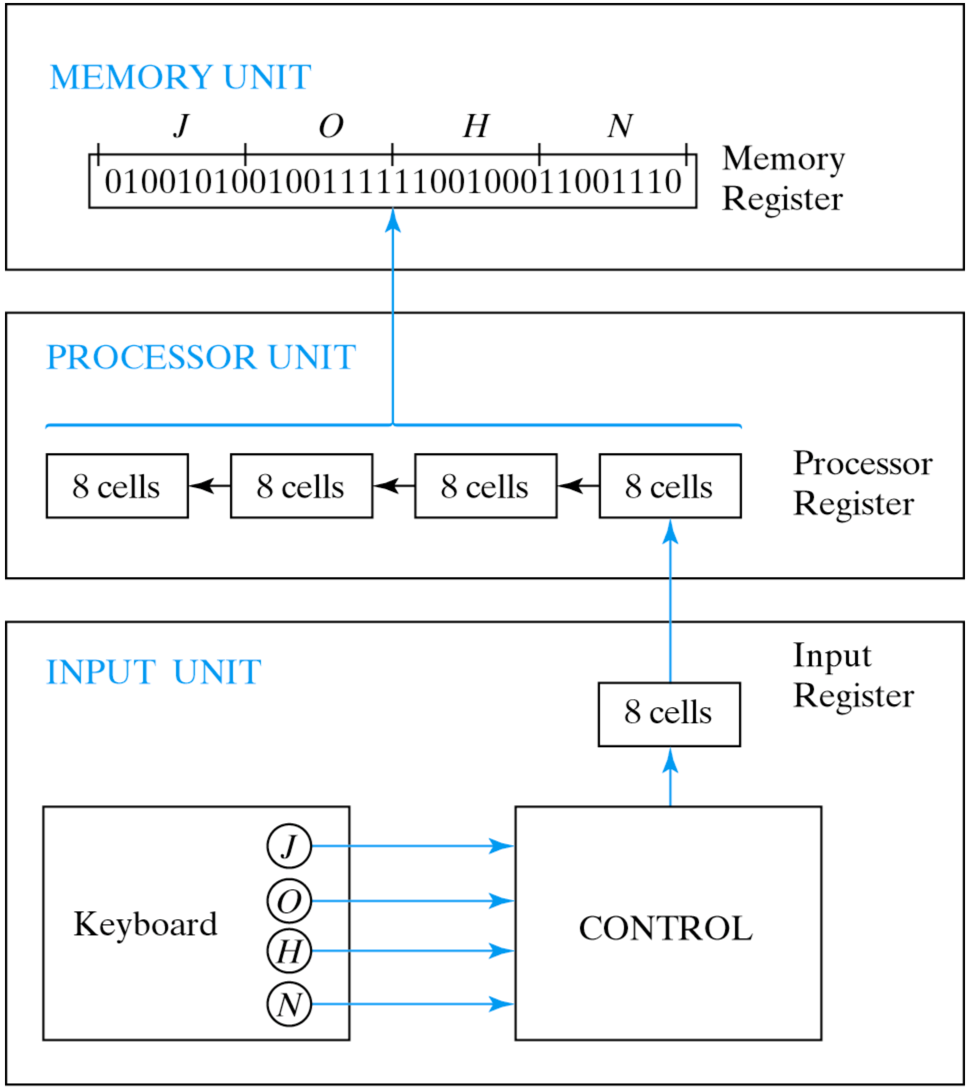
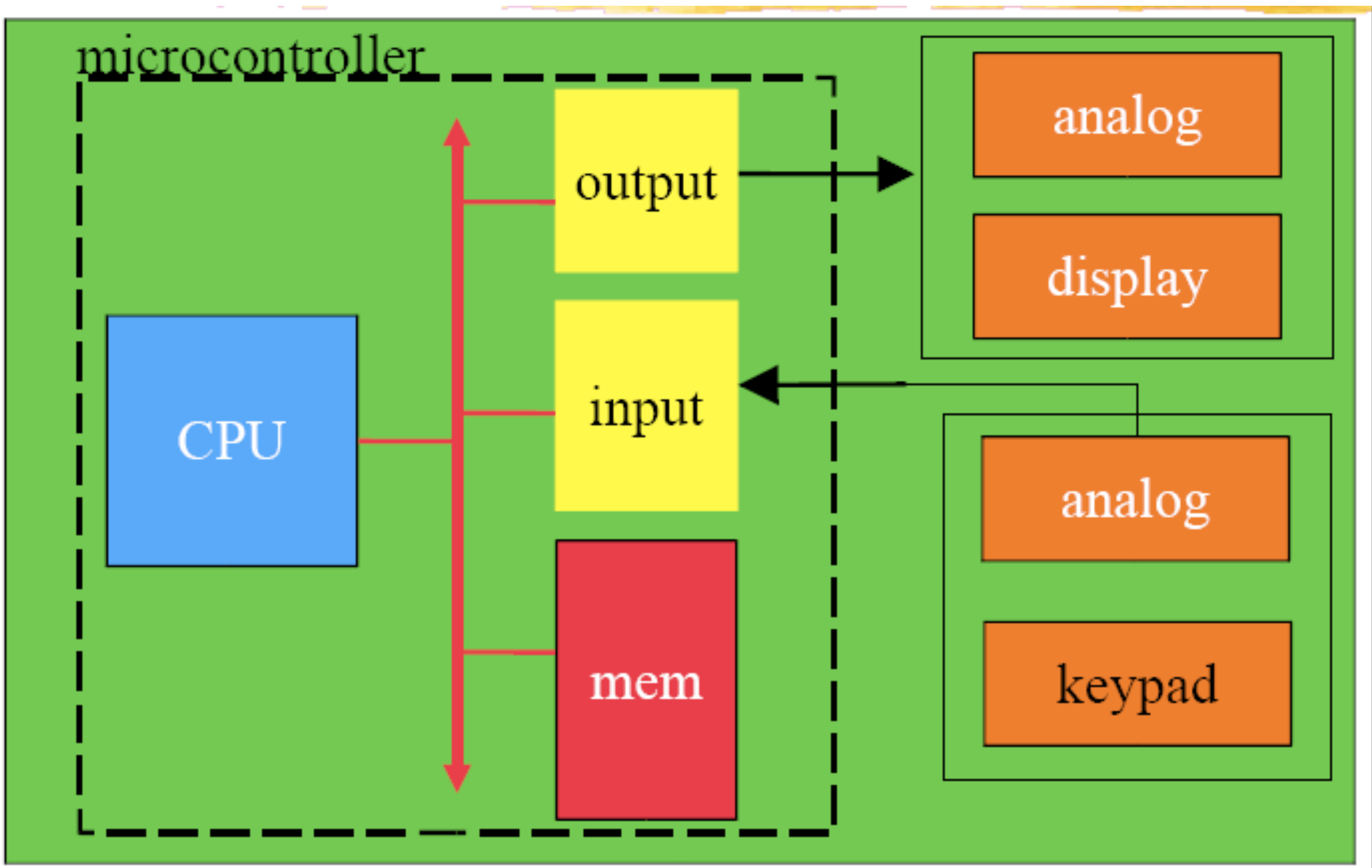


Fig. 1-1 Transfer of information with registers

M. Morris Mano
DIGITAL DESIGN, 3e.


microcontroller
output
input
mem
CPU
analog
display
analog
keypad

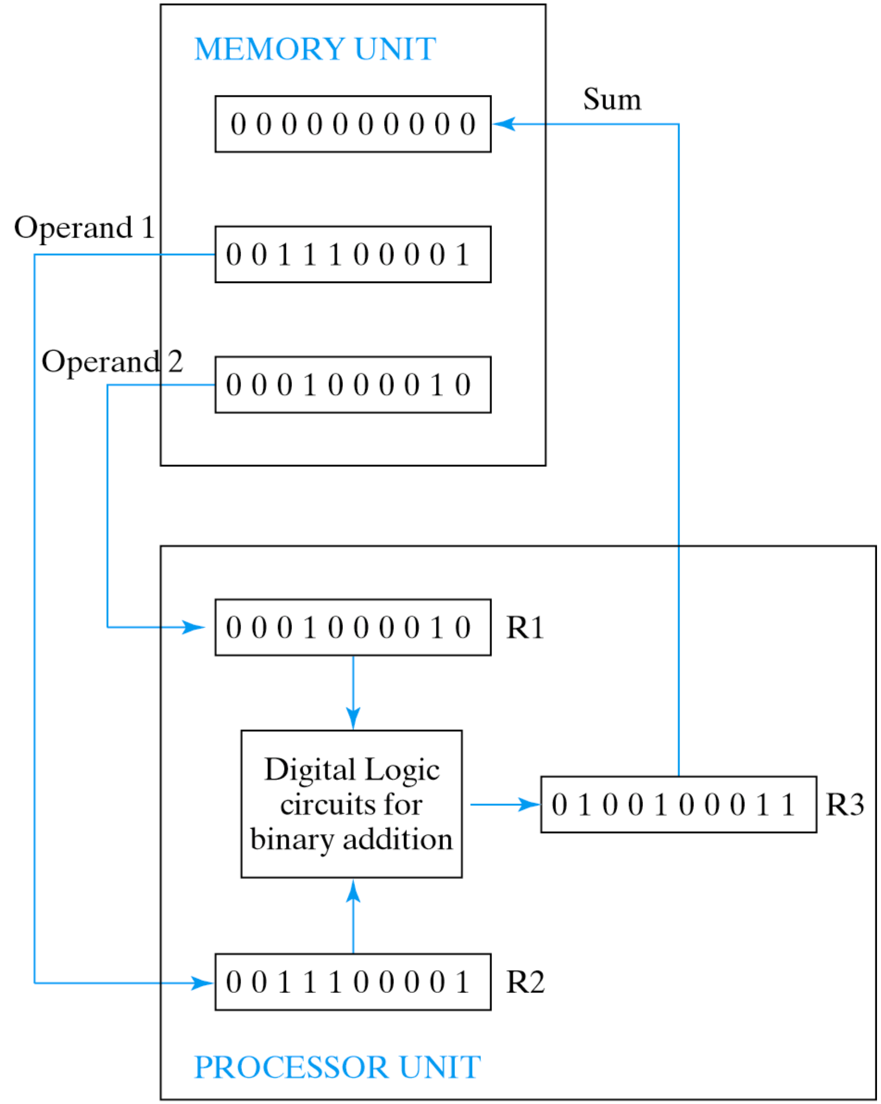


Fig. 1-2 Example of binary information processing

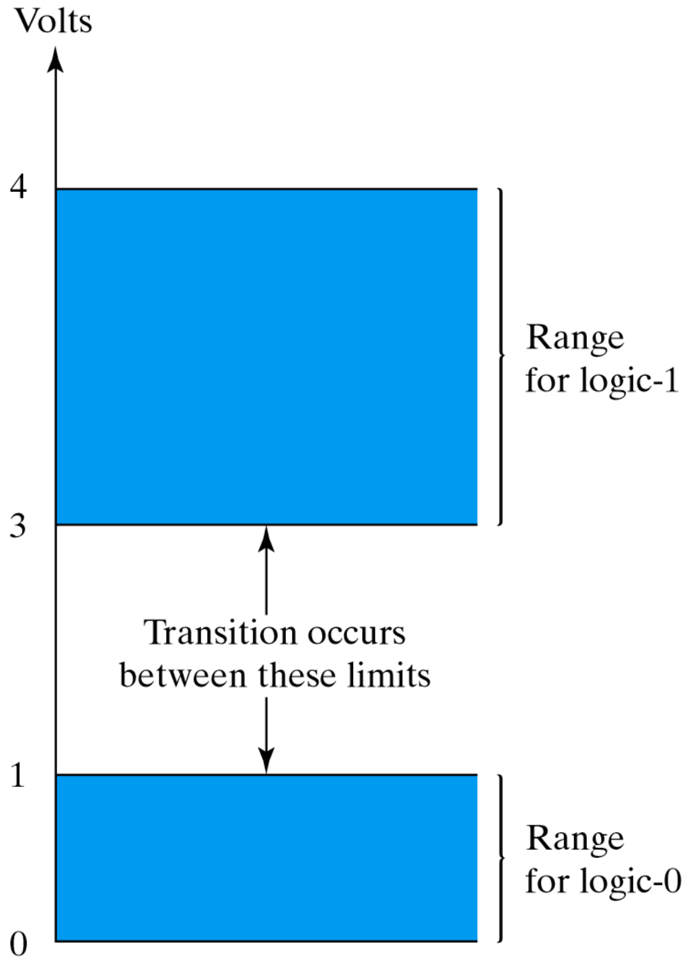


Fig. 1-3 Example of binary signals

Números binarios

- usan base 2
- digitos son 0 y 1
- "digitos" binarios $\Rightarrow$ bits
- 8 bits = 1 byte

Conversión binario-decimal

- posición del bit determina potencia de 2

- $$\begin{array}{ccccc} & 2^3 & & 2^1 & \\ \downarrow & \downarrow & & \downarrow & \\ \boxed{0} & \boxed{1} & \boxed{1} & \boxed{0} & \boxed{1} \\ \uparrow & & \uparrow & & \uparrow \\ 2^4 & & 2^2 & & 2^0 \end{array} \Rightarrow 01101_2 = (1 \times 2^0 + 1 \times 2^2 + 1 \times 2^3)_{10}$$
  $$= 1 + 4 + 8 = 13_{10}$$

- para fracciones, use potencias negativas

$$0.\underset{2^{-1}}{0}\underset{2^{-2}}{1}\underset{2^{-3}}{1}{}_2 = 2^{-2} + 2^{-3} = \frac{1}{4} + \frac{1}{8} = 0.25 + 0.125 = 0.375_{10}$$

subscrito indica base

Octal

- base es 8
- agrupar número binario en grupos de 3 bits y convertir
- digitos validos son $\emptyset - 7$

$$011101.101_2 = 1+4+8+16 + \frac{1}{2} + \frac{1}{8} = 29.625_{10}$$

- $\underbrace{011}_{3}\,\underbrace{101}_{5} . \underbrace{101}_{5}\ {}_2 \rightarrow 35.5_8$

- conversion octal - decimal $\Rightarrow$ puede usar potencias de 8

$$35.5_8 = (3 \times 8^1 + 5 \times 8^0 + 5 \times 8^{-1})_{10}$$
$$= 24 + 5 + \frac{5}{8} = 29.625_{10}$$

hex

- base es 16
- agrupar bits en grupos de 4 y convertir
- digitos validos son $\emptyset$-9, A, B, C, D, E, F
- conversion hex - decimal $\rightarrow$ usar potencias de 16

$$\underbrace{0001}_{1}\,\underbrace{1101}_{D} . \underbrace{1010}_{A} = 1D.A_{16}$$

$$= \underbrace{16}_{1\times16^1} + \underbrace{13}_{13(D)\times16^0} + \underbrace{\frac{10}{16}}_{A(10)\times16^{-1}} = 29.625_{10}$$

Conversion decimal-binario (enteros)

$$\frac{N}{2} = x + y = \text{cociente} + \text{residuo}$$

cociente → nueva N

si residuo $= \phi$, bit es ϕ

" " $= \frac{1}{2}$, bit es 1

Se continua dividiendo entre 2 hasta que N es ϕ

Ejemplo:

$23_{10} \div 2 = 11 + \frac{1}{2} \Rightarrow a_0 = 1$

$11 \div 2 = 5 + \frac{1}{2} \Rightarrow a_1 = 1$

$5 \div 2 = 2 + \frac{1}{2} \Rightarrow a_2 = 1$

$2 \div 2 = 1 + 0 \Rightarrow a_3 = 0$

$1 \div 2 = 0 + \frac{1}{2} \Rightarrow a_4 = 1$

$\therefore 23_{10} = 10111_2$

Fracciones

$N \times 2 = \text{entero} + \text{fraccion}$

entero → nuevo bit

fraccion → nueva N

$.2 \times 2 = 0.4 \rightarrow a_{-1} = 0$

$.4 \times 2 = 0.8 \rightarrow a_{-2} = 0$

$.8 \times 2 = 1.6 \rightarrow a_{-3} = 1$

$.6 \times 2 = 1.2 \rightarrow a_{-4} = 1$

$.2 \times 2 = 0.4 \rightarrow$ vemos que la secuencia se repetirá

$\therefore 0.2_{10} \simeq 0.001100110011\ldots_2$

Suma de números binarios

carry-in	a	b	z = a + b	carry-out
	bits			
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$\begin{array}{r} 9 \\ +\ 6 \\ \hline 15 \end{array} \qquad \begin{array}{r} 01001 \\ 00110 \\ \hline 01111_2 \end{array} = 15_{10}$$

$$\begin{array}{r} 9 \\ +\ 3 \\ \hline 12 \end{array} \qquad \begin{array}{r} {\scriptstyle 1\,1} \\ 01001 \\ 00011 \\ \hline 01100_2 \end{array} = 12_{10}$$

Números con signo

Tres formas de expresarlos:

1. asignar el bit de la izq. al signo y los demás a la magnitud (signo +magnitud)
1. Complemento de 1
3. " " 2

Ejemplo de signo +magnitud (8 bits)

$00101101_2 = 45_{10}$

$10101101_2 = -45_{10}$

Si usamos signo + magnitud, podemos sumar números A y B

- mirando los signos de los dos números
- si el signo es el mismo
  - sumamos magnitudes
  - le asignamos el signo original a la suma
- si los signos no son iguales
  - restamos la magnitud menor de la mayor
  - le asignamos el signo de la mayor al resultado

Signo + magnitud → dificil de implementar

Complemento de 1

Para formar el número negativo, cambiamos los Ø por 1 y los 1 por Ø

Ejemplo

$0010\,1101_2 = +45_{10}$

$1101\,0010_2 = -45_{10}$ (usando nomenclatura de complemento de 1)

Complemento de 2

Le sumamos 1 al complemento de 1

∴ En nomenclatura de complemento de 2

$0010\,1101_2 = +45_{10}$

$1101\,0010$ ← 1's complement

$1101\,0011$ ← 2's complement $= -45_{10}$

Si sumamos +45 y (−45) en binario

$$\begin{array}{rl} 0010\,1101 & +45 \\ 1101\,0011 & -45 \\ \hline 1\,|\,0000\,0000 & \end{array}$$

↑ descartamos porque usamos 8 bits

Ejemplo

$11_{10} + (-6_{10})$ usando 5 bits

$+6 = 00110$

$-6 = 11001 + 1 = 11010$

$$
\begin{array}{rr}
+11 = & 01011 \\
+(-6) = & 11010 \\
\hline
& 1\,|\,00101
\end{array}
\rightarrow \text{resultado} = 5_{10}
$$

$\uparrow$

descartamos

Ejercicios 1.18 , 1.20

Table 1-4
Binary Coded Decimal (BCD)

Decimal symbol	BCD digit
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

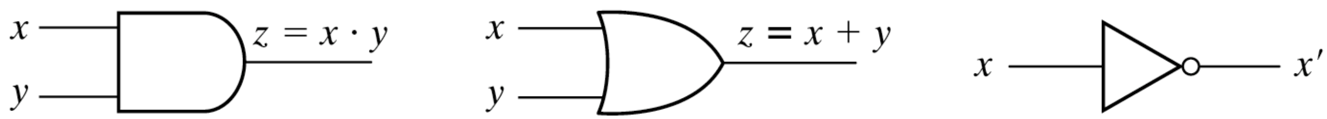
BCD Arithmetic:

Must add 0110_2 if sum is larger that 1010_2

BCD carry	1	1		
	0001	1000	0100	184
	+0101	0111	0110	+576
Binary sum	0111	10000	1010	
Add 6		0110	0110	
BCD sum	0111	0110	0000	760

Table 1-7
American Standard Code for Information Interchange (ASCII)

	$b_7b_6b_5$							
$b_4b_3b_2b_1$	**000**	**001**	**010**	**011**	**100**	**101**	**110**	**111**
0000	NUL	DLE	SP	0	@	P	`	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	−	=	M	]	m	}
1110	SO	RS	.	>	N	∧	n	~
1111	SI	US	/	?	O	—	o	DEL



(a) Two-input AND gate (b) Two-input OR gate (c) NOT gate or inverter

Fig. 1-4 Symbols for digital logic circuits

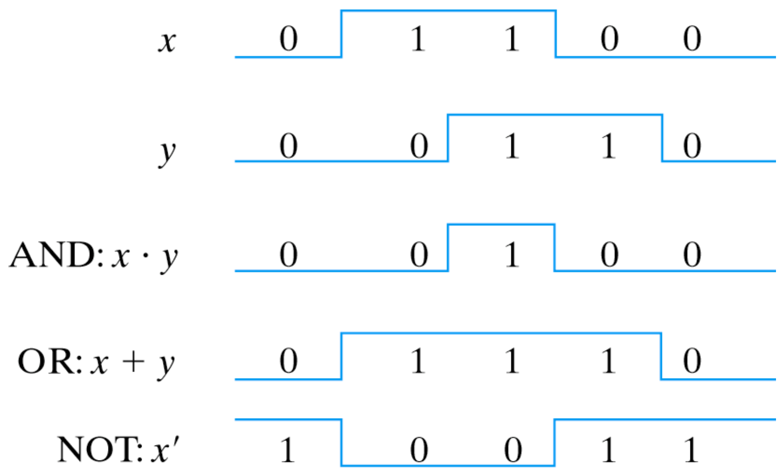
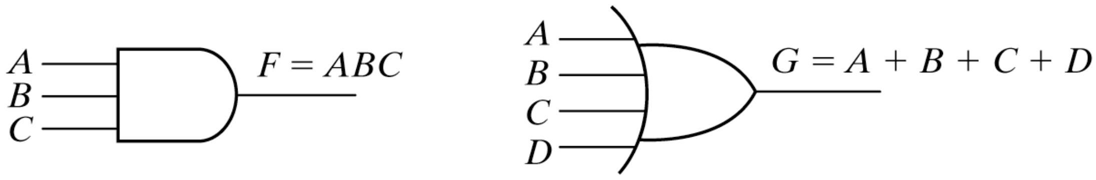


Fig. 1-5 Input-output signals for gates



(a) Three-input AND gate (b) Four-input OR gate

Fig. 1-6 Gates with multiple inputs

M. Morris Mano
DIGITAL DESIGN, 3e.