

ELETRÔNICA DIGITAL

Sistema Digital

É qualquer sistema de transmissão ou processamento de informações no qual a informação assume apenas valores discretos.

Vantagens dos Sistemas e Circuitos Digitais

- Maior confiabilidade
- Maior imunidade a ruídos (transmissão, processamento, gravação de dados ou informações)
- Capacidade de transmissão de sinais a longas distâncias
- Facilidade de manuseio (processamento de dados)
- Facilidade e precisão na leitura de dados
- Facilidade de interfaceamento
- Possibilidade de realizar várias tarefas simultaneamente
- Capacidade de processamento de grandes quantidades de informações (cálculos, comparações, análises)
- Menor custo
- Maior rapidez

Tipos de Circuitos Digitais

- **Circuitos Combinacionais:** São circuitos em que a saída depende apenas das várias combinações entre as variáveis de entrada (circuitos sem realimentação).
- **Circuitos Seqüenciais:** Sua saída depende não somente das variáveis de entrada presentes, mas também da seqüência das entradas anteriores (circuitos realimentados).

SISTEMAS DE NUMERAÇÃO

Conjunto de dígitos (símbolos) separados e distintos que podem ser combinados e assim representar infinitas quantidades.

Em geral são sistemas posicionais, ou seja, o valor de cada dígito depende da posição que ele ocupa no número.

Sistema Decimal

É composto de **dez** dígitos: 0, 1, 2, ..., 8, 9 (Base 10)

$$5986,32 = 5 \times 10^3 + 9 \times 10^2 + 8 \times 10^1 + 6 \times 10^0 + 3 \times 10^{-1} + 2 \times 10^{-2}$$

Sistema Binário

É composto de **dois** dígitos: 0 e 1 (Base 2)

$$(1011,01)_2 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2} = (11,25)_{10}$$

Sistema Octal

É composto de **oito** dígitos: 0, 1, 2, ..., 6, 7 (Base 8)

$$(1775,3)_8 = 1 \times 8^3 + 7 \times 8^2 + 7 \times 8^1 + 5 \times 8^0 + 3 \times 8^{-1} = (1021,375)_{10}$$

Sistema Hexadecimal

Composto de dezesseis dígitos: 0, 1, ..., 9, A, B, ..., E, F (Base 16)

$$(9FA)_{16} = 9 \times 16^2 + 15 \times 16^1 + 10 \times 16^0 = (2554)_{10}$$

Conversão de Decimal para outras Bases

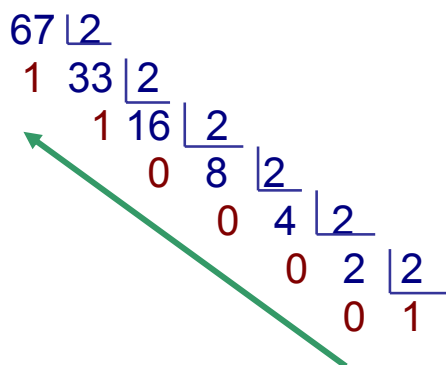
Dividir a parte inteira do n° . a ser convertido pelo n° . de base (2, 8 ou 16), até o último quociente inteiro, que será o bit mais significativo do n° . já convertido. Os bits seguintes serão os restos obtidos nas divisões, obedecendo a seqüência: do último ao primeiro resto.

Multiplicar parte fracionária pelo n° . de base, extrair do produto a parte inteira, que é o bit mais significativo, e multiplicar o restante novamente pelo n° . de base. Repetir esta operação até que se obtenha zero (e o resultado é exato) ou até que se chegue a uma repetição (que é uma dízima) ou ainda a resultados não exatos (nem zero, nem dízima).

Exemplos:

Base 10 para Base 2: $(67,625)_{10} = (?)_2$

Parte inteira



$$(67)_{10} = (1000011)_2$$

Parte Fracionária

$0,625$	$0,250$	$0,500$
$\underline{\times 2}$	$\underline{\times 2}$	$\underline{\times 2}$
$1,250$	$0,500$	$1,000$


$$(0,625)_{10} = (0,101)_2$$

$$(67,625)_{10} = (1000011,101)_2$$




Base 10 para Base 8: $(92,635)_{10} = (?)_8$

Parte inteira

$$\begin{array}{r} 92 \overline{) 8} \\ 4 \overline{) 11} \overline{) 8} \\ 3 \end{array}$$


Parte Fracionária

0,635	0,08	0,64	0,12	0,96	0,68	0,44	0,52
<u>x8</u>	<u>x8</u>	<u>x8</u>	<u>x8</u>	<u>x8</u>	<u>x8</u>	<u>x8</u>	<u>x8</u>
5,080	0,64	5,12	0,96	7,68	5,44	3,52	4,16



$$(92,635)_{10} = (134,50507534...)_{8}$$

Base 10 para Base16: $(1000,635)_{10} = (?)_{16}$

Parte Inteira

$$\begin{array}{r} 1000 \overline{) 16} \\ 8 \overline{) 62} \overline{) 16} \\ 14 \end{array}$$


mas $(14)_{10} = (E)_{16}$

Parte Fracionária

0,635	0,16	0,56	0,96	0,36	0,76	0,16
<u>x16</u>	<u>x16</u>	<u>x16</u>	<u>x16</u>	<u>x16</u>	<u>x16</u>	<u>x16</u>
10,160	2,56	8,96	15,36	5,76	12,16	2,56 <u>repet.</u>

$$(1000,635)_{10} = (3E8,A28F5C28F5C..)_{16}$$

Conversão entre os sistemas binário, octal e hexadecimal

Pode ser feita de maneira imediata, por exemplo, de octal para binário ou de hexa para binário, basta substituir o dígito em octal ou hexa, pelo seu equivalente binário, respectivamente com três ou quatro dígitos.

Exemplo:

$$(54,31)_8 = (101100,011001)_2$$

$$\begin{array}{cccc} 5 & 4, & 3 & 1 \\ 101 & 100, & 011 & 001 \end{array}$$

$$(9F2,5C)_{16} = (100111110010,01011100)_2$$

$$\begin{array}{ccccc} 9 & F & 2, & 5 & C \\ 1001 & 1111 & 0010, & 0101 & 1100 \end{array}$$

Analogamente, para converter um n°. binário em octal ou hexa, basta agrupá-lo respectivamente em três ou quatro bits, sempre a partir do menos significativo, na parte inteira e do mais significativo, na parte fracionária e substituir por seu equivalente octal ou hexa.

Exemplo:

$$\overbrace{(1000110011)}_2 = (\overbrace{1063})_8$$

$$\begin{array}{cccc} 001 & 000 & 110 & 011 \\ 1 & 0 & 6 & 3 \end{array}$$

$$\overbrace{(1111101000)}_2 = (\overbrace{3E8})_{16}$$

$$\begin{array}{ccc} 0011 & 1110 & 1000 \\ 3 & 14=E & 8 \end{array}$$

Aritmética Binária

Circuitos digitais trabalham muito mais facilmente em binário do que em qualquer outra base, já que o número de variáveis é menor (0 e 1). As operações aritméticas (adição, subtração, multiplicação e divisão), que, portanto em circuitos digitais são feitas no sistema binário, obedecem as mesmas regras (algoritmos) do sistema decimal.

Tabela da Adição

$$0 + 0 = 0$$

$$0 + 1 = 1$$

$$1 + 0 = 1$$

$$1 + 1 = 0 \text{ e "vai 1" para o próximo bit (carry)}$$

$$\begin{array}{r} 1111 \\ 1001 \\ + 1111 \\ \hline 11000 \end{array} \begin{array}{l} (9)_{10} \\ (15)_{10} \\ (24)_{10} \end{array}$$

Tabela da Subtração

$$0 - 0 = 0$$

$$1 - 0 = 1$$

$$1 - 1 = 0$$

$$0 - 1 = 1 \text{ e "empresta 1" do próximo bit (borrow)}$$

$$\begin{array}{r} 11101 \\ - 10011 \\ \hline 01010 \end{array} \begin{array}{l} (29)_{10} \\ (19)_{10} \\ (10)_{10} \end{array}$$

Multiplicação

$$\begin{array}{r} 1001 \quad (9)_{10} \\ \times 1011 \quad (11)_{10} \\ \hline 1001 \\ 1001 \\ 0000 \\ 1001 \\ \hline 1100011 \quad (99)_{10} \end{array}$$

Divisão

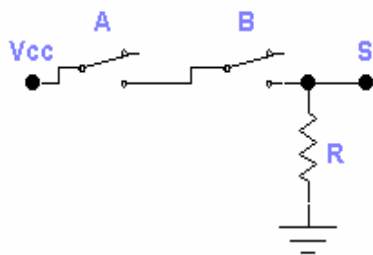
$$\begin{array}{r} (145)_{10} \quad 10010001 \\ - 1011 \quad (11)_{10} \\ \hline 001110 \\ - 1011 \\ \hline 001101 \\ - 1011 \\ \hline 0010 \quad (2)_{10} \end{array}$$

FUNÇÕES LÓGICAS

Básicas: And, Or e Inversora

Compostas: Nand, Nor, Ou Exclusivo e Coincidência

1- Função And (E)



Operação Básica

$$0 \cdot 0 = 0$$

$$0 \cdot 1 = 0$$

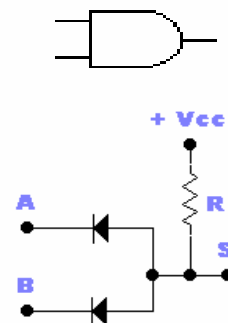
$$1 \cdot 0 = 0$$

$$1 \cdot 1 = 1$$

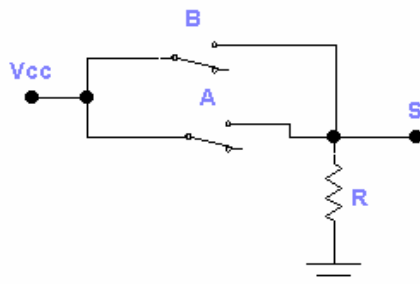
onde “ \cdot ” significa “E”

$$S = A \cdot B$$

Símbolo



2 – Função OR (Ou)



Operação Básica

$$0 + 0 = 0$$

$$0 + 1 = 1$$

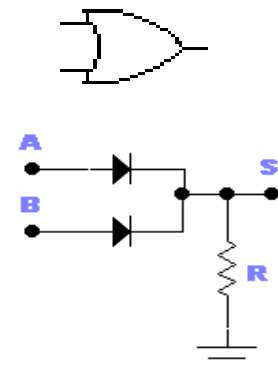
$$1 + 0 = 1$$

$$1 + 1 = 1$$

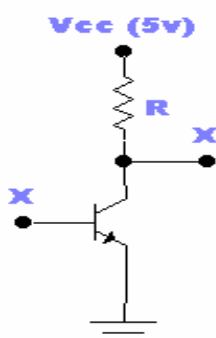
onde “+” significa “Ou”

$$S = A + B$$

Símbolo



3 - Função Inversora (Complementação ou Not)



Operação Básica

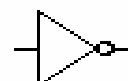
$$\text{Se } \overline{X} = 0 \quad X = 1$$

$$\text{Se } \overline{X} = 1 \quad X = 0$$

A barra sobre a variável, significa “inversão” ou “complementação”

$$S = \overline{X}$$

Símbolo



Funções Lógicas Compostas

4 - Função NAND (Not + And)

Símbolo



Operação Básica

$$\begin{aligned}\overline{0 \cdot 0} &= 1 \\ \overline{0 \cdot 1} &= 1 \\ \overline{1 \cdot 0} &= 1 \\ \overline{1 \cdot 1} &= 0\end{aligned}$$

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

5 – Função NOR (Not + Or)

Símbolo



Operação Básica

$$\begin{aligned}\overline{0 + 0} &= 1 \\ \overline{0 + 1} &= 0 \\ \overline{1 + 0} &= 0 \\ \overline{1 + 1} &= 0\end{aligned}$$

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

6 – Função “Exclusive Or” (Ou exclusivo)

Símbolo



Operação Básica

$$\begin{aligned}0 \oplus 0 &= 0 \\ 0 \oplus 1 &= 1 \\ 1 \oplus 0 &= 1 \\ 1 \oplus 1 &= 0\end{aligned}$$

$$S = A \oplus B = \bar{A}B + A\bar{B}$$

7 – Função Coincidência (Not Exclusive Or)

Símbolo



Operação Básica

$$\begin{aligned}0 \ominus 0 &= 0 \\ 0 \ominus 1 &= 1 \\ 1 \ominus 0 &= 1 \\ 1 \ominus 1 &= 0\end{aligned}$$

$$S = A \ominus B = \bar{A}\bar{B} + AB$$

As expressões $\overline{A + B} = \overline{A} \cdot \overline{B}$ e $\overline{A \cdot B} = \overline{A} + \overline{B}$ constituem os “Teoremas de De Morgan” e são utilizadas em processos de simplificação de circuitos lógicos.

Expressões Booleanas

Expressões compostas por variáveis lógicas, representando associações entre diversas *portas lógicas* (*circuitos que executam funções lógicas*).

Exemplos:

1 – Representar a expressão $S = \overline{A} \overline{B} + \overline{C} D$

2 – Representar a expressão $S = \overline{A \cdot (C + D)} + \overline{B E}$

Tabela Verdade

Utilizada para avaliar o resultado de uma expressão booleana para todas as combinações possíveis entre as variáveis de entrada.

Exemplo: Faça a Tabela Verdade da expressão do exemplo 1

A	B	C	D	$A\bar{B}$	$\bar{C}D$	S
0	0	0	0	0	1	1
0	0	0	1	0	1	1
0	0	1	0	0	1	1
0	0	1	1	0	0	0
0	1	0	0	0	1	1
0	1	0	1	0	1	1
0	1	1	0	0	1	1
0	1	1	1	0	0	0
1	0	0	0	1	1	1
1	0	0	1	1	1	1
1	0	1	0	1	1	1
1	0	1	1	1	0	1
1	1	0	0	0	1	1
1	1	0	1	0	1	1
1	1	1	0	0	1	1
1	1	1	1	0	0	0

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

Duas expressões booleanas são equivalentes se e somente se apresentarem as mesmas tabelas verdade.

Exemplo: Verifique a igualdade $A\bar{B} + C = (A + C) \cdot (\bar{B} + C)$

A	B	C	$A\bar{B}$	$A\bar{B} + C$	$A + C$	$\bar{B} + C$	$(A + C) \cdot (\bar{B} + C)$
0	0	0	0	0	0	1	0
0	0	1	0	1	1	1	1
0	1	0	0	0	0	0	0
0	1	1	0	1	1	1	1
1	0	0	1	1	1	1	1
1	0	1	1	1	1	1	1
1	1	0	0	0	1	0	0
1	1	1	0	1	1	1	1

FAMÍLIAS LÓGICAS

Os circuitos integrados podem ser classificados por **densidade de integração**, ou seja, por número de portas:

SSI – Integração em Pequena Escala (até 11 portas)

MSI - Integração em Média Escala (de 12 até 100 portas)

LSI - Integração em Grande Escala (de 101 até 1000 portas)

VLSI - Integração em Escala Muito Grande (mais de 1000)

São utilizadas duas tecnologias básicas para fabricação de CI'S:

Bipolar – Integração de transistores bipolares na pastilha de Silício (SSI e MSI)

MOS – Integração de transistores de efeito de campo (LSI e VLSI)

Pertencem a uma mesma **família digital**, um grupo de dispositivos compatíveis (podem ser conectados entre si), com os mesmos níveis lógicos tensões de alimentação.

Família Digital utilizando tecnologia bipolar:

Obsoletas

DTL – Lógica “Diodo Transistor”

RTL - Lógica “Resistor Transistor”

RCTL - Lógica “Resistor Capacitor Transistor”

Mais popular

TTL - Lógica “Transistor Transistor”

Aplic. em altíssima velocidade

ECL - Lógica “Emissor Acoplado”

Alta imunidade a ruído

HTL – “High Threshold Logic”

Família Digital utilizando Tecnologia MOS

LSI-Microp.
e Memórias

Obsoletas
Muito lentas

PMOS – MOSFET canal P

NMOS – MOSFET canal N

Mais utilizadas
Baixo consumo

CMOS – MOSFET's complementares (canal P e N)

A maioria dos CI's da **Família TTL** pertencem às séries 54 e 74 (Portas, Flip-flop's, decodificadores, Contadores, etc.)

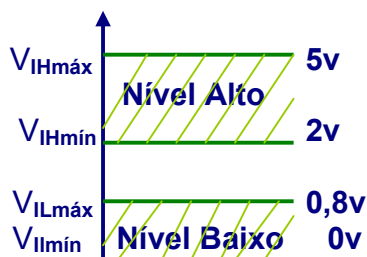
A série 54 é utilizada em aplicações militares –
Temperatura: de -55°C a 125°C
Tensão de alimentação: $5\text{v} \pm 0,5\text{v}$

A série 74 é para uso geral, em aplicações comerciais –
Temperatura: de 0°C a 70°C
Tensão de alimentação: $5\text{v} \pm 0,25\text{v}$

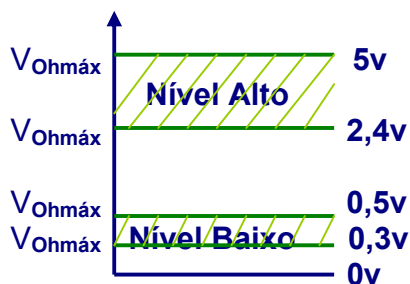
Sinais de Entrada e Saída

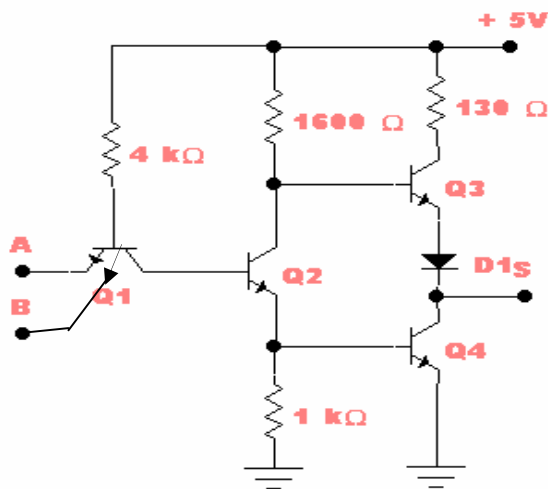
Nível Lógico	Tensão de Entrada	Tensão de Saída
0	$\leq 0,8\text{v}$	$\leq 0,4\text{v}$
1	$\geq 2,0\text{v}$	$\geq 2,4\text{v}$

Tensão de Entrada



Tensão de Saída





Como exemplo, mostramos o circuito ao lado que utiliza tecnologia TTL, é uma porta lógica **NAND** se considerarmos as entradas A e B (dois emissores) ou um **Inversor**, se considerarmos somente a entrada A (um emissor).

Na **família CMOS**, os CI's pertencem às séries 54C, 74C e 4000. As séries 74C e 54C correspondem diretamente a seus homônimos das séries 54 e 74 TTL. Funcionam, com confiabilidade garantida em uma faixa de temperatura que vai de -55°C a 125°C (54C) e de - 40°C a 85°C (74C).

A tensão de alimentação está na faixa de 3v a 15v (54C/74C) e de 3v a 18v (série 4000)

Para tensão de alimentação de 15v e temperatura ambiente (25°C), tem-se:

Nível Lógico	Tensão de Entrada	Tensão de Saída
0	$\leq 4v$	$\leq 0,05v$
1	$\geq 11v$	$\geq 14,95v$

Dispositivos que se utilizam da tecnologia CMOS têm baixa dissipação de potência se comparados aos de tecnologia TTL (por isso são usados em equipamentos alimentados por bateria: calculadoras, relógios de pulso, lap-top's etc.).

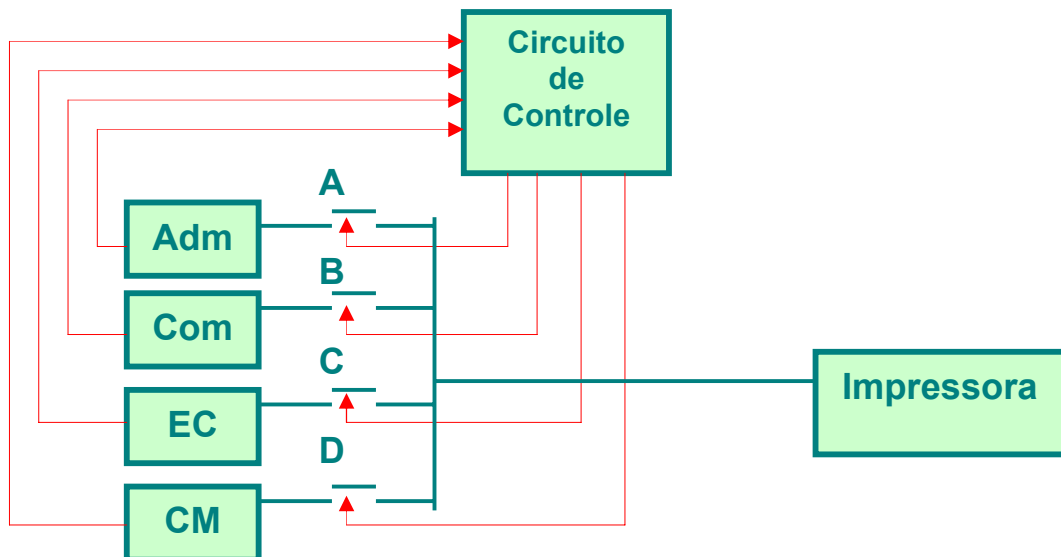
Possuem alta imunidade a ruído e sua desvantagem é a baixa velocidade (tempo de atraso de propagação).

Circuitos Combinacionais

Saída depende exclusivamente das diversas combinações entre as variáveis de entrada.

Exemplo: Como conectar quatro computadores a uma única impressora obedecendo as seguintes prioridades:

- 1^a Computador do setor administrativo (A)
- 2^a Computador do setor de compras (B)
- 3^a Computador da engenharia clínica (C)
- 4^a Computador da central de medicamentos (D)



Para que qualquer dos computadores seja ligado à impressora, a chave correspondente deve estar fechada. Essa conexão será controlada por um circuito combinacional estabelecido a partir da análise da situação.

Adotando nível lógico “1” para computador solicitando impressão e nível lógico “1” para chave fechada, podemos montar a Tabela Verdade e portanto estabelecer o circuito que executa a função extraída da tabela.

A	B	C	D	S _A	S _B	S _C	S _D
0	0	0	0	∅	∅	∅	∅
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	0
0	1	0	0	0	1	0	0
0	1	0	1	0	1	0	0
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	0
1	0	0	0	1	0	0	0
1	0	0	1	1	0	0	0
1	0	1	0	1	0	0	0
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	1	0	0	0

Após simplificar as expressões pelo método gráfico ou algébrico, chega-se às seguintes expressões:

$$S_A = A$$

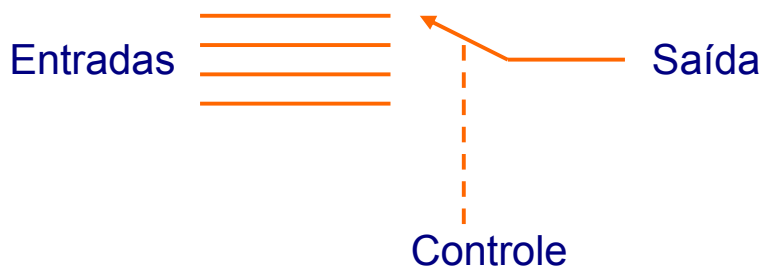
$$S_B = \bar{A} B$$

$$S_C = \bar{A} \bar{B} C$$

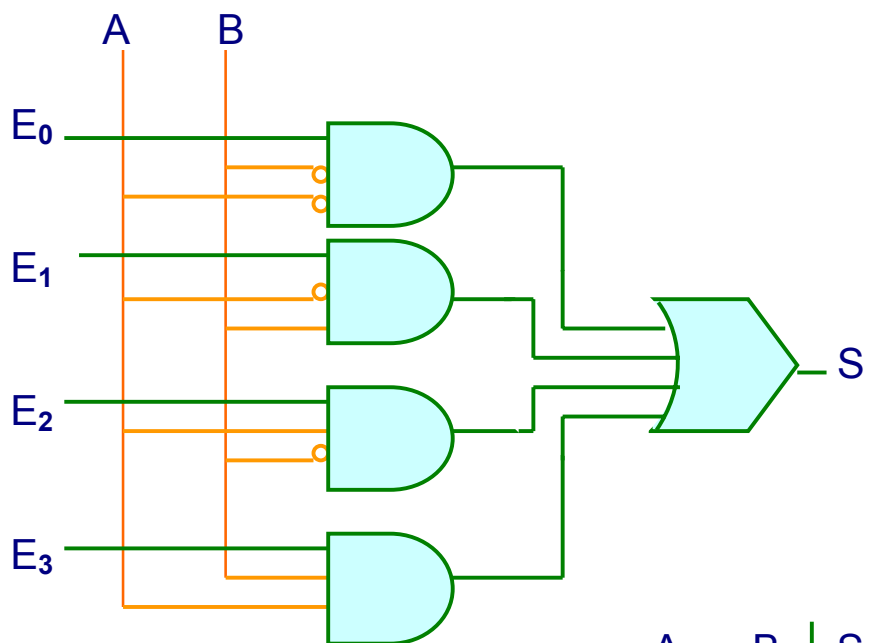
$$S_D = \bar{A} \bar{B} \bar{C} D$$

Circuitos Multiplexadores

Um circuito multiplexador executa a função de selecionar uma única de suas entradas e enviá-la para a saída.



Exemplo de implementação de um Multiplexador com portas lógicas (ou digitais)

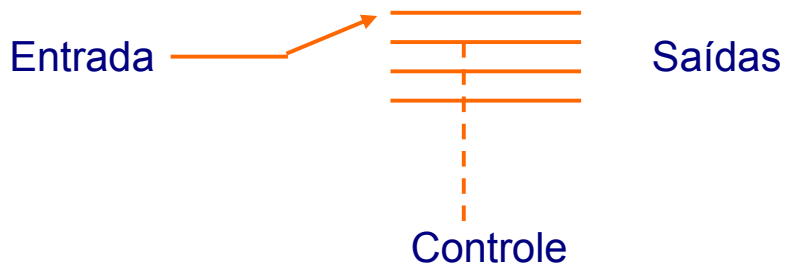


A	B	S
0	0	E ₀
0	1	E ₁
1	0	E ₂
1	1	E ₃

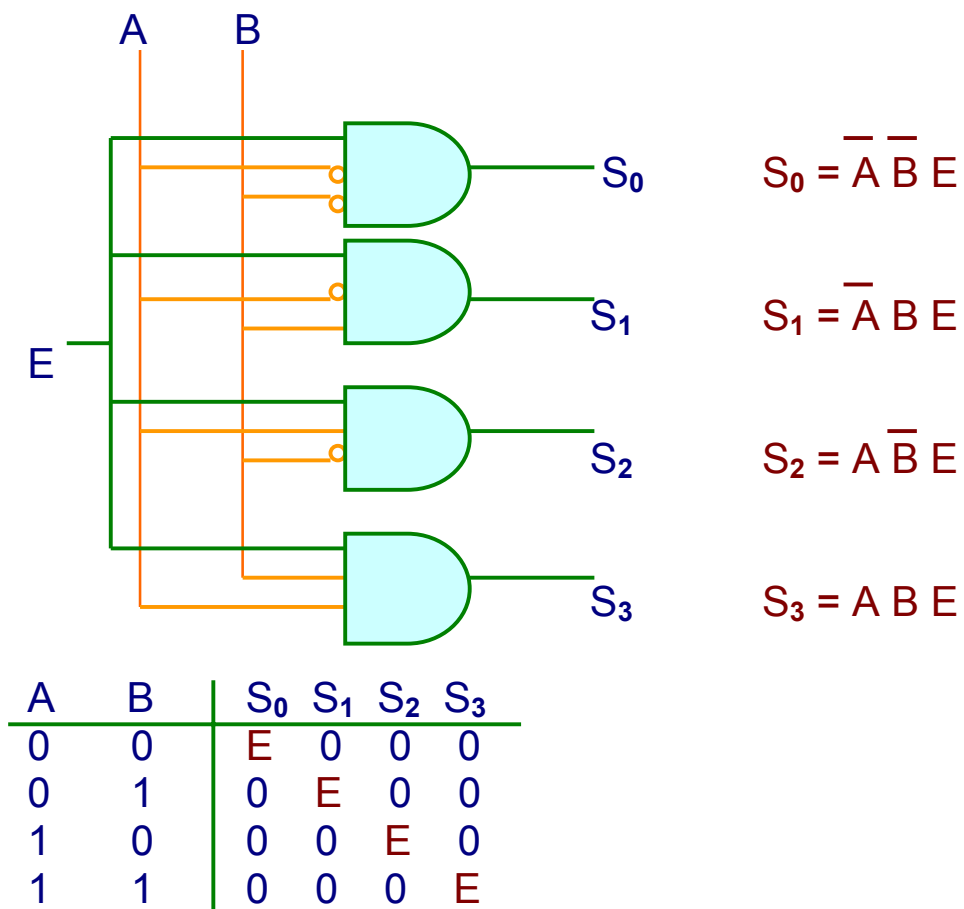
$$S = \bar{A} \bar{B} E_0 + \bar{A} B E_1 + A \bar{B} E_2 + A B E_3$$

Circuitos Demultiplexadores

Um circuito demultiplexador executa a função de selecionar a entrada e enviá-la para uma única de suas saídas.



Exemplo de implementação de um demultiplexador com portas lógicas (ou digitais)



CIRCUITOS SEQUENCIAIS

Sua saída não depende somente das entradas presentes mas também da seqüência das entradas anteriores, ou seja, precisa recordar a seqüência das entradas passadas para produzir a saída presente.

FLIP-FLOP (ou Multivibradores Biestáveis)

São os circuitos mais freqüentemente utilizados como elementos de memória. Pode assumir dois estados estáveis de saída e possui duas saídas que são complementares entre si.

Flip-Flop RS Básico

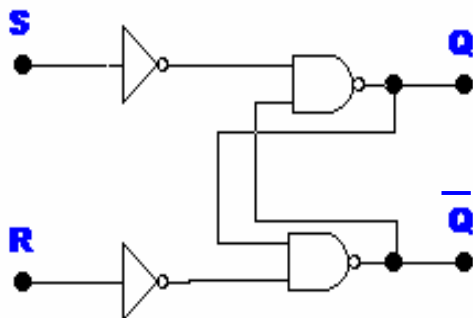
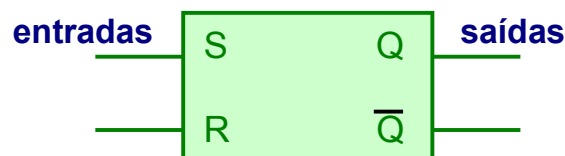


Tabela Verdade

S	R	Q	\bar{Q}
0	0	Q_a	\bar{Q}_a
0	1	0	1
1	0	1	0
1	1	não permitido	

Quando $S=R=1$ $Q=\bar{Q}$, o que é impossível

Flip-Flop RS comandado por um Pulso de Clock

Em circuitos seqüenciais de grande porte é comum sincronizar as operações de todos os flip-flop's através de um gerador de pulsos comum denominado "clock".

Neste caso quando a entrada de clock for igual a zero, o flip-flop irá permanecer no mesmo estado, mesmo que ocorram variações nas entradas R e S. Quando a entrada de clock assumir o valor "1", o circuito atuará como um flip-flop RS básico.

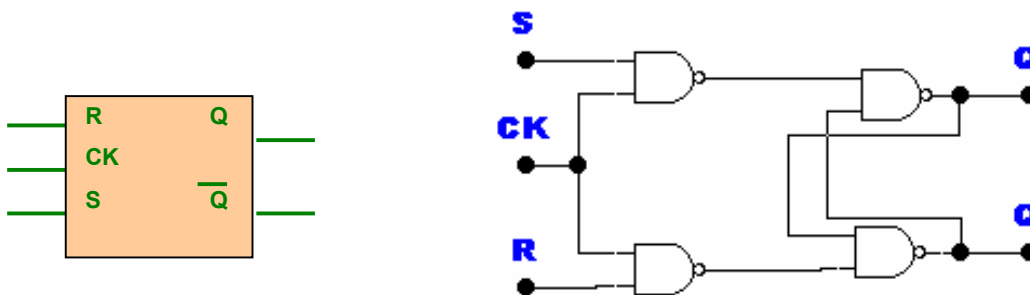


Tabela Verdade

CK	R	S	Q _f
0	X	X	Q _a
1	0	0	Q _a
1	0	1	1
1	1	0	0
1	1	1	Não permitido

onde Q_f é o estado final e Q_a é o estado anterior

Flip-Flop JK

O flip-flop RS possui um estado de entrada não permitido, (quando ambas as entradas forem iguais a “1”), o que muitas vezes dificulta sua utilização. Para contornar este problema foi desenvolvido o flip-flop JK que não possui restrição quanto ao estado de entrada.

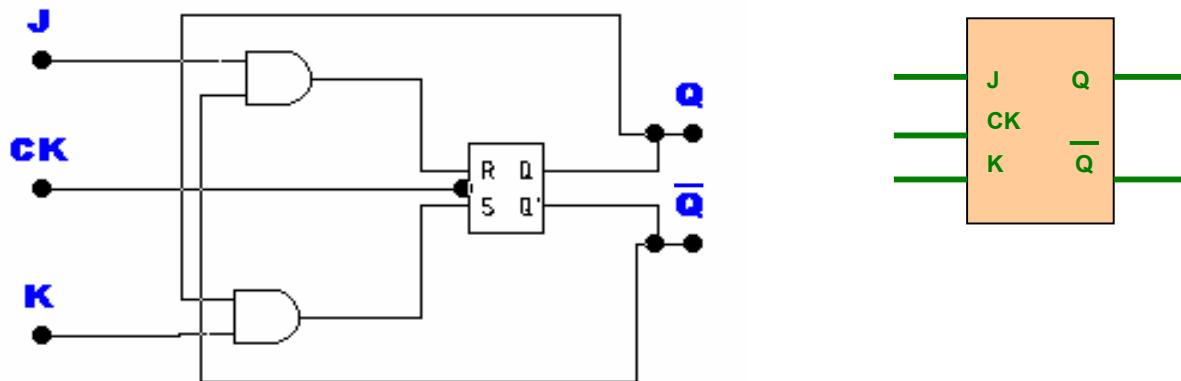
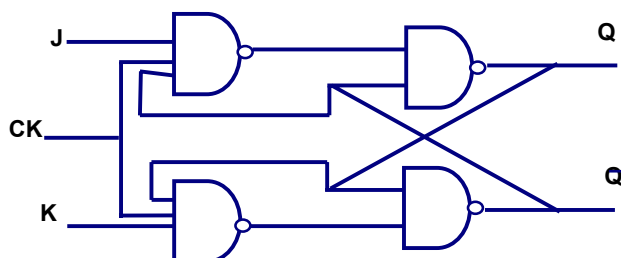


Tabela Verdade

CK	J	K	Q_f
0	X	X	Q_a
1	0	0	Q_a
1	0	1	0
1	1	0	1
1	1	1	$\overline{Q_a}^*$



* No caso de $J=K=1$, para se obter $Q_f = \overline{Q_a}$ é necessário que o clock volte a 0 imediatamente após a aplicação das entradas, caso contrário a saída entrará em oscilação provocando novamente a indeterminação.

Flip-Flop com entradas de Preset e Clear

É comum em algumas situações, desejar impor às saídas dos flip-flop's valores que sejam independentes de suas entradas convencionais, por exemplo, zerar as saídas antes de se iniciar uma determinada operação. Isto é possível através de entradas preset e clear.

Com a entrada de clock igual a zero e conseqüente bloqueio das entradas J e K (ou R e S), podemos impor ao circuito a saída $Q=1$ através da aplicação de nível zero à entrada Preset. Analogamente, podemos fazer $Q=0$ aplicando nível zero à entrada Clear.

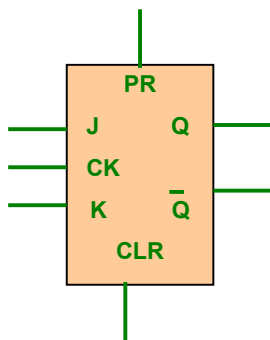


Tabela Verdade

CK	Clear	Preset	Q
0	0	0	Não permitido
0	0	1	0
0	1	0	1
x	1	1	Func. normal

Flip-Flop JK Mestre-Escravo (Master-Slave)

O flip-flop JK do tipo “mestre-escravo” possui a mesma tabela verdade do flip-flop JK comum. A diferença entre eles está no instante em que ocorre a mudança do estado de saída.

No flip-flop convencional a saída muda de estado no instante em que se aplica um pulso de clock e no flip-flop JK mestre-escravo, a saída muda de estado na descida do pulso de clock.

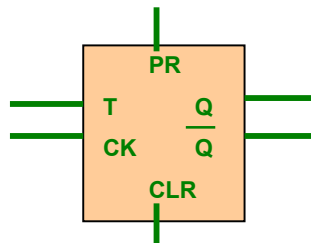
Tabela Verdade

CK	J	K	Q_f
0	X	X	Q_a
	0	0	Q_a
	0	1	0
	1	0	1
	1	1	Q_a

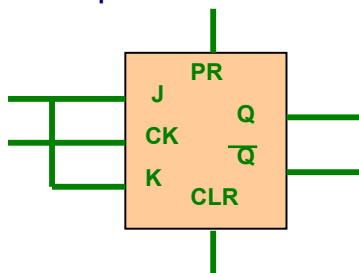
Flip-Flop Tipo T (“Toggle” - comutado)

Possui apenas uma entrada de comando e duas saídas Q e \overline{Q} complementares.

T	Q_f
0	Q_a
1	$\overline{Q_a}$



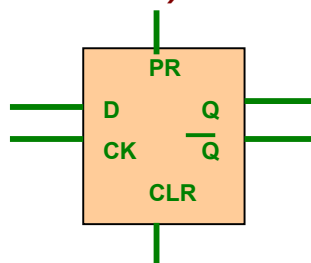
Forma de Implementar



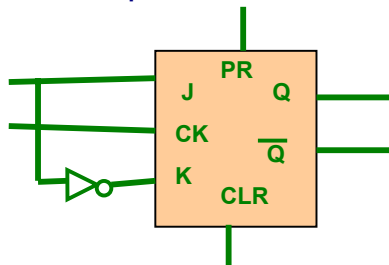
J	K	T	Q_f
0	0	0	Q_a
0	1	-	-
1	0	-	-
1	1	1	$\overline{Q_a}$

Flip-Flop Tipo D (“Data” - dado)

D	Q_f
0	0
1	1



Forma de Implementar



J	K	D	Q_f
0	0	-	-
0	1	0	0
1	0	1	1
1	1	-	-

Principais aplicações dos Flip-Flop's

Registadores de Deslocamento; Conversores Série/Paralelo e Paralelo/Série; Contadores; Memórias.

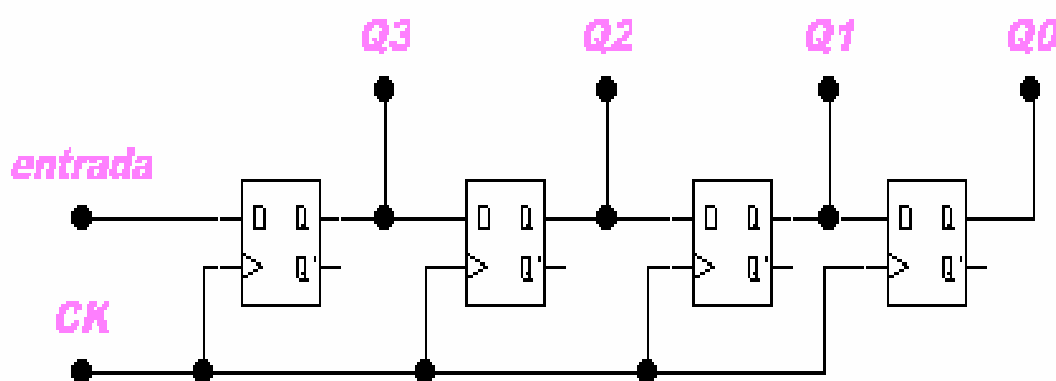
REGISTRADORES DE DESLOCAMENTO (Shift Register)

Um flip-flop é capaz de armazenar durante o período em que sua entrada de clock for igual a zero, um sinal correspondente a seu estado anterior na saída Q (um bit apenas).

Para que possamos armazenar informações com mais de um bit será necessário associarmos “em cascata” um número de flip-flop’s “JK mestre-escravo” ou “Tipo D” equivalente a quantidade de bits da informação a ser armazenada.

Esse sistema é denominado **Registrador de Deslocamento**.

Alem do armazenamento de dados o Shift Register em sua configuração básica pode também ser utilizado como Conversor Série/Paralelo onde a informação entra no modo série (bit a bit) e sai no modo paralelo (todos os bits da informação estão disponíveis simultaneamente) através das saídas Q.



Como exemplo, para armazenarmos (ou transferirmos) a informação $I = 1010$ ($I_3 I_2 I_1 I_0$) deveremos inicialmente zerar os flip-flop's através da entrada clear e injetarmos a informação, bit a bit, a partir do menos significativo (I_0).

A cada descida do pulso de clock o bit injetado no 1º flip-flop será transferido para o 2º, o do 2º para o 3º e assim sucessivamente até que, na 4ª descida do clock, todos os bits estarão disponíveis nas respectivas saídas.

Bit de injetado	Descidas de clock	Q_3	Q_2	Q_1	Q_0
$I_0 = 0$	1ª	0	0	0	0
$I_1 = 1$	2ª	1	0	0	0
$I_2 = 0$	3ª	0	1	0	0
$I_3 = 1$	4ª	1	0	1	0

Dessa forma os bits da informação estarão armazenados nas saídas Q_0 , Q_1 , Q_2 e Q_3 (conversor série/paralelo) até que nova entrada seja injetada e/ou até a próxima descida de clock.

Se mantivermos o clock até a 7ª descida teremos a informação saindo, bit a bit através de Q_0 , ou seja, entrando no modo série e saindo no modo série (conversor série/série).

Com adaptações adequadas esse sistema pode também ser transformado em conversor paralelo/série e paralelo/paralelo.

CONTADORES DIGITAIS

São dispositivos com uma entrada de clock e “n” saídas (de Q_0 a Q_{n-1}). A essas saídas, que são dígitos binários, são associados pesos correspondentes a $2^0, 2^1, \dots, 2^n$, fazendo com que a sequência Q_0, Q_1, \dots, Q_{n-1} represente um número binário.

Pertencem à classe dos circuitos sequenciais. São formados por flip-flop's interligados e podem ser síncronos (todos os flip-flop's são comandados por um único sinal de clock) e assíncronos (o instante de mudança de estado de um flip-flop é comandado pelo flip-flop precedente).

Podem realizar contagens nas formas crescente, decrescente e módulo K, onde o contador conta até K-1 e depois reinicia a contagem.

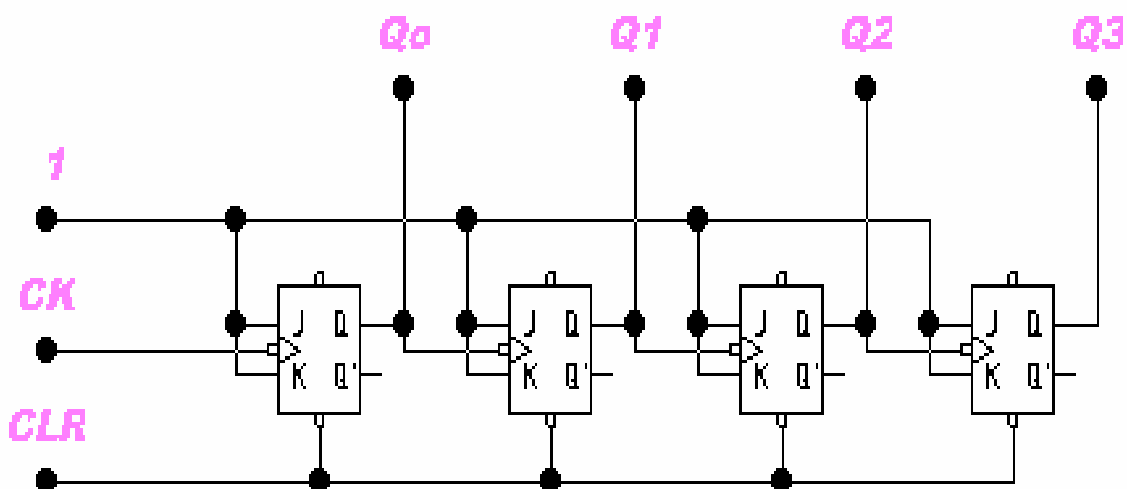
Contador Assíncrono

O pulso de clock é ligado apenas ao primeiro flip-flop. As entradas de clock dos flip-flop's seguintes são ligadas à saída do flip-flop anterior.

Contador de Pulsos Assíncrono Utilizando Flip-Flop's JK

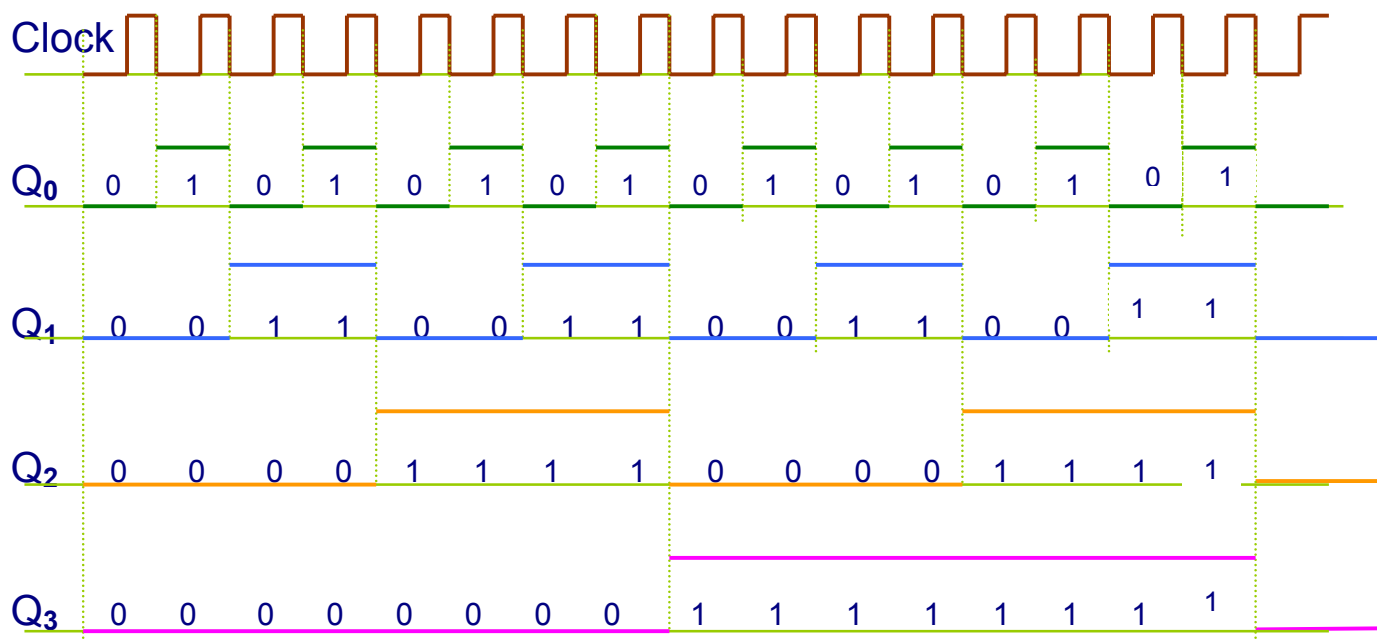
As entradas J e K são iguais a “1” para todos os flip-flop's, portanto eles complementam suas saídas sempre que a entrada de clock muda de “1” para “0”.

São utilizados flip-flop's mestre-escravo, sensíveis à borda de descida.



Antes de se iniciar a contagem, coloca-se “0” na entrada clear para zerar todos os flip-flop’s.

A cada descida de pulso de clock, o 1º flip-flop irá mudar de estado, sendo que esta troca será aplicada a entrada do 2º flip-flop, fazendo com que este troque de estado a cada descida da saída Q_0 e assim sucessivamente.



Considerando Q_0 como o bit menos significativo (LSB) e Q_3 como o mais significativo (MSB), teremos nas saídas o sistema binário com quatro bits em sequência de 0000 a 1111.

Analisando o gráfico observamos que o período de Q_0 é o dobro do período do clock, portanto sua frequência é metade da do clock, a de Q_1 é metade da de Q_2 e assim sucessivamente. Esse circuito pode ser utilizado como um divisor de frequências de sinais aplicados a entrada de clock.

Nesse caso a divisão será por um número múltiplo de 2^N onde N é o número de flip-flop’s utilizados.

MEMÓRIAS DIGITAIS

São dispositivos que armazenam informações codificadas digitalmente e que podem representar números, letras, caracteres quaisquer, comandos de operações, endereços, etc..

As informações são/estão armazenadas nas memórias em células básicas denominadas “*localidades de memória*” e a cada uma dessas localidades está associado um código de acesso ou “*endereço de memória*”, geralmente um conjunto de dígitos dos sistemas binário ou hexadecimal.

Classificação das Memórias

1- Acesso (*seqüencial ou aleatório*)

Para que se possa acessar uma determinada localidade de memória é necessário que seu endereço seja indicado através de terminais de endereço que pode chegar até a localidade selecionada, diretamente (*acesso aleatório*) ou passando por todas as localidades anteriores até que, seqüencialmente, se chegue a localidade selecionada (*acesso seqüencial*).

No caso do acesso aleatório, o tempo de acesso é menor que para o acesso seqüencial.

2- Volatilidade

Quando se corta a alimentação, as memórias podem ou não manter os dados armazenados. Memórias cujo armazenamento de dados depende da manutenção da alimentação (voláteis) são feitas geralmente a partir de semicondutores, com flip-flop's compondo sua célula básica. (RAM).

Memórias que, mesmo sem alimentação continuam com os dados armazenados, como é o caso das ROM's, são denominadas de não voláteis.

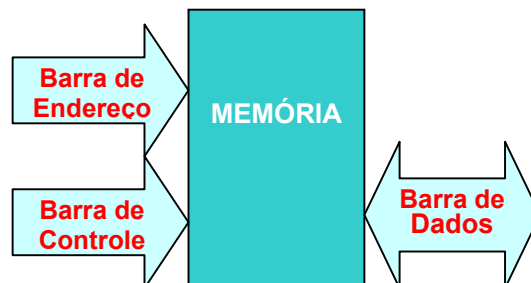
3- Troca de Dados

Podem ser de escrita e leitura (nelas podem ser armazenados ou colhidos dados, como é o caso das RAM's) ou de apenas leitura (pode-se apenas ler dados previamente gravados, como é o caso das ROM's)

4- Tipos de Armazenamento

As memórias podem ser estáticas (quando um dado é inserido ele permanece até que se corte a alimentação) ou dinâmicas (periodicamente os dados precisam ser reinseridos, ou seja deve ser feito um “refresh” na memória.

ESTRUTURA GERAL E ORGANIZAÇÃO DE UMA MEMÓRIA



Uma memória acessa ou armazena as informações digitais mediante endereçamento em lugares denominados “localidades de memória”.

No que se refere à **quantidade de dados armazenados**, as memórias são especificadas pela notação $p \times b$, onde p representa o n.º. de posições ou localidades de memória e b o n.º. de bits da informação armazenada por localidade. Por exemplo, as principais memórias usuais estão organizadas como: 32×8 ; 128×8 ; $1k \times 4$; $64k \times 8$; $2M \times 8$. O n.º. de localidades é sempre múltiplo de 2^n onde n é o n.º. de fios ou terminais.

$$K = 2^{10} = 1.024 \text{ e } M = 2^{20} = 1.048.576,$$

assim uma memória de $64k \times 8$ possui $64 \times 1.024 = 65.536$ localidades, com **8 bits** em cada uma e necessita de **16 terminais** para endereçamento ($64k = 2^6 \cdot 2^{10} = 2^{16}$, ou seja $n = 16$).

A “capacidade da memória” significa o n.º. total de bit's que podem ser armazenados. Uma memória de $64k \times 8$ pode armazenar até **524.288 bits**.

Para que se possa acessar uma determinada localidade de memória é necessário um conjunto de bits denominado “palavra de endereço”, que determina a localidade ou o endereço a ser acessado. Normalmente é utilizado o sistema hexadecimal de numeração (1 dígito hexa = 4 dígitos binário)

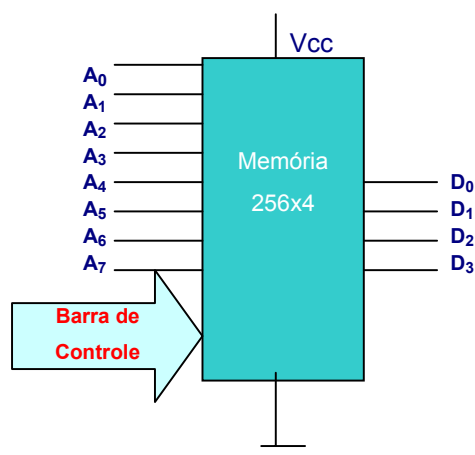
“Mapeamento de Memória”

Uma memória de 256 localidades pode ser mapeada da seguinte forma:

Endereçamento das localidades em binário $A_7 A_6 A_5 A_4 A_3 A_2 A_1 A_0$	Endereçamento das localidades em hexa	Localidades	Conteúdo
0 0 0 0 0 0 0 0	00	L_0	I_0
0 0 0 0 0 0 0 1	01	L_1	I_1
0 0 0 0 0 0 1 0	10	L_2	I_2
.	.	.	.
1 0 1 0 0 1 1 1	A7	L_{167}	I_{167}
1 0 1 0 1 0 0 0	A8	L_{168}	I_{168}
.	.	.	.
1 1 1 1 1 1 1 0	FE	L_{254}	I_{254}
1 1 1 1 1 1 1 1	FF	L_{255}	I_{255}

Esta memória precisa de 8 fios (terminais) para endereçamento ($2^8 = 256$).

Se esta memória for de 256x4, seu esquema será:

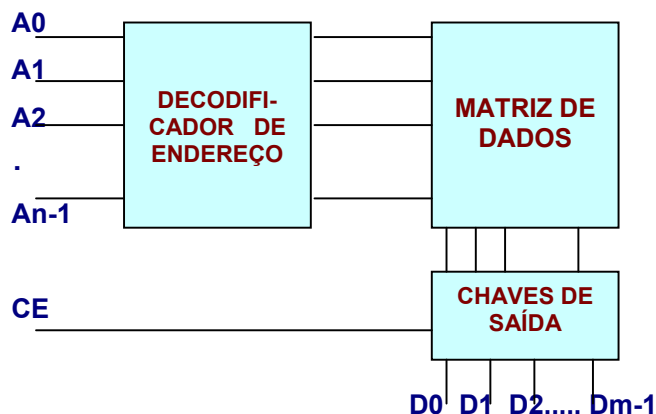


Principais Tipos de Memória

Memória ROM – Read Only Memory

Permite somente a leitura dos dados nela previamente gravados pelo fabricante. Muito utilizada para armazenamento de programas operacionais, circuitos de geração de caracteres ou construção de circuitos combinacionais. São memórias **não voláteis** e possuem **acesso aleatório**.

Arquitetura Interna das Memórias ROM

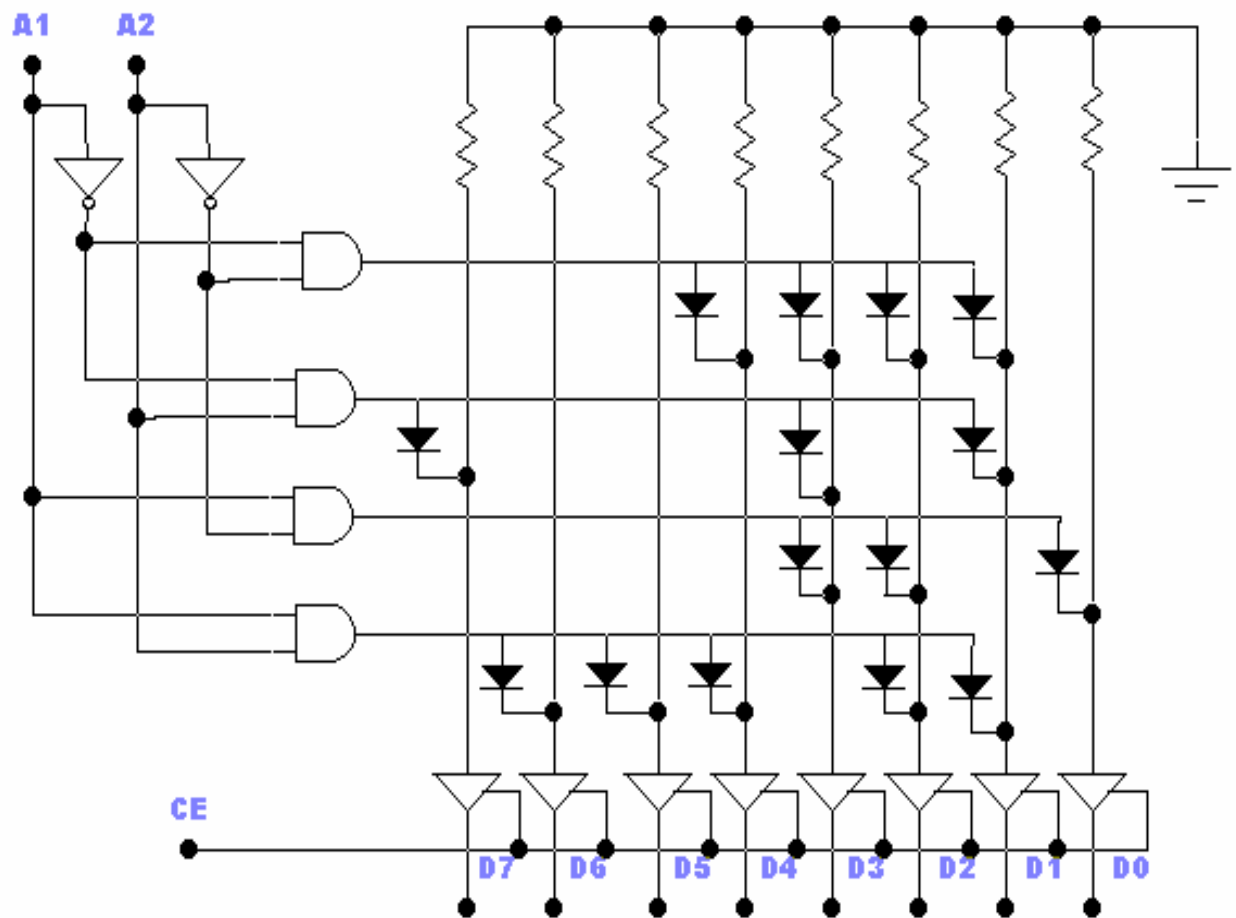


O decodificador de endereços é responsável por ativar uma saída de cada vez (multiplex), em função do endereçamento.

A matriz de dados é um arranjo de linhas e colunas em cuja intercessão é possível ao fabricante, através da colocação adequada de diodos, armazenar dados.

Exemplo – Memória de 4 x 8

Endereço		Dados								
A ₁	A ₀	Hexa	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀
0	0	1E	0	0	0	1	1	1	1	0
0	1	8A	1	0	0	0	1	0	1	0
1	0	0D	0	0	0	0	1	1	0	1
1	1	76	0	1	1	1	0	1	1	0



Variações da Memória ROM

PROM (Programmable Read Only Memory)

Permite o armazenamento de dados pelo próprio usuário, uma única vez. Depois disso torna-se uma ROM

EPROM (Eraseble PROM) ou UVROM

Permite a programação conforme as PROM's com a vantagem de poderem ser apagadas e depois serem reprogramadas.

Isso se dá expondo-se o chip a uma luz ultra-violeta através de um orifício existente no encapsulamento.

É de grande utilidade em microprocessadores para o armazenamento de sistemas operacionais básicos.

EEPROM ou E²PROM (Eletrically EPROM)

Memórias RAM – Random Access Memory

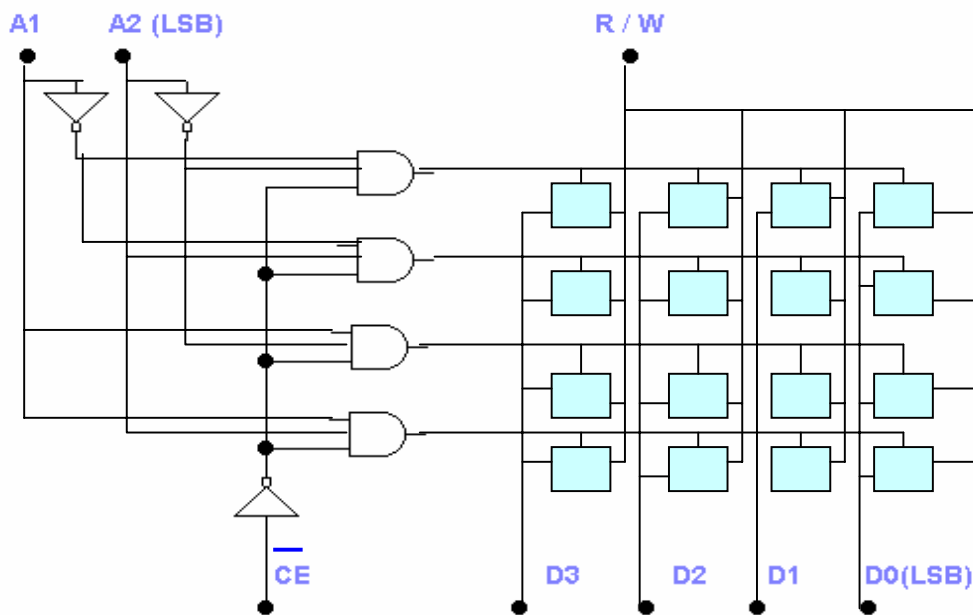
Permitem a escrita e a leitura de dados. São voláteis e também possuem acesso aleatório e, portanto, assim como as ROM's, possuem um tempo de acesso muito reduzido.

São muito utilizadas em equipamentos digitais com dispositivos para armazenamento de forma temporária e que necessitam de acesso rápido.

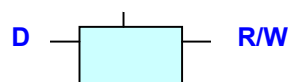
Variações da Memória RAM

SRAM (RAM's Estáticas) - Utilizam um Flip-Flop como célula básica de memória.

DRAM (RAM's Dinâmicas) – Circuitos mais simples que utilizam o efeito capacitivo que aparece internamente no semicondutor e por isso perdem os dados conforme ocorre a descarga dos “capacitores virtuais”. Precisam de um microprocessador para controlar a re-inserção periódica dos dados (circuitos de refresh).



Arquitetura interna das memórias RAM – SRAM 4 x 4

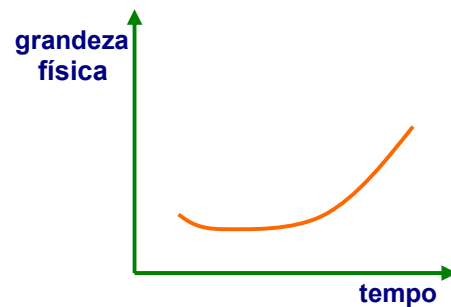


CONVERSORES DIGITAL/ANALÓGICO E ANALÓGICO/DIGITAL

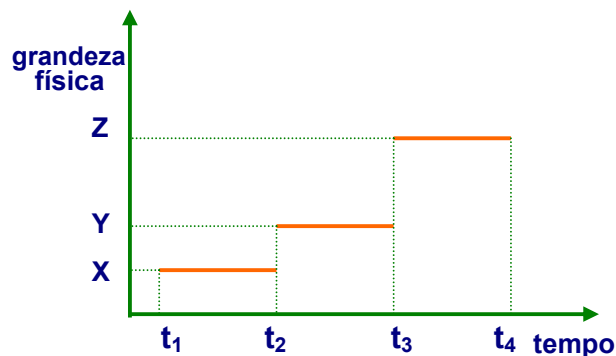
Na natureza, em termos macroscópicos, todos os fenômenos ocorrem de forma contínua (analógica), ou seja, entre dois valores ainda existem sempre infinitos valores. Para que se possa trabalhar utilizando técnicas digitais para processamentos de dados analógicos, são necessários conversores que transformem os sinais analógicos em digitais e, após o processamento, converta os sinais digitais resultantes, novamente em sinais analógicos.

Grandezas Analógicas:

- Velocidade
- Temperatura
- Corrente
- Tensão
- Resistência
- etc.

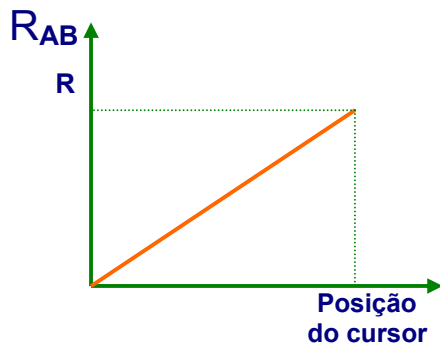
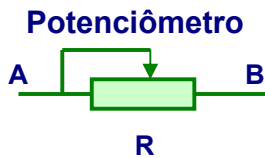


Pode-se representar essas grandezas estabelecendo intervalos de tempo e calculando o valor médio da grandeza a cada um desses intervalos. Assim a grandeza já digitalizada, varia aos saltos em relação ao tempo.

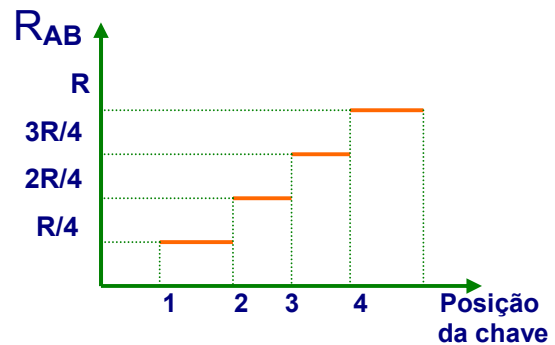
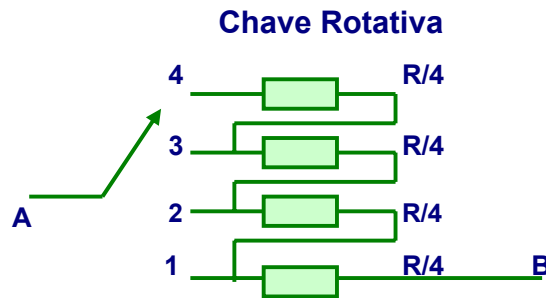


EXEMPLO

Analógico

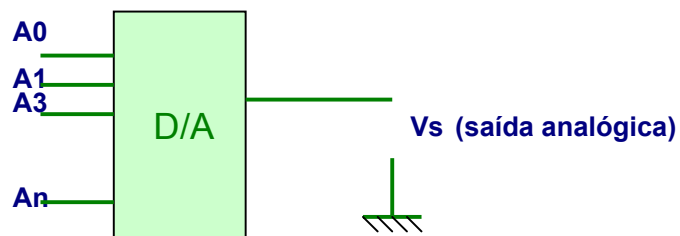


Digital

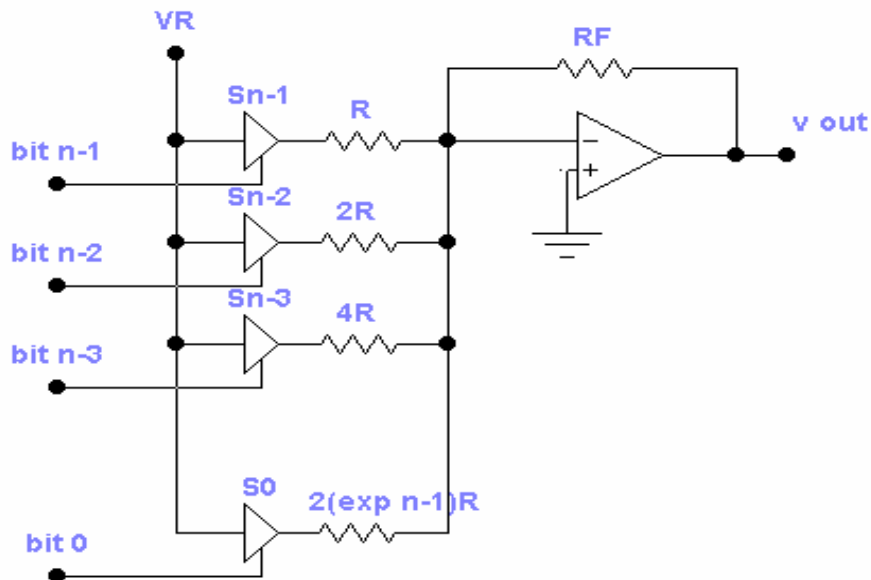


Conversores Digital / Analógico

Faz a conversão de uma variável digital, geralmente codificada em BCD 8421, em uma variável analógica. Nesse caso, uma tensão correspondente ao valor binário da entrada.



Exemplo de um Conversor D/A utilizando Amplificador Operacional com Resistores de valores proporcionais



V_R → Tensão de Referência (estável)

S_0 a S_{n-1} → Buffers (chaves controladas digitalmente)
 Se bit de controle = 0, saída da chave = 0V
 Se bit de controle = 1, saída da chave = $-V_R$

Bit₀ a Bit_{n-1} → Bits de entrada (sinal digital)

V_{out} → Tensão de Saída (“analógica”)

R → Resistores de precisão (valores múltiplos de 2^n)

$$V_{out} = -\frac{R_F}{R} \left(b_{n-1} \cdot V_R + b_{n-2} \cdot \frac{V_R}{2} + b_{n-3} \cdot \frac{V_R}{4} + \dots + b_0 \cdot \frac{V_R}{2^{n-1}} \right)$$

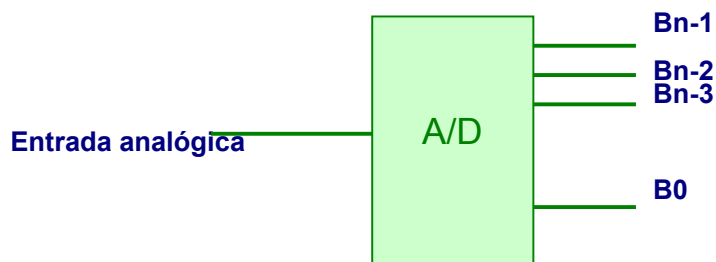
Para 4 bits, todos em nível lógico 1, $V_R = -5V$ e $R_F = R$, temos:

$$V_s = (5 + 5/2 + 5/4 + 5/8) = 9,375V \text{ e}$$

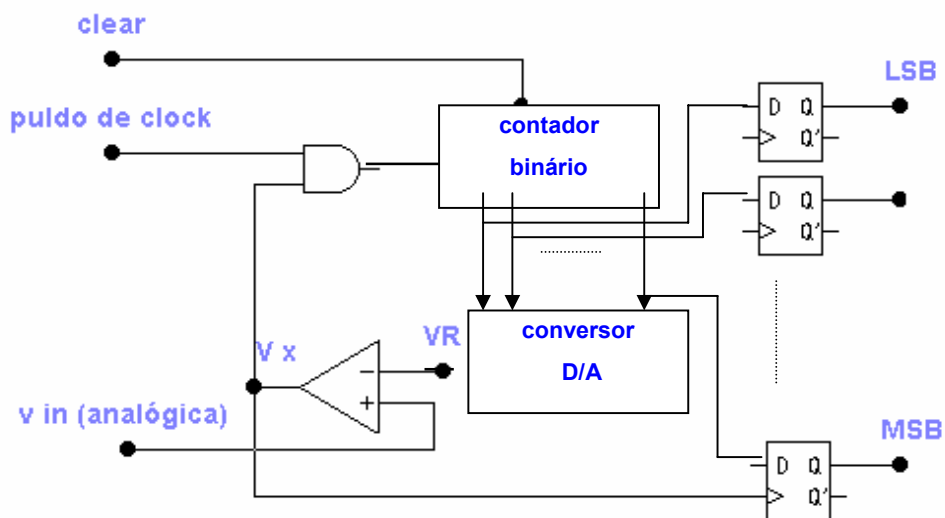
$$\text{Resolução} = 9,375/2^4 = 585 \text{ mV / bit}$$

Conversores Analógico / Digital

Faz a conversão de grandezas analógicas (mundo real) em sinais equivalentes digitais para que possam ser manipuladas através de técnicas digitais



Exemplo de Conversor A/D



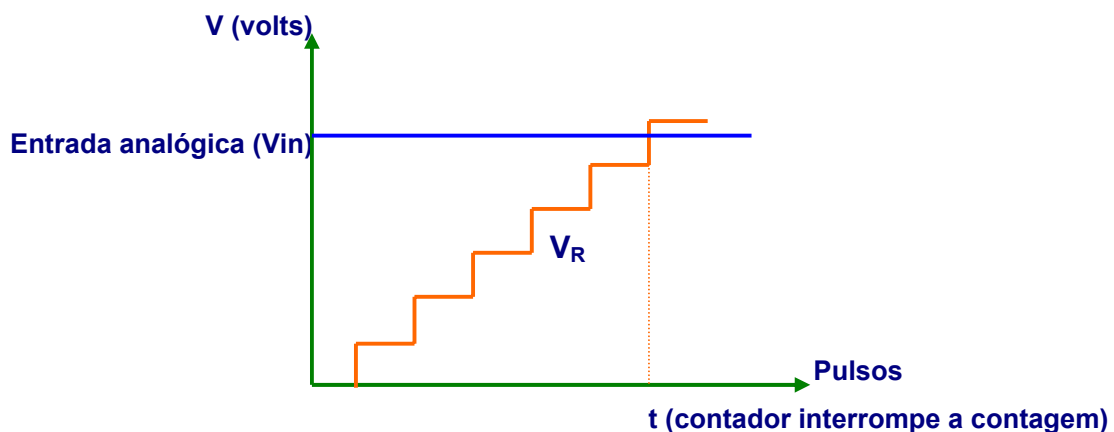
Neste caso, o amplificador operacional está saturado e atua como **comparador**, ou seja, quando a tensão na entrada inversora é maior que a da entrada não inversora, a tensão de saída é zero.

Caso contrário, a tensão de saída é igual à tensão de alimentação Vcc.

Inicialmente, através de um pulso de clear, o contador é zerado.

Neste instante a tensão de saída do conversor D/A (V_R) também será zero, portanto V_X (saída do comparador) é igual a “1” pois $V_R < V_{in}$, assim a porta And libera as entradas do pulso de clock para o contador e os flip-flop's tipo D estão desativados.

Quando a saída V_R ultrapassar V_{in} , a tensão V_X sofrerá uma transição para zero bloqueando a entrada do pulso de clock, liberando os flip-flop's tipo D e finalizando a conversão.



Fatores importantes na escolha de um conversor A/D:

*Tempo Mínimo de Conversão

Este tempo determina a máxima frequência de amostragem do conversor.

Ex.: Para o conversor AD574, o tempo mínimo de conversão é de $25 \mu s$, portanto $f_{amostragem} = 1 / \text{tempo} = 1 / 25 \mu s = 40 KHz$

*Número de Bits

Relacionado com o número de linhas que ele possui no barramento de saída, determina a resolução do conversor.

Ex.: O AD574 possui 12 bits de saída. Supondo que para uma tensão analógica de entrada de 10,24 V, todos os bits de saída estejam em nível lógico “1”, a resolução será:

$$\text{Resol.} = \frac{V_{\text{Smáx}}}{2^n} = \frac{10240\text{mV}}{2^{12}} = \frac{10240\text{mV}}{4096} = 2,5\text{mV/bit}$$

*Valor em Binário do sinal digitalizado

Para determinar o valor em binário (digital) na saída de um conversor para um determinada tensão de entrada deve-se calcular o valor da relação $V_{\text{in}} / \text{Resolução}$ e depois convertê-lo para binário.

Ex.: Para uma tensão de entrada de 5120mV no conversor AD574 que tem resolução de 2,5mV/bit, temos:

$$N_{\text{saída}} = \frac{V_{\text{in}}}{\text{Resol.}} = \frac{5120\text{mV}}{2,5\text{mV/bit}} = 2048$$

Em hexa

$$\begin{array}{r} 2048 \quad | 16 \\ 0 \quad 128 \quad | 16 \\ \quad \quad 0 \quad \quad 8 \end{array}$$

E em binário

$$\begin{array}{ccc} 8 & 0 & 0 \\ 1000 & 0000 & 0000 \end{array}$$

Bit	B ₁₁	B ₁₀	B ₉	B ₈	B ₇	B ₆	B ₅	B ₄	B ₃	B ₂	B ₁	B ₀
Valor Binário	1	0	0	0	0	0	0	0	0	0	0	0

Outros tipos de conversores A/D:

- De aproximação sucessiva
- Integrador por dupla inclinação (R_F é substituído por um capacitor)
- Paralelo

MICROPROCESSADORES

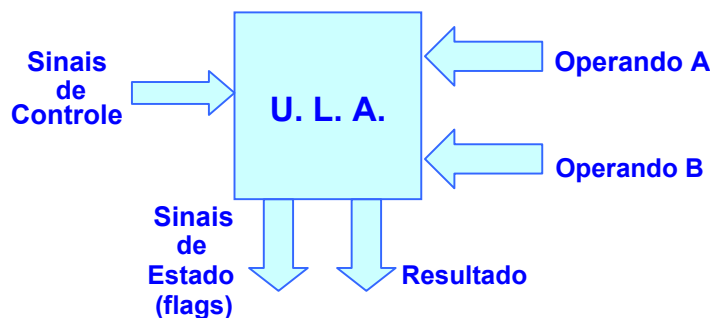
Um microprocessador é constituído de três partes fundamentais:

- Unidade Lógica Aritmética (ULA)
- Seção de Registradores
- Seção de Temporização e Controle



Unidade Lógica – Aritmética

Unidades capazes de executar grande variedade de informações.



Sinais de Controle: Indicam à ULA qual é a função a ser executada com os “operandos”

Resultado: saída (de 8 bits, no caso do 8085) contendo o resultado da operação executada

Flags: trazem as informações adicionais sobre o resultado. Exemplo: flag Z (zero), flag C (carry), etc.

Operações com um Operando

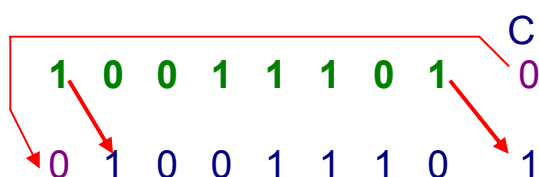
1. **Clear (limpe)** – todos os bits do operando são zerados
2. **Complementação (ou inversão)** – todos os bits do operando são alterados para seu nível lógico complementar
3. **Incremento** – aumenta “um” no operando (contador)
4. **Decremento** – diminui “um” no operando
5. **Deslocamento (shift)** – os bits do operando são deslocados para a direita ou para a esquerda e o bit vago é preenchido com zero

Ex.: Deslocamento à esquerda

Operando antes do deslocamento	1 0 0 1 1 1 0 1
Operando depois do deslocamento	0 0 1 1 1 0 1 0

6. **Rotação (rotate)** – É uma operação de deslocamento modificada na qual o flag C (carry) torna-se parte de um registrador de deslocamento circular, juntamente com o operando

Ex.: Rotação à direita



Operações com dois Operandos

1. Soma – A ULA produz a soma binária de dois operandos. Geralmente um dos operandos vem do acumulador e o outro da memória. O resultado é enviado para o acumulador.

2. Subtração – $A \leftarrow A - M$

A operando do acumulador

M operando da memória ou de outro registrador auxiliar

3. Comparação – É a mesma operação que a da subtração, só que o resultado não é atribuído ao acumulador. É feita para indicar se um operando é igual, maior ou menor que o outro. Vários flags serão afetados.

4. E Lógico – $A \leftarrow A \bullet M$ executa a função “AND”

5 – OU Lógico - $A \leftarrow A + M$ executa a função “OR”

6- OU Exclusivo - $A \leftarrow A \oplus M$

Esta lista de operações, no caso do 8085, não inclui as operações mais complexas como multiplicação, divisão e raízes quadradas, que devem ser programadas à parte.

Microprocessadores mais modernos já processam essas operações dentro da própria ULA.

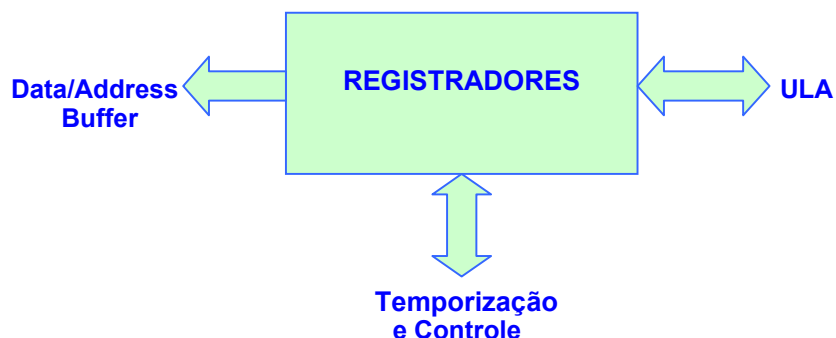
Seção de Registradores

O 8085 possui registradores internos que são similares a posições de memória de leitura e escrita.

O número e o tipo de registradores constituem parte importantíssima na arquitetura de um microprocessador e tem um grande efeito sobre o “esforço” de programação necessário em uma dada aplicação.

A estrutura dos registradores pode variar consideravelmente de μp para μp , porém as funções básicas executadas por eles são as mesmas.

São utilizados para armazenar dados, endereços, códigos de instruções e informações sobre o estado das várias operações dos μp 's, contadores, etc.



O 8085 possui sete registradores de uso geral de 8 bit's (1byte) cada um (*A, B, C, D, E, H, L*)

A

- Acumulador: Pode “acumular” os resultados de uma operação lógica ou aritmética realizada pelo 8085;
- Um operando do acumulador A pode ser manipulado de formas especiais, não permitidas em outros registradores;
- Pode receber dados de periféricos como teclados, chaves, sensores, conversores A/D e D/A

B, C, D, E, H

- Utilizados para armazenar informações para posterior utilização.
- Possuem acesso direto à memória
- Possuem acesso direto pelo programa
- Possuem acesso direto à ULA
- Podem ser endereçados separadamente ou em pares, formando registradores de 16 bits

O 8085 possui ainda três registradores especiais:

- “*Stack Pointer*” – *SP (indicador de pilha)* – registrador de 16 bits utilizado para armazenar endereço de memória de 16 bits (memória de escrita e leitura). “Pilha” é a área das memórias separada para que o 8085 possa armazenar temporariamente as informações dos registradores. O SP contém o endereço da posição de memória que guarda o último dado enviado para a pilha.
- “*Program Counter*” – *PC (contador de programa)* – registrador de 16 bits onde está armazenado o endereço da memória que contém a próxima instrução a ser executada.
- “*Instruction Register*” – *IR (registrador de instruções)* – registrador de 8 bits utilizado para receber a instrução a ser executada, determinada pelo PC

FLAGS

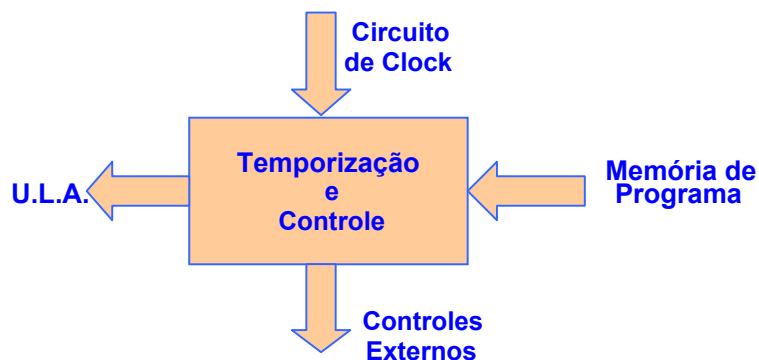
O 8085 possui 5 flags (flip-flop's) que são ajustados conforme as operações lógicas ou aritméticas vão sendo executadas na ULA.:

- **Z (Zero)** – Assume um nível lógico “1” quando o resultado da operação é zero e nível lógico “0”, caso o resultado seja diferente de zero
- **S (Sinal)** – Nível lógico “1” para resultados negativos e Nível lógico “0” para resultados positivos.
- **P (Paridade)** – Nível lógico “1” quando o resultado de uma operação apresentar no. par de bits com nível lógico “1” e Nível lógico “0”, para no. ímpar de bits com nível lógico “1”.
- **C (Carry)** – Quando o resultado de uma operação apresentar um “vai-um” no bit mais significativo, assume nível lógico “1”, caso contrário, nível lógico “0”.
- **AC (Auxiliar Carry)** – Quando o resultado de uma operação apresentar um “vai-um” no bit 3, assume nível lógico “1”, caso contrário, nível lógico “0”.

Seção de Temporização e Controle

Sua função principal é buscar e decodificar instruções na memória de programa e então gerar sinais de controle necessários para que a ULA e a seção dos registradores executem essas instruções.

Além disso gera também sinais de controle externos que são enviados para outros elementos do sistema no qual está inserido (controle de processos, instrumentos, equipamentos, microcomputadores, etc.).



- Cada μp tem seu conjunto único de sinais de controle que são descritos com detalhes nos manuais de operação
- Todos os sinais de temporização e controle são gerados em sincronismo, a partir de um circuito de clock que é conectado ao 8085. O clock do 8085 é de 6,25MHz que, internamente é dividido por dois, gerando 3,125MHz.
- Para executar instruções de entrada ou saída de dados (in/out), o 8085 utiliza 16 linhas de endereço ($A_0 - A_{15}$), 8 linhas de dados ($D_0 - D_7$) e os sinais de controle. Desta forma ele pode endereçar até $2^{16} = 65536$ posições (endereços) que incluem memórias, periféricos e registradores.

Aplicações dos Microprocessadores

Sua principal aplicação é na substituição da lógica convencional fixa, com as vantagens de:

- Ter maior versatilidade, devido à facilidade de implementação de funções extras;
- Ocupar espaço físico muito menor;
- Propiciar projetos menos complexos, mais fáceis de serem implementados e de menor custo;
- Integrar-se facilmente a outros sistemas analógicos através de circuitos de interfaceamento como por exemplo, conversores A/D e D/A

Sendo assim, são amplamente utilizados em sistemas de controle de processos; instrumentos e equipamentos médicos de última geração; instrumentação eletrônica de medida (multímetros digitais, osciloscópios, etc.); sistemas de transmissão de sinais (telefonia fixa e móvel, dados, imagens, áudio, etc.); microcomputadores.

Principais Fabricantes

Intel (principal): 8080, 8085, 8086, 80286, 80386, 80486, Pentium.

Motorola: 6800, 6802, 6809, 68000

Zilog: Z80, Z8000

Signetics: 2650