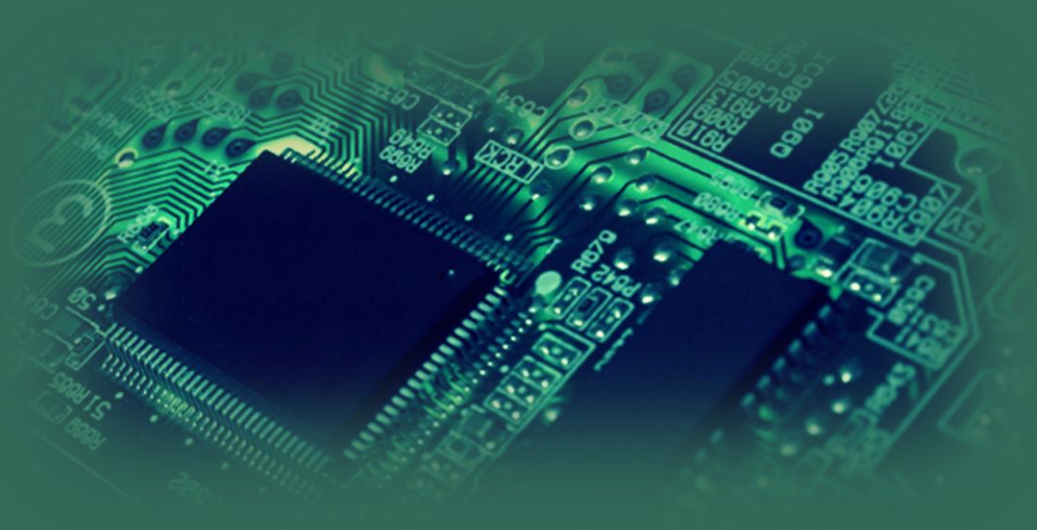
INSTITUTO FEDERAL
Santa Catarina



Eletrônica Digital para Técnicos I

Matheus Leitzke Pinto

Revisão 1

Sobre este material

Este material foi feito para auxiliar os alunos do cursos técnicos do Departamento de Eletrônica (DAELN) do IFSC – Câmpus Florianópolis – nas disciplinas introdutórias de eletrônica digital, relacionadas com o conteúdo de circuitos combinacionais.

O material é encontrado para download em: https://repositorio.ifsc.edu.br.

“Luke: "Eu não acredito!"

Yoda: "É por isso que você falha.""

SUMÁRIO

1 INTRODUÇÃO

1.1 A eletrônica digital

A eletrônica é o ramo da engenharia responsável por controlar tensão, corrente e potência elétrica, através de **dispositivos eletrônicos**. É difícil encontrar na literatura conceitos que diferenciam dispositivos eletrônicos de elétricos, podendo ambos ser tratados como a mesma coisa. Também pode-se definir como sendo dispositivos elétricos, aqueles vistos em disciplinas básicas de eletromagnetismo, como resistores, capacitores e indutores, também denominados **componentes passivos**. Em geral, os dispositivos eletrônicos recebem um conjunto de valores de entradas (tensão ou corrente) e geram um conjunto de valores de saída como apresentado na Figura 1.1.

Figura 1.1

A eletrônica também divide-se em dois grandes ramos: a **eletrônica analógica** e a **eletrônica digital**. Na eletrônica analógica, os dispositivos são projetados para trabalhar dentro de uma faixa de tensão ou corrente, e geram saídas diferentes para os infinitos valores dentro dessa faixa. Por exemplo, se o dispositivo receber o valor de tensão 0,9999 V irá gerar uma tensão na sua saída, se receber o valor de tensão 1 V irá gerar outro valor de tensão diferente na sua saída. Diz-se que os componentes de eletrônica analógica trabalham com uma **faixa contínua de valores**.

O transistor de junção bipolar é um exemplo de dispositivo analógico. Como pode ser observado na Figura 1.2, ao receber uma corrente pequena no seu pino de base (entrada), gera uma corrente proporcionalmente maior no seu pino de coletor (saída).

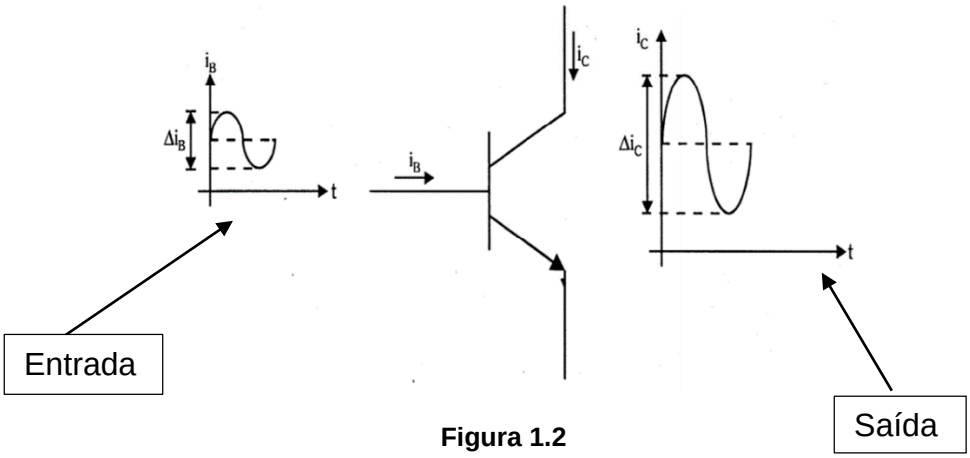


Figura 1.2

Já na eletrônica digital, os dispositivos também são projetados para trabalhar dentro de uma faixa de tensão ou corrente, entretanto interpretam uma sequência de valores como sendo a mesma coisa. Por exemplo, se o dispositivo receber valores de tensão na faixa de 0 V à 1 V irá gerar uma mesma tensão na sua saída, se receber outros valores de tensão na faixa de 2 V à 5 V irá gerar outra tensão na sua saída. Dessa forma, na eletrônica digital, os dispositivos interpretam os valores em **degraus**, por isso diz-se que trabalham com uma **faixa discreta de valores**. Os circuitos que contém dispositivos digitais são denominados de **circuitos digitais** (ou **circuitos lógicos**), e são mais fáceis de projetar que os circuitos analógicos quando o objetivo é acionamento e controle.

Os circuitos digitais geralmente interpretam as infinitas tensões dentro de uma faixa como dois valores, chamados **dígitos binários**, ou **bits**. Esses valores podem ser chamados de **ALTO** e **BAIXO**, **LIGADO** e **DESLIGADO**, **zero** (**0**) e **um** (**1**), etc.

Por exemplo, na Figura 1.3 (a), o gráfico representa o sinal elétrico de entrada em volts com o tempo em um circuito digital. Já na Figura 1.3 (b), o gráfico representa como o sistema digital interpreta esse sinal em dois valores: ALTO ou BAIXO.

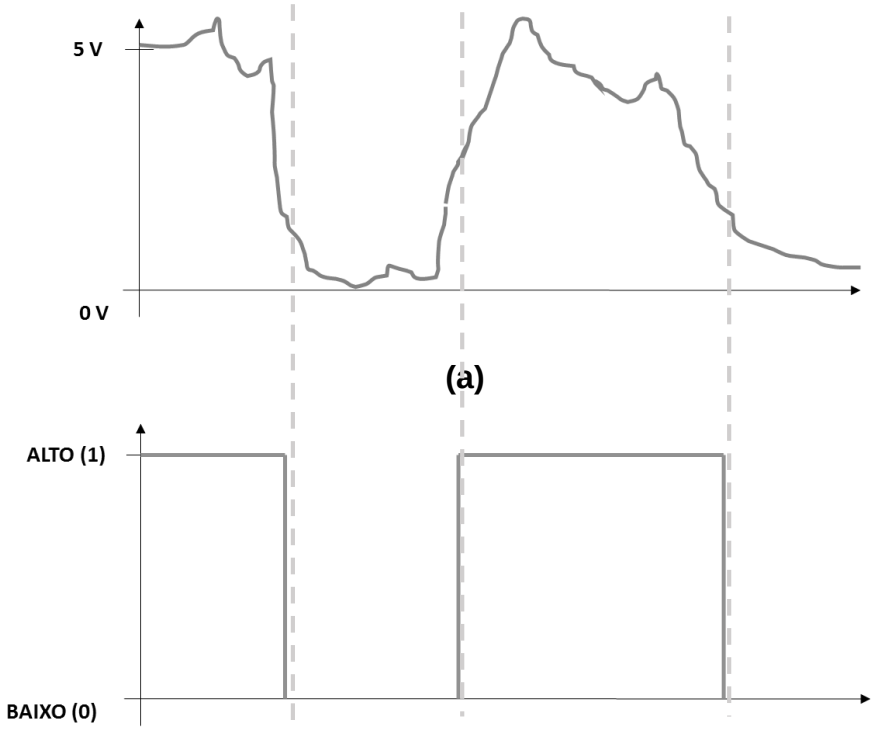


Figura 1.3

1.2 O sistema binário de numeração

Os dígitos binários fazem parte do denominado **sistema binário de numeração**. Já os números que estamos acostumados a lidar que são uma sequência de dígitos de 0 à 9 fazem parte do **sistema decimal de numeração**, pois contém dez dígitos diferentes.

Dessa forma, um circuito digital pode interpretar e gerar uma sequência de dígitos binários. Por exemplo, o gráfico apresentado na Figura 1.3 (b), denominado **forma de onda**, pode representar uma sequência de bits 10110 como apresentado na Figura 1.4. Esse valor pode representar um valor especifico que o sistema digital interpreta.

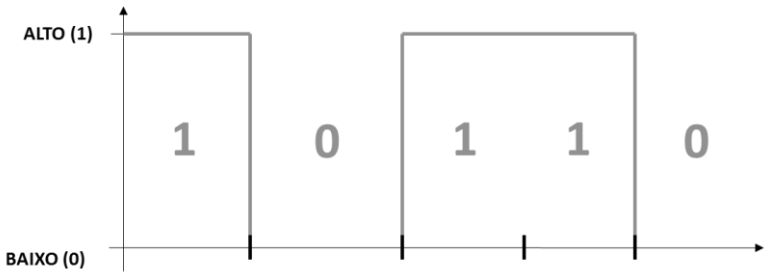


Figura 1.4

Assim como no sistema decimal, o sistema binário possui uma sequência de números. A relação entre a sequência de números do sistema decimal e binário pode ser observada na Tabela 1-1. Dessa forma, 10 em binário corresponde à 2 em decimal, por exemplo.

Tabela 1-1

DECIMAL	BINÁRIO
0	0
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111
16	10000

2 CIRCUITOS E FUNÇÕES LÓGICAS

2.1 Introdução

Os equipamentos digitais, tais como: computadores, relógios, etc., utilizam **circuitos digitais/lógicos**. Tais circuitos são constituídos de partes elementares denominadas **portas lógicas**. Esses elementos realizam uma **função lógica** ou **booleana.**

2.2 Portas lógicas básicas

Uma porta lógica é um circuito elétrico que pode receber apenas dois valores de tensão (por exemplo: +5 V e 0 V; +12 V e 0 V; +3,3 V e 0 V; etc.) que representam o valor **0 lógico** ou **1 lógico**. De acordo com a combinação dos valores de entrada, retorna uma tensão de saída que também representa um valor lógico. Um símbolo genérico de porta lógica é visto na Figura 2.1

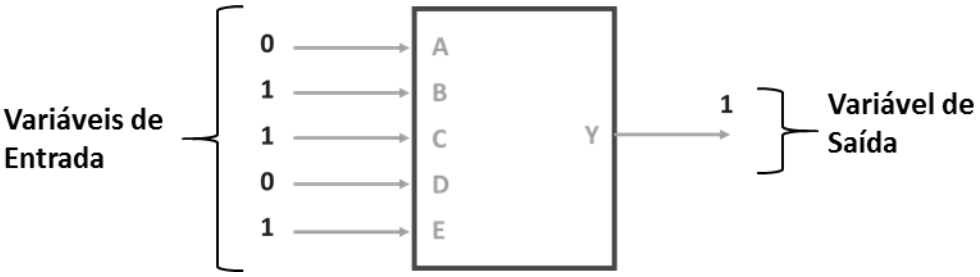


Figura 2.1

2.2.1 Porta NOT

Também denominada **porta inversora**, **negação** ou **complemento**, para uma tensão de entrada, atribui à sua saída o valor inverso da entrada, ou seja, se a variável de entrada for 0 lógico, na saída teremos nível 1 lógico, e se a variável de entrada for 1 lógico, na saída teremos nível 0 lógico.

A porta NOT pode ser representada pelo seguinte símbolo:

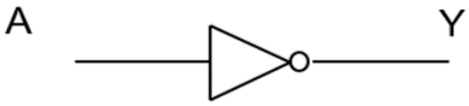
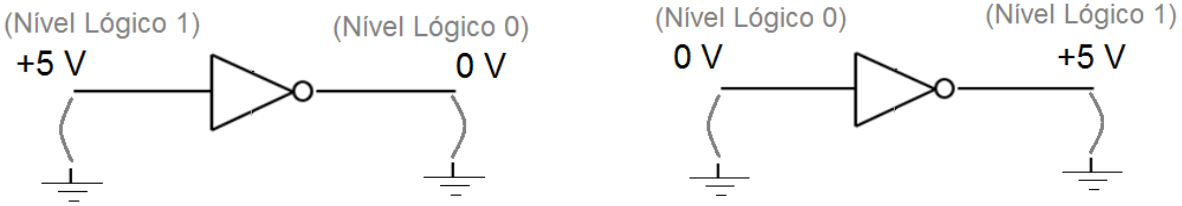


Figura 2.2

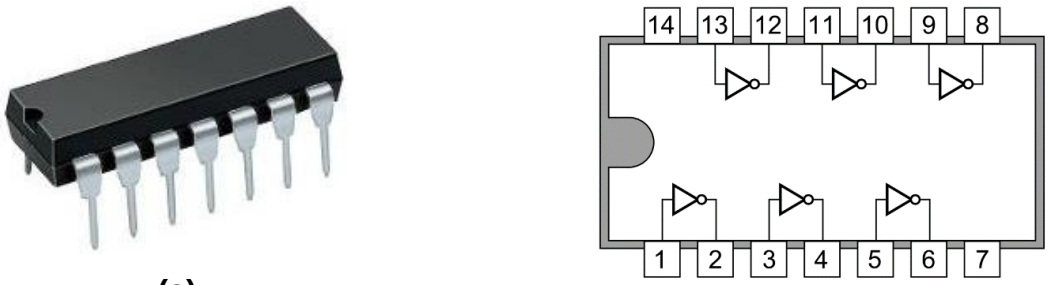
Na Figura, o terminal A é a entrada da porta e o terminal Y é saída da porta. Se para essa porta, 1 lógico é 5 V e 0 lógico é 0 V, então a Figura 2.3 apresenta as duas possibilidades de entradas e saídas.



(a) **(b)**

Figura 2.3

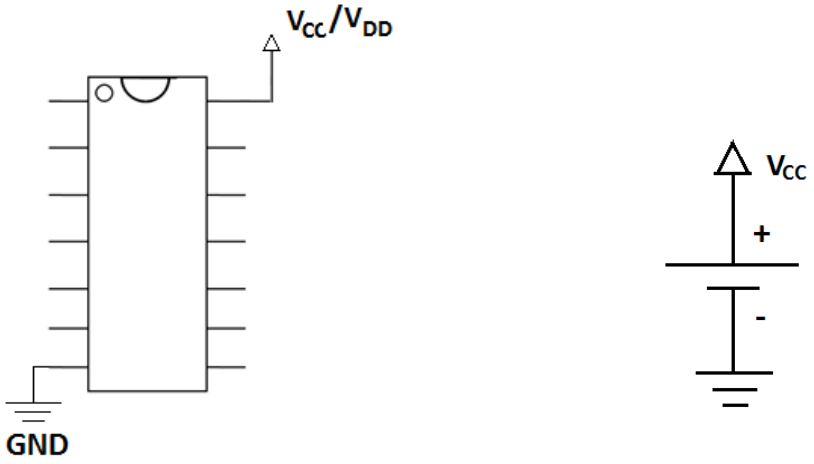
O **circuito integrado** (**CI**) digital 7404 é um dispositivo que implementa portas NOT. Na Figura 2.4 (a) é mostrado uma foto do CI e na Figura 2.4 (b) onde estão localizadas as portas nos pinos. Esse tipo de encapsulamento de CI é denominado DIP (*Dual In-line Package*).



(a) **(b)**

Figura 2.4

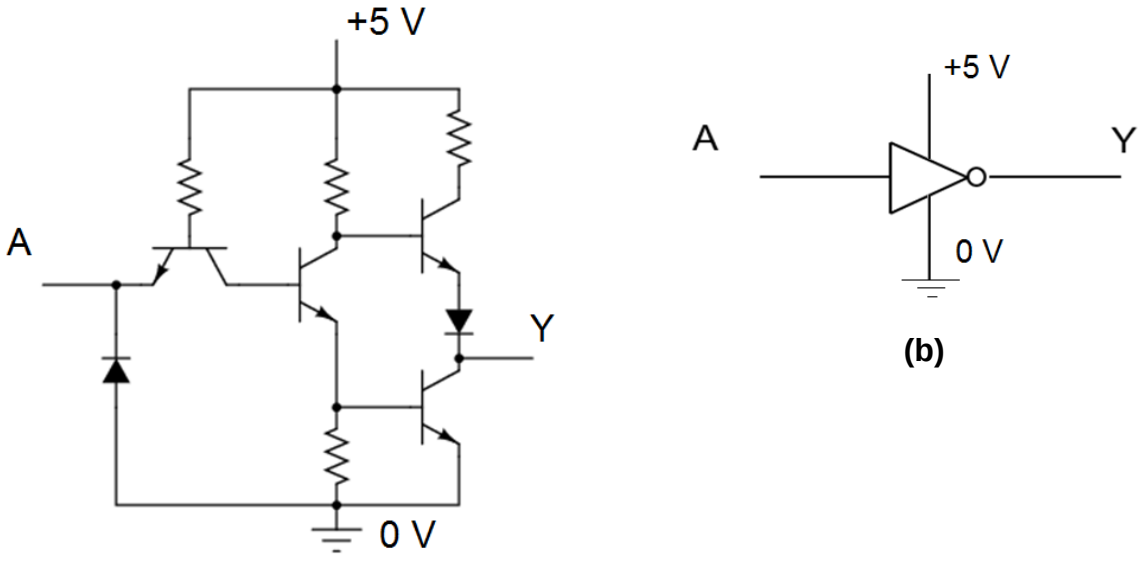
O pino 14 é é alimentado por +5 V e o pino 7 é denominado GND. O Vcc é originado do polo positivo de uma fonte de tensão, enquanto GND vêm do polo negativo da mesma fonte, como apresentado na Figura 4.4 (b).



(a) (b)

Figura 2.5

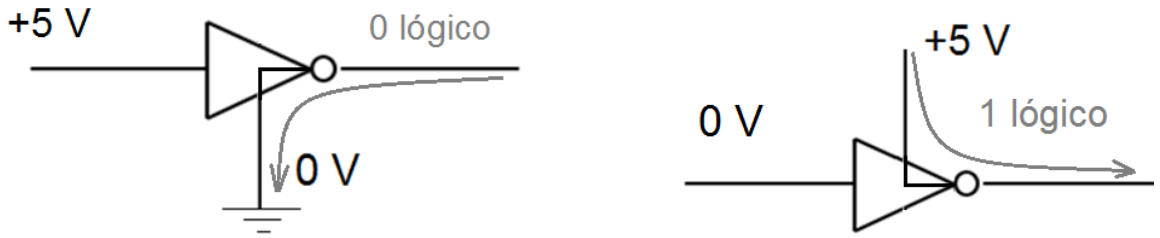
Cada porta NOT nesse CI é composto de transistores, resistores e diodos. Na Figura 2.6 (a) é exemplificado o circuito de uma porta NOT. Como pode ser visto, cada porta contém uma alimentação de +5 V e 0 V. Essa alimentação que gera a tensão saída em cada porta. A Figura 2.6 (b) é um símbolo da porta NOT mostrando as alimentações.



(a) **(b)**

Figura 2.6

Quando se coloca 0 V na entrada de uma porta, ela interpreta como 0 lógico e coloca na saída 5 V (1 lógico). Quando se coloca 5 V na entrada de uma porta, ela interpreta como 1 lógico e coloca na saída 0 V (0 lógico). O circuito transistorizado internamente abre um caminho entre sua saída e o +5 V ou o 0 V, como apresentado na Figura 4.5 (a).



(a) (b)

Figura 2.7

Quando a saída da porta é 1 lógico (+5 V), uma corrente sai da porta lógica. Quando a saída da porta é 0 lógico (0 V), uma corrente é absorvida pela porta lógica.

Ainda, na Figura 2.8 é apresentado uma foto microscópica do CI 74HC00 contendo quatro portas NAND. Note que os círculos pretos são os ***pads*** do CI, cujas funções são a conexão com os pinos do involucro através de fios bem finos.

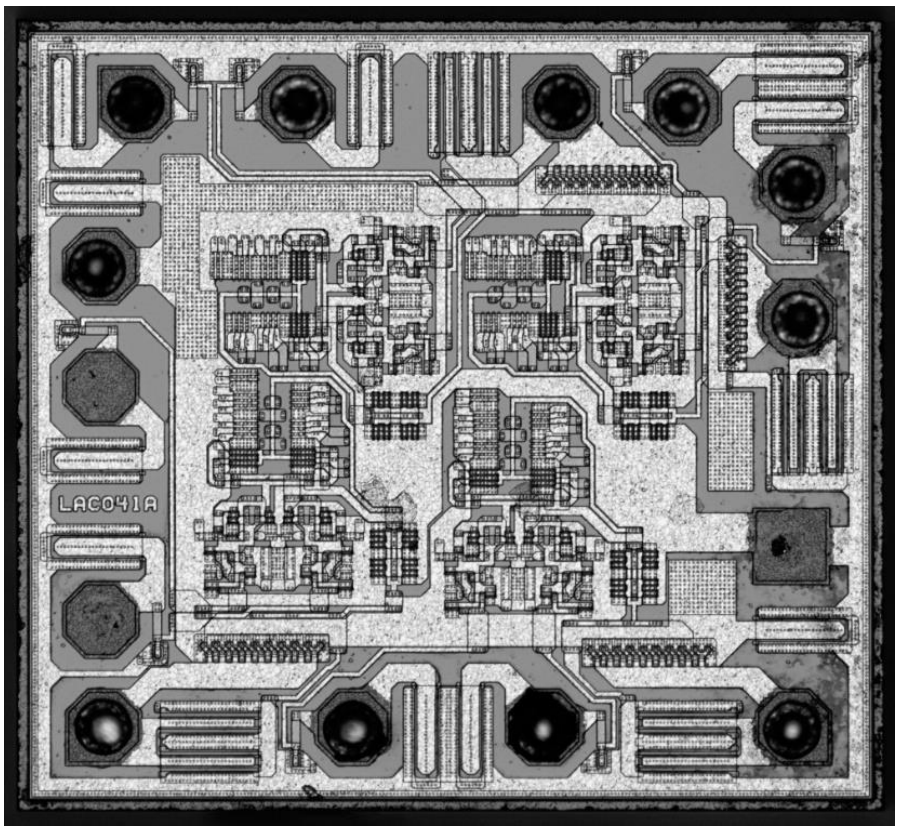


Figura 2.8

Nas Figura 2.9 e Figura 2.10 são apresentadas uma aplicação de portas NOT no controle de um LED. Uma chave é utilizada para selecionar se a entrada da porta é 0 V ou

+5 V. Se a entrada for 0 V, o LED acende, pois é polarizado diretamente na saída da porta. Caso a entrada for +5 V, a saída será 0 lógico, fazendo com que a tensão no LED seja nula e o mesmo fica apagado.

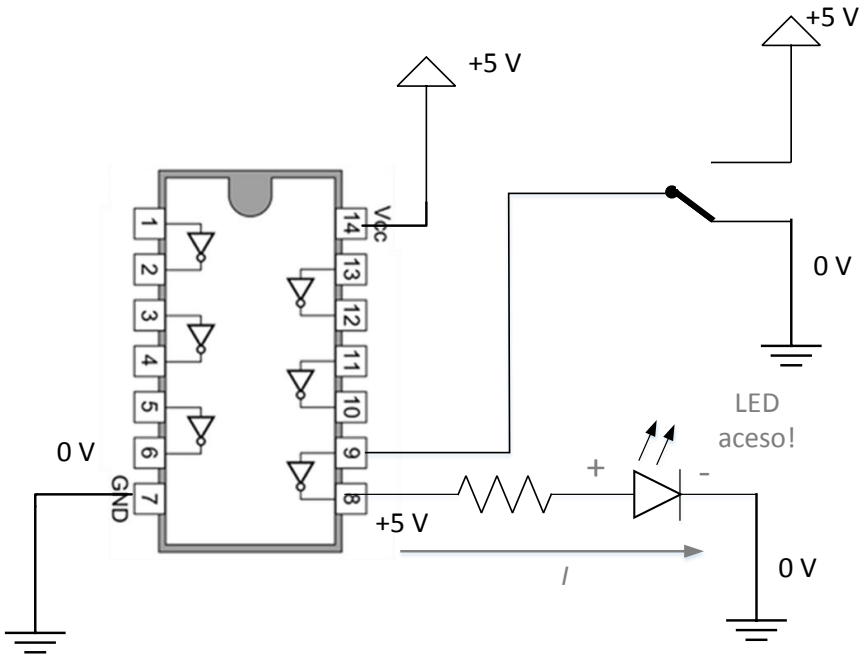


Figura 2.9

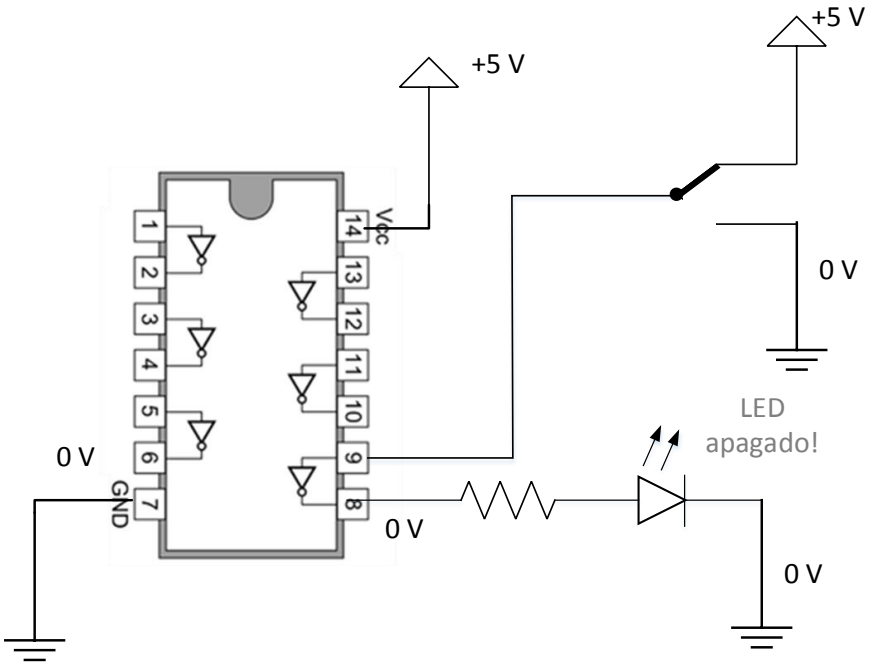


Figura 2.10

Note que a chave poderia ser substituída pela saída de um sensor. Com criatividade pode-se criar diversos arranjos com circuitos digitais. No circuito da Figura 2.11, a chave é substituída por um sensor (de presença, de nível, etc.) que em condições normais emite +5 V (1 lógico). Caso o sensor detecte algo, responde na sua saída com 0 V (0 lógico). Nesse caso, o LED acende, indicado que que algo foi detectado. Note que o comum (0 V) do sensor é o mesmo do CI.

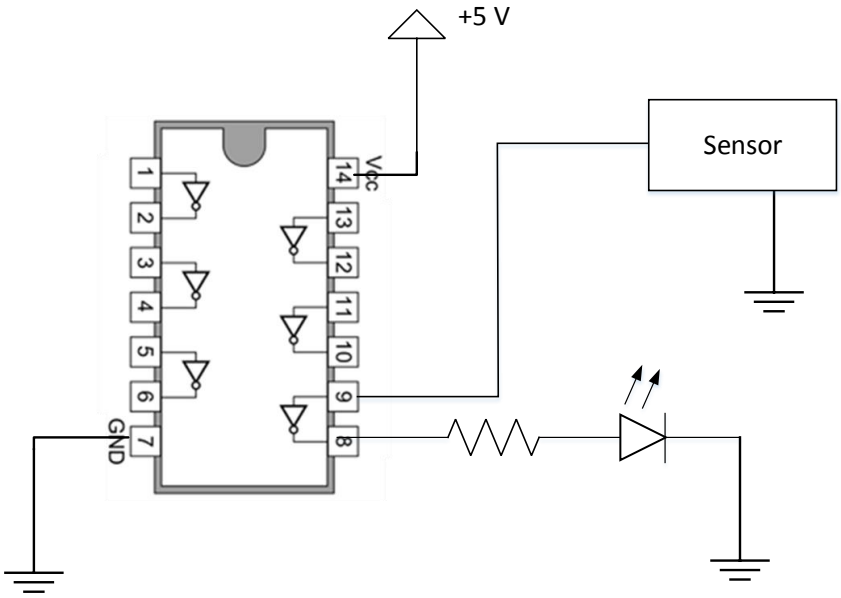


Figura 2.11

A porta NOT é a representação física da **função lógica NOT,** também denominada **negação** ou **complemento**. Essa função é expressa por:

$Y = \overline{A} \rightarrow$ onde se lê ***Y é igual à A negado.***

Se A = 0, então:

- $Y = \overline{0} = 1$

Se A = 1, então:

- $Y = \overline{1} = 0$

A **tabela-verdade** é um mapa onde colocamos as possíveis combinações das variáveis de entrada com os respectivos resultados. A tabela-verdade da função NOT é:

A	Y
0	1
1	0

2.2.2 Porta AND

Na porta AND, existem duas entradas, se uma das entradas for 0 lógico, a saída também será 0 lógico, independentemente do nível lógico presente na outra entrada.

A porta AND pode ser representada pelo seguinte símbolo:

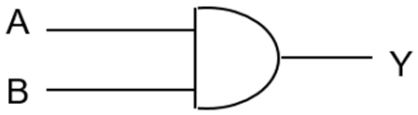


Figura 2.12

Na Figura, os terminais A e B são as entradas da porta e o terminal Y é saída da porta. Se para essa porta, 1 lógico é 5 V e 0 lógico é 0 V, então a Figura 2.13 apresenta as quatro possibilidades de entradas e saídas.

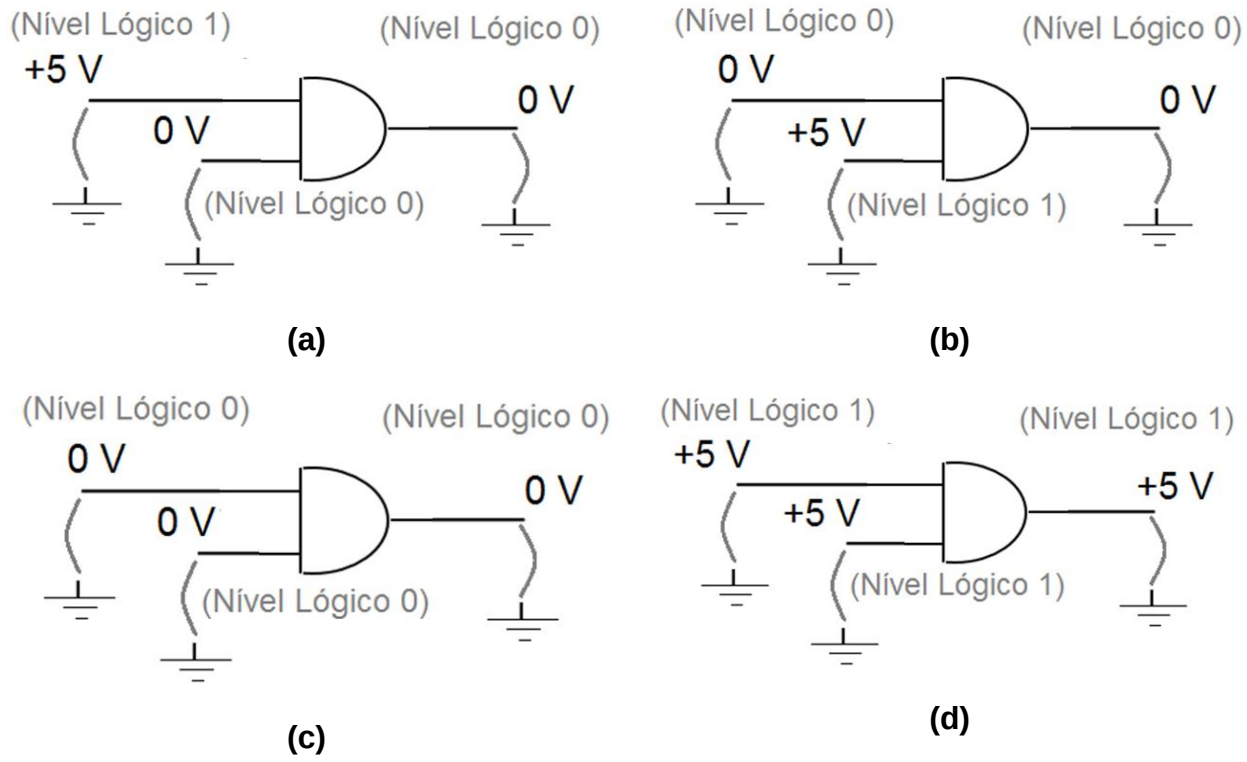


Figura 2.13

O CI digital 7408 é um dispositivo que implementa portas AND. Na Figura 2.14 é apresentado um alarme que dispara quando um dos sensores conectados nas entradas da porta está em nível lógico 0.

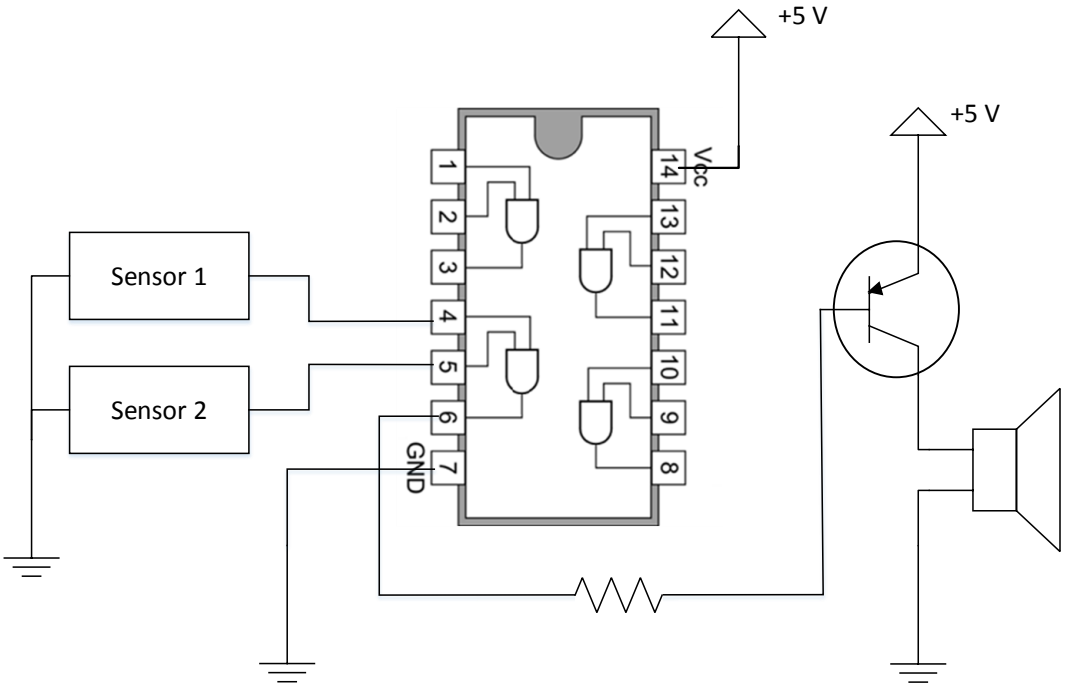


Figura 2.14

Como os CIs digitais oferecem pouca corrente em suas saídas, a função do transistor na Figura é amplificar a corrente para poder ligar o alto-falante.

Também denominada **multiplicação lógica**, na função AND, se uma das entradas for zero, a saída também será zero, independentemente do nível lógico presente nas outras entradas.

A expressão lógica que representa a função AND é:

$Y = A \cdot B \rightarrow$ onde se lê ***Y é igual a A AND B.***

A tabela-verdade da função AND é:

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

1	1	1

2.2.3 Porta OR

A porta OR pode ser representada pelo seguinte símbolo:

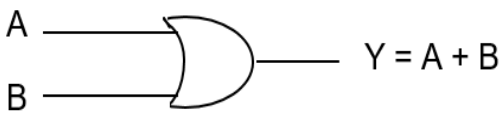


Figura 2.15

Também denominada **soma lógica**, na função OR, se uma das entradas for um, a saída também será um, independentemente do nível lógico presente nas outras entradas.

A expressão lógica que representa a função OR é:

$Y = A + B \rightarrow$ onde se lê ***Y é igual a A OR B.***

A tabela-verdade da função OR é:

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

2.2.4 Portas NAND e NOR

É possível negar a saída de uma porta AND ou OR através de um inversor na saída como apresentado na Figura 2.16.

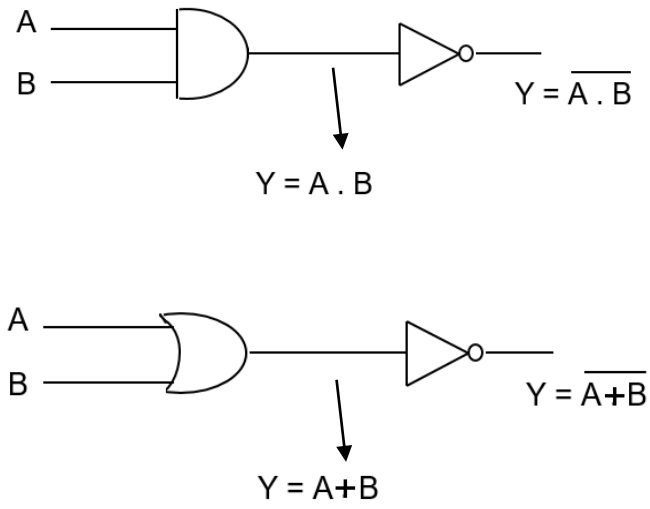


Figura 2.16

Esses dois circuitos lógicos simples podem ser considerados como uma porta lógica cada. Uma é a **porta NAND** (**NOT-AND**) e a outra é a **porta NOR** (**NOT-OR**). Os símbolos são apresentados na Figura 2.17.

NAND **NOR**

Figura 2.17

A tabela-verdade de cada uma são apresentadas na Figura 2.18.

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

NAND

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

NOR

Figura 2.18

Na Figura 2.19 são apresentados os CIs 7400 e 7402 que implementam portas NAND e portas NOR. respectivamente.

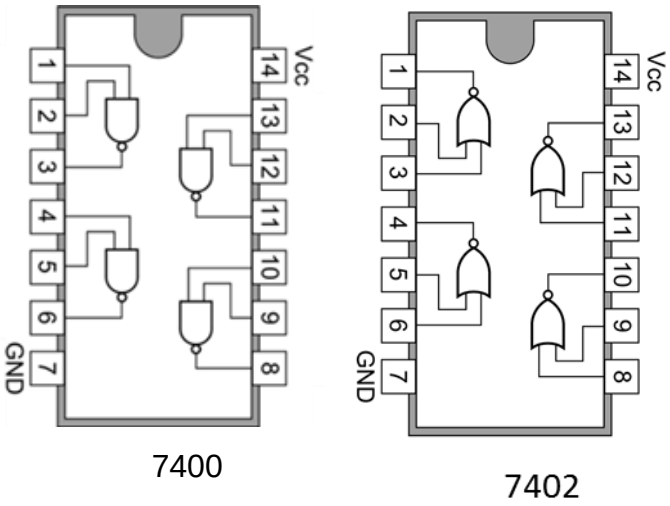


Figura 2.19

2.2.5 *Buffer* digital

O ***buffer* digital** é a porta que a princípio parece ter a função mais inútil dentre as portas, pois a sua lógica é a mesma de um fio:

$Y = A \rightarrow$ onde se lê ***Y é igual à A.***

O *buffer* digital pode ser representado pelo seguinte símbolo:

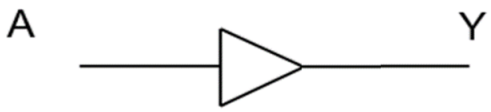


Figura 2.20

Porém, na prática um circuito de buffer tem as funções de **reforçador de corrente elétrica** e/ou **conversor de nível lógico**.

Como exemplo, observe o circuito apresentado na Figura 2.21. Trata-se do uso do acionamento de um motor elétrico que opera à 12 V por um sistema digital que trabalha com 3,3 V. Dessa forma, não é possível ligar diretamente a saída do sistema digital ao motor, e por essa maneira coloca-se um CI 7407 que converte o nível lógico 1 de 3,3 V do sistema digital para 12 V do motor.

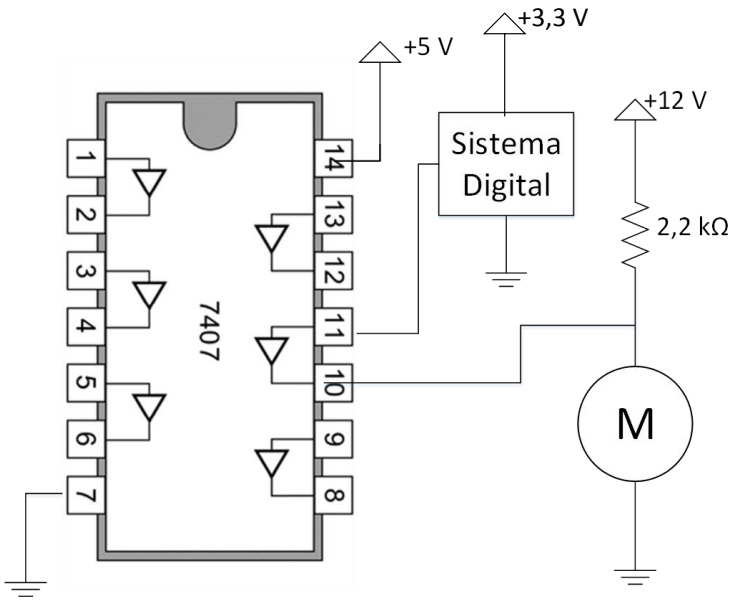


Figura 2.21

O CI deve ser alimentado com 5 V e reconhece uma tensão de 2 V à 5,5 V como nível lógico 1 e 0 V à 0,8 V como nível lógico 0. Note uma característica interessante do 7407: foi ligado em sua saída um resistor de 2k2, denominado **resistor de *pull-up***, que está conectada aos 12 V. É dito que sua saída está em **coletor aberto**. Nesse tipo de circuito, o pino de saída do chip é o coletor de um transistor como apresentado na Figura 2.22.

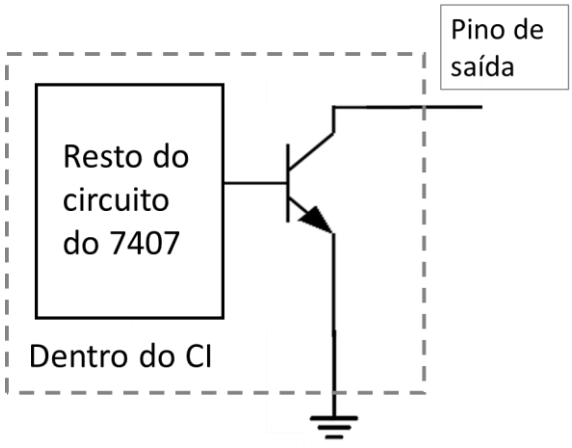


Figura 2.22

Quando a saída for nível lógico 0, o transistor irá operar como uma chave fechada e a saída do pino irá ligar ao emissor que está ligado ao nível lógico 0, como apresentado na Figura 2.24. Nesse caso, a tensão de 12 V é “ignorada” pelo motor, pois o caminho dos 12 V ao emissor é mais “fácil” do que dos 12 V ao motor. Isso faz com que não haja rotação do motor. Quando a saída for nível lógico 1, o transistor irá operar como uma chave aberta,

e o pino ficará “desconectado” do circuito de saída. Isso permite que no circuito de saída seja fornecida uma tensão e corrente maiores, como apresentado na Figura 2.24. O limite da tensão no pino de saída do buffer 7407 é 30 V.

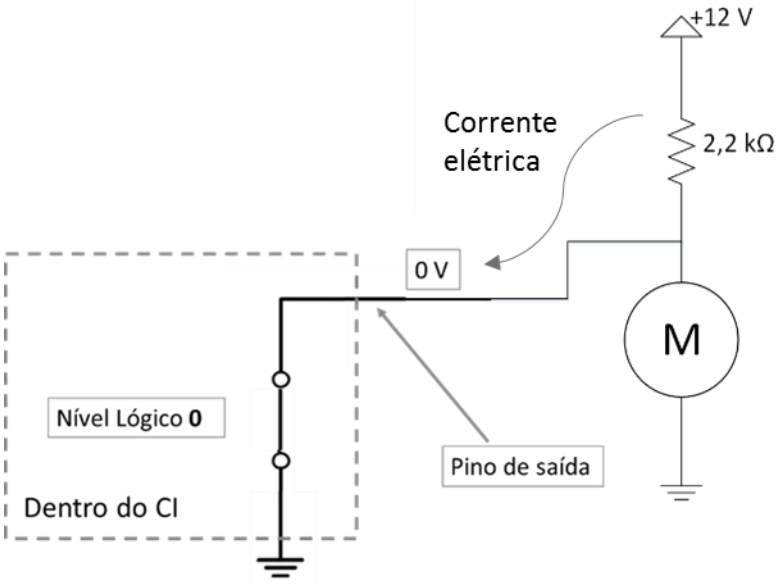


Figura 2.23

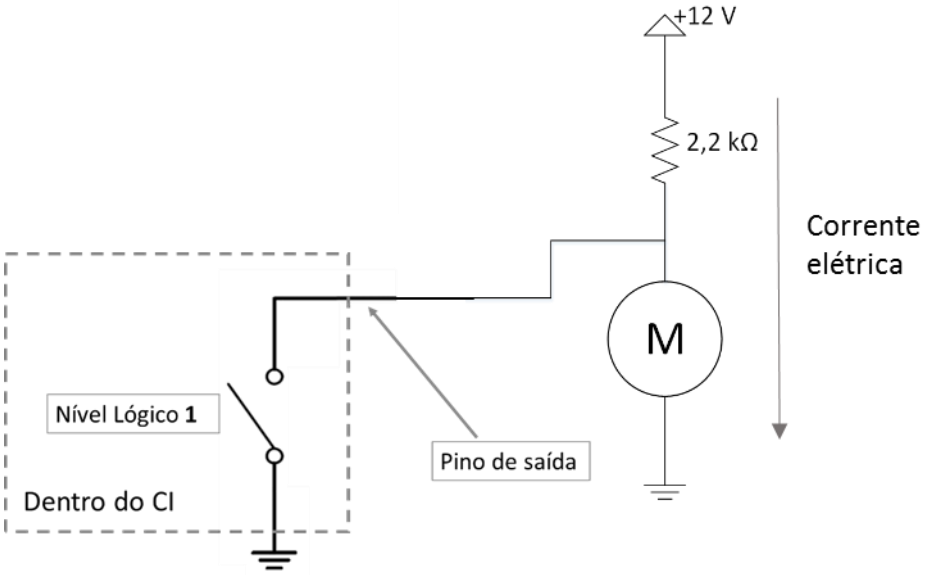


Figura 2.24

2.2.6 Dupla negação

É importante salientar algo nesse ponto: quando utiliza-se negação duas vezes, tem o resultado original. Por exemplo, uma AND com a saída negada torna-se uma NAND, mas se negarmos novamente a saída, como apresentado na Figura 2.25, obtém-se novamente a função AND.

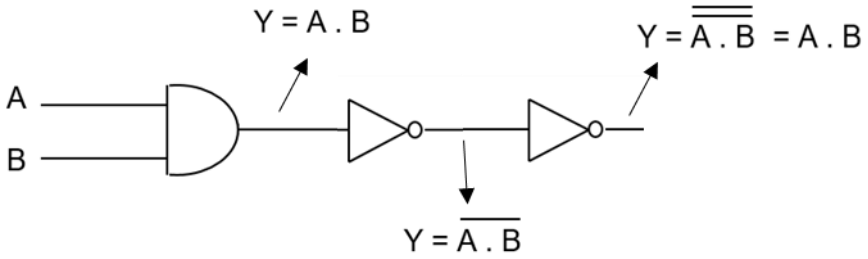


Figura 2.25

3 Avaliação e Síntese de Circuitos Lógicos

3.1 Avaliação de expressões booleanas

Uma expressão tem geralmente mais de uma função lógica. Quando queremos avaliar uma expressão, devemos montar sua tabela verdade. Considere a seguinte expressão:

$$Y = A + B \cdot \overline{C}$$

Dividimos a expressão em **subexpressões** elementares. Para isso, cada operador tem uma **precedência**.

- 1º Parênteses;
- 2º Negação;
- 3º Multiplicação lógica;
- 4º Soma lógica.

No exemplo, não tem-se parênteses. Então, começamos pela negação:

$$\overline{C}$$

Dentro da negação, não há mais expressão. Paramos aqui e temos nossa primeira subexpressão. Vamos chamar essa expressão de **s1**.

Agora, avaliamos a multiplicação lógica:

$$B \cdot s1$$

Vamos chamar essa expressão de **s2**

Agora, avaliamos a soma lógica:

$$A + s2$$

Chegamos na expressão final.

Montamos a tabela verdade da expressão criando colunas para cada subexpressão, como mostra Figura 3.1.

A	B	C	s1	s2	Y
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

Figura 3.1

Temos que s1 é C negado. Dessa forma, colocamos na coluna de s1, todos os valores de C negado correspondente à linha. Resolvemos gradativamente cada subexpressão até chegar em Y e obtermos a tabela apresentada na Figura 3.2.

A	B	C	s1	s2	Y
0	0	0	**1**	**0**	**0**
0	0	1	**0**	**0**	**0**
0	1	0	**1**	**1**	**1**
0	1	1	**0**	**0**	**0**
1	0	0	**1**	**0**	**1**
1	0	1	**0**	**0**	**1**
1	1	0	**1**	**1**	**1**
1	1	1	**0**	**0**	**1**

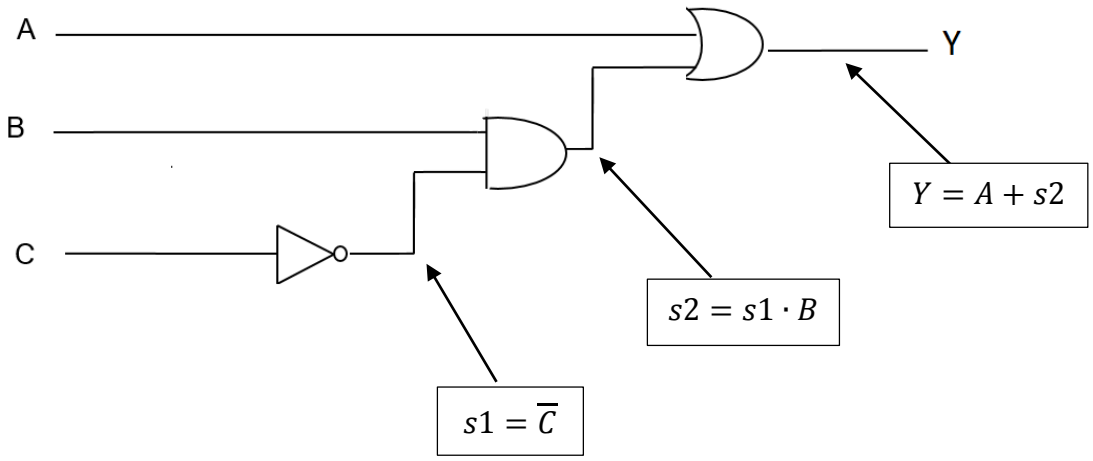
Figura 3.2

Note que $\overline{A \cdot B}$ é diferente de $\overline{A} \cdot \overline{B}$. No primeiro caso, $\overline{A \cdot B}$, a negação está sendo aplicada ao resultado de $A \cdot B$. No segundo caso, $\overline{A} \cdot \overline{B}$, primeiro aplica-se a negação nas variáveis individualmente e depois aplica-se a função AND.

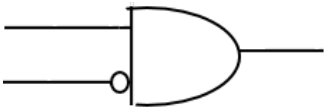
3.2 Síntese de circuitos lógicos a partir de expressões

Os passos para desenhar o circuito lógico a partir de uma equação são praticamente as mesmas para a avaliação da expressão.

Exemplo: da expressão do exemplo da seção 3.3: $Y = A + B \cdot \overline{C}$:



Podemos substituir o desenho de um inversor na entrada por um círculo apenas:



Com a mesma lógica, pode-se seguir o caminho inverso: obter a expressão lógica a partir do circuito.

3.3 Funções e portas XOR e XNOR

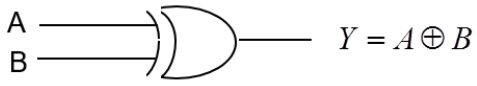
Outra classe de funções bastante utilizada é **XOR** (**Exclusive-OR**) e **XNOR** (**Exclusive-NOR**).

Na função XOR, se o número de entradas com nível lógico um for ímpar, a saída terá nível lógico um; caso contrário terá nível lógico zero.

A expressão lógica que representa a função XOR é:

$Y = A \oplus B \rightarrow$ onde se lê ***Y é igual a A XOR B.***

Sua porta lógica e tabela verdade podem ser vistos na Figura 3.3.



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

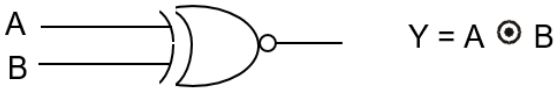
Figura 3.3

O operador XOR tem a mesma precedência do operador OR.

A função XNOR é a negação da XOR:

$Y = A \odot B \rightarrow$ onde se lê ***Y é igual a A XNOR B.***

Sua porta lógica e tabela verdade podem ser vistos na Figura 3.4



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

Figura 3.4

3.4 Portas com mais de duas entradas

Uma porta lógica pode ter mais de duas entradas (três, quatro, cinco, etc), como apresentadas na Figura 3.5. Na mesma figura pode-se notar que existe uma equivalência entre portas com mais entradas e portas de duas entradas.

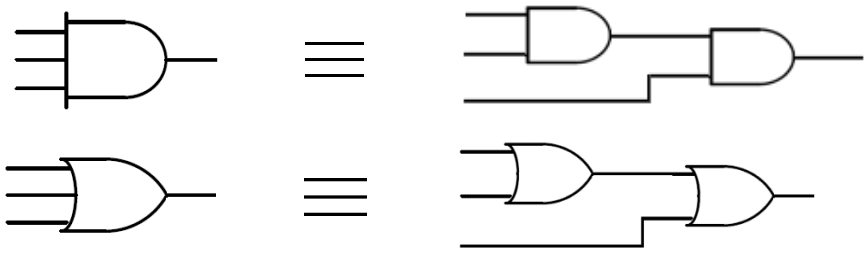


Figura 3.5

Do ponto de visto lógico, uma porta de três entradas é aplicação de uma operação lógica entre três entradas. Para a porta AND, a função seria:

$$Y = A \cdot B \cdot C$$

Já para uma porta NOR:

$$Y = A + B + C$$

O mesmo é válido para portas XOR. Deve-se ter cuidado apenas com portas NAND, NOR e XNOR, que são negações de operações básicas. Dessa forma, para uma NAND, por exemplo, de três entradas, primeiramente aplica-se a função AND, para enfim aplicar a função NOT, como apresentado na Figura 3.6.

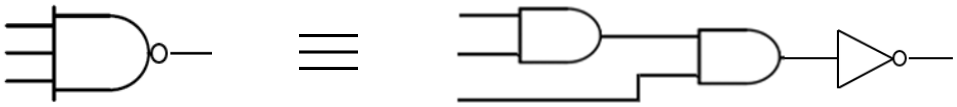


Figura 3.6

O que corresponde à seguinte função lógica:

$$Y = \overline{A \cdot B \cdot C}$$

3.5 Síntese de circuitos lógicos a partir de tabela verdade

No dia a dia de nossa vida profissional nos defrontamos com determinadas situações em que necessitamos construir circuitos lógicos que atuem de acordo com estas situações. Para isto, devemos proceder da seguinte maneira:

a) Construir a tabela-verdade que representa esta situação.
b) Retirar da tabela-verdade a expressão de saída.
c) Montar o circuito lógico que executa a expressão de saída obtida.

Exemplo:

Tem-se três motores A, B e C que consomem cada um 50 KVAR (medida de potência reativa em circuitos de corrente alternada) de potência reativa e 2 bancos de capacitores, um de 50 KVAR e um de 100 KVAR com o objetivo de compensar a potência reativa gerada pelos motores, conforme Figura 3.7. Construir um circuito lógico que comande a ligação destes capacitores à rede elétrica de modo que a potência reativa total consumida pela instalação, a qualquer instante, seja nula.

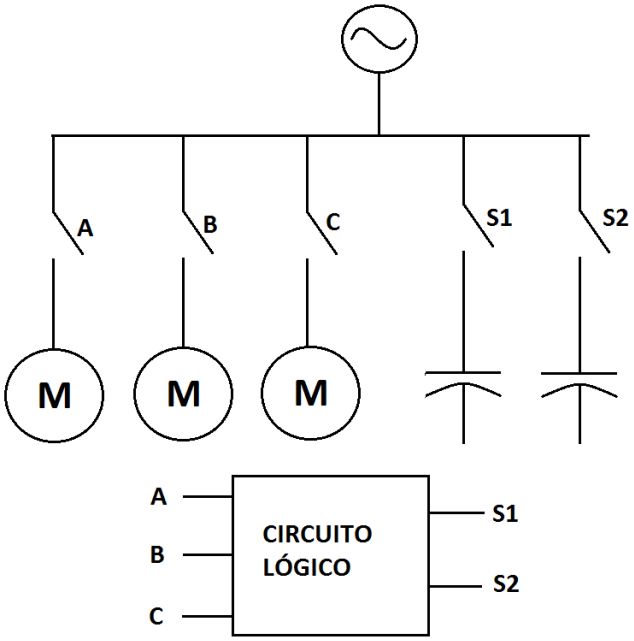


Figura 3.7

a) Construir a tabela-verdade:

C	B	A	S1	S2
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

b) Retirar da tabela-verdade a expressão de saída:

Para isto, utilizamos o método da **soma de produtos**. Para cada “1” em cada saída da tabela-verdade escrevemos um termo na forma C.B.A na expressão de saída, sendo cada um desses termos separado pela função OR. Após, negamos as variáveis de entrada que são “0”.

S1 =

S2 =

c) Montar o circuito lógico

O circuito pode ainda ser **simplificado**, de forma a reduzir o número de portas lógicas utilizadas. Os métodos geralmente empregados são **simplificação algébrica de expressões lógicas** ou **mapas de *Karnaugh***. Existem ferramentas de software encontradas na internet que implementam esses métodos para simplificação de um circuito com base na expressão lógica fornecida, como em [x] e [x]. Dessa forma, não retirando a importância do aprendizado desses métodos, nesse material não serão apresentados os métodos de simplificação.

4 CIRCUITOS INTEGRADOS DIGITAIS

O CI ou chip é encapsulado em uma embalagem de plástico ou de cerâmica, a partir da qual saem alguns pinos para tornar possível a conexão do CI com outros dispositivos como apresentado na Figura 4.1.

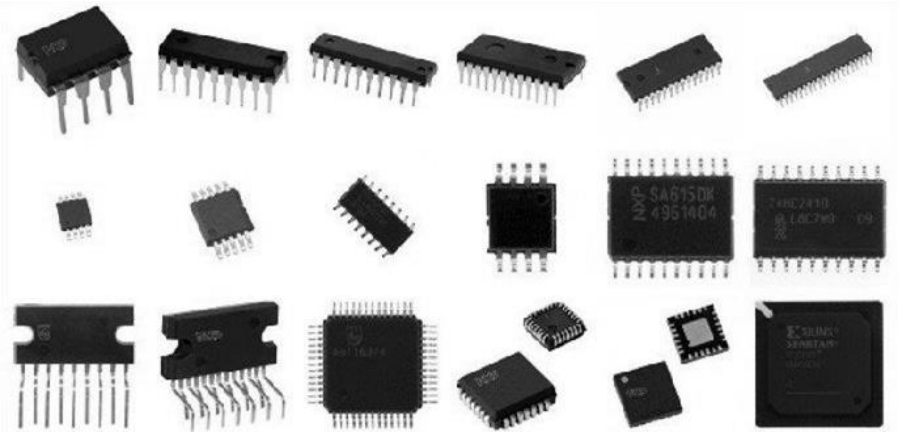


Figura 4.1

Dentro desses encapsulamentos é que se encontram os CIs de fato. A Figura 4.2 (a) apresenta um CI devidamente lacrado, e a Figura 4.2 (b), o mesmo CI exposto.

(a) (b)

Figura 4.2

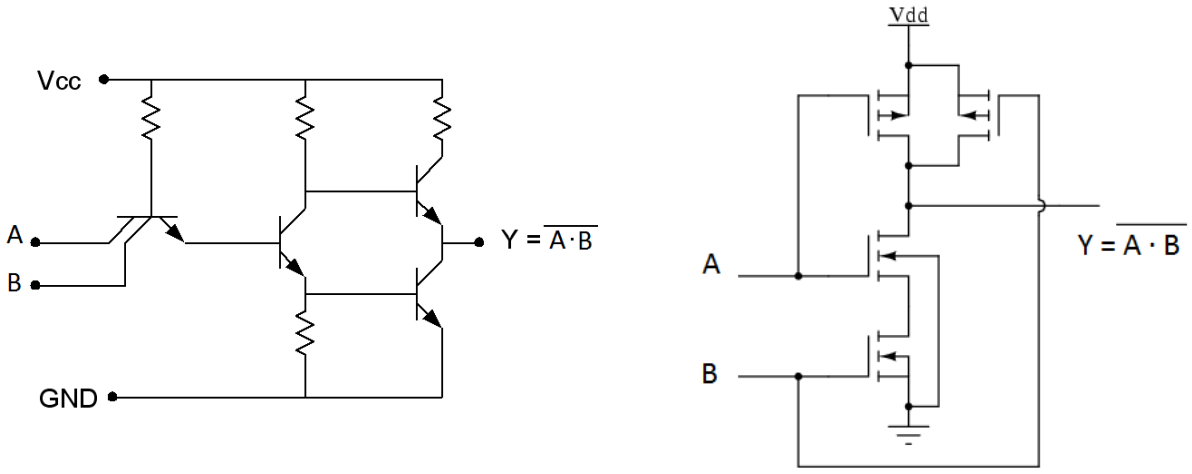
Os CIs digitais são muitas vezes classificados de acordo a complexidade de seus circuitos medida pela quantidade de portas lógicas existentes no seu substrato. Existem, atualmente, cinco graus de complexidade, definidos conforme abaixo:

Tabela 4-1

Complexidade	Número de portas
Integração em pequena escala (SSI)	Menos de 12 portas
Integração em escala média (MSI)	De 12 a 99
Integração em grande escala (LSI)	De 100 a 9999
Integração em escala muito grande (VLSI)	De 10000 a 99999
Integração em escala ultragrande (ULSI)	100000 ou mais

Existem basicamente duas **famílias** (tecnologias) de CIs que lideram o mercado: **TTL** (***Transistor-transistor logic***) e **CMOS** (***complementary metaloxide semiconductor***). A família TTL são CIs compostos basicamente de transistores de junção bipolar (NPN ou PNP) e resistores. Já os CIs CMOS têm como principal elemento de circuito os transistores MOSFET (canal N e canal P).

A Figura 4.3 (a) apresenta o esquemático de uma porta NAND TTL, enquanto na Figura 4.3 (b) tem-se a mesma porta em CMOS.

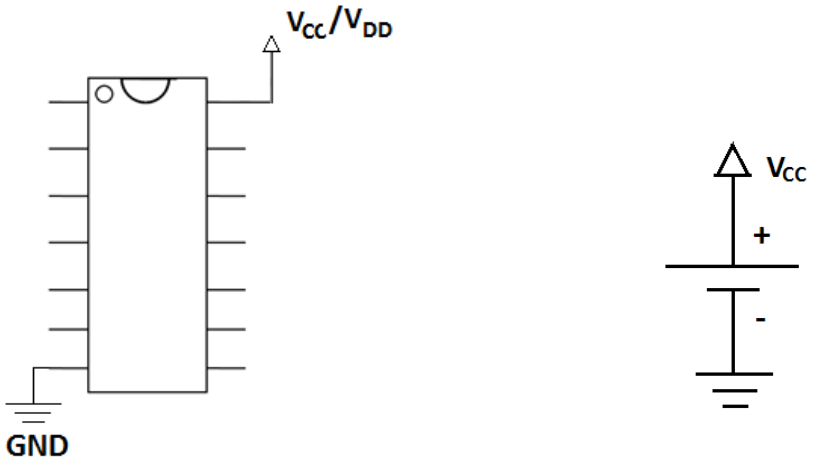


(a) **(b)**

Figura 4.3

Os terminais **V_{CC}/V_{DD}** e **GND** apresentados vêm da alimentação do chip. Todo CI tem ao menos dois pinos de alimentação: uma positiva e outra de 0 V (denominada GND), como apresentado na Figura 4.4 (a).

Quando trabalhamos com CIs da família TTL, chamamos essa alimentação positiva de V_{CC}, enquanto que para CIs CMOS essa alimentação é denominada V_{DD}. Tanto V_{CC}, como V_{DD} são originadas do polo positivo de uma fonte de tensão, enquanto GND vêm do polo negativo da mesma fonte, como apresentado na Figura 4.4 (b).



(a) (b)

Figura 4.4

4.1 Representação de valores lógicos em CIs

O nível lógico 1 é representado por uma tensão próxima à V_{CC}/V_{DD}, enquanto o nível lógico 0 é representado por uma tensão próxima à 0 V.

Quando uma porta lógica em um CI quer indicar em sua saída o nível lógico 1, a mesma internamente abre um caminho entre sua saída e o V_{CC}/V_{DD}, como apresentado na Figura 4.5 (a).

(a) (b)

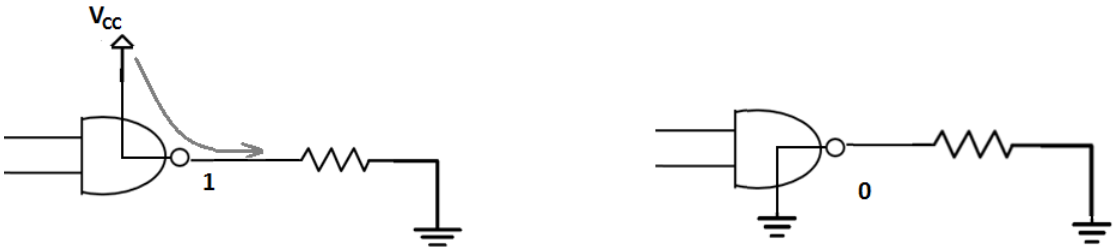
Figura 4.5

Dessa forma, a saída da porta lógica quando quer representar o nível lógico 1 é V_{CC}/V_{DD}. Há perdas internamente entre V_{CC} e a saída da porta lógica, de forma que a tensão

na saída pode não ser exatamente V_{CC}. Note que uma corrente sai da porta lógica nesse caso.

De outro modo, se uma porta lógica em um CI quer indicar em sua saída o nível lógico 0, a mesma internamente abre um caminho entre sua saída e o GND (0 V), como apresentado na Figura 4.5 (b). Note que uma corrente entra na porta lógica nesse caso.

A saída de uma porta lógica pode estar conectada à entrada de outra porta lógica ou à uma carga qualquer, como apresentado na Figura 4.6.

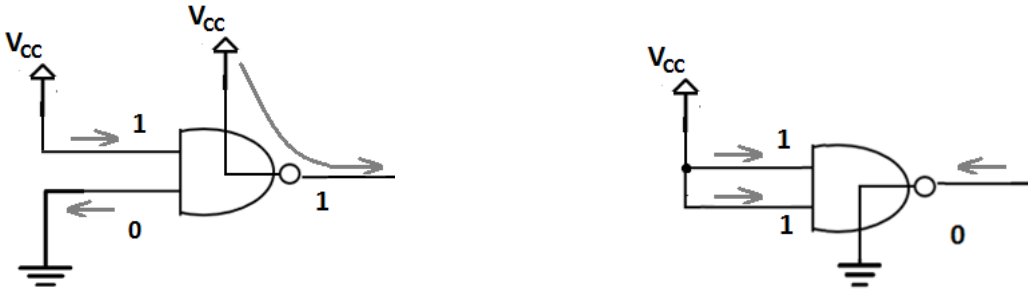


(a) (a)

Figura 4.6

No exemplo apresentado, aplicar o nível 0 na saída faz com que não haja tensão no resistor, já que ambos os lados são 0 V. Isso é apenas um exemplo de aplicação, dentre vários que podemos criar.

Uma porta lógica ao receber uma tensão V_{CC}/V_{DD} na sua entrada, irá interpretar o nível lógico 1. Se receber uma tensão de 0 V irá interpretar como sendo o nível lógico 0. Analise os exemplos apresentados na Figura 4.7.



(a) (b)

Figura 4.7

A tensão pode advir de diversos meios, principalmente da saída de outra porta lógica.

4.2 Parâmetros de tensão em circuitos integrados

Como mencionado, nem sempre o nível lógico 0 é exatamente 0 V, e nem sempre o nível lógico 1 é exatamente V_{CC}/V_{DD}. Dessa forma, um CI tem os seguintes parâmetros de tensão.

- **V_{OH}** - É o valor mínimo de tensão de saída correspondente ao nível lógico alto que um CI vai gerar, como apresentado na Figura 4.8 (b).
- **V_{OL} (máximo)** - É o valor máximo de tensão de saída correspondente ao nível lógico baixo que um CI vai gerar, como apresentado na Figura 4.8 (b).

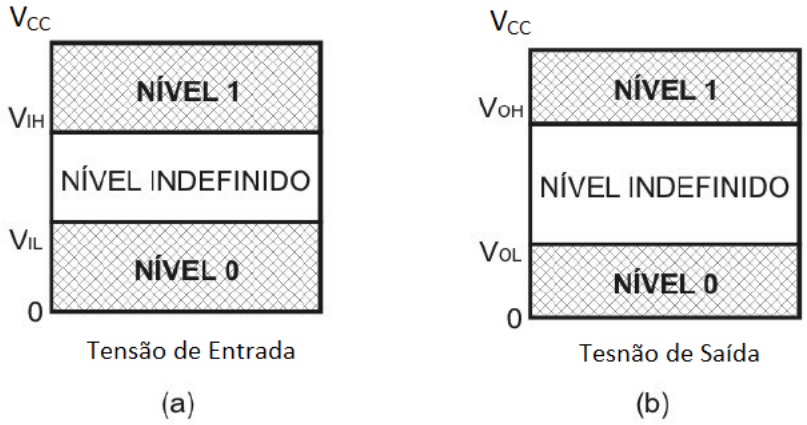


Figura 4.8

Se o dispositivo gerar uma tensão entre V_{OH} e V_{OL}, pode ser que o mesmo esteja danificado, ou uma má ligação entre os componentes foi feita.

De forma semelhante, uma porta lógica pode aceitar na sua entrada um valor menor que V_{CC}/V_{DD} para representar o nível lógico 1, e um valor maior que 0 V para representar o nível lógico 0:

- **V_{IH}** - É o valor mínimo de tensão de entrada correspondente ao nível lógico alto, como apresentado na Figura 4.8 (a).
- **V_{IL}** - É o valor máximo de tensão de entrada correspondente ao nível lógico baixo, como apresentado na Figura 4.8 (a).

Se o dispositivo receber uma tensão entre V_{IH} e V_{IL}, não há garantia de como ele irá interpretar esse valor.

4.3 Fan-out

Em geral, a saída de um circuito lógico é projetada para alimentar várias entradas de outros circuitos lógicos. O ***fan-out***, também chamado de **fator de carga**, é definido como o número máximo de entradas de circuitos lógicos que uma saída pode alimentar de maneira confiável. Por exemplo, uma porta lógica com *fan-out* de 3 pode alimentar até 3 portas lógicas padrão como mostra Figura 4.9. Se tal número não for respeitado, os níveis de tensão na saída do circuito poderão não respeitar as especificações.

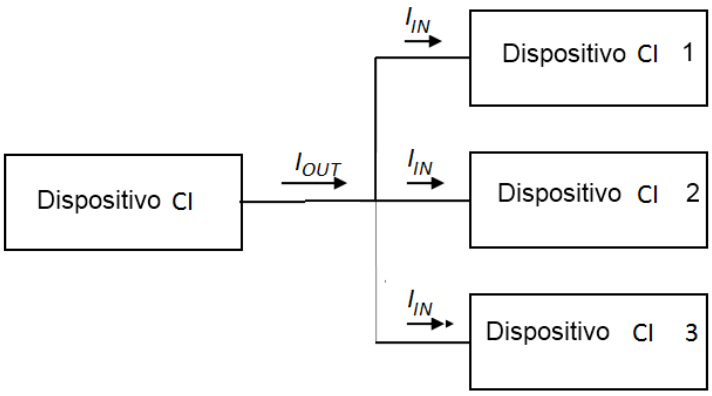


Figura 4.9

4.4 Outras terminologias

Outras características são importantes em determinados projetos, como **exigências de alimentação**, **imunidade ao ruído** e **tempo de retardo de propagação**. Entretanto, ficará a cargo de leitor mais curioso à pesquisa sobre isso nas referências bibliográficas.

4.5 Família TTL

Em 1964, a ***Texas Instruments Corporation*** introduziu a primeira linha de circuitos integrados **TTL-padrão**. As séries denominadas **54/74** formam uma das famílias lógicas de integrados mais utilizadas até hoje. Ambas as séries utilizam uma tensão de alimentação de +5v, sendo que a diferença entre elas é de que a série 74 opera de forma confiável com Vcc na faixa de 4,75 à 5,25v, enquanto a série 54 tolera variações de tensão de alimentação na faixa de 4,5v a 5,5v. Outra diferença é que a série 74 foi projetada para operar em

temperaturas ambientes de 0ºC a 70ºC, enquanto que a série 54 opera na faixa de -55ºC a +125ºC. Em razão de sua maior tolerância às variações de temperatura e de tensão, os CIs da série 54 são mais caros do que os da série 74, sendo empregados em aplicações onde a confiabilidade da operação precise ser mantida em condições extremas, tais como aplicações militares e espaciais.

Atualmente, não só a Texas, mas também um grande número de outros fabricantes produzem circuitos integrados TTL. Felizmente, todos utilizam o mesmo sistema de numeração para identificar os diversos CIs da família, de forma que a descrição do integrado associada a uma numeração é a mesma, não importando qual o fabricante do CI. Cada fabricante acrescenta um prefixo próprio à descrição do CI. Por exemplo, a Texas usa o prefixo **SN**, a ***National Semiconductor*** adota o prefixo **DM** e a ***Signets*** adota o prefixo **S**. Desta forma, dependendo do fabricante, você poderá encontrar um chip NOR quádruplo, denominado DM7402, SN7402, S7402 ou alguma outra designação relativa a outro fabricante.

4.5.1 Outras séries TTL

Várias outras séries TTL foram desenvolvidas depois da TTL-padrão. Estas outras séries fornecem uma ampla variedade de escolha dos parâmetros de velocidade e potência consumida. Alguns comentários sobre cada série:

- **Série TTL 74L de baixa potência**: Os circuitos são essencialmente os mesmos da série TTL-padrão, exceto pelo fato de seus resistores internos serem todos de maior valor, fazendo com que esta série se caracterize por um baixo consumo de potência a custa de um sensível aumento no tempo de retardo de propagação. Os dispositivos TTL de potência baixa são numerados 74L00, 74L04, 74L02, etc.
- **Série TTL 74H de alta velocidade**: Diminuindo as resistências internas, um fabricante pode abaixar as constantes de tempo internas, diminuindo o tempo de retardo de propagação. Porém, as resistências menores aumentam a dissipação de potência. Dispositivos deste tipo são numerados 74H00, 74H04, 74H02, etc.
- **Série TTL 74S Schottky:** Estes dispositivos apresentam praticamente a mesma dissipação de potência da série TTL 74H, porém operam com o dobro da velocidade. Isto é conseguido devido ao acréscimo de diodos schottky no circuito interno do dispositivo. Estes são numerados 74S00, 74S04, 74S02, etc.
- **Série TTL 74LS de baixa potência:** Aumentando as resistências internas, bem como utilizando diodos schottky, os fabricantes alcançaram um melhor compromisso entre baixa potência e alta velocidade. Estes dispositivos são numerados 74LS00, 74LS04, 74LS02, etc.

- **Série TTL 74AS Schootky Avançada (AS-TTL):** Esta série tem um conjunto de aperfeiçoamentos em relação a 74S, principalmente no que concerne a velocidade de operação com um consumo de potência extremamente baixo. Estes dispositivos são numerados 74AS00, 74AS04, 74AS02, etc.
- **Série TTL 74ALS Schoottky Avançada de Baixa Potência (74ALS-TTL):** Esta série oferece uma sensível melhora em relação à série 74LS no que diz respeito à velocidade de operação e a potência consumida. Estes dispositivos são numerados 74ALS00,74ALS04, 74ALS02, etc.

4.5.2 Entradas desconectadas (em flutuação)

Qualquer entrada de um circuito TTL que é deixada desconectada (aberta ou não-alimentada) age exatamente como se o **nível lógico 1** estivesse aplicado a ela. Isto significa que, em qualquer CI TTL, todas as entradas serão 1 se não estiverem conectadas a nenhuma fonte de sinal lógico ou à terra. Quando uma entrada for deixada aberta, diz-se que a mesma está em **flutuação**.

4.6 Família CMOS

As principais vantagens do MOSFET residem no fato de ele ser relativamente simples, de ter um custo de fabricação bem baixo, de ser pequeno e de consumir muito pouca potência. A fabricação dos CIs CMOS é menos complexa que a fabricação dos CIs TTL. Além disso, os MOSFETS ocupam muito menos espaço no chip do que os transistores bipolares. Um transistor bipolar ocupa cinquenta vezes mais espaço no chip que o MOSFET. Um outro aspecto sobre a tecnologia CMOS é o fato de seus CIs não usarem resistores na sua construção. Os resistores usam parte da área de chip ocupada pelos CIs TTL.

Em resumo, os CIs CMOS podem acomodar um número muito maior de elementos de circuito em uma área de chip do que os CIs TTL. Esta vantagem está evidenciada no fato de que a tecnologia CMOS ser mais utilizada que a TTL nas técnicas de maior densidade de integração (LSI, VLSI). A alta densidade de integração dos CIs CMOS permite a construção de sistema de alta confiabilidade, em virtude da redução do número de conexões externas necessárias à implementação de determinada função lógica.

A principal desvantagem da técnica CMOS é a velocidade de operação relativamente baixa de seus componentes, se comparada com as apresentadas por componentes da famílias TTL. Em muitas aplicações, a velocidade de operação não é um fator chave para

o sucesso do projeto. Nesses casos, a tecnologia CMOS transforma-se numa alternativa muito superior a lógica TTL.

4.6.1 Características das séries CMOS

Dentro da família CMOS de CIs, existem várias séries, entre as quais, destacam-se:

- **Série 4000 / 14000:** Primeiras séries da família CMOS, sendo ainda bastante utilizadas em função de implementarem diversas funções ainda não disponíveis nas novas séries.
- **Série 74C:** Esta série é compatível pino por pino, função por função, com dispositivos TTL de mesmo número. Isto possibilita a substituição de alguns circuitos TTL por seu equivalente desenvolvido na categoria CMOS. Sua performance é idêntica à da série 4000.
- **Série 74HC (CMOS de alta velocidade):** É uma versão melhorada da série 74C. A velocidade dos dispositivos desta série é comparável com a velocidade dos dispositivos da série TTL 74LS. Outra melhoria da série 74HC é a sua capacidade de suportar altas correntes na sua saída.
- **Série 74HCT:** Esta também é uma série CMOS de alta velocidade. A principal diferença entre esta série e a 74HC é o fato de ela ser desenvolvida para ser compatível em termos de tensões com dispositivos da família TTL.

A seguir, comentaremos algumas das características operacionais e de performance das séries CMOS.

- **Tensão de alimentação:** As séries 4000 e 74C operam com valores de tensão na faixa de 3 a 15 volts, de forma que a regulação de tensão destas fontes não constitui um ponto crítico para tais séries. Já as séries 74HC e 74HCT operam com a alimentação na faixa de 2 a 6v. Se dispositivos CMOS e TTL forem utilizados em conjunto, a tensão de alimentação deverá ser de 5v.
- **Níveis de tensão:** Quando as saídas CMOS só estiverem alimentando entradas CMOS, os níveis de tensão de saída serão muito próximos a 0v para nível lógico baixo e +Vdd para o nível alto..

- **Entradas não-utilizadas:** As entradas de um circuito CMOS não podem nunca ser deixadas desconectadas. Todas as entradas CMOS desnecessárias a uma particular aplicação devem ser conectadas a uma fonte de tensão fixa (0 v ou +Vdd), ou a uma outra porta de entrada.
- **Sensibilidade à eletricidade estática:** Todos os dispositivos eletrônicos, em maior ou menor grau, são sensíveis a danos causados pela ação da eletricidade estática. O corpo humano armazena uma grande quantidade de eletricidade estática. Por exemplo, quando caminhamos por um carpete, uma carga estática de 30000v pode distribuir-se por nosso corpo. Se tocarmos um dispositivo eletrônico, parte desta carga poderá ser transferida pelo dispositivo, podendo vir a danificá-lo.

A lógica CMOS é especialmente suscetível a danos causados pela eletricidade estática, enquanto que a família TTL não é tão afetada. Isto porque, a família CMOS apresenta uma alta impedância de entrada, uma pequena carga estática fluindo por esta alta impedância resulta em uma tensão muito alta. Esta mesma carga, fluindo por sobre a baixa impedância da entrada TTL, produzirá uma tensão bem menor, com menor possibilidade de causar danos ao circuito.

A seguir são listadas algumas das precauções adotadas para a utilização de circuitos CMOS:

- Conecte a terra o chassi de todos os instrumentos de teste, estações soldadoras e a própria bancada (se for de metal). Isto previne o aparecimento de carga estática sobre tais dispositivos, carga esta que poderia ser transferida para qualquer placa do circuito impresso ou para qualquer CI que seja colocado em contato com eles.
- Conecte-se você a terra através de uma pulseira especial. Isto fará com que as cargas estáticas de seu corpo sejam descarregadas para a terra. A pulseira contém um resistor de 1M ohm que limita a corrente a um valor não letal, caso você acidentalmente toque em uma fonte de tensão viva enquanto estiver trabalhando.
- Mantenha os CIs, em especial os CMOS, guardados em espuma condutiva ou embalagens de alumínio. Isto manterá todos os pinos do CI em curto-circuito juntos, de maneira que não haverá possibilidade do surgimento de tensão entre dois quaisquer de seus pinos.

- Armazene as placas de circuito impresso em plástico condutivo ou envelopes metálicos.
- Não deixe desconectada nenhuma entrada de qualquer CI que não esteja sendo utilizado, pois entradas abertas tendem a capturar cargas estáticas espúrias.

4.7 Resistores de *pull-up* e *pull-down*

Considere o exemplo em que se queira alimentar um pino com lógica ZERO, quando uma chave é pressionada e com lógica UM quando a chave é solta.

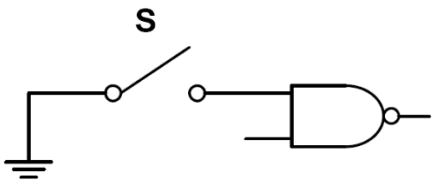


Figura 4.10

Não é bom deixar os pinos de um CI TTL desconectados e não se deve deixar pinos CMOS desconectados. Para contornar esse problema, coloca-se um resistor com valor alto de resistência (geralmente entre 3,3 kΩ e 10 kΩ) na entrada do pino, ligado à alimentação (nível lógico UM), como mostra Figura 4.11 (a). Quando a chave estiver aberta, uma corrente pequena, mas suficiente irá fluir para a porta, indicando nível lógico UM. Quando estiver fechada, o nível lógico ZERO irá predominar no pino. Chama-se esse resistor de ***pull-up*** (tradução grosseira: **puxa o nível ALTO**).

Se, por outro lado, desejar que ao pressionar a chave, seja nível UM no pino, e ao soltar seja nível ZERO, utiliza-se um resistor de ***pull-down*** (tradução grosseira: **puxa o nível BAIXO**), como apresentado na Figura 4.11 (b).

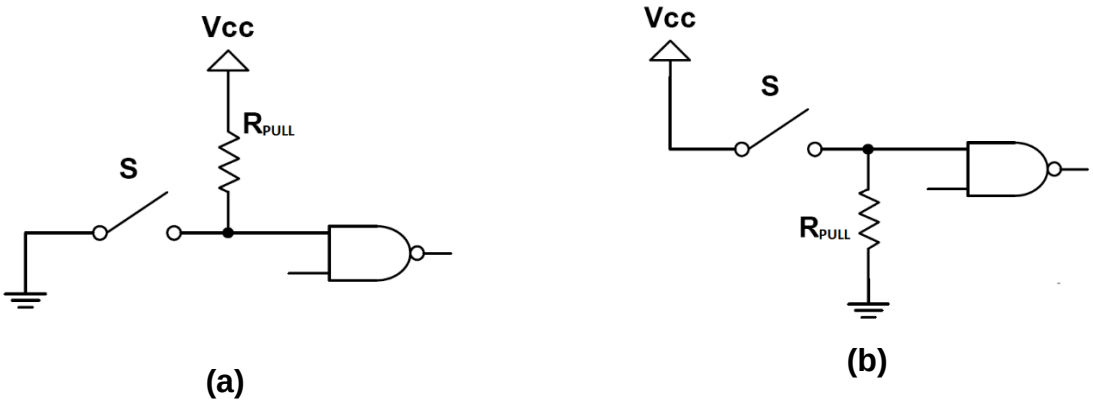


Figura 4.11

5 CIRCUITOS COMBINACIONAIS DE INTERCONEXÃO

Servem para interligar as diferentes partes de um sistema digital.

5.1 Multiplexadores

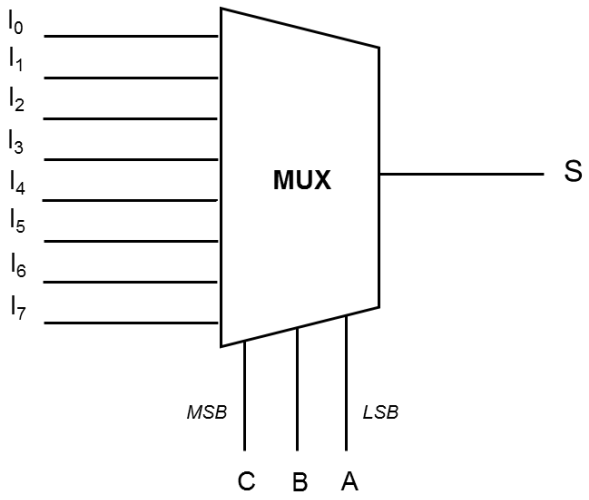


Figura 5.1

O **circuito multiplexador** (**MUX**), cujo símbolo foi representado na Figura 6.1, possui várias **entradas de informações**, identificadas como I_0, I_1, I_2, ..., I_7 e um número menor de **entradas de endereço**, identificadas como C, B e A.

De acordo com o nível de tensão aplicado às entradas de endereço C, B e A, teremos uma única entrada de informação conectada à saída. Por exemplo, suponhamos que nas entradas de endereço C, B e A temos, respectivamente, $101_2 = 5_{10}$, então, na saída (S) do multiplexador teremos o mesmo nível de tensão (0 ou 1) que está presente na entrada I_5, ou seja, somente a entrada I_5 será conectada à saída do multiplexador.

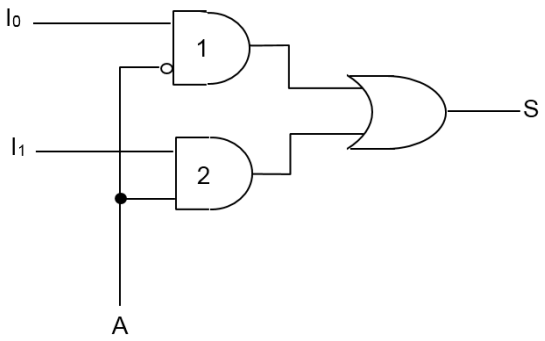
A relação entre o número de entradas de informações (N) e o número de entradas de endereços (P) é dada pela equação:

$$N = 2^P$$

Assim, para construirmos um multiplexador com 16 entradas de informações seriam necessárias 4 entradas de endereço, pois $2^4 = 16$. Podemos utilizar o mesmo raciocínio para qualquer número de entradas de informações.

5.1.1 Circuitos internos de um MUX

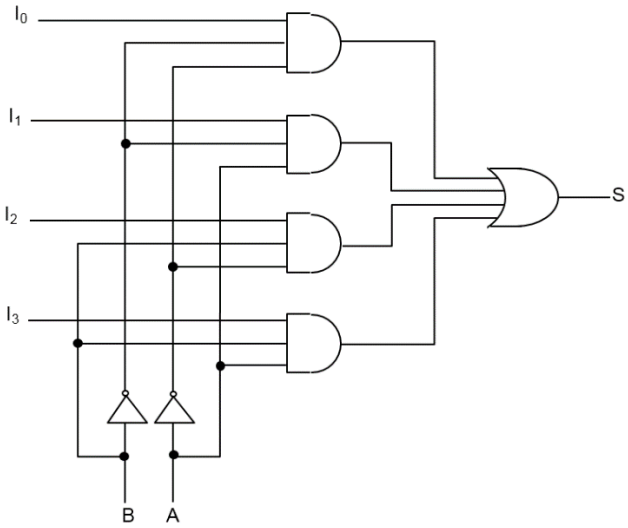
Com duas entradas de informações:



A	S
0	I_0
1	I_1

Figura 5.2

Com quatro entradas de informações:



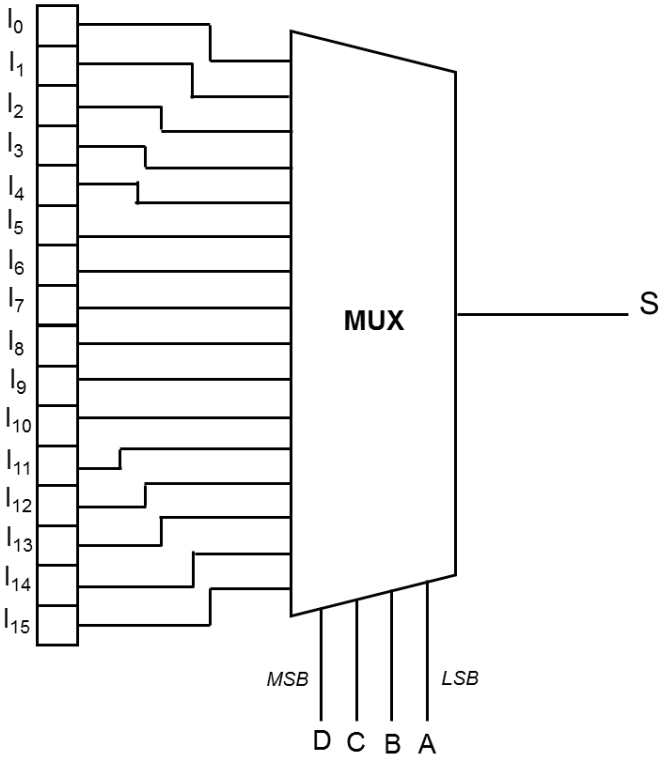
B	A	S
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

Figura 5.3

5.1.2 Utilização do MUX na construção de circuitos combinacionais

O circuito multiplexador pode ser utilizado também para montarmos circuitos combinacionais. Nas entradas de endereço do MUX ligamos as variáveis de entrada (...D, C, B, A) e nas entradas de informação colocamos o valor (0 ou 1) que desejamos obter na saída, para cada combinação das variáveis de entrada da tabela-verdade.

Ex.: Dada a tabela-verdade abaixo, monte o circuito utilizando um multiplexador de 16 entradas de informações.



D	C	B	A	S
0	0	0	0	**1**
0	0	0	1	**0**
0	0	1	0	**1**
0	0	1	1	**0**
0	1	0	0	**0**
0	1	0	1	**0**
0	1	1	0	**1**
0	1	1	1	**0**
1	0	0	0	**1**
1	0	0	1	**0**
1	0	1	0	**1**
1	0	1	1	**1**
1	1	0	0	**1**
1	1	0	1	**1**
1	1	1	0	**1**
1	1	1	1	**1**

Figura 5.4

5.2 Demultiplexadores

Os circuitos **demultiplexadores** (**DEMUX**), cujo símbolo é representado na Figura 5.5, possuem uma entrada de informação, identificada pela letra “I” e várias saídas de informações identificadas como S_0, S_1, S_2,..., S_7. Dependendo do nível de tensão (0 ou 1) aplicado às entradas de endereço, teremos a entrada de informação do circuito conectada a uma única saída de informação. Por exemplo, suponhamos que os níveis de tensão das entradas de endereço C, B e A são, respectivamente, $110_2 = 6_{10}$. Neste caso, teremos na

saída S_6 o mesmo nível de tensão que temos na entrada de informação I, ou seja, se na entrada de informação I tivermos nível lógico “1”, na saída S_6 teremos nível lógico “1”, se na entrada de informação tivermos nível lógico “0”, na saída S_6 teremos nível lógico “0”. As saídas que não estão conectadas a entrada de informação apresentam nível lógico “1”.

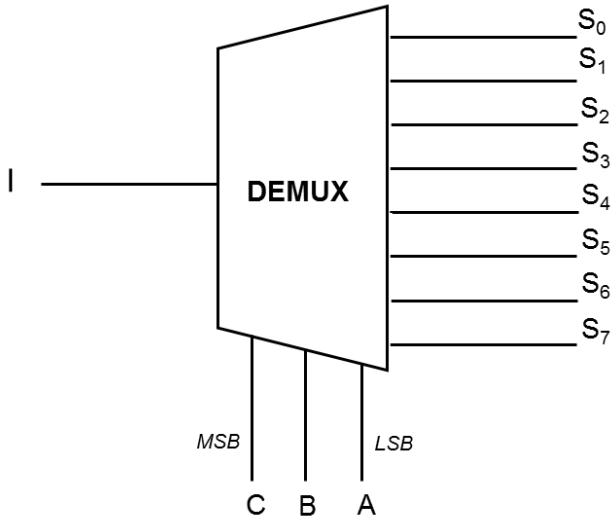
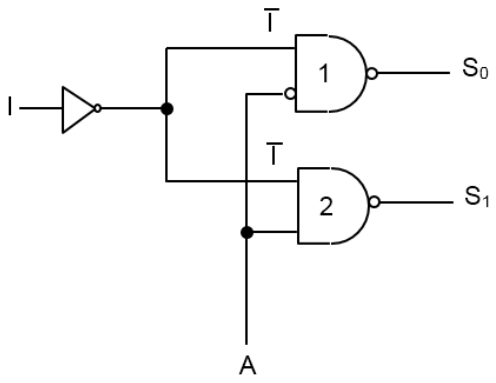


Figura 5.5

A relação que existe entre o número de saídas de informações e o número de entradas de endereço de um demultiplexador é o mesmo que existe entre o número de entradas de informações e o número de entradas de endereço de um multiplexador.

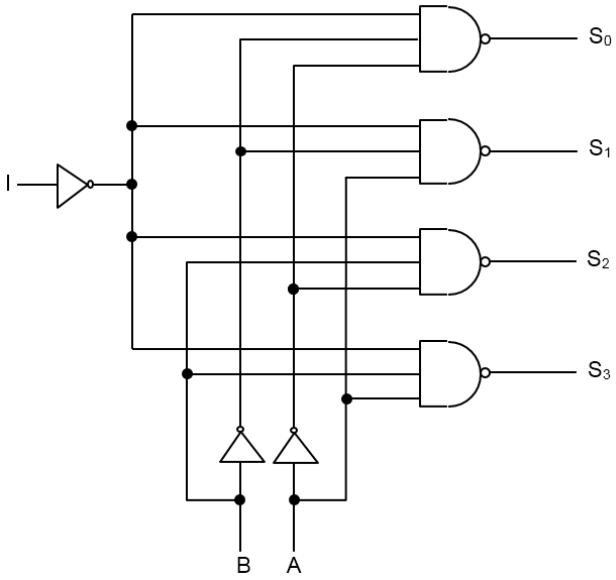
5.2.1 Circuitos internos de um DEMUX

Com duas entradas de informações:



A	S_0	S_1
0	*I*	1
1	1	*I*

Com quatro entradas de informações:



B	A	S_0	S_1	S_2	S_3
0	0	*I*	1	1	1
0	1	1	*I*	1	1
1	0	1	1	*I*	1
1	1	1	1	1	*I*

5.3 MUX e DEMUX em CIs da Familia TTL

Existem diversos modelos de MUX e DEMUX na família TTL, diferenciando-se pela quantidade de entradas/saídas e pela quantidade de componentes em um mesmo chip. Dentre os CIs, tem-se o 74150, como apresentado na Figura 5.6.

Figura 5.6

Este CI é um multiplexador com 16 entradas de informações. Em operação normal, a entrada de habilitação $\overline{EN}$ ou (***Strobe***) deve ser igual a “0”, se $\overline{EN}$ for igual a “1”, a saída do circuito terá nível lógico “1”, independentemente do nível de tensão das entradas de informação e das entradas de endereço.

Com $\overline{EN}$ = 0, o conteúdo da entrada selecionada pelas entradas de endereço aparece invertido na saída $\overline{S}$.

Da mesma forma, tem-se o 74155, um CI quem contém dois demultiplexadores de 2 entradas de informações, como apresentado na Figura 5.7.

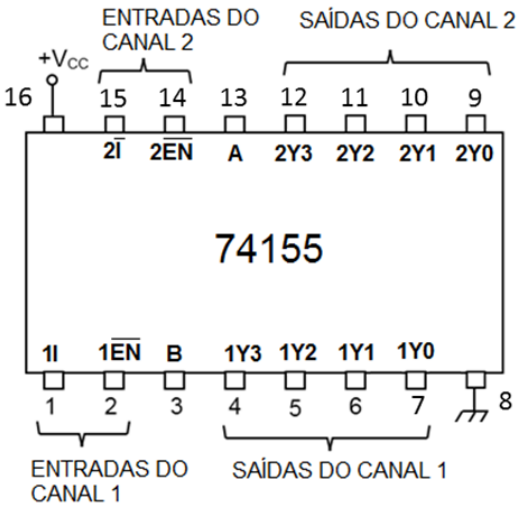


Figura 5.7

As entradas 1$\overline{EN}$ e 2$\overline{EN}$ habilitam os canais 1 e 2, respectivamente, assim como o $\overline{EN}$ do 74150. Quando cada um desabilitado, o canal correspondente terá todas suas saídas no nível lógico “1”. Quando o canal 1 estiver habilitado, o conteúdo da entrada 1I aparecerá invertido na saída selecionada entre 1Y3 à 1Y0, pelas entradas de endereço A e B. Quando o canal 2 estiver habilitado, o conteúdo da entrada 2$\overline{I}$ aparecerá na saída selecionada entre 2Y3 à 2Y0, pelas entradas de endereço A e B.

5.4 O Relé

Antes de prosseguir para a parte prática, vamos entender o funcionamento do **relé**, um dispositivo eletromecânico, cuja função é permitir (ALTO), ou não (BAIXO), a passagem de altos valores de corrente, suportando grandes tensões, através de uma corrente/tensão menor.

Para entender o funcionamento dos relés, observe a Figura 5.8, onde são apresentadas sua estrutura, simbologia e um típico dispositivo (JQC-3F), e a relação entre os terminais. Um contato denominado de armadura, preso à uma mola, faz a ligação entre o comum e um entre dois terminais: **Normalmente Fechado** (**NF**) e **Normalmente Aberto** (**NA**).

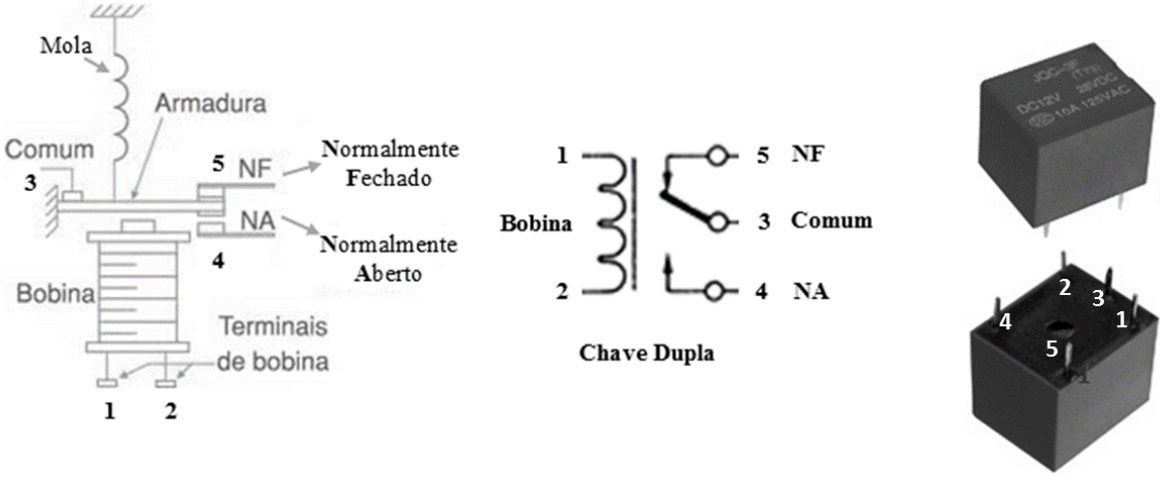


Figura 5.8

Quando é aplicada uma corrente nos terminais da bobina, a mesma gera um campo magnético que atrai a armadura e a faz entrar em contato com o terminal NA. Quando não há corrente na bobina, o terminal NF fica conectado à armadura.

Dessa forma, a corrente na bobina é responsável por realizar o chaveamento na saída do relé. A saída do relé pode ser ou os terminais NF e comum, ou NA e comum, dependendo da aplicação.

Alguns exemplos de cargas de maior potência seriam uma lâmpada, um motor, etc. Além do relé, é utilizado geralmente um transistor em conjunto, como apresentado na Figura 5.9. O transistor NPN controla a bobina do relé, pois os CIs TTL/CMOS, não podem fornecer a corrente necessária para ativar o mesmo. Já o relé é responsável por abrir ou fechar o circuito que alimenta uma lâmpada.

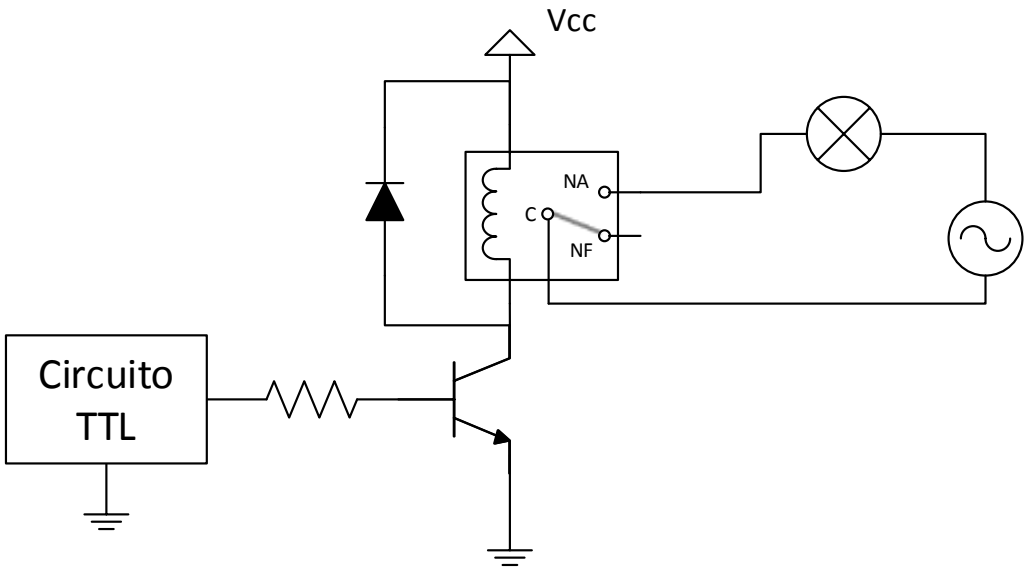


Figura 5.9

O diodo em paralelo com o relé como apresentado na figura serve como proteção, pois ao ligar os contatos de saída do relé, um pico de corrente reversa pode danificar o transistor

5.5 Codificadores e decodificadores

Um **codificador** torna possível a passagem de um código conhecido (decimal, por exemplo) para um desconhecido (binário, por exemplo). Um **decodificador** tem a função inversa.

5.5.1 Codificador Decimal/Binário

Aceita N entradas, onde cada uma representa um número decimal, e gera o número binário correspondente na saída, como apresentado nas Figura 5.10 e Figura 5.11.

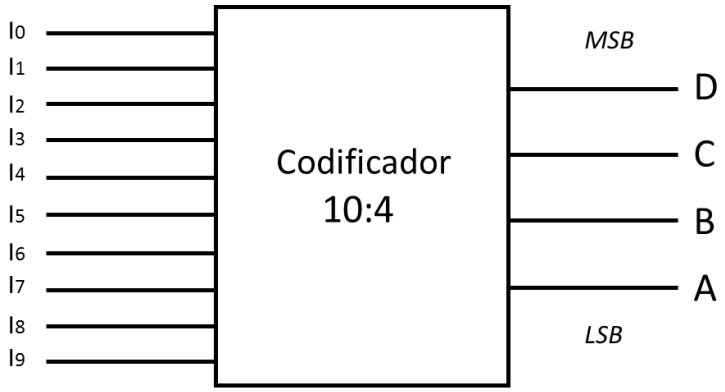


Figura 5.10

I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	D	C	B	A
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***	***0***
0	0	0	***1***	0	0	0	0	0	0	***0***	***0***	***1***	***1***
0	0	0	0	***1***	0	0	0	0	0	***0***	***1***	***0***	***0***
0	0	0	0	0	***1***	0	0	0	0	***0***	***1***	***0***	***1***
0	0	0	0	0	0	***1***	0	0	0	***0***	***1***	***1***	***0***
0	0	0	0	0	0	0	***1***	0	0	***0***	***1***	***1***	***1***
0	0	0	0	0	0	0	0	***1***	0	***1***	***0***	***0***	***0***
0	0	0	0	0	0	0	0	0	***1***	***1***	***0***	***0***	***1***

Figura 5.11

Quando um digito decimal é convertido em um correspondente binário de 4 bits, dizemos que ele foi codificado em **BCD** (***Binary Coded Decimal*: decimal codificado em binário**). Um número com 3 dígitos seria codificado em BCD da seguinte forma:

$$237_{10} = \underline{0010}\ \underline{0011}\ \underline{0111}_{BCD}$$

5.5.2 Decodificador Binário/Decimal

Aceita um código BCD na entrada e ativa a saída correspondente ao número decimal, como apresentado nas Figura 5.12 e Figura 5.13.

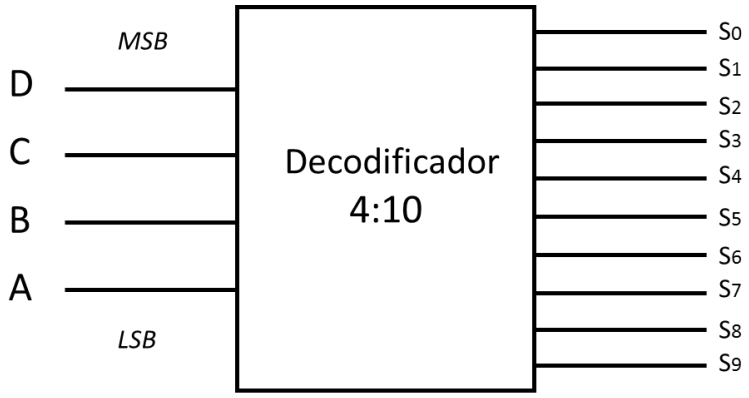


Figura 5.12

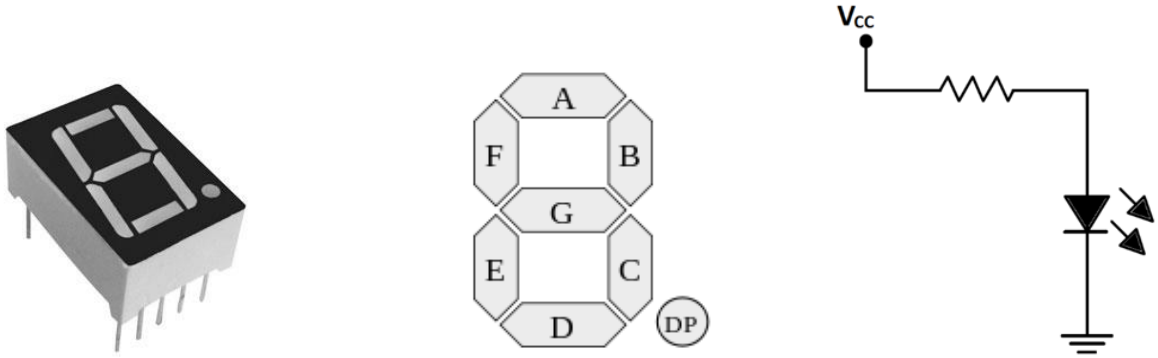
D	C	B	A	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9
0	0	0	0										
0	0	0	1										
0	0	1	0										
0	0	1	1										
0	1	0	0										
0	1	0	1										
0	1	1	0										
0	1	1	1										
1	0	0	0										
1	0	0	1										

Figura 5.13

5.5.3 Decodificador BCD/Display 7 segmentos

O display de 7 segmentos, como a apresentado na Figura 5.14 (a), é um dos displays mais utilizados na atualidade. Pode mostrar um número de 0 à F através do acendimento combinado de certo número de seus 7 segmentos, mais um segmento que indica ponto.

Os segmentos são identificados por letras de A à G, como apresentado na Figura 5.14 (b).



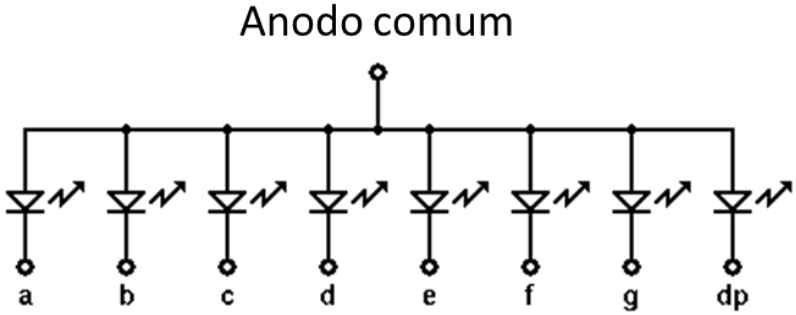
(a) **(b)** **(c)**

Figura 5.14

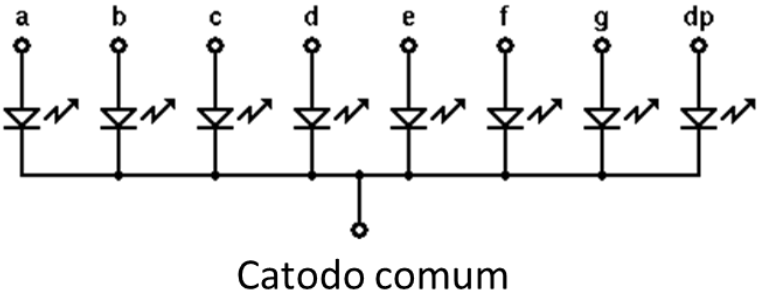
Os segmentos geralmente são LEDS (diodos emissores de luz). Dessa forma, quando polarizado diretamente, o LED no segmento ascende, como mostra Figura 5.14 (c). Os LEDs de displays de 7 segmentos operam em uma corrente de 10 à 20 mA, e uma tensão entre 1,5 V à 3,3 V. Para limitar a corrente quando trabalha-se com tensões maiores que sua tensão nominal (Vcc, por exemplo), utilizam-se resistores em série.

Os displays podem ser classificados em:

- **Catodo comum**: neste tipo de display todos os LEDs estão interligados pelo catodo que vai ao terra. Devemos aplicar nível lógico “1” para acender o segmento do display. Ex.: FND 560 (ver Figura 5.15 (a))
- **Anodo comum**: No display tipo anodo comum todos os Led’s estão interligados pelo anodo que vai à Vcc. Devemos aplicar nível lógico “0” para acender o segmento do display. Ex.: FND 567 (ver Figura 5.15 (b))



(a)



(b)

Figura 5.15

O decodificador BCD/7 segmentos faz com que o número decimal equivalente ao código BCD das entradas seja mostrado em um display de sete segmentos, como apresentado nas Figura 5.16 e Figura 5.17.

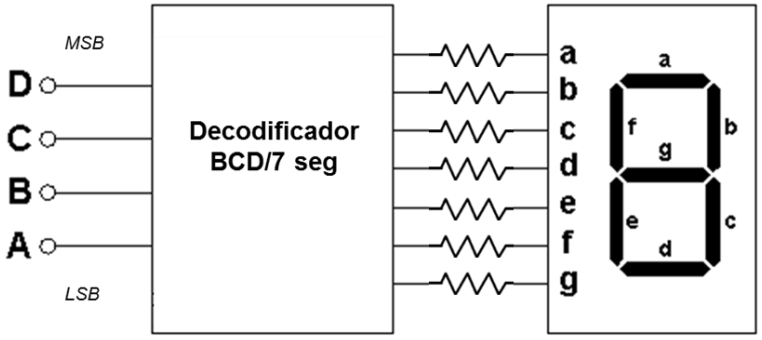


Figura 5.16

Número	D	C	B	A	A	b	c	d	e	f	g
0	0	0	0	0							
1	0	0	0	1							
2	0	0	1	0							
3	0	0	1	1							
4	0	1	0	0							
5	0	1	0	1							
6	0	1	1	0							
7	0	1	1	1							
8	1	0	0	0							
9	1	0	0	1							
10	1	0	1	0							
11	1	0	1	1							
12	1	1	0	0							
13	1	1	0	1							
14	1	1	1	0							
15	1	1	1	1							

Figura 5.17

5.6 Decodificadores BCD/7 seg em CIs da família TTL

5.6.1 Decodificador 7448

Como apresentado na Figura 5.18, o 7448 é um circuito decodificador onde as saídas serão ativadas quando apresentarem nível alto "1". Este CI é adequado para controlar um display de sete segmentos de catodo comum.

Figura 5.18

5.6.2 Decodificador 7446/7447

É um decodificador decimal que também pertence à família TTL, no qual as saídas são ativas em nível baixo (0). Portanto, este circuito integrado é adequado para controlar um display cujos segmentos se acendam com nível "0" (anodo comum).

A pinagem é idêntica à do circuito integrado 7448. Como visto, a única diferença é de que as saídas (a, b,...., g) são negadas, isto é, ativas em nível baixo.

6 SISTEMAS DE NUMERAÇÃO

Outros sistemas de numeração também são uteis quando trabalha-se com circuitos digitais, como o **hexadecimal** que possui 16 dígitos diferentes e o **octal** que possui 8 dígitos diferentes, entretanto o uso do octal decaiu. A relação entre esses sistemas de numeração é apresentada na Tabela 6-1.

Tabela 6-1

DECIMAL	BINÁRIO	OCTAL	HEXADECIMAL
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	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
16	10000	20	10

Para diferenciar os números dos diferentes sistemas de numeração quando for conveniente, cada valor terá um sobrescrito do seu sistema. Exemplo:

- Decimal: 24_{10}, 12_{10}, 10010_{10};
- Binário: 10_2, 1101_2, 10000_2;
- Hexadecimal: 10_{16}, E_{16}, $A9E_{16}$.

6.1 Alguns conceitos

- **Byte**: número de 8 bits.
- **Nibble**: número de 4 bits.
- **kiloBit** (kb): 2^{10} (1024) bits.
- **kiloByte** (kB): 2^{10} (1024) bytes = 8192 (8 x 1024) bits.
- **megaBit** (Mb): 2^{20} (1024 x 1024) bits
- **megaByte** (MB): 2^{20} (1024 x 1024) bytes

- **gigaBit** (Gb): 2^{30} (1024 x1024 x 1024) bits
- **gigaByte** (GB): 2^{30} (1024 x1024 x 1024) bytes
- **MSB** (***Most Significant Bit* - bit mais significativo**): bit mais à esquerda. Ex.: **1**0011
- **LSB** (***Least Significant Bit* - bit menos significativo**): bit mais à direira. Ex.: 1001**1**
- **MSD** (***Most Significant Digit* - digito mais significativo**): digito mais à esquerda. Ex.: $\mathbf{1}0011_2$, $\mathbf{4}67_{10}$, $\mathbf{3}EC_{16}$
- **LSD** (***Least Significant Digit* - digito menos significativo**): digito mais à direira. Ex.: $1001\mathbf{1}_2$, $46\mathbf{7}_{10}$, $3E\mathbf{C}_{16}$

6.2 Conversão de binário para decimal

Todo digito no sistema decimal tem um **peso**. Por exemplo:

$$4679_{10} = 4000 + 600 + 70 + 9$$

Note que o 4 representa um valor (peso) maior no número, já o 9 possui um peso menor. Cada digito tem um peso associado.

- $4000 = 4 \times \mathbf{10^3}$
- $600 = 6 \times \mathbf{10^2}$
- $70 = 7 \times \mathbf{10^1}$
- $9 = 9 \times \mathbf{10^0}$

Pode-se concluir que:

$$4679_{10} = 4 \times \underline{10^3} + 6 \times \underline{10^2} + 7 \times \underline{10^1} + 9 \times \underline{1}$$

Ou seja, o digito de menor peso (LSD), tem peso $10^0 = 1$, enquanto o peso do digito de maior peso (MSD) depende do número de dígitos. Se forem N dígitos será 10^{N-1}.

Os números binários também têm um peso decimal associado. Pode-se obter, por exemplo, os pesos dos dígitos de 10010_2 da seguinte forma:

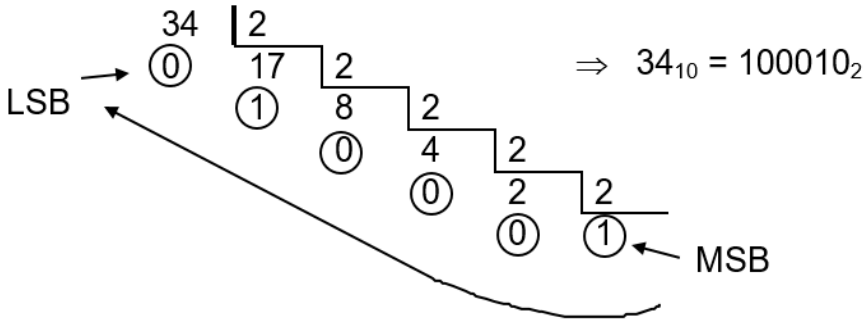
$$10010_2 = 1 \times \underline{2^4} + 0 \times \underline{2^3} + 0 \times \underline{2^2} + 1 \times \underline{2^1} + 0 \times \underline{2^0}$$

Realizando essa soma, obtêm-se o valor decimal correspondente:

$$16 + 0 + 0 + 2 + 0 = 18_{10}$$

6.3 Conversão de decimal para binário

Método das **divisões sucessivas**. Exemplo: Converter 34_{10} para binário.



6.4 Faixa de contagem no sistema binário

Quando trabalhamos com números binários na eletrônica digital, estamos restritos à um número especifico de bits. Por exemplo, na Figura 6.1 é apresentada todas as combinações da sequência de números que se pode obter com 4 bits.

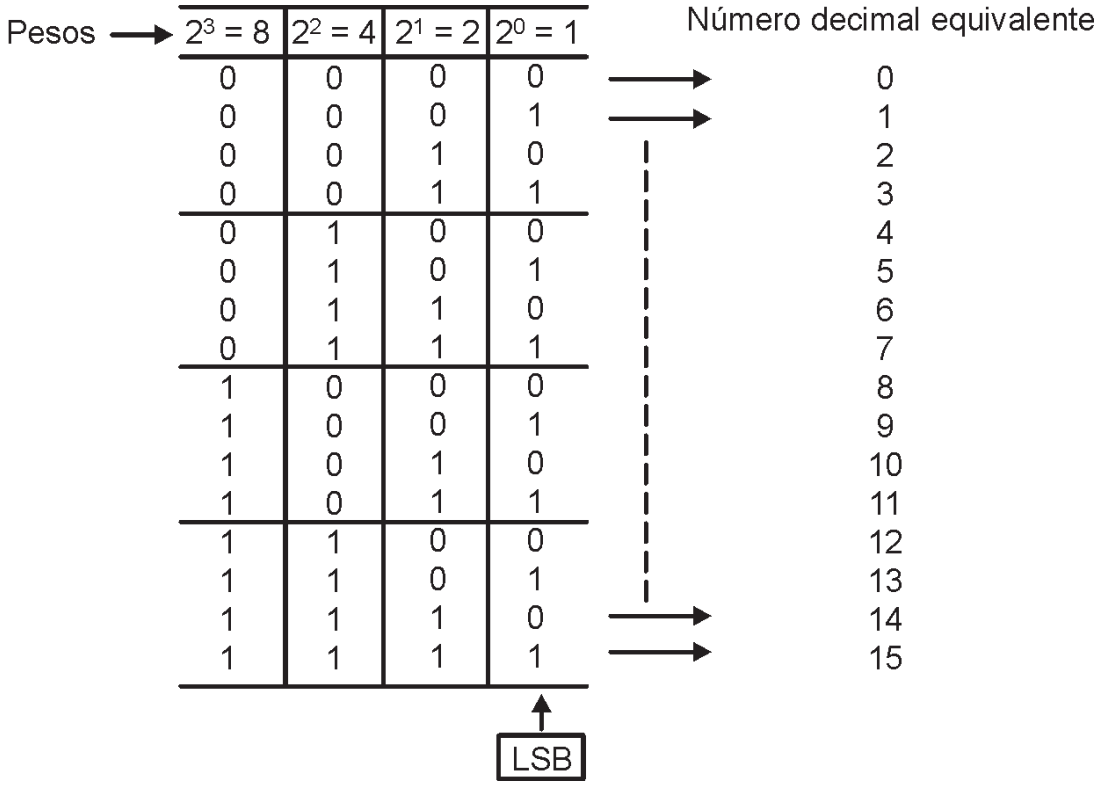


Figura 6.1

O primeiro número, o zero, da contagem tem todos os bits iguais a zero. Na sequência, os bits de menor peso alternam mais rapidamente, enquanto os de maior peso alternam mais lentamente. A alternância entre os bits numa mesma posição é igual ao peso do bit.

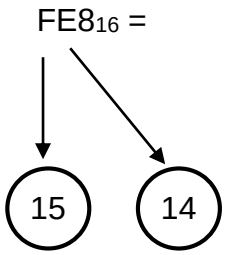
Com N bits podemos representar 2^N números binários diferentes, em uma faixa de 0 à $2^N - 1$ em decimal. Por exemplo, com 4 bits:

- Número de valores representáveis: $2^4 = 16$ números;
- Faixa de valores: 0000_2 (0_{10}) à 1111_2 (15_{10}).

6.5 Conversão de hexa para decimal

A conversão de hexadecimal para decimal é semelhante à conversão de binário para decimal, apenas que devemos converter os algarismos maiores do que 9 (letras) para decimal.

Exemplo:



$FE8_{16}$ = $15 \times \underline{16^2} + 14 \times \underline{16^1} + 8 \times \underline{16^0}$

$= 3840 + 224 + 8$

$= 4072_{10}$

6.6 Conversão de hexa para binário

Cada digito hexa é convertido no equivalente binário de 4 bits. Exemplo:

$9F2_{16}$ = 9 F 2

1001 1111 0010

$= 100111110010_2$

6.7 Conversão de binário para hexa

Separa-se o número em grupos de *nibbles*.

1110100110_2 = $\underline{00}11$ 1010 0110

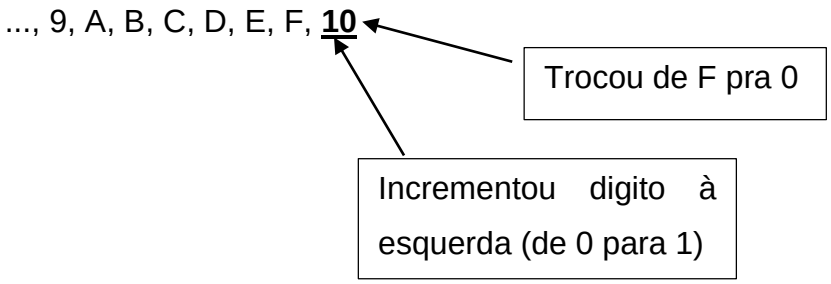
3 A 6

$= 3A6_{16}$

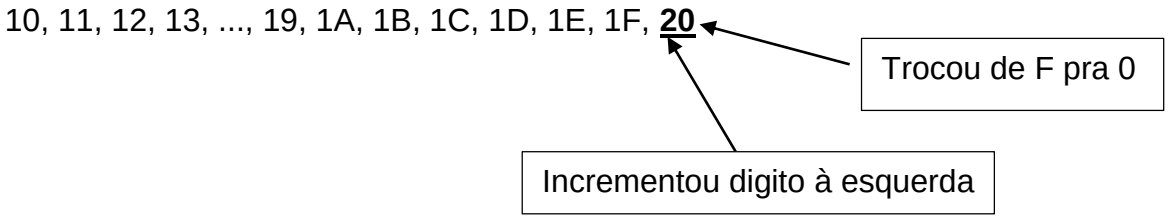
Acrescenta zeros a esquerda para formar *nibble.*

6.8 Faixa de contagem no sistema hexa

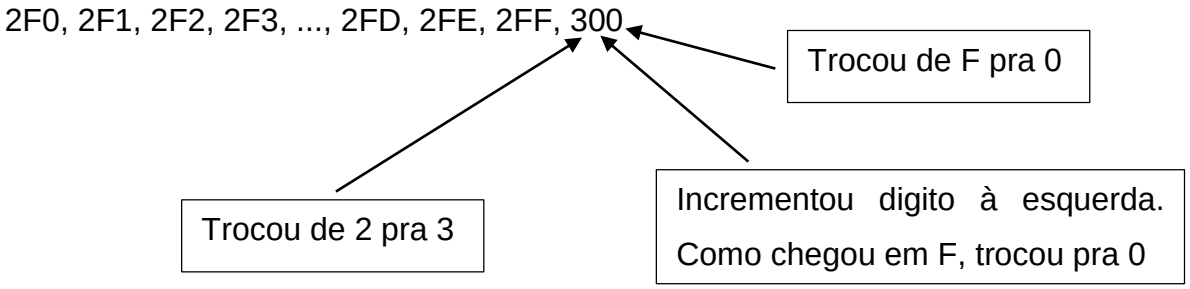
Quando contamos em hexa, cada digito pode ser incrementado de 0 à F. Quando o digito de uma posição chega ao valor F, este volta para 0 e o digito da próxima posição é incrementado. Exemplo:



Continuando:



Indo mais além:



Note que esse procedimento é o mesmo quando se está contando números decimais ou binários. Apesar disso, utilizamos a técnica de contagem em binário vista anteriormente, por ser mais fácil.

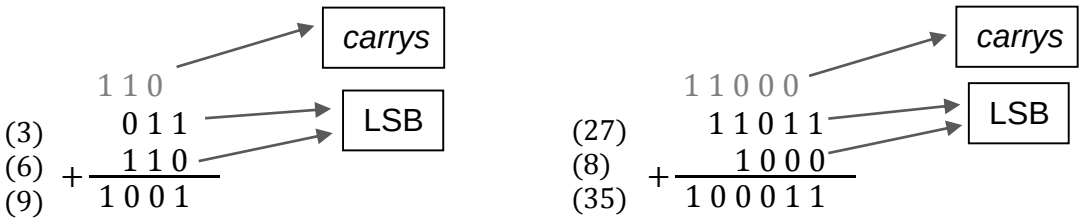
7 ARITMÉTICA DIGITAL

Os sistemas digitais (como computadores e calculadoras, por exemplo) realizam várias operações aritméticas com números representados no formato binário. Será visto nesse capitulo as operações aritméticas básicas realizadas por estes sistemas. Esse conhecimento é particularmente útil quando se programam **microcontroladores**.

7.1 Adição binária

A adição binária é realizada exatamente da mesma forma que a adição de números decimais.

Exemplo:

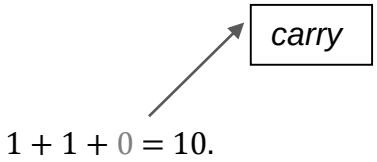


(a) **(b)**

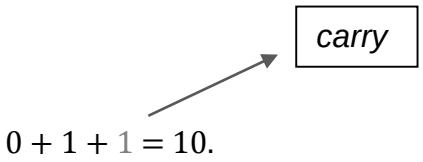
Os ***carrys*** são os bits de “vai um” que resultam da soma de dois bits numa mesma coluna. Por exemplo, em (a) começando a soma dos bits menos significativos resulta em:

$$1 + 0 = 1.$$

Nesse caso, o resultado foi 1, ou 01, pois o zero à esquerda não influencia no valor do número. Esse 0 à esquerda é o carry da soma. Usa-se esse resultado para somar os bits da próxima coluna:



Nesse caso, o resultado da soma é 10, de forma que o carry é o bit 1. Então, na próxima coluna:



Esse resultado gerou um último *carry*, na última coluna, que é o bit 1.

Não será entrado em detalhes acerca de subtração, já que os sistemas digitais realizam essa operação em forma de adição utilizando o método de **complemento de 2**, como será visto.

7.2 Multiplicação e divisão binária

Existem diversas formas para um sistema digital implementar multiplicação e divisão em números binários. Para multiplicação, uma forma, é aplicar **somas sucessivas**, onde o valor binário multiplicando é somado tantas vezes por ele mesmo, quanto for seu multiplicador. Para a divisão, o processo é semelhante, entretanto se faz **subtrações sucessivas**.

Outra forma, seria efetuar a multiplicação ou divisão da mesma forma que é realizada em números decimais. Isso faz com que o sistema digital realize apenas uma operação sem ter que realizar somas ou subtrações sucessivas, a preço de mais *hardware*.

Não será entrado em mais detalhes esse assunto, ficando a cargo do leitor mais curioso pesquisar nas referências.

7.3 Representação de números com sinal

7.3.1 Sinal Magnitude

Os sistemas digitais armazenam os números binários em **elementos de memória** para poder realizar operações sobre eles. Esses elementos de memória podem armazenar uma quantidade fixa de bits, de acordo com o sistema digital. Por exemplo, um dispositivo

que contém elementos de memória que armazenam 6 bits, podem representar valores na faixa de 000000_2 à 111111_2 (0_{10} à 63_{10}). Isso representa a **magnitude** do número.

Como a maioria dos sistemas digitais efetua operações tanto com números negativos como positivos, é necessário representar de alguma forma o sinal do número (+ ou -). Isso é feito normalmente acrescentando ao número um outro bit denominado **bit de sinal**.

Exemplo: Um elemento de memória com capacidade de 7 bits usará 6 bits para o valor absoluto (magnitude) e 1 bit para o sinal, como apresentado na Figura 7.1.

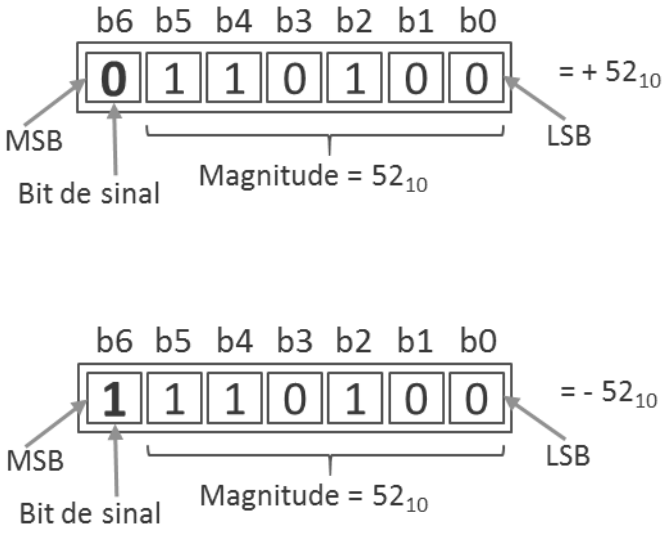


Figura 7.1

Esse sistema de representação é denominado **sinal-magnitude**. Embora, bastante direto para fins de compreensão, a implementação desse sistema em circuitos lógicos é mais complexa, além de ser possível duas representações para o zero, + 0 e - 0.

Dessa forma, o sistema mais usado para representar números binários com sinal é o **sistema em complemento de dois**.

7.3.2 Complemento de dois

Antes de saber como é esse sistema, temos de determinar o complemento de um e de dois de um número.

- **Forma de complemento de 1:**

Consiste em substituir cada 0 por 1 e cada 1 por 0, ou seja, seu complemento. Ex.:

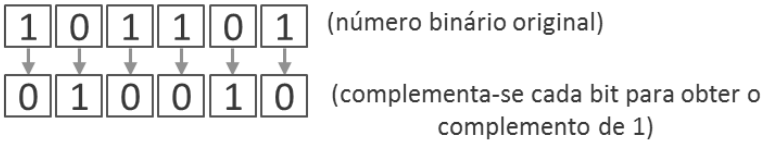


Figura 7.2

- **Forma de complemento de 2:**

Realiza-se o complemento de 1 e soma-se o valor 1. Ex.:

1 0 1 1 0 1	(número binário de 45_{10})
0 1 0 0 1 0	(complemento de 1)
+ 1	(soma-se 1 ao complemento de 1)
0 1 0 0 1 1	(complemento de 2 de 45_{10})

O sistema de complemento de 2 para representação de números com sinal funciona da seguinte forma:

- Se o número for positivo será representado da mesma forma que em sinal-magnitude, ou seja, na forma binária direta com um bit de sinala 0 no MSB.
- Se o número for negativo, a magnitude é representada na forma de complemento de 2 com um bit de sinal 1 no MSB.

Exemplo:

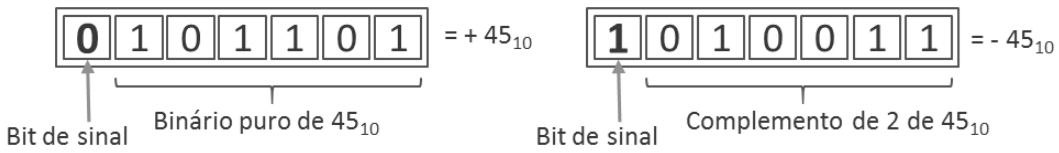


Figura 7.3

Podemos realizar o procedimento de forma mais rápida. Para isso, basta efetuar o complemento de 2 do número positivo com sinal, o que irá gerar o número negativo correspondente. O bit de sinal é gerado no processo.

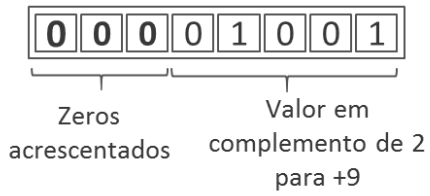
Ex.:

$$\begin{array}{rl} \mathbf{0}\ 1\ 0\ 1\ 1\ 0\ 1 & (+45_{10}) \\ 1\ 0\ 1\ 0\ 0\ 1\ 0 & \text{(complemento de 1)} \\ +\qquad\qquad\ 1 & \text{(soma-se 1)} \\ \hline \mathbf{1}\ 0\ 1\ 0\ 0\ 1\ 1 & (-45_{10}) \end{array}$$

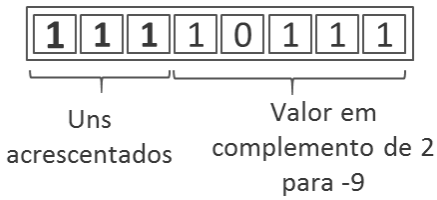
Quando um sistema representa números em complemento de 2, a negação do número é aplicar o complemento de 2 no número. Dessa forma, a negação de um número positivo é o seu negativo correspondente, e vice-e-versa.

Como já mencionado, os elementos de memória são capazes de armazenar um número fixo de bits. Para armazenar um número positivo de 5 bits em um elemento de memória de 8 bits acrescentamos 0s à frente. Se for negativo acrescentamos 1s.

Ex.: Número positivo 01001_2. Com 8 bits, será armazenado desta forma:



Para o número negativo 10111_2, será:



7.4 Faixa de contagem do sistema de complemento de dois

Na Figura 7.4 são apresentados todos os valores de 4 bits que podem ser representados no sistema de complementos de 2.

Valor decimal	Binário (complemento de 2)
+7	**0111**
+6	**0110**
+5	**0101**
+4	**0100**
+3	**0011**
+2	**0010**
+1	**0001**
0	**0000**
-1	**1111**
-2	**1110**
-3	**1101**
-4	**1100**
-5	**1011**
-6	**1010**
-7	**1001**
-8	**1000**

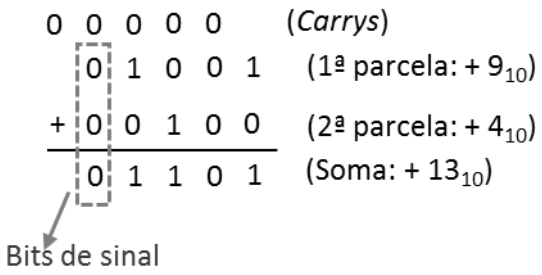
Figura 7.4

Note que o valor 1000_2 (-8_{10}) não tem correspondente positivo (complemento). Sempre se tem um valor negativo a mais que um positivo em uma faixa de valores em complemento de 2.

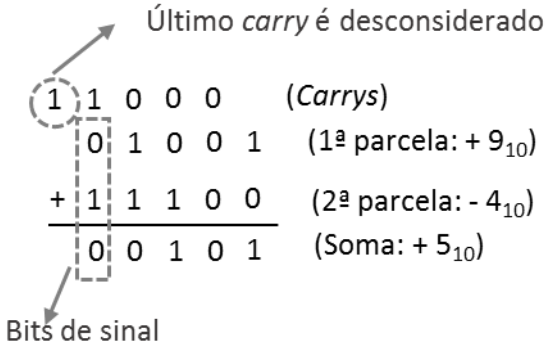
7.5 Adição e subtração no sistema em complemento de dois

Analisaremos 5 casos.

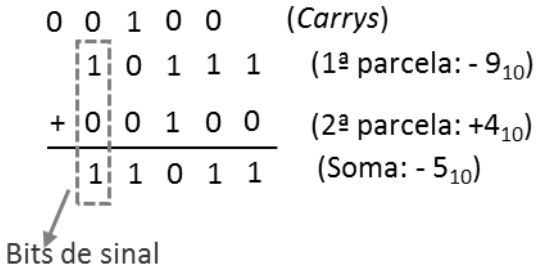
Caso 1: dois números positivos. Exemplo:



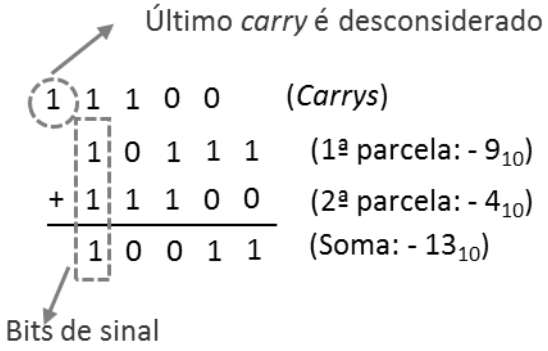
Caso 2: um número positivo e outro negativo menor em magnitude. Exemplo:



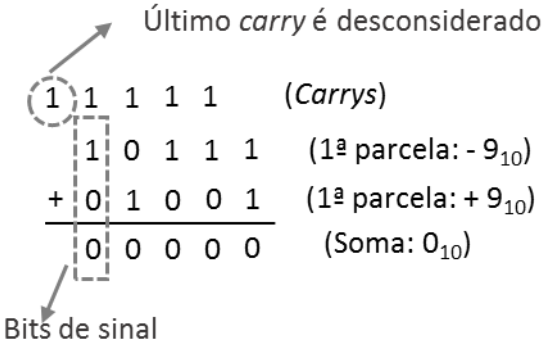
Caso 3: um número positivo e outro negativo maior em magnitude. Exemplo:



Caso 4: dois números negativos. Exemplo:



Caso 5: números iguais e de sinais opostos. Exemplo:

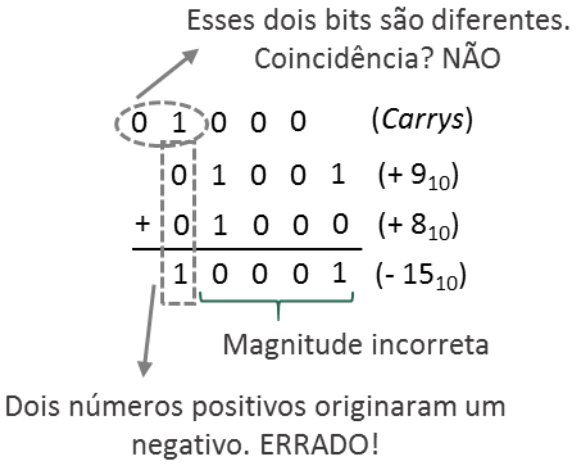


A subtração consiste na soma de um número positivo e outro negativo. Para isso, deve-se aplicar o complemento de 2 no subtraendo e adicionar ao minuendo.

7.6 *Overflow* aritmético

Nos casos 1 à 5 anteriores sobre adição, os números eram constituídos de 5 bits – 1 bit para o sinal e 4 bits para a magnitude. As respostas também. O *carry* da posição do sexto bit era desconsiderado (denominado ***carry* de saída**), isso porque os resultados das somas não excediam a faixa de representação. Porém, vamos analisar quando a faixa de representação excede.

Exemplo: adição de $+9_{10}$ com $+8_{10}$.



A resposta deveria ser $+17_{10}$, mas a magnitude 17 requer mais que 4 bits, e, portanto, ocorreu um ***overflow* aritmético**. Em uma soma você pode verificar se houve *overflow*, quando os dois últimos bits de *carry* forem diferentes,

7.7 Unidade Lógica e Aritmética (ULA)

Uma **unidade lógica e aritmética** - **ULA** (ou do inglês **Arithmetic Logic Unit** – **ALU**) - é um circuito digital capaz de realizar diversas operações lógicas (OR, AND, NOT, etc.) e aritméticas (soma, subtração, multiplicação, divisão, etc.). A quantidade de operações

realizadas por uma ULA depende do circuito. Tem-se diversas ULAs em CIs TTL/CMOS, como também existem ULAs internamente à processadores de computadores.

Um símbolo genérico de ULA é apresentado na Figura 7.5.

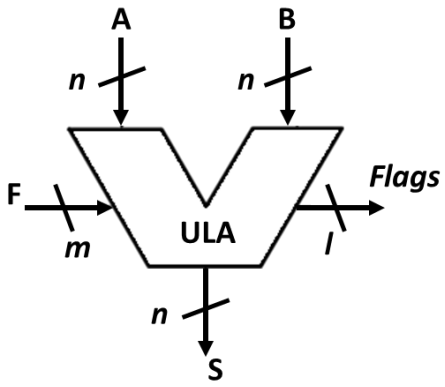


Figura 7.5

Onde A e B são os valores, ou **palavras**, de entrada da ULA, ambos contendo *n* bits (4, 8, 16, etc.). Sobre A e B que será aplicada uma função lógica/aritmética definida pela entrada F, contendo *m* bits. O resultado da operação irá ser gerado na saída S. Os bits denominados ***flags***, tem a função adicional de informação, por exemplo, se ocorreu *overflow*, se o resultado da operação é zero, etc.

7.8 O CI CD4008

O CI CD4008, como apresentado na Figura 7.6, é um somador de 4 bits (**palavras de entrada** de 4 bits).

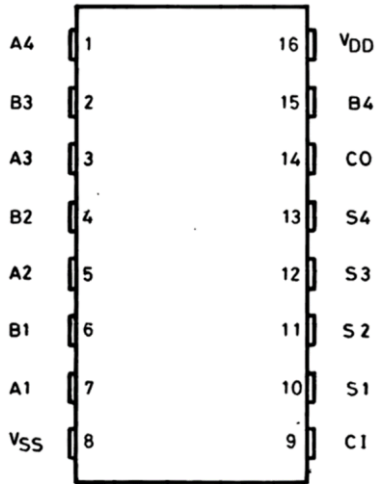


Figura 7.6

- A e B: valores ou **palavras** de entrada de 4 bits cada.
- CI: chamado de ***carry* de entrada**, corresponde ao carry aplicado ao LSB do número. Para somas CN = 0, para subtrações CN = 1.
- CO: último carry gerado pela soma dos MSBs (***carry* de saída**).
- S: **palavra de saída** de 4 bits, com o resultado da operação.

REFERÊNCIAS

[1] IDOETA, I.V. e CAPUANO, F.G. **Elementos de eletrônica digital**. 38.ed. São Paulo: Editora Érica, 2006.

[2] TOCCI, Ronald; WIDMER, Neal. **Sistemas digitais: princípios e aplicações**. 10.ed. São Paulo: Pearson Prentice Hall, 2010.

[3] AZEVEDO JR., J.B. **TTL/CMOS – Teoria e aplicação em circuitos digitais**. 3.ed. São Paulo: Editora Érica, v.1 e 2, 1984.

[4] RIBEIRO, Dágnon, UGOSKI, Paulo e MEDINA, Ricardo. **Apostila de eletrônica digital**. Pelotas: Gráfica IF.

[5] BIGNELL, J.W. e DONOVAN, R.L. **Eletrônica digital**. São Paulo: Makron Books, v.1 e 2, 1995.

[6] SOURCE FOURGE. ***The boolean expression reducer***. Disponivel em: https://sourceforge.net/projects/bexpred/

[7] WOLFRAN ALPHA. ***Boolean Algebra Calculator*** . http://www.wolframalpha.com/widget/widgetPopup.jsp?p=v&id=4c86f3bbcab249f879058d1825887571&title=Boolean%20Algebra%20Calculator&theme=green

[8] FLOYD, Tom. Sistemas Digitais: Fundamentos e Aplicações