MINISTÉRIO DA EDUCAÇÃO
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA CATARINENSE
CAMPUS LUZERNA

Engenharia de Controle e Automação

Sistemas Digitais

Professor: Ricardo Kerschbaumer

Aluno: ______________________________

25 de outubro de 2020

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
CATARINENSE

Sumário

Lista de figuras

Aula 1 - Conceitos introdutórios

Nesta aula serão apresentados alguns conceitos necessários ao entendimento dos conteúdos de sistemas digitais. A princípio serão apresentadas as diferenças entre sinais analógicos e sinais digitais. Em seguida serão apresentados os fundamentos das representações dos sinais digitais e as vantagens e desvantagens de sua utilização. Finalmente serão discutidos os circuitos digitais, seus aspectos elétricos, a conversão dos sinais entre o ambiente analógico e o digital e as formas de onda relacionadas.

1.1. Sistemas digitais

Os Sistemas digitais ou circuitos digitais, ou ainda circuitos lógicos são definidos como circuitos eletrônicos que empregam a utilização de sinais elétricos em apenas dois níveis de tensão de forma a definir uma representação de valores binária.

Os sinais elétricos transportados por estes circuitos são chamados de **sinais digitais** ou **sinais binários**.

1.1.1. Sinais digitais e sinais analógicos

Os sistemas digitais são sistemas no qual os sinais têm um número finito de valores discretos, normalmente dois, se contrapondo a sistemas analógicos onde os sinais têm valores pertencentes a um conjunto contínuo ou infinito de valores.

A Figura 1 - Sinal analógico.Figura 1 apresenta um gráfico da tensão em função do tempo, representando um sinal analógico.

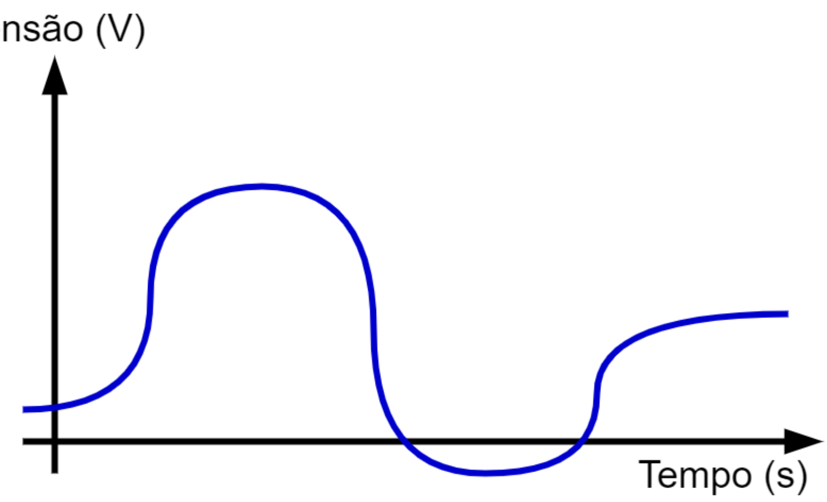

Figura 1 - Sinal analógico.

Nesta figura é possível observar que a amplitude do sinal pode assumir qualquer valor.

A Figura 2 por sua vez apresenta um gráfico da tensão em função do tempo, representando um sinal digital.

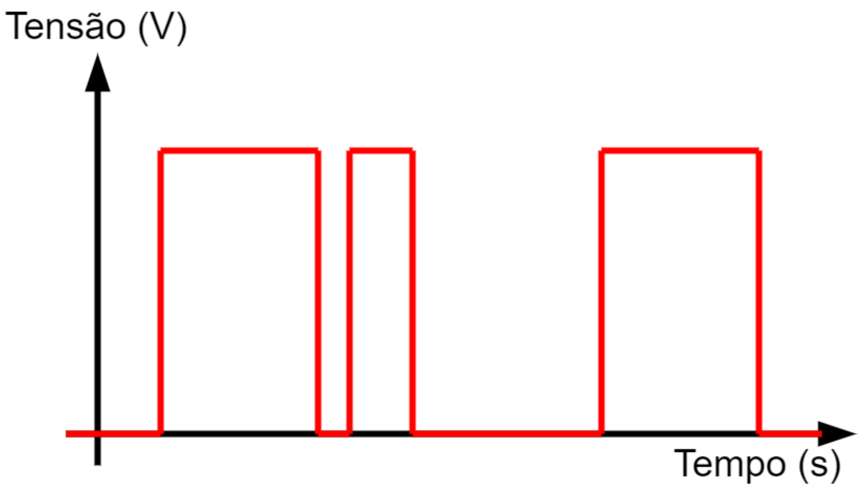

Figura 2 - Sinal digital.

Nesta figura é possível observar que o sinal pode assumir apenas dois valores distintos, um com tensão mais baixa e outro com tensão mais alta.

1.1.2. Representação de sinais digitais

Como já mencionado, sinais digitais costumas assumir dois estados. Assim é possível utilizar sinais digitais para representar fenômenos com as mesmas características. Por exemplo, um sinal digital pode ser utilizado para representar o estado de uma lâmpada, ligada ou desligada.

É comum dizermos que um sinal digital, ou seja, um sinal binário, pode assumir dois estados lógicos. Estes estados têm várias denominações, ligado e desligado, ou alto e baixo, ou ainda verdadeiro e falso. Utilizaremos a seguinte notação para representar os estados de um sinal digital.

O digito 0 (zero) representa o valor falso ou baixo, enquanto o digito 1 (um) representa o valor verdadeiro ou alto.

1.1.3. Porque usar sinais digitais

Os sinais digitais e consequentemente os sistemas digitais são utilizados por apresentarem algumas vantagens sobre os sinais analógicos. As principais vantagens dos sinais digitais são.

- Os sinais digitais são muito mais imunes a distorções, ruídos e interferências.
- Os circuitos digitais são mais confiáveis e robustos.
- Os circuitos digitais são fáceis de projetar e mais baratos.
- A implementação de hardware em circuitos digitais é mais flexível.

1.2. Grandezas digitais

Um único sinal digital pode assumir apenas dois estados, como já mencionado. Porém para a maioria das aplicações, dois estados não são suficientes. Imagine que se deseja contar o número de pessoas em uma sala, seria necessária uma quantidade muito maior de estados para representar este número. Para contornar este tipo de situação utiliza-se não apenas um, mas vários sinais digitais para representar uma determinada grandeza. Neste contexto surgem diferentes nomenclaturas para determinados números de sinais binários agrupados. A nomenclatura utilizada para os sinais digitais é o **bit** (***Bi**nary dig**it***). A seguir são apresentadas as principais notações utilizadas para grupos de bits.

- Um sinal composto por apenas um **bit** é um sinal binário único.
- Um sinal composto por quatro Bits é chamado de **nibble**.
- Um sinal composto por oito Bits é chamado de **byte**.
- Um sinal composto por dezesseis Bits é chamado de **word**.

A Figura 3 mostra graficamente a relação entre os diferentes grupos de Bits.

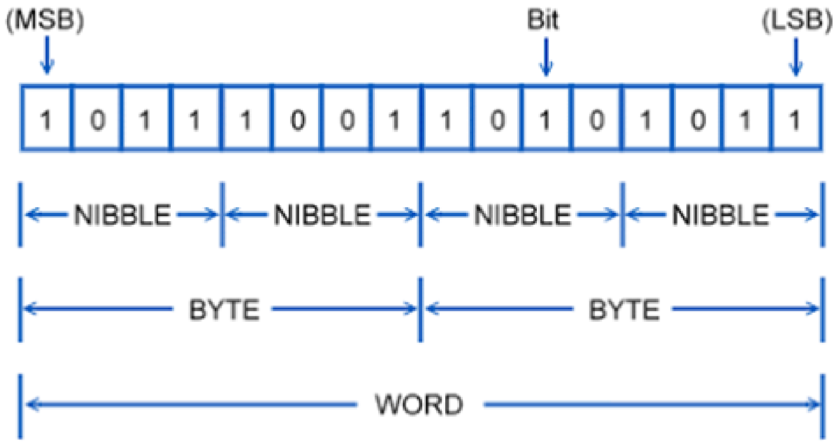

Figura 3 - Grandezas digitais

Quando se trata do armazenamento de informações digitais é comum a utilização de múltiplos para representar grandes quantidades de bits. A seguir são apresentados alguns múltiplos comumente utilizados.

- 1 kilobyte (KB) = 1024 bytes
- 1 megabyte (MB) = 1024 kilobytes
- 1 gigabyte (GB) = 1024 megabytes
- 1 terabyte (TB) = 1024 gigabytes
- 1 petabyte (PB) = 1024 terabytes

Apesar de muito utilizada esta notação não é totalmente correta, pois foi desenvolvida para numeração de base 10 e não para representação binária. Assim, o múltiplo k (kilo), por exemplo, deveria representar 1000 e não 1024. Para evitar confusões foi desenvolvida uma outra nomenclatura especificamente para sistemas digitais. A Figura 4 apresenta os múltiplos de byte.

Profixo Binário (IEC)			Profixo do SI		
Nome	**Símbolo**	**Múltiplo**	**Nome**	**Símbolo**	**Múltiplo**
kibibyte	KiB	2^{10}	kilobyte	kB	10^{3}
mebibyte	MiB	2^{20}	megabyte	MB	10^{6}
gibibyte	GiB	2^{30}	gigabyte	GB	10^{9}
tebibyte	TiB	2^{40}	terabyte	TB	10^{12}
pebibyte	PiB	2^{50}	petabyte	PB	10^{15}

Figura 4 - Múltiplo de byte

1.2.1. Contando em binário

Existem várias formas de representar as informações em formato digital, apenas para ilustrar este conceito a Tabela 1 apresenta a representação dos números de 0 a 9 na forma de bits.

Tabela 1 - Números na forma binária

Números	Informação binária
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

Estas representações serão assunto das próximas aulas.

1.3. Circuitos digitais

Circuitos digitais são a implementação de sistemas digitais na forma de circuitos eletrônicos, onde os sinais digitais são representados por sinais elétricos.

1.3.1. Aspectos elétricos

Para a construção de circuitos eletrônicos digitais é necessário implementar os sinais lógicos 0 e 1 na forma de sinais elétricos. Existem várias formas de fazer isso, as principais serão estudadas mais a frente. A título de exemplo a Figura 5 apresenta os níveis de tensão dos estados lógicos em um sistema digital de 5 V.

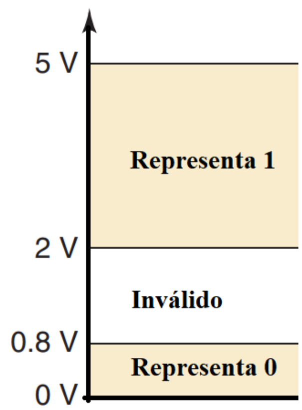

Figura 5 - Níveis de tensão

É comum dizermos que o nível lógico 0 é representado por 0 V e o nível lógico 1 é representado por 5 V, mas na prática é um pouco mais complicado. Como pode ser observado na

Figura 5, existem faixas de tensão que representam cada um dos níveis. Isso torna os sistemas digitais mais robustos e imunes a interferências.

Para simplificar os sinais são representados como na Figura 6.

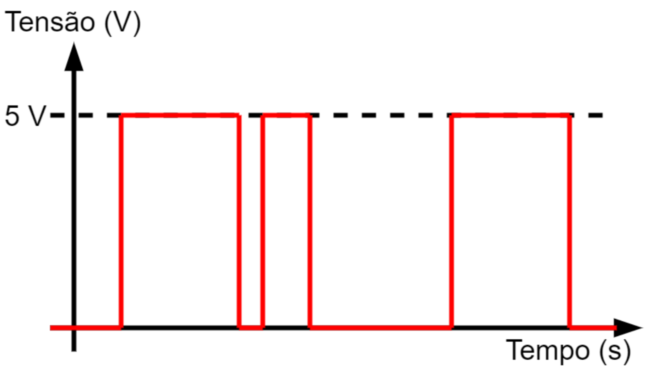

Figura 6 - Sinal digital típico.

Existem muitas outras tecnologias de implementação de circuitos digitais, mas geralmente o nível 0 é representado por uma tensão baixa (perto de 0 V) e o nível 1 é geralmente representado por uma tensão mais alta.

1.3.2. Conversores A/D e D/A

Uma vez que os sinais do mundo físico são analógicos, é necessário convertê-los para sinais digitais e vice-versa sempre que os sinais digitais tenham que interagir com os sinais do meio físico.

A conversão de sinais analógicos em sinais digitais é realizada por um dispositivo eletrônico chamado conversor analógico digital (A/D). A Figura 7 mostra o diagrama de um conversor analógico digital de 8 bits.

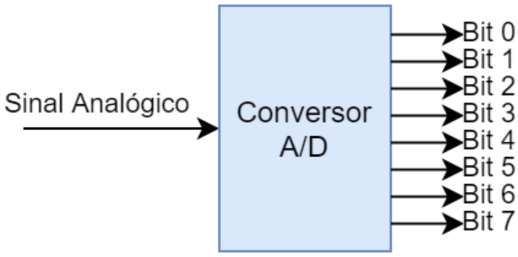

Figura 7 - Conversor Analógico Digital.

Da mesma forma, a conversão de sinais digitais em sinais analógicos é realizada por um dispositivo eletrônico chamado conversor digital analógico (D/A). A mostra o diagrama de um conversor digital analógico de 8 bits.

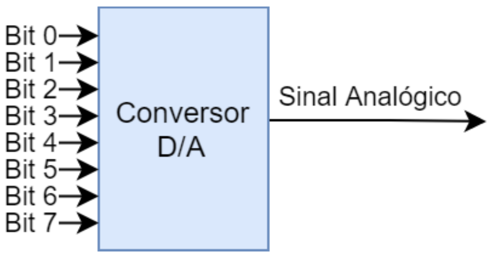

Figura 8 - Conversor Digital Analógico

A maioria dos sistemas digitais é utilizado para processar sinais oriundos de sensores analógicos, como por exemplo, microfones, sensores de temperatura, sensores de luminosidade etc. Assim, estes sinais são primeiro convertidos de analógicos para digitais para só então serem processados. Este processamento é então realizado por circuitos digitais projetados para este fim. Os resultados deste processamento são também sinais digitais, que devem então ser convertidos em sinais analógicos, utilizando um conversor digital analógico. Os sinais analógicos resultantes podem então ser enviado para atuadores, como por exemplo, alto falantes, motores etc.

A Figura 9 apresenta um diagrama de blocos de um sistema de processamento digital de sinais analógicos.

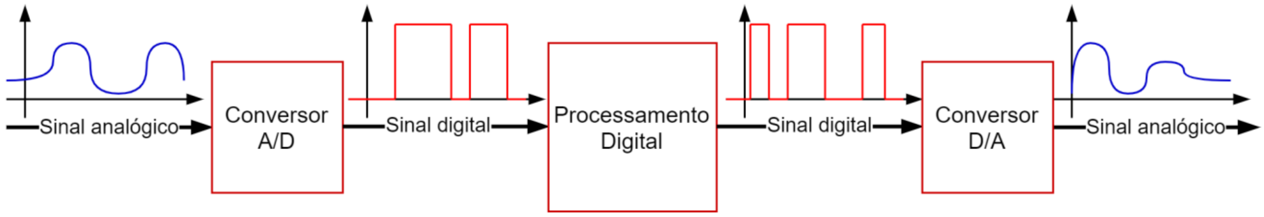

Figura 9 - Processamento digital de sinais

1.3.3. Formas de onda digitais

Nos estudos de sistemas digitais é comum a utilização de gráficos que representam as formas de onda de sinais digitais. Destes gráficos é possível obter várias informações importantes, como a amplitude do sinal, o período e a frequência. A Figura 10 apresenta um gráfico típico de um sinal digital.

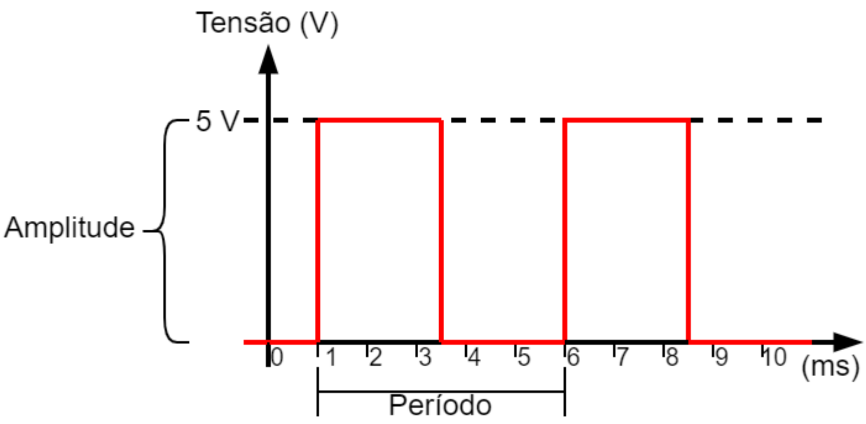

Figura 10 - Forma de onda de sinal digital

No eixo vertical é possível verificar a **amplitude** deste sinal, neste caso 5 V. No eixo horizontal é apresentada a escala de tempo, neste caso em milissegundos. Se o sinal apresentado no gráfico for um sinal repetitivo é possível observar também seu **período**. O período de um sinal é o tempo que ele leva para se repetir. No gráfico da figura p período é de 5 ms ou 0,005 s. Em função do período pode-se calcular a **frequência** do sinal através da seguinte formula.

$$F = \frac{1}{P}$$

Onde *F* é a frequência e *P* o período. No exemplo da figura a frequência é de 200 Hz.

1.4. Exercícios

1) Dado o sinal digital da Figura 11 determine a amplitude, o período e a frequência.

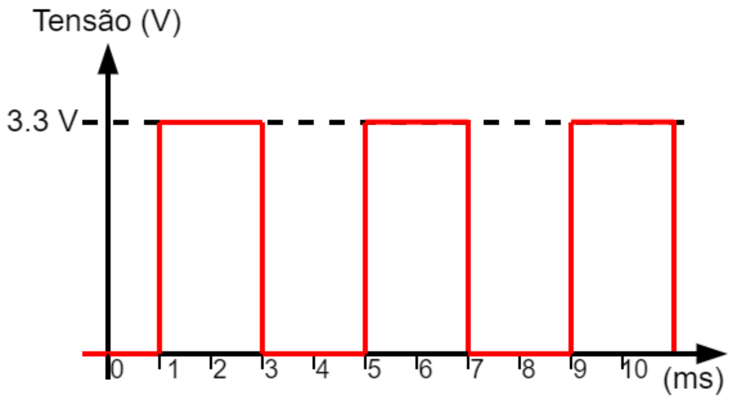

Figura 11 - Forma de onda do exercício 1.

a) Amplitude = _________
b) Período = ___________
c) Frequência = ________

Aula 2 - Códigos e sistemas numéricos

Nesta aula serão abordados assuntos relacionados a forma com que os sistemas digitais codificam as informações. Inicialmente serão abordados os sistemas de numeração, enfatizando o sistema binário, o octal, o hexadecimal e o BCD. Na sequência estudaremos os métodos de conversão entre os diversos sistemas de numeração. O assunto seguinte é a representação de números negativos. Por fim estudaremos a representação de números de ponto flutuante e os códigos alfanuméricos.

2.1. Sistemas de numeração

Os sistemas de numeração são formas de representar os números. Cada sistema de numeração possui um conjunto específico de caracteres. O número de caracteres de cada conjunto é chamado de base do sistema de numeração. A seguir serão apresentados os principais sistemas de numeração relacionados aos sistemas digitais.

2.1.1. Sistema decimal

O sistema decimal é o mais conhecido sistema de numeração e é amplamente empregado em todo o mundo. Este sistema utiliza um conjunto de 10 caracteres ou símbolos (0, 1, 2, 3, 4, 5, 6, 7, 8 e 9) para representar ou números. Estes caracteres são também chamados de dígitos. Como são 10 caracteres que compõe a base do sistema decimal, este sistema é também conhecido como sistema de base 10.

Neste sistema cada dígito possui um peso relacionado a uma potência de sua base, neste caso, a base 10. A Figura 1Figura 12 apresenta a representação do número 5432,789 na base 10.

10^3	10^2	10^1	10^0	10^{-1}	10^{-2}	10^{-3}
5	4	3	2 ,	7	8	9

Figura 12 - Sistema de numeração de base 10.

É possível observar que a direita da vírgula as potências da base são negativas, sinalizando pesos menores do que 1. Já a direita da vírgula as potências da base são positivas. O peso de cada uma das posições ocupadas pelos dígitos é a base elevada a potência relacionada a sua posição.

Para a determinação do valor de um número na base 10 deve se fazer a soma de cada um de seus dígitos multiplicado pelo peso da posição onde ele se encontra. Veja o exemplo a seguir.

$$5 * 10^3 + 4 * 10^2 + 3 * 10^1 + 2 * 10^0 + 7 * 10^{-1} + 8 * 10^{-2} + 8 * 10^{-3} = 5432,789$$

2.1.2. Sistema binário

O sistema binário segue a mesma lógica, porém ele utiliza um conjunto de apenas 2 caracteres ou símbolos (0 e 1) para representar ou números. Assim este sistema de numeração é chamado de sistema de base 2.

Neste sistema de numeração os pesos dos dígitos são todos potências de 2. A Figura 13 apresenta a representação do número 1000110 na base 2. Quando a base de um número é diferente da base 10 é comum colocar-se o número da base subscrita no final do número, assim o número anterior seria 1000110_2.

2^6	2^5	2^4	2^3	2^2	2^1	2^0
64	32	16	8	4	2	1
1	0	0	0	1	1	0

Figura 13 - Sistema de numeração de base 2.

Para a determinação do valor de um número em na base 2 se deve fazer a soma de cada um de seus dígitos (0 ou 1) multiplicado pelo peso da posição onde ele se encontra. Veja o exemplo a seguir.

$$1000110_2 = 1 * 2^6 + 0 * 2^5 + 0 * 2^4 + 0 * 2^3 + 1 * 2^2 + 1 * 2^1 + 0 * 2^0 = 70_{10}$$

Quando se utiliza apenas números inteiros, sem a parte fracionária, o maior número que pode ser representado por um conjunto de bits pode ser determinado pela seguinte expressão.

$$Maior\ Valor = 2^{número\ de\ bits} - 1$$

Assim como no sistema decimal, o sistema binário pode utilizar dígitos a direita da vírgula. A Figura 14 apresenta um exemplo.

2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}
16	8	4	2	1	0,5	0,25
1	0	1	0	1,	1	1

Figura 14 - Número fracionário na base 2

A determinação do valor de um número nestas condições segue a mesma lógica dos anteriores, veja a seguir.

$$10101{,}11_2 = 1 * 2^4 + 0 * 2^3 + 1 * 2^2 + 0 * 2^1 + 1 * 2^0 + 1 * 2^{-1} + 1 * 2^{-2} = 21{,}75_{10}$$

O sistema de numeração binário é o sistema utilizado nos sistemas digitais, daí sua grande importância para nossos estudos.

2.1.3. Sistema octal

O sistema octal possui este nome porque utiliza um conjunto de 8 caracteres ou símbolos (0, 1, 2, 3, 4, 5, 6 e 7) para representar ou números. Assim este sistema de numeração é chamado de sistema de base 8.

Assim como nos sistemas de numeração anteriores, os pesos dos dígitos são todos potências da base, neste caso, 8. A Figura 15 apresenta a representação do número 4205,47 na base 8.

8^3	8^2	8^1	8^0	8^{-1}	8^{-2}
4	2	0	5,	4	7

Figura 15 - Sistema de numeração de base 8.

Para a determinação do valor de um número em na base 8 deve-se, assim como nos anteriores, fazer a soma de cada um de seus dígitos multiplicado pelo peso da posição onde ele se encontra. Veja o exemplo a seguir.

$$4205,47_8 = 4 * 8^3 + 2 * 8^2 + 0 * 8^1 + 5 * 8^0 + 4 * 8^{-1} + 7 * 8^{-2} = 2181,609375_{10}$$

O sistema octal é importante para os estudos de sistemas digitais pois os números (0 a 7) podem ser representados por 3 bits, assim a representação binária de números em formato octal é facilitada. A Figura 16 apresenta como cada número em octal pode ser representado por 3 bits.

4	2	0	5	4	7
100	010	000	101	100	111

Figura 16 - Relação entre octal e binário.

2.1.4. Sistema hexadecimal

O sistema hexadecimal é parecido com o sistema octal, porém utiliza 4 bits cara cada caractere. Assim, o sistema hexadecimal utiliza um conjunto de 16 caracteres ou símbolos (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E e F) para representar ou números. Assim este sistema de numeração é chamado de sistema de base 16. Neste sistema as letras A, B, C, D, E e F correspondem respectivamente aos números decimais 10, 11, 12, 13, 14 e 15, ou seja, A=10, B=11, C=12, D=13, E=14 e F=15.

Assim como nos sistemas de numeração anteriores, os pesos dos dígitos são todos potências da base, neste caso, 16. A Figura 17 apresenta a representação do número 3A4,F2 na base 16.

16^2	16^1	16^0	16^{-1}	16^{-2}
3	A	4,	F	2

Figura 17 - Sistema de numeração de base 16.

Para a determinação do valor de um número em na base 16 deve-se, assim como nos anteriores, fazer a soma de cada um de seus dígitos multiplicado pelo peso da posição onde ele se encontra. Veja o exemplo a seguir.

$$3A4,F2_{16} = 3 * 16^2 + A * 16^1 + 4 * 16^0 + F * 16^{-1} + 2 * 16^{-2} = 932,9453125_{10}$$

O sistema hexadecimal é importante para os estudos de sistemas digitais pois os números (0 a F) podem ser representados por 4 bits, assim a representação binária de números em formato hexadecimal é facilitada. A Figura 18 apresenta como cada número em hexadecimal pode ser representado por 4 bits.

Hexadecimal	3	A	4	F	2
Decimal	3	10	4	15	2
Binário	0011	1010	0100	1111	0010

Figura 18 - Relação entre hexadecimal, decimal e binário.

Para relacionar os diferentes sistemas de numeração, a Figura 19 mostra uma tabela comparativa dos números de 0 a 20.

Decimal	Binário					Octal		Hexa	
0					0		0		0
1					1		1		1
2				1	0		2		2
3				1	1		3		3
4			1	0	0		4		4
5			1	0	1		5		5
6			1	1	0		6		6
7			1	1	1		7		7
8		1	0	0	0	1	0		8
9		1	0	0	1	1	1		9
10		1	0	1	0	1	2		A
11		1	0	1	1	1	3		B
12		1	1	0	0	1	4		C
13		1	1	0	1	1	5		D
14		1	1	1	0	1	6		E
15		1	1	1	1	1	7		F
16	1	0	0	0	0	2	0	1	0
17	1	0	0	0	1	2	1	1	1
18	1	0	0	1	0	2	2	1	2
19	1	0	0	1	1	2	3	1	3
20	1	0	1	0	0	2	4	1	4
Pesos →	16	8	4	2	1	8	1	16	1

Figura 19 - Comparativo entre as bases.

2.1.5. Sistema BCD

O sistema ou código BCD (do inglês ***B**inary **C**oded **D**ecimal*) é uma forma diferente de codificar números decimais em binário. Nesta codificação, cada dígito de um número decimal é codificado separadamente por uma combinação de 4 bits. A seguir são apresentados dois exemplos de codificação no sistema BCD.

37 = “0011” “0111”

459 = “0100” “0101” “1001”

É importante salientar que o sistema BCD se diferencia do sistema binário convencional porque separa os bits em grupos de 4.

2.1.6. Código Gray

O código Gray é também uma forma binária de representa números, porém tem o diferencial de possuir uma distância unitária entre os números. A Figura 20 apresenta os números de 0 a 9 representados através do código Gray.

Decimal	Gray			
0	0	0	0	0
1	0	0	0	1
2	0	0	1	1
3	0	0	1	0
4	0	1	1	0
5	0	1	1	1
6	0	1	0	1
7	0	1	0	0
8	1	1	0	0
9	1	1	0	1

Figura 20 - Código Gray

Observando a figura é possível notar que de um número para outro apenas um bit muda. Este código é mais utilizado em sistemas eletromecânicos, onde a comutação das chaves consome mais energia e produz ruídos. Assim, o uso do código Gray garante que qualquer mudança altera apenas um bit, minimizando o consumo de energia e o ruído.

2.2. Conversão entre bases

É comum que necessitemos converter um número em uma determinada base em seu valor equivalente em outra base. Nas seções anteriores foi apresentado que para converter um número em qualquer base para a base decimal, deve-se fazer a soma de cada um de seus dígitos multiplicado pelo peso da posição onde ele se encontra. As seções a seguir apresentas os procedimentos para realizar as conversões de números na base decimal para outras bases.

2.2.1. Converter inteiro decimal para outras bases

Para converter números inteiros da base decimal para qualquer outra base se utiliza um método chamado método de divisões sucessivas (DS). O funcionamento deste método é simples, basta ir dividindo sucessivamente o número inteiro decimal pela base (b) que se deseja utilizar. O resultado é a composição de todos os restos parciais das divisões. A seguir são apresentados os passos para a aplicação do método.

1) Efetue a divisão do número inteiro decimal (N) pela base (b) de forma a obter o quociente (Q_1) e o resto (R_1). O resto (R_1) e o quociente (Q_1) devem ser colocados respectivamente embaixo do número inteiro decimal (N) e da base (b), veja a Figura 21.

$$\begin{array}{c|c} N & b \\ \cline{2-2} R_1 & Q_1 \end{array}$$

Figura 21 - Primeiro passo da divisão sucessiva.

2) Repita o passo anterior dividindo o quociente (Q_1) pela base (b) para obter o quociente (Q_2) e o resto (R_2), veja a Figura 22.

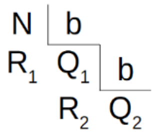

Figura 22 - Segundo passo da divisão sucessiva.

Este processo deve ir se repetindo até que seja encontrado um quociente (Q_n) com valor menor do que a base (b), veja a Figura 23.

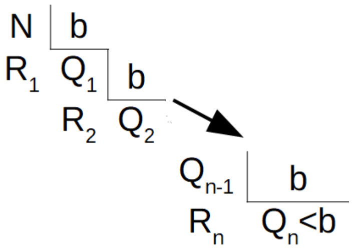

Figura 23 - Conclusão do método da divisão sucessiva.

A obtenção do resultado acontece da seguinte forma. O número inteiro na base b (I_b) é obtido do último quociente (Q_n) e dos restos obtidos nas divisões sucessivas conforme a expressão a seguir.

$$\mathbf{I_b = (Q_n\ R_n\ R_{n-1}\ R_{n-2} \ldots R_1)_b}$$

Para facilitar o entendimento do método, vejamos alguns exemplos. A Figura 24 apresenta um exemplo de conversão de um inteiro decimal para a base 2 (binário).

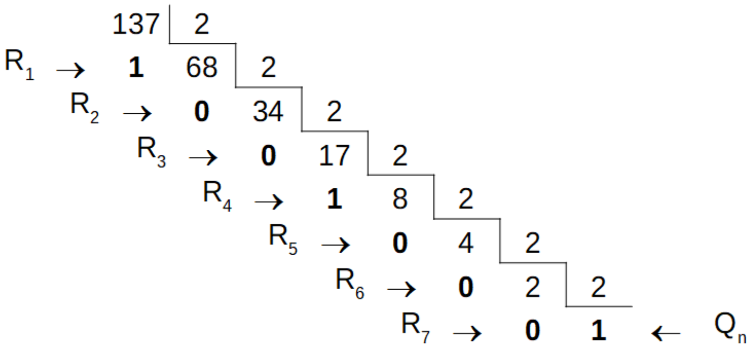

Figura 24 - Exemplo 1 de divisão sucessiva.

Neste exemplo o número inteiro decimal 137_{10} é convertido para a base 2. O valor binário resultado é 10001001_2. Para comprovar este resultado pode-se fazer a soma de cada um dos dígitos multiplicado pelo peso da posição onde ele se encontra, veja a Figura 25.

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1
1	0	0	0	1	0	0	1

Figura 25 - Verificação do resultado da conversão.

Observando a informação em formato binário e os respectivos pesos obtém-se a seguinte expressão, onde o resultado pode ser comprovado.

$$10001001_2 = 1 * 2^7 + 0 * 2^6 + 0 * 2^5 + 0 * 2^4 + 1 * 2^3 + 0 * 2^2 + 0 * 2^1 + 1 * 2^0 = 137_{10}$$

Para um segundo exemplo faremos a conversão do número 9418_{10} para a base 16 (Hexadecimal). Veja a Figura 26.

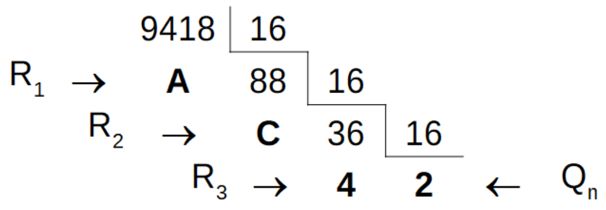

Figura 26 - Exemplo 2 de divisão sucessiva.

O número inteiro decimal 9418_{10} é convertido para a base 16 fica $24CA_{16}$.

Para comprovar este resultado pode-se fazer a soma de cada um dos dígitos multiplicado pelo peso da posição onde ele se encontra, veja a Figura 27.

16^3	16^2	16^1	16^0
2	4	C	A

Figura 27 - Verificação do resultado da conversão.

Observando a informação em formato hexadecimal e os respectivos pesos obtém-se a seguinte expressão, onde o resultado pode ser comprovado.

$$24CA_{16} = 2 * 16^3 + 4 * 16^2 + C * 16^1 + A * 16^0 = 9418_{10}$$

É importante salientar que este método se aplica a números inteiros, em caso de números com parte fracionária, outros métodos devem ser empregados.

2.3. Números negativos

Os sistemas digitais também devem ser capazes de operar com números negativos, assim é necessária uma forma de representar digitalmente estes números.

A seguir serão apresentados três métodos de representação de números negativos na forma de bits.

2.3.1. Sinal e magnitude

A primeira forma de representar digitalmente números negativos é através do sinal e da magnitude, também conhecida como sinal e módulo. Nesta representação, um bit adicional é adicionado a esquerda do número, em sua forma binária tradicional, para representar o sinal. Assim, se este bit adicional for 0 o número é positivo, e se for 1 o número é negativo.

A Figura 28 apresenta os números 11_{10}, -11_{10}, 9_{10} e -9_{10} no formato de sinal e magnitude. Observe que o primeiro bit representa o sinal. Quando este bit é 1 o número é negativo.

		Sinal	Magnitude			
11_{10}	=	**0**	1	0	1	1
-11_{10}	=	**1**	1	0	1	1
9_{10}	=	**0**	1	0	0	1
-9_{10}	=	**1**	1	0	0	1

Figura 28 - Representação por sinal e magnitude.

A notação de sinal e magnitude é a forma mais simples de representação de números negativos, porém esta representação não favorece a realização de operações aritméticas com estes números. Esta notação também possui duas representações para o zero o 0_2 e o -0_2.

2.3.2. Complemento de um

Complemento de um é outra forma de representar números negativos na forma binária. Nesta representação a combinação de bits que representa o valor negativo é obtida aplicando-se o complemento (inversão) bit a bit no valor positivo representado em sinal e magnitude. A Figura 29 apresenta os números 11_{10}, -11_{10}, 9_{10} e -9_{10} no formato de complemento de um. Observe que para números negativos todos os bits são invertidos, inclusive o sinal.

		Sinal	Magnitude			
11_{10}	=	**0**	1	0	1	1
-11_{10}	=	**1**	0	1	0	0
9_{10}	=	**0**	1	0	0	1
-9_{10}	=	**1**	0	1	1	0

Figura 29 - Representação em complemento de um.

A representação em complemento de um facilita a construção de circuitos aritméticos digitais, porém ainda tem o inconveniente de duas representações para o valor zero, o 0_2 e o -0_2.

2.3.3. Complemento de dois

A representação de números negativos em formato binário mais comum nos sistemas digitais atuais é o complemento de dois. Nesta notação o valor negativo é obtido aplicando-se o complemento (inversão) bit a bit no valor positivo representado em sinal e magnitude, em seguida soma-se 1 a este valor. A Figura 30 apresenta os números 11_{10}, -11_{10}, 9_{10} e -9_{10} no formato de complemento de dois.

		Sinal	Magnitude			
11_{10}	=	**0**	1	0	1	1
-11_{10}	=	**1**	0	1	0	1
9_{10}	=	**0**	1	0	0	1
-9_{10}	=	**1**	0	1	1	1

Figura 30 - Representação em complemento de dois.

Observando o exemplo pode-se notar que o número 11_{10} tem sua representação em binário igual a 1011_2, para encontrar o valor binário em complemento de dois para o número -11_2 inicialmente adiciona-se um bit de sinal a representação do valor com sinal positivo, assim, chega-se a 01011_2. O próximo passo é inverter todos os bits, o resultado é 10100_2. A última etapa é somar 1 a este valor, o resultado é $\mathbf{10101_2}$.

A representação em complemento de dois é amplamente utilizar por apresentar algumas vantagens relevantes. Os circuitos digitais para fazer a adição e a subtração são muito simples podendo inclusive ser unificados.

Outra vantagem desta notação é que o número zero apresenta apenas uma forma de representação.

A título de exemplo a Figura 31 apresenta os números de -8_{10} a 7_{10} na forma binária sem sinal e em complemento de dois.

Decimal	Binário (Sinal + 3 bits)	
	Sem sinal	Complemento de 2
-8	-	1000
-7	-	1001
-6	-	1010
-5	-	1011
-4	-	1100
-3	-	1101
-2	-	1110
-1	-	1111
0	000	0000
1	001	0001
2	010	0010
3	011	0011
4	100	0100
5	101	0101
6	110	0110
7	111	0111

Figura 31 - Números negativos em complemento de dois.

2.4. Ponto flutuante

As técnicas de representação de números em sistemas digitais que foram estudadas até aqui são úteis para números inteiros, porém em muitas circunstâncias é necessário o uso de números reais. Para representar números reais, os sistemas digitais atuais utilizam um padrão chamado IEEE 754. Nas seções a seguir será apresentado o básico deste padrão

2.4.1. O padrão IEEE 745

Este padrão apresenta duas opções para a codificação dos números reais. A primeira se chama **ponto flutuante de precisão simples**, e possui 32 bits de tamanho. O primeiro bit (mais significativo) é dedicado a representação do sinal (S). Os 8 bits seguintes são dedicados ao expoente (E). Os 23 bits restantes são dedicados a representação da fração. A Figura 32 apresenta esta estrutura. Observe que o número de bits é sempre 32, independente do valor que se esteja representando.

$2^{7}\ 2^{6}\ 2^{5}\ 2^{4}\ 2^{3}\ 2^{2}\ 2^{1}\ 2^{0}\ 2^{-1}\ 2^{-2}\ 2^{-3}\ 2^{-4}\ 2^{-5}\ 2^{-6}\ 2^{-7}\ \ldots\ 2^{-22}\ 2^{-23}$

S	Expoente (E)	Parte fracionária (F)
1 bit	8 bits	23 bits

Figura 32 - Ponto flutuante de precisão simples.

Para a representação de números neste formado é considerada a representação em notação científica normalizada, ou seja, com exatamente um dígito diferente de zero antes do ponto binário.

Para que se possa obter o valor numérico correspondente a representação em ponto flutuante de precisão simples deve-se fazer a seguinte operação.

$$Valor = -1^{S} \cdot (1 + F) \cdot 2^{E-127}$$

Onde $1 \leq E \leq 254$

Existe também no padrão IEEE 754 uma representação de ponto flutuante de 64 bits. Esta representação é chamada de **ponto flutuante de precisão dupla**. A Figura 33 apresenta esta estrutura.

$2^{10}\ 2^{9}\ 2^{8}\ 2^{7}\ 2^{6}\ 2^{5}\ 2^{4}\ 2^{3}\ 2^{2}\ 2^{1}\ 2^{0}\ 2^{-1}\ 2^{-2}\ 2^{-3}\ 2^{-4}\ 2^{-5}\ 2^{-6}\ 2^{-7}\ \ldots\ 2^{-51}\ 2^{-52}$

S	Expoente (E)	Parte fracionária (F)
1 bit	11 bits	52 bits

Figura 33 - Ponto flutuante de precisão dupla

Nesta notação, para que se possa obter o valor numérico correspondente deve-se fazer a seguinte operação.

$$Valor = -1^{S} \cdot (1 + F) \cdot 2^{E-1023}$$

Onde $1 \leq E \leq 2046$

O termo E sofre um deslocamento tanto na representação de precisão simples como na representação de precisão dupla. Este deslocamento permite expressar expoentes positivos ou negativos, o que por sua vez permite representar números muito grandes ou muito pequenos.

Quando se trabalha com pontos flutuantes normalizados no padrão IEEE 754 deve levar em conta os seguintes fatores:

- No termo (1 + F) o “1” não aparece no valor binário, pois se assume que o número está em notação científica com exatamente um dígito diferente de zero antes do ponto binário.
- As equações não permitem a representação do número zero, assim, o padrão é preencher os campos E e F com zeros para representar o número zero.
- É possível representar o infinito preenchendo o campo E com 1’s e o campo F com 0’s, com o sinal apropriado.

- Quando o campo E é preenchido com 1's e o campo F é diferente de zero tem-se uma situação de número inválido, indicado por NaN (***N**ot **a** **N**umber*). Esta notação é útil para representar resultados inválidos como zero dividido por zero por exemplo.
- Representações com E igual a zero e F diferente de zero, indicam números não normalizados e devem ser evitadas.

A Figura 34 apresenta uma tabela que resume as possíveis representações no padrão IEEE 754. O valor "max" é 255 para números de precisão simples e 2047 para números de precisão dupla.

Sinal (S)	Expoente (E)	Parte Fracionária (F)	Valor
0 / 1	0	0	+0 / -0
0 / 1	max	0	$+\infty$ / $-\infty$
0 / 1	max	$\neq 0$	NaN
0 / 1	0	$\neq 0$	Não normalizado
0 / 1	De 1 a max -1	Qualquer	Normalizado

Figura 34 - Possíveis representações no padrão IEEE 754

Vejamos como exemplo a representação do número $-12{,}25_{10}$. A Figura 35 apresenta a representação deste número em ponto flutuante de precisão simples.

	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}	...	2^{-22}	2^{-23}
S	Expoente (E)								Parte fracionária (F)									
1	1	0	0	0	0	0	1	0	1	0	0	0	1	0	0	...	0	0

Figura 35 - Exemplo de ponto flutuante de precisão simples.

Observado as informações binárias e seus respectivos pesos temos:

$$S = 1$$

$$E = 1 \cdot 2^7 + 1 \cdot 2^1 = 130$$

$$F = 1 \cdot 2^{-1} + 1 \cdot 2^{-5} = 0{,}53125$$

Aplicando estes valores na fórmula tem-se:

$$Valor = -1^1 \cdot (1 + 0{,}53125) \cdot 2^{130-127} = -12{,}25$$

2.5. Código alfanumérico

Sistemas digitais também necessitam operar com informações na forma de textos, assim é necessário codificar na forma binária também letras. A codificação deste tipo de informações é chamada de codificação alfanumérica.

Os códigos alfanuméricos, também chamados de códigos de caracteres (letras), são códigos binários usados para representar dados como letras do alfabeto, números (digitos), símbolos matemáticos e sinais de pontuação, de uma forma que é compreensível e processável por um computador.

Este tipo de codificação é utilizada por dispositivos de entrada e saída, como teclados, monitores, impressoras bem como para a transmissão de informações pelas redes de computadores por exemplo.

Existem várias codificações em uso atualmente, como ASCII, Unicode, UTF8 etc. Um dos mais simples e mais utilizado é o ASCII, que será abordado a seguir.

2.5.1. Código ASCII

O Código ASCII (**A**merican **S**tandard **C**ode for **I**nformation **I**nterchange) é um código muito popular usado em todos os sistemas digitais, ele utiliza 7 bits para representar 128 caracteres.

Os primeiros 32 (0 a 31) caracteres são ditos caracteres não imprimíveis, e servem como comando de controle para os periféricos. Os caracteres de 32 a 127 representam os símbolos letras e dígitos utilizados normalmente em nosso dia a dia. A Figura 36 apresenta a tabela ASCII, que contém todos estes 128 caracteres.

Decimal	Binário	Símbolo	Decimal	Binário	Símbolo	Decimal	Binário	Símbolo	Decimal	Binário	Símbolo
0	0	NUL	32	100000	espaço	64	1000000	@	96	1100000	`
1	1	SOH	33	100001	!	65	1000001	A	97	1100001	a
2	10	STX	34	100010	"	66	1000010	B	98	1100010	b
3	11	ETX	35	100011	#	67	1000011	C	99	1100011	c
4	100	EOT	36	100100	$	68	1000100	D	100	1100100	d
5	101	ENQ	37	100101	%	69	1000101	E	101	1100101	e
6	110	ACK	38	100110	&	70	1000110	F	102	1100110	f
7	111	BEL	39	100111	'	71	1000111	G	103	1100111	g
8	1000	BS	40	101000	(	72	1001000	H	104	1101000	h
9	1001	HT	41	101001	)	73	1001001	I	105	1101001	i
10	1010	LF	42	101010	*	74	1001010	J	106	1101010	j
11	1011	VT	43	101011	+	75	1001011	K	107	1101011	k
12	1100	FF	44	101100	,	76	1001100	L	108	1101100	l
13	1101	CR	45	101101	-	77	1001101	M	109	1101101	m
14	1110	SO	46	101110	.	78	1001110	N	110	1101110	n
15	1111	SI	47	101111	/	79	1001111	O	111	1101111	o
16	10000	DLE	48	110000	0	80	1010000	P	112	1110000	p
17	10001	DC1	49	110001	1	81	1010001	Q	113	1110001	q
18	10010	DC2	50	110010	2	82	1010010	R	114	1110010	r
19	10011	DC3	51	110011	3	83	1010011	S	115	1110011	s
20	10100	DC4	52	110100	4	84	1010100	T	116	1110100	t
21	10101	NAK	53	110101	5	85	1010101	U	117	1110101	u
22	10110	SYN	54	110110	6	86	1010110	V	118	1110110	v
23	10111	ETB	55	110111	7	87	1010111	W	119	1110111	w
24	11000	CAN	56	111000	8	88	1011000	X	120	1111000	x
25	11001	EM	57	111001	9	89	1011001	Y	121	1111001	y
26	11010	SUB	58	111010	:	90	1011010	Z	122	1111010	z
27	11011	ESC	59	111011	;	91	1011011	[	123	1111011	{
28	11100	FS	60	111100	<	92	1011100	\	124	1111100	|
29	11101	GS	61	111101	=	93	1011101	]	125	1111101	}
30	11110	RS	62	111110	>	94	1011110	^	126	1111110	~
31	11111	US	63	111111	?	95	1011111	_	127	1111111	Delete

Figura 36 - A tabela ASCII.

Além dos 128 caracteres da tabela ASCII padrão existem mais 128 caracteres que são uma versão expandida da tabela ASCII. Esta versão expandida contempla os caracteres de 128 a 255. Existem diversas versões desta expansão, estas versões são dependentes do idioma do sistema por exemplo. A Figura 37 apresenta os caracteres de 128 a 255 da tabela ASCII utilizada em um computador rodando Windows 10 em português do Brasil.

Decimal	Binário	Símbolo	Decimal	Binário	Símbolo	Decimal	Binário	Símbolo	Decimal	Binário	Símbolo
128	10000000	Ç	160	10100000	á	192	11000000	└	224	11100000	Ó
129	10000001	ü	161	10100001	í	193	11000001	┴	225	11100001	ß
130	10000010	é	162	10100010	ó	194	11000010	┬	226	11100010	Ô
131	10000011	â	163	10100011	ú	195	11000011	├	227	11100011	Ò
132	10000100	ä	164	10100100	ñ	196	11000100	─	228	11100100	õ
133	10000101	à	165	10100101	Ñ	197	11000101	┼	229	11100101	Õ
134	10000110	å	166	10100110	ª	198	11000110	ã	230	11100110	µ
135	10000111	ç	167	10100111	º	199	11000111	Ã	231	11100111	þ
136	10001000	ê	168	10101000	¿	200	11001000	╚	232	11101000	Þ
137	10001001	ë	169	10101001	®	201	11001001	╔	233	11101001	Ú
138	10001010	è	170	10101010	¬	202	11001010	╩	234	11101010	Û
139	10001011	ï	171	10101011	½	203	11001011	╦	235	11101011	Ù
140	10001100	î	172	10101100	¼	204	11001100	╠	236	11101100	ý
141	10001101	ì	173	10101101	¡	205	11001101	═	237	11101101	Ý
142	10001110	Ä	174	10101110	«	206	11001110	╬	238	11101110	¯
143	10001111	Å	175	10101111	»	207	11001111	¤	239	11101111	´
144	10010000	É	176	10110000	░	208	11010000	ð	240	11110000	
145	10010001	æ	177	10110001	▒	209	11010001	Ð	241	11110001	±
146	10010010	Æ	178	10110010	▓	210	11010010	Ê	242	11110010	‗
147	10010011	ô	179	10110011	│	211	11010011	Ë	243	11110011	¾
148	10010100	ö	180	10110100	┤	212	11010100	È	244	11110100	¶
149	10010101	ò	181	10110101	Á	213	11010101	ı	245	11110101	§
150	10010110	û	182	10110110	Â	214	11010110	Í	246	11110110	÷
151	10010111	ù	183	10110111	À	215	11010111	Î	247	11110111	¸
152	10011000	ÿ	184	10111000	©	216	11011000	Ï	248	11111000	°
153	10011001	Ö	185	10111001	╣	217	11011001	┘	249	11111001	¨
154	10011010	Ü	186	10111010	║	218	11011010	┌	250	11111010	·
155	10011011	ø	187	10111011	╗	219	11011011	█	251	11111011	¹
156	10011100	£	188	10111100	╝	220	11011100	▄	252	11111100	³
157	10011101	Ø	189	10111101	¢	221	11011101	¦	253	11111101	²
158	10011110	×	190	10111110	¥	222	11011110	Ì	254	11111110	■
159	10011111	ƒ	191	10111111	┐	223	11011111	▀	255	11111111	

Figura 37 - Tabela ASCII expandida.

Para aplicações mais complexas, como por exemplo, navegadores de internet ou editores de texto, codificações mais complexas são utilizadas para representar digitalmente os caracteres. Estas codificações permitem por exemplo a codificação de caracteres dos alfabetos orientais.