Estratégia Concursos

Aula 00 - Prof. Thais Martins

SED-SC (Professor - Controle e Processos Industriais) Conhecimentos Específicos (Parte de Eletricidade) - 2026 (Pós-Edital)

Autor:

Mariana Moronari, Thais Martins

02 de Abril de 2026

Índice

Apresentação da Professora

Olá, concurseiro(a)!

Se tem algo que sempre fez parte da minha trajetória, é o estudo. Minha jornada acadêmica e profissional é marcada pelo aprendizado contínuo, pela superação de desafios e, acima de tudo, pela busca por conhecimento com estratégia.

Sou a professora **Thais Martins**, doutora em **Engenharia Elétrica** pela **Universidade Federal de Juiz de Fora (UFJF)** e pós-doutora pela mesma instituição. Atualmente, realizo meu **segundo pós-doutorado na COPPE/UFRJ**, onde desenvolvo pesquisas avançadas na área. Minha jornada acadêmica começou na **Universidade Federal de Lavras (UFLA)**, onde me formei em **Engenharia de Controle e Automação** e concluí o **mestrado em Engenharia de Sistemas e Automação**.

Ao longo da minha trajetória, fui aprovada em **oito concursos públicos** para a área de ensino superior, ficando em **primeiro lugar em seis deles**. Essas aprovações foram para **professora temporária em instituições como CEFET-MG, UFLA e UERJ**. Também fui professora no Centro Universitário de Lavras (UNILAVRAS), lecionando para diferentes áreas da engenharia.

Hoje, divido meu tempo entre diferentes frentes: atuo como professora para concursos públicos na área de **Engenharia Elétrica e Eletrônica**, sigo minha jornada como **pesquisadora no pós-doutorado**, e também desenvolvo outros projetos que me desafiam continuamente. Mas, acima de tudo, **ensinar é uma das minhas maiores paixões**.

A experiência de passar por tantos desafios acadêmicos e profissionais me ensinou algo valioso: **estudar da forma certa faz toda a diferença**. Mais do que acumular conhecimento, é preciso aprender **como estudar**, identificar o que realmente importa e aplicar estratégias que otimizam o aprendizado.

Se você quer tornar sua preparação mais eficiente e estratégica, meu intuito é te auxiliar nisso. **Com dedicação, método e direcionamento, a aprovação deixa de ser um sonho e se torna um objetivo alcançável.**

Estamos juntos nessa jornada, rumo à aprovação!

@thaismartins.profa
profa.thaism@gmail.com

Considerações Iniciais

Sejam bem-vindos ao nosso estudo sobre **Eletrônica Digital**, uma área fundamental e frequentemente cobrada em **concursos** de Engenharia Elétrica e Eletrônica. Dominar esses conceitos é essencial para compreender sistemas digitais e resolver questões práticas que aparecem frequentemente nessas provas.

A Eletrônica Digital teve seu desenvolvimento impulsionado em 1938, quando o matemático e engenheiro norte-americano Claude Shannon (1916–2001) aplicou a **Álgebra Booleana**, criada por **George Boole** (1815–1864), na análise de circuitos de chaveamento elétrico. Essa abordagem inovadora permitiu a tradução de operações lógicas em implementações físicas com dispositivos binários.

Neste curso, iniciaremos com as **Bases Numéricas** mais utilizadas, como Binária, decimal e Hexadecimal, e suas operações de adição e subtração. Em seguida, estudaremos as **Portas Lógicas** fundamentais — AND, OR, NOT, entre outras — que constituem a base da lógica digital.

A compreensão desses conceitos é essencial para garantir um bom desempenho em concursos e avançar com segurança para tópicos mais complexos. Vamos começar!

ELETRÔNICA DIGITAL

Bases Numéricas

Trabalharemos com **4 sistemas numéricos**: **binário**, **octal**, **decimal** e **hexadecimal**. Os sistemas **binário**, **octal** e **hexadecimal** guardam uma **correlação** entre si, visto que as bases destes sistemas possuem em comum o fato de **serem todas potências de 2**. E, por guardarem uma correlação tão íntima, as conversões entre estas bases são *relativamente* fáceis.

O **sistema numérico binário** (*ou base numérica binária*) pode representar qualquer número com apenas **dois algarismos**: 0 e 1. No sistema binário, o termo dígito binário (*binary digit*) é abreviado para o termo **bit** e representa um único "0" ou um único "1" dentro de um número maior, composto por diversos zeros e uns. O sistema numérico binário precisa de uma determinada resolução em números possíveis de bits para poder representar os demais sistemas numéricos, conforme será visto a seguir.

Um outro termo recorrente na análise de sistemas numéricos binários é o de um conjunto de 8 bits, que forma **1 byte**.

Também nas representações por sistemas numéricos binários se empregam os termos"*dígito mais significativo*" e "*dígito menos significativo*".

O *dígito mais significativo* é o **bit mais à esquerda** do número na base binária. Ele é assim chamado pois seu valor é o que possui **maior relevância**, ou seja, representa o número equivalentemente superior na base decimal.

Já o *dígito menos significativo* é o **bit mais à direita** do número na base binária. Ele é assim chamado pois seus valor é o que possui **menor relevância**, ou seja, representa o número equivalentemente inferior na base decimal.

O **sistema numérico octal** (*ou base numérica octal*) pode representar qualquer número com apenas **8 algarismos**: 0, 1, 2, 3, 4, 5, 6, 7.

O **sistema numérico decimal** é o que usamos no nosso dia a dia. Ele pode representar qualquer número com **10 algarismos**: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

Já o **sistema numérico hexadecimal** pode representar qualquer número com **16 símbolos**: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

A título de curiosidade segue uma tabela com a equivalência entre os números primordiais da base decimal para as demais bases:

BASE DECIMAL	BASE BINÁRIA	BASE OCTAL	BASE HEXADECIMAL
0	0000	00	0
1	0001	01	1
2	0010	02	2
3	0011	03	3
4	0100	04	4
5	0101	05	5
6	0110	06	6
7	0111	07	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Para se converter qualquer número da **base decimal** para **outra base**, deve-se empregar **divisões sucessivas pelo número da base de destino**, ou seja, se temos um número "X" na base 10, e queremos representa-lo na base 2, devemos dividir este número "X" por 2 "n" vezes, sempre se anotando o **resto da divisão**. A seguir iremos melhor definir este conceito.

Conversões de uma base qualquer para decimal

Pensemos no número 1010 na **base binária** - *doravante descrito como* ***(1010)₂***.

Queremos encontrar seu equivalente na **base decimal**. O procedimento é o seguinte:

$$(1010)_2 = 1\times 2^3 + 0\times 2^2 + 1\times 2^1 + 0\times 2^0 = 8+0+2+0 = (10)_{10}$$

Ou seja, multiplicamos o número "0" ou "1" na base 2 pelas **potências de 2** representativas de sua **posição** *(ou seja, o mais à esquerda ganha maior relevância - maior expoente - e vai se reduzindo até o expoente zero*).

Pensemos agora no número 12 na **base octal** - ou $(12)_8$. Queremos encontrar seu equivalente na **base decimal**. O procedimento é idêntico, porém agora usamos como **base onúmero 8**.

$$(12)_8 = 1\times 8^1 + 2\times 8^0 = 8+2 = (10)_{10}$$

Por último, pensemos agora no "*número*" A1 na **base hexadecimal** - ou $(A1)_{16}$. Queremos encontrar seu equivalente na **base decimal**. *Como fazer*?

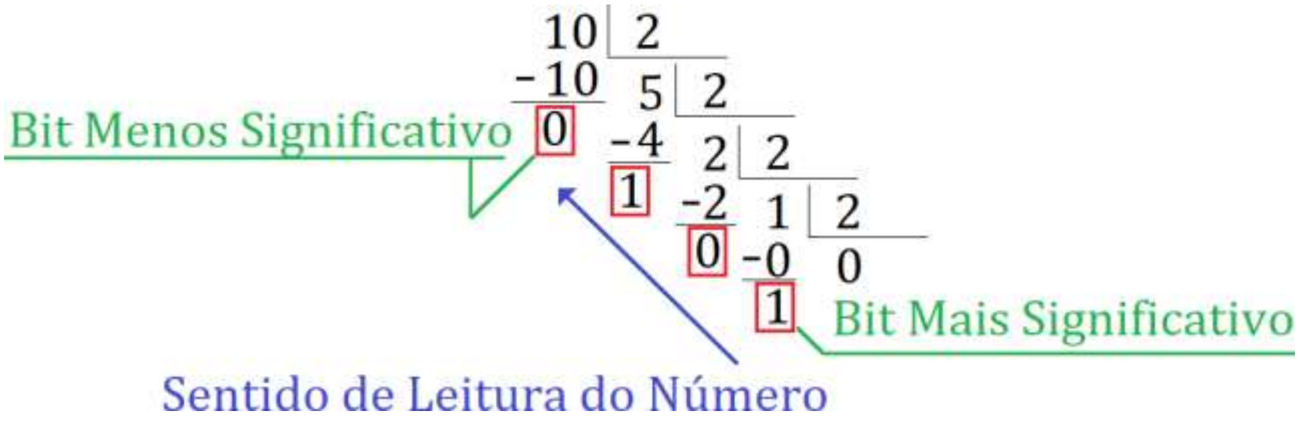
Bom, como não adianta multiplicar "*A1*" por nada, devemos saber **previamente** a correlação das letras da **base hexadecimal** com os seus equivalentes na **base decimal**, conforme disposto na tabela apresentada na página anterior. Sendo assim:

$$(A)_{16} = 10\times 16^1 + 1\times 16^0 = 160+1 = (161)_{10}$$

Simples, não?

Conversões de decimal para uma base qualquer

Façamos agora o **caminho contrário**. Temos o número 10 na **base decimal** e queremos encontrar seu equivalente na **base binária**. O procedimento é o seguinte:



Divide-se **sucessivamente** o número na **base original** pelo número da **base de destino**, preservando-se o valor dos **restos das divisões**, **até que o <u>quociente da última divisão</u> <u>seja zero</u>**.

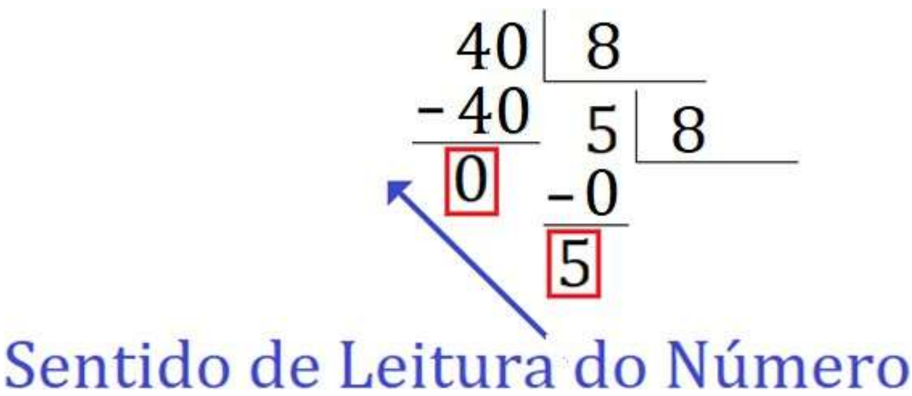
Assim o **número na base de destino** será aquele formando pelos **restos das divisões**,no sentido apontado na Figura acima.

Neste caso, o número formado é o $(1010)_2$ corresponde ao número $(10)_{10}$.

Façamos agora a conversão do **número 40 na base decimal** - ou $(40)_{10}$ - para a **baseoctal**. O procedimento é o mesmo.

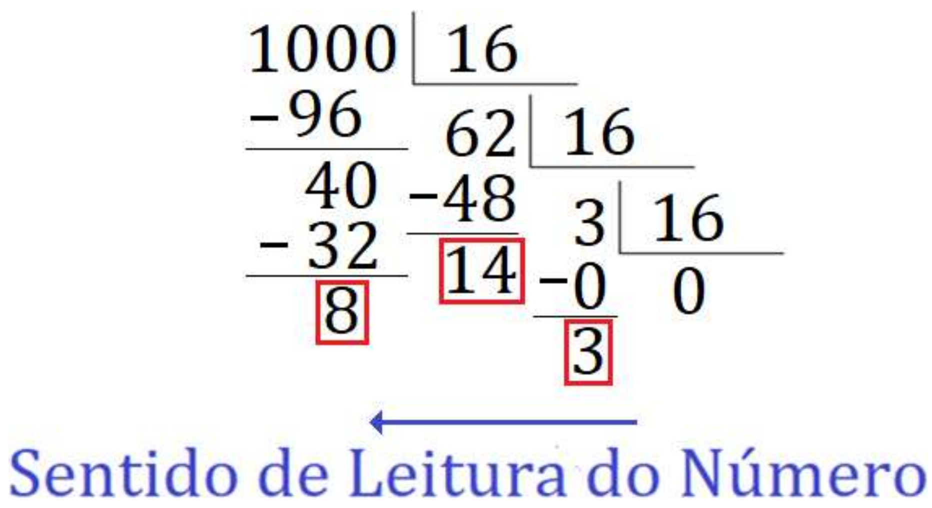
Divide-se sucessivamente o **número na base original** pelo **número da base de destino**, preservando-se o valor dos **restos das divisões, até que o quociente da última divisão seja zero**.

O resultado é o número 50 na base octal - ou $(50)_8$.



Por último, façamos agora a conversão do **número 1000 na base decimal** - ou $(1000)_{10}$

- para a **base hexadecimal**. O procedimento é semelhante.



Como um dos **restos** resultou no **número 14**, e tal número possui um equivalente na **basehexadecimal** (*a letra "E"*), a representação do número $(1000)_{10}$ fica sendo $(3E8)_{16}$.

Conversões entre bases binária, octal e hexadecimal

Para se realizar a **conversão** entre estas **bases**, por elas serem **todas potências de 2**,é possível se empregar uma técnica que torna a **conversão muito mais rápida**.

Tomemos por base o **número binário 010010010011**. Este número possui **12 bits**. Para convertê-lo para a **base octal**, por exemplo, primeiro o separamos em grupos de **3 bits** (*afinal 2^3 = **8**, e buscamos a base **octal***).

$$(010010010011)_2 = (010\,010\,010\,011)_2$$

Ok, nada de novo por aqui. Ainda tenho o mesmo número na base 2...

Agora que vem a parte interessante! Vamos escrever estes números tal como se fosse para a base decimal, porém para cada grupo de 3 bits, desta maneira:

$$(010\,010\,010\,011)_2 = \left(\underbrace{010}_{(2)_8}\,\underbrace{010}_{(2)_8}\,\underbrace{010}_{(2)_8}\,\underbrace{011}_{(3)_8}\right)_2$$

Ou seja, o número $(010010010011)_2$ equivale ao número $(2223)_8$.

Vamos fazer o mesmo para a **base hexadecimal**. Tomemos de novo o número $(010010010011)_2$. Para convertê-lo para a **base hexadecimal**, agora o separamos em gruposde 4 (*afinal 2^4 = **16**, e buscamos a base hexadecimal*):

$$(010010010011)_2 = (0100\,1001\,0011)_2$$

E fazemos o **mesmo procedimento** adotado anteriormente para a base octal:

$$(0100\,1001\,0011)_2 = \left(\underbrace{0100}_{(4)_{16}}\,\underbrace{1001}_{(9)_{16}}\,\underbrace{0011}_{(3)_{16}}\right)_2$$

Ou seja, o número $(010010010011)_2$ equivale ao número $(493)_{16}$.

E se o número não puder ser separado em grupos uniformes de 3 ou 4 bits?

Boa pergunta! Então você deve agrupá-los sempre em grupos de 3 ou de 4 bits, vindoda direita para a esquerda, e os bits que "*faltarem*" você **preenche com zeros**. Tomemos porexemplo o número $(0101001011)_2$, um número com 10 bits. Vamos convertê-los para as bases **octal** e **hexadecimal**:

$$(0101001011)_2 = \begin{cases} \left(\underbrace{000}_{(0)_8}\,\underbrace{101}_{(5)_8}\,\underbrace{001}_{(1)_8}\,\underbrace{011}_{(3)_8}\right)_2 = (513)_8 \\ \left(\underbrace{0001}_{(1)_{16}}\,\underbrace{0100}_{(4)_{16}}\,\underbrace{1011}_{(B)_{16}}\right)_2 = (14B)_{16} \end{cases}$$

Os bits "faltantes" foram destacados na cor cinza claro.

Com isto finalizamos esta parte de representações numéricas e suas conversões. A seguir serão apresentadas as **operações de soma e subtração com números binários, e em seguida as portas lógicas.**

Soma Aritmética com Números Binários

É possível realizar a **soma aritmética** que se faz com os números na **base decimal** também na **base binária**, entretanto não se pode esquecer de algumas peculiaridades.

Na **aritmética binária**, o bit de "*carry*" (*popularmente conhecido como "vai 1"*) é **fundamental** e aparece em praticamente todas as operações.

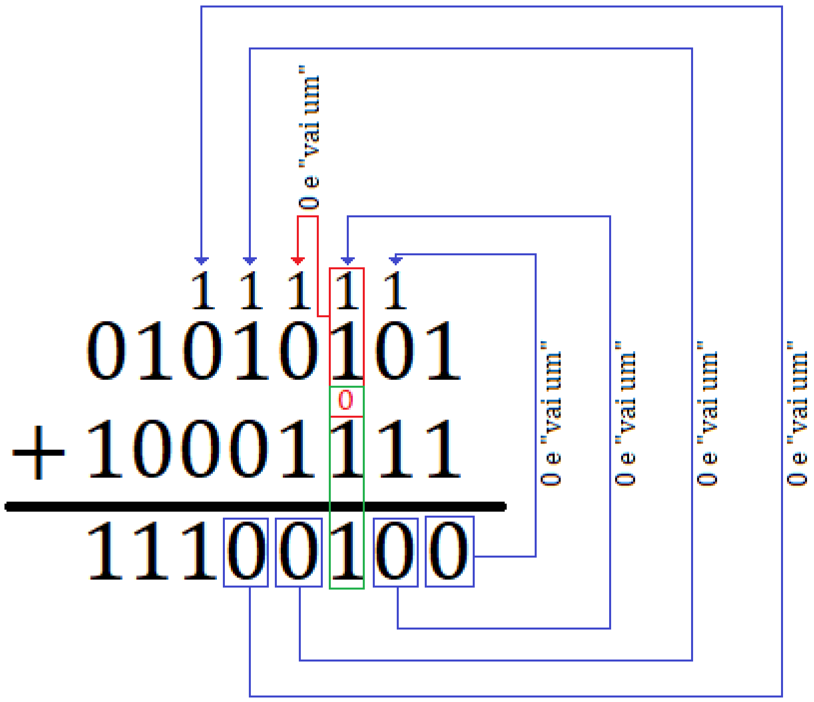
A **aritmética binária** é basicamente feita em cima do que é mostrado na Tabela a seguir.

Bit A	Bit B	Bit A + Bit B
0	0	0
0	1	1
1	0	1
1	1	0 e "*vai um*"

Um **erro comum** é associar a operação "OR" das **operações booleanas** coma **operação aritmética de soma,** pois ambas são definidas pelo operador "+".

Deve-se estar atento ao contexto do que é pedido na questão.

Vamos empregar a **soma de dois bytes A e B**, com o byte A sendo definido por $(01010101)_2$ e o byte B sendo definido por $(10001111)_2$. Começa-se a operação tal como na base decimal, ou seja, da direita para a esquerda.



Bom, se fazemos a soma, então também é possível fazer subtrações, certo?

Sim! Porém, assim como existem regras para se fazer **soma** com números binários existem regras para se fazer **subtração** com números binários.

A **lógica** aqui é a de se somar à parcela da qual se quer subtrair determinada quantidade um **número negativo**, desta forma: consideremos os números $(2)_{10}$ e $(-2)_{10}$ e a operação de soma

$$2+(-2)=0$$

Para se implementar um **número negativo** na **base binária**, a convenção usada é a de **inverter todos os bits**. Esta é a convenção chamada de **complemento de 1**. Ou seja, sabendo que o número $(0010)_2$ representa o número $(2)_{10}$, então para se obter $(-2)_{10}$ basta **inverter os bits**, ficando $(1101)_2$. E assim teremos sempre o bit mais significativo como sendoo chamado bit de sinal (onde 0 irá representar o sinal "+" e o 1 irá representar o sinal "-").

Com isto, um número binário com **4 bits**, em vez de poder representar de $(0)_{10}$ a $(15)_{10}$, iria representar de $(-7)_{10}$, descrito por $(1111)_2$, até o que viria a ser o **absurdo** de **$(-0)_{10}$**, dado por $(1000)_2$. E a parte positiva iria de $(+0)_{10}$, **também absurda** e dada por $(0000)_2$, até $(+7)_{10}$, dada por $(0100)_2$.

Por isto se usa em **operações aritméticas com números binários** o **complemento de 2**, que conserta esta deficiência do complemento de 1.

O **complemento de 2** nada mais é que **somar 1 ao número já representado em complemento de 1**. Assim:

$0010 \longrightarrow$ Número 2 na base binária

$\boxed{1101} \longrightarrow$ Número -2 na base binária em complemento de 1

$$\begin{array}{r} \boxed{1101} \\ +\,0001 \\ \hline 1110 \end{array}$$

$1110 \longrightarrow$ Número -2 na base binária em complemento de 2

Entretanto, agora poderemos representar com **4 bits** os números **$(-8)_{10}$**, dado por $(1111)_2$, até o **$(+7)_{10}$**, dado por $(0111)_2$. O **número zero** na **base binária** fica sendo representado apenas por $(0000)_2$.

Por último, vamos apresentar o conceito de *bit de overflow*. O *overflow* acontece quando uma **operação aritmética** recebe um **bit de *carry*** (*o "vai-um"*) na sua **última operação** e este bit não tem com o que se somar e fica "*sobrando*". Desta maneira:

Soma entre dois números positivos

$$\begin{array}{r} 0010 \\ +\,0011 \\ \hline 0101 \end{array}$$

Soma entre dois números negativos

$$\begin{array}{r} \text{Overflow} - \boxed{1}\,1100 \\ +\,1010 \\ \hline 0110 \end{array}$$

Na Figura acima se verifica que a **soma** à sua esquerda, envolvendo os números $(0010)_2$, que representa o número $(2)_{10}$, com o número $(0011)_2$, que representa o número $(3)_{10}$,resulta no número $(0101)_2$, que representa o número $(5)_{10}$. **Tudo correto**, **sem overflow**!

Já na soma ao lado temos **2 números negativos** em **complemento de 2**. O número $(1100)_2$, que representa o número $(-4)_{10}$, e o número $(1010)_2$, que representa o número $(-6)_{10}$, que resultam no número $(0110)_2$, que representa o número $(+6)_{10}$, ***o que está claramente errado***, visto que **a soma de dois números negativos não pode resultar em um número positivo**. **Este erro pode é apontado pelo bit de overflow destacado acima**!

Isto acontece porque com **4 bits** não se consegue representar o número $(-10)_{10}$. Para isto seriam necessários pelo menos **5 bits**.

Variáveis Booleanas

Através das **tabelas verdade** podemos determinar a **saída** de um determinado **circuito lógico**. Tal tabela relaciona **todas as variações possíveis das entradas de um circuito lógico** e apresenta as respectivas **saídas** para cada uma destas combinações. A seguir apresenta-se um exemplo de **tabela verdade** com duas **variáveis de entrada** (A e B)e uma **variável de saída** (Y).

A	B	Y
0	0	0
0	1	1
1	0	0
1	1	1

A primeira linha da tabela mostra as **variáveis de entrada** e as **variáveis de saída**. Note que o número de linhas abaixo da primeira representa as 4 combinações possíveis destas duas variáveis (A = 0 e B = 0; A = 0 e B = 1; A = 1 e B = 0; A = 1 e B = 1).

Se fossem 3 variáveis de entrada, então haveria 8 combinações. Se fossem 4 variáveis de entrada, haveria 16 combinações.

Percebe como o número de combinações é sempre 2^N, onde N é o número de **variáveis de entrada**? Isso sempre irá ocorrer.

O resultado deste circuito lógico é igual às combinações onde a saída seja "1". Em outras palavras, **nesta tabela verdade**, sempre que a **entrada B** for igual a "1", **a saída Y** será também igual a "1", então podemos dizer que Y = B, pois o valor lógico de A não irá importar no resultado deste circuito lógico.

Portas Lógicas

Algumas combinações de **entradas** já possuem blocos padronizados, chamados de **portas lógicas**. Estas **portas lógicas** são denominadas de acordo com a sua respectiva função:

- Porta **AND** (ou seja, realiza a função lógica "**E**");
- Porta **OR** (ou seja, realiza a função lógica "**OU**");
- Porta **NOT** (ou seja, realiza a função de **inverter o valor lógico** de sua entrada)
- Porta **XOR** (ou seja, realiza a função lógica de "**OU-EXCLUSIVO**").

Os blocos apresentados demonstram sempre **2 entradas** (*com exceção da porta NOT, que apresenta uma única entrada*), porém isto **não é uma regra**. As portas que realizam operação entre **pelo menos 2 entradas**, podem realizar com **N entradas**, sem problema algum, bastando que para isto se modifique sua tabela verdade.

Estes blocos podem ser encadeados uns nos outros, de modo que formem um **circuito lógico** mais complexo. A análise de um **circuito lógico** com diversas portas encadeadas virá a seguir através de um exemplo.

Nas **expressões lógicas** que seguem, sempre que **uma barra** aparecer sobre a **variável booleana**, indica que esta variável está "*negada*", ou seja, seu valor lógico originário na **tabela verdade** que deu origem à expressão era 0. Quando a **variável booleana** aparecer **sem barra**, indica que seu valor lógico originário na **tabela verdade** que deu origem à expressão era 1.

Estes blocos padronizados são os que seguem:

PORTA LÓGICA "OR"

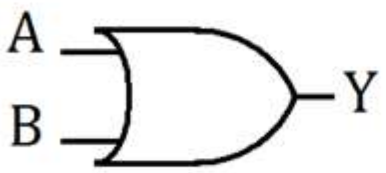


TABELA VERDADE

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

EXPRESSÃO LÓGICA

$$Y = A + B$$

A saída de uma porta **OR** será 1 sempre que pelo menos uma das entradas possuirvalor lógico 1.

PORTA LÓGICA "AND"

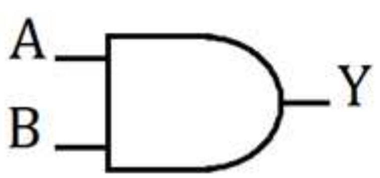


TABELA VERDADE

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

EXPRESÃO LÓGICA

$$Y = A . B$$

A saída de uma porta **AND** será 1 sempre que todas as entradas forem iguais a 1.

PORTA LÓGICA "NOT"

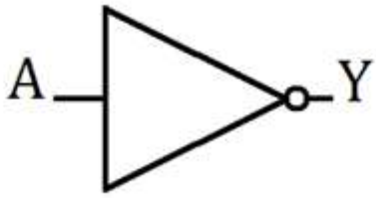


TABELA VERDADE

A	Y
0	1
1	0

EXPRESSÃO LÓGICA

$$Y = \overline{A}$$

A saída de uma porta **NOT** será sempre o inverso do valor lógico da entrada.

PORTA LÓGICA "XOR"

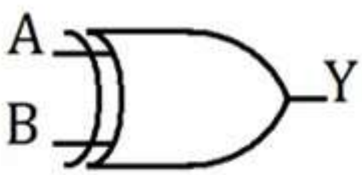


TABELA VERDADE

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

EXPRESSÃO LÓGICA

$$Y = A.\overline{B} + \overline{A}.B = A \oplus B$$

A saída de uma porta **XOR** será sempre 1 sempre que as entradas possuírem valores lógicos diferentes entre si.

Há ainda as portas lógicas **NAND** (ou seja, operação **AND** acrescida da operação **NOT**)e **NOR** (ou seja, operação **OR** acrescida da operação **NOT**). Tais portas **são muito importantes** na implementação de circuitos lógicos de maneira **mais eficiente**. Porém isto será visto *a posteriori*, após a apresentação de **alguns teoremas booleanos fundamentais**.

Teoremas Booleanos e de De Morgan

Os **teoremas booleanos** são **regras** para tratarmos de **circuitos lógicos**. São eles:

Teoremas para a Operação AND:

1. $X \cdot 0 = 0$, ou seja, qualquer variável booleana que faça uma operação **AND** com um valorlógico 0, terá como saída o valor lógico 0.

2. $X \cdot 1 = X$, ou seja, qualquer variável booleana que faça uma operação AND com um valor lógico 1, terá como saída a própria variável booleana.

3. $X \cdot X = X$, ou seja, qualquer variável booleana que faça uma operação **AND** consigomesma, terá como saída o valor da própria variável booleana.

4. $X \cdot \bar{X} = 0$ ou seja, qualquer variável booleana que faça uma operação **AND** consigo mesma negada, terá como saída o valor lógico 0.

Teoremas para a Operação OR:

1. $X + 0 = X$, ou seja, qualquer variável booleana que faça uma operação **OR** com um valorlógico 0, terá como saída a própria variável.

2. $X + 1 = 1$, ou seja, qualquer variável booleana que faça uma operação **OR** com um valor lógico 1, terá como saída sempre o valor lógico 1.

3. $X + X = X$, ou seja, qualquer variável booleana que faça uma operação **OR** consigo mesma, terá como saída o valor da própria variável booleana.

4. $X+\overline{X}=1$, ou seja, qualquer variável booleana que faça uma operação **OR** consigo mesma negada, terá como saída o valor lógico 1.

Além destes, há ainda um outro teorema que é **fundamental** para a implementação de circuitos lógicos eficientes, o chamado **Teorema de De Morgan**. Este teorema nos diz que:

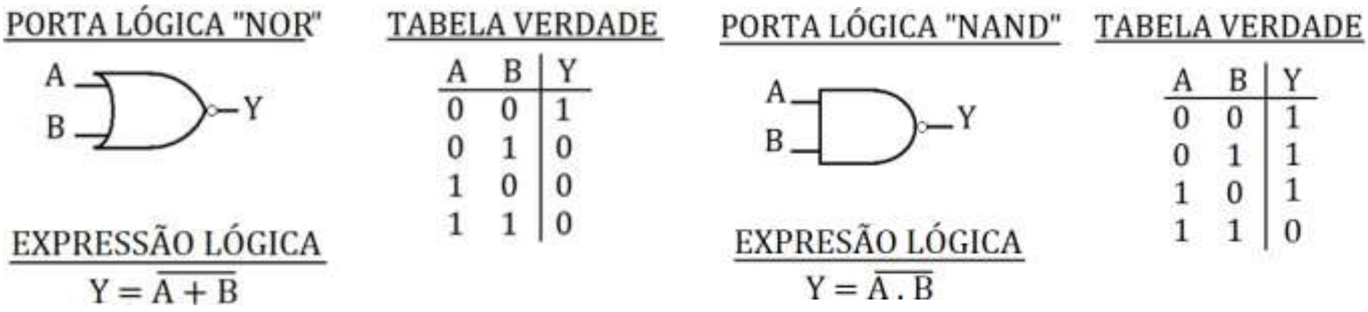
$$\underbrace{\overline{(A\cdot B)}}_{Operação\ NAND}=\overline{A}+\overline{B} \qquad \underbrace{\overline{A+B}}_{Operação\ NOR}=\overline{A}\cdot\overline{B}$$

Todas as **expressões booleanas** consistem em combinações das operações básicas aqui vistas (**OR**, **AND** e **NOT**). Entretanto, é possível implementar qualquer expressão booleana empregando **APENAS** portas **NAND** e portas **NOR**. A esta característica se dá o nome de universalidade das portas **NAND** e **NOR**. E tal universalidade advém diretamente do**Teorema de De Morgan**.

Os **circuitos integrados** que implementam as **funções lógicas** possuem **não uma**, **porém várias portas lógicas implementadas em si**. Se em um determinado circuito lógico tivermos a necessidade de implementar apenas uma porta **OR**, e diversas outras portas **AND** e **NOT**, teremos "*desperdiçado*" espaço em nossa placa com um circuito integrado que faráapenas uma função.

Em vez disso, podemos **trocar todas as nossas portas lógicas** por suas equivalentes **NAND** ou **NOR** e implementarmos em nossa placa **apenas** portas **NAND** ou **NOR**. A probabilidade de que economizemos espaço em nossa placa é **muito maior**. Isso gerará também a chamada **economia de escala**, uma vez que se nosso circuito for ser implementado em uma linha de montagem, poderemos comprar somente circuitos integrados de portas **NAND**, gerando economia de custos com o fornecedor, pois agora compraremos inúmeros circuitos integrados de portas **NAND**, em vez de conjuntos menores de circuitos integrados de portas **NOT**, **AND** e **OR**.

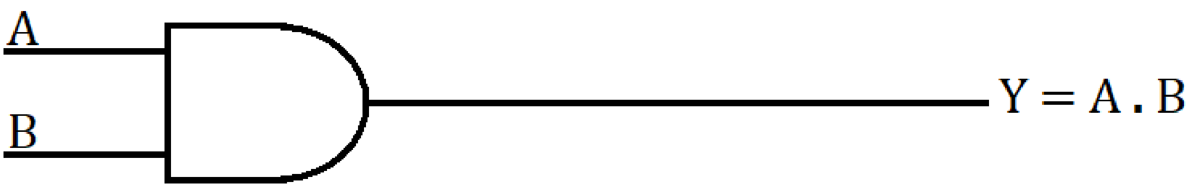
Mas, antes de prosseguirmos, vamos apresentar as portas **NAND** e **NOR**, do mesmo jeito que apresentamos as portas **OR**, **AND**, **NOT** e **XOR**.

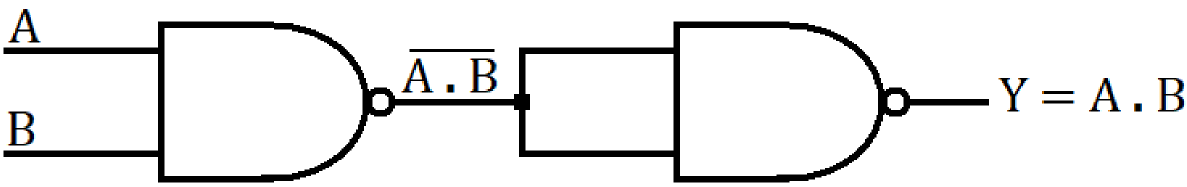


A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

Exemplo: *Como implementar uma porta **AND** através de portas **NAND***? Perceba que na representação de cima a porta **AND** possui duas entradas: A e B.





Na representação acima temos as mesmas duas entradas A e B entrando numa porta **NAND**. A saída é $\overline{A \cdot B}$. Em seguida, fazemos com que esta saída $\overline{A \cdot B}$ entre nas duas entradas da **NAND** posterior.

*Dos teoremas de portas **AND** temos que X . X= X, lembra*?

Acontece que agora é uma porta **NAND**, que nada mais é do que uma porta **AND** com uma outra porta **NOT** **acopladada à sua saída**. Sendo assim, o resultado da expressão é o mesmo Y = A . B da primeira.

RESUMO

1. Bases Numéricas e Conversões

- Binária (base 2), Octal (base 8), Decimal (base 10) e Hexadecimal (base 16).
- Conversões entre bases: binária $\leftrightarrow$ octal (grupo de 3 bits), binária $\leftrightarrow$ hexadecimal (grupo de 4 bits).
- Conversão de decimal para outras bases: divisões sucessivas anotando o resto.
- Conversão de outras bases para decimal: multiplicação de cada dígito pela base elevada à sua posição.

2. Operações com Números Binários

- Soma binária com carry (*vai 1*).
- Subtração com complemento de 2: inverte os bits (complemento de 1) e soma 1.
- Overflow: ocorre quando o resultado não pode ser representado com o número de bits disponíveis.

3. Variáveis e Expressões Booleanas

- Variáveis booleanas podem assumir valores 0 ou 1.
- Tabelas verdade definem todas as combinações possíveis entre variáveis de entrada.

4. Portas Lógicas Fundamentais

- AND (E), OR (OU), NOT (inversor), XOR (OU-exclusivo).
- Portas NAND e NOR: portas universais.
- Portas podem ter mais de 2 entradas.

5. Teoremas Booleanos

- $\mathrm{X} \cdot 0 = 0, \quad \mathrm{X} \cdot 1 = \mathrm{X}, \quad \mathrm{X} \cdot \mathrm{X} = \mathrm{X}, \quad \mathrm{X} \cdot \mathrm{X} = 0$
- $\mathrm{X} + 0 = \mathrm{X}, \quad \mathrm{X} + 1 = 1, \quad \mathrm{X} + \mathrm{X} = \mathrm{X}, \quad \mathrm{X} + \mathrm{X} = 1$
- De Morgan: $(X \cdot Y)' = X' + Y'$ e $(X + Y)' = X' \cdot Y'$

6. Implementação de Portas com NAND/NOR

- Qualquer expressão lógica pode ser implementada apenas com portas NAND ou NOR.
- Vantagens: economia de espaço e custo em circuitos integrados.

Resumo Final - O que mais cai em concursos

- **Conversão entre bases numéricas.**
- Soma e subtração binária com complemento de 2.
- **Tabelas verdade e expressão booleana equivalente.**
- Teoremas booleanos e De Morgan.
- **Identificação e funcionamento das portas lógicas.**
- Implementação de circuitos com NAND e NOR.

QUESTÕES COMENTADAS

1. **(CESGRANRIO - 2014 - Petrobras - Técnico(a) de Projetos, Construção e Montagem Júnior - Eletrônica)**

A negação da função lógica representada pela expressão $ABC + A + B + C$ equivale a

A) $\overline{ABC} + \bar{A} + \bar{B} + \bar{C}$
B) $\overline{ABC} + \overline{A + B + C}$
C) $\bar{A} + \bar{B} + \bar{C} + \overline{A + B + C}$
D) $\bar{A} \cdot \bar{B} \cdot \bar{C}$
E) $\overline{ABC}$

Gabarito: Letra D

Comentário:

Nesta questão, o que a banca quer de você é a negação de uma expressão lógica. Vamos juntos, passo a passo. Considere:

$$F = ABC + A + B + C$$

Passo 1 – Entendendo o que a questão pede:

Queremos encontrar a negação de F, ou seja, a expressão: F' ($equivalente\ a\ \bar{F}$)

Dica de prova: Sempre que a questão pedir negação de função lógica, pense direto em aplicar o Teorema de De Morgan, ok?

Passo 2 – Aplicando o Teorema de De Morgan:

O Teorema de De Morgan nos diz o seguinte:

A negação de uma soma lógica (OR) se transforma em uma multiplicação lógica (AND) com todos os termos negados. E vice-versa.

Aplicando isso na nossa expressão:

F' = (ABC + A + B + C)'

Transformamos o OR em AND, e negamos cada pedaço:

F' = (ABC)' . (A)' . (B)' . (C)'

Passo 3 – Agora vamos simplificar cada termo:

A negação de (ABC) usando De Morgan fica:

(ABC)' = A' + B' + C'

Então agora temos:

F' = (A' + B' + C') . A' . B' . C'

Passo 4 – Fazendo a análise final:

Agora preste atenção a uma dica que eu sempre dou:

Sempre que tivermos uma multiplicação lógica (AND) com os termos A', B', e C', qualquer outro fator que já contenha eles não muda o resultado final.

Por quê? Porque ao fazer a multiplicação, se um termo já contém A' . B' . C', a soma que veio antes (do De Morgan) não vai eliminar esses fatores.

Então, podemos simplificar para:

$$\boldsymbol{F' = A'.B'.C'} = \bar{A} \cdot \bar{B} \cdot \bar{C}$$

Passo 5 – Conferindo com as alternativas:

A alternativa correta é:

Alternativa D: $\bar{A} \cdot \bar{B} \cdot \bar{C}$

Portanto, nesta questão, o mais importante era aplicar o Teorema de De Morgan com calma, termo por termo, e depois fazer uma boa análise da simplificação.

Dica final para concursos: Sempre que aparecer "negação de expressão lógica", pense direto no De Morgan e siga o passo a passo.

2. (CESGRANRIO - 2014 - Petrobras - Técnico(a) de Projetos, Construção e Montagem Júnior - Eletrônica)

O mapa de Karnaugh mostrado acima representa a função lógica]

AB \ CD	00	01	11	10
00	0	0	1	1
01	0	0	1	1
11	1	1	0	0
10	1	1	0	0

A) $AC + \bar{B}D$
B) $AC + A\bar{C}$
C) $BD + B\bar{D}$
D) $A \oplus C$
E) $B \oplus D$

Gabarito: Letra D

Comentário:

Vamos resolver a questão detalhadamente, analisando o mapa de Karnaugh e encontrando a função lógica correspondente.

Passo 1 – Identificação das Variáveis:

O mapa de Karnaugh fornecido trabalha com quatro variáveis:

- A e B (linhas)

- C e D (colunas)

A disposição das linhas (AB) segue a ordem: 00, 01, 11, 10

A disposição das colunas (CD) segue a ordem: 00, 01, 11, 10

Passo 2 – Análise dos valores da função:

Agora vamos olhar para o mapa e **identificar onde temos "1"**, ou seja, onde a saída da função é verdadeira.

Observando o mapa, vemos que a função é 1 para as seguintes posições (com base nas combinações de AB e CD):

- AB=00, CD=10 → F=1

- AB=01, CD=11 → F=1

- AB=11, CD=00 → F=1

- AB=11, CD=01 → F=1

- AB=10, CD=00 → F=1

- AB=10, CD=01 → F=1

Passo 3 – Obtenção da expressão Booleana:

Com base nessas posições, podemos escrever os mintermos correspondentes e depois simplificar.

Porém, ao analisar os agrupamentos possíveis e a distribuição dos 1s no mapa de Karnaugh, percebemos que os valores "1" ocorrem justamente nas situações onde as variáveis **A** e **C** são diferentes entre si. Essa configuração é típica de uma **operação XOR (OU exclusivo)**.

- E os outros termos (B, D)?

Se a função fosse sensível a B e D, o mapa mostraria os 1s em posições específicas de B e D, e não em todas.

Se os 1s aparecem em todas as colunas de B e D para um mesmo A e C, então essas duas variáveis (B e D) não influenciam o resultado.

Por isso, ao simplificar a função, só aparecem A e C na expressão.

Passo 4 – Identificação da expressão final:

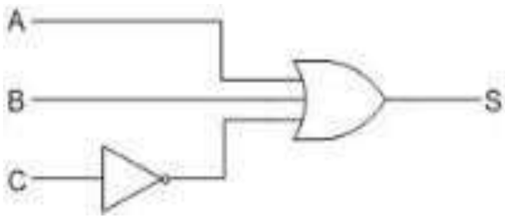
A forma mais simplificada da função, ao analisar o mapa de Karnaugh, é:

$$F = A \oplus C$$

Onde $\oplus$ representa a operação lógica XOR (OU exclusivo), indicando que a saída será 1 quando A e C forem diferentes.

Portanto, o gabarito correto é a **Alternativa D**.

3. (CESGRANRIO - 2014 - Petrobras - Técnico(a) de Projetos, Construção e Montagem Júnior - Elétrica)



Qual a função lógica S que representa o circuito acima?

A) $A + B + C$
B) $A + B + \bar{C}$
C) $A + B + C$
D) $\bar{A} + B + C$
E) $A \cdot B \cdot \bar{C}$

Gabarito: Letra B

Comentário:

Analisando o Circuito Lógico:

O circuito é composto por três tipos de portas lógicas: OR, NOT e entradas diretas. Vamos analisar com atenção cada parte do circuito:

O circuito possui:

- Três entradas: A, B e C.

- Um inversor (NOT) ligado à entrada C, gerando a saída ¬C (ou C').

- Um portão OR (OU) com três entradas: A, B e a saída do inversor (¬C).

As entradas do OR são:

- Entrada direta de A.

- Entrada direta de B.

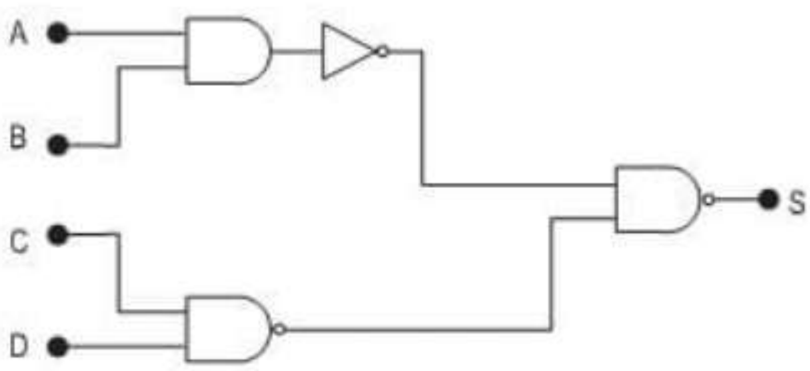
- Saída do inversor aplicado a C (ou seja, C negado → C').

Logo, o portão OR realiza a seguinte operação lógica:

$$S = A + B + C' = A + B + \bar{C}$$

Portanto, a alternativa correta é: **Letra B**

4. (CESGRANRIO - 2014 - Petrobras - Técnico(a) de Manutenção Júnior - Eletrônica)



O circuito digital acima representa um circuito composto por três portas lógicas com quatro entradas A, B, C e D e uma saída S.

Qual é a expressão booleana da saída S?

A) $\overline{\overline{(A \cdot B)} + \overline{(B \cdot D)}}$

B) $\overline{(A \cdot B)} + \overline{(C \cdot D)}$

C) $\overline{\overline{(A \cdot D)} \cdot \overline{(C + D)}}$

D) $\overline{(A \oplus B)} + \overline{(C \cdot D)}$

E) $\overline{\overline{(A \cdot B)} \cdot \overline{(C \cdot D)}}$

Gabarito: Letra E

Comentário:

Vamos resolver essa questão passo a passo:

Passo 1 – Identificação das portas lógicas no circuito:

O circuito é composto por:

- Duas portas AND iniciais:

• Primeira AND recebe as entradas A e B, gerando a saída A . B.

• Segunda NAND recebe as entradas C e D, gerando a saída (C . D)'.

- Uma porta NOT (Inversores):

  • Porta NOT inverte a saída da primeira AND, gerando (A . B)'.

- Uma porta NAND final:

  • Essa porta AND recebe como entradas as duas saídas das portas anteriores: (A . B)' e (C . D)'.

  • A saída S será a multiplicação lógica dessas duas saídas seguida de uma negação: ((A . B)' . (C. D)')'

Passo 2 – Análise com Teorema de De Morgan:

Não há necessidade de aplicar o Teorema de De Morgan neste caso, pois o circuito já nos forneceu diretamente a forma final da função:

A saída é o resultado da multiplicação entre os complementos das duas portas AND.

Expressão final da saída S:

$$S = ((A\,.\,B)'\,.\,(C\,.\,D)')' = \overline{\overline{(A \cdot B)} \cdot \overline{(C \cdot D)}}$$

Portanto, a alternativa correta é:

Alternativa E

5. (CESGRANRIO - 2014 - Petrobras - Técnico(a) de Manutenção Júnior - Eletrônica)

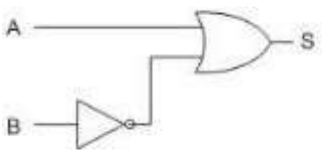
A Tabela verdade a seguir refere-se a um circuito lógico com duas entradas **A** e **B** e uma saída **S**.

A Tabela verdade a seguir refere-se a um circuito lógico com duas entradas **A** e **B** e uma saída **S**.

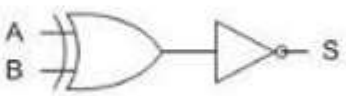
A	B	S
0	0	1
0	1	0
1	0	0
1	1	1

Qual é o circuito lógico que a Tabela representa?

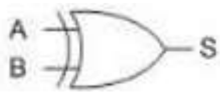
A)



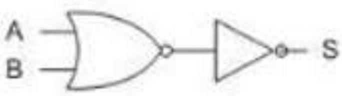
B)



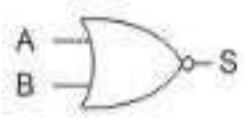
C)



D)



E)



Gabarito: Letra B

Comentário:

Nesta questão, quero te dar uma dica valiosa: sempre que aparecer uma tabela verdade na prova, o segredo é identificar o padrão de comportamento da saída em relação às entradas. Saber reconhecer padrões típicos como XOR, XNOR, AND ou OR pode te fazer ganhar tempo e acertar com segurança. Vamos juntos resolver passo a passo!

Passo 1 – Analisando a Tabela Verdade:

Observando a tabela, a saída S é igual a 1 nos seguintes casos:

A	B	S
0	0	1
0	1	0
1	0	0
1	1	1

- Quando A = 0 e B = 0 → S = 1.

- Quando A = 1 e B = 1 → S = 1.

E a saída é 0 quando:

- A = 0 e B = 1 → S = 0.

- A = 1 e B = 0 → S = 0.

Passo 2 – Identificando a Função Lógica:

Perceba que a saída é 1 quando as entradas são IGUAIS (00 ou 11) e 0 quando são diferentes.

Esse é exatamente o comportamento de uma porta lógica XNOR (OU Exclusivo Negado).

Passo 3 – Construindo a Expressão Booleana:

A expressão da porta XNOR pode ser escrita como:

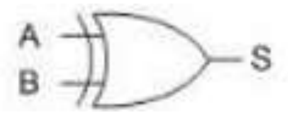
$$S = A \cdot B + A' \cdot B'$$

Ou de forma simplificada, usando a notação de porta lógica:

$$S = A \odot B$$

Passo 4 – Ligando com o circuito da questão:

A alternativa B mostra uma porta XOR seguida de uma porta NOT, o que na prática corresponde a uma porta XNOR.



Portanto, a alternativa correta é: **Alternativa B**

6. (CESGRANRIO - 2012 - Petrobras - Técnico de Manutenção Júnior - Eletrônica-2012)

Dada a expressão booleana

$$F(w, x, y, z) = x + x \cdot y' \cdot z + x' \cdot y' \cdot z + w \cdot x + w' \cdot x + x' \cdot y'$$

na qual "+" significa "ou lógico", "." significa "e lógico" e " ' " significa a "negação lógica", qual das expressões é equivalente à expressão dada?

A) $x + y + z$
B) $x + y'$
C) $x' + y$
D) $x' + y' + z'$
E) $x' + y'$

Gabarito: Letra B

Comentário:

Esse tipo de questão é clássico em concursos! O segredo aqui é ir com calma e trabalhar a simplificação usando as propriedades da Álgebra Booleana. Uma dica importante: quando a expressão parecer grande e confusa, foque em identificar os termos repetidos e aplicar identidades conhecidas como a Distributiva e a Identidade. Agora vamos ao passo a passo:

Expressão Original:

$$F(w,x,y,z) = x + x.y'.z + x'.y'.z + w.x + w'.x + x'.y'$$

Aplicação da Distribuição e Identidade:

1. Combine os termos que possuem x:

$$x + w.x + w'.x = x.(1 + w + w') = x.1 = x$$

Agora a expressão fica:

$$F = x + x.y'.z + x'.y'.z + x'.y'$$

Combine os termos com x' aplicando a distribuição:

$$x' . (y' . z + y') = x' . y'$$

Explicando o motivo:

$$y' . z + y' = y' (z + 1) = y'$$

Agora temos:

$$F = x + x' . y'$$

Expressão Final Simplificada:

$$F = x + y'$$

Portanto, a alternativa correta é:

Alternativa B

7. (CESPE / CEBRASPE - 2024 - Petrobras - Técnico Júnior - Ênfase: Manutenção - Instrumentação)

Assinale a opção que corresponde à conversão do número 00011111 de base binária para base hexadecimal.

A) 25
B) 1F
C) 4E
D) 39
E) 2F

Gabarito: Letra B

Comentário:

Passo 1: Dividir o Número Binário em Grupos de 4 Bits

Primeiro, dividimos o número binário em grupos de 4 bits, começando da direita para a esquerda. Se necessário, adicionamos zeros à esquerda para completar os grupos de 4 bits.

$$0001\ 1111$$

Passo 2: Converter Cada Grupo de 4 Bits para Decimal

Agora, convertemos cada grupo de 4 bits para seu valor decimal correspondente.

<u>Grupo 1 (0001):</u>

- $\mathbf{0 * 2^3 = 0}$
- $\mathbf{0 * 2^2 = 0}$
- $\mathbf{0 * 2^1 = 0}$
- $\mathbf{0 * 2^0 = 1}$

Portanto, 0001 em binário é 1 em decimal.

Grupo 2 (1111):

- $1 * 2^3 = 8$
- $1 * 2^2 = 4$
- $1 * 2^1 = 2$
- $1 * 2^0 = 1$

Somando os valores:

$$8 + 4 + 2 + 1 = 15$$

Portanto, 1111 em binário é 15 em decimal, que é representado como F em hexadecimal.

Dica extra: Tabela de Conversão – Decimal, Binário e Hexadecimal

Essa é uma tabela essencial para quem está estudando sistemas de numeração. Saber de cabeça a equivalência entre decimal, binário e hexadecimal vai te ajudar a resolver não só essa, mas várias outras questões de conversão e análise de circuitos digitais em provas de concurso.

Decimal	Binário	Hexadecimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

Dica de prova: Sempre que cair questão de conversão ou mapeamento de bits, lembre-se dessa tabela. Memorizar pelo menos os números de 0 a 15 pode te salvar em várias situações!

Passo 3: Combinar os Valores Hexadecimais

Agora, combinamos os valores hexadecimais dos grupos:

$$0001\ 1111 \rightarrow 1F$$

A conversão correta do número 00011111 da base binária para a base hexadecimal é **1F**. Portanto, a resposta correta é **letra B**.

8. (CESPE / CEBRASPE - 2024 - Petrobras - Técnico Júnior - Ênfase: Manutenção - Instrumentação)

Em relação à eletrônica digital, julgue o próximo item.

Em uma porta lógica AND de duas entradas, a saída será igual a 1 sempre que as entradas também forem iguais a 1.

A) Certo

B) Errado

Gabarito: Certo

Comentário:

Esse tipo de questão é uma verdadeira "clássica de prova". Saber o funcionamento básico das portas lógicas é essencial para quem está se preparando para concursos na área de eletrônica digital.

Funcionamento da Porta AND:

A porta lógica AND realiza uma multiplicação lógica entre as entradas. A regra é simples e sempre cai em prova:

- A saída será 1 (verdadeira) somente quando TODAS as entradas forem 1 (verdadeiras).

- Se pelo menos uma entrada for 0 (falsa), a saída será 0 (falsa).

📋 Tabela Verdade da Porta AND:

A	B	Saída (A . B)
0	0	0
0	1	0
1	0	0
1	1	1

Análise da Questão:

A frase afirma que a saída será 1 sempre que as entradas forem iguais a 1. Isso está perfeitamente de acordo com o funcionamento da porta lógica AND.

Conclusão: A afirmação está correta. Portanto, a resposta certa é a **letra A** (**Certo**).

9. (CESPE / CEBRASPE - 2024 - INPI - Pesquisador Em Propriedade Industrial – Área: P4)

No que se refere a aplicações e princípios da eletrônica digital, julgue o item que segue.

Conforme o primeiro teorema de De Morgan, o complemento do produto de duas variáveis é igual à soma dos complementos das variáveis individuais; em termos de portas, ele pode ser expresso por $\overline{A \cdot B} = \bar{A} + \bar{B}$

A) Certo
B) Errado

Gabarito: Certo

Comentário:

Esse é o tipo de questão que cobra a aplicação prática de um dos teoremas mais importantes da Álgebra Booleana: o Primeiro Teorema de De Morgan. Saber aplicar esse teorema é fundamental para resolver questões de eletrônica digital. Vamos entender de forma simples e organizada:

Primeiro Teorema de De Morgan:

O teorema afirma que:

$$\overline{A \cdot B} = \bar{A} + \bar{B}$$

Ou seja:

- O complemento de um produto (AND) é igual à soma (OR) dos complementos individuais.

O que isso significa na prática?

1. Se temos uma porta AND com entradas A e B e aplicamos uma NOT na saída, teremos:

Saída = $\overline{A \cdot B}$

2. Segundo o Teorema de De Morgan, isso é o mesmo que fazer uma NOT em A, uma NOT em B, e depois fazer um OR entre os resultados:

Saída = $\bar{A} + \bar{B}$

Representação usando portas lógicas:

- Porta AND + NOT (NAND): $\overline{A \cdot B}$

- OU: Porta OR após duas NOTs: $\bar{A} + \bar{B}$

📋 Tabela Verdade para Comprovação:

A	B	A . B	¬(A . B)	¬A	¬B	¬A + ¬B
0	0	0	1	1	1	1
0	1	0	1	1	0	1
1	0	0	1	0	1	1
1	1	1	0	0	0	0

Portanto, os valores da coluna $\overline{A \cdot B}$ são exatamente iguais aos da coluna $\bar{A} + \bar{B}$, comprovando o Teorema de De Morgan. Portanto, a afirmação do enunciado está correta.

Resposta: Letra A (Certo)

10. (CS-UFG / UFG - 2019)

Tratando-se de portas lógicas e álgebra booleana, a tabela abaixo diz respeito à operação

A	B	X
0	0	1
0	1	0
1	0	0
1	1	1

A) $X = \overline{A + B}$
B) $X = \overline{A \oplus B}$
C) $X = A + B$
D) $X = A \oplus B$

Gabarito: Letra B

Comentário:

Esse tipo de questão é clássico em provas que envolvem portas lógicas e álgebra booleana. A grande dica aqui é: olhe sempre para o padrão de saída da tabela verdade antes de tentar escolher a alternativa.

Análise da Tabela Verdade:

A	B	X
0	0	1
0	1	0
1	0	0
1	1	1

Ou seja, a saída X só é igual a 1 quando:

- As duas entradas A e B forem IGUAIS (00 ou 11).

E X é 0 quando as entradas forem DIFERENTES (01 ou 10).

Identificando a Operação:

Esse comportamento corresponde exatamente ao que chamamos de uma porta XNOR (OU Exclusivo Negado).

- Enquanto o XOR (A ⊕ B) fornece 1 quando as entradas são diferentes,

- A XNOR ($\overline{A \oplus B}$) faz o contrário: fornece 1 quando as entradas são iguais.

Portanto, a alternativa correta é a letra B, pois ela representa a operação XNOR (OU Exclusivo Negado). Isso confirma o que observamos na tabela verdade.

Resposta final: Letra B (XNOR).

11. (CS-UFG / UFG - 2019)

A representação do número decimal 513 em

octal é:

a) 201.

b) 0101 0001 0011.

c) 1001.

d) 0001 0000 0001.

Gabarito: Letra C

Comentário:

Nesta questão é necessário realizar a conversão de bases numéricas. A dica de ouro é: o processo de conversão de decimal para octal é realizado por meio de divisões sucessivas.

Como fazer a conversão de decimal para octal:

Para converter um número da base 10 (decimal) para a base 8 (octal), siga os passos:

- 1. Divida o número decimal por 8.
- 2. Anote o resto da divisão.
- 3. Divida o quociente novamente por 8.
- 4. Repita o processo até que o quociente seja zero.
- 5. Por fim, leia os restos de baixo para cima (do último para o primeiro).

Exemplo aplicado ao número 513:

Divisão	Quociente	Resto
513 ÷ 8	64	1
64 ÷ 8	8	0
8 ÷ 8	1	0
1 ÷ 8	0	1

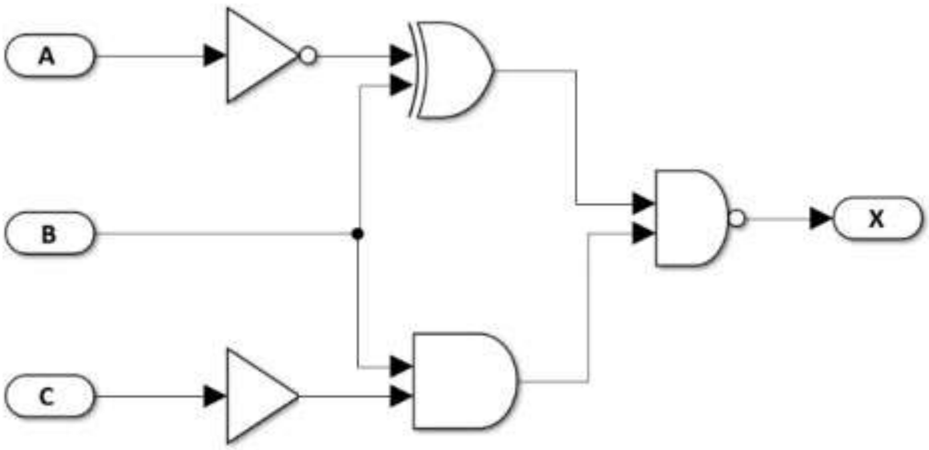
Agora, lendo os restos de baixo para cima, temos: 1001 (base 8).

Portanto, o número decimal 513 equivale a 1001 em octal, o que corresponde à:

Alternativa C

12. (CS-UFG / UFG - 2019)

Considere o circuito combinacional a seguir.



A expressão para a saída X é:

a) $X = \overline{(\overline{A} \oplus B) \cdot (B \cdot C)}$.

b) $X = \overline{(\overline{A} + B) \cdot (B \cdot C)}$.

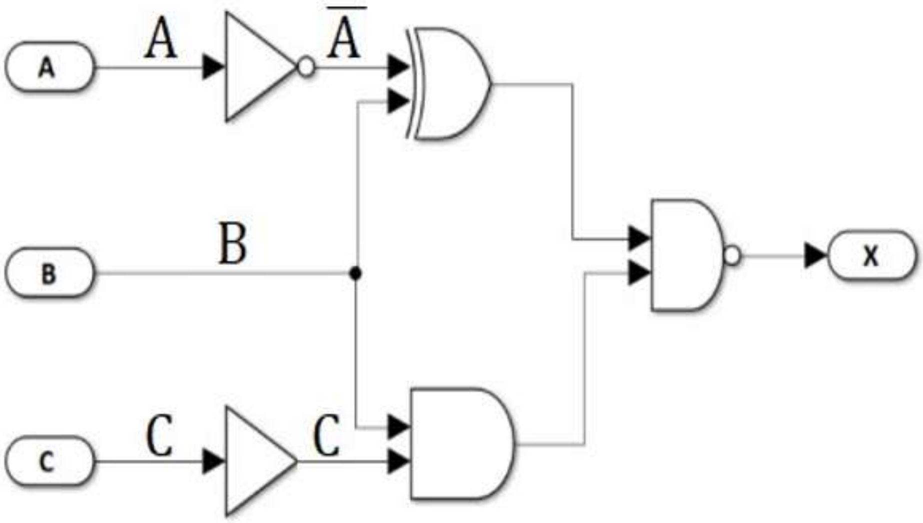
c) $X = \overline{(\overline{A} \oplus B) \cdot (B \cdot \overline{C})}$.

d) $X = \overline{(\overline{A} + B) \cdot (B \cdot \overline{C})}$.

Gabarito: Letra A

Comentário:

Vamos redesenhar o circuito e ir montando a resposta passo a passo.

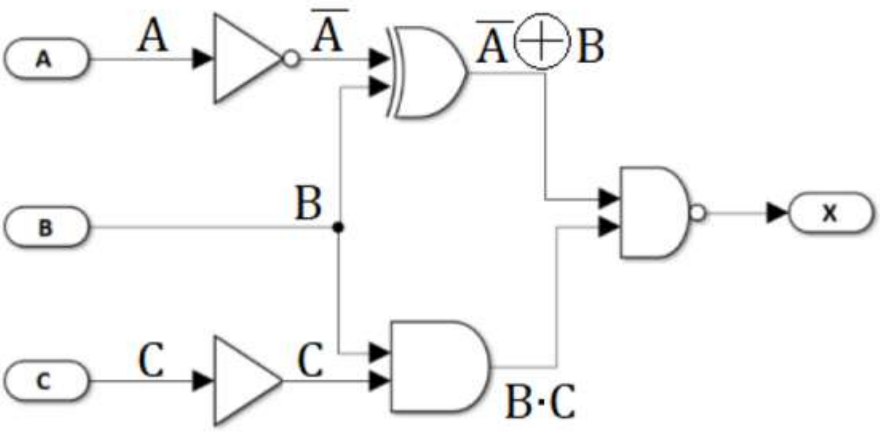


Primeiramente, o sinal A adentra o **inversor lógico** (**porta NOT**), gerando como saída $\overline{A}$.

Em segundo lugar, o sinal B entra junto com $\overline{A}$ na **porta XOR** (**ou-exclusivo**).

E, em terceiro lugar, o sinal C entra em um bloco que é apenas um buffer, ou seja, não produz efeito nenhum, so**mente cria uma interface de separação** entre o sinal C de entrada e o sinal C de saída.

Próximo passo:



Incluiu-se o sinal $\overline{A} \oplus B$ na saída da porta **XOR** e o sinal $B \cdot C$ na saída da porta **AND**.

Estes dois sinais fazem a entrada da porta **NAND** que vem em seguida.

A seguir se apresenta o **resultado final** X do circuito digital.

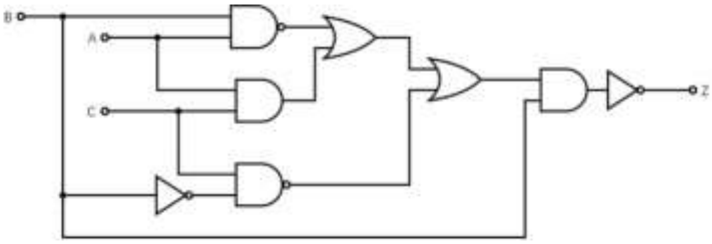
$$X = \overline{(\overline{A} \oplus B) \cdot B \cdot C}$$

Gabarito: Letra A.

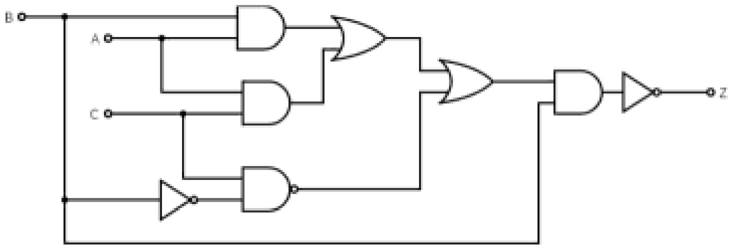
13. (FUNDEP/Prefeitura de Uberlândia - 2019)

Considerando quatro variáveis binárias A, B, C e Z, assinale a alternativa que apresenta o circuito lógico que pode ser utilizado para implementar a expressão: $Z=\overline{\left(\overline{AB}+CA+\overline{\overline{B}C}\right)B}$.

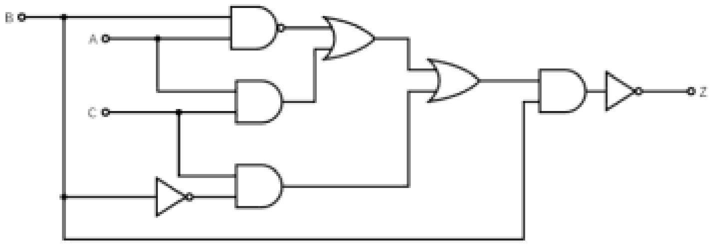
a)



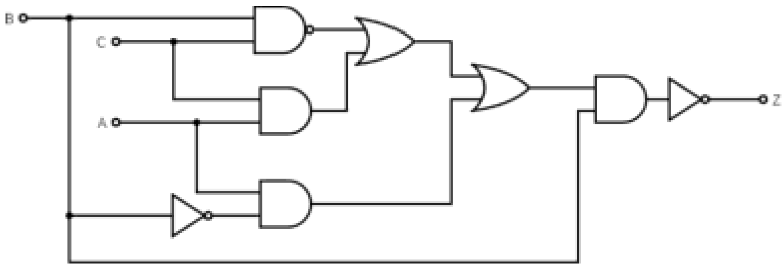
b)



c)



d)



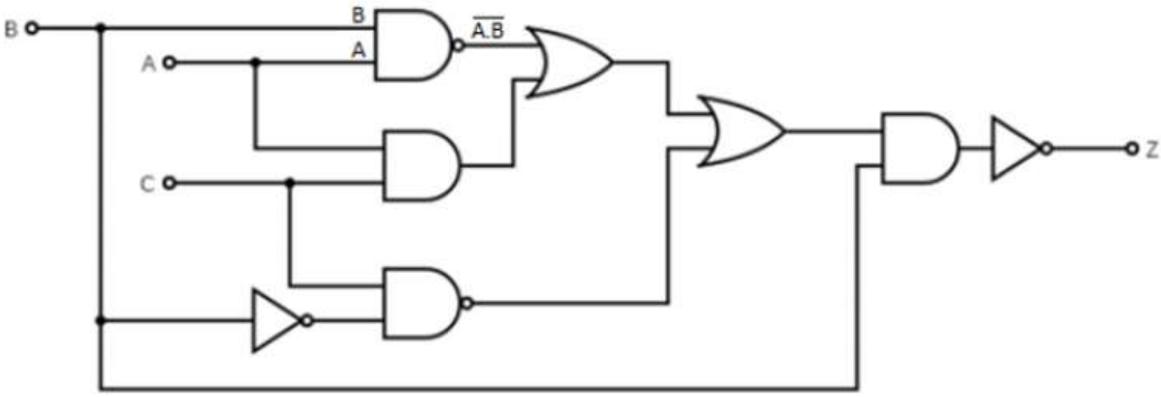
Gabarito: Letra A

Comentário:

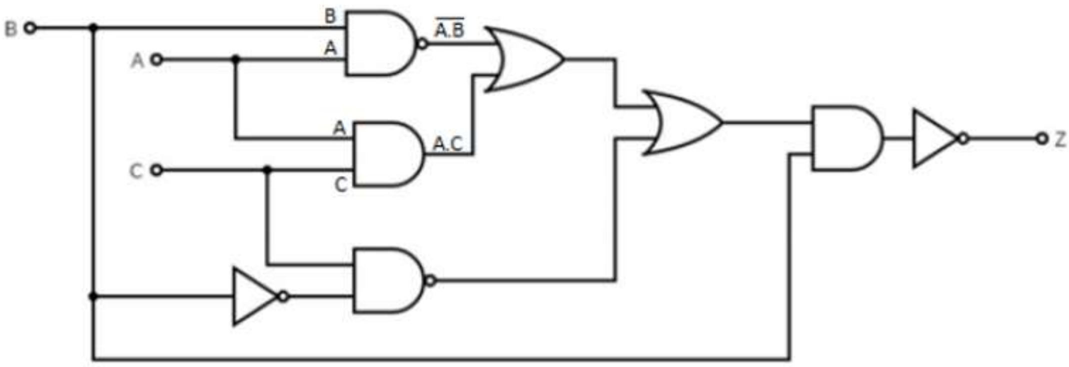
Para resolver esta questão é necessário **testar** os **circuitos lógicos** das alternativas no sentido de verificar se a **expressão lógica** que ela gera coincide com a fornecida pelo cabeçalho.

Comecemos pelo **circuito lógico** da alternativa A.

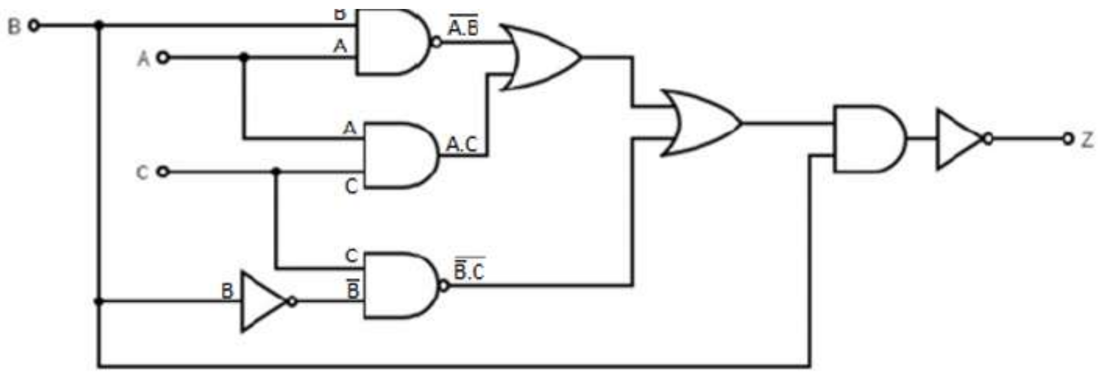
Nossa análise inicia da esquerda para a direita e de cima para baixo. A primeira **porta lógica** é uma **NAND** (ou um seja, uma função "E" negada). Nela adentram os **sinais** "B" e "A". O **resultado lógico** desta combinação é $\overline{A.B}$ (ou seja, a negação da associação entre os sinais A e B). Vamos escrever a expressão.



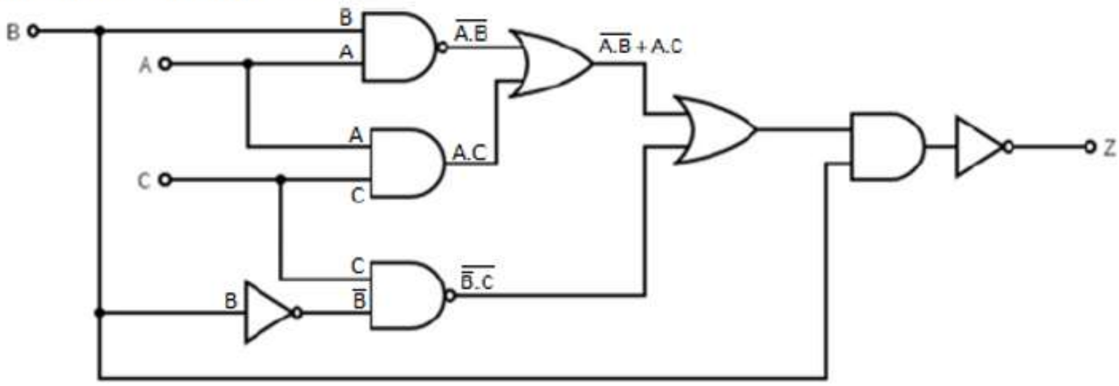
A seguir analisamos a **porta lógica** logo abaixo. Trata-se de uma porta **AND** (função "E"). Nela adentram os sinais "A" e "C". O **resultado lógico** desta combinação é $A.C$. Vamos escrever esta expressão.



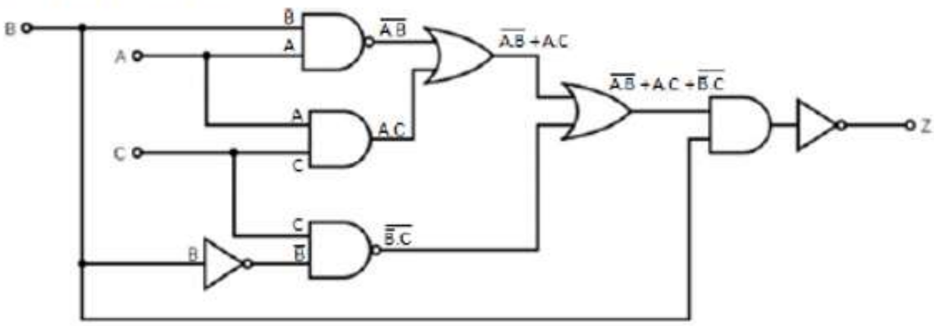
A seguir analisamos a **porta lógica** abaixo. Trata-se de uma porta **NAND**. Nela adentram os sinais "C" e "B" negado. O **resultado lógico** desta combinação é $\overline{\overline{B}.C}$. Vamos escrever esta expressão.



Subindo novamente e indo para a próxima **porta lógica**. Trata-se de uma porta **OR** (função "OU"). Nela adentram os sinais $\overline{A.B}$ e $A.C$. O **resultado lógico** desta combinação é $\overline{A.B}+A.C$. Vamos escrever esta expressão.



A seguir, a próxima **porta lógica** também é uma **OR**, com sinais de entrada dados por $\overline{A.B}+A.C$ e por $\overline{B.C}$. O **resultado lógico** desta expressão é $\overline{A.B}+A.C+\overline{B.C}$. Vamos escrever esta expressão.



Por último, o sinal $\overline{A.B}+A.C+\overline{B.C}$ é combinado com o sinal "B" em uma **porta lógica AND** associada a uma **porta lógica NOT**, o que gera uma **porta lógica NAND**, cujo resultado é a expressão Z que a questão busca. Sendo assim, tem-se que $Z=\overline{(\overline{A.B}+A.C+\overline{\overline{B}.C}).B}$, que bate com a expressão do cabeçalho e é o gabarito da questão. Portanto, a **alternativa correta é a letra A**.

14. (FUNDEP/Prefeitura de Uberlândia - 2019)

Em relação aos sistemas digitais, a conversão decimal binário é de fundamental importância. Assinale a alternativa que apresenta o equivalente binário ao número decimal 35.

a) 100001

b) 100111

c) 110001

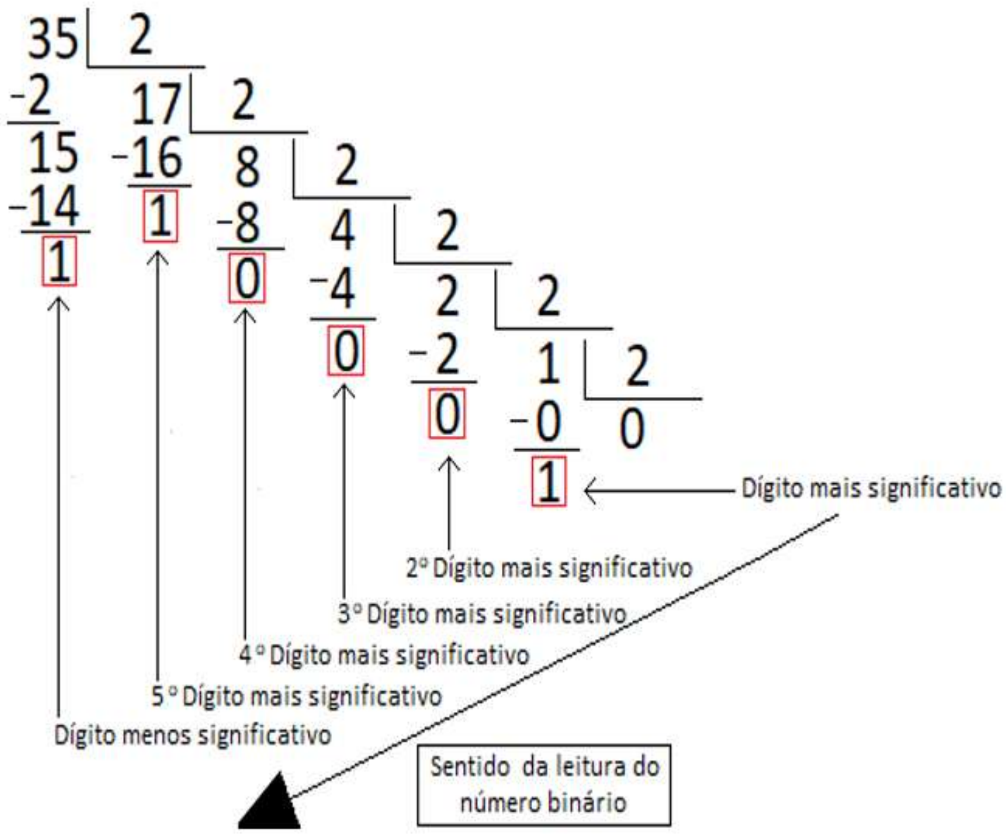
d) 100011

Gabarito: Letra D

Comentário:

Para se fazer a conversão de um **número decimal** para um **número binário** é necessário fazer uma **sucessão de divisões por 2.** O resultado é então lido ao contrário, conforme

segue.



Portanto, o **número 35 em decimal é representado em binário pelo número 100011**.

Assim, a **alternativa correta é a letra D**.

15. (FUNDEP/Prefeitura de Uberlândia - 2019)

O sistema de base 16, ou hexadecimal, utiliza 16 símbolos para a representação numérica. Assinale a alternativa que apresenta a representação decimal para o número hexadecimal 5D3B.

a) 32

b) 23866

c) 23867

d) 512

Gabarito: Letra C

Comentário:

A conversão de um **número hexadecimal** para **decimal** pode ser da seguinte maneira.

$$(5D3B)_{16} = \left(5 \times 16^3 + D \times 16^2 + 3 \times 16^1 + B \times 16^0\right)_{10}$$

Entendendo a Conversão:

Ok, mas como multiplicar "**D**" e "**B**"??? Como assim?? Bom, você lembra que os números hexadecimais são:

$$0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F$$

E que:

- **A** em hexadecimal equivale a **10** em decimal.
- **B** em hexadecimal equivale a **11** em decimal.
- **C** equivale a **12**, **D** equivale a **13**, **E** equivale a **14**, e **F** equivale a **15**.

Substituindo os valores correspondentes:

$$(5D3B)_{16} = \left(5 \times 16^3 + 13 \times 16^2 + 3 \times 16^1 + 11 \times 16^0\right)_{10}$$

Efetuando as multiplicações e somas:

$$(5 \times 16^3) = 5 \times 4096 = 20480$$

$$(13 \times 16^2) = 13 \times 256 = 3328$$

$$(3 \times 16^1) = 3 \times 16 = 48$$

$$(11 \times 16^0) = 11 \times 1 = 11$$

Somando todos os valores obtidos:

$$20480 + 3328 + 48 + 11 = 23867$$

Resposta final:

$$(5D3B)_{16} = (23867)_{10}$$

Portanto, a **alternativa correta é a letra C**.

16. (FGV/Prefeitura de Salvador - 2019)

Considere a tabela verdade de um dado circuito digital, a seguir.

a	b	c	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

A saída Y desse circuito é:

a) $Y = bc + a\bar{c} + \bar{a}c + \bar{a}b$

b) $Y = b\bar{c} + a\bar{c} + \bar{a}c + \bar{a}b$

c) $Y = b\bar{c} + a\bar{c} + \bar{a}c + ac$

d) $Y = b\bar{c} + a\bar{c} + bc + \bar{a}b$

e) $Y = b\bar{c} + a\bar{c} + \bar{a}\bar{c} + \bar{a}b$

Gabarito: Letra B

Comentário:

Desenhando o **Mapa de Karnaugh** desta tabela verdade de **3 bits** e associando os termos que podem ser associados entre si. Lembrando que os **termos são agrupáveis sempre em potências de 2** (ou seja, 2^0, 2^1, 2^2, ...) e **sempre com vizinhos na horizontal ou vertical**.

a \ bc	00	01	11	10
0	0	1	1	1
1	1	0	0	1

E a expressão que ele representa fica sendo: $Y = \overbrace{\bar{a}c}^{Vermelho} + \underbrace{\bar{a}b}_{Verde} + \overbrace{b\bar{c}}^{Azul} + \underbrace{a\bar{c}}_{Roxo}$. Logo, a alternativa correta é a **letra B**.

17. FUNDEP/SAAE de Itabira-MG - 2019)

Analise o mapa de Karnaugh a seguir.

	$\overline{C}\overline{D}$	$\overline{C}D$	CD	$C\overline{D}$
$\overline{A}\overline{B}$	1	0	0	1
$\overline{A}B$	0	1	1	0
AB	0	1	1	0
$A\overline{B}$	0	0	0	0

A equação simplificada a partir desse mapa é:

a) $S=\overline{A}\cdot\overline{C}+\overline{A}\cdot\overline{D}\cdot C.$

b) $S=A\cdot\overline{B}\cdot\overline{C}+\overline{B}\cdot D.$

c) $S=A\cdot B\cdot D+B\cdot\overline{C}\cdot\overline{D}.$

d) $S=\overline{A}\cdot\overline{B}\cdot\overline{D}+B\cdot D.$

Gabarito: Letra C

Comentário:

Agrupando os termos no **Mapa de Karnaugh**, temos:

	$\overline{C}\overline{D}$	$\overline{C}D$	CD	$C\overline{D}$
$\overline{A}\overline{B}$	1	0	0	1
$\overline{A}B$	0	1	1	0
AB	0	1	1	0
$A\overline{B}$	0	0	0	0

Esta expressão pode ser resumida então em:

$$S=B\cdot D+\overline{A}\cdot\overline{B}\cdot\overline{D}$$

Portanto, **o gabarito é a letra D.**

LISTA DE QUESTÕES

1. (CESGRANRIO - 2014 - Petrobras - Técnico(a) de Projetos, Construção e Montagem Júnior - Eletrônica)

A negação da função lógica representada pela expressão $ABC + A + B + C$ equivale a

A) $\overline{ABC} + \bar{A} + \bar{B} + \bar{C}$
B) $\overline{ABC} + \overline{A + B + C}$
C) $\bar{A} + \bar{B} + \bar{C} + \overline{A + B + C}$
D) $\bar{A} \cdot \bar{B} \cdot \bar{C}$
E) $\overline{ABC}$

2. (CESGRANRIO - 2014 - Petrobras - Técnico(a) de Projetos, Construção e Montagem Júnior - Eletrônica)

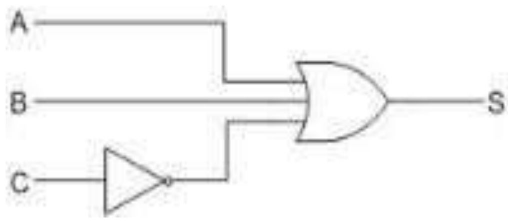
O mapa de Karnaugh mostrado acima representa a função lógica]

AB \ CD	00	01	11	10
00	0	0	1	1
01	0	0	1	1
11	1	1	0	0
10	1	1	0	0

A) $AC + \bar{B}D$

B) $AC + A\bar{C}$
C) $BD + B\overline{D}$
D) $A \oplus C$
E) $B \oplus D$

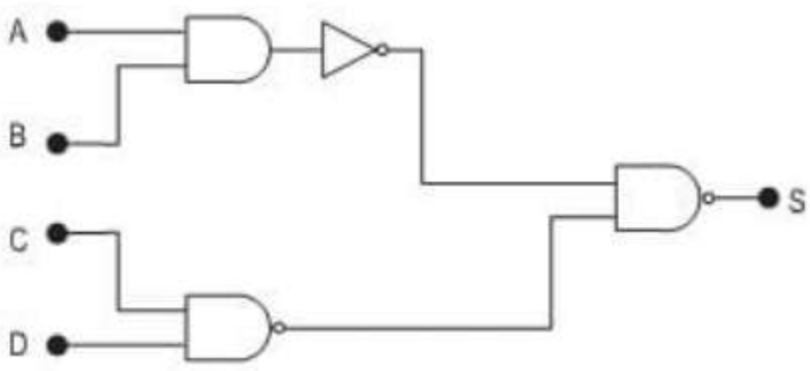
3. (CESGRANRIO - 2014 - Petrobras - Técnico(a) de Projetos, Construção e Montagem Júnior - Elétrica)



Qual a função lógica S que representa o circuito acima?

A) $A \ + \ B + C$
B) $A + B + \ \bar{C}$
C) $A + \ B + C$
D) $\bar{A} + B + C$
E) $A \cdot B \ \cdot \bar{C}$

4. (CESGRANRIO - 2014 - Petrobras - Técnico(a) de Manutenção Júnior - Eletrônica)



O circuito digital acima representa um circuito composto por três portas lógicas com quatro entradas A, B, C e D e uma saída S.

Qual é a expressão booleana da saída S?

A) $\overline{\overline{(A \cdot B)} + \overline{(B \cdot D)}}$

B) $\overline{(A \cdot B)} + \overline{(C \cdot D)}$

C) $\overline{\overline{(A \cdot D)} \cdot \overline{(C + D)}}$

D) $\overline{(A \oplus B)} + \overline{(C \cdot D)}$

E) $\overline{\overline{(A \cdot B)} \cdot \overline{(C \cdot D)}}$

5. (CESGRANRIO - 2014 - Petrobras - Técnico(a) de Manutenção Júnior - Eletrônica)

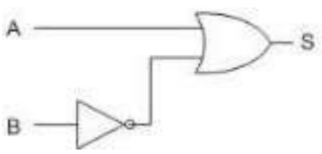
A Tabela verdade a seguir refere-se a um circuito lógico com duas entradas **A** e **B** e uma saída **S**.

A Tabela verdade a seguir refere-se a um circuito lógico com duas entradas **A** e **B** e uma saída **S**.

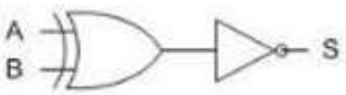
A	B	S
0	0	1
0	1	0
1	0	0
1	1	1

Qual é o circuito lógico que a Tabela representa?

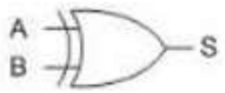
A)



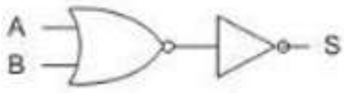
B)



C)



D)



E)

6. (CESGRANRIO - 2012 - Petrobras - Técnico de Manutenção Júnior - Eletrônica-2012)

Dada a expressão booleana

$$F(w,x,y,z) = x + x \cdot y' \cdot z + x' \cdot y' \cdot z + w \cdot x + w' \cdot x + x' \cdot y'$$

na qual “+” significa “ou lógico”, “.” significa “e lógico” e “ ’ ” significa a “negação lógica”, qual das expressões é equivalente à expressão dada?

A) $x + y + z$
B) $x + y'$
C) $x' + y$
D) $x' + y' + z'$
E) $x' + y'$

7. (CESPE / CEBRASPE - 2024 - Petrobras - Técnico Júnior - Ênfase: Manutenção - Instrumentação)

Assinale a opção que corresponde à conversão do número 00011111 de base binária para base hexadecimal.

A) 25
B) 1F
C) 4E
D) 39
E) 2F

8. (CESPE / CEBRASPE - 2024 - Petrobras - Técnico Júnior - Ênfase: Manutenção - Instrumentação)

Em relação à eletrônica digital, julgue o próximo item.

Em uma porta lógica AND de duas entradas, a saída será igual a 1 sempre que as entradas também forem iguais a 1.

A) Certo

B) Errado

9. (CESPE / CEBRASPE - 2024 - INPI - Pesquisador Em Propriedade Industrial – Área: P4)

No que se refere a aplicações e princípios da eletrônica digital, julgue o item que segue.

Conforme o primeiro teorema de De Morgan, o complemento do produto de duas variáveis é igual à soma dos complementos das variáveis individuais; em termos de portas, ele pode ser expresso por $\overline{A \cdot B} = \bar{A} + \bar{B}$

A) Certo
B) Errado

10. (CS-UFG / UFG - 2019)

Tratando-se de portas lógicas e álgebra booleana, a tabela abaixo diz respeito à operação

A	B	X
0	0	1
0	1	0
1	0	0
1	1	1

A) $X = \overline{A + B}$
B) $X = \overline{A \oplus B}$
C) $X = A + B$
D) $X = A \oplus B$

11. (CS-UFG / UFG - 2019)

A representação do número decimal 513 em octal é:

a) 201.

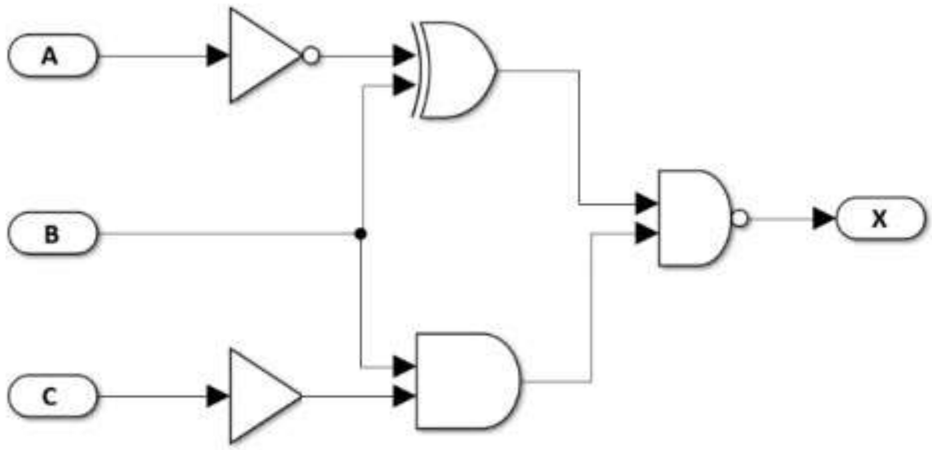
b) 0101 0001 0011.

c) 1001.

d) 0001 0000 0001.

12. (CS-UFG / UFG - 2019)

Considere o circuito combinacional a seguir.



A expressão para a saída X é:

a) $X=\overline{(\overline{A}\oplus B)\cdot(B\cdot C)}$.

b) $X=\overline{(\overline{A}+B)\cdot(B\cdot C)}$.

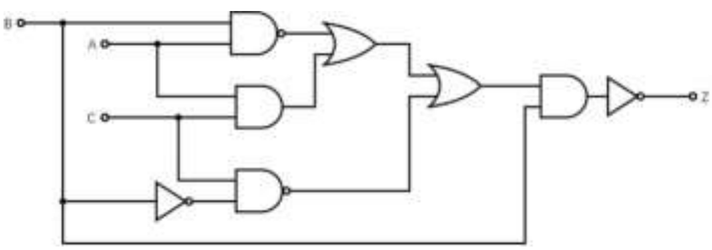
c) $X=\overline{(\overline{A}\oplus B)\cdot(B\cdot\overline{C})}$.

d) $X=\overline{(\overline{A}+B)\cdot(B\cdot\overline{C})}$.

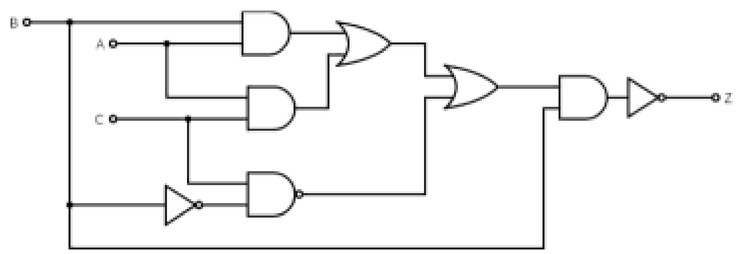
13. (FUNDEP/Prefeitura de Uberlândia - 2019)

Considerando quatro variáveis binárias A, B, C e Z, assinale a alternativa que apresenta o circuito lógico que pode ser utilizado para implementar a expressão: $Z=\overline{\left(\overline{AB}+CA+\overline{\overline{B}C}\right)B}$.

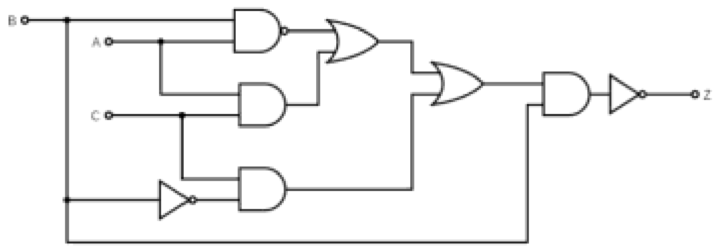
a)



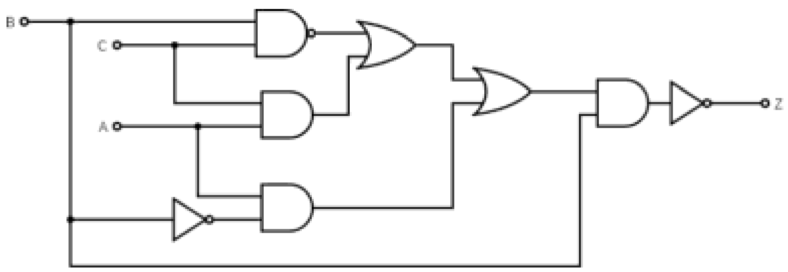
b)



c)



d)



14. (FUNDEP/Prefeitura de Uberlândia - 2019)

Em relação aos sistemas digitais, a conversão decimal binário é de fundamental importância. Assinale a alternativa que apresenta o equivalente binário ao número decimal 35.

a) 100001

b) 100111

c) 110001

d) 100011

15. (FUNDEP/Prefeitura de Uberlândia - 2019)

O sistema de base 16, ou hexadecimal, utiliza 16 símbolos para a representação numérica. Assinale a alternativa que apresenta a representação decimal para o número hexadecimal 5D3B.

a) 32

b) 23866

c) 23867

d) 512

16. (FGV/Prefeitura de Salvador - 2019)

Considere a tabela verdade de um dado circuito digital, a seguir.

a	b	c	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

A saída Y desse circuito é:

a) $Y = bc + a\bar{c} + \bar{a}c + \bar{a}b$

b) $Y = b\bar{c} + a\bar{c} + \bar{a}c + \bar{a}b$

c) $Y = b\bar{c} + a\bar{c} + \bar{a}c + ac$

d) $Y = b\bar{c} + a\bar{c} + bc + \bar{a}b$

e) $Y = b\bar{c} + a\bar{c} + \bar{a}\bar{c} + \bar{a}b$

17. FUNDEP/SAAE de Itabira-MG - 2019)

Analise o mapa de Karnaugh a seguir.

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	1	0	0	1
$\bar{A}B$	0	1	1	0
AB	0	1	1	0
$A\bar{B}$	0	0	0	0

▪ A equação simplificada a partir desse mapa é:

a) $S = \bar{A} \cdot \bar{C} + \bar{A} \cdot \bar{D} \cdot C.$

b) $S = A \cdot \bar{B} \cdot \bar{C} + \bar{B} \cdot D.$

c) $S = A \cdot B \cdot D + B \cdot \bar{C} \cdot \bar{D}.$

d) $S = \bar{A} \cdot \bar{B} \cdot \bar{D} + B \cdot D.$

Gabarito

1. D
2. D
3. B
4. E
5. B
6. B
7. B
8. Certo
9. Certo
10. B
11. C
12. A
13. A
14. D
15. C
16. B
17. D

Referências Bibliográficas

TOCCI, Ronald J.; WIDMER, Neal S.; MOSS, Gregory L. Sistemas Digitais: Princípios e Aplicações. 11. ed. São Paulo: Pearson, 2011.

FLOYD, Thomas L. Fundamentos de Sistemas Digitais. 10. ed. São Paulo: Pearson, 2013.

MANO, M. Morris; CILETTI, Michael D. Design Digital. 5. ed. São Paulo: Pearson, 2014.

ROTH, Charles H.; KINNEY, Larry L. Fundamentos de Lógica Digital com VHDL. 6. ed. São Paulo: Cengage Learning, 2013.

BARBOSA, Márcio Costa. Sistemas Digitais: Fundamentos e Aplicações. 2. ed. LTC, 2016.

LEACH, Donald P.; MALONEY, Albert Paul. Técnicas de Circuitos Digitais. 6. ed. São Paulo: Makron Books, 2001.

WAKERLY, John F. Digital Design: Principles and Practices. 4th ed. Pearson, 2006.

ESSA LEI TODO MUNDO CONHECE:
PIRATARIA É CRIME.

Mas é sempre bom revisar o porquê e como você pode ser prejudicado com essa prática.

1. Professor investe seu tempo para elaborar os cursos e o site os coloca à venda.

2. Pirata divulga ilicitamente (grupos de rateio), utilizando-se do anonimato, nomes falsos ou laranjas (geralmente o pirata se anuncia como formador de "grupos solidários" de rateio que não visam lucro).

3. Pirata cria alunos fake praticando falsidade ideológica, comprando cursos do site em nome de pessoas aleatórias (usando nome, CPF, endereço e telefone de terceiros sem autorização).

4. Pirata compra, muitas vezes, clonando cartões de crédito (por vezes o sistema anti-fraude não consegue identificar o golpe a tempo).

5. Pirata fere os Termos de Uso, adultera as aulas e retira a identificação dos arquivos PDF (justamente porque a atividade é ilegal e ele não quer que seus fakes sejam identificados).

6. Pirata revende as aulas protegidas por direitos autorais, praticando concorrência desleal e em flagrante desrespeito à Lei de Direitos Autorais (Lei 9.610/98).

7. Concurseiro(a) desinformado participa de rateio, achando que nada disso está acontecendo e esperando se tornar servidor público para exigir o cumprimento das leis.

8. O professor que elaborou o curso não ganha nada, o site não recebe nada, e a pessoa que praticou todos os ilícitos anteriores (pirata) fica com o lucro.

Deixando de lado esse mar de sujeira, aproveitamos para agradecer a todos que adquirem os cursos honestamente e permitem que o site continue existindo.