Fundamentals of Computer Systems

Thinking Digitally

Martha A. Kim

Columbia University

Fall 2015

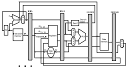
Computer Systems Work Because of Abstraction

Application Software

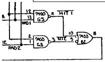
Operating Systems

```
  ;; voice 1 wave select
ld      a,(#CH1_W_NUM)
and     a
ld      a,(#CH1_W_SEL)
jr      nz,#00b4
ld      a,(#CH1_E_TABLE0)
```

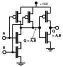
Architecture



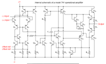
Micro-Architecture



Logic

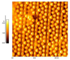


Digital Circuits



Analog Circuits

Devices



Physics

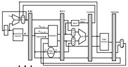
Computer Systems Work Because of Abstraction

Application Software — COMS 3157, 4156, et al.

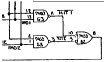
Operating Systems — COMS W4118

```
;; voice 1 wave select
ld      a,(#CH1_W_NUM)
and     a
ld      a,(#CH1_W_SEL)
jr      nz,#00b4
ld      a,(#CH1_E_TABLE0)
```

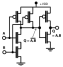
Architecture — Second Half of 3827



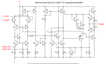
Micro-Architecture — Second Half of 3827



Logic — First Half of 3827

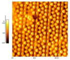


Digital Circuits — First Half of 3827



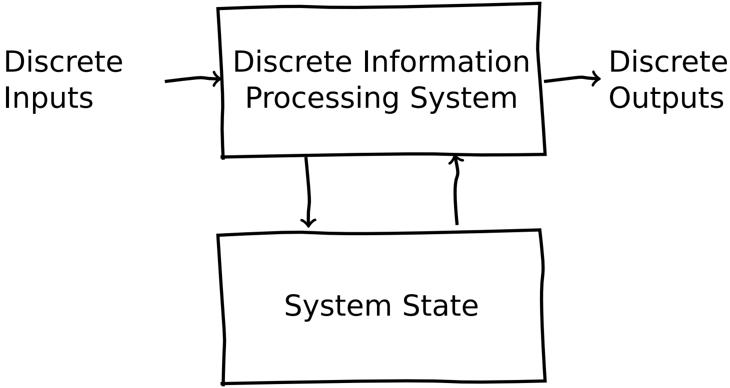
Analog Circuits — ELEN 3331

Devices — ELEN 3106



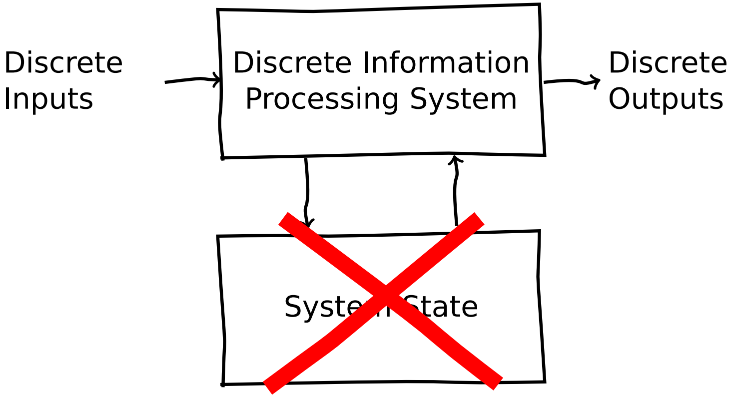
Physics — ELEN 3106 et al.

Simple information processing system



First half of the course

Simple information processing system



First quarter of the course

Administrative Items

`http://www.cs.columbia.edu/~martha/courses/3827/au15/`

`https://piazza.com/class/idq5hhzggizmq`

Prof. Martha A. Kim

martha@cs.columbia.edu

469 Computer Science Building

Lectures 10:10–11:25 AM Tue, Thur
501 Schermerhorn Hall
Sep 8–Dec 10
Holidays: Nov 3 (Election Day), Nov 26 (Thanksgiving)

Office Hours

The six (and counting) TAs and I will all offer office hours.

Always consult the course calendar (linked from course webpage) for the latest schedule.

`https://www.google.com/calendar/embed?src=8g48vdedcbb85k7jn4or`

Assignments and Grading

Weight	What	When
40%	Six homeworks	See Webpage
30%	Midterm exam #1	October TBA
30%	Midterm exam #2	December TBA

Homework is due at the beginning of lecture.

We will drop the lowest of your six homework scores; you can __________ one assignment with no penalty.

There will be no extensions.

Rules and Regulations

You may consult and collaborate with classmates on homework, but *you must turn in your own work*.

List your collaborators on your homework.

Use your judgement about outside resources. E.g., Reading wikipedia is fine, but asking stackoverflow.com to help debug your assembly code is not. In unclear situations, ask.

Do not cheat.

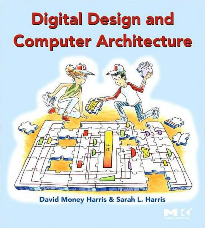
Tests will be closed-book with a one-page “note sheet” of your own devising.

The Text(s): Alternative #1

No required text. There are two recommended alternatives.

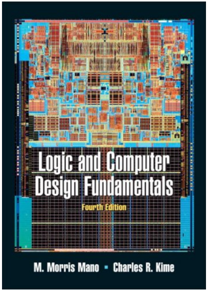
- David Harris and Sarah Harris. *Digital Design and Computer Architecture.*

  Almost precisely right for the scope of this class: digital logic and computer architecture.

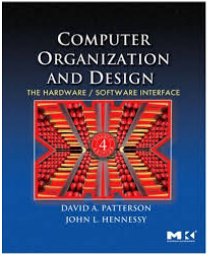


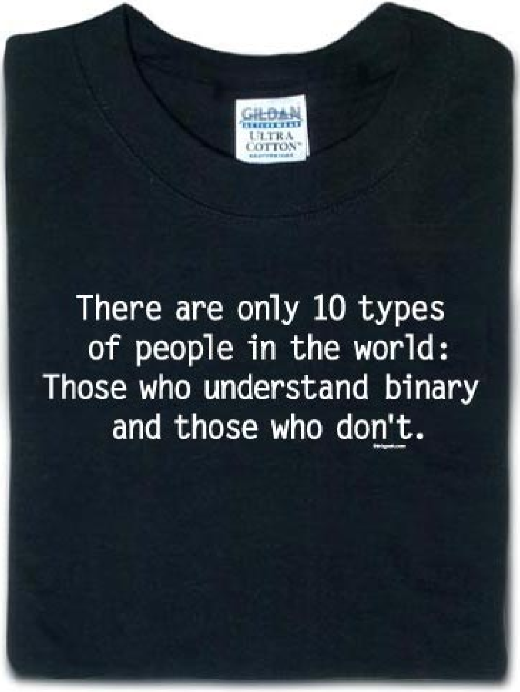
The Text(s): Alternative #2

- M. Morris Mano and Charles Kime. *Logic and Computer Design Fundamentals, 4th ed.*



- Computer Organization and Design, The Hardware/Software Interface, 4th ed. *David A. Patterson and John L. Hennessy*





thinkgeek.com

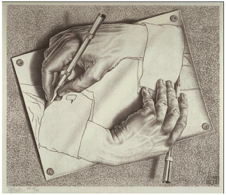
The Decimal Positional Numbering System

Ten figures: 0 1 2 3 4 5 6 7 8 9

$7 \times 10^2 + 3 \times 10^1 + 0 \times 10^0 = 730_{10}$

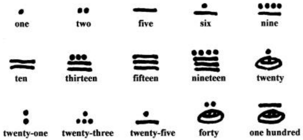
$9 \times 10^2 + 9 \times 10^1 + 0 \times 10^0 = 990_{10}$

Why base ten?

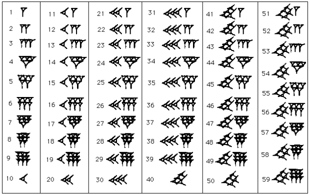


Which Numbering System Should We Use? Some Older Choices:

Roman: I II III IV V VI VII VIII IX X



Mayan: base 20, Shell = 0

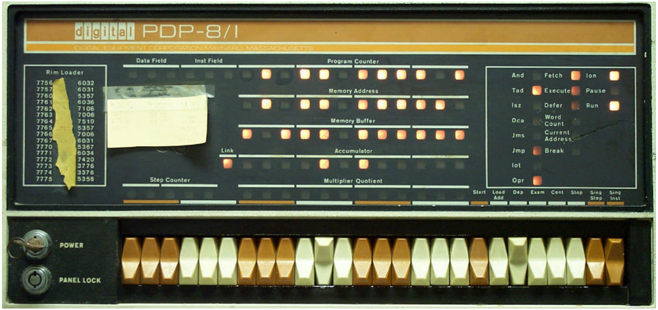


Babylonian: base 60

Hexadecimal, Decimal, Octal, and Binary

Hex	Dec	Oct	Bin
0	0	0	0
1	1	1	1
2	2	2	10
3	3	3	11
4	4	4	100
5	5	5	101
6	6	6	110
7	7	7	111
8	8	10	1000
9	9	11	1001
A	10	12	1010
B	11	13	1011
C	12	14	1100
D	13	15	1101
E	14	16	1110
F	15	17	1111

Binary and Octal



DEC PDP-8/I, c. 1968

Oct	Bin
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

$$\begin{aligned} \text{PC} &= 0 \times 2^{11} + 1 \times 2^{10} + 0 \times 2^{9} + 1 \times 2^{8} + 1 \times 2^{7} + 0 \times 2^{6} + \\ &\quad 1 \times 2^{5} + 1 \times 2^{4} + 1 \times 2^{3} + 1 \times 2^{2} + 0 \times 2^{1} + 1 \times 2^{0} \\ &= 2 \times 8^{3} + 6 \times 8^{2} + 7 \times 8^{1} + 5 \times 8^{0} \\ &= 1469_{10} \end{aligned}$$

Hexadecimal Numbers

Base 16: 0 1 2 3 4 5 6 7 8 9 A B C D E F

Instead of groups of 3 bits (octal), Hex uses groups of 4.

$$
\begin{aligned}
\text{CAFEF00D}_{16} &= 12 \times 16^7 + 10 \times 16^6 + 15 \times 16^5 + 14 \times 16^4 + \\
&\quad 15 \times 16^3 + 0 \times 16^2 + 0 \times 16^1 + 13 \times 16^0 \\
&= 3,405,705,229_{10}
\end{aligned}
$$

C	A	F	E	F	0	0	D				Hex
11001010111111101111000000001101											Binary
3	1	2	7	7	5	7	0	0	1	5	Octal

Computers Rarely Manipulate True Numbers

Infinite memory still very expensive

Finite-precision numbers typical

32-bit processor: naturally manipulates 32-bit numbers

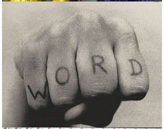
64-bit processor: naturally manipulates 64-bit numbers

How many different numbers can you represent with 5 binary / octal / decimal / hexadecimal digits?

Jargon

Bit Binary digit: 0 or 1

Byte Eight bits



Word Natural number of bits for the processor, e.g., 16, 32, 64

LSB Least Significant Bit (“rightmost”)

MSB Most Significant Bit (“leftmost”)

Decimal Addition Algorithm

$$\begin{array}{r} 434 \\ +628 \\ \hline \end{array}$$

$$4 + 8 = 12$$

+	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9	10
2	2	3	4	5	6	7	8	9	10	11
3	3	4	5	6	7	8	9	10	11	12
4	4	5	6	7	8	9	10	11	12	13
5	5	6	7	8	9	10	11	12	13	14
6	6	7	8	9	10	11	12	13	14	15
7	7	8	9	10	11	12	13	14	15	16
8	8	9	10	11	12	13	14	15	16	17
9	9	10	11	12	13	14	15	16	17	18
10	10	11	12	13	14	15	16	17	18	19

Decimal Addition Algorithm

$$\begin{array}{r} 1 \\ 434 \\ +628 \\ \hline 2 \end{array}$$

$$4 + 8 = 12$$
$$1 + 3 + 2 = 6$$

+	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9	10
2	2	3	4	5	6	7	8	9	10	11
3	3	4	5	6	7	8	9	10	11	12
4	4	5	6	7	8	9	10	11	12	13
5	5	6	7	8	9	10	11	12	13	14
6	6	7	8	9	10	11	12	13	14	15
7	7	8	9	10	11	12	13	14	15	16
8	8	9	10	11	12	13	14	15	16	17
9	9	10	11	12	13	14	15	16	17	18
10	10	11	12	13	14	15	16	17	18	19

Decimal Addition Algorithm

$$\begin{array}{r} 1 \\ 434 \\ +628 \\ \hline 62 \end{array}$$

$$\begin{aligned} 4 + 8 &= 12 \\ 1 + 3 + 2 &= 6 \\ 4 + 6 &= 10 \end{aligned}$$

+	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9	10
2	2	3	4	5	6	7	8	9	10	11
3	3	4	5	6	7	8	9	10	11	12
4	4	5	6	7	8	9	10	11	12	13
5	5	6	7	8	9	10	11	12	13	14
6	6	7	8	9	10	11	12	13	14	15
7	7	8	9	10	11	12	13	14	15	16
8	8	9	10	11	12	13	14	15	16	17
9	9	10	11	12	13	14	15	16	17	18
10	10	11	12	13	14	15	16	17	18	19

Decimal Addition Algorithm

$$\begin{array}{r} {\color{red}1\ \ 1\ \ } \\ 434 \\ +628 \\ \hline {\color{blue}062} \end{array}$$

$$\begin{aligned} 4+8 &= {\color{red}1}{\color{blue}2} \\ 1+3+2 &= {\color{blue}6} \\ 4+6 &= {\color{red}10} \end{aligned}$$

+	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9	10
2	2	3	4	5	6	7	8	9	10	11
3	3	4	5	6	7	8	9	10	11	12
4	4	5	6	7	8	9	10	11	12	13
5	5	6	7	8	9	10	11	12	13	14
6	6	7	8	9	10	11	12	13	14	15
7	7	8	9	10	11	12	13	14	15	16
8	8	9	10	11	12	13	14	15	16	17
9	9	10	11	12	13	14	15	16	17	18
10	10	11	12	13	14	15	16	17	18	19

Decimal Addition Algorithm

$$\begin{array}{r} \color{red}{1\ 1\ \ } \\ 434 \\ +628 \\ \hline \color{blue}{1062} \end{array}$$

$$\begin{aligned} 4+8 &= \color{red}{1}\color{blue}{2} \\ 1+3+2 &= \color{blue}{6} \\ 4+6 &= \color{red}{10} \end{aligned}$$

+	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9	10
2	2	3	4	5	6	7	8	9	10	11
3	3	4	5	6	7	8	9	10	11	12
4	4	5	6	7	8	9	10	11	12	13
5	5	6	7	8	9	10	11	12	13	14
6	6	7	8	9	10	11	12	13	14	15
7	7	8	9	10	11	12	13	14	15	16
8	8	9	10	11	12	13	14	15	16	17
9	9	10	11	12	13	14	15	16	17	18
10	10	11	12	13	14	15	16	17	18	19

Binary Addition Algorithm

$$
\begin{array}{r} 10011 \\ +11001 \\ \hline \end{array}
$$

$$1+1 \quad = \quad 10$$

+	0	1
0	00	01
1	01	10
10	10	11

Binary Addition Algorithm

$$\begin{array}{r} 1 \\ 10011 \\ +11001 \\ \hline 0 \end{array}$$

$$1 + 1 = 10$$

$$1 + 1 + 0 = 10$$

+	0	1
0	00	01
1	01	10
10	10	11

Binary Addition Algorithm

$$
\begin{array}{r}
11 \\
10011 \\
+11001 \\
\hline
00
\end{array}
$$

$$
\begin{aligned}
1 + 1 &= 10 \\
1 + 1 + 0 &= 10 \\
1 + 0 + 0 &= 01
\end{aligned}
$$

+	0	1
0	00	01
1	01	10
10	10	11

Binary Addition Algorithm

$$
\begin{array}{r}
011 \\
10011 \\
+11001 \\
\hline
100
\end{array}
$$

$$
\begin{aligned}
1 + 1 &= 10 \\
1 + 1 + 0 &= 10 \\
1 + 0 + 0 &= 01 \\
0 + 0 + 1 &= 01
\end{aligned}
$$

+	0	1
0	00	01
1	01	10
10	10	11

Binary Addition Algorithm

$$\begin{array}{r} 0011 \\ 10011 \\ +11001 \\ \hline 1100 \end{array}$$

$$\begin{aligned} 1+1 &= 10 \\ 1+1+0 &= 10 \\ 1+0+0 &= 01 \\ 0+0+1 &= 01 \\ 0+1+1 &= 10 \end{aligned}$$

+	0	1
0	00	01
1	01	10
10	10	11

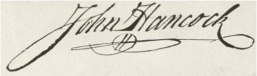
Binary Addition Algorithm

$$\begin{array}{r} 10011 \\ 10011 \\ +11001 \\ \hline 101100 \end{array}$$

$$\begin{aligned} 1+1 &= 10 \\ 1+1+0 &= 10 \\ 1+0+0 &= 01 \\ 0+0+1 &= 01 \\ 0+1+1 &= 10 \end{aligned}$$

+	0	1
0	00	01
1	01	10
10	10	11

Signed Numbers: Dealing with Negativity



How should both positive and negative numbers be represented?

Signed Magnitude Numbers

You are most familiar with this:
negative numbers have a leading $-$

In binary, a leading 1 means negative:

$0000_2 = 0$

$0010_2 = 2$

$1010_2 = -2$

$1111_2 = -7$

$1000_2 = -0$?

Can be made to work, but addition is annoying:

If the signs match, add the magnitudes and use the same sign.

If the signs differ, subtract the smaller number from the larger; return the sign of the larger.

One's Complement Numbers

Like Signed Magnitude, a leading 1 indicates a negative One's Complement number.

To negate a number, complement (flip) each bit.

$0000_2 = 0$

$0010_2 = 2$

$1101_2 = -2$

$1000_2 = -7$

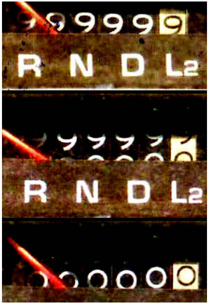
$1111_2 = -0?$

Addition is nicer: just add the one's complement numbers as if they were normal binary.

Really annoying having a -0: two numbers are equal if their bits are the same or if one is 0 and the other is -0.

0 CALORIE COLA
pepsi MAX
zero calories
maximum taste
NOT ALL ZEROS ARE CREATED EQUAL
ZERO CALORIES. MAXIMUM PEPSI® TASTE.
pepsi MAX zero calories

Two's Complement Numbers



Really neat trick: make the most significant bit represent a *negative* number instead of positive:

$1101_2 = -8 + 4 + 1 = -3$

$1111_2 = -8 + 4 + 2 + 1 = -1$

$0111_2 = 4 + 2 + 1 = 7$

$1000_2 = -8$

Easy addition: just add in binary and discard any carry.

Negation: complement each bit (as in one's complement) then add 1.

Very good property: no -0

Two's complement numbers are equal if all their bits are the same.

Number Representations Compared

Bits	Binary	Signed Mag.	One's Comp.	Two's Comp.
0000	0	0	0	0
0001	1	1	1	1
⋮				
0111	7	7	7	7
1000	8	−0	−7	−8
1001	9	−1	−6	−7
⋮				
1110	14	−6	−1	−2
1111	15	−7	−0	−1

Smallest number
Largest number

Fixed-point Numbers

How to represent fractional numbers? In decimal, we continue with negative powers of 10:

$$
\begin{aligned}
31.4159 &= 3 \times 10^1 + 1 \times 10^0 + \\
&\quad 4 \times 10^{-1} + 1 \times 10^{-2} + 5 \times 10^{-3} + 9 \times 10^{-4}
\end{aligned}
$$

The same trick works in binary:

$$
\begin{aligned}
1011.0110_2 &= 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + \\
&\quad 0 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3} + 0 \times 2^{-4} \\
&= 8 + 2 + 1 + 0.25 + 0.125 \\
&= 11.375
\end{aligned}
$$

Need a bigger range? Try Floating Point Representation.

Floating point can represent very large numbers in a compact way.

A lot like scientific notation, -7.776×10^3, where you have the *mantissa* (-7.776) and *exponent* (3).

But for this course, think in binary: $-1.10x2^{0111}$

The bits of a 32-bit word are separated into fields. The IEEE 754 standard specifies

- which bits represent which fields (bit 31 is sign, bits 30-23 are 8-bit exponent, bits 22-00 are 23-bit fraction)
- how to interpret each field

Characters and Strings? ASCII.

The ASCII code
American Standard Code for Information Interchange

www.theasciicode.com.ar

ASCII control characters

DEC	HEX	Simbolo ASCII	
00	00h	NULL	(carácter nulo)
01	01h	SOH	(inicio encabezado)
02	02h	STX	(inicio texto)
03	03h	ETX	(fin de texto)
04	04h	EOT	(fin transmisión)
05	05h	ENQ	(enquiry)
06	06h	ACK	(acknowledgement)
07	07h	BEL	(timbre)
08	08h	BS	(retroceso)
09	09h	HT	(tab horizontal)
10	0Ah	LF	(salto de linea)
11	0Bh	VT	(tab vertical)
12	0Ch	FF	(form feed)
13	0Dh	CR	(retorno de carro)
14	0Eh	SO	(shift Out)
15	0Fh	SI	(shift In)
16	10h	DLE	(data link escape)
17	11h	DC1	(device control 1)
18	12h	DC2	(device control 2)
19	13h	DC3	(device control 3)
20	14h	DC4	(device control 4)
21	15h	NAK	(negative acknowle.)
22	16h	SYN	(synchronous idle)
23	17h	ETB	(end of trans. block)
24	18h	CAN	(cancel)
25	19h	EM	(end of medium)
26	1Ah	SUB	(substitute)
27	1Bh	ESC	(escape)
28	1Ch	FS	(file separator)
29	1Dh	GS	(group separator)
30	1Eh	RS	(record separator)
31	1Fh	US	(unit separator)
127	20h	DEL	(delete)

ASCII printable characters

DEC	HEX	Simbolo	DEC	HEX	Simbolo	DEC	HEX	Simbolo
32	20h	espacio	64	40h	@	96	60h	`
33	21h	!	65	41h	A	97	61h	a
34	22h	"	66	42h	B	98	62h	b
35	23h	#	67	43h	C	99	63h	c
36	24h	$	68	44h	D	100	64h	d
37	25h	%	69	45h	E	101	65h	e
38	26h	&	70	46h	F	102	66h	f
39	27h	'	71	47h	G	103	67h	g
40	28h	(	72	48h	H	104	68h	h
41	29h	)	73	49h	I	105	69h	i
42	2Ah	*	74	4Ah	J	106	6Ah	j
43	2Bh	+	75	4Bh	K	107	6Bh	k
44	2Ch	,	76	4Ch	L	108	6Ch	l
45	2Dh	-	77	4Dh	M	109	6Dh	m
46	2Eh	.	78	4Eh	N	110	6Eh	n
47	2Fh	/	79	4Fh	O	111	6Fh	o
48	30h	0	80	50h	P	112	70h	p
49	31h	1	81	51h	Q	113	71h	q
50	32h	2	82	52h	R	114	72h	r
51	33h	3	83	53h	S	115	73h	s
52	34h	4	84	54h	T	116	74h	t
53	35h	5	85	55h	U	117	75h	u
54	36h	6	86	56h	V	118	76h	v
55	37h	7	87	57h	W	119	77h	w
56	38h	8	88	58h	X	120	78h	x
57	39h	9	89	59h	Y	121	79h	y
58	3Ah	:	90	5Ah	Z	122	7Ah	z
59	3Bh	;	91	5Bh	[	123	7Bh	{
60	3Ch	<	92	5Ch	\	124	7Ch	|
61	3Dh	=	93	5Dh	]	125	7Dh	}
62	3Eh	>	94	5Eh	^	126	7Eh	~
63	3Fh	?	95	5Fh	_	theASCIIcode.com.ar		

Extended ASCII characters

DEC	HEX	Simbolo	DEC	HEX	Simbolo	DEC	HEX	Simbolo	DEC	HEX	Simbolo
128	80h	Ç	160	A0h	á	192	C0h	└	224	E0h	Ó
129	81h	ü	161	A1h	í	193	C1h	┴	225	E1h	ß
130	82h	é	162	A2h	ó	194	C2h	┬	226	E2h	Ô
131	83h	â	163	A3h	ú	195	C3h	├	227	E3h	Ò
132	84h	ä	164	A4h	ñ	196	C4h	─	228	E4h	õ
133	85h	à	165	A5h	Ñ	197	C5h	┼	229	E5h	Õ
134	86h	å	166	A6h	ª	198	C6h	ã	230	E6h	µ
135	87h	ç	167	A7h	º	199	C7h	Ã	231	E7h	þ
136	88h	ê	168	A8h	¿	200	C8h	╚	232	E8h	Þ
137	89h	ë	169	A9h	®	201	C9h	╔	233	E9h	Ú
138	8Ah	è	170	AAh	¬	202	CAh	╩	234	EAh	Û
139	8Bh	ï	171	ABh	½	203	CBh	╦	235	EBh	Ù
140	8Ch	î	172	ACh	¼	204	CCh	╠	236	ECh	ý
141	8Dh	ì	173	ADh	¡	205	CDh	═	237	EDh	Ý
142	8Eh	Ä	174	AEh	«	206	CEh	╬	238	EEh	¯
143	8Fh	Å	175	AFh	»	207	CFh	¤	239	EFh	´
144	90h	É	176	B0h	░	208	D0h	ð	240	F0h	
145	91h	æ	177	B1h	▒	209	D1h	Ð	241	F1h	±
146	92h	Æ	178	B2h	▓	210	D2h	Ê	242	F2h	‗
147	93h	ô	179	B3h	│	211	D3h	Ë	243	F3h	¾
148	94h	ö	180	B4h	┤	212	D4h	È	244	F4h	¶
149	95h	ò	181	B5h	Á	213	D5h	ı	245	F5h	§
150	96h	û	182	B6h	Â	214	D6h	Í	246	F6h	÷
151	97h	ù	183	B7h	À	215	D7h	Î	247	F7h	¸
152	98h	ÿ	184	B8h	©	216	D8h	Ï	248	F8h	°
153	99h	Ö	185	B9h	╣	217	D9h	┘	249	F9h	¨
154	9Ah	Ü	186	BAh	║	218	DAh	┌	250	FAh	·
155	9Bh	ø	187	BBh	╗	219	DBh	█	251	FBh	¹
156	9Ch	£	188	BCh	╝	220	DCh	▄	252	FCh	³
157	9Dh	Ø	189	BDh	¢	221	DDh	¦	253	FDh	²
158	9Eh	×	190	BEh	¥	222	DEh	Ì	254	FEh	■
159	9Fh	ƒ	191	BFh	┐	223	DFh	▀	255	FFh	