

# Experimento 5

## Circuitos Combinacionais: Codificador e Decodificador

**Matheus Cardoso de Souza, 202033507**  
**Ualiton Ventura da Silva, 202033580**  
**Grupo G42**

<sup>1</sup>Dep. Ciéncia da Computação – Universidade de Brasília (UnB)  
CIC0231 - Laboratório de Circuitos Lógicos

matheus-cardoso.mc@aluno.unb.br, 202033580@aluno.unb.br

**Abstract.** *The current report aims to build and analize logic circuits that implement encoder and decoder functionalities; Having encoders with 10 bits for input and 4 bits as their output, while decoders with 4 bits for input and 10 bits for output. We also discuss about the importance of encoders and decoders, a few of their applications in real world problems and finally the implementation of a conversor of Gray code values into symbols of the decimal sistem.*

**Resumo.** *O presente relatório tem como objetivo a elaboração e análise de circuitos codificadores e decodificadores, sendo os codificadores compostos em sua entrada por 10 bits e saída de 4, já os decodificadores o inverso, ou seja, 4 bits de entrada e 10 de saída. Abordamos também a importância de codificadores e decodificadores, algumas de suas aplicações em problemas reais e também a implementação de um conversor de valores no código de Gray para símbolos do sistema decimal.*

### 1. Introdução

Codificadores tem como objetivo a simplificação de um conjunto de bits dados em uma versão simplificada sem perder a informação desejada; Por exemplo, se tivermos uma entrada de 10 bits poderíamos produzir saídas de 4 bits, mas observa-se que somente será possível se algumas das combinações de entrada obtiverem a mesma saída, porque para 10 bits existem  $2^{10} = 1024$  combinações e sua saída por possuir 4 bits é capaz de produzir apenas  $2^4 = 16$  saídas. [Mandelli 2021]

Diante desse caso existem duas possibilidades: Ou de fato as saídas possuirão alguns resultados iguais para entradas distintas; Ou para casos como estes, em que as entradas são diferentes mas as saídas são iguais, parte destas entradas serão desconsideradas por serem irrelevantes para o problema (vide [Koike and Mandelli 2021] para maiores informações).

Utilizando decodificadores temos o comportamento oposto ao codificador, então para um dado conjunto de informações simplificadas ocorrerá a decomposição em uma maior quantidade de bits; Por exemplo, para uma entrada de 4 bits poderíamos ter uma decomposição em 10 bits, sendo esse um comportamento desejado para a reconstituição de uma informação simplificada em sua original.

Os circuitos que implementam funcionalidades de codificadores e decodificadores são muito úteis por permitirem, assim como o nome sugere, a conversão de uma forma de codificação de dados em outra e, dessa forma, proporciona uma capacidade de pessoas e máquinas operarem com dados. Tomemos por exemplo o fato de que, para o computador, é muito conveniente que os dados sejam manipulados na forma de números binários (isso permite que circuitos lógicos e elétricos sejam construídos com maior resistência a ruído); Entretanto, os computadores são apenas ferramentas, e quem de fato fará a interpretação dos dados é um ser humano, que opera de forma mais comum com números de base decimal. Tal conflito entre diferentes codificações de dados pode ser facilmente resolvido com codificadores e decodificadores, que transformam dados em base binária para decimal e vice-versa. Esse exemplo foi, entretanto, admitidamente simplista. Existem muitas outras aplicações importantes para codificadores e decodificadores, como por exemplo compressão de dados (transformando inputs de tamanho  $2^n$  em outputs com tamanho  $n$ ), conversão de dados, entre outros.

Neste relatório, nos debruçaremos principalmente sobre a conversão de dados no código de *Gray* para símbolos no sistema decimal e vice-versa.

## 1.1. Objetivos

Os textos subsequentes deste relatório tem como fim a elaboração de codificadores e decodificadores, sendo estes codificadores compostos em sua entradas por 10 bits e saída de 4 bits, sendo expressos no código de *Gray*. Já os decodificadores serão de maneira oposta composto por entradas de 4 bits, no código de *Gray*, e tendo por saída, dados de 10 bits; Como resultado, por ambos circuitos possuirem lógica relacionada ocorrerá a codificação de uma informação e através do decodificador ocorrerá a reconstituição da informação.

## 1.2. Materiais

Em função da natureza do ensino a distância, os presentes experimentos não foram realizados usando-se materiais e equipamentos físicos, mas sim emulados por meio do Deeds.

A seguir estão enumerados os materiais utilizados:

- Componentes de Entrada (Gerador de clock e chaves de estado)
- Componentes de Saída
- Portas Lógicas **OR, AND, NOT**

## 2. Procedimentos

Passaremos a apresentar os experimentos requeridos.

### 2.1. Elaboração de Codificador

Para a elaboração do circuito, deve-se analisar a seguinte tabela:

**Tabela 1. Tabela Verdade para o Decodificador**

Entradas										Saídas			
$e_0$	$e_1$	$e_2$	$e_3$	$e_4$	$e_5$	$e_6$	$e_7$	$e_8$	$e_9$	<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>
1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	0	0	0	1	1
0	0	0	1	0	0	0	0	0	0	0	0	1	0
0	0	0	0	1	0	0	0	0	0	0	1	1	0
0	0	0	0	0	1	0	0	0	0	0	1	1	1
0	0	0	0	0	0	1	0	0	0	0	0	1	0
0	0	0	0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	0	0	0	1	0	1	1	0	0
0	0	0	0	0	0	0	0	0	1	1	1	0	1

Sendo que “0” é codificado como “0000000001”, “1” como “0000000010”, ..., “9” como “1000000000”. Temos que para cada 10 bits de entrada temos que somente 1 será dado como 1, por exemplo, não temos a entrada “0000000011”.

Denotando cada um dos 10 bits como  $e_9, e_8, e_7, e_6, e_5, e_4, e_3, e_2, e_1, e_0$ . Assim, deve-se analisar também que cada bit também descreve seu respectivo símbolo; Por exemplo,  $e_8$  descreve o símbolo “8”, pois, quando  $e_8 = 1$  temos que todos os outros serão iguais a 0, portanto, estaremos descrevendo o símbolo “8”.

Para maior simplicidade podemos analisar o fato de que para a saída  $A$ , depende de somente as entradas  $e_8$  e  $e_9$ , como cada caso irá ocorrer de maneira individual,  $A$  poderá ser descrito como:

$$F_A(e_8, e_9) = e_8 + e_9 \quad (1)$$

Observação, é importante atentar-se ao fato de que em casos onde o valor selecionado não envolve “8” ou “9”, obrigatoriamente  $e_8 = 0$  e  $e_9 = 0$ , assim,  $A$  também será igual a 0.

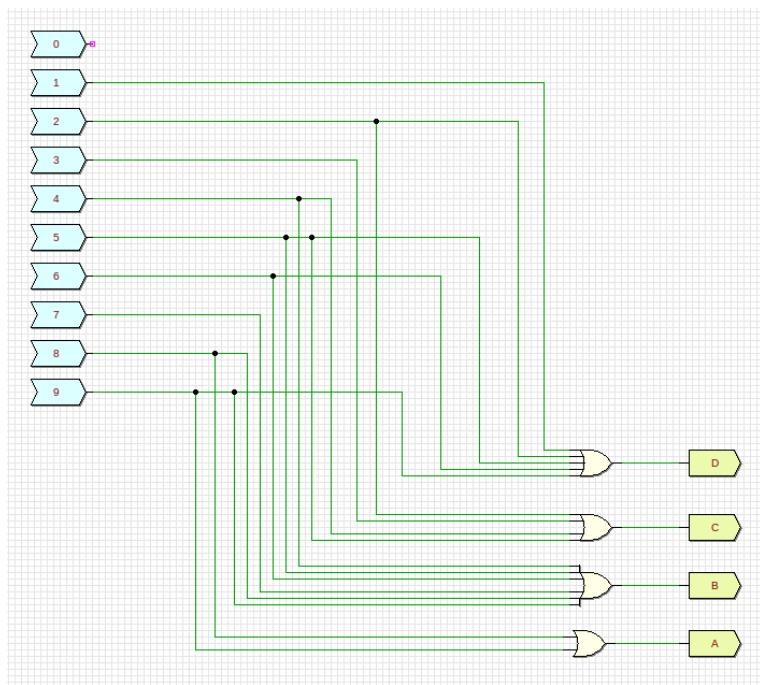
Utilizando raciocínio semelhante para os demais pode ser elaborado outra das equações booleanas, temos:

$$F_A(e_8, e_9) = e_8 + e_9 \quad (2)$$

$$F_B(e_4, e_5, e_6, e_7, e_8, e_9) = e_4 + e_5 + e_6 + e_7 + e_8 + e_9 \quad (3)$$

$$F_C(e_2, e_3, e_4, e_5) = e_2 + e_3 + e_4 + e_5 \quad (4)$$

$$F_D(e_1, e_2, e_5, e_6, e_9) = e_1 + e_2 + e_5 + e_6 + e_9 \quad (5)$$

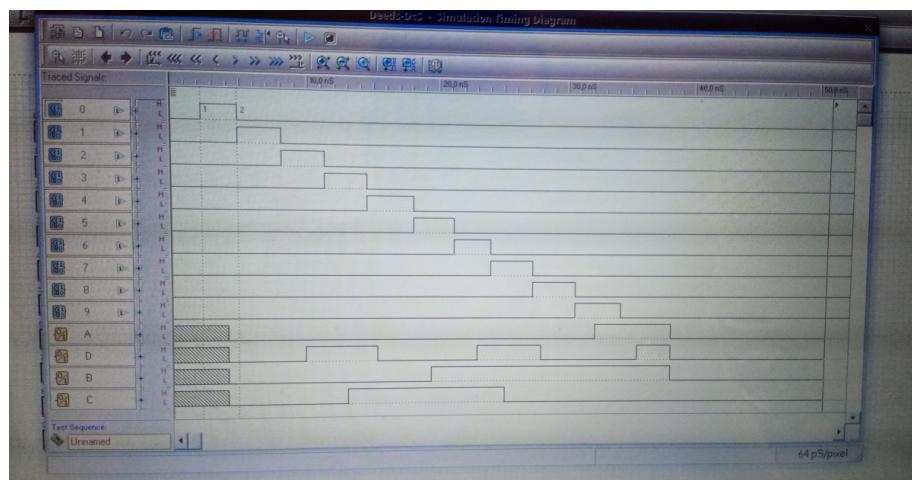


**Figura 1. Subcircuito Comparador**

Importante notar que não todas as saídas são descritas, pois, possuindo 10 bits de entrada, haverão  $2^{10}$  de combinações, portanto haveriam  $2^{10}$  de saídas, contudo nos atentamos a somente algumas, todas as outras são casos não relevantes para o problema em questão.

## 2.2. Simulações do Codificador

O diagrama esquemático será descrito por:



**Figura 2. Gráfico em forma de onda para o circuito do Codificador**

Para conferir o vídeo deste experimento, acesse o seguinte link:  
<https://youtu.be/x9ct4AeliNU>

### 2.3. Obtendo as funções booleanas para o decodificador

Nesta seção temos por objetivo obter as funções booleanas para o decodificador de números em código de *Gray* para números em decimal.

Como explicado em [Mandelli 2021], e com em maiores detalhes em [Koike and Mandelli 2021], para criarmos uma função que funcione como decodificador, precisaremos obter uma função separada para cada um dos outputs existentes. Assim sendo, teremos um total de 10 funções, uma para cada símbolo do código decimal. E, para obtermos as funções individuais, devemos considerar quais inputs no código de *Gray* produzirão o resultado para o símbolo em decimal com valor lógico 1.

Tendo por base o código de *Gray* especificado em [Code 2021], é fácil verificarmos quais devem ser as funções booleanas de cada saída. Reproduzimos em seguida cada uma delas:

$$e_0 = \overline{A} \cdot \overline{B} \cdot \overline{C} \cdot \overline{D} \quad (6)$$

$$e_1 = \overline{A} \cdot \overline{B} \cdot \overline{C} \cdot D \quad (7)$$

$$e_2 = \overline{A} \cdot \overline{B} \cdot C \cdot D \quad (8)$$

$$e_3 = \overline{A} \cdot \overline{B} \cdot C \cdot \overline{D} \quad (9)$$

$$e_4 = \overline{A} \cdot B \cdot C \cdot \overline{D} \quad (10)$$

$$e_5 = \overline{A} \cdot B \cdot C \cdot D \quad (11)$$

$$e_6 = \overline{A} \cdot B \cdot \overline{C} \cdot D \quad (12)$$

$$e_7 = \overline{A} \cdot B \cdot \overline{C} \cdot \overline{D} \quad (13)$$

$$e_8 = A \cdot B \cdot \overline{C} \cdot \overline{D} \quad (14)$$

$$e_9 = A \cdot B \cdot \overline{C} \cdot D \quad (15)$$

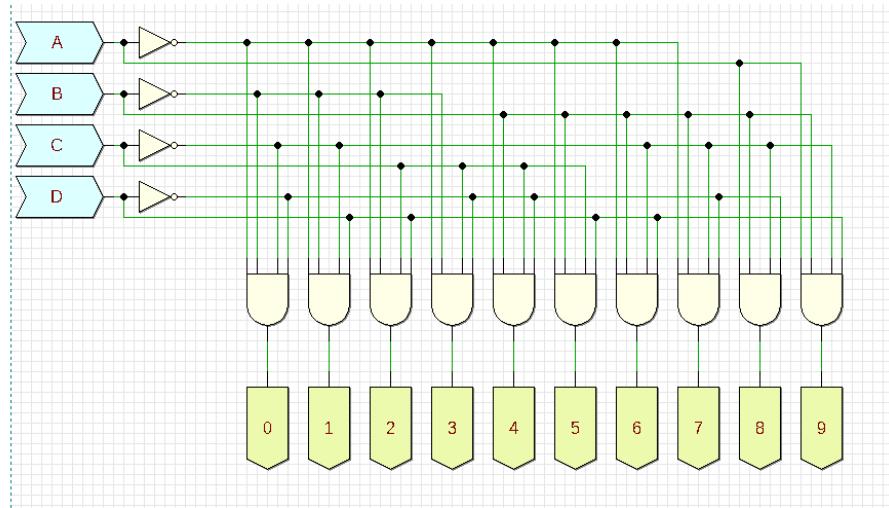
É interessante notarmos que não mencionamos sobre valores *don't care* na tabela, mas eles existem, e são referentes a todas as entradas que não correspondam ao valores dentro do intervalo  $0 \leq x \leq 9$ . Teoricamente poderíamos tentar otimizar as funções apresentadas anteriormente, fazendo uso de métodos como mapa de Karnaugh ou Quine-McCluskey, mas fazendo isso teríamos um circuito com um delay um pouco maior, devido ao fato de haverem mais portas lógicas acopladas, aumentando o caminho crítico do circuito, assim como explicado em [Koike and Mandelli 2021] e [Mandelli 2021]. Optamos por ter um delay menor no circuito final.

Agora, de posse das funções finais para cada um dos outputs esperados, podemos criar um circuito lógico que satisfaça as referidas funções booleanas. O mapeamento de funções para portas lógicas é bem direto nesse caso, pois é constituido apenas de portas **AND** e **NOT**, assim como é possível observar na figura 3

Assim, concluímos as etapas de obtenção das funções booleanas para o decodificador e também da criação do diagrama lógico total.

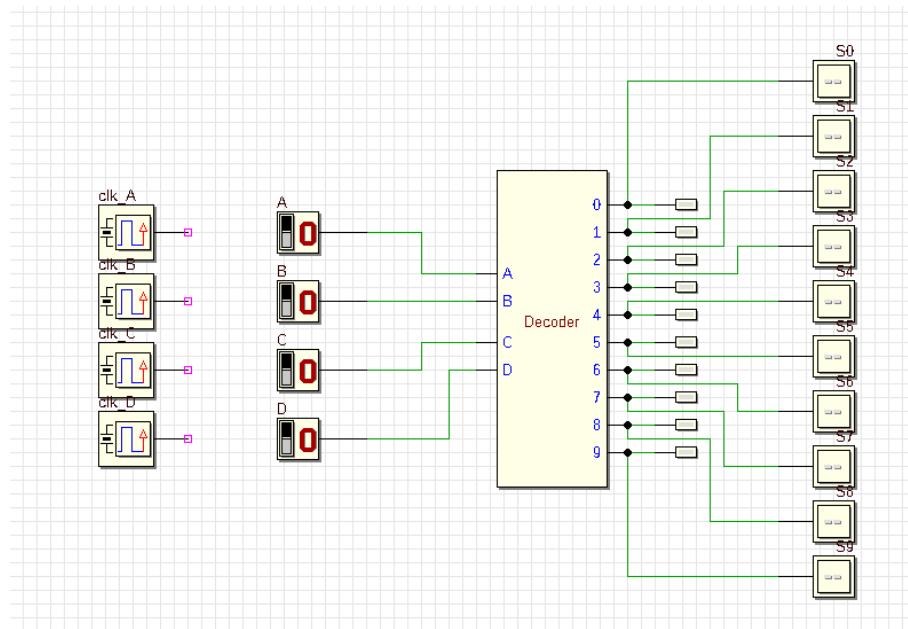
### 2.4. Implementando e testando o *Decodificador*

Passamos agora a mostrar a implementação do circuito lógico total do decodificador, bem como seu funcionamento demonstrado em vídeo.



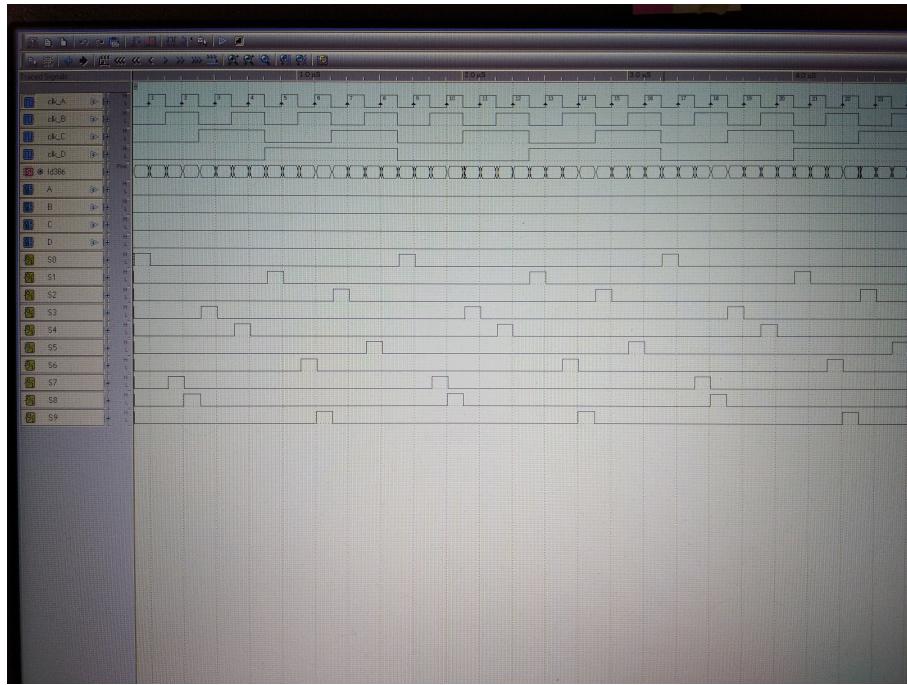
**Figura 3. Circuito Lógico para o Decodificador**

A figura 4 mostra a montagem do circuito, suas entradas e também suas saídas. É interessante notar que colocamos as entradas interativas bem como os clocks no circuito. As entradas interativas facilitam na demonstração em vídeo, e os clocks tornam fácil a criação do gráfico de ondas que pode ser visualizado na figura 5.



**Figura 4. Circuito Lógico do Decodificador com inputs e outputs**

De posse das informações na figura 5, podemos preencher a tabela verdade para esse decodificador; Levando em conta os inputs válidos, inválidos e os valores *don't care*, chegamos à tabela 2.



**Figura 5. Gráfico de Onda do Decodificador**

**Tabela 2. Tabela Verdade para o Decodificador**

Entradas				Saídas									
A	B	C	D	$e_0$	$e_1$	$e_2$	$e_3$	$e_4$	$e_5$	$e_6$	$e_7$	$e_8$	$e_9$
0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0	0	0
0	0	1	1	0	0	1	0	0	0	0	0	0	0
0	0	1	0	0	0	0	1	0	0	0	0	0	0
0	1	1	0	0	0	0	0	1	0	0	0	0	0
0	1	1	1	0	0	0	0	0	1	0	0	0	0
0	1	0	1	0	0	0	0	0	0	0	1	0	0
0	1	0	0	0	0	0	0	0	0	0	0	1	0
1	1	0	0	0	0	0	0	0	0	0	0	0	1
1	1	0	1	0	0	0	0	0	0	0	0	0	1
1	1	1	1	*	*	*	*	*	*	*	*	*	*
1	1	1	0	*	*	*	*	*	*	*	*	*	*
1	0	1	0	*	*	*	*	*	*	*	*	*	*
1	0	1	1	*	*	*	*	*	*	*	*	*	*
1	0	0	1	*	*	*	*	*	*	*	*	*	*
1	0	0	0	*	*	*	*	*	*	*	*	*	*

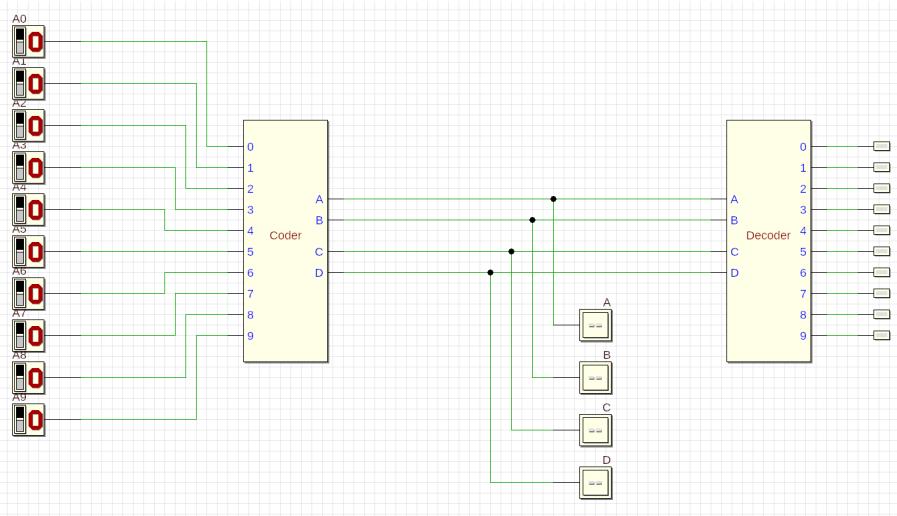
Assim, concluímos as simulações para o circuito do decodificador.

Para conferir o vídeo deste experimento, acesse o seguinte link:  
<https://youtu.be/elAwvxQr8o>

## 2.5. Integração de *Codificador* e *Decodificador*

Para finalizarmos este experimento, realizaremos a integração entre o codificador, feito no item 2.2 e o decodificador do item 2.4.

A montagem final, integrando todos os circuitos, pode ser visualizada na figura 6:



**Figura 6. Circuito de *Codificador* e *Decodificador* integrados**

Agora, para a visualização do funcionamento do circuito, disponibilizamos o vídeo do experimento, que pode ser acessado pelo seguinte link: <https://youtu.be/BpFrHE5Ggb8>

## 3. Análise dos Resultados

Todos os tópicos apresentados foram implementados de forma satisfatória, assim como requisitado. Passaremos agora às considerações finais de cada um dos tópicos de forma individual e, por último, faremos uma avaliação do conjunto de experimentos deste relatório.

### 3.1. Análise em 2.1 e 2.2

Para os itens 2.1 e 2.2 temos que de fato ocorreram os resultados esperados, concluindo que apesar da não utilização de soma de produtos e produtos de somas ainda assim pudemos descrever o comportamento desejado, ou seja, o codificador simplificou as entradas a saídas que descrevem representações do código de gray, sendo que estas mesmas entradas são descritas por símbolos decimais, que por sua vez são descritos através da composição de 10 bits.

### 3.2. Análise em 2.3 e 2.4

Para os itens 2.3 e 2.4, pudemos a inicio teorizar o circuito utilizado e então comprovar sua eficácia, assim, ao ser colocado em prática foi possível a conclusão de seu real funcionamento, sendo de fato o esperado.

### 3.3. Análise em 2.5

Dado que os circuitos elaborados nas seções anteriores tiveram o comportamento desejado, temos que por fim ao criarmos uma entrada de um codificador descrito por funções do tipo  $f_k(\text{entrada}) = \text{saída}$  (onde  $k \in \{A, B, C, D\}$ ), temos então que ao invertermos esta função, ou seja,  $g_k(\text{saída}) = \text{entrada}$ , ao realizarmos a composição de funções, ou seja,  $g_k \circ f_k(\text{entrada}) = g(f(\text{entrada}))$  teremos que  $g(f(\text{entrada})) = \text{entrada}$ .

Nos experimentos realizados, temos que o codificador produz o comportamento inverso daquela do decodificador, podendo ser aplicado o mesmo raciocínio a saídas esperadas, temos que ao associá-los como resultado deverá haver o mesmo que tínhamos na saída, e de fato obtivemos tal comportamento.

## 4. Conclusão

Temos, por fim, que de fato pudemos descrever e elaborar codificadores e decodificadores, possuindo o comportamento esperado. Devemos nos atentar ao fato de que, apesar da comprovação e elaboração dos circuitos, esses estão limitados ao escopo deste relatório, não descrevemos generalizações que aplicam-se a todos os possíveis codificadores e decodificadores.

É necessário observar também que além da comprovação de funcionamento foi possível compreender que codificadores e decodificadores possuem comportamentos opostos, assim descrito no último item.

Foi interessante notarmos também como pudemos, de forma relativamente simples, converter codificações de dados distintas, e, de certa forma, até fazermos compressão e expansão de dados, haja vista que criamos um circuito que pode livremente converter dados com códigos diferentes representados em 4 bits (o código de *Gray*) e também os símbolos do sistema decimal (representados com 10 bits). Assim, pudemos demonstrar de forma prática alguns dos pontos mencionados na introdução sobre a importância prática de codificadores e decodificadores como sistemas que permitem conversão entre distintos formatos de codificações e também como forma de circuito que permite compressão e expansão de dados.

## Referências

- [Code 2021] Code, W. G. (2021). Gray code table. [https://www.wikiwand.com/en/Gray\\_code](https://www.wikiwand.com/en/Gray_code).
- [Koike and Mandelli 2021] Koike, C. and Mandelli, M. G. (2021). Codificadores e decodificadores. <https://aprender3.unb.br/mod/url/view.php?id=386056>.
- [Lamar 2021] Lamar, M. V. (2021). Laboratório de circuitos lógicos.
- [Mandelli 2021] Mandelli, M. G. (2021). Codificadores. <https://aprender3.unb.br/mod/resource/view.php?id=386054>.

## Auto-Avaliação

Respostas:

Questão	Resposta
1	c
2	c
3	a
4	b
5	c
6	e → NOT A defeituosa, sempre com nível 0 na saída
7	b → Porta NOT D defeituosa, sempre em 1
8	f → Porta NOT A defeituosa, sempre em 1