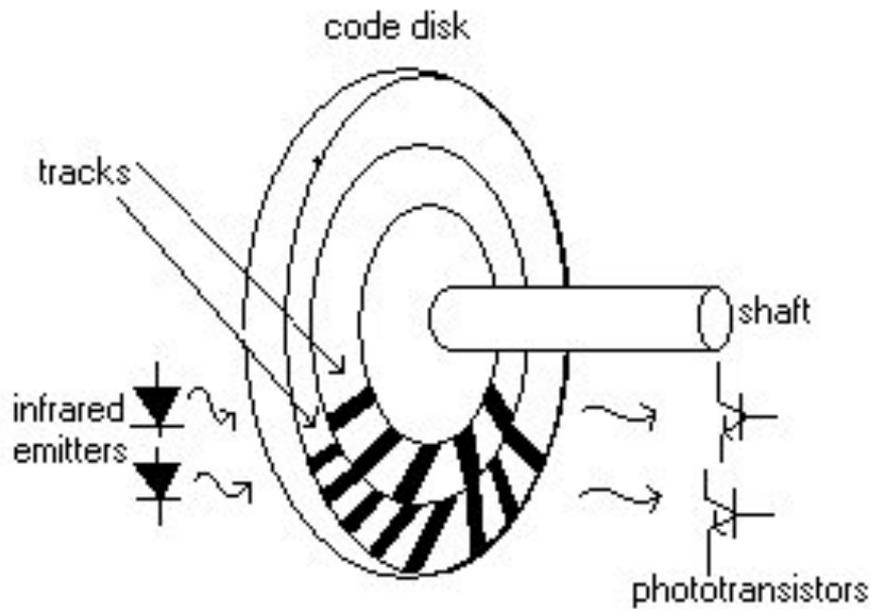


Fig 1. A rotary optical encoder

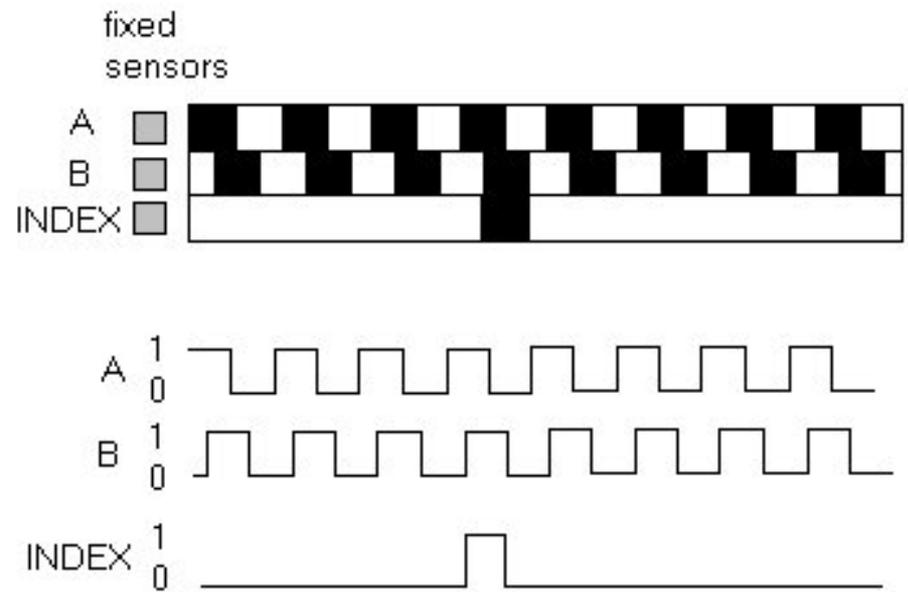


Fig 5. Incremental encoder disk track patterns

Encoders Relativos

(com detecção em quadratura)

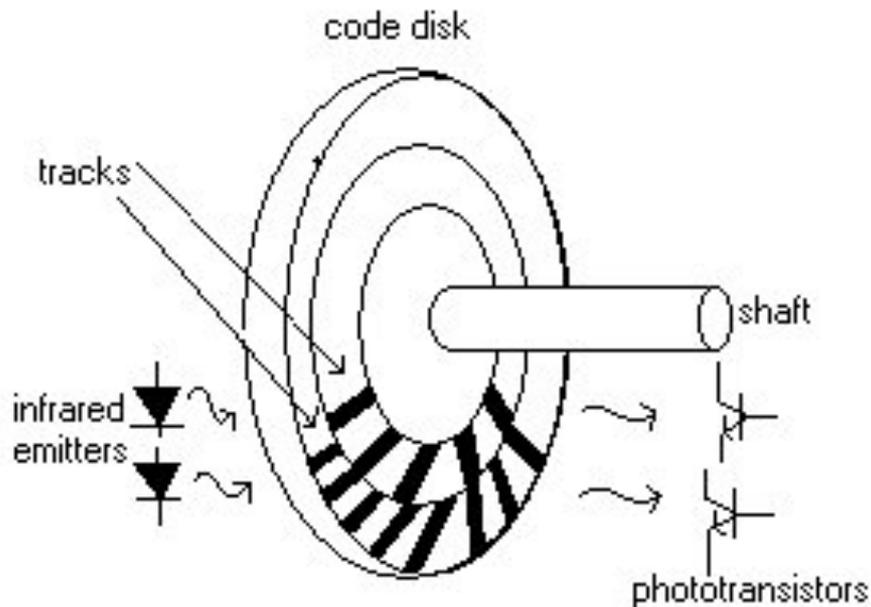


Fig 1. A rotary optical encoder

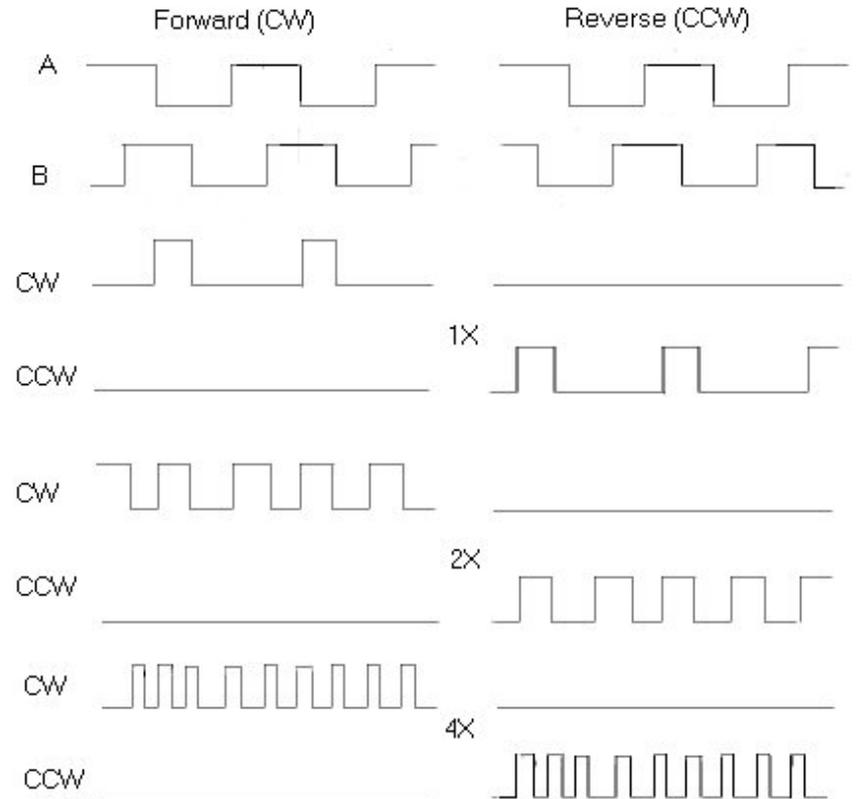


Fig 6. Quadrature direction sensing and resolution enhancement. (CW = clockwise, CCW= counter-clockwise)

Código Gray:: Encoders Absolutos

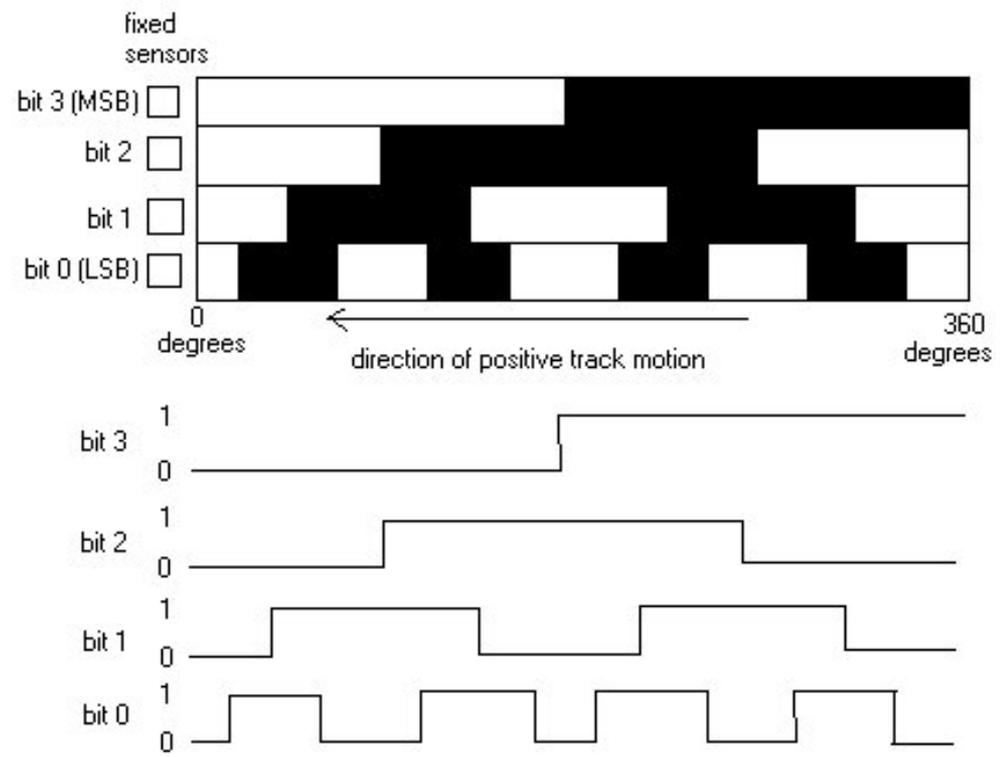
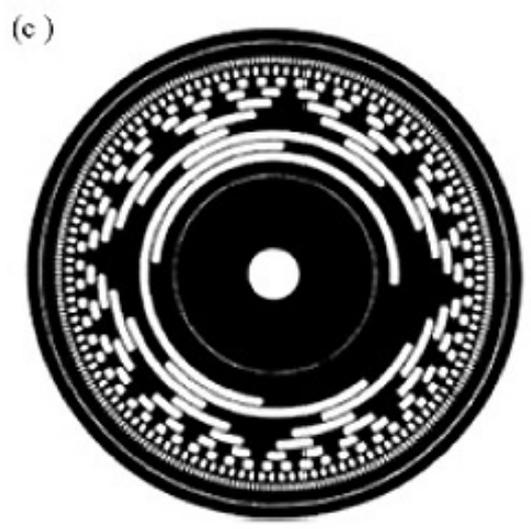
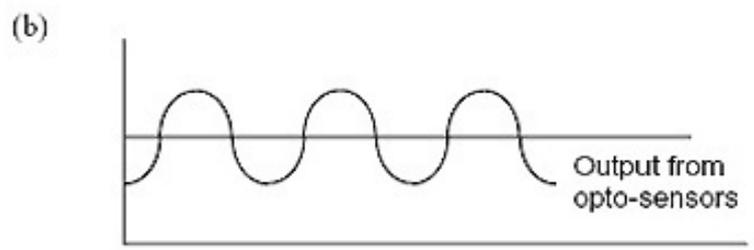
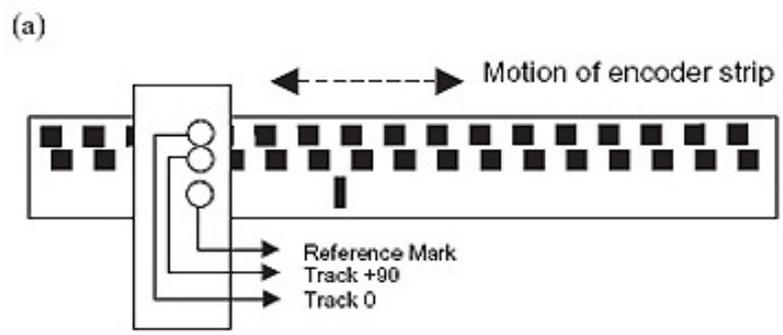
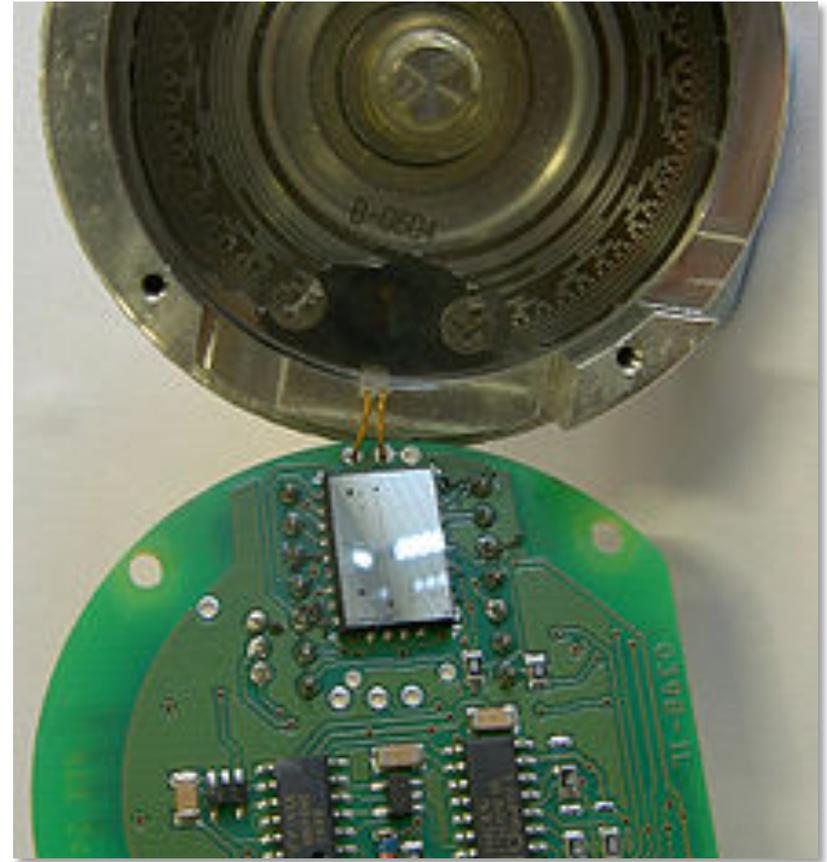
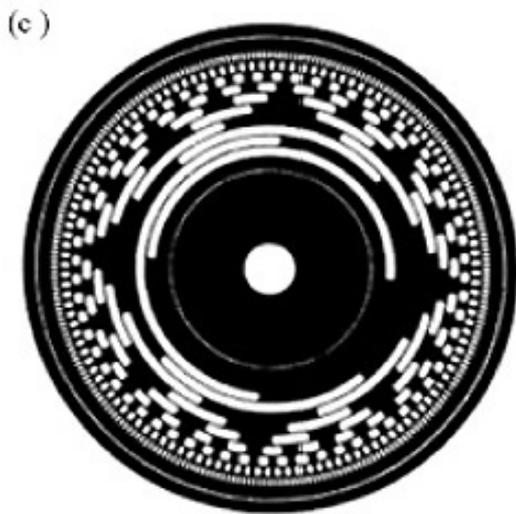
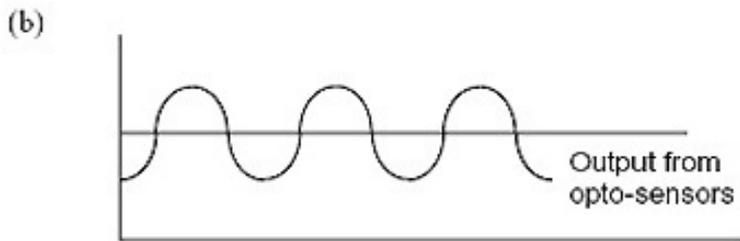
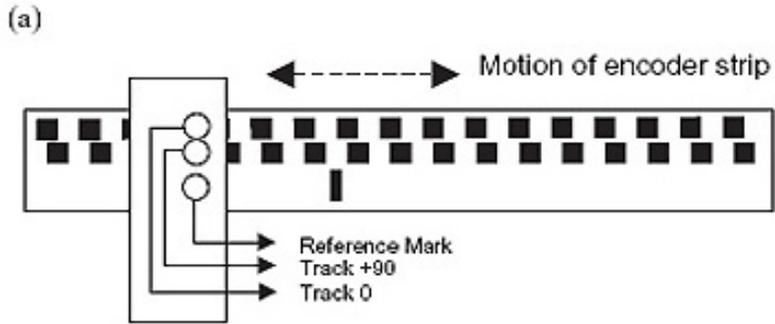


Fig 2. 4-Bit gray code absolute encoder disk track patterns

Código Gray:: Encoders Absolutos



Código Binário:: Não é Encoder Absoluto!

Seq. **Gray** (4-bits)

×

Seq. **Binária** (4-bits)

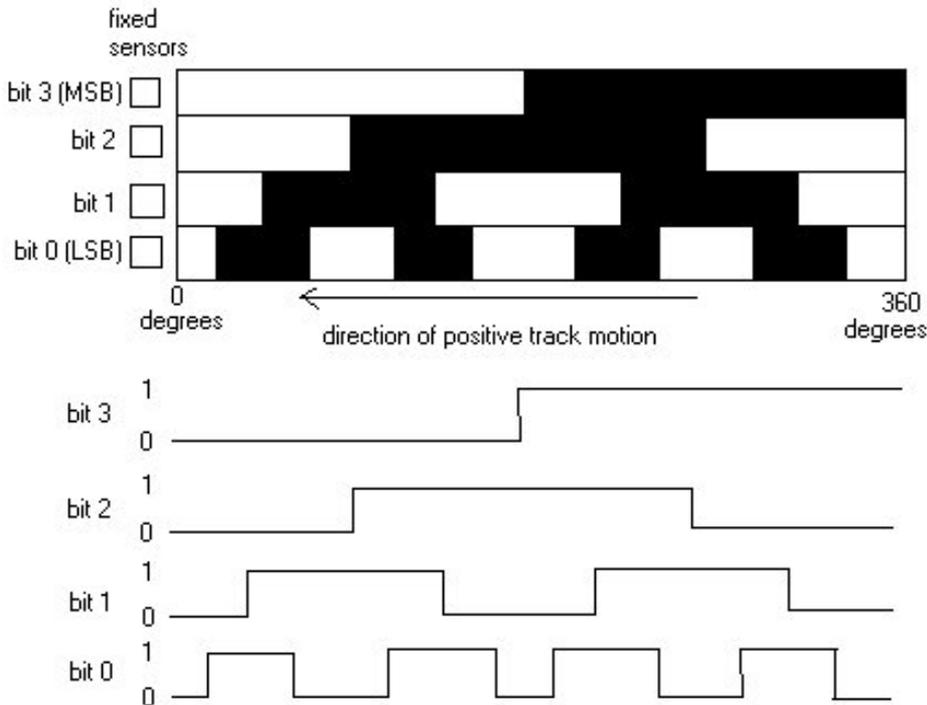


Fig 2. 4-Bit gray code absolute encoder disk track patterns

Menos erros
A cada passo: 1 bit apenas varia de estado!

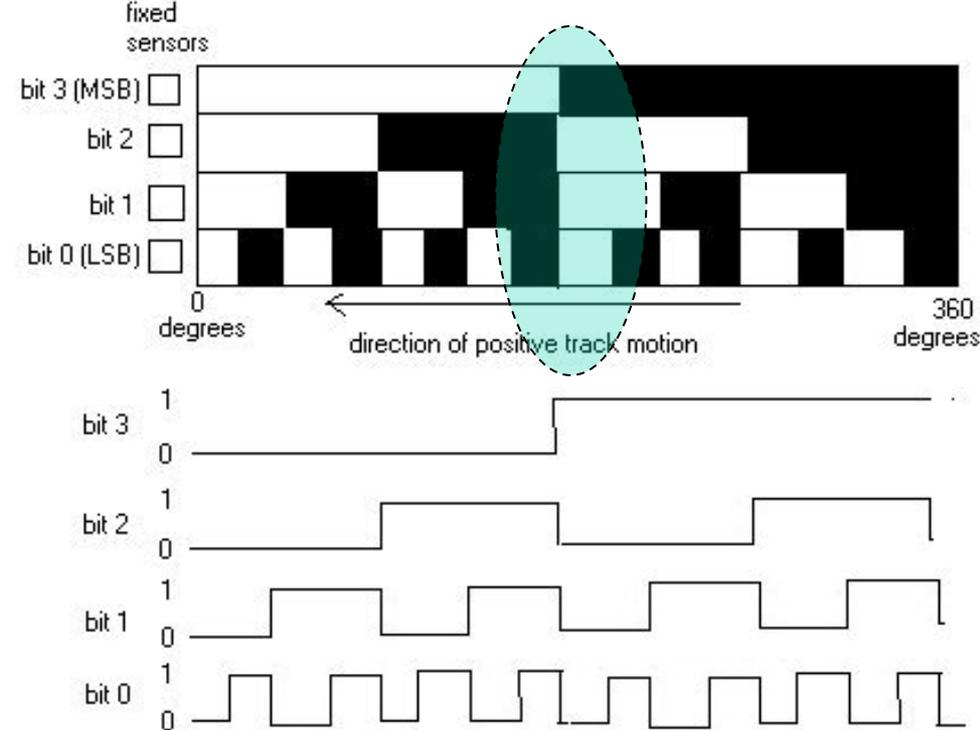


Fig 3. 4-Bit binary code absolute encoder disk track patterns

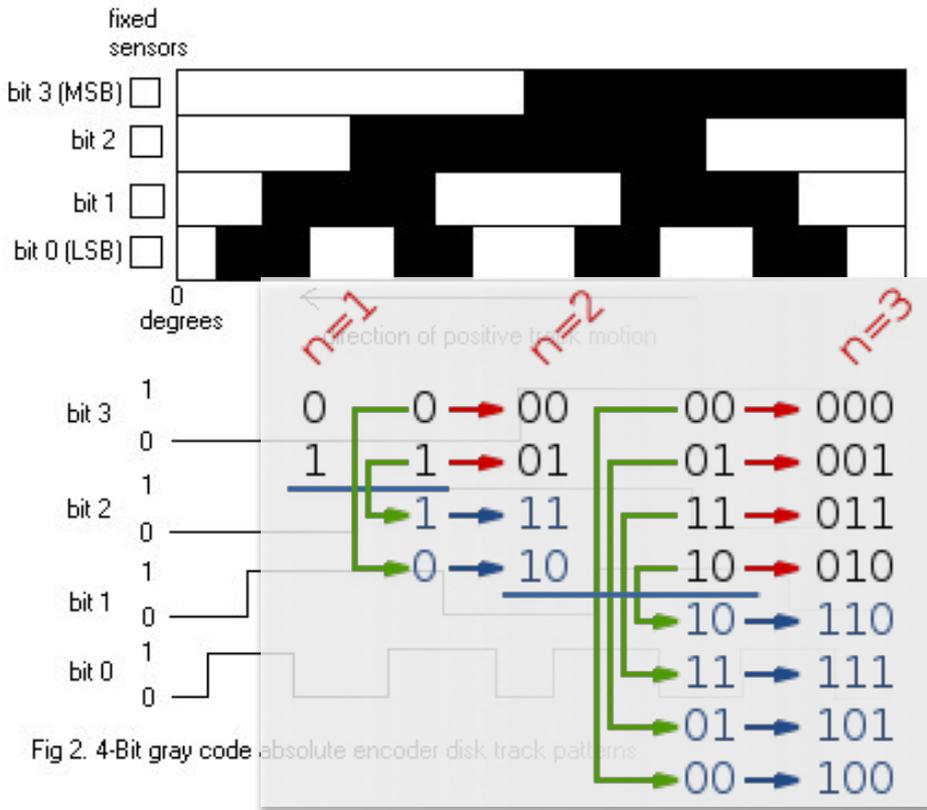
“Saltos grandes” => + erros

Código Binário:: Não é Encoder Absoluto!

Seq. **Gray** (4-bits)

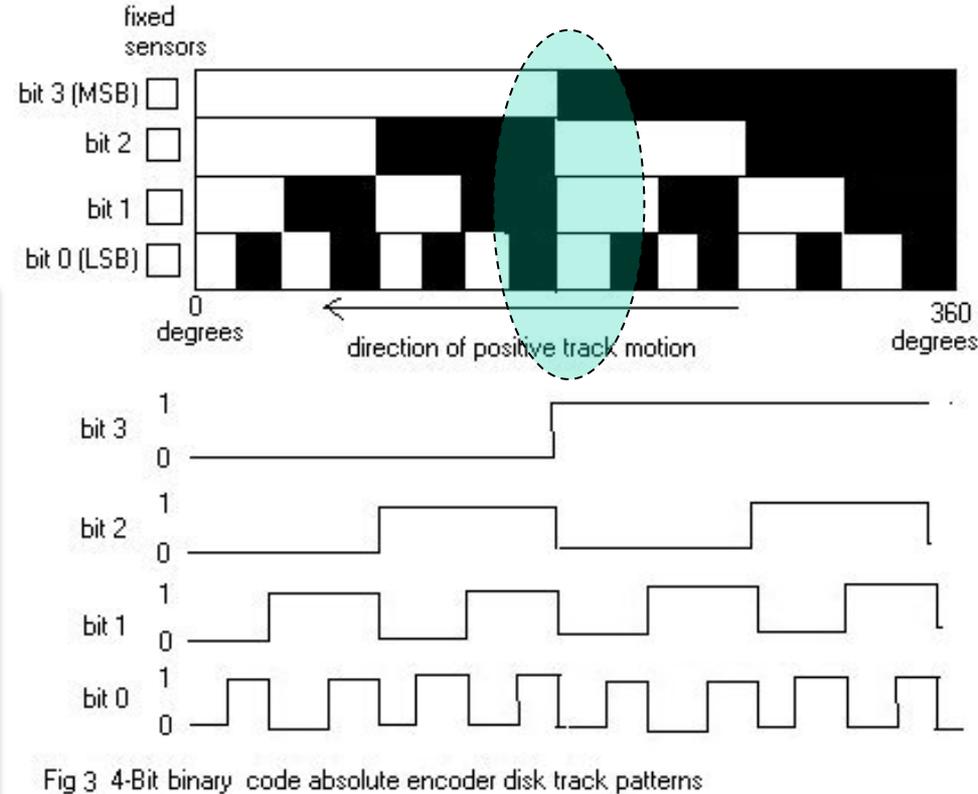
×

Seq. **Binária** (4-bits)



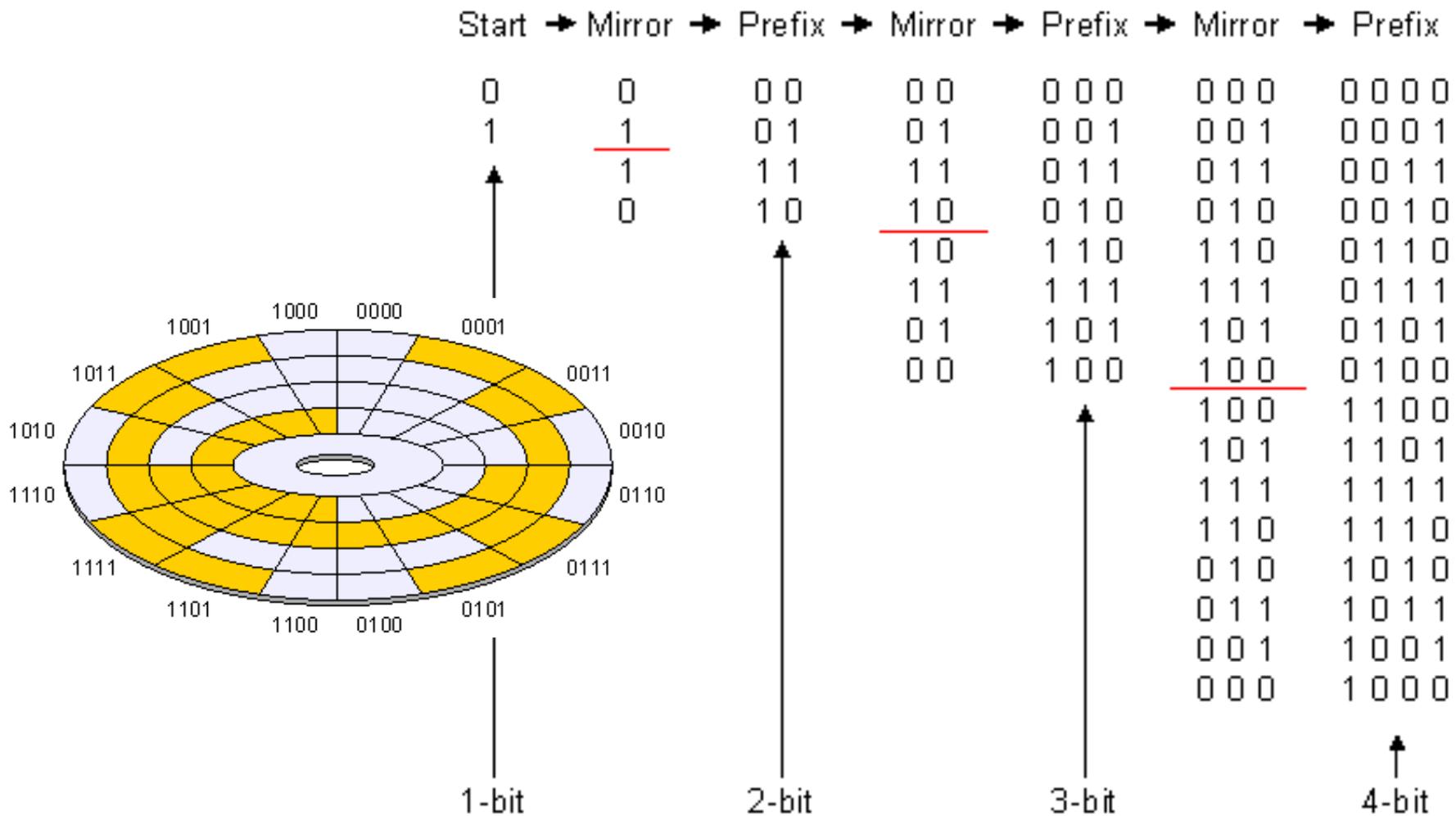
Menos erros

A cada passo: 1 bit apenas varia de estado!



“Saltos grandes” => + erros

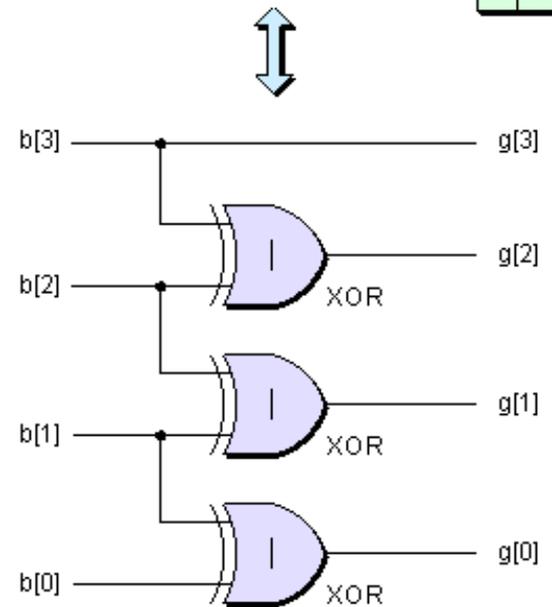
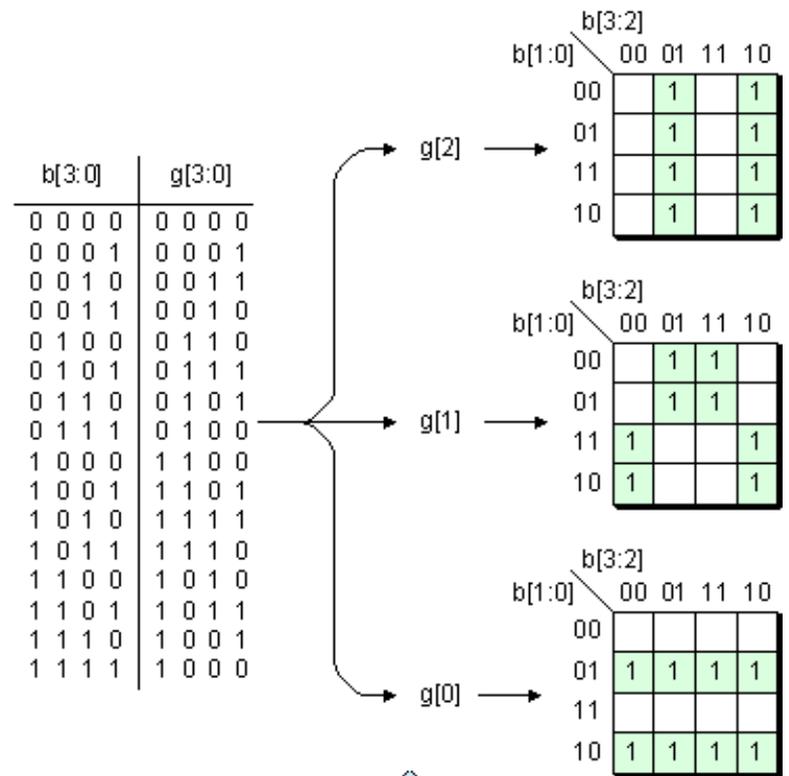
Sequência Gray



Conversão Binário → Gray

	<u>b[3:0]</u>	<u>g[3:0]</u>	
	0 0 0 0	0 0 0 0	
	0 0 0 1	0 0 0 1	
	0 0 1 0	0 0 1 1	
	0 0 1 1	0 0 1 0	
	0 1 0 0	0 1 1 0	
	0 1 0 1	0 1 1 1	
	0 1 1 0	0 1 0 1	
	0 1 1 1	0 1 0 0	
	1 0 0 0	1 1 0 0	
	1 0 0 1	1 1 0 1	
	1 0 1 0	1 1 1 1	
	1 0 1 1	1 1 1 0	
	1 1 0 0	1 0 1 0	
	1 1 0 1	1 0 1 1	
	1 1 1 0	1 0 0 1	
	1 1 1 1	1 0 0 0	

Binary Code → ← Gray Code



Conversão Gray → Binário

	b[3:0]	g[3:0]	
	0 0 0 0	0 0 0 0	
	0 0 0 1	0 0 0 1	
	0 0 1 0	0 0 1 1	
	0 0 1 1	0 0 1 0	
	0 1 0 0	0 1 1 0	
	0 1 0 1	0 1 1 1	
	0 1 1 0	0 1 0 1	
	0 1 1 1	0 1 0 0	
	1 0 0 0	1 1 0 0	
	1 0 0 1	1 1 0 1	
	1 0 1 0	1 1 1 1	
	1 0 1 1	1 1 1 0	
	1 1 0 0	1 0 1 0	
	1 1 0 1	1 0 1 1	
	1 1 1 0	1 0 0 1	
	1 1 1 1	1 0 0 0	

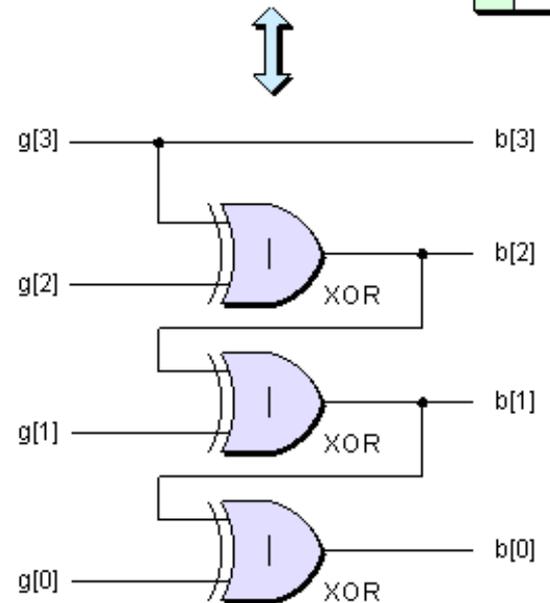
Binary Code ← Gray Code

g[3:0]	b[3:0]	
0 0 0 0	0 0 0 0	
0 0 0 1	0 0 0 1	
0 0 1 1	0 0 1 0	
0 0 1 0	0 0 1 1	
0 1 1 0	0 1 0 0	
0 1 1 1	0 1 0 1	
0 1 0 1	0 1 1 0	
0 1 0 0	0 1 1 1	
1 1 0 0	1 0 0 0	
1 1 0 1	1 0 0 1	
1 1 1 1	1 0 1 0	
1 1 1 0	1 0 1 1	
1 0 1 0	1 1 0 0	
1 0 1 1	1 1 0 1	
1 0 0 1	1 1 1 0	
1 0 0 0	1 1 1 1	

	g[3:2]	g[1:0]		
00	00	01	11	10
01	1			1
11	1			1
10	1			1

	g[3:2]	g[1:0]		
00	00	01	11	10
01		1		1
11	1		1	
10	1		1	

	g[3:2]	g[1:0]		
00	00	01	11	10
01	1		1	
11		1		1
10	1		1	



Demonstração Código Gray:

<http://demonstrations.wolfram.com/GrayCodesErrorReductionWithEncoders/>

The screenshot shows a web browser window displaying the Wolfram Demonstrations Project page for "Gray Code's Error Reduction with Encoders". The page features a navigation bar with a search box and links for "EXPLORE", "LATEST", "ABOUT", "PARTICIPATE", and "AUTHORIN". The main content area is titled "Gray Code's Error Reduction with Encoders" and includes a "Replay" button. Below the title, there are two circular diagrams representing Gray and Binary codes. The Gray code diagram is labeled "Gray" and shows a circular arrangement of black and white segments. The Binary code diagram is labeled "Binary" and shows a similar arrangement. Below each diagram, the binary code "0111" is shown for the Gray code and "0101" for the Binary code, both labeled "Step: 5". A central orange button says "Interact Now! Get free Wolfram CDF Player »". To the right of the main content, there is a "Share" section with social media icons, an "Embed Interactive Demonstration New!" section with a script tag, and a "Download Demonstration as CDF" and "Download Author Code" section. At the bottom right, there is a "Related Demonstrations" section with a list of other demonstrations by Michael Schreiber.

Wolfram Demonstrations Project™ 9427 Interactive Demonstrations Powered by CDF Technology

SEARCH EXPLORE LATEST ABOUT PARTICIPATE AUTHORIN

Gray Code's Error Reduction with Encoders

Replay »

Movement 2.12 Misalignment 0 Encoder Rotary Linear

Interact Now!
Get free Wolfram CDF Player »

Gray Binary

0111 Step: 5 0101 Step: 5

Real step: 5

Share: [Email] [LinkedIn] [Facebook] [Twitter] [StumbleUpon] [Digg]

Embed Interactive Demonstration New!
<script type="text/javascript" src="http://>

Download Demonstration as CDF »
Download Author Code » (preview »)
Files require Wolfram CDF Player or Mathematica.

Related Demonstrations More by Author

- Binary Gray Code Michael Schreiber
- Elementary Cellular Automaton Rules by Gray Code Michael Schreiber
- Mixed Base Gray Codes Michael Schreiber
- Gray Indexed Minimum Change Permutation Michael Schreiber