CAPÍTULO 17

TÉCNICAS DIGITAIS

SISTEMAS DE NUMERAÇÃO

Os sistemas de numeração foram desenvolvidos na história da humanidade atendendo às crescentes necessidades.

Inicialmente o homem, por conveniência utilizou-se dos dedos como forma de contagem, criando o sistema decimal.

Com o advento do computador, outros sistemas vieram a ser criados, visando maior facilidade de representação interna codificada. Dentre os mais comuns podemos citar os sistemas Binário, Octal e Hexadecimal, que adequam-se às necessidades ou funções internas de diversos equipamentos.

O sistema decimal, porém, nunca foi deixado de lado como forma de representação numérica, convencionada para nós, humanos.

Sistema decimal de numeração

O sistema decimal é um sistema de base 10, no qual existem dez algarismos para representação de uma quantidade: 0, 1, 2, 3, 4, 5, ..., 9.

O menor algarismo de uma determinada base é zero (0) e o maior é igual a base menos 1 (10 – 1 = 9).

No exemplo 1 a seguir temos um número na base 10.

$(583)_{10}$

Podemos decompor este número em potência de dez, já que sua base é 10 e fazendo isso teremos:

$$(5 \times 100) + (8 \times 10) + (3 \times 1) = 583$$

Neste exemplo podemos notar que o algarismo menos significativo (no caso o três) multiplica-se a unidade (1 ou 10^0), o segundo algarismo (o oito) multiplica-se a dezena (10 ou 10^1) e o mais significativo (no caso o cinco) multiplica-se a centena (100 ou 10^2). A soma desses resultados irá representar o número.

Exemplo 2:

$(1592)_{10}$

Decompondo o mesmo teremos:

$$1 \times 10^3 + 5 \times 10^2 + 9 \times 10^1 + 2 \times 10^0 = 1592$$
$$1000 + 500 + 90 + 2 = 1592$$

Exemplo 3:

$(583{,}142)_{10}$

Notamos que no exemplo 3 temos um número com uma parte fracionária. Vejamos então sua decomposição em potência de dez:

$5 \times 10^2 + 8 \times 10^1 + 3 \times 10^0 + 1 \times 10^{-1} + 4 \times 10^{-2} + 2 \times 10^{-3}$ ou

$500 + 80 + 3 + 1/10 + 4/100 + 2/1000$ ou ainda

$500 + 80 + 3 + 0{,}1 + 0{,}04 + 0{,}002 = 583{,}142$

Sistema binário de numeração

No sistema binário a base é 2 (b = 2) e existem apenas dois algarismos para representar uma determinada quantidade: o algarismo 0 (zero) e o algarismo 1 (um).

Para representar a quantidade zero, utilizamos o algarismo 0, para representar a quantidade um, utilizamos o algarismo 1.

No sistema decimal, nós não possuímos o algarismo dez e representamos a quantidade de uma dezena utilizando o algarismo 1 (um) seguido do algarismo 0 (zero). Nesse caso, o algarismo 1 (um) significa que temos um grupo de uma dezena e o algarismo 0 (zero) nenhuma unidade, o que significa dez.

No sistema binário agimos da mesma forma, para representar a quantidade dois, utilizamos o algarismo 1 (um) seguido do algarismo 0 (zero). O algarismo 1 (um) significará que temos um grupo de dois elementos e o 0 (zero) um grupo de nenhuma unidade, representando assim o número dois.

Exemplo:

Seja o número $(1011)_2$ e façamos a sua decomposição em potência só que desta vez a base será dois:

$$1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$$

$$1000 + 000 + 10 + 1 = (1011)_2$$

Sistema octal de numeração

No sistema octal a base é oito e temos oito algarismos para representar qualquer quantidade. Esses algarismos são: 0, 1, 2, 3, ...7.

Para a formação de um número, utilizam-se esses algarismos e toda vez que tivermos uma quantidade igual ao valor da base, soma-se um (1) ao algarismo de valor posicional imediatamente superior como fazemos no sistema decimal.

Notamos também que, em qualquer base o maior algarismo é igual ao valor da base menos um (1) e o número de algarismos é sempre igual ao da base.

Exemplo:

Decompondo o número $(361)_8$ em potência de base oito temos:

$$3 \times 8^2 + 6 \times 8^1 + 1 \times 8^0$$

$$3 \times 100 + 6 \times 10 + 1 \times 1 = (361)_8$$

Podemos escrever que a base elevada a uma determinada potência é igual a um (1 seguido de tantos zeros quantos forem os valores das potências, assim temos:

$2^3 = 1000$ $\quad 2^2 = 100$ $\quad 2^1 = 10$

$10^3 = 1000$ $\quad 10^2 = 100$ $\quad 10^1 = 10$

$8^3 = 1000$ $\quad 8^2 = 100$ $\quad 8^1 = 10$

No sistema decimal, o número 100 aparece após o número 99 na ordem crescente.

No sistema binário, o número 100 aparece após o número 11 na ordem crescente.

No sistema octal, o número 100 aparece após o número 77 na ordem crescente.

Sistema hexadecimal de numeração

No sistema hexadecimal de numeração, a base é dezesseis e dispomos de dezesseis algarismos para representação de uma determinada quantidade de coisas. Como existem apenas dez algarismos numéricos utilizamos também algarismos alfanuméricos. Portanto temos os seguintes algarismos:

0, 1, 2, 3,.....9, A, B, C, D, E e F.

Hex	Dec
A	10
B	11
C	12
D	13
E	14
F	15

Exemplo:

Tomemos o número $(2C0A)_{16}$ e façamos sua decomposição.

$$2 \times 16^3 + C \times 16^2 + 0 \times 16^1 + A \times 16^0 \text{ ou}$$

$$2 \times 4096 + 12 \times 256 + 0 \times 16 + 10 \times 1 =$$

$$= 8192 + 3072 + 0 + 10 =$$

$$= (11274)_{10}$$

Complemento de um número

O complemento de um número é o que falta a este número para atingir o valor da base.

Exemplo:

Complemento de $(7)_{10}$ $\quad 10 - 7 = 3$

No sistema binário para chegar-se ao complemento, obtem-se primeiramente o falso complemento.

$$\underline{(1011)_2}$$
$$0100 \longrightarrow \text{Complemento falso}$$

Complemento verdadeiro consiste em somar-se 1 (um) ao complemento falso.

$$\begin{array}{r} 0100 \\ +\ 1 \\ \hline 0101 \end{array}$$

Conversão de bases

<u>Conversão para base decimal</u> – Para convertermos um número representado em qualquer sistema numérico, para o sistema decimal usamos a notação posicional e resolvemos a expressão como na base decimal.

Seja o número 1101 no sistema binário. A notação posicional seria:

$1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 =$

$1 \times 8 + 1 \times 4 + 0 \times 2 + 1 \times 1 =$

$8 + 4 + 0 + 1 = (13)_{10}$

Portanto $(1101)_2 = (13)_{10}$

Como segundo exemplo o número 107 do sistema octal. A notação posicional seria:

$1 \times 8^2 + 0 \times 8^1 + 7 \times 8^0 =$

$1 \times 64 + 0 \times 8 + 7 \times 1 =$

$64 + 0 + 7 = (71)_{10}$

Portanto $(107)_8 = (71)_{10}$

<u>Conversão do sistema decimal para outras bases</u> – Para conversão da base 10 para outras bases, o método consiste em divisões sucessivas pela base desejada, até que o quociente seja nulo. Os restos das divisões indicarão o resultado da conversão, sendo o primeiro resto equivalente ao dígito menos significativo e o último ao mais significativo.

Exemplo 1

Façamos a conversão do número $(934)_{10}$ para base hexadecimal.

→ (10 A)

934 | 16
1º resto 6 | 58 | 16
2º resto 10 | 3 | 16
3º resto 3 | 0

Portanto $(934)_{10} = (3A6)_{16}$

Exemplo 2

Conversão do número $(76)_{10}$ para a base 8.

76 | 8
4 | 9 | 8
1 | 1 | 8
1 | 0

Portanto $(760_{10} = (114)_8$

Exemplo 3

Conversão do número $(12)_{10}$ para a base 2.

12 | 2
0 | 6 | 2
0 | 3 | 2
1 | 1 | 2
1 | 0

Portanto $(12)_{10} = (1100)_2$

Contagem nas diversas bases

Na tabela de contagem nos sistemas de base decimal, binária, octal e hexadecimal observa-se que um número expresso num sistema de base menor exige maior quantidade de algarismos do que outro, de base maior, para representar a mesma quantidade.

DECI-MAL	BINARIA	OCTAL	HEXA-DEC.
0	0	0	0
1	1	1	1
2	$10 = 2^1$	2	2
3	11	3	3
4	$100 = 2^2$	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	$1000 = 2^3$	$10 = 8^1$	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	$10000=2^4$	20	$10=16^1$
-	-	-	-
31	11111	37	1F
32	$100000=2^5$	40	20
-	-	-	-
63	111111	77	3F
64	$1000000=2^6$	$100=8^2$	40
-	-	-	-
99	1100011	143	63
100	1100100	144	64
-	-	-	-
127	1111111	177	7F
128	$10000000=2^7$	200	80
-	-	-	-
255	11111111	377	FF
256	$100000000=2^8$	400	$100=16^2$
-	-	-	-
-	-	$1000=8^3$	-

Códigos

Ao códigos são formas de representação de caracteres alfanuméricos.

São vários os códigos existentes havendo porém vantagens de um ou outro, de acordo com a aplicação ou funções internas do equipamento.

Código BCD 8421 – A sigla BCD representa as iniciais de "Bynary Coded Decimal", que significa uma codificação no sistema decimal em binário. Os termos seguintes (8421) significam os pesos de cada coluna, isto é, $8 = 2^3$, $4 = 2^2$, $2 = 2^1$ e $1 = 2^0$. O valor corresponderá à soma dos pesos onde na coluna houver o "bit" um (1).

DECIMAL	BCD 8	4	2	1
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

O número de "bits" de um código é o número de dígitos binários que este possui.

O código BCD 8421 é um código de 4 "bits".

Código excesso 3 – Consiste na transformação do número decimal, no binário correspondente, somando-se a ele três unidades.

Exemplo:

$(0)_{10} = (0000)_2$

Somando-se três unidades, teremos 0011

DECIMAL	EXCESSO 3
0	0 0 1 1
1	0 1 0 0
2	0 1 0 1
3	0 1 1 0
4	0 1 1 1
5	1 0 0 0
6	1 0 0 1
7	1 0 1 0
8	1 0 1 1
9	1 1 0 0

O código Excesso 3 é utilizado em circuitos aritméticos.

Código Johnson – Baseia-se no deslocamento de "bits" e é utilizado na construção do Contador Johnsos.

DECIMAL	JOHNSON
0	0 0 0 0 0
1	0 0 0 0 1
2	0 0 0 1 1
3	0 0 1 1 1
4	0 1 1 1 1
5	1 1 1 1 1
6	1 1 1 1 0
7	1 1 1 0 0
8	1 1 0 0 0
9	1 0 0 0 0

Código Gray ou sistema de numeração refletido – Sua principal característica é que, em contagens sucessivas, apenas um "bit" varia.

A codificação Gray é mostrada na tabela a seguir, onde os campos em destaque representam um "espelho" a ser refletido para a contagem seguinte, acrescentando-se um "bit" 1 (um) imediatamente à esquerda.

DECIM.	BINÁRIO	GRAY
0	0 0 0 0 0	0 0 0 0 0
1	0 0 0 0 1	0 0 0 0 1
2	0 0 0 1 0	0 0 0 1 1
3	0 0 0 1 1	0 0 0 1 0
4	0 0 1 0 0	0 0 1 1 0
5	0 0 1 0 1	0 0 1 1 1
6	0 0 1 1 0	0 0 1 0 1
7	0 0 1 1 1	0 0 1 0 0
8	0 1 0 0 0	0 1 1 0 0
9	0 1 0 0 1	0 1 1 0 1
10	0 1 0 1 0	0 1 1 1 1
11	0 1 0 1 1	0 1 1 1 0
12	0 1 1 0 0	0 1 0 1 0
13	0 1 1 0 1	0 1 0 1 1
14	0 1 1 1 0	0 1 0 0 1
15	0 1 1 1 1	0 1 0 0 0
16	1 0 0 0 0	1 1 0 0 0

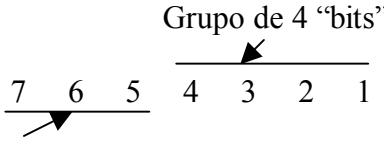
Este tipo de codificação garante que, com a variação de apenas um "bit" de uma contagem para outra, reduzam-se as conseqüências negativas geradas pela mudança de estado simultânea de registradores.

Código ASCII – O código ASCII é um tipo de codificação BCD, largamente utilizado em computadores digitais e em equipamentos de comunicação de dados. A sigla ASCII é formada pelas iniciais de American Standard Code for Information Interchange (Código Padrão Americano para Intercâmbio de Informações).

Consiste de um código binário de sete "bits" para transferir informações entre computadores e seus periféricos e em comunicação de dados a distância.

Com um total de sete "bits", podemos representar $2^7 = 128$ estados diferentes ou caracteres, que são usados para representar os números decimais de 0 a 9, letras do alfabeto e alguns caracteres especiais de controle.

É formado por dois grupos de "bits", sendo um de 4 "bits" e outro de 3 "bits".



Formato do caráter no Código ASCII

CARACTER	ASCII 7 6 5 4 3 2 1
0	0 1 1 0 0 0 0
1	0 1 1 0 0 0 1
2	0 1 1 0 0 1 0
-	
9	0 1 1 1 0 0 1
-	
A	1 0 0 0 0 0 1
B	1 0 0 0 0 1 0
-	
Z	1 0 1 1 0 1 0
-	
a	1 1 0 0 0 0 1
b	1 1 0 0 0 1 0
-	
z	1 1 1 1 0 1 0

Exemplos de representações no código ASCII

OPERAÇÕES BINÁRIAS

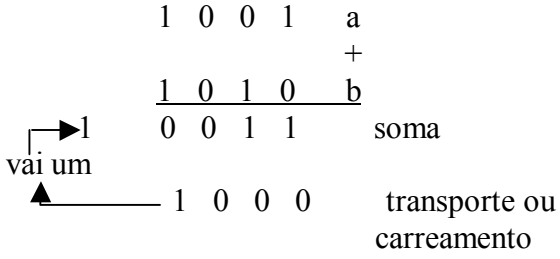
A eletrônica em seus primórdios, tinha sus cálculos baseados em álgebra convencional, através de sistemas analógicos ou lineares.

Com o advento de máquinas mais sofisticadas, processadores eletrônicos, sistemas de comunicação e controle digitais, os problemas vieram a ser resolvidos baseados em álgebra especial, não linear, mas binária, isto é, baseada em dois valores. É a álgebra Booleana.

Aritmética binária

As regras utilizadas em operações binárias no sistema decimal, são também seguidas nas mesmas operações em outros sistema de numeração. Neste capitulo trataremos de algumas técnicas que tornam mais simples a efetuação destas operações.

Adição no sistema binário – Para efetuarmos a adição no sistema binário, devemos agir como uma adição no sistema decimal, lembrando que no sistema binário temos apenas dois algarismos.

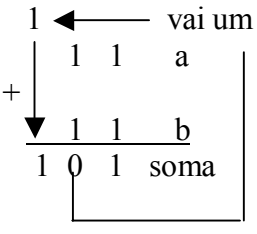


A tabela mostra a operação soma e o transporte em separado. O símbolo + é o operador soma.

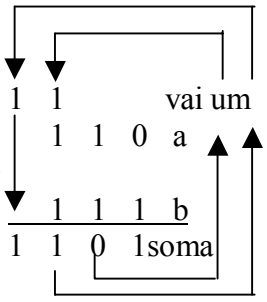
Como 1 + 1 = 10 no sistema binário, o resultado é 0 (zero) e o transporte para a coluna imediatamente à esquerda é 1 (um). Esse transporte é idêntico ao do sistema decimal, pois quando tivermos uma soma igual ou maior que a base, haverá um "vai um" que será somado ao dígito de valor posicional imediatamente superior.

Exemplo 1:

a + b
$(11)_2 + (10)_2 = (101)_2$
3 + 2 = 5



Exemplo 2:



a + b
$(110)_2 + (111)_2 = (1101)_2$ +
6 + 7 = 13

Exemplo 3

```
                  1 1
  a   +   b         1 1 0 0 1  a
11001 + 1011 = 100100  +
 25   +  11 =   36      1 0 1 1  b
            soma 1 0 0 1 0 0
```

Subtração no sistema binário- O método de resolução é análogo a uma subtração no sistema decimal:

```
  0 0 0 1   empréstimo
    ↗
  0 1 1 0   a
- 0 1 0 1   b
  0 0 0 1   diferença
```

Exemplo 1

```
  a   -   b          1 1 1   a
111 - 100 = 011    - 1 0 0   b
  7 -   4 =   3      0 1 1 diferença
```

Exemplo 2

```
                      1 empréstimo
  a   -   b         1 0 0 0   a
1000 - 111 = 1        1 1 1   b
  8  -  7  = 1     0 0 0 1 diferença
```

Nos exemplos acima foram utilizados números tais que a > b. Consideremos agora um caso com a < b.

```
  a   -    b
10110 - 11001 = -00011
 22   -   25 =   -3

    1 0 1 1 0   a
-1 -1    -1     empréstimo
    1 1 0 0 1   b
    1 1 1 0 1   resultado parcial
```

Neste exemplo, seguindo-se as regras anteriores, observa-se que houve um empréstimo que ficou devedor. Nesta situação efetua-se a operação " complemento" , que consiste em inverter-se os bits "0" por "1" e vice-versa, somando-se "1" em seguida.

```
  1 1 1 0 1   resultado parcial

  0 0 0 1 0   resultado invertido
+         1     (complemento)
  0 0 0 1 1
  0 0 0 1 1   resultado final
```

Multiplicação no sistema binário- Procede-se como em multiplicações no sistema decimal, tendo-se como regra básica:

```
0 x 0 = 0
0 x 1 = 0
1 x 0 = 0
1 x 1 = 1
```

Exemplo 1

```
  a   x b          1 0 0 0   a
1000 x 1 = 1000  x       1   b
  8  x 1 =   8     1 0 0 0   produto
```

Exemplo 2

```
                     1 0 1 0 1   a
  a    x  b         x       1 0   b
10101 x 10 = 101010    0 0 0 0 0
 21   x  2 =  42     1 0 1 0 1
          produto 1 0 1 0 1 0
```

Divisão no sistema binário- Procede-se como em divisões no sistema decimal.

Exemplo:

```
  a    :  b
111100 : 1100 = 101
  60   :  12 =  5

  1 1 1 1 0 0 | 1 1 0 0
- 1 1 0 1     | 1 0 1
  0 0 1 1 0   |
-   0 0 0 0   |
    0 1 1 0 0 |
  -   1 1 0 0 |
      0 0 0 0   resto
```

ÁLGEBRA DE BOOLE

Em meados do século passado G. Boole desenvolveu um sistema matemático de análise lógica. Esse sistema é conhecido como Álgebra de Boole.

A álgebra booleana é baseada em apenas dois estados. Estes estados poderiam, por exemplo, ser representados por tensão alta e tensão baixa ou tensão positiva e tensão negativa.

Assim como na álgebra linear, encontramos vários tipos de funções, como veremos a seguir.

Simplificação de funções

Função "E" ou "AND" – É aquela cujo resultado equivale à multiplicação de duas ou mais variáveis.

S = A . B (onde se lê A e B)

Para melhor entendimento veja a figura 17-1.

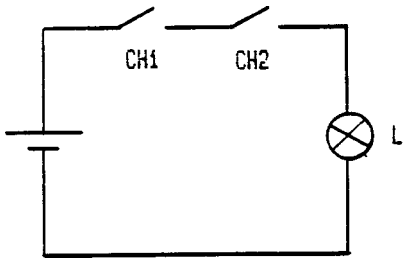


Figura 17-1 Circuito da função E ou AND

Convenções:

Chave aberta = 0
Chave fechada = 1
Lâmpada apagada = 0
Lâmpada acesa = 1

Situações possíveis:

Ch 1 aberta e Ch 2 aberta =lâmpada apagada
0 * 0 = 0

Ch 1 aberta e Ch 2 fechada =lâmpada apagada
0 * 1 = 0

Ch 1 fechada e Ch 2 aberta =lâmpada apagada
1 * 0 = 0

Ch 1 fechada e Ch 2 fechada=lâmpada acesa
1 * 1 = 1

Concluímos que a lâmpada só acenderá quando a Ch 1 e a Ch 2 estiverem fechada, correspondendo a equação A . B = S

Tabela Verdade da função E ou AND – É um mapa onde colocamos todas as situações possíveis, com os respectivos resultados.

A	B	S = A . B
0	0	0
0	1	0
1	0	0
1	1	1

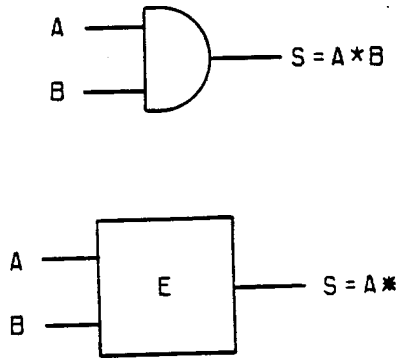


Figura 17-2 Simbologia da função E ou AND

O número de situações possíveis constante na tabela verdade é igual a 2^N, onde N é o número de variáveis de entrada.

Uma porta "E" com duas entradas tem $2^N = 2^2 = 4$ situações possíveis.

Podemos encontrar portas lógicas com três ou mais entradas como mostrado na figura 17-3.

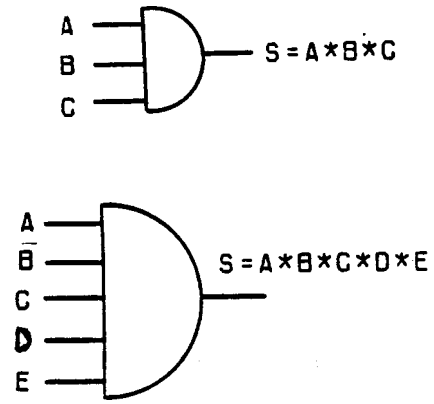


Figura 17-3 Portas E ou AND de três e de cinco entradas.

Função OU ou OR – É aquela que assume o valor um (1) na saída, quando uma ou mais variáveis na entrada forem iguais a um (1), e assume o valor zero (0) se, e somente se, todas as entradas forem iguais a zero (0).

S = A + B (S igual a A ou B)

Para melhor compreensão veja a figura 17-4

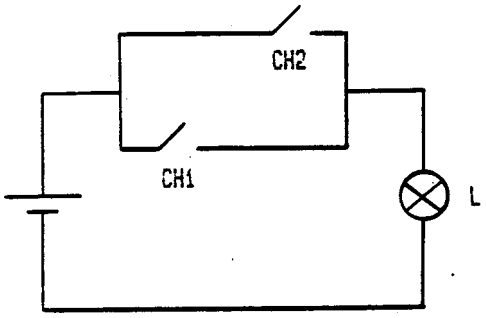


Figura 17-4 Circuito da função OU ou OR

Convenções:

Chave aberta = 0
Chave fechada = 1
Lâmpada apagada = 0
Lâmpada acesa = 1

Situações possíveis:

Ch 1 aberta e Ch 2 aberta =lâmpada apagada
0 + 0 = 0

Ch 1 aberta e Ch 2 fechada =lâmpada acesa
0 + 1 = 1

Ch 1 fechada e Ch 2 aberta =lâmpada acesa
1 + 0 = 1

Ch 1 fechada e Ch 2 fechada =lâmpada acesa
1 + 1 = 1

Concluímos que a lâmpada acenderá quando pelo menos uma das chaves estiver ligada, correspondendo à equação A + B = S.

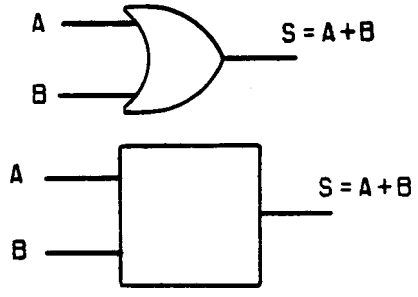


Figura 17-5 Simbologia da função OU ou OR

Tabela Verdade da função OU ou OR

A	B	S = A + B
0	0	0
0	1	1
1	0	1
1	1	1

Portas OR também podem ser encontradas com 3 ou mais entradas.

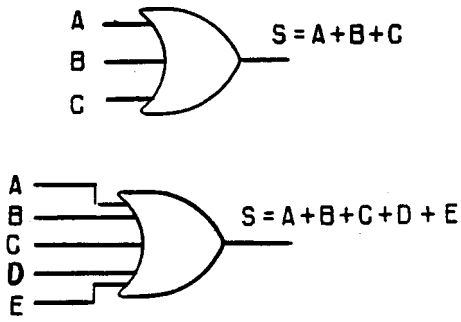


Figura 17-6 Exemplos de portas OR

Função NOT ou NÃO – A função NÃO, complemento ou inversão, é aquela que inverte o estado da variável, isto é, "0" inverte para"1" e "1" inverte para "0". Veja a figura 17-7.

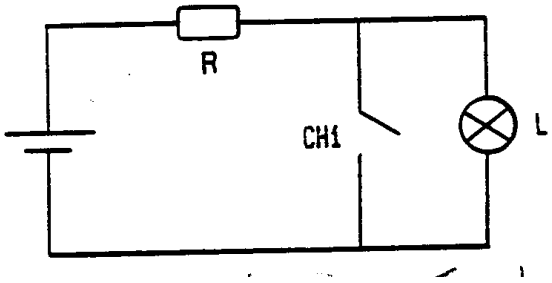


Figura 17-7 Circuito da função NOT ou NÃO

Convenções:

Chave aberta = 0
Chave fechada = 1
Lâmpada apagada = 0
Lâmpada acesa = 1

Situações possíveis:

Chave 1 aberta = lâmpada acesa
0 = 1

Chave 1 fechada = Lâmpada apagada
1 = 0

Tabela verdade da função NOT ou NÃO

A	$\overline{A}$
0	1
1	0

Onde $\overline{A}$ representa o inverso de A

Função NÃO E ou NAND – É uma combinação das funções "E" e "NÃO", que é representada da seguinte forma:

$S = \overline{A * B}$ (S igual a A e B barrados, ou A e B "not").

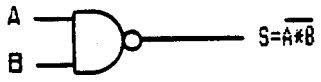


Figura 17-8 Simbologia NAND

Tabela Verdade da função NAND

A	B	$\overline{A*B}$ (S)
0	0	1
0	1	1
1	0	1
1	1	0

Função "NÃO OU" ou NOR – É a combinação das funções OU e NÃO, que é representada da seguinte forma:

$S = \overline{A + B}$ (S igual a A ou B barrado, ou A ou B "not").

Tabela Verdade NÃO OU ou NOR

A	B	$S = \overline{A + B}$
0	0	1
0	1	0
1	0	0
1	1	0

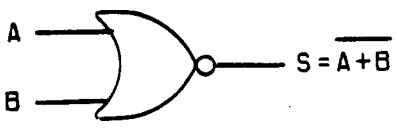


Figura 17-9 Simbologia NOR

Funções XOR ou XNOR – As portas NAND e NOR são ditas portas universais, porque vários circuitos podem ser derivados, utilizando apenas estes tipos de portas.

Podemos criar diversas funções combinando os vários tipos de portas lógicas, dentre elas as denominadas XOR e XNOR.

Tabela Verdade e simbologia

a) XOR ou "OU EXCLUSIVO" – Nesta função teremos "1" na saída, quando as entradas forem desiguais.

A	B	S = A + B
0	0	0
0	1	1
1	0	1
1	1	0

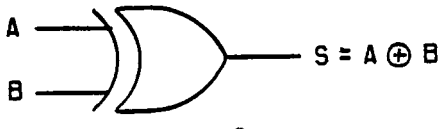


Figura 17-10 Simbologia XOR

b) XNOR ou "NOR EXCLUSIVO"- Nesta função teremos "1" na saída, quando as entradas forem iguais.

A	B	$S = \overline{A + B}$
0	0	1
0	1	0
1	0	0
1	1	1

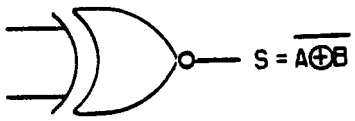


Figura 17-11 Simbologia XNOR

As portas XOR e XNOR são denominadas portas COMPARADORAS.

A porta XOR é denominada comparadora de desigualdade e a porta XNOR comparadora de igualdade.

Formas canônicas

As tabelas verdade de circuitos padrão nem sempre conseguem representar todas as

funções lógicas. Há circuitos cujas funções diferem do padrão. Estes circuitos poderão ser representados através de FORMAS CANÔNICAS.

Forma canônica disjuntiva– É a forma canônica mais utilizada. Para cada uma das entradas, atribui-se o valor "0" ou "1", estabelecendo-se uma expressão representativa da função f = 1.

ENTRADAS			SAÍDA	
A	B	C	f	
0	1	0	1	$\overline{A}*B*\overline{C}$
0	1	1	1	$\overline{A}*B*C$
1	0	0	1	$A*\overline{B}*\overline{C}$
1	1	0	1	$A*B*\overline{C}$

$f=\overline{A}\overline{B}\overline{C}+\overline{A}BC+A\overline{B}\overline{C}+AB\overline{C}$ Forma Canônica

Circuitos geradores de produtos canônicos- São circuitos que geram as formas canônicas básicas, onde são estabelecidas e combinadas as entradas para todas as variações.

Se quisermos gerar os produtos canônicos possíveis com "n" variáveis, necessitaremos de 2^n portas de "n" entradas.

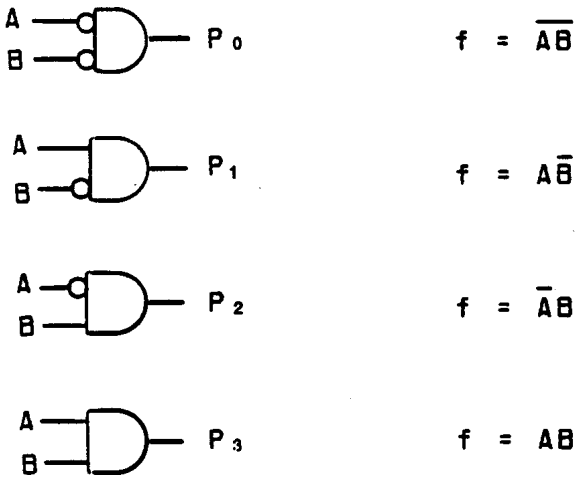


Figura 17-12 Exemplos com duas variáveis

CIRCUITOS DE COMUTAÇÃO

Os circuitos lógicos de um equipamento precisam ser compatíveis às necessidades do projeto. Na execução de funções lógicas, as entradas e saídas são variáveis, requisitando padrões de comutação.

Dentre as características dos circuitos de comutação, podemos citar o nível lógico, o tempo de propagação, a potência dissipada, a imunidade à ruídos e o "fan-out".

Níveis lógicos

Os níveis lógicos são as tensões designadas como estado "1" e estado "0" binários, para um certo tipo de circuito digital.

Os valores nominais para os dois níveis são bem determinados mas, na prática, os valores obtidos podem variar, devido à tolerância dos componentes internos do circuito integrado, variações da fonte de alimentação, temperatura e outros fatores. Geralmente os fabricantes fornecem os valores máximos e mínimos admitidos para cada um dos níveis lógicos.

É muito importante conhecer os níveis lógicos de um determinado tipo de integrado pois, deste modo, ao trabalhar com equipamentos digitais, será fácil identificar os estados lógicos das entradas e saídas.

Tempo de propagação

O tempo de propagação (Propagation Delay) é a medida do tempo de operação de um circuito lógico. A velocidade de operação é uma das características mais importantes e, para a maior parte das aplicações digitais, uma alta velocidade de operação, ou seja, um baixo tempo de propagação é benéfico.

O tempo de propagação exprime o espaço de tempo necessário para que a saída de um circuito digital responda a uma mudança de nível de entrada; é composto pelo acúmulo de tempos de transição e retardo associados a qualquer circuito lógico.

Quando a tensão de entrada de um circuito digital muda de "0" para "1", ou vice-versa, a saída deste circuito responderá após certo período de tempo finito.

A figura 17-13 dá um exemplo de tempo de propagação; temos aí representada a entrada de um circuito digital e, logo abaixo, a saída correspondente.

Veja que a transição de "0" para "1" na entrada ocasiona uma transição de "1" para "0" na saída e que a transição de saída ocorre um certo tempo após a transição de entrada. Isto é que chamamos de tempo de propagação.

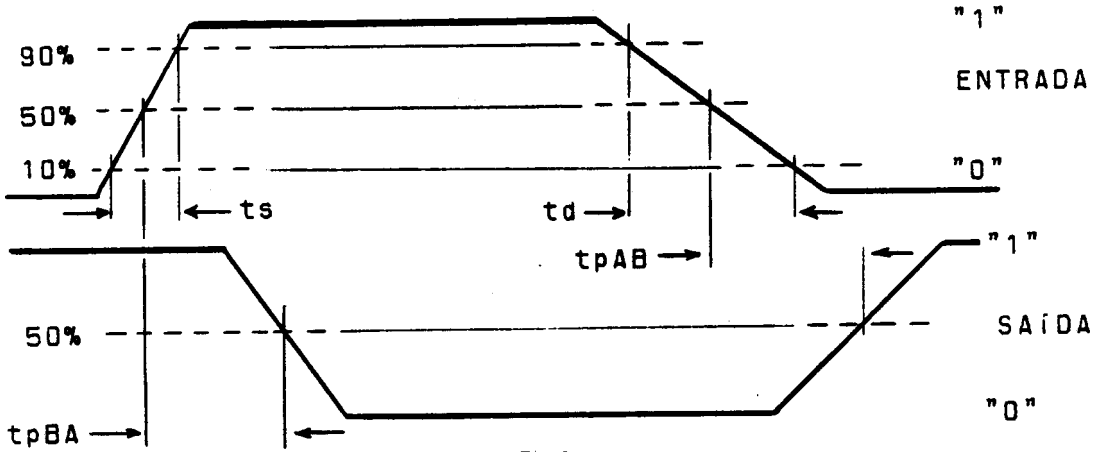


Figura 17-13 Tempo de propagação

O tempo de propagação (tp) é medido geralmente entre os pontos de 50% de amplitude, da transição inicial da entrada para a transição inicial da saída ou da transição final da entrada para a transição final da saída.

Observe ainda que existem dois tipos de tempo de propagação: um deles ocorre quando a entrada passa do nível baixo para o nível alto (tpBA), e o outro quando a entrada passa de alto para baixo (tpAB). Os dois tipos de tempos de propagação são geralmente diferentes, devido às características dos circuitos lógicos.

Os tempos de subida e descida dos pulsos de entrada e saída também são importantes. Define-se tempo de subida (ts), como o período de tempo tomado pelo pulso para subir de 10% a 90% de sua amplitude máxima. O tempo de descida (td), é o necessário para o pulso descer de 90% a 10% dessa mesma amplitude.

Para a maioria dos circuitos integrados digitais, os tempos de subida e descida são bastante reduzidos. Podem ser conseguidos tempos de transição de 1 nanosegundo. Alguns tipos de circuitos digitais modernos apresentam tempos de propagação que chegam a apenas algumas dezenas de nanosegundos. Os tempos de transição são normalmente menores que os tempos de propagação.

Os tempos de propagação podem variar consideravelmente devido a tolerâncias de fabricação, fiação, etc. e são cumulativos. Quando portas e outros circuitos lógicos combinacionais são ligados uns aos outros, os tempos de propagação se somam.

Se existe mais de um nível de lógica, isto é, mais de uma estrutura, o tempo de propagação total, de entrada e saída, é a soma dos tempos de propagação de cada um dos níveis.

Potência dissipada

É a potência consumida por um circuito lógico operando em um ciclo de carga de 50%, isto é, tempos iguais nos estados “0” e “1”.

A potência total dissipada por um circuito é uma consideração importante no projeto de um equipamento digital, pois uma elevada dissipação em potência, significa um grande consumo de energia elétrica.

Além disso, a potência total dissipada irá determinar o tamanho e o custo da fonte de alimentação.

O calor liberado pelos circuitos, também relacionado à potência dissipada pelos mesmos, às vezes torna necessário o seu resfriamento ou o uso de aparelhos de ar condicionado, para garantir o bom funcionamento do equipamento.

A potência dissipada por uma porta pode variar da ordem de alguns microwatts até 100 miliwatts.

Compromisso velocidade-potência

As duas características descritas, velocidade potência dissipada, são diretamente interdependentes em todos os tipos de circuitos lógicos digitais. A relação entre elas é tal que a velocidade se apresenta proporcional à potência dissipada, ou seja, tanto mais rápida a comutação de um circuito lógico, maior será a potência dissipada.

Os circuitos lógicos de alta velocidade empregam transistores bipolares não saturados que, associados a resistências internas de baixos

valores, produzem um alto consumo de potência.

Os circuitos integrados do tipo MOS (Metal-Oxide-Semiconductor), consomem um mínimo de potência devido as altas impedâncias inerentes a esses componentes. No entanto, refletem em velocidades de comutação muito baixas, limitando sua operação a freqüências baixas. Pelo seu consumo bastante reduzido, adequam-se perfeitamente aos equipamentos portáteis operados a bateria, onde a alta velocidade não for necessária.

Imunidade a ruídos

A imunidade a ruídos é uma medida da característica de baixa ou não interferência de sinais externos indesejáveis. Considera-se ruído qualquer sinal estranho, gerado externamente ou pelo próprio equipamento, e que é acrescentado ou superposto aos sinais padrão do sistema.

Esse ruído pode ser um nível de tensão variando lentamente, picos de tensão, ou sinais de alta frequência e pequena duração. O ruído pode provocar uma comutação no circuito lógico, para um estado indesejável num momento impróprio.

A imunidade da maioria dos circuitos lógicos é de aproximadamente 10% a 50% do valor da tensão de alimentação. Isto significa que um pico será rejeitado, caso sua amplitude seja inferior a 10% ou 50% da tensão de alimentação.

A imunidade a ruídos é uma consideração de grande importância, porque a maioria dos sistemas digitais gera uma quantidade considerável de ruído em comutações de alta velocidade. Além disso, muitos equipamentos digitais são utilizados em ambientes industriais de ruído intenso, onde transientes provenientes das linhas de força e de outros equipamentos elétricos podem causar falsas comutações nos circuitos lógicos.

"Fan-out"

"Fan-out" é uma característica que indica o quanto de carga pode ser ligado à saída de um circuito digital. É geralmente expresso em termos de número de cargas padrão que a saída de uma porta lógica aceita, sem afetar o nível lógico nominal, velocidade, temperatura ou outras características.

Uma porta lógica pode, por exemplo, apresentar um "Fan-out" igual a 10, o que indica que até dez entradas de portas poderiam ser ligadas à saída deste circuito lógico, sem afetar a sua operação.

FAMÍLIAS DE CIRCUITOS LÓGICOS

Como podem ser notados, os circuitos lógicos possuem características que deverão ser observadas durante o projeto, para que o mesmo utilize os componentes adequados à aplicação do equipamento. De acordo com estas características, os circuitos lógicos são agrupados em famílias.

Entende-se por famílias de circuitos lógicos, os tipos de estruturas internas que permitem a confecção dos blocos lógicos em circuitos integrados.

Dentre as famílias podemos destacar:

- RTL (Resistor-Transistor Logic).
- DTL (Diode-Transistor Logic).
- HTL (High Threshold Logic).
- TTL (Transistor-Transistor Logic).
- ECL (Emitter-Coupled Logic).
- C-MOS (Complementary MOS).

Tecnologia MOS

A família MOS (Metal Oxide Semiconductor) compõe-se de circuitos formados por MOSFETS, que são transistores de efeito de campo construídos a partir da tecnologia MOS, apresentando como características o baixo consumo e uma alta capacidade de integração, isto é, a colocação de uma grande quantidade de componentes lógicos num mesmo encapsulamento.

Comparação entre famílias

<u>Família RTL (Resistor-Transistor Logic)</u>

Utiliza transistores e resistores, sendo das primeiras famílias utilizadas, formando portas NOR como principal bloco lógico.

Suas principais características são:

- Possui boa imunidade a ruídos
- Tempo de propagação da ordem de 12 ns
- Potência dissipada por bloco lógico, da ordem de 10 mw.
- Alimentação 3V ± 10%

Família DTL (Diode-Transistor Logic)

Utiliza diodos e transistores, sendo um desenvolvimento da lógica de diodos, permitindo a formação de blocos "E", "OU", "NAND" e "NOR".

Suas principais características são:

- Imunidade a ruídos da ordem de 0,8V.
- Tempo de propagação da ordem de 30ns.
- Potência dissipada da ordem de 10 mw por bloco lógico.
- Alimentação 5V ± 10%.

Família HTL (High Threshold Logic)

Utiliza diodos e transistores como a DTL, acrescentando um diodo Zener, para aumento do nível de entrada, estabelecendo alta imunidade à ruídos.

Suas principais características são:

- Alta imunidade a ruidos.
- Alto tempo de propagação.
- Alta potência dissipada, da ordem de 60 mw.

Família TTL (Transistor-Transistor Logic)

É oriunda da família DTL, porém utilizando transistores multiemissores, que permitem a eliminação dos diodos e resistores de entrada, trazendo maior velocidade e menor custo, tornando-a das mais difundidas.

Suas principais características são:

- Boa imunidade a ruídos
- Tempo de propagação da ordem de 10 ns.
- Potência dissipada da ordem de 20 mw por bloco lógico.
- Identificação Comercial – série 74 – Faixa de temperatura de 0° a 75° C.

Família ECL (Emitter Coupled Logic)

Utiliza nos circuitos, acoplamento pelo emissor dos transistores, o que os faz operar em regime de não saturação, permitindo a mais alta velocidade de comutação dentre as famílias.

Suas principais características são:

- Boa imunidade a ruídos.
- Muito baixo tempo de propagação, da ordem de 3 ns.
- Potência dissipada da ordem de 25 mw por bloco.
- Alimentação -5,2 V ± 20%.

Família C-MOS (Complementary MOS)

É uma variação da família MOS, consistindo basicamente de pares de canais MOS complementares. Esta técnica tem como vantagem em relação ao MOS convencional, uma maior velocidade de comutação, da ordem de 80 ns, contra 300 ns.

Suas principais características são:

- Baixa dissipação de potência, da ordem de 10μw.
- Alto índice de integração.
- Alta imunidade a ruídos
- Ainda elevado tempo de propagação, da ordem de 60 a 70 ns.

- Larga faixa de alimentação de 3 a 18 V.

Métodos de fabricação

Existem três formas básicas de se fabricar circuitos integrados. O método mais difundido é o chamado monolítico; os outros são o de película fina, o de película espessa e o híbrido.

Método Monolítico – O circuito integrado monolítico é construído inteiramente de um único pedaço de silício semicondutor, chamado pastilha ou "chip". Materiais semicondutores são difundidos sobre esta base, dando origem a diodos, transistores e resistores. Como resultado, o circuito inteiro, com todos os componentes e interligações, forma-se sobre uma base única, dando origem ao termo "monolítico".

Os circuitos integrados monolíticos digitais se subdividem em dois tipos básicos: os Bipolares e os do tipo MOS, diferindo fundamentalmente no tipo de transistor utilizado.

Os circuitos MOS, são mais fáceis de obter e ocupam menos espaço, desta forma é possível incluir muito mais circuitos num "chip" apresentando uma maior densidade de componentes e custo menor.

Método de película fina ou espessa – Neste método, os circuitos são obtidos depositando-se os materiais sobre uma base não condutora, como a cerâmica, formando resistores, capacitores e indutores. Normalmente os

dispositivos semicondutores não são obtidos por este processo.

Método Híbrido – O circuito integrado híbrido é formado pela combinação de circuitos monolíticos e circuitos de película. Os híbridos oferecem uma grande variedade de combinações entre circuitos integrados e componentes, resultando em várias funções que não poderiam ser obtidas com circuitos integrados específicos.

Classificação dos circuitos integrados digitais

Os circuitos integrados digitais podem ser classificados basicamente em três grupos:

SSI – Small Scale Integration (Integração em Pequena Escala);

MSI – Médium Scale Integration (Integração em Média Escala);

LSI – Large Scale Integration (Integração em Grande Escala).

Os circuitos SSI representam a forma mais básica e simples dos circuitos integrados: são amplificadores ou portas, que realizam uma função elementar, devendo ser interligados externamente, caso queiramos formar circuitos funcionais completos..

Os circuitos MSI são mais complexos, formados por várias portas interligadas, compondo circuitos funcionais completos, a maioria contendo doze ou mais circuitos, desempenhando funções como um decodificador, um contador, um multiplexador.

Os circuitos LSI contêm 100 ou mais portas ou dispositivos equivalentes, formando grandes circuitos funcionais, equivalentes a vários circuitos MSI. Seu maior campo de aplicação é o das memórias e micro processadores.

Encapsulamento de integrados

Atualmente há três tipos de encapsulamento para acomodar "chips":

TO5 ou "caneca".

FLAT PACK ou invólucro chato.

DIP (Dual In-line Pack) ou em linha dupla.

Encapsulamento TO5 – Esta foi a primeira versão de encapsulamento usada em circuitos integrados, a partir de um invólucro padrão para transistores. Sua principal vantagem reside em seu grande poder de dissipação de calor, e por esta razão encontra maior aplicação nos circuitos lineares.

Figura 17-14 Encapsulamento tipo caneca(TO5)

Encapsulamento chato (Flat Pack) – Apresenta o menor tamanho entre todos eles, sendo assim empregado onde se deseja uma elevada densidade de componentes na placa. Os invólucros têm um formato achatado e são apropriados para soldagem sobre circuitos impressos, podendo ficar muito próximos um dos outros.

Encontram aplicações onde o espaço é crítico, como por exemplo, em aviação, sistemas militares de alta confiabilidade e equipamentos industriais especiais.

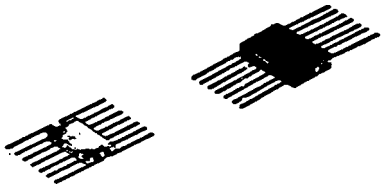


Figura 17-15 Encapsulamento " chato" (Flat Pack)

Encapsulamento DIP (Dual In-Line Package) – O DIP ou encapsulamento em linha dupla, é assim chamado porque exibe duas fileiras paralelas de terminais, tendo sido projetado para adaptar-se às máquinas de inserção automática de componentes em placas de circuitos impressos.

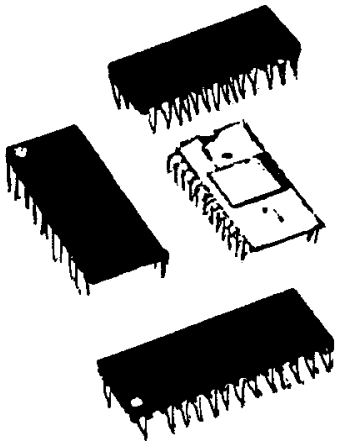


Figura 17-16 Encapsulamento em linha dupla (DIP)

Pode ser encontrado desde o MINI-DIP de oito pinos, ao gigante de quarenta pinos. A maioria dos SSI apresenta-se em encapsulamentos de 8, 14 ou 16 pinos, enquanto o MSI com 14, 16 ou 24 pinos. Finalmente os LSI são encontrados mais freqüentemente com 24, 28 ou 40 pinos.

CIRCUITOS COMBINACIONAIS

Conceitos

Circuito lógico combinacional, ou simplesmente circuito combinacional, é aquele cujo estado de saída é uma função exclusiva das combinações possíveis das variáveis de entrada.

Os circuitos lógicos combinacionais que iremos estudar, são divididos em três categorias:

a – Circuitos Lógicos Básicos
- Porta AND (E).
- Porta OR (OU).
- Porta NOT (NÃO).

b – Circuitos Lógicos Universais
- Porta NAND (NÃO E).
- Porta NOR (NÃO OU).

C – Circuitos Comparadores
- Porta XOR (OU EXCLUSIVO).
- Porta XNOR (NÃO OU EXCLUSIVO)

Circuitos Lógicos Básicos – As portas E, OU e INVERSORA, são ditas básicas porque, através delas, todas as funções lógicas podem ser obtidas.

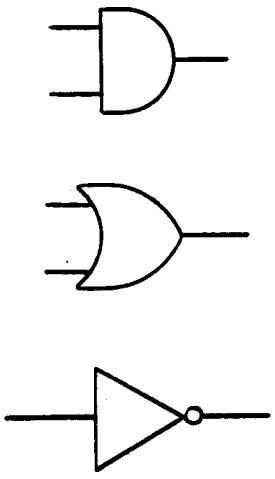


Figura 17-17 Portas básicas

Circuitos Lógicos Universais – Dentre todas as portas lógicas, as portas NAND e NOR, são as mais utilizadas, pois qualquer tipo de circuito lógico pode ser obtido através delas.

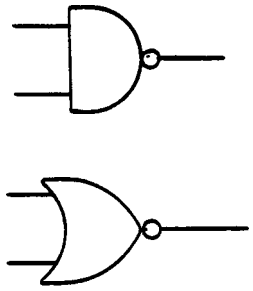


Figura 17-18 Portas NAND e NOR

Circuitos Comparadores – As portas XOR e XNOR são consideradas circuitos comparadores e encontram vasta aplicação onde for necessário comparar expressões ou tomar uma decisão.

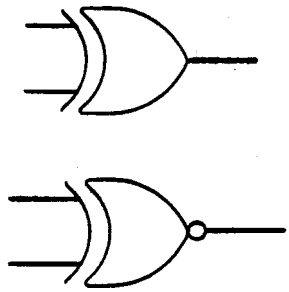


Figura 17-19 Portas XOR e XNOR

Codificadores e decodificadores

Um Codificador tem a função de "tradutor" de um código (linguagem) conhecido ou comum, para um código desconhecido ou incomum.

Um Decodificador tem a função de "tradutor" de um código (linguagem) desconhecido ou incomum, para um código conhecido ou comum.

Da relação dos "bits" 0 e 1 do sistema binário, com os estados lógicos 0 e 1, surgiu a aplicação de circuitos lógicos em calculadoras, com operações realizadas no sistema binário.

Cabe aqui uma pergunta. Por que não empregar nas calculadoras eletrônicas circuitos que realizem operações diretamente no sistema decimal?

A resposta é simples: os circuitos teriam que discernir 1 entre 10 níveis diferentes, contra

1 entre 2, o que os tornaria complicados, caros e volumosos.

Para facilitar a operação da máquina, a entrada dos dados a serem calculados e o resultado das operações, devem estar na forma decimal, que é o código comum aos humanos.

Vê-se, então, a necessidade de componentes lógicos conversores, dotados de circuitos codificadores e decodificadores que realizem as conversões decimal-binário ou binário-decimal.

Estes codificadores e decodificadores, são na verdade circuitos lógicos combinacionais cujas saídas dependem dos estados lógicos das entradas.

Um número decimal pode ser codificado de tal maneira que a operação digital possa ser desempenhada utilizando-se números binários. A conversão de um sistema para o outro é realizada por circuitos codificadores. O circuito que tem a função inversa é denominado decodificador.

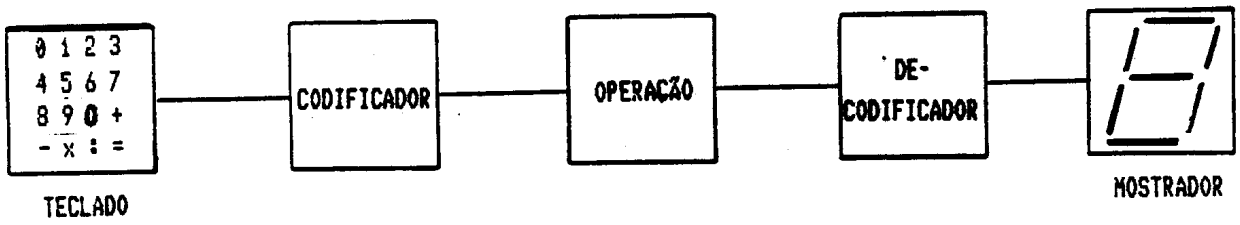


Figura 17-20 Diagrama bloco básico de uma calculadora

Circuito Codificador –

Um codificador consiste de portas lógicas que convertem um número decimal para outro código de representação.

Na figura 17-21 a seguir, temos um circuito codificando um grupo de chaves que representam números decimais, para fornecimento de um código binário de 4 ”bits”.

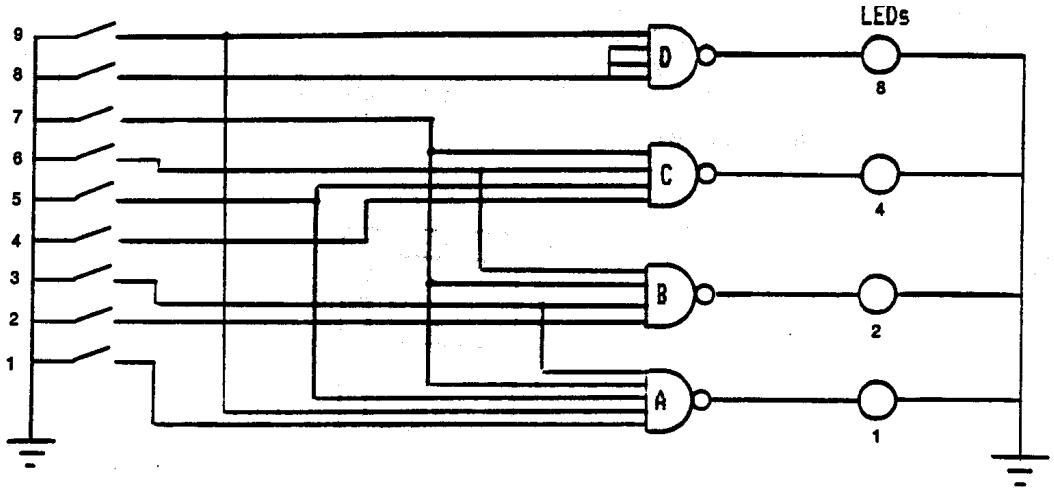


Figura 17-21 Circuito codificador

Quando todas as chaves estiverem abertas, teremos nível “1” (alto) na entrada de todas as portas NAND, ocasionando todas as saídas em nível “0” (baixo), gerando o binário”0000”.

Ao pressionarmos a chave “1”, um nível **baixo** na entrada da porta “A”, ocasionará um **alto** em sua saída, indicando o binário “0001”.

Pressionando a chave “2”, teremos um nível **alto** da saída da porta “B”, acendendo o Led correspondente, indicando o binário”0010”.

Acionando a chave ”6”, as portas B e C terão saída **alta**, ocasionando a indicação binária “0110”.

“Display” de segmentos – A apresentação do resultado anteriormente descrita, não é satisfatória, pois nem todos os humanos compreendem a representação binária. São necessários então, componentes que nos proporcionem uma forma simples de representação.

Os “displays” de sete segmentos são componentes mais comuns para representação numérica. Estes “displays” possibilitam representarmos números de cimais e alguns outros símbolos. São compostos por segmentos que podem ser ativados individualmente, permitindo combinações.

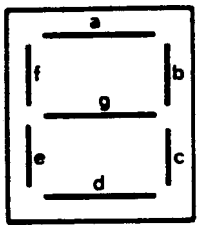


Figura 17-22 “Display” de sete segmentos

Para formação do algarismo zero (0), necessitamos ativar os segmentos “a”, “b”, “c”, “d”, “e” e “f”, desativando o segmento “g”.

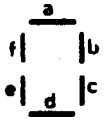


Figura 17 23 Representação do zero (0)

A representação do algarismo quatro (4) requer a ativação dos segmentos “b”, “c”, “f” e “g”.

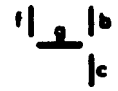


Figura 17 24 Representação do quatro (4)

Circuito Decodificador – Como o código interno normalmente utilizado é o binário, torna-se necessário um decodificador que permita a ativação individual dos segmentos. Este decodificador possui a seguinte tabela verdade:

DECIMAL	BCD 8421 A B C D 8 4 2 1	CÓDIGO DE 7 SEGMENTOS a b c d e f g
0	0 0 0 0	1 1 1 1 1 1 0
1	0 0 0 1	0 1 1 0 0 0 0
2	0 0 1 0	1 1 0 1 1 0 1
3	0 0 1 1	1 1 1 1 0 0 1
4	0 1 0 0	0 1 1 0 0 1 1
5	0 1 0 1	1 0 1 1 0 1 1
6	0 1 1 0	1 0 1 1 1 1 1
7	0 1 1 1	1 1 1 0 0 0 0
8	1 0 0 0	1 1 1 1 1 1 1
9	1 0 0 1	1 1 1 1 0 1 1

As funções da tabela poderão ser obtidas através do circuito da figura 17-25.

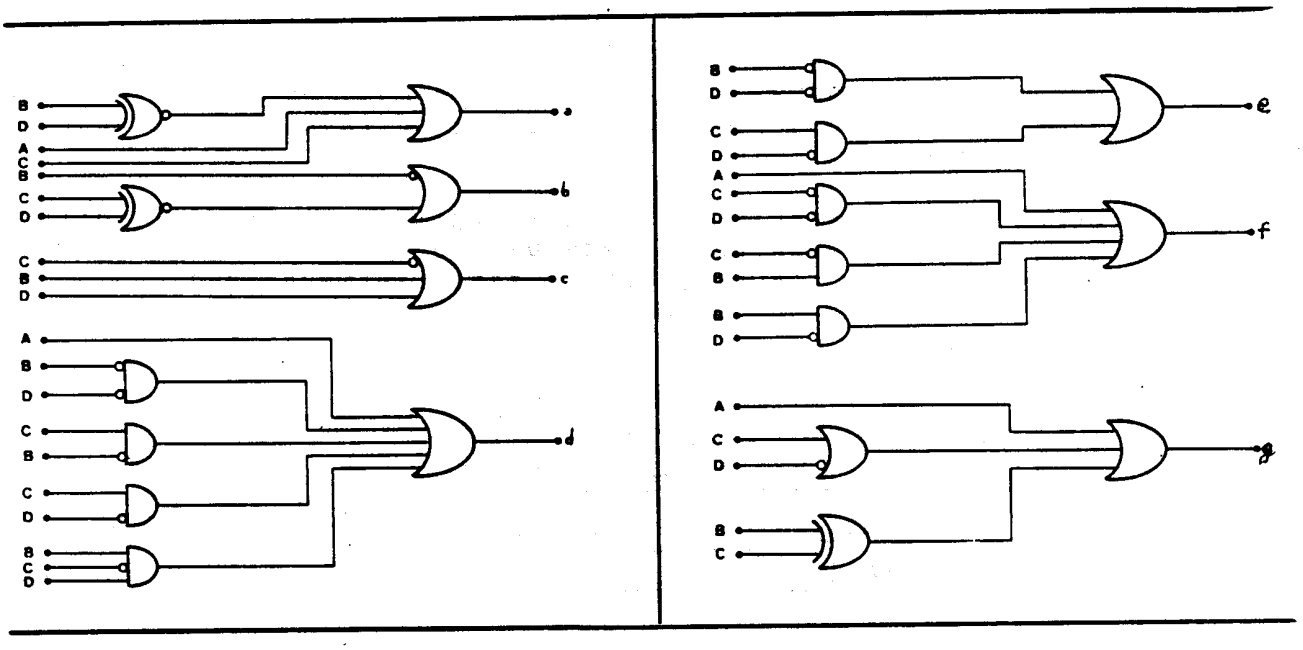


Figura 17-25 Decodificador para “Display” de sete segmentos

Somadores e subtratores

Somador – Se quisermos somar dois dígitos binários, teremos duas entradas para o circuito de soma, havendo quatro combinações para estas entradas: (0 + 0), (0 + 1), (1 + 0) e (1 + 1).

Na aritmética binária, "1" mais "1" (1+1) é igual a 0 (zero) e um dígito 1 é transportado para a coluna da esquerda.

A + B = S	T
0 + 0 = 0	0
0 + 1 = 1	0
1 + 0 = 1	0
1 + 1 = 0	1

De acordo com a tabela verdade, a função soma (S) pode ser executada por uma porta XOR (OU EXCLUSIVA)), e a função transporte (T) por uma porta AND.

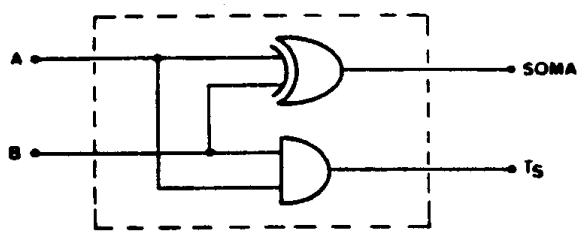


Figura 17-26 Meio Somador (Half Adder)

Para somar as colunas menos significativas, será suficiente o circuito acima, com duas entradas, que é denominado Meio Somador (Half Adder), porém ao somarmos as demais colunas teremos que considerar uma terceira entrada, o transporte da coluna anterior.

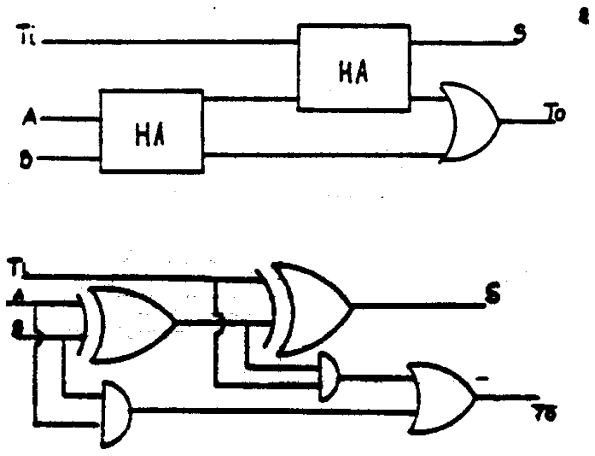


Figura 17-27 Somador Completo (Full Adder)

O somador que executa a soma dos dígitos mais significativos e que possui uma terceira entrada para o transporte, é denominado Somador Completo (Full Adder), sendo formado por dois "Half Adders" (HÁ) e uma porta OR.

Um somador será composto de vários "Full Adder" (HÁ), para a coluna menos significativa.

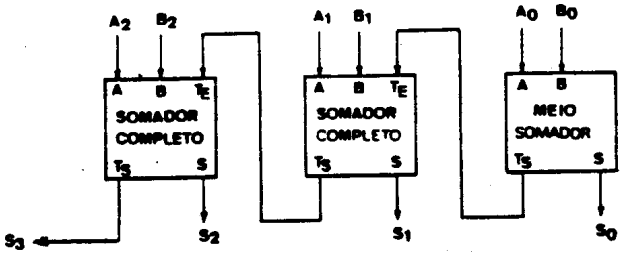


Figura 17-28 Somador para dois dígitos de três "bits"

Subtrator – Na aritmética binária, "0" menos "1" (0 – 1) é igual a "1" e um dígito 1 é tomado emprestado da coluna da esquerda.

A - B = S	E
0 - 0 = 0	0
0 - 1 = 1	1
1 - 0 = 1	0
1 - 1 = 0	0

Toma 1 emprestado

Analogamente ao somador, para subtrairmos dígitos na coluna menos significativa, fazemos uso de um Meio Subtrator (Half Subtractor) e, para as demais colunas, utilizamos o Subtrator Completo (Full Subtractor).

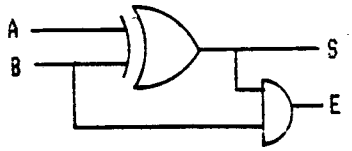


Figura 17-29 Meio Subtrator (Half Subtractor)

Um subtrator será composto de vários "Full Subtractors" (FS), para as colunas mais significativas e um "Half Subtractor" (HS), para a coluna menos significativa.

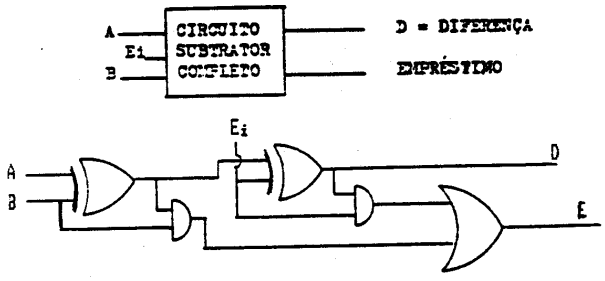


Figura 17-30 Subtrator Completo (Full Subtractor)

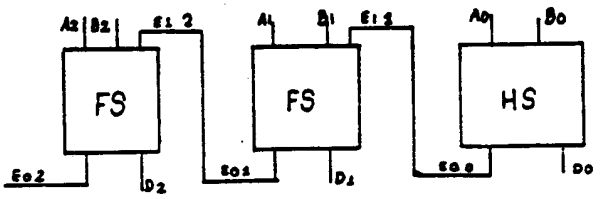


Figura 17-31 Subtrator para dois dígitos de três "Bits"

Multiplexadores e Demultiplexadores

Os Multiplexadores são componentes que permitem selecionar um dado, dentre diversas fontes, como uma chave seletora de várias posições.

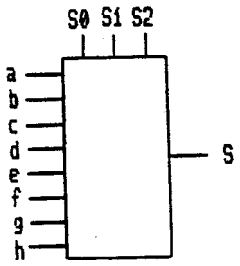


Figura 17-32 Multiplexador

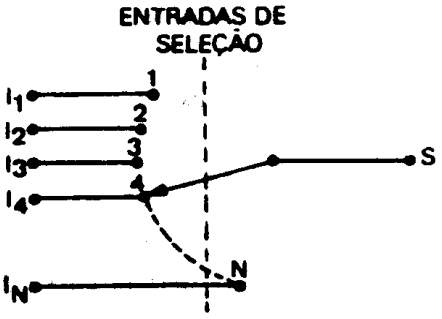


Figura 17-33 Circuito equivalente

Através do circuito equivalente verificamos que a saída poderá estar ligada a qualquer das entradas, bastando posicionar a chave. No multiplexador, a seleção é feita de acordo como valor digital das entradas de seleção (S0), (S1) e (S2), com pesos binários 1, 2 e 4, respectivamente. As entradas de "A" a "H", corresponderão a valores decimais de 0 a 7. Na saída, teremos o nível da entrada, cujo valor decimal corresponde ao valor binário das entradas seletoras.

Os Demultiplexadores são componentes que distribuem o nível de uma única entrada, para uma, dentre as várias saídas, de acordo com o valor binário das entradas seletoras.

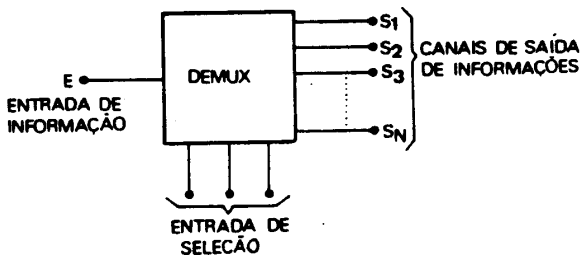


Figura 17-34 Demultiplexador

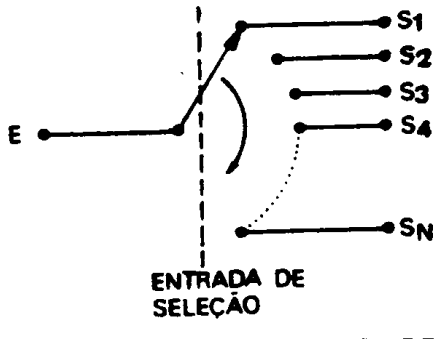


Figura 17-35 Circuito equivalente

CIRCUITOS SEQUENCIAIS

Os circuitos combinacionais vistos anteriormente apresentam as saídas dependentes de variáveis de entrada.

Os circuitos seqüenciais têm as saídas dependentes de variáveis de entrada e de seus estados anteriores que foram armazenados.

Circuitos seqüenciais são normalmente sistemas pulsados, isto é, operam sob o comando de pulsos denominados "Clock".

Dentre os componentes utilizados em circuitos seqüenciais, o "Flip-Flop" é um dispositivo fundamental, que permite, por suas características, o armazenamento de estados lógicos anteriores.

Flip-Flop

Flip-Flop é um dispositivo que possui dois estados estáveis. Um pulso em suas entradas poderá ser armazenado, e transformado em nível lógico estável.

Há vários tipos de Flip-Flop, que podem ser representados basicamente conforme a figura 17-36.

Figura 17-36 Flip-Flop

Um pulso na entrada "S", será armazenado, tornando "Q" verdadeiro e "$\overline{Q}$" falso. Um pulso na entrada " R" , será armazenado, tornando "Q" falso e "$\overline{Q}$" verdadeiro.

Flip-Flop tipo "RS" (Latch)

	S	R	S1	R1	Q,	$\overline{Q}$
A	1	0	0	1	1	0
B	0	0	1	1	1	0
C	0	1	1	0	0	1
D	0	0	1	1	0	1
E	1	1	ilegal			

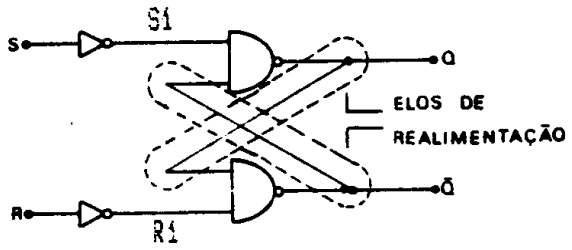


Figura 17-37 Flip-Flop tipo "RS"

Flip-Flop "RS" comandado por Clock – Substituem-se os inversores na entrada do RS básico, por portas NAND.

	S	R	CLK	S1	R1	Q	$\overline{Q}$
A	1	0	0	1	1	0	0
B	1	0	1	0	1	1	0
C	0	0	0	1	1	1	0
D	1	1	1	ilegal			

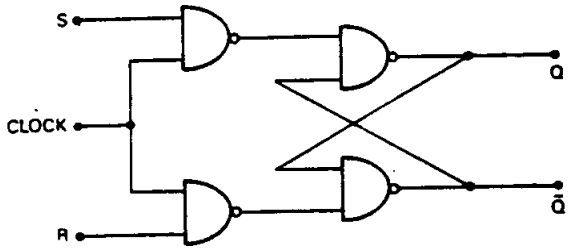


Figura 17-38 Flip-Flop "RS" comandado por CLOCK

Flip-Flop JK – Os Flip-Flop "RS" possuem um estado não permitido, quando as entradas "R" e "S" são iguais a "1" acarretando uma saída indeterminada. O Flip-Flop "JK" resolve este problema, utilizando um "RS" realimentado.

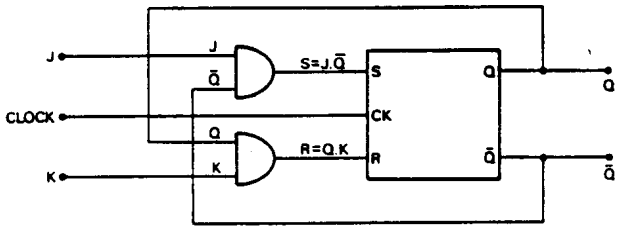


Figura 17-39 Flip-Flop "JK"

De acordo com o circuito, o FF JK, com as entradas J e K no estado "1", terá seu estado complementado a cada "clock", isto é, se estiver "setado" (saída Q = 1), complementará (Q - >0 e $\overline{Q}$ - > 1), se estiver "ressetado" (saída Q = 0), complementará (Q - > 1 e $\overline{Q}$ - > 0).

Flip-Flop "JK" Mestre-Escravo – No FF JK, no momento em que o Clock for igual a "1", o circuito funcionará como um combinacional, passando o estado das entradas J e K diretamente para a saída.

Para evitar este inconveniente, criou-se o Flip-Flop JK Mestre-Escravo (Master-Slave), que consiste basicamente de dois FF JK, permitindo a comutação do FF, apenas na transição positiva ou negativa do Clock.

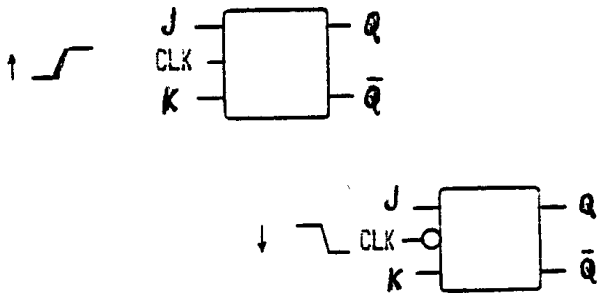


Figura 17-40 Flip-Flop JK Mestre-Escravo

Flip-Flop tipo "T" – Consiste de um FF JK com as entradas J e K interligadas. Sua característica é de complementar-se toda vez que a entrada estiver igual a "1", mantendo-se no último estado quando a entrada for igual a "0".

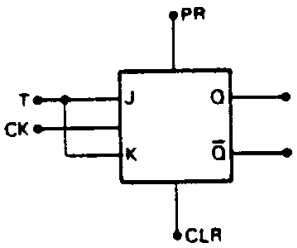


Figura 17-41 Flip-Flop tipo "T"

Flip-Flop tipo "D" – Consiste de um FF JK com as entradas interligadas através de um inversor, permitindo que seja "setado" (colocado no estado "1") quando, no momento do Clock a entrada estiver igual a "1", e que seja "ressetado" (colocado no estado "0"), quando, no momento do Clock a entrada estiver igual a "0".

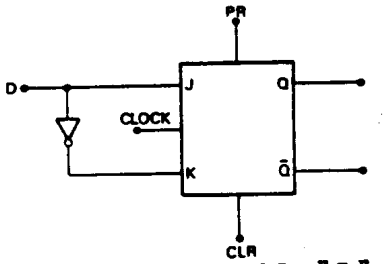


Figura 17-42 Flip-Flop tipo "D"

Contadores

São circuitos digitais compostos de Flip-Flops, que variam seus estados, sob comando de um Clock, de acordo com uma sequência pré-determinada.

O que determinará a capacidade de um contador, será o número de Flip-Flop utilizados.

Contador de pulsos – Consiste de um grupo de FF Master-Slave de comutação na transição negativa do Clock, configurados em série, de tal modo que a saída de cada estágio terá a metade da frequência do estágio anterior.

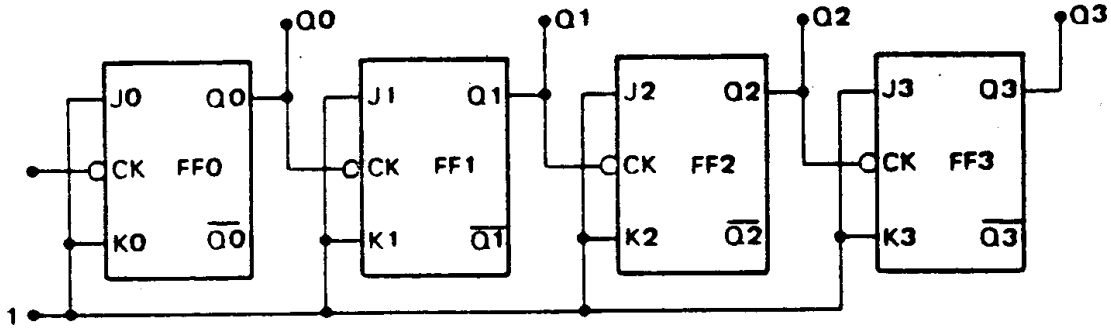


Figura 17-43 Contador de pulsos

Contadores decrescentes

O circuito que efetua a contagem crescente é o mesmo para contagem decrescente, com a diferença de utilizar as saídas "Q" dos FFs. A tabela verdade de um contador crescente corresponderá ao complemento da tabela de um contador decrescente.

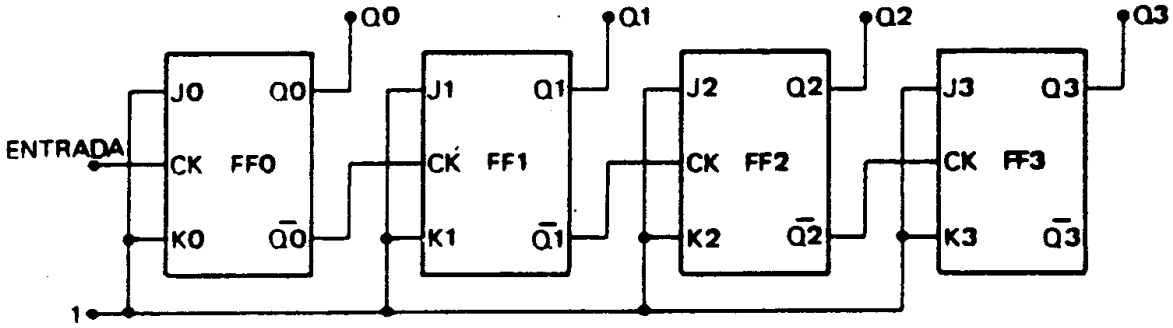


Figura 17-44 Contador decrescente

Registradores (Shift Registers)

O flip-Flop tem a característica de armazenar o valor de um "bit", mesmo que sua entrada não esteja mais presente. Se necessitarmos guardar informações com uma quantidade de "bits" maior que um (1), o Flip-Flop será insuficiente. Para isso utilizamo-nos de um componente denominado Registrador de Deslocamento (Shift Register), que compõe-se de um certo número de Flip-Flops, de forma que as saídas de um alimentem as entradas do FF seguinte. Cada estágio do registrador armazenará o sinal de entrada no momento do Clock. Serão necessários tantos "Clocks", quantos forem os "bits" a serem armazenados.

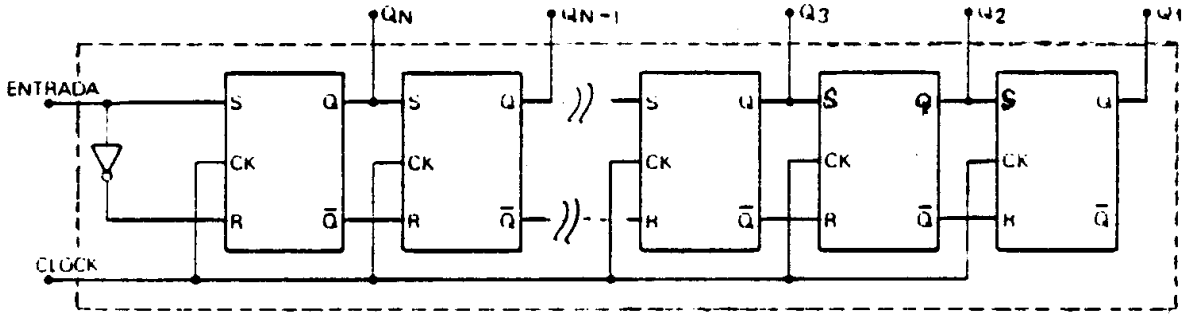


Figura 17-45 Registrador de Deslocamento (Shift Register)

Este tipo de registrador é bastante utilizado nas conversões de sistemas seriais para sistemas paralelos, onde a entrada recebe os sinais serialmente, recebendo ao final a informação completa paralela.

MEMÓRIAS

Memórias são dispositivos que armazenam informações. Essas informações poderão ser números, letras, ou caracteres quaisquer

Tipos de memórias

Podemos classifica-las quanto a:

a) Acesso.
b) Volatilidade.
c) Possibilidade de regravação.
d) Retenção.

Acesso – As memórias armazenam as informações em áreas internas chamadas "endereços".

Dependendo da codificação utilizada, cada endereço conterá um conjunto de "bits", ao qual chamamos "palavra".Cada endereço conterá uma palavra de memória.

Podemos acessar palavras de memória de duas maneiras:

-Acesso Seqüencial.
-Acesso Aleatório.

No acesso Seqüencial, o endereçamento será feito em sequência, isto é, para uma dada posição de memória todos os endereços precisam ser acessados desde o primeiro endereço. Em virtude disto, o tempo de acesso dependerá do lugar onde a informação estiver armazenada.

Como exemplo comparativo, podemos citar a fita cassete. Para acessarmos uma música que esteja no meio da fita, precisaremos percorre-la desde o princípio.

No acesso Aleatório, o endereçamento é feito diretamente na palavra desejada, sem necessidade de passar-se pelas posições intermediárias. Estas memórias são conhecidas por RAM (Random Access Memory). Como principal vantagem têm o tempo de acesso, que é reduzido e idêntico para qualquer endereço.

Como exemplo comparativo, podemos citar um disco. Para acessarmos qualquer música, bastará posicionar o braço do toca-discos na mesma.

Volatilidade – Podem ser voláteis e não voláteis. As memórias voláteis são aquelas que perdem a informação armazenada quando da interrupção da sua alimentação.

As memórias não voláteis são aquelas que mantém armazenadas as informações, mesmo na ausência de alimentação.

Possibilidade de regravação – As memórias de Escrita / Leitura permitem o acesso a qualquer

endereço, para consulta da informação (Leitura) (Leitura) ou para alteração da informação (Gravação).

São utilizadas em processos onde é necessária a constante alteração das informações. São normalmente identificadas como RAM (Random Access Memory).

As memórias apenas de Leitura (Read Only Memory ou ROM) são aquelas cuja informação somente estará disponível para Leitura.

São utilizadas em processos onde a informação é necessária para consulta ou inicialização de uma rotina. Possuem capacidade de armazenamento, isto é, quantidade de endereços, inferior às RAM´s.

Quanto a esta classificação podemos citar:

a) PROM (Programable Read Only Memory) - São memórias apenas para leitura, que permitem que a sua programação, isto é, a gravação inicial seja feita pelo usuário. Esta gravação é permanente, não permitindo alterações, passando ela a operar como uma ROM.
b) EPROM (Eraseble / Programable Read Only Memory) – São memórias que funcionam como PROM´s, que permitem, porém o seu apagamento e posterior regravação. O processo de apagamento é possível por meio de um "banho" ultravioleta, através de janelas no seu encapsulamento.
c) EEPROM – São EPROM´s que permitem sua regravação por meios elétricos, sem necessidade de banhos Ultravioleta.

É conveniente lembrar que, embora as EEPROM´s permitam regravações, a sua aplicação é diferente das RAM´s. As EEPROM´s, assim como as EPROM´s, PROM´s e ROM´s, são utilizadas para armazenamento de informações que durante um processo são apenas consultadas, como as instruções para sequência de um programa. A característica de regravação em alguns tipos de ROM, tem por finalidade permitir alterações nestas instruções, sem a necessidade de substituição de componentes.

Retenção – Classificam-se em Estáticas e Dinâmicas.

As memórias de armazenamento Estático, retém os dados inseridos enquanto a alimentação estiver presente.

As memórias Dinâmicas, por outro lado, possuem um efeito capacitivo, isto é, perdem as informações carregadas, após um determinado tempo, necessitando de ciclos periódicos de "recarga" (Refresh Cycle). As memórias Estáticas são mais caras e de menor capacidade.

Endereçamento

Como já foi visto anteriormente, cada posição de memória, é acessada através de um endereço, logo teremos tantos endereços quantas forem as posições de memória. A capacidade de memória corresponderá à quantidade de endereços possíveis.

Com dois "bits" como variáveis, obtemos quatro combinações, que nos permitem acessar quatro endereços: posições 00, 01, 10 e 11. Com "n" bits variáveis podemos obter 2^n endereços.

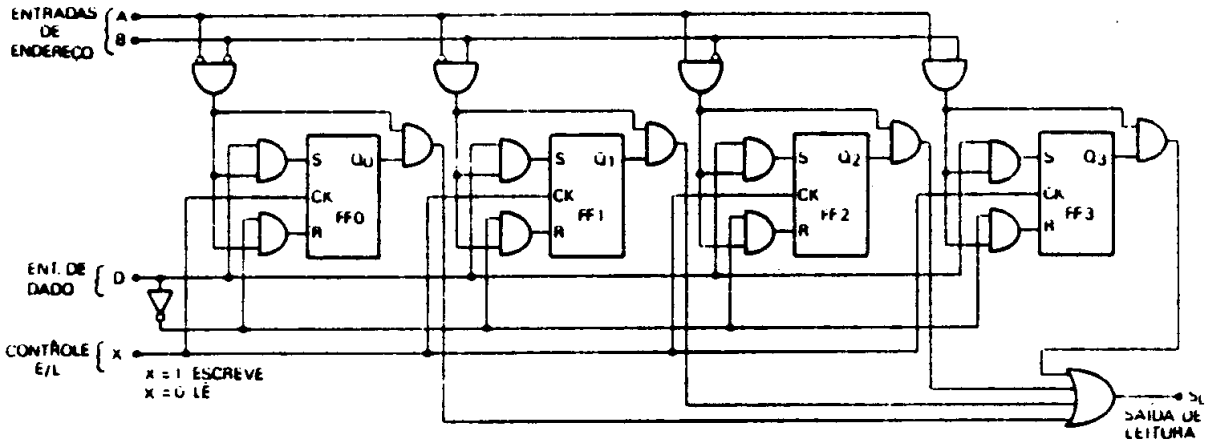


Figura 17-46 Memória RAM de quatro "bits"

Palavra de memória – Cada endereço de memória corresponderá a quantidade mínima de informações que poderá ser acessada. Esta informação poderá compor-se de um ou mais "bits".

Esta quantidade de "bits" por endereço, chamada de "palavra de memória", dependerá dos circuitos associados a ela e ao código interno utilizado.

As palavras mais comuna compõem-se de 8, 16 ou mesmo 32 "bits". A cada posição acessada, serão lidos paralelamente 8, 16 ou 32 "bits".

Byte – É o nome dado ao agrupamento de "bits" que represente um tipo de informação identificável e dependerá da filosofia do fabricante do equipamento.

Normalmente um "byte" é composto por 8 "bits".

Qualquer caractere significativo será representado na forma de um "BYTE". Uma memória com 1 kilobytes (1 kB), indica uma capacidade de armazenamento de 1000 caracteres.

Aplicação

Memórias são aplicadas de formas diversas, mas sempre que for necessário o armazenamento temporário ou permanente de informações.

Uma informação poderá ser um valor a ser processado, o resultado de uma operação, ou mesmo a própria sequência com as instruções da operação.

Valores fixos ou variáveis, em processamento, são chamados "DADOS". Seqüências de instruções de operação são chamadas de "PROGRAMAS".

As instruções de um programa, são normalmente armazenadas em memórias do tipo "ROM", pois são informações fixas. Dados são armazenados normalmente em memórias do tipo "RAM.

CONVERSÃO DE SINAIS

Existem basicamente dois tipos de sinais: Analógicos e Digitais

Sistemas digitais e analógicos não são compatíveis entre si, necessitando de conversores.

Os conversores têm por finalidade transformar sinais digitais em analógicos e vice-versa.

Sistemas Analógicos e Digitais

Entende-se por ANALÓGICA, toda variação linear ou contínua de um sinal. Grandezas físicas como temperatura, pressão, tensão, resistência, variam de forma analógica.

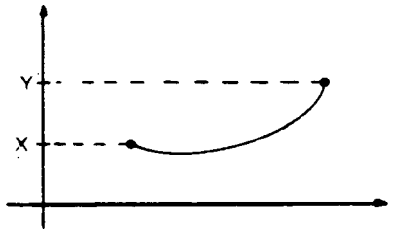


Figura 17-47 Gráfico de variação Analógica

Entende-se por DIGITAL, toda variação discreta, isto é, em degraus definidos ou "steps".

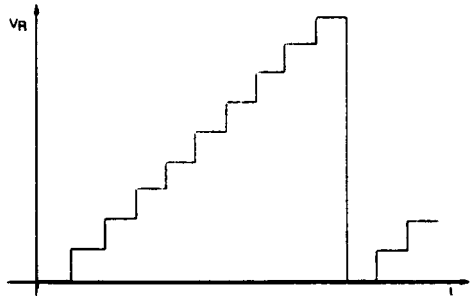


Figura 17-48 Gráfico de variação Digital

Os sistema digitais, como internamente se utilizam de valores binários, somente reconhecem duas variações discretas, o zero (0) e o um (1). Aos sinais utilizados por estes sistemas chamamos digitais binários.

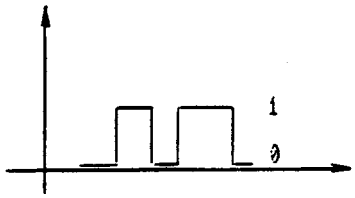


Figura 17-49 Gráfico de variação digital binária

Amplificadores operacionais– São componentes lineares cuja finalidade é amplificar uma diferença entre dois sinais, possuindo ganho controlável. Diferenças de amplitude entre dois sinais são amplificadas gerando uma saída proporcional a entrada.

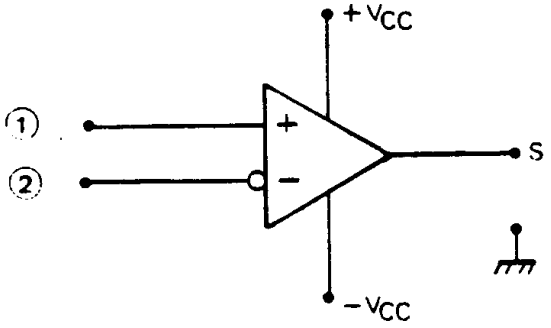


Figura 17-50 Amplificador operacional

A existência de duas entradas, sendo uma inversora e a outra não inversora permite que, dependendo de sua utilização, o sinal de saída seja normal ou invertido.

O limite de amplificação, isto é, o valor máximo de amplitude de saída, dependerá das alimentações do amplificador, limitando-se aos seus valores. A partir daí, o amplificador estará saturado, mantendo a saída fixa até que a diferença entre as entradas seja reduzida.

Conversor Digital-Analógico

É utilizado quando for necessária a conversão de uma variável digital em variável analógica.

A variável digital é normalmente codificada em BCD 8421. A saída analógica assumirá valores de grandeza correspondentes às variações digitais da entrada.

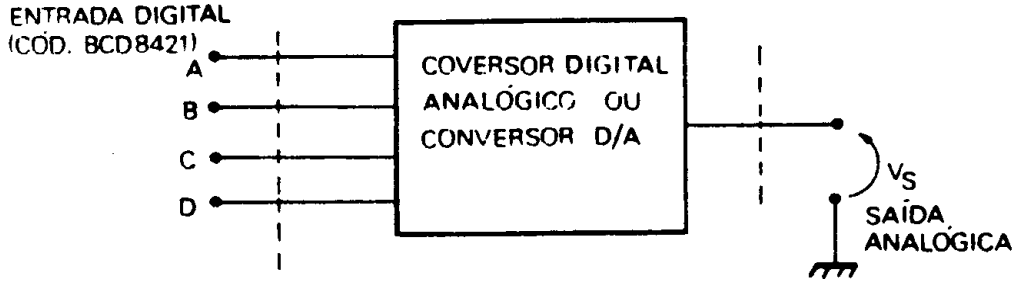


Figura 17-51 Conversor D / A

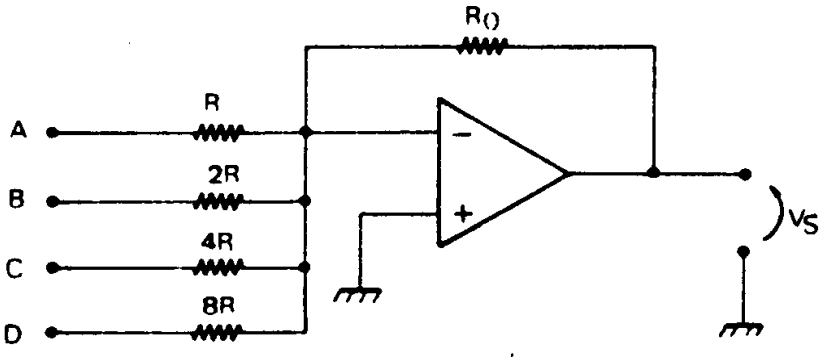


Figura 17-52 Circuito básico

Sistemas de computação digital, não são capazes de gerar sinais analógicos linearmente, mas dependendo da precisão desejada, poderão ser utilizados mais "bits", que gerando mais "steps", darão condições de geração de sinais bastante próximos dos analógicos.

Conversor Analógico-Digital

É utilizado quando for necessária a conversão de uma variável analógica em variável digital.

O conversor efetua vários passos até a conversão final, utilizando-se de um contador, um conversor D / A, um amplificador operacional atuando como comparador e Flip-Flops.

O circuito é basicamente constituído por um contador de década, gerando um código BCD 8421, que é aplicado ao conversor D/A, que por sua vez apresenta na saída uma tensão de referência (VR). Esta tensão de referência é comparada no amplificador operacional, com o sinal analógico de entrada (Ve).

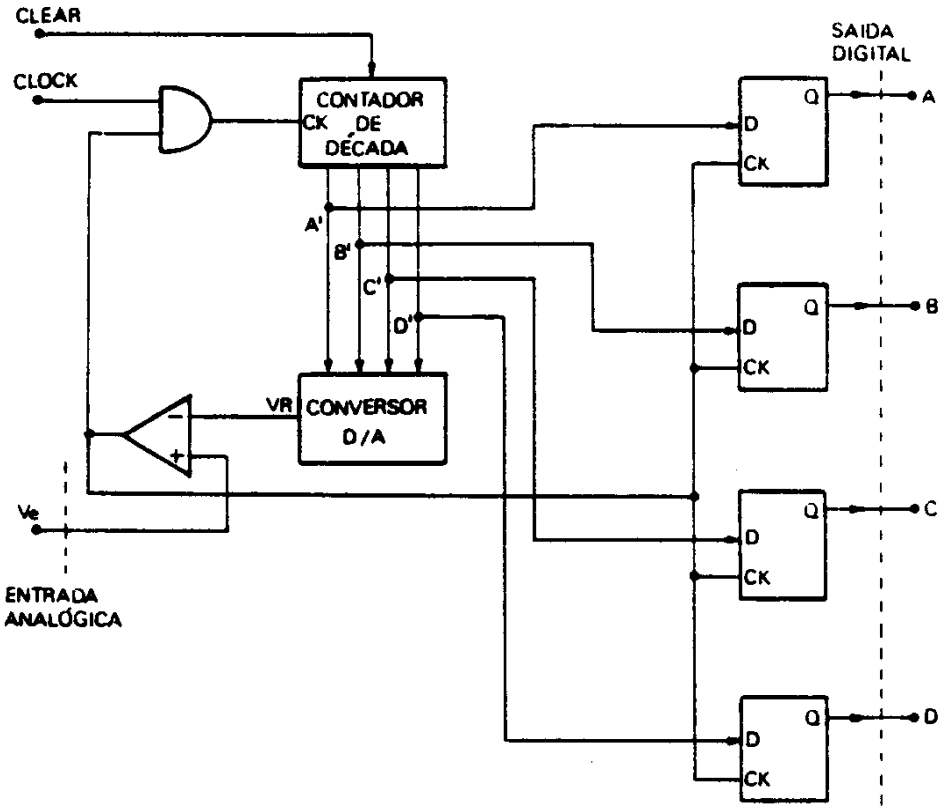


Figura 17-53 Conversor A / D

Enquanto VR for menor que Ve, a saída do operacional habilitará o incremento do contador a cada "clock".Quando VR = Ve, o operacional dará saída "0" desabilitando o contador e, simultaneamente habilitando a transferência do conteúdo do contador para os Flip-Flops, que apresentarão na saída o valor digital correspondente à entrada analógica.