Conceitos Básicos

BITs e BYTEs

- Bit = BInary digiT = vale sempre 0 ou 1 - elemento básico de informação
- Byte = 8 bits processados em paralelo (ao mesmo tempo)
- Word = n bytes (depende do processador em questão)
- Double word = 2 words
- Nibble = 4 bits (utilidade para BCD)

- Posição de bits:

Para 1 byte:

7	6	5	4	3	2	1	0
0	1	0	1	0	1	0	1

Para 1 word (de 16 bits):

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

byte alto (high byte) | byte baixo (low byte)

Little Endian X Big Endian

Words são armazenados em bytes consecutivos, em memórias de largura de 8 bits.

Exemplo:

1025_{10} = 00000000 00000000 00000100 00000001_2

Endereço	Representação Big-Endian (MOTOROLA)	Representação Little-Endian (INTEL)
00	00000000	00000001
01	00000000	00000100
02	00000100	00000000
03	00000001	00000000

Memória

- **Memória: local do computador (hardware) onde se armazenam temporária ou definitivamente dados (números, caracteres e instruções)**
- **Posição de memória ou endereço: localidade física da memória onde se encontra o dado.**
- **Organização da memória:**

Endereço	Conteúdo
...	...
4MB	10110101
...	...
1048576	01001010
...	...
1765	01001101
...	...
4	01010000
3	11111111
2	11101001
1	11011010
0	01100100

ORGANIZAÇÃO DE SISTEMAS DE COMPUTAÇÃO

Representação binária de números não sinalizados

Qualquer número em qualquer base ➔ $N = \sum_{i=0}^{n-1} d_i \text{ X } base^i$

Ex. $1025_{10} = 1 \text{ X } 10^3 + 0 \text{ X } 10^2 + 2 \text{ X } 10^1 + 5 \text{ X } 10^0$

a) 1 byte

$00100111_2 = 0 \text{ X } 2^7 + 0 \text{ X } 2^6 + 1 \text{ X } 2^5 + 0 \text{ X } 2^4 + 0 \text{ X } 2^3 + 1 \text{ X } 2^2 + 1 \text{ X } 2^1 + 1 \text{ X } 2^0$

$= 0 + 0 + 32 + 0 + 0 + 4 + 2 + 1 = 39_{10}$

$00100111_2 = 0010_2 \quad 0111_2$

$= 27_{16}$

b) 1 word

$0101011101101110_2 = 0 \text{ X } 2^{15} + 1 \text{ X } 2^{14} + ... + 1 \text{ X } 2^2 + 1 \text{ X } 2^1 + 0 \text{ X } 2^0$

$= 22382_{10}$

$0101\ 0111\ 0110\ 1110_2$ = $576E_{16}$ **(mais fácil de representar!)**

high byte = 0101 0111b = 57_{16}

low byte = 0110 1110b = $6E_{16}$

ORGANIZAÇÃO DE SISTEMAS DE COMPUTAÇÃO

- **Inteiros binários não sinalizados**
  - **Dado um número de n-bits**

$$x = x_{n-1}2^{n-1} + x_{n-2}2^{n-2} + \cdots + x_1 2^1 + x_0 2^0$$

- Intervalo: 0 to $+2^n - 1$
- Usando 32 bits

  0 to +4,294,967,295

- Exemplo

  $0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 1011_2$

  $= 0 + \ldots + 1\times2^3 + 0\times2^2 + 1\times2^1 + 1\times2^0$

  $= 0 + \ldots + 8 + 0 + 2 + 1 = 11_{10}$

Conversão entre bases numéricas

Tipo de conversão	Procedimento
Decimal ➔ Binário	Divisões sucessivas por 2 até se obter zero no quociente. Leitura dos dígitos binários de baixo para cima.
Binário ➔ Decimal	Soma de potências de 2 cujo expoente é a posição do bit e cujo coeficiente é o próprio bit.
Hexadecimal ➔ Binário	Expandir cada dígito hexa em quatro dígitos binários segundo seu valor.
Binário ➔ Hexadecimal	Compactar cada quatro dígitos binários em um único dígito hexa segundo seu valor.
Decimal ➔ Hexadecimal	Divisões sucessivas por 16 até se obter zero no quociente; leitura dos dígitos de baixo para cima.
Hexadecimal ➔ Decimal	Soma de potências de 16 cujo expoente é a posição do dígito e cujo coeficiente é o valor do próprio dígito hexa.

- **Hexadecimal - Base 16**
  - **Representação compacta de strings de bits**
  - **4 bits por dígito hexadecimal**

0	0000	4	0100	8	1000	c	1100
1	0001	5	0101	9	1001	d	1101
2	0010	6	0110	a	1010	e	1110
3	0011	7	0111	b	1011	f	1111

- Exemplo: eca8 6420
  - 1110 1100 1010 1000 0110 0100 0010 0000

- **Exercícios**

Fazer as seguintes conversões

a) **135_{10} para as bases 2, 8 e 16**
b) **10110_{2} para as bases 8, 10 e 16**
c) **374_{8} para as bases 2,10 e 16**
d) **$AB4F_{16}$ para as bases 2,8,10**

- **Exercícios**

Fazer as seguintes conversões

a) 135_{10} para as bases 2, 8 e 16

Base 2 = 1000 0111_2 **Base 8=207_8** **Base 16 = 87_{16}**

	Q	R
135/2	67	1
67/2	33	1
33/2	16	1
16/2	8	0
8/2	4	0
4/2	2	0
2/2	1	0
1/2	0	1

7	6	5	4	3	2	1	0
1	0	0	0	0	1	1	1

- **Exercícios**

Fazer as seguintes conversões

b) 10110_2 para as bases 8, 10 e 16

$10110_2 = 26_8 = 16_{16} = 22_{10}$

7	6	5	4	3	2	1	0
			1	0	1	1	0

$010\ \ 110_2 = 26_8$

$10110_2 = 1 \times 2^4 + 1 \times 2^2 + 1 \times 2^1 = 16 + 4 + 2 = 22_{10}$

$0001\ \ 0110_2 = 16_{16}$

- **Exercícios**

Fazer as seguintes conversões

c) 374_8 para as bases 2,10 e 16

	8	7	6	5	4	3	2	1	0
	0	1	1	1	1	1	1	0	0

$3\ 7\ 4_8 = 011\ 111\ 100_2$

$374_8 = 3 \text{ X } 8^2 + 7\text{X } 8^1 + 4 \text{ X } 8^0 = 192 + 56 + 32 = 252_{10}$

$374_8 = 1111\ 1100_2 = FC_{16}$

- **Exercícios**

Fazer as seguintes conversões

d) $AB4F_{16}$ para as bases 2,8,10

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	1	0	1	0	1	1	0	1	0	0	1	1	1	1

$AB4F_{16}$ = 1010 1011 0100 1111 $_2$

$AB4F_{16}$ =1 010 101 101 001 111 $_2$ = 125517 $_8$

$AB4F_{16}$ = 10 x 16^3 + 11 x 16^2 +4 x 16^1 +15 x 16^0 = 40960 + 2816 + 64 + 15 = 43855 $_{10}$

OBSERVAÇÃO

CONTAGEM DECIMAL

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

CONTAGEM BINÁRIA

		0
		1
	1	0
	1	1
1	0	0
1	0	1
1	1	0
1	1	1

CONTAGEM OCTAL

		0
		1
		2
		3
		4
		5
		6
		7
	1	0
	1	1
	1	2
	1	3
	1	4
	1	5
	1	6
	1	7
	2	0

Representação binária de números sinalizados

- **Representação com sinal e magnitude**
  - O bit mais significativo é o sinal do número ➔ se for 1 o número é negativo se for 0 o número é positivo

Exemplo 1: 01110001_2

valor não sinalizado $= 0 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 +$
$+ 0 \times 2^1 + 1 \times 2^0 =$
$= 64 + 32 + 16 + 1 = 113_{10}$

valor sinalizado bit de sinal = 0 => " + " (positivo)
$= 1 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 +$
$= 0 \times 2^1 + 1 \times 2^0 =$
$= 64 + 32 + 16 + 1 = 113_{10}$ ➔logo= $+113_{10}$

Exemplo 2: 10110001_2

valor não sinalizado $= 1 \times 2^7 + 0 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 +$
$+ 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 =$
$= 128 + 32 + 16 + 1 = 177_{10}$

valor sinalizado bit de sinal = 1 => " - " (negativo)
$= 0 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 +$
$+ 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$
$= 32 + 16 + 1 = 49_{10}$ ➔ logo $= -49_{10}$

Exemplo 3:

$70FF_{16}$ = 0111000011111111_2

valor não sinalizado = $0 \times 2^{15} + 1 \times 2^{14} + \ldots + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$

valor sinalizado ➔ bit de sinal = 0 => " + " (positivo)

$= +(0 \times 2^{15} + 1 \times 2^{14} + \ldots + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0)$

Exemplo 4:

$C777_{16}$ = 1100011101110111_2

valor não sinalizado = $1 \times 2^{15} + 1 \times 2^{14} + \ldots + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$

valor sinalizado ➔ bit de sinal = 1 => " - " (negativo)

$= -(1 \times 2^{14} + \ldots + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0)$

- **Inteiros Sinalizados - Complemento de 2**
  - **Dado um número de n-bits**

$$x = -x_{n-1}2^{n-1} + x_{n-2}2^{n-2} + \cdots + x_1 2^1 + x_0 2^0$$

- Intervalo: -2^{n-1} to $+2^{n-1} - 1$
- Usando 32 bits

  –2,147,483,648 to +2,147,483,647
- Exemplo

  $1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1100_2$

  $= -1\times 2^{31} + 1\times 2^{30} + \ldots + 1\times 2^{2} + 0\times 2^{1} + 0\times 2^{0}$

  $= -2{,}147{,}483{,}648 + 2{,}147{,}483{,}644 = -4_{10}$

- **Representações possíveis de números sinalizados**

Sinal e Magnitude	Complemento de 1	*Complemento de 2*
000 = +0	*000 = +0*	*000 = +0*
001 = +1	*001 = +1*	*001 = +1*
010 = +2	*010 = +2*	*010 = +2*
011 = +3	*011 = +3*	*011 = +3*
100 = -0	100 = -3	100 = -4
101 = -1	101 = -2	101 = -3
110 = -2	110 = -1	110 = -2
111 = -3	111 = -0	111 = -1

- **Representação em Complemento de 2 ➔ utilizada pois temos apenas uma representação para o zero e podemos fazer a soma e subtração com apenas um circuito.**

- **(-3 em C2) → 011 (+3) → 100 → 100 + 1 → 101 (-3) em c2**
- **(-2 em c2) → 010 (+2) → 101 → 101 + 1 → 110 (-2) em c2**

- **Números sinalizados de 32 bits, em Complemento de 2:**

0000 0000 0000 0000 0000 0000 0000 0000$_2$ = 0$_{10}$
0000 0000 0000 0000 0000 0000 0000 0001$_2$ = + 1$_{10}$
0000 0000 0000 0000 0000 0000 0000 0010$_2$ = + 2$_{10}$
...
0111 1111 1111 1111 1111 1111 1111 1110$_2$ = + 2,147,483,646$_{10}$
0111 1111 1111 1111 1111 1111 1111 1111$_2$ = + 2,147,483,647$_{10}$ *maxint*
1000 0000 0000 0000 0000 0000 0000 0000$_2$ = – 2,147,483,648$_{10}$ *minint*
1000 0000 0000 0000 0000 0000 0000 0001$_2$ = – 2,147,483,647$_{10}$
1000 0000 0000 0000 0000 0000 0000 0010$_2$ = – 2,147,483,646$_{10}$
...
1111 1111 1111 1111 1111 1111 1111 1101$_2$ = – 3$_{10}$
1111 1111 1111 1111 1111 1111 1111 1110$_2$ = – 2$_{10}$
1111 1111 1111 1111 1111 1111 1111 1111$_2$ = – 1$_{10}$

Representação em Complemento de 2 de um número:

- Partindo-se da representação do negativo do valor a ser achado, nega-se este número (negar ➔ inverter e somar 1)

Exemplo 1:

-5 em Complemento de 2 (com 1 bit de sinal de 4 para a magnitude)

Partindo-se da representação do $+5_{10} = 00101_2$ ➔(invertendo os bits) = 11010 ➔ (somando 1) = 11011_2 = - 5 em Complemento de 2

Exemplo 2:

+5 em Complemento de 2 (com 1 bit de sinal de 4 para a magnitude)

Partindo-se da representação do $-5_{10} = 11011_2$ ➔(invertendo os bits) = 00100_2 ➔ (somando 1) = 00101_2 = +5 em Complemento de 2

- **Achar a representação em C2 dos números → 4 bits**

  a) -7 b)+4 c) -8 d) +8

a) +7 = 0111 → 1000 → 1001

b) +4 = 0100

c) -8 = 1000

d) -8 → 1000 →0111 → 1000 → OVERFLOW

- **Conversão de números com n bits em números com mais que n bits:**
  - **copiar o bit mais significativo (bit de sinal) nos outros bits (extensão do sinal):**

Exemplo: (números em C2)

0010 → 0000 0010

1010 → 1111 1010

Obs → 1010 → 0101 → 0110 (+6) → 1010 = -6

Operações de soma e adição binárias

- **Como aprenderam no primeiro grau: (vai-um/vem-um)**

```
   0111 (7)        0111 (7)        0110 (6)
 + 0110 (6)      - 0110 (6)      - 0101 (5)
 ----------------------------------------
   1101 (13)       0001 (1)        0001 (1)
```

- **Adição e subtração em complemento de 2 é feito como se fosse uma soma:**
  - **subtração usando adição de números negativos**

```
      0111  (=+7)
   + 1010   (=-6)
   ---------
1|    0001  (=1)
```

Overflow

- **Overflow (resultado maior (menor) que a palavra do computador pode representar):**

Exemplo:

- **Quando na operação abaixo ocorre e quando não ocorre overflow ???**

```
  0111 (7) ou (+7)
+ 0001 (1) ou (+1)
  ----
  1000
```

Detecção de Overflow

- Não existe overflow quando adicionamos um número positivo e um negativo
- Não existe overflow quando os sinais dos números são os mesmos na subtração
- Ocorre overflow quando os valores afetam o sinal:
  - Somando dois números positivos dá um número negativo
  - Somando dois números negativos dá um número positivo
  - Subtrai um número negativo de um positivo e dá negativo
  - Subtrai um número positivo de um negativo e dá positivo

Exercício

- Considere as operaçôes A + B e A – B
  - Pode ocorrer overflow se B = 0 ?
  - Pode ocorrer overflow se A = 0 ?

1 - EXERCÍCIO – S&M com 4 bits

- a) (-3) + (+4) b) (-7) + (+5) c) (+4) +(+4)
- d) (+7) - (+4)

2- EXERCÍCIO – C2 com 4 bits

- a) (-3) + (+4) b) (-7) + (+5) c) (+4) +(+4)
- d) (+7) - (+4) e) (-3) – (-4)

ORGANIZAÇÃO DE SISTEMAS DE COMPUTAÇÃO

- **EXERCÍCIOS – S&M com 4 bits**

- **a) (-3) + (+4) = + 4 – (+3) b) (-7) + (+5) = -((+7) – (+5)) c) (+4) +(+4)**
- **d) (+7) - (+4) e) (-3) – (-4)**

- **a) +4 0100 -> 00 (10) 0 → 0 0 1 (10)**

  -

  +3 0011 0 0 1 1

  = +1 0001 0 0 0 1

  b) +7 = 0111

  +5 = 0101

  = + 2= 0010 → -2 = 1010

- **EXERCÍCIOS – S&M com 4 bits**
- **c) (+4) +(+4)   d) (+7) - (+4)   e) (-3) – (-4) = (-3) + (+4) = (+4) – (+3)**

- **c) +4 = 0100**
  +
  +4 = 0100
  -8 + 1000 (?????OVERFLOW)

  e) +4 = 0100
  +3 = 0011
  +1 = 0001

- **d) +7 = 0111**
  -
  +4 = 0100
  +3 = 0011

ORGANIZAÇÃO DE SISTEMAS DE COMPUTAÇÃO

- **EXERCÍCIOS – C2 com 4 bits**

- **a) (-3) + (+4)   b) (-7) + (+5)   c) (+4) +(+4)**
- **d) (+7) - (+4) = (+7) + (-4)   e) (-3) – (-4) = (-3) + (+4)**

- **d) +7 = 0111   0111**
- **+ 4 = 0100 → (-4) → 1011+1 = 1100**
- **1 0011 (+3)**

- **3) +3 → 0011 → -3 = 1100+1 = 1101**
- **+4   = 0100**
- **1 0001**

Multiplicação Binária

- **Exemplo:**

1010 X 0101

```
     1 0 1 0
    X 1 0 1
    _______
     1 0 1 0
   0 0 0 0
 1 0 1 0
0 0 0 0
____________
0 1 1 0 0 1 0
```

Multiplicação Binária

- **Exemplo:**

0010 X 0011

```
      0 0 1 0
    X 0 0 1 1
    _________
      0 0 1 0
    0 0 1 0
  0 0 0 0
0 0 0 0
_____________
0 0 0 0 1 1 0
```

Divisão Binária

- **Exemplo:**

1 1 0 0 1 0 / 101

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

Representação de Números em Ponto Flutuante

- **Padrão IEEE 754 – normalizado, expoente em excesso 127**

$$N = (-1)^S \times 1{,}M \times 2^E$$

  - **precisão simples**

31	30 … 23	22 … 0
S	EXPOENTE EM EXCESSO	MANTISSA

  - **precisão dupla**

63	62 … 55	54 … 32
S	EXPOENTE EM EXCESSO	MANTISSA

31 … 0
MANTISSA

Exemplo 1: Qual é a representação no padrão IEEE 754 de:

$-0{,}75_{10}$

Normalizando

31	30 23	22 0

Exemplo 2: Qual o decimal correspondente ?

31	30 23	22 0
1	1 0 0 0 0 0 0 1	1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Exemplo

$$-0{,}75_{10} = -0{,}11_2 \times 2^{0} = -1{,}10000_2 \times 2^{-1}$$

Normalizando $-1{,}10000_2 \times 2^{-1}$

31	30 … 23	22 … 0
1	0 1 1 1 1 1 1 0	1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

$$EXP = E + 127 = -1 + 127 = 126 \rightarrow 01111110$$

$$-0{,}75_{10} = -(0{,}5 + 0{,}25) = -(2^{-1} + 2^{-2}) = -0{,}1100$$

$2^8\ 2^7\ 2^6\ 2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$	$,\ 2^{-1}\ 2^{-2}\ 2^{-3}\ 2^{-4}\ldots$

Exemplo: Qual o decimal correspondente ?

31	30 — 23	22 — 0
1	1 0 0 0 0 0 0 1	1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

$$N = -\,(1+0{,}5) \times 2^{(129-127)} = -1{,}5 \times 4 = -6{,}0_{10}$$

MANTISSA → $0{,}10000......0_2 = 2^{-1} = 0{,}5$

$2^8\ 2^7\ 2^6\ 2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$	$,\ 2^{-1}\ 2^{-2}\ 2^{-3}\ 2^{-4}....$

Representação de Caracteres Alfanuméricos

- **Tabela ASCII (American Standard Code Interchange Information)**

Exemplo:

64	@		96	´
65	**A**		97	**a**
66	**B**		98	**b**
67	**C**		99	**c**
68	**D**		100	**d**
69	**E**		101	**e**
70	**F**		102	**f**
71	**G**		103	**g**
72	**H**		104	**h**
73	**I**		105	**i**

48	**0**
49	**1**
50	**2**
51	**3**
52	**4**
53	**5**
54	**6**
55	**7**
56	**8**
57	**9**

ORGANIZAÇÃO DE SISTEMAS DE COMPUTAÇÃO

Tabela ASCII

Dec	Hex	Oct	Chr	Dec	Hex	Oct	HTML	Chr	Dec	Hex	Oct	HTML	Chr	Dec	Hex	Oct	HTML	Chr
0	0	000	NULL	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	Start of Header	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	Start of Text	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	End of Text	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	End of Transmission	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	Enquiry	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	Acknowledgment	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	Bell	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	Backspace	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	Horizontal Tab	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	Line feed	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	Vertical Tab	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	Form feed	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	Carriage return	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	Shift Out	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	Shift In	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	Data Link Escape	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	Device Control 1	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	Device Control 2	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	Device Control 3	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	Device Control 4	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	Negative Ack.	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	Synchronous idle	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	End of Trans. Block	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	Cancel	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	End of Medium	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	Substitute	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	Escape	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	File Separator	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	Group Separator	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	Record Separator	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	Unit Separator	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	Del

asciicharstable.com