S. R.

MINISTÉRIO DA DEFESA NACIONAL
FORÇA AÉREA PORTUGUESA
CENTRO DE FORMAÇÃO MILITAR E TÉCNICA

Curso de Formação de Praças - RC

COMPÊNDIO

TÉCNICAS DIGITAIS

EPR: SAJ João Marques

CCF 335-37

Fevereiro 2009

ENTIDADE FORMADORA ACREDITADA POR DGERT DIRECÇÃO-GERAL DO EMPREGO E DAS RELAÇÕES DE TRABALHO

S.  R.

MINISTÉRIO DA DEFESA NACIONAL
FORÇA AÉREA PORTUGUESA
CENTRO DE FORMAÇÃO MILITAR E TÉCNICA

CARTA DE PROMULGAÇÃO

FEVEREIRO 2009

1. O Compêndio de "Técnicas Digitais" é uma Publicação "NÃO CLASSIFICADA".

2. Esta publicação entra em vigor logo que recebida.

3. É permitido copiar ou fazer extractos desta publicação sem autorização da entidade promulgadora.

O COMANDANTE

José Alberto Fangueiro da Mata

COR/PILAV

REGISTO DE ALTERAÇÕES

IDENTIFICAÇÃO DA ALTERAÇÃO, Nº DE REGISTO, DATA	DATA DE INTRODUÇÃO	DATA DE ENTRADA EM VIGOR	ASSINATURA, POSTO E UNIDADE DE QUEM INTRODUZIU A ALTERAÇÃO

Cursos:	Curso de Formação de Praças - RC
Nome do Compêndio:	Técnicas Digitais
Disciplina:	Técnicas Digitais
Data de elaboração:	Dezembro 2009
Elaborado Por:	SAJ/MELECA João Marques
Verificado Por:	Gabinete da Qualidade da Formação
Comando G. Formação:	TCOR / ENGAER José Saúde 
Director de Área:	MAJ / TMMEL Abílio Carmo 
Director de Curso:	TEN / TMMEL José Martins
Formador:	SAJ / MELECA João Marques

ATENÇÃO:

Esta publicação destina-se a apoiar os formandos a frequentarem o Curso de Formação de Praças das especialidades MMA e MARME na disciplina de Técnicas Digitais.

Não pretendendo ser uma publicação exaustiva do curso em questão, apresenta-se como uma ferramenta de consulta quer durante a duração do curso, quer após a sua conclusão.

ÍNDICE

SISTEMAS DE NUMERAÇÃO 5

DEFINIÇÕES 5
SISTEMA BINÁRIO 5
SISTEMA OCTAL 6
SISTEMA HEXADECIMAL 7
CONVERSÃO BINÁRIO DECIMAL 9
CONVERSÃO OCTAL DECIMAL 9
CONVERSÃO HEXADECIMAL DECIMAL 9
CONVERSÃO DECIMAL BINÁRIO 10
CONVERSÃO DECIMAL OCTAL 11
CONVERSÃO DECIMAL HEXADECIMAL 12
CONVERSÃO BINÁRIO OCTAL 13
CONVERSÃO BINÁRIO HEXADECIMAL 14
ARITMÉTICA BINÁRIA (SOMA) 15
ARITMÉTICA BINÁRIA (SUBTRACÇÃO) 17
ARITMÉTICA BINÁRIA (MULTIPLICAÇÃO) 18
ARITMÉTICA BINÁRIA (DIVISÃO) 19
COMPLEMENTO A 2 20
SUBTRACÇÃO COM O MÉTODO COMPLEMENTO A 2 24
OPERAÇÕES LÓGICAS ELEMENTARES 25
PORTAS LÓGICAS ELEMENTARES 26

ARQUITECTURA BÁSICA DOS COMPUTADORES 33

TERMINOLOGIA 33
TIPOS DE MEMÓRIAS (ROM) 34
TIPOS DE MEMÓRIAS (RAM) 36
TECNOLOGIA USADA EM COMPUTADORES 38

CONVERSÃO DE DADOS 43

CONVERSORES DIGITAIS ANALÓGICOS 43
CONVERSOR ANALÓGICOS DIGITAIS 47

BIBLIOGRAFIA 55

LISTA DE PÁGINAS EM VIGOR ..... LPV - 1

SISTEMAS DE NUMERAÇÃO

DEFINIÇÕES

Base de um sistema de numeração: Representa o número de símbolos distintos usados para representar qualquer quantidade, dentro desse sistema.

Número: É uma abstracção matemática que é utilizada para fins de quantificação

Valor intrínseco: Valor propriamente dito do próprio dígito ou algarismo por si só.

Valor posicional. É o valor a que cada dígito está associado e que depende da posição que ele ocupa dentro do número.

SISTEMA BINÁRIO

Neste sistema de numeração utilizam-se somente dois símbolos: 0 e 1

Após o número 1 voltamos ao 0, assim 1 + 1 = 10, ou seja, é 0 e vai 1

Normalmente designa-se por sistema de numeração de base 2 ou binário natural.

Cada dígito denomina-se Bit (**Bi**nary Digi**t**)

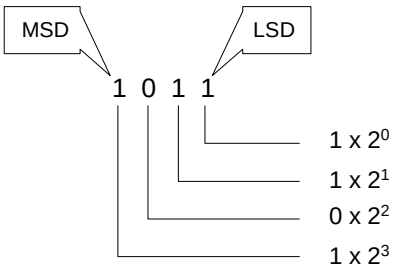
Aqui os pesos são 1, 2, 4, 16, 32, … que correspondem às potências 2^0, 2^1, 2^2, 2^3, 2^4, 2^5, …

Os pesos fraccionários são $\frac{1}{2}$, $\frac{1}{4}$, $\frac{1}{8}$, etc, que correspondem a 2^{-1}, 2^{-2}, 2^{-3}

Exemplos:

$1011_{(2)} = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$

$101{,}101_{(2)} = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3}$



SISTEMA OCTAL

É um sistema que utiliza 8 dígitos (Octal): 0, 1, 2, 3, 4, 5, 6, 7

Após o número 7 voltamos ao 0, assim 7 + 1 = 10, ou seja, é 0 e vai 1

Este sistema tornou-se muito prático no tratamento de informação digital, a qual é costume utilizar números de oito elementos binários.

Facilita a representação de números binários com muitos bits.

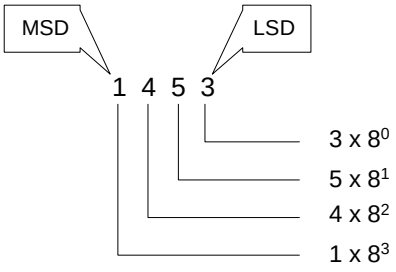
Cada dígito octal equivale a um número binário com 3 dígitos:

Exemplo: $111_{(2)} = 7_{(8)}$

$101_{(2)} = 5_{(8)}$

$100_{(2)} = 4_{(8)}$

Também neste sistema, cada dígito tem um valor numérico e um valor posicional:



Representação de um número octal através do desenvolvimento de potências de base octal:

Exemplo: $123_{(8)} = 1 \times 8^2 + 2 \times 8^1 + 3 \times 8^0$

Os expoentes de um número fraccionário (octal) através das posições dos dígitos.

Representação de um número fraccionário (octal) através do desenvolvimento de potências:

Exemplo: $756,205_{(8)} = 7 \times 8^2 + 5 \times 8^1 + 6 \times 8^0 + 2 \times 8^{-1} + 0 \times 8^{-2} + 5 \times 8^{-3}$

Os expoentes à direita da vírgula são negativos e os seus valores indicam quantas casas estão desviados.

SISTEMA HEXADECIMAL

É um sistema que utiliza 16 dígito, ou seja, algarismos de 0 a 9 e letras de A a F:

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

Após o número F voltamos ao 0, assim F + 1 = 10, ou seja, é 0 e vai 1

Comparando com o sistema octal, este sistema tem mais vantagens, pois torna-se mais fácil a representação de números binários com muitos bits.

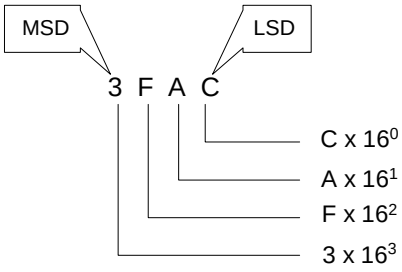
Cada dígito hexadecimal equivale a um número binário com 4 dígitos.

Exemplo: $1111_{(2)} = F_{(16)}$

$1010_{(2)} = A_{(16)}$

$1001_{(2)} = 9_{(16)}$

Também aqui, cada dígito tem um valor numérico e um valor posicional:



Representação de um número através do desenvolvimento de potências de base 16 (Hexadecimal):

Exemplo: $AFA2_{(16)} = A \times 16^3 + F \times 16^2 + A \times 16^1 + 2 \times 16^0$

NOTA: Mais uma vez se verifica que os valores das potências referem-se aos valores das posições dos dígitos:

Os pesos dos dígitos do número $AFA2_{(16)}$, por ordem crescente, são: 16^3, 16^2, 16^1, 16^0

Número fraccionário:

Exemplo: $F16,FAC_{(16)} = F \times 16^2 + 1 \times 16^1 + 6 \times 16^0 + F \times 16^{-1} + A \times 16^{-2} + C \times 16^{-3}$

Sistema Decimal	Sistema Binário	Sistema Octal	Sistema Hexadecimal
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	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

CONVERSÃO BINÁRIO DECIMAL

Recorrendo ao método do desenvolvimento em potências de 2, obteremos para o seguinte exemplo:

$11011,101_{(2)} = 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3}$

$= 1 \times 16 + 1 \times 8 + 0 \times 4 + 1 \times 2 + 1 \times 1 + 1 \times 0,5 + 0 \times 0,25 + 1 \times 0,125$

$= 16 + 8 + 2 + 1 + 0,5 + 0,125$

$= 27,625_{(10)}$

NOTA: No caso do sistema binário, o desenvolvimento em potências da base é equivalente à soma dos pesos.

CONVERSÃO OCTAL DECIMAL

Aplica-se também o método do polinómio resultante do desenvolvimento das potências da base:

Exemplo: $734,45_{(8)} = 7 \times 8^2 + 3 \times 8^1 + 4 \times 8^0 + 4 \times 8^{-1} + 5 \times 8^{-2}$

$= 448 + 24 + 4 + 0,5 + 0,078125$

$= 476,578125_{(10)}$

CONVERSÃO HEXADECIMAL DECIMAL

Utiliza-se o método anterior, mudando apenas a base.

Exemplo: $2AF3,5_{(16)} = 2 \times 16^3 + A \times 16^2 + F \times 16^1 + 3 \times 16^0 + 5 \times 16^{-1}$

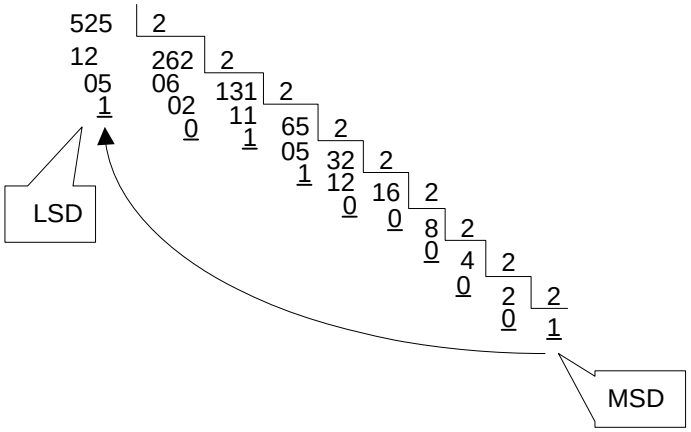
$= 2 \times 4096 + 10 \times 256 + 15 \times 16 + 3 \times 1 + 5 \times 0{,}0625$

$= 10995{,}3125_{(10)}$

CONVERSÃO DECIMAL BINÁRIO

Para passar um número inteiro da base decimal para binário, divide-se o número inteiro por 2; o quociente torna a dividir-se por 2, e assim sucessivamente; os restos obtidos e o último quociente constituem o número no sistema binário.

Exemplo: converter o número $525_{(10)}$ para binário.



$525_{(10)} = 1000001101_{(2)}$

MSD – É o bit mais significativo, o quociente da última divisão.

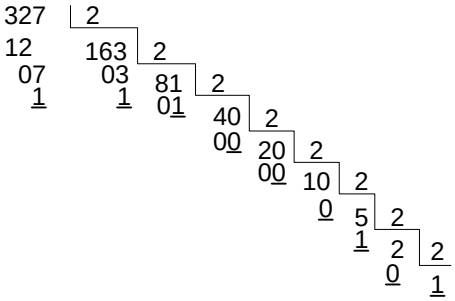
LSD – É o bit menos significativo, o resto da primeira divisão.

Se o número decimal tiver parte fraccionária, a parte inteira converte-se do mesmo modo que vimos anteriormente e a parte fraccionária multiplica-se por 2; a parte inteira deste produto é o algarismo mais significativo da parte fraccionária do número.

Se a parte fraccionária restante for de novo multiplicada por 2, a nova parte inteira será o algarismo mais significativo e assim sucessivamente.

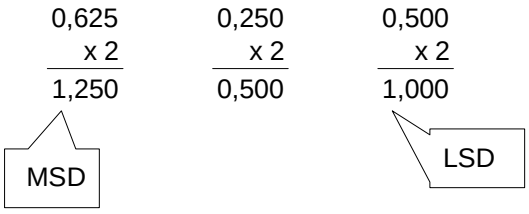
Exemplo: Converter o número $327{,}625_{(10)}$ em binário.

A parte inteira é $327_{(10)}$, que se converte assim:



$327_{(10)} = 101000111_{(2)}$

Para obter a parte fraccionária faremos:



$0,625_{(10)} = 0,101_{(2)}$

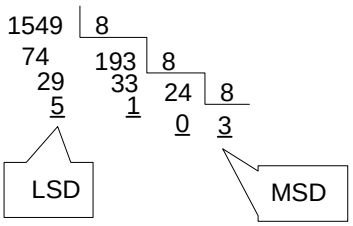
Ou seja: $327,625_{(10)} = 101000111,101_{(2)}$

CONVERSÃO DECIMAL OCTAL

Neste caso, procede-se da mesma forma que para o saco da conversão decimal binário, ressalvando-se o facto de estarmos a trabalhar agora com um sistema de base 8.

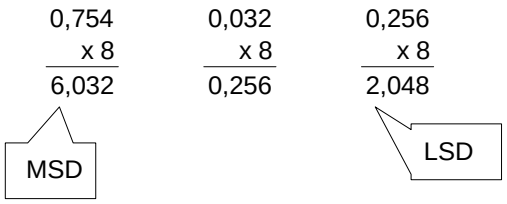
Exemplo: Converter $1549,754_{(10)}$ em octal.

1º Vamos converter a parte inteira:



$1549_{(10)} = 3015_{(8)}$

2º Vamos converter agora a parte fraccionária:



$0{,}754_{(10)} = 0{,}602_{(8)}$

Ou seja:

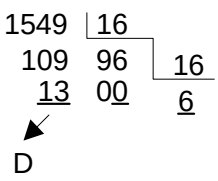
$1549{,}754_{(10)} = 3015{,}602_{(8)}$

CONVERSÃO DECIMAL HEXADECIMAL

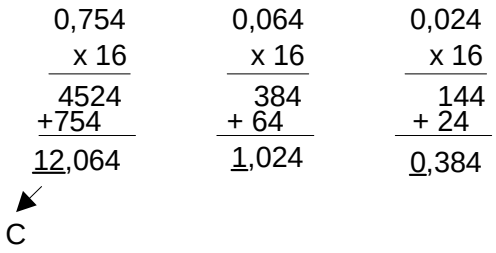
A parte fraccionária, quando a houver, é obtida através do método das multiplicações sucessivas, à semelhança do que se faz para as conversões de decimal para binário ou octal.

Exemplo: Converter $1549{,}754_{(10)}$ em hexadecimal.

Parte inteira:



Parte fraccionária:



Ou seja:

$1549{,}754_{(10)} = 60D{,}C10_{(16)}$

CONVERSÃO BINÁRIO OCTAL

Esta conversão baseia-se no princípio de que escrever cada dígito octal, são necessários três (3) dígitos binários (bits).

Tabela de correspondências	
Binário	Octal
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

Adicionando zeros à esquerda da parte inteira e à direita da parte fraccionária (não se alterando portanto o valor do número), transforma-se o nº de bits do número binário num múltiplo de 3.

Depois, agrupando-os 3 a 3 a partir do ponto binário e em ambos os sentidos, utiliza-se a tabela de correspondências, para obter o equivalente octal.

Exemplos:

a) $1101,11_{(2)}$ = 001 101, 110 $_{(2)}$ = $15,6_{(10)}$

1 5, 6

b) $25,7_{(8)}$ = 010 101, $111_{(2)}$

2 5, 7

CONVERSÃO BINÁRIO HEXADECIMAL

À semelhança da conversão binário/octal, esta conversão também se baseia no princípio de que, para escrever cada dígito hexadecimal, são necessários 4 dígitos binários (bits).

Tabela de correspondências	
Binário	Hexadecimal
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8

1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

Adicionando zeros à esquerda da parte inteira e à direita da parte fraccionária, transforma-se o nº de bits do número binário num múltiplo de 4.

Depois, agrupando-os 4 a 4, a partir do ponto binário e em ambos os sentidos, utiliza-se a tabela de correspondências, para obter o equivalente hexadecimal.

Exemplos:

a) $11101,11_{(2)}$ = 0001 1101, $1100_{(2)}$ = $1D,C_{(16)}$

1 D, C

b) $78,E_{(16)}$ = 0111 1000, $1110_{(2)}$

7 8, E

ARITMÉTICA BINÁRIA (SOMA)

A aritmética binária é fundamental em todos os computadores digitais e em muitos outros sistemas digitais.

As operações em binário efectuam-se de modo semelhante aos decimais, utilizando-se neste caso as tabuadas em binário.

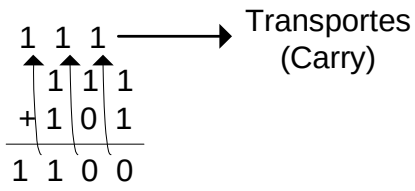
Tabuada da adição	
X + Y	Soma
0 + 0	0 e vai 0
0 + 1	1 e vai 0
1 + 0	1 e vai 0
1 + 1	0 e vai 1

Exemplo:

$7_{(10)} + 5_{(10)} = 12_{(10)}$

$7_{(10)} = 111_{(2)}$

$5_{(10)} = 101_{(2)}$ então:



Confirmando:

$$1100_{(2)} = 1 \times 2^3 + 1 \times 2^2 + 0 + 0 = 8 + 4 = 12_{(10)}$$

ARITMÉTICA BINÁRIA (SUBTRACÇÃO)

A operação subtracção em binário baseia-se também na tabuada que para ela foi criada:

Tabuada da subtracção	
X – Y	Subtracção
0 - 0	0 e vai 0
0 - 1	1 e vai 1
1 - 0	1 e vai 0
1 - 1	0 e vai 0

Exemplo:

$10100_{(2)} - 111_{(2)} = 01101_{(2)}$

$$\begin{array}{r} 1\ 0\ 1\ 0\ 0 \\ +\ 1\ 1\ 1 \\ \hline 0\ 1\ 1\ 0\ 1 \end{array}$$

$1\ 1\ 1\ 1$ → Transportes (Borrow)

Confirmação:

$10100_{(2)} = 16 + 4 = 20_{(10)}$

$111_{(2)} = 4 + 2 + 1 = 7_{(10)}$

$$20_{(10)} - 7_{(10)} = 13_{(10)}$$

$1101_{(2)} = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 8 + 4 + 0 + 1 = 13_{(10)}$

ARITMÉTICA BINÁRIA (MULTIPLICAÇÃO)

Tabuada da multiplicação	
X . Y	Multiplicação
0 x 0	0
0 x 1	0
1 x 0	0
1 x 1	1

Exemplo:

$110_{(2)} \times 101_{(2)} = 11110_{(2)}$

```
    1 1 0
  x 1 0 1
  -------
    1 1 0
  0 0 0
+ 1 1 0
  -------
  1 1 1 1 0
```

Confirmação:

$110_{(2)} = 6_{(10)}$

$101_{(2)} = 5_{(10)}$

$$6_{(10)} \times 5_{(10)} = 30_{(10)}$$

$11110_{(2)} = 1 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 16 + 8 + 4 + 2 = 30_{(10)}$

ARITMÉTICA BINÁRIA (DIVISÃO)

Para a divisão não existe tabuada pois na sua resolução utilizam-se apenas operações de multiplicação e subtracção.

Exemplo:

$11000_{(2)} \div 111_{(2)}$

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

$Q = 11_{(2)}$

$R = 11_{(2)}$

Confirmação:

$11000_{(2)} = 1 \times 2^4 + 1 \times 2^3 + 0 + 0 + 0 = 16 + 8 = 24_{(10)}$

$111_{(2)} = 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 4 + 2 + 1 = 7_{(10)}$

```
24 | 7
 3 | 3
```

$Q = 3_{(10)}$

$R = 3_{(10)}$

COMPLEMENTO A 2

A necessidade de representar, em binário, números relativos (+ ou -) e o benefício que seria para alguns dispositivos digitais se conseguíssemos transformar as subtracções em somas, levou à criação da representação em complemento a 2.

O que é o complemento a 2 de um número binário?

O complemento a 2 é uma forma de representar números binários negativos.

Quando escrevemos uma quantidade em algarismos, podemos simplesmente colocar atrás do valor o sinal + ou -, para indicar respectivamente se o número é positivo ou negativo.

No entanto, num circuito digital, apenas existem 0's e 1's, por isso, tem que haver um modo de representar o sinal.

Estabeleceu-se então, que o dígito de maior peso (MSD) indica o sinal de um número.

Se esse bit for "1", então o número é negativo, se for "0", o número será positivo.

Quando usamos bit de sinal, a capacidade de representação é diferente de quando não usamos bit de sinal.

Exemplo:

Capacidade de representação com 3 bits			
Sem bit de sinal		Com bit de sinal	
Binário	Decimal	Binário	Decimal
000	0	011	+3
001	1	010	+2
010	2	001	+1
011	3	000	0
100	4	111	-1

101	5	110	-2
110	6	101	-3
111	7	100	-4

A capacidade de representação com sinal para n bits determina-se através do seguinte intervalo:

$$\left[-\left(2^{n-1}\right); 2^{n-1}-1\right]$$

Exemplo:

Se o número de bits que estamos a usar é 3, como poderá ser efectuada uma soma sem exceder a capacidade de representação?

$$\left[-\left(2^{3-1}\right); 2^{3-1}-1\right] = [-4;3]$$

Então com 3 bits podemos representar números desde -4 até +3.

A capacidade de representação sem bit de sinal determina-se do seguinte modo:

$$\text{Número máximo} = 2^{n}-1$$

Exemplo:

Se o número de bits que estamos a usar é 3, qual o valor máximo em decimal que podemos representar sem exceder a capacidade de representação?

$$\text{Número máximo} = 2^{3}-1 = 7$$

O que acontece quando excedemos a capacidade de representação?

Exemplos com 4 bits:

a) $+5_{(10)} + 6_{(10)} = 11_{(10)}$ (Com bit sinal)

$+5_{(10)} = 0101_{(2)}$

$+6_{(10)} = 0110_{(2)}$

$$\begin{array}{r} 0101 \\ +\ 0110 \\ \hline 1011_{(2)} \end{array} = -5_{(10)}$$

Como se pode verificar, o resultado do exercício está errado porque com 4 bits a soma não pode dar um valor que esteja fora do intervalo:

$$\left[-\left(2^{4-1}\right); 2^{4-1}-1\right] = \left[-8; +7\right]$$

b) $8_{(10)} + 8_{(10)} = 16_{(10)}$ (Sem bit sinal)

$$\begin{array}{r} 1000 \\ +\ 1000 \\ \hline 1\ 0000_{(2)} = 0 \end{array}$$

Overflow

Neste caso o resultado está incorrecto porque o resultado ultrapassou a capacidade de representação com 4 bits: $2^4 - 1 = 15$

c) $+7_{(10)} + 7_{(10)} = 14_{(10)}$ (Com bit sinal)

$+7_{(10)} = 0111_{(2)}$

$$\begin{array}{r} 0111 \\ +\ 0111 \\ \hline 1110_{(2)} = -2_{(10)} \end{array}$$

Como se pode verificar, o MSD é 1, logo a resultado é negativo e portanto a soma está errada.

Como se forma o complemento a 2 de um número?

1º - Invertem-se todos os bits do número.

2º - Soam-se "1" ao resultado da inversão.

Exemplo:

a) $+2_{(10)} = 0010_{(2)}$

Para representar o número $-2_{(10)}$, começamos por representar a quantidade indicada como um número positivo:

$$0010_{(2)}$$

Invertemos todos os bits, ficando:

$$1101_{(2)}$$

Somamos 1 ao resultado da inversão:

$$\begin{array}{r} 1101 \\ +1 \\ \hline 1110_{(2)} \end{array} = -2_{(10)}$$

Nota: o MSB indica que o número é negativo.

b) $+7_{(10)} = 0111_{(2)}$

Para representar o número $-7_{(10)}$, começamos por representar a quantidade indicada como um número positivo:

$$0111_{(2)}$$

Invertemos todos os bits, ficando:

$$1000_{(2)}$$

Somamos 1 ao resultado da inversão:

$$\begin{array}{r} 1000 \\ +1 \\ \hline 1001_{(2)} \end{array} = -7_{(10)}$$

Nota: o MSB indica que o número é negativo.

c) $+17_{(10)} = 010001_{(2)}$

Para representar o número $-17_{(10)}$, começamos por representar a quantidade indicada como um número positivo:

$$010001_{(2)}$$

Invertemos todos os bits, ficando:

$$101110_{(2)}$$

Somamos 1 ao resultado da inversão:

$$\begin{array}{r} 101110 \\ +1 \\ \hline 101111_{(2)} \end{array} = -17_{(10)}$$

Nota: o MSB indica que o número é negativo.

SUBTRACÇÃO COM O MÉTODO COMPLEMENTO A 2

$$X - (+Y) = X + (-Y)$$

Método: Calcula-se o complemento a 2 do subtractivo e transforma-se a subtracção numa soma.

Exemplos:

a) $13_{(10)} - 9_{(10)} = 4_{(10)} \rightarrow 13_{(10)} + (-9)_{(10)} = 4_{(10)}$

1º passo: começamos por calcular o complemento a 2 de $+9_{(10)}$

$+9_{(10)}$ = Complemento a 1: $01001_{(2)}$

Soma-se 1 : $10110 + 1$

Resultado: $10111_{(2)} = -9_{(10)}$

2º passo: Somam-se as duas parcelas, desprezando os transportes para além do MSD:

Nota: Despreza-se o "1" do transporte porque estamos a subtrair números com 5 dígitos, o resultado nunca poderá aparecer na forma de 6 bits.

O MSD passou a ser o "0", querendo isto dizer que se trata de um número positivo.

$+9_{(10)}$ = $01101_{(2)}$

$$\begin{array}{r} 01101 \\ +10111 \\ \hline 1\ 00100_{(2)} = +4_{(10)} \end{array}$$

MSD

Este bit é desprezado

b) $9_{(10)} - 13_{(10)} = 4_{(10)} \rightarrow 9_{(10)} + (-13)_{(10)} = -4_{(10)}$

1º passo: começamos por calcular o complemento a 2 de $+13_{(10)}$

$+13_{(10)} =$ $01101_{(2)}$

Complemento a 1: 10010

Soma-se 1 : + 1

Resultado: $10011_{(2)} = -13_{(10)}$

2º passo: Somam-se as duas parcelas:

$01001 \longrightarrow 9_{(10)}$

$+10011$

$11100_{(2)} = -4_{(10)}$

Bit sinal

Nota: O bit de sinal é "1", logo o resultado é um número negativo e está representado em complemento a 2.

Como fazer para saber qual o valor desse número?

Calculamos o seu complemento a 2:

$-4_{(10)} =$ $11100_{(2)}$

Complemento a 1: 00011

Soma-se 1 : + 1

Resultado: $00100_{(2)} = +4_{(10)}$

OPERAÇÕES LÓGICAS ELEMENTARES

As operações lógicas elementares têm semelhança com operações aritméticas comuns, inclusive alguns símbolos são idênticos, mas não são necessariamente coincidentes:

1) Operação OR

É similar à adição comum, mas a correspondência não é plena. Símbolo usual é o mesmo da adição.

Exemplo: X = A + B (lê-se X igual a A **ou** B). Um outro símbolo, comum em linguagem de programação, é a barra vertical (X = A | B).

2) Operação AND

É similar à multiplicação comum e há correspondência, como poderá ser visto adiante. Símbolo usual é o mesmo da multiplicação.

Exemplo: X = A . B (lê-se X igual a A **e** B). Muitas vezes, também de forma semelhante à álgebra comum, o sinal de ponto é suprimido: X = AB. O *e comercial* (&) é um símbolo usado em algumas linguagens (X = A & B).

3) Operação NOT

Também denominada negação ou complemento, pode ser considerada similar ao negativo da álgebra comum. Entretanto, não há correspondência plena porque a álgebra de Boole não usa sinal negativo. Símbolo usual é uma barra acima (ou antes) da variável.

Exemplo: X= $\overline{A}$ (lê-se X igual a **não** A). Alguns outros símbolos são o sinal de exclamação (X = !A) e o apóstrofo (X = A').

PORTAS LÓGICAS ELEMENTARES

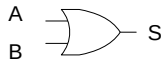
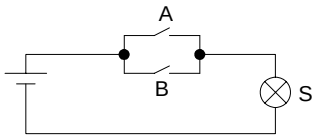
Portas lógicas são dispositivos práticos que executam funções booleanas básicas, isto é, as operações fundamentais OR, AND, NOT e algumas delas derivadas. Na actualidade, a sua implementação é quase sempre em circuitos electrónicos integrados, mas podem ser componentes discretos, circuitos eléctricos com relés, dispositivos ópticos, circuitos hidráulicos ou mesmo mecanismos.

Considerando circuitos eléctricos ou electrónicos, deve-se notar que os valores lógicos 0 e 1 são representados por tensões ou correntes, normalmente em determinadas faixas. Entretanto, na análise lógica, esse dado não é levado em conta e os valores de entradas e saídas são sempre referidos a 0 ou a 1.

Porta OR

Nesta porta, a saída S é igual à operação booleana OR entre os valores das entradas. Na figura abaixo, está representado o símbolo usual e, a tabela de verdade da função.

A função booleana (ou lógica) é S = A + B.

Símbolo	Tabela de Verdade			Circuito equivalente
	A	B	S	
	0	0	0	
	0	1	1	
	1	0	1	
	1	1	1	

A figura mostra um circuito eléctrico simples com dois interruptores e uma lâmpada. Neste caso, o interruptor desligado é nível lógico 0 e interruptor ligado é o nível lógico 1. A lâmpada liga quando pelo menos um dos interruptores está ligado. Portanto, o circuito opera conforme a tabela de verdade ao lado.

Postulados da operação OR

$X + 0 = X$

$X + 1 = 1$

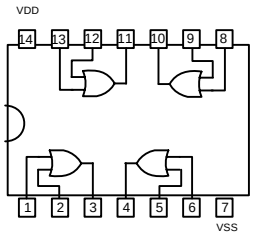
$X + X = X$

$X + \overline{X} = 1$

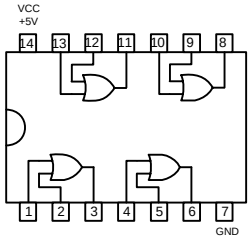
A	B	S	Postulados	
0	0	0	A + 0 = A	A+ A = A
0	1	1	A + 1 = 1	$A + \overline{A} = 1$
1	0	1	A + 0 = A	$A + \overline{A} = 1$
1	1	1	A + 1 = 1	A + A = A

As figuras ao lado dão a identificação dos pinos do circuito integrado 4071 (CMOS) e 7432 (TTL). Cada circuito integrado é constituído por 4 portas OR. O pino 7 é ligado à massa (Ground) e o pino 14 é ligado a 5 Volts.

4071

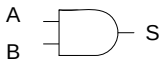
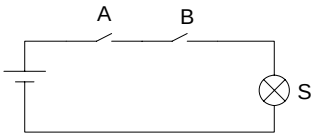


7432



Porta AND

A saída S é igual à operação booleana E entre os valores das entradas. Símbolo usual conforme a figura e a tabela de verdade em baixo. A figura mostra um circuito simples com interruptores. Agora, os contactos estão em série e a saída só terá nível 1 quando todas as entradas forem também 1. A função S = A . B

Símbolo	Tabela de Verdade			Circuito equivalente
	A	B	S	
	0	0	0	
	0	1	0	
	1	0	0	
	1	1	1	

Postulados da operação AND

$X \cdot 0 = 0$

$X \cdot 1 = X$

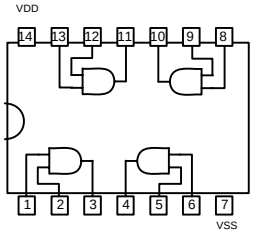
$X \cdot X = X$

$X \cdot \overline{X} = 0$

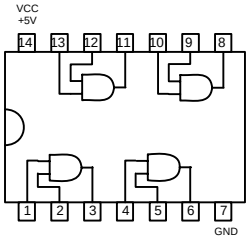
A	B	S	Postulados	
0	0	0	$A \cdot 0 = 0$	$A \cdot A = A$
0	1	0	$A \cdot 1 = A$	$A \cdot \overline{A} = 0$
1	0	0	$A \cdot 0 = 0$	$A \cdot \overline{A} = 0$
1	1	1	$A \cdot 1 = A$	$A \cdot A = A$

As figuras ao lado dão a identificação dos pinos do circuito integrado 4081 (CMOS) e 7408 (TTL). Cada circuito integrado é constituído por 4 portas AND. O pino 7 é ligado à massa (Ground) e o pino 14 é ligado a 5 Volts.

4081

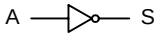


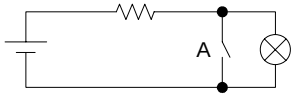
7408



Porta NOT

Na porta NOT, a saída S está invertida relativamente à entrada A. Na Figura abaixo podemos ver o símbolo usual, a tabela de verdade e circuito eléctrico simples para a função.

Símbolo	Tabela de verdade	
	A	S
	0	1
	1	0

Circuito equivalente


A função lógica é $S = \overline{A}$. A porta NOT é também denominada **inversor**. Para simplificar os diagramas, o símbolo é apenas um pequeno círculo se estiver junto de uma entrada ou saída de outra porta lógica.

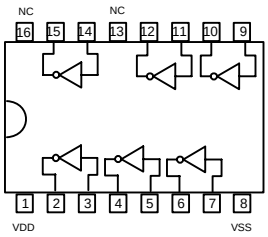
Postulados da operação NOT

$X = \overline{X}$

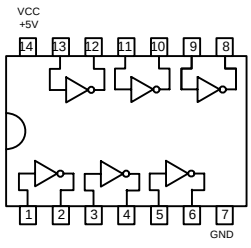
$X = \overline{\overline{X}}$

As figuras ao lado dão a identificação dos pinos do circuito integrado 4049 (CMOS) e 7404 (TTL). Cada circuito integrado é constituído por 6 portas NOT.

4049

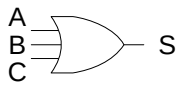
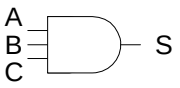


7404



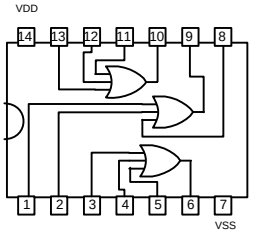
Portas com mais de duas entradas

Em razão da operação que executa, a porta NÃO admite apenas uma entrada. As portas OR e AND (e outras delas derivadas) podem ter qualquer número n ≥ 2 de entradas.

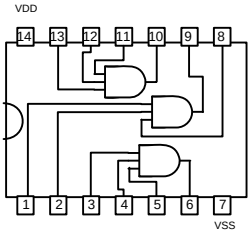
OR de 3 entradas					AND de 3 entradas				
Símbolo	A	B	C	S	Símbolo	A	B	C	S
	0	0	0	0		0	0	0	0
	0	0	1	1		0	0	1	0
	0	1	0	1		0	1	0	0
	0	1	1	1		0	1	1	0
	1	0	0	1		1	0	0	0
	1	0	1	1		1	0	1	0
	1	1	0	1		1	1	0	0
	1	1	1	1		1	1	1	1

Na figura acima, pode-se ver o símbolo e a tabela de verdade para a porta OR de 3 entradas, S = A + B + C, e ao lado a porta AND de três entradas, S = A . B . C.

4075



4073



Porta NOR

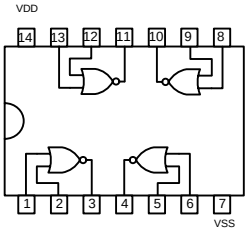
É uma porta OR com um inversor (NOT) na saída, que, nos diagramas, pode ser representado por um pequeno círculo conforme já comentado.

Expressão lógica segundo álgebra de Boole: S = $\overline{A+B}$

Devido à acção do inversor, os resultados são complementares aos da porta OR.

NOR			
Símbolo	A	B	S
	0	0	1
	0	1	0
	1	0	0
	1	1	0

4001



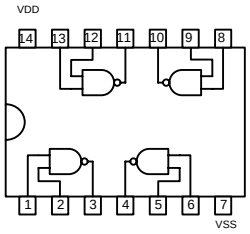
Porta NAND

De forma similar à anterior, apresenta resultados complementares aos da porta E devido ao inversor na saída. Símbolo usual e tabela de verdade para duas entradas nas Figuras 01-c e 01-d deste tópico.

Função lógica: S = $\overline{A.B}$

NAND			
Símbolo	A	B	S
	0	0	1
	0	1	1
	1	0	1
	1	1	0

4011



ARQUITECTURA BÁSICA DOS COMPUTADORES

TERMINOLOGIA

- Bit

Um digito binário pode representar 0 ou 1 e é normalmente chamado de bit.

- Byte

Um conjunto de oito bits é chamado de Byte

- Software

Refere-se aos programas escritos para computador

- Hardware

É o nome dado aos dispositivos físicos e circuitos do computador

- CPU (Central Processing Unit)

O CPU controla as operações de um computador. O CPU lê as instruções que estão na memória, descodifica-as numa série de acções, e leva a cabo estas acções numa série de passos.

O CPU também contem o contador de endereços ou o registo de ponteiro de instrução, que contém o endereço da próxima instrução que vai ser lida da memória. Também tem registos de uso geral para armazenar dados em binário; e todo o circuito de controlo que gera os sinais para os barramentos de endereço e dados.

- IC (Integrated Circuits)

Nome dado a um circuito integrado. Componente constituído por transístores e outros componentes com uma determinada função.

- Bus de endereços

Este bus consiste em 16, 20, 24 ou 32 linhas de sinal em paralelo. Nestas linhas o CPU envia o endereço de memória onde se vai ler ou escrever.

O número de endereços que o CPU pode endereçar é determinado pelo número de linhas de endereço. Se o CPU tem n linhas de endereço, então pode endereçar 2^n posições de memória.

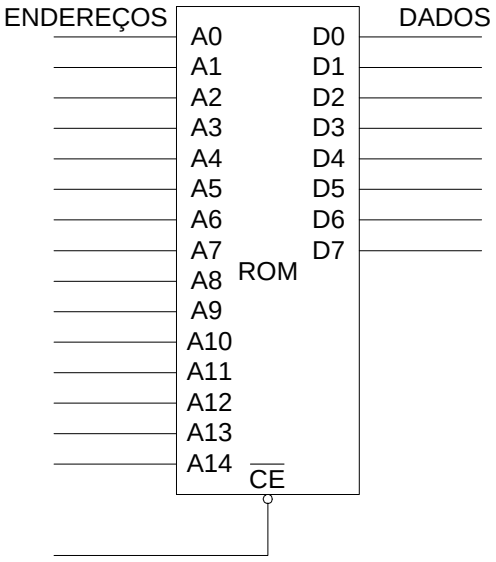
- Bus de dados

O bus de dados consiste em 8, 16, 32, ou 64 linhas de sinal em paralelo. Através destas linhas o CPU consegue ler ou escrever dados da memória ou de uma entrada/saída. Há muitos dispositivos ligados ao bus de dados, mas apenas uma das suas saída está activada de cada vez.

TIPOS DE MEMÓRIAS (ROM)

O termo ROM significa Read Only Memory (memória somente de leitura). Existem vários tipos de ROM que podem ser programadas, lidas, apagadas e programadas com novos dados, mas a principal característica da ROM é que não é volátil. Isto significa que os dados podem ser armazenados e não se perdem quando a energia é removida da memória.

A figura mostra o símbolo esquemático de uma ROM. De acordo com o BUS de dados D0 a D7, esta ROM pode armazenar em cada endereço 8 bits de dados. As saídas de dados são TRI-STATE. Isto significa que cada saída pode ter um nível lógico 0, um nível lógico 1, ou uma alta impedância. No estado de alta impedância a saída é desligada do que estiver ligado a ela. Se a entrada $\overline{CE}$ da ROM não estiver accionada, então todas as saídas de dados estarão em alta impedância. Algumas ROMs também ficam em modo de baixo consumo quando a entrada $\overline{CE}$ não esta accionada. Se a entrada $\overline{CE}$ é accionada, a ROM é activada, e as saídas também serão activadas. Assim as saídas ficarão a um nível lógico normal 0 ou 1.



Todos os dados na ROM ficam armazenados em forma de uma lista numerada. O número que identifica a localização de cada palavra armazenada na lista é chamado endereço. Podemos dizer o número de palavras armazenadas na ROM pelo número de entradas de endereços. O número de palavras é igual a 2N, N é igual ao número de linhas de endereço. A ROM da figura anterior tem 15 linhas de endereço, A0 até A14, assim o número de palavras é 215 ou 32 768. A data sheet refere-se a este dispositivo como 32K x 8 ROM. Isto significa que tem 32K de endereços com 8 bits por endereço.

Para obter uma palavra na saída da ROM, tem que fazer duas coisas. Tem que aplicar o endereço nas entradas de endereços, A0 a A14, e accionar a entrada $\overline{CE}$.

Agora vamos ver para que são necessárias as saídas TRI-STATE numa ROM. Vamos supor que queremos guardar mais que 32K de dados. Nós podemos fazer isto conectando duas ou mais ROMs em paralelo, de modo que podemos endereçar um dos 32 768 endereços em cada uma. O conjunto de linhas paralelas usado para enviar o endereço ou dados é chamado BUS. As saídas de dados das ROMs são ligados em paralelo de modo que qualquer uma das ROMs possa enviar os dados para um BUS de dados comum. Se estas ROMs tiverem apenas as saídas normais com dois estados, irá ocorrer um problema sério quando ambas as ROMs tentarem enviar dados para o BUS. Esta ligação entre as saídas provavelmente irá destruir algumas saídas e fornecer informação sem qualquer significado. Uma vez que as ROMs têm saídas com TRI-STATE, nós podemos usar circuitos externos para assegurar que apenas uma ROM de cada vez seja activada. Há um princípio importante aqui, sempre que houver varias saídas ligadas ao BUS, as saídas devem ser TRI-STATE, e apenas uma ROM deve ser activada de cada vez.

No início deste capítulo foi mencionado que algumas ROMs podem ser apagadas e reprogramadas com novos dados. Em seguida estão descritos alguns tipos de ROMs.

ROM – Programada durante o fabrico.

PROM – Programada pelo utilizador colocando os 1s a 0s. Não pode ser alterada excepto para colocar outros 1s a 0s.

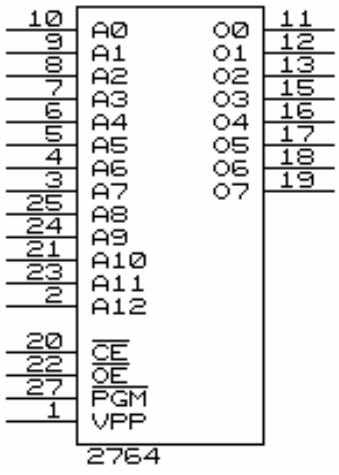
EPROM – Programada electricamente pelo utilizador colocando os 1s a 0s; pode ser apagada com luz ultravioleta através da janela de quartzo.

EEPROM – Programada electricamente pelo utilizador; pode ser apagada com sinais eléctricos de modo a ser reprogramada no circuito.

O $\overline{OE}$ (Output Enable) é colocado a 0 sempre que se deseja ler algo da RAM.

E(PROM) (2764)

Esta memória utiliza 13 linhas de endereço (A0 a A12), permitindo assim o acesso a 8KBytes de endereços, e 8 linhas de dados (D0 a D7).



Existem 4 terminais de controlo ($\overline{CE}, \overline{OE}, \overline{PGM}, VPP$), o $\overline{CE}$ (Chip Enable) deve estar a 0 para a memória ser seleccionada.

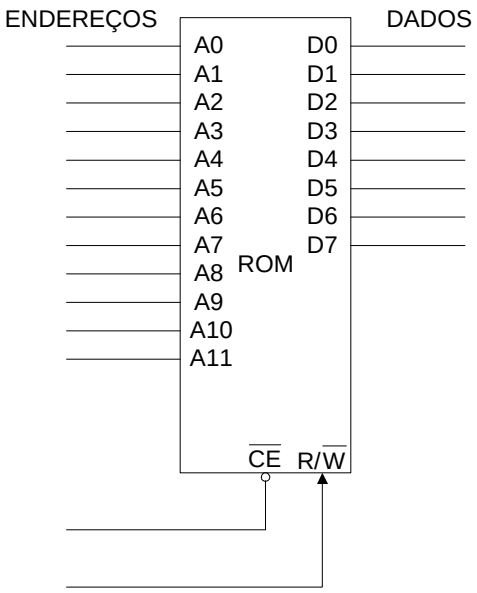
O $\overline{OE}$ (Output Enable) é colocado a 0 sempre que se deseja ler algo da ROM.

No $\overline{PGM}$ (ProGraM) coloca-se um 0 e no VPP coloca-se a tensão de programação sempre que se deseja programar uma nova E(PROM).

TIPOS DE MEMÓRIAS (RAM)

O nome RAM significa Random Access Memory (memória de acesso aleatório), mas uma vez que as ROMs também são de acesso aleatório, o nome deveria ser memória de escrita e leitura. As RAMs são usadas também para armazenar dados em binário. A RAM estática é essencialmente uma matriz de flip-flops. Assim, nós podemos dados num endereço da RAM em qualquer altura aplicando os dados nas entradas de dados e accionando os flip-flops. A informação guardada nos flip-flops mantém-se enquanto a energia estiver aplicada na RAM. Esta memória é volátil porque os dados perdem-se quando a energia é desligada.

A figura seguinte mostra o símbolo esquemático de uma RAM comum. Esta RAM tem 12 linhas de endereço, A0 até A11, por isso pode armazenar 2^{12} (4096) palavras de dados. As oito linhas de dados indicam que a RAM pode armazenar 8 bits em cada endereço. Quando estamos a escrever na RAM, estas linhas funcionam como entradas. A entrada $\overline{CE}$, é usada para activar a RAM para escrita ou leitura. A entrada $R/\overline{W}$ é colocada a lógico 1 para ler da RAM e colocada a lógico 0 para escrever na RAM. Aqui está como todas estas linhas funcionam para ler e escrever neste dispositivo.

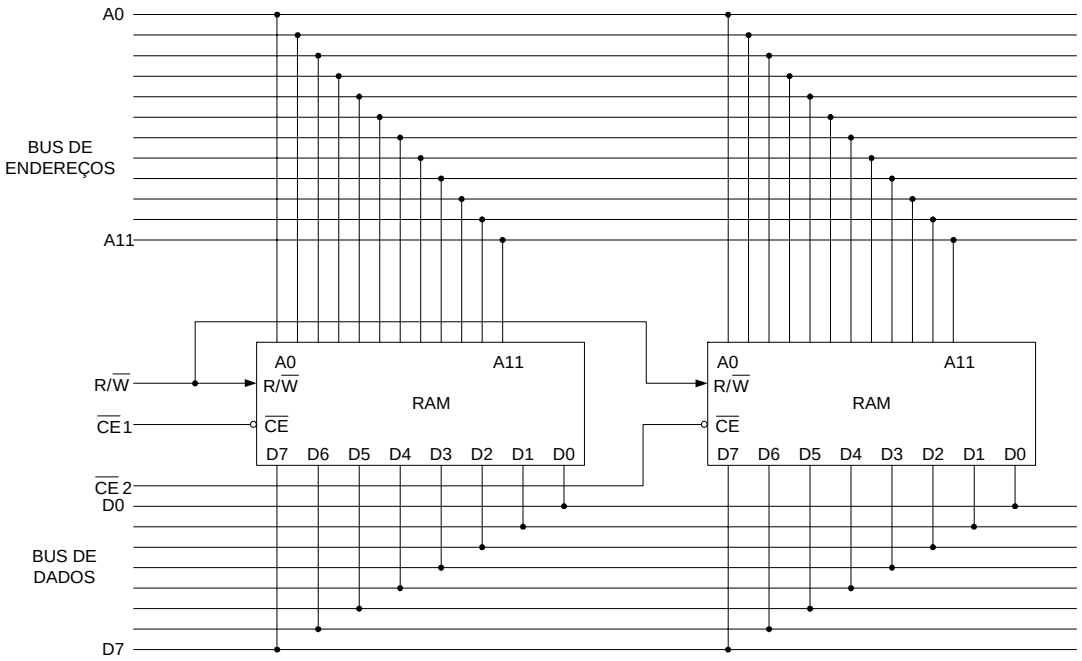


Para escrever na RAM, aplicamos o endereço desejado nas entradas de endereço, colocamos a entrada $\overline{CE}$ a lógico 0 para accionar o dispositivo, e colocamos a entrada $R/\overline{W}$ a lógico 0 para indicar à RAM que queremos escrever nela. Aplicamos então os dados que queremos escrever nas linhas de dados durante um tempo específico. Para ler informação da RAM, indicamos o endereço desejado, colocamos a entrada $\overline{CE}$ a lógico 0, e colocamos a entrada $R/\overline{W}$ a lógico 1 para indicar à RAM que queremos ler. Para uma operação de leitura as saídas de dados estarão activas para fornecer os dados que estão no endereço indicado pelas

linhas de endereço.

As memórias estáticas SRAM que acabamos de ver guardam os dados numa matriz de flip-flops. Nas memórias dinâmicas DRAM, os dados binários 0s e 1s são guardados numa carga eléctrica num condensador. Uma vez que estes condensadores ocupam menos espaço que um flip-flop, uma RAM dinâmica pode armazenar muito mais bits que uma memória estática do mesmo tamanho. A desvantagem das memórias dinâmicas é que os condensadores descarregam-se. O estado lógico armazenado em cada condensador deve ser refrescado todos os 2 milisegundos.

A memória RAM pode ser ligada em paralelo do mesmo modo que a ROM. Na figura seguinte está representa a ligação de duas RAMs em paralelo. O funcionamento do BUS é igual ao esquema com ROMs. Neste esquema com RAMs existe a entrada $R/\overline{W}$ que é comum a todas as RAMs que estiverem ligadas em paralelo. Apenas uma RAM pode ser seleccionada através da entrada $\overline{CE}$.

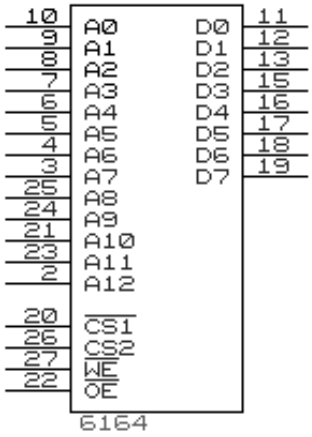


RAM (6164)

Esta memória utiliza 13 linhas de endereço (A0 a A12), permitindo assim o acesso a 8KBytes de endereços, e 8 linhas de dados (D0 a D7).

Existem 4 terminais de controlo ($\overline{CS1}, \overline{CS2}, \overline{WE}, \overline{OE}$), estes 4 terminais são activos a 0, o $\overline{CS1}$ e $\overline{CS2}$ (Chip Select) devem estar ambos a 0 para a memória ser seleccionada.

$\overline{WE}$ (Write Enable) é colocado a 0 sempre que se deseja escrever algo na RAM.



TECNOLOGIA USADA EM COMPUTADORES

O PC rapidamente precisou de vários melhoramentos, para acompanhar o rápido desenvolvimento do software, e por isso precisou de mais capacidade na RAM e no disco. Esta necessidade foi ultrapassada com a introdução do PC XT com os seguintes melhoramentos:

Foi aumentado o número de slots para 8

Foram acrescentados um disco e um adaptador

Foram juntos dois port's, um paralelo e um série

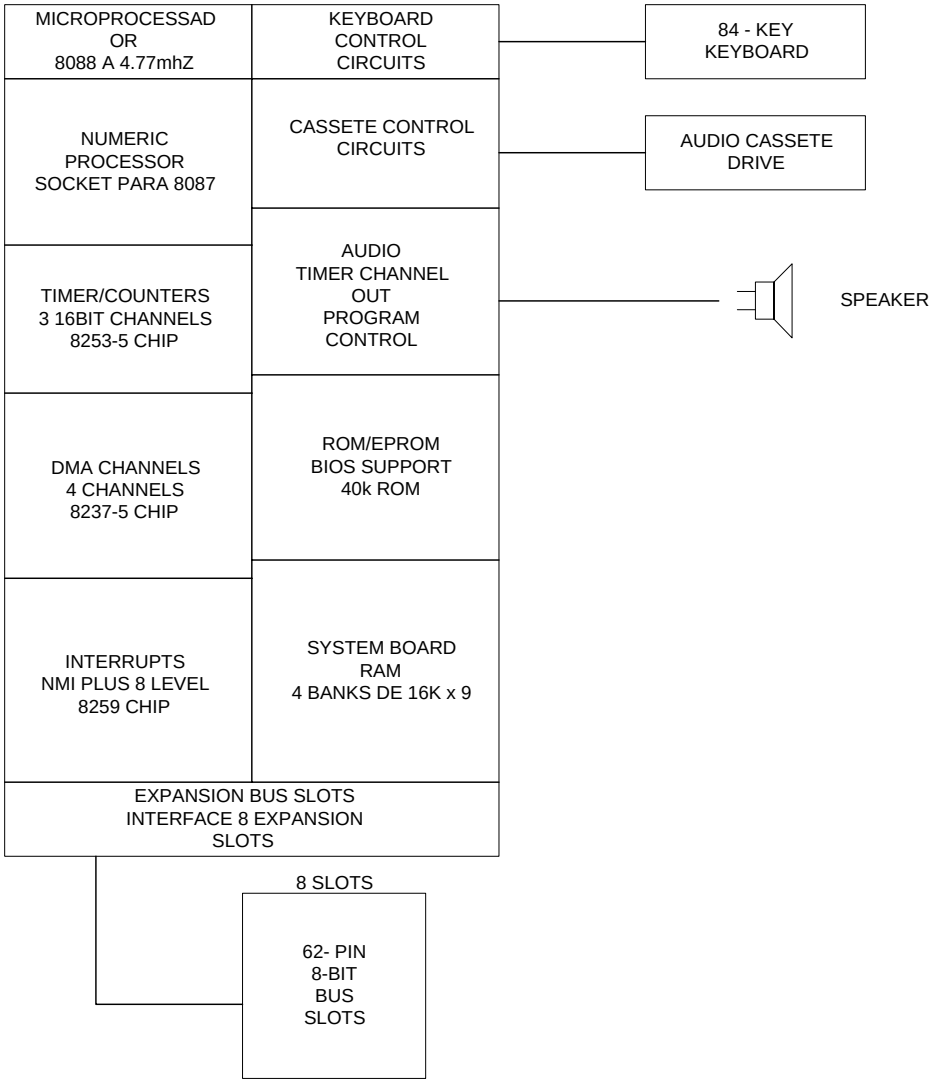
A memória RAM foi aumentada para 256K

A potência da fonte de alimentação foi aumentada de 65 para 135W

A capacidade das disquetes foi aumentada para 360K

O tamanho físico da caixa foi mantido igual ao IBM PC.

O PC XT não aumentou a velocidade do microprocessador, entretanto outros fabricantes aumentaram a performance do sistema, aumentando a velocidade do clock para 8 e até mesmo 10MHz. Alguns fabricantes mudaram mesmo o microprocessador para o 8086. Uma memória de 16 bits versus uma de 8 bits, resulta em um aumento de performance.

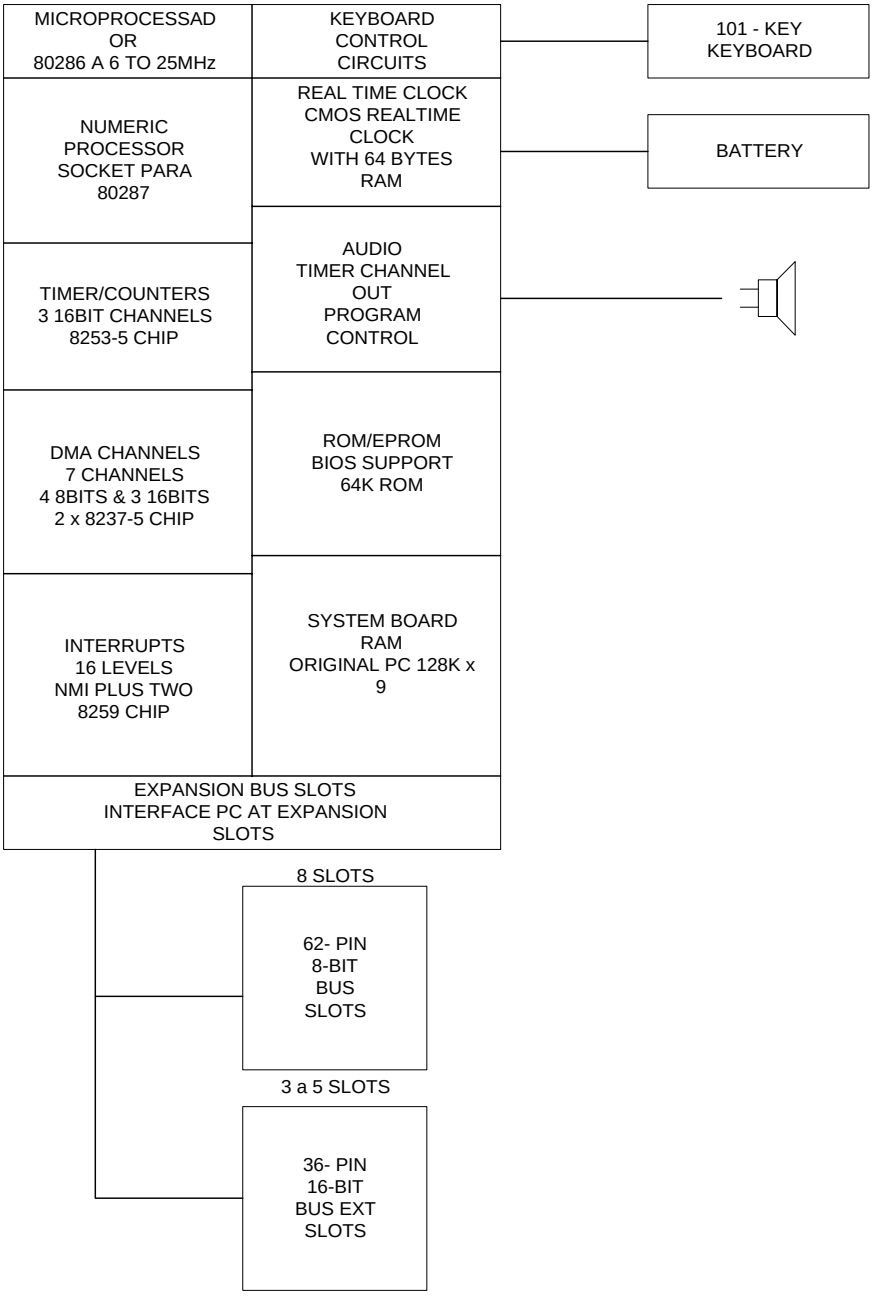


O PC XT

A introdução do PC AT marcou o uso da nova geração de microprocessadores da Intel 80286. Para acomodar este microprocessador de 16 bits de dados e 24 bits de endereços, o AT estendeu o BUS do sistema. Era necessário entretanto manter a compatibilidade com o sistema anterior, por isso as slots de expansão do AT continham as anteriores do PC XT.

O 80286 ofereceu uma melhor performance, com uma maior frequência de clock, um menor número de clocks por instrução, novas instruções, e um BUS de 16bits. O 80286 suporta dois modos de operação: modo real e protegido. O modo real emula o 8088/8086 com um ambiente de 1Mbyte de RAM, o que permite que o software do 8088/8086 seja executa sem necessitar de alterações. No modo protegido, pode trabalhar num ambiente de até 16Mbytes de RAM e em ambiente multitarefa.

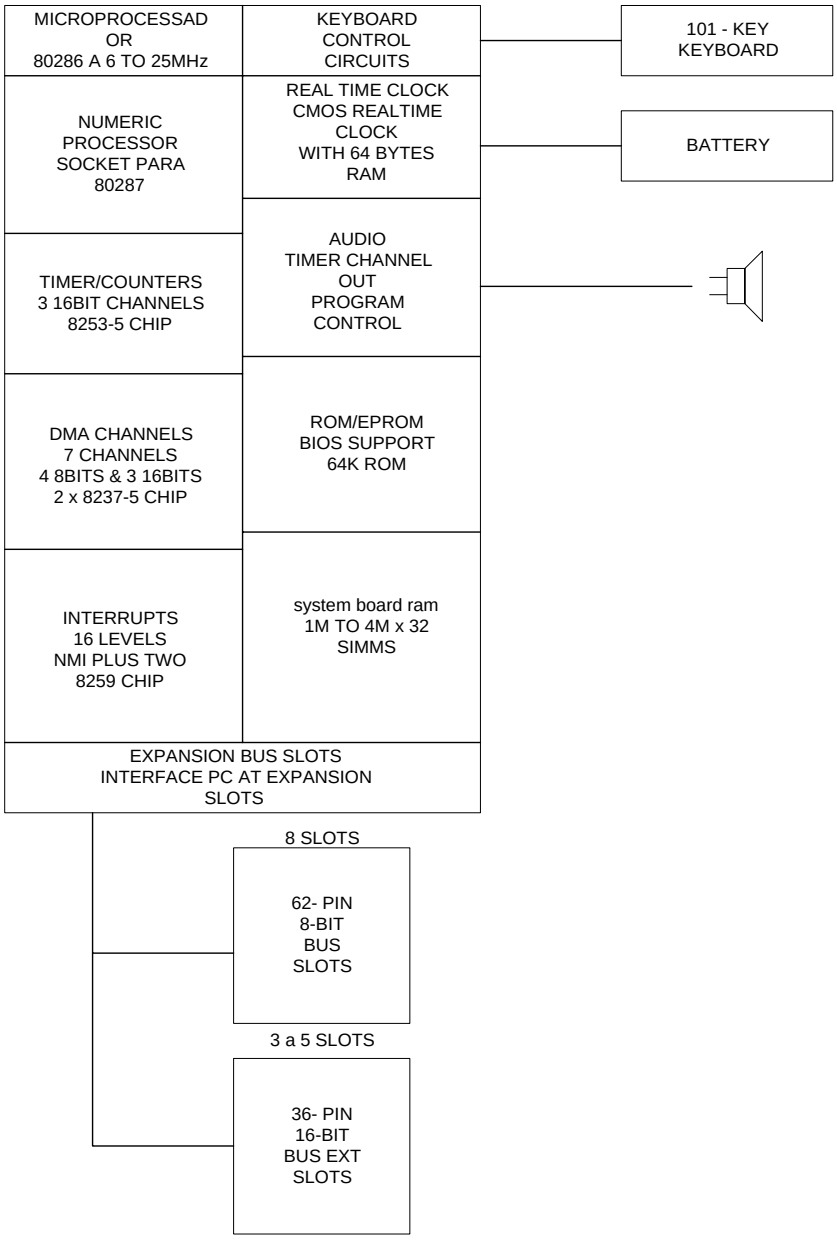
Com a introdução do 286 e do BUS AT, foi introduzido o floppy de 1.2M (disquete flexível). O AT juntou também um relógio com calendário alimentado por uma pilha, e o sistema de vídeo EGA (Enhanced Graphics Adapter).



O PC AT

O passo seguinte foi a introdução do microprocessador de 32 bits da Intel 80386 com a respectiva evolução até aos processadores actuais. A IBM introduziu o 80386 na família PS/2, o resto da indústria manteve o BUS AT a trabalhar com o 80386.

O sistema com o 80386, tem um BUS de dados e de endereços de 32bits.

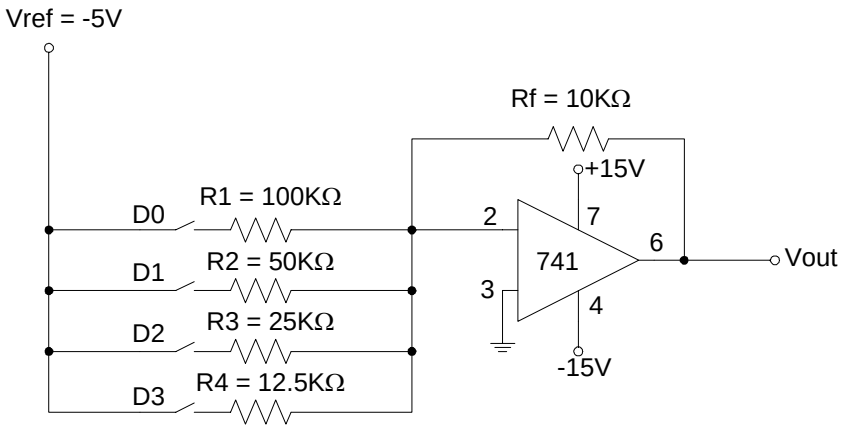


386 PC AT

CONVERSÃO DE DADOS

CONVERSORES DIGITAIS ANALÓGICOS

A função do conversor digital analógico é converter uma palavra em binário para uma tensão ou corrente proporcional ao seu valor. Para vermos como isto é feito vamos analisar o circuito da figura seguinte.



Uma vez que a entrada não inversora está ligada à massa, o amplificador vai colocar uma tensão no pino 6 de modo que a tensão na entrada inversora seja sempre igual a 0V. Quando um dos interruptores é ligado, a corrente vai fluir dos –5V através da resistência até à entrada não inversora. Para que a tensão nesta entrada seja 0V é necessário que esta corrente flua toda pela resistência Rf.

Se por exemplo fechar o interruptor D0, uma corrente de 0.05mA vai circular pela resistência R1. Para que esta corrente flua toda pela resistência Rf, é necessário por uma tensão de 0.05mA x 10KΩ ou 0.5V na saída do amplificador. Se fecha também o interruptor D1, vai circular mais 0.1mA. Para que as duas correntes (0.05mA + 0.1mA) circulem pela resistência Rf, o amplificador tem que ter a saída uma tensão igual a 0.15mA x 10KΩ ou 1.5V.

As resistências produzem uma corrente com um peso igual ao peso em binário do respectivo interruptor, são somadas pelo amplificador para produzir uma tensão que é proporcional. A palavra em binário colocada nos interruptores produz uma tensão à saída proporcional. Tecnicamente a tensão à saída é digital porque só pode ter alguns valores fixos, tal como o display de um voltímetro digital. Entretanto, a saída simula um sinal analógico, por isso dizemos que é analógico.

O interruptor D3 representa o bit mais significativo porque quando é accionado produz a maior corrente. Note que Vref é negativo, mas a saída é positiva.

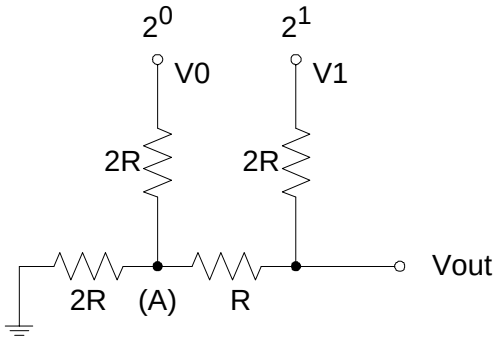
CONVERSOR DIGITAL ANALÓGICO R-2R

Na figura seguinte podemos ver um exemplo de um conversor digital analógico de 2 bits que utiliza uma malha de resistências.

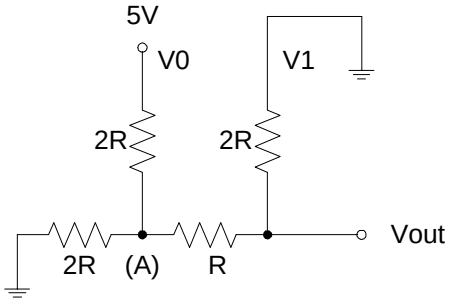
Vamos fazer uma análise deste circuito com auxílio da tabela de verdade.

V1	V0	Vout
0V	0V	0V
0V	5V	1.25V
5V	0V	2.5V
5V	5V	3.75V

Na primeira condição, quando ambas as entradas estão ligadas a 0V, não existe corrente a circular no circuito e por isso não há queda de tensão. Vout é igual a 0V.



Vamos agora analisar a segunda condição, V1 = 0V e V0 = 5V. Em primeiro lugar vamos calcular a tensão no ponto (A).



- Calcular a resistência equivalente entre o ponto (A) e a massa:

$$R(A) = \frac{2R \text{ x } 3R}{(2+3)R} = \frac{6}{5}R$$

- Calcular a tensão no ponto (A):

$$V(A) = 5V \text{ x } \frac{{}^{6}\!/_{5}R}{2R + {}^{6}\!/_{5}R} = 5V \text{ x } \frac{6}{16} = 1.875V$$

Podemos agora calcular Vout do mesmo modo que foi calculado V(A):

$$Vout = 1.875V \text{ x } \frac{2R}{3R} = 1.875V \text{ x } \frac{2}{3} = 1.25V \;\; ; \;\; Vout = 5V \text{ x } \frac{6}{16} \text{ x } \frac{2}{3} = 5V \text{ x } \frac{12}{48} = 5V \text{ x } \frac{1}{4} = 1.25V$$

Todas as outras tensões podem ser analisadas da mesma forma. Verifique que a tensão aumenta em passos de 1.25V. A ultima condição da tabela é igual a 3.75V. Falta sempre mais um passo no final da tabela para atingir a tensão máxima (3.75V + 1.25V = 5V), qualquer que seja o número de bits usados.

A fórmula simplificada usada para calcular a tensão de saída é:

$$Vout = \frac{V1}{2} + \frac{V0}{4}$$

CONVERSOR R-2R DE 8 BITS

Este conversor é igual ao anterior com a vantagem de haver maior número de bits. A fórmula usada para obter a tensão de saída é:

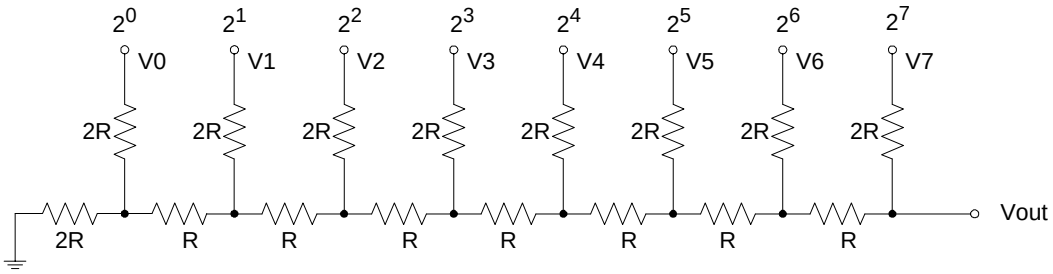
$$Vout = \frac{V7}{2} + \frac{V6}{4} + \frac{V5}{8} + \frac{V4}{16} + \frac{V3}{32} + \frac{V2}{64} + \frac{V1}{128} + \frac{V0}{256}$$

Uma das primeiras características de um conversor é a sua resolução. Isto é determinado pelo número de bits. Um conversor com 8 bits como este tem 2^8 ou 256 níveis de saída. O exemplo do circuito anterior tem 2 bits e por isso apenas 4 níveis de saída. A resolução também pode ser expressa em percentagem. A resolução de um conversor de 2 bits é de ${}^{1}\!/_{4}$ x 100% ou de 25%. A resolução de um conversor de 8 bits é de ${}^{1}\!/_{256}$ x 100% ou de 0.39%.

A próxima característica é a tensão máxima que se pode obter à saída do conversor. Quanto maior o

número de bits, mais próxima é a tensão máxima da tensão digital.

$$Vmax = \frac{2^{n^o\,bits} - 1}{2^{n^o\,bits}} = \frac{255}{256} = 0.996 \text{ x } 5V = 4.98V$$



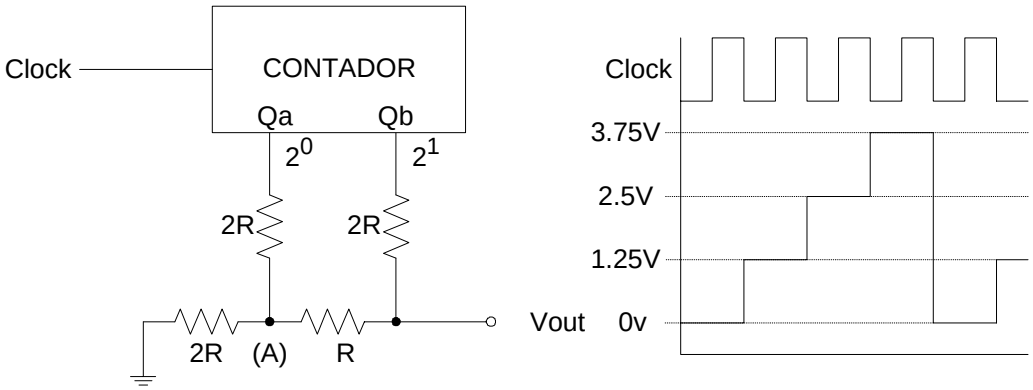
Existe outra característica importante de um conversor D/A que é a linearidade. A linearidade é uma medida de quanto a saída do conversor se desvia de uma linha recta à medida que a tensão do conversor progride. Idealmente, o desvio entre a saída do conversor e uma linha não recta não deve ser maior que $\pm \frac{1}{2}$ do valor do bit menos significativo. Entretanto, muitos conversores D/A têm erros de linearidade maiores que esse valor.

Ainda existe outra característica para ver que é o tempo de atraso. Quando se altera a palavra em binário aplicada na entrada do conversor, a saída vai variar para o valor apropriado. Entretanto a saída só vai assumir esse valor passado algum tempo. O tempo que leva a variar a saída $\pm \frac{1}{2}$ do bit menos significativo chama-se tempo de atraso do conversor. Por exemplo, o conversor de 10 bits DAC1020 tem um tempo de atraso típico de 500ns. Esta característica é importante quando um conversor é construído para trabalhar em alta frequência.

Aplicações do conversor D/A

Estes conversores têm muitas aplicações, além disso são muito usados nos computadores e leitores de Compact Disk. No leitor de CD, são usados conversores D/A de 14 ou 16 bits para converter os dados lidos do disco num sinal áudio. A maior parte dos sintetizadores de voz contêm um conversor D/A.

Vamos utilizar um conversor D/A para ligar na saída dum contador de 2 bits.



Verifique que a tensão de saída é semelhante a uma onda dente de serra em escada. Cada degrau é igual ao peso do bit menos significativo e também igual a $Vout = 5V \times \frac{1}{4} = 1.25V$ como já verificado anteriormente. O tempo de cada degrau corresponde também à duração de um ciclo completo do Clock. Quanto maior número de bits do contador e do conversor maior será o número de degraus entre o valor mínimo e máximo. Isto significa uma melhor resolução da onda dente de serra.

CONVERSOR ANALÓGICOS DIGITAIS

A função do conversor analógico digital A/D é produzir uma palavra digital que representa a magnitude de um valor analógico. As especificações de um conversor A/D são muito semelhantes com as do conversor D/A. A resolução de um conversor A/D refere-se ao número de bits na palavra digital à saída do conversor. Um conversor A/D de 8 bits, por exemplo, tem uma resolução de $1/256$. A precisão e linearidade têm o mesmo significado no conversor A/D que no conversor D/A.

Uma outra característica importante num conversor A/D é o tempo de conversão. Isto é simplesmente o tempo que o conversor leva para produzir uma saída em binário para um valor analógico aplicado na entrada. Quando nos referimos a um conversor de alta velocidade, queremos dizer que ele tem um tempo de conversão muito pequeno.

Existem várias maneiras para fazer uma conversão A/D, vamos analisar alguns dos métodos mais usados.

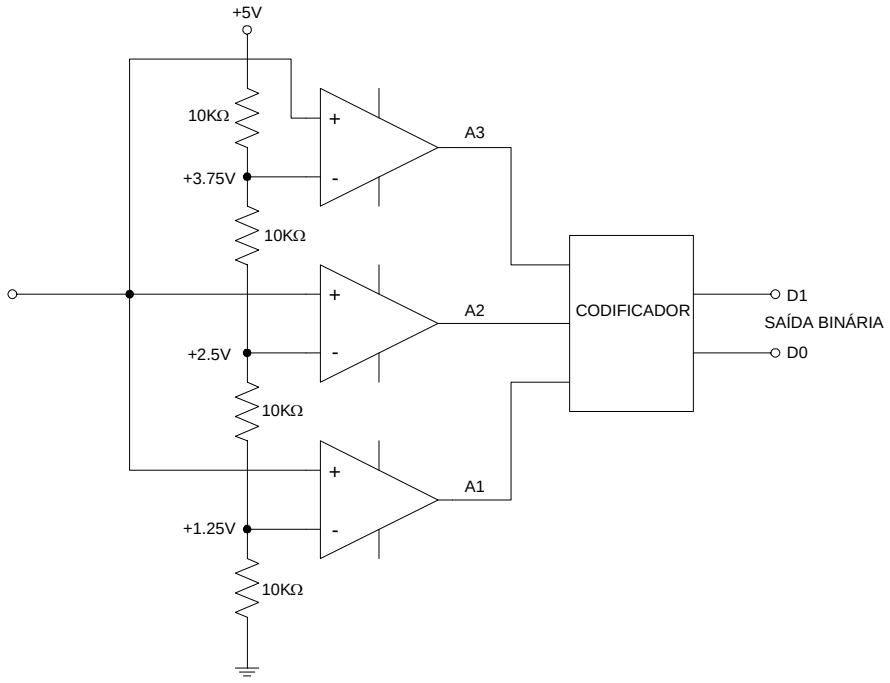
TIPOS DE CONVERSORES A/D

Conversor com comparador paralelo

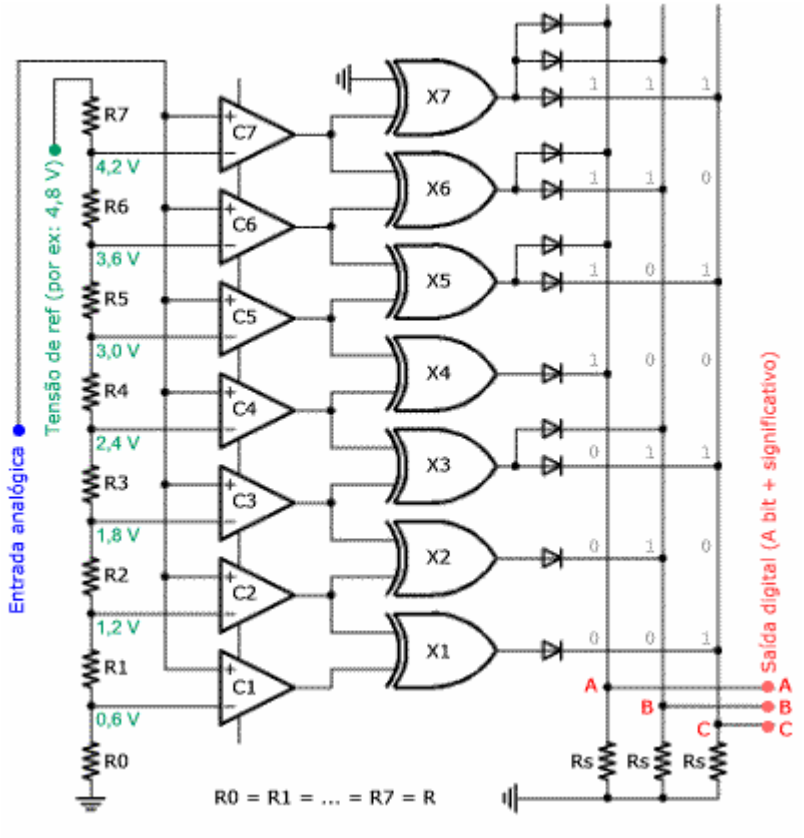
A figura seguinte mostra um circuito conversor A/D de 2 bits usando comparadores. Um divisor de tensão fornece as tensões de referência na entrada de cada um dos comparadores. A tensão no topo do divisor representa a tensão do nível lógico 1. Se a tensão na entrada de um comparador é superior à tensão de referencia na entrada não inversora, a saída do comparador passa a nível lógico 1. As saídas dos comparadores dão-nos uma representação digital da tensão de entrada. Com uma tensão de entrada de 2.6V, por exemplo, as saídas dos comparadores A1 e A2 vão estar a nível lógico 1.

A maior vantagem do conversor A/D paralelo é a sua velocidade, que é simplesmente o tempo de atraso dos comparadores. A saída dos comparadores é um formato binário não standard, mas pode ser convertido para qualquer código binário desejado com algumas portas lógicas. A maior desvantagem de um conversor A/D paralelo é o número de comparadores necessários para produzir um resultado com uma resolução razoável.

O conversor de 2 bits da figura seguinte requer 3 comparadores. Para construir um conversor de N bits precisa de $2^N - 1$ comparadores. Para um conversor de 8 bits, precisa de 255 comparadores, e para um conversor de 10 bits precisa de 1023 comparadores. Existem conversores A/D com comparadores paralelos em circuitos integrados para serem usados em aplicações onde a alta velocidade é necessária, mas são relativamente caros. Alguns conversores com comparadores paralelos podem fazer uma conversão em menos de 10ns.

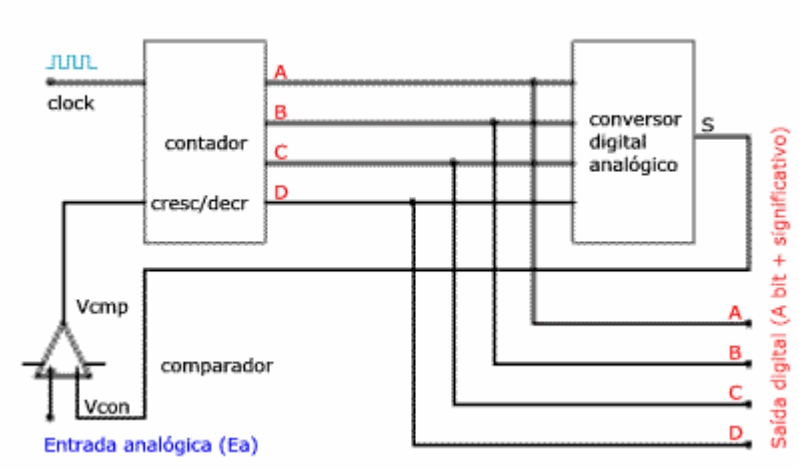


Conversor de 3 bits:



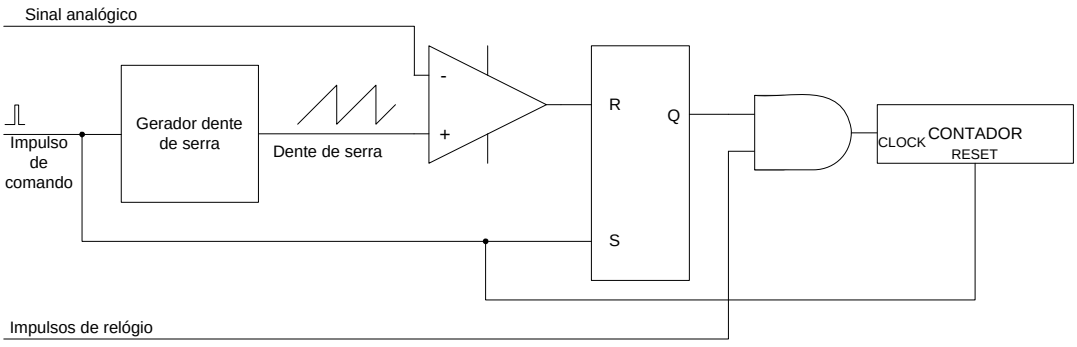
No Conversor Delta, os impulsos de clock alimentam continuamente a entrada do contador, o qual dispõe de uma entrada digital que comuta, de acordo com o nível lógico, o sentido da contagem (crescente ou decrescente). Enquanto a entrada analógica é maior que Vcon, a saída do comparador é 1 e o contador funciona no modo crescente. Quando Vcon se torna maior que "Ea", a saída do comparador vai para 0 e o contador funciona no modo decrescente.

Isso leva Vcon a um valor imediatamente abaixo de "Ea", invertendo o processo. Assim, considerando "Ea" constante, o contador opera continuamente entre dois valores próximos de "Ea", não havendo necessidade dos flip-flops de armazenamento. Se o valor de "Ea" muda, o patamar de operação também muda.



Conversor de rampa simples

Este tipo de conversor é bastante vulgar, o esquema é apresentado na figura seguinte. Um impulso de comando dá inicio à onda dente de serra, acciona o flip-flop RS e acciona o RESET do contador. A partir deste momento a saída do flip-flop é 1 e é aplicada na porta AND. O contador começa a contar a partir do zero e a tensão da onda dente de serra começa a subir a partir do zero. Enquanto a tensão de dente de serra for inferior à tensão do sinal analógico, a contagem continua, mas quando a tensão dente de serra passa pela tensão analógica o comparador comuta e acciona a entrada R do flip-flop. Isto desactiva a porta AND e termina assim a contagem. O número da contagem é assim proporcional à tensão analógica. O impulso de comando seguinte coloca o contador na posição inicial, a dente de serra a zero e começa uma nova contagem.



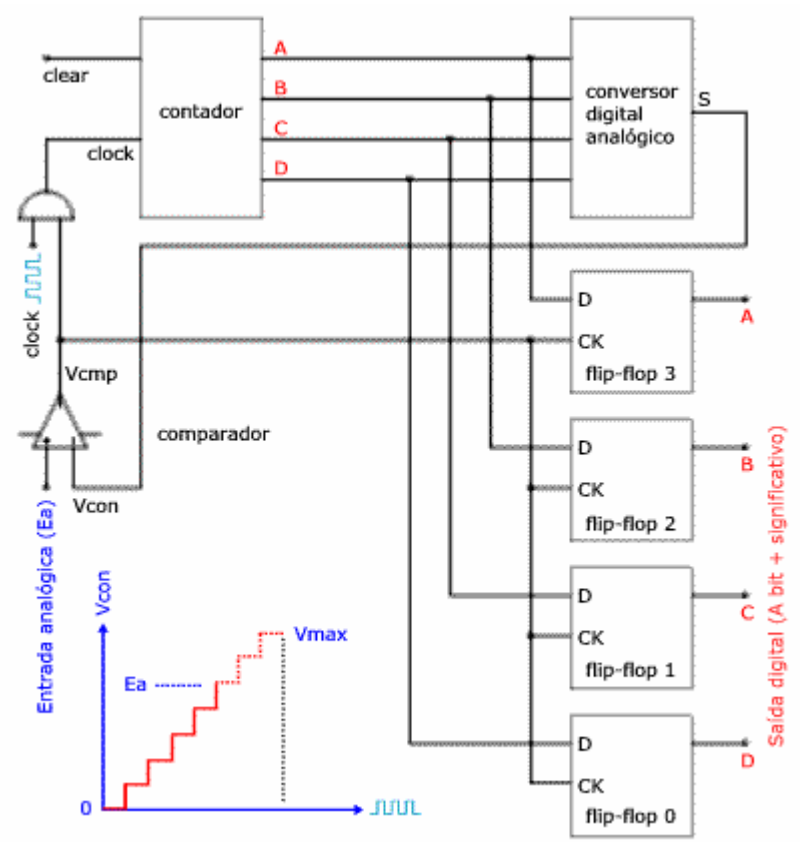
Este conversor também pode ser feito da seguinte forma; a saída de um contador (de 4 bits neste exemplo) é ligada na entrada de um conversor digital analógico. Supomos de início que a entrada de clock do contador é continuamente alimentada com uma sequência de impulsos. Nesta situação, a tensão Vcon na saída S do conversor varia entre 0 e um valor Vmax, que depende do contador e das características do

conversor digital analógico.

Mas no circuito há um comparador e uma porta AND na entrada de clock. Enquanto a tensão Vcon for menor que a da entrada analógica "Ea", a saída do comparador é 1 e os impulsos de clock são dirigidos ao contador.

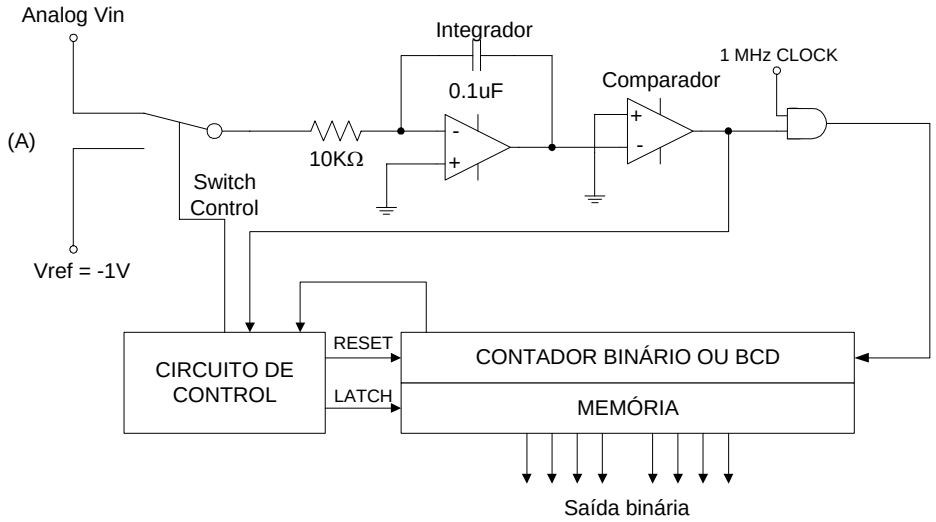
No momento em que Vcon se torna maior que "Ea", a saída do comparador passa para 0, bloqueando os impulsos de clock e, portanto, a contagem.

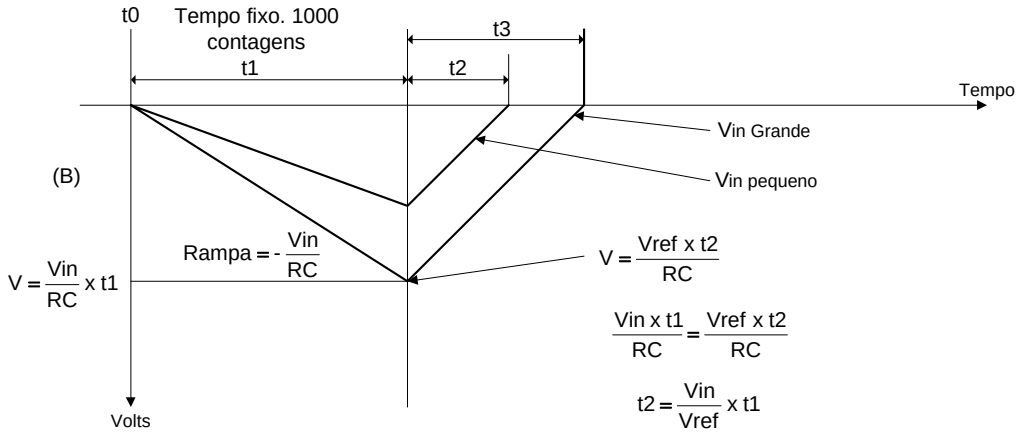
Uma vez que a saída do comparador também vai para a entrada de clock dos flip-flops (tipo D), o valor digital da saída do contador é armazenado nos mesmos (lembrar que flip-flops tipo D só permitem a mudança de estado na transição de clock, neste caso descendente). Portanto, a saída digital armazenada nos flip-flops tem relação linear com a entrada analógica "Ea".



Conversor A/D de dupla rampa

A figura seguinte (A) mostra o esquema bloco de um conversor A/D de dupla rampa. Este tipo de conversor é normalmente usado nos multímetros digitais porque pode fornecer um grande numero de bits de resolução a baixo custo. Vamos ver como ele funciona.





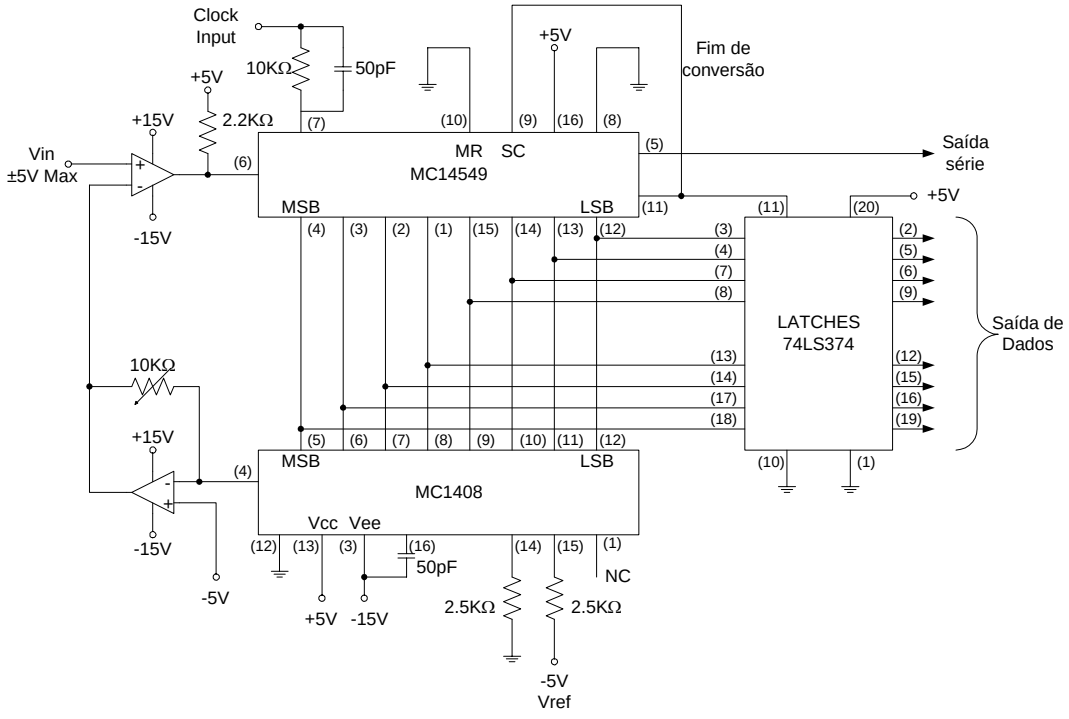
Para começar, o circuito de controlo coloca todos os contadores a zero e liga a entrada do integrador à tensão de entrada analógica. Se assumir que a tensão de entrada é positiva, então a saída do integrador descerá para negativo como está representado em (B) na figura anterior. Assim que a saída do integrador desça alguns microvolts abaixo de zero, a saída do comparador passará a nível lógico 1. A saída do comparador a 1 acciona a porta AND e permite que o clock de 1 MHz chegue ao contador. Normalmente após 1000 impulsos contados no contador, o circuito de controlo comuta a entrada do integrador para a tensão de referencia negativa e coloca todos os contadores a zero. Com uma tensão de entrada negativa, a saída do integrador subirá para positiva como se mostra na direita da figura anterior (B). Quando a saída do integrador passa por zero, a saída do comparador passa a nível lógico 0 e desliga o sinal de 1 MHz para os

contadores. O numero de contagens necessárias para a tensão de saída do integrador voltar a zero é directamente proporcional à tensão de entrada. Por exemplo, uma tensão na entrada de +2V produz uma contagem de 2000. Uma vez que a resistência e o condensador no integrador são usados tanto pela tensão de entrada como pela tensão de referencia, pequenas variações nos seus valores devido às variações de temperatura não tem efeito na precisão da conversão.

A principal desvantagem dum conversor A/D de dupla rampa é a sua baixa velocidade de conversão.

Conversor A/D de aproximações sucessivas

A figura seguinte mostra um conversor A/D de aproximações sucessivas de 8 bits. O coração deste conversor é um registo de aproximações sucessivas como por exemplo o MC14549, que funciona da seguinte maneira.



No primeiro impulso de clock do início de um ciclo de conversão, o MC14549 coloca a saída do bit mais significativo a 1 para o conversor D/A MC1408. O conversor D/A e o amplificador convertem este bit para uma tensão que é aplicada na entrada do comparador. Se esta tensão é maior que a tensão de entrada na outra entrada do comparador, a saída do comparador passa a lógico 0 e avisa o MC14549 para desligar este bit (MSB) porque é muito grande. Se a tensão do conversor D/A é inferior à tensão de entrada, a saída do comparador passa a lógico 1, e avisa o MC14549 que deve manter este bit (MSB) ligado. No próximo

impulso de clock, o MC14549 vai ligar o próximo bit a seguir ao MSB para o conversor D/A. Baseado na resposta obtida no comparador, MC14549 irá manter ou desligar este bit. O MC14549 prossegue assim até ao bit menos significativo. Quando a conversão estiver completa, o resultado em binário está presente na saída do MC14549, e o MC14549 envia um sinal de fim conversão. Apenas são necessários nove impulsos de clock para efectuar toda a conversão. No esquema anterior, o sinal de fim de conversão é usado para memorizar a informação nas latches, assim a informação pode ser usada para ser lida por um micro-computador. Se a saída de fim de conversão é ligada na entrada SC (inicio de conversão), então o conversor vai efectuar conversões sucessivamente.

BIBLIOGRAFIA

Padilla, António J. G.-Sistemas Digitais.

Hall, Douglas V. Hall-Microprocessors and Interfacing Programming and Hardware

Matemática II

Apontamentos Pessoais

LISTA DE PÁGINAS EM VIGOR

PÁGINAS	EM VIGOR
CAPA (Verso em branco)	ORIGINAL
CARTA DE PROMULGAÇÃO (Verso em branco)	ORIGINAL
REGISTO DE ALTERAÇÕES (Verso em branco)	ORIGINAL
1 (Verso em branco)	ORIGINAL
3 (Verso em branco)	ORIGINAL
5 a 40	ORIGINAL
41 (Verso em branco)	ORIGINAL
43 a 54	ORIGINAL
55 (Verso em branco)	ORIGINAL
LPV-1 (Verso em branco)	ORIGINAL