5.- Funciones ALU

Erratas

Lib. prob ejerc. 5.6

Fig 5.6.2 → minus 1 (ca2)

1.- Representación de números

2.- Sumadores y restadores

3.- " complemento a 1

4.- Comparadores

5.- ALU'S

1.- Representación de números

Sobre 4 bits como ejemplo

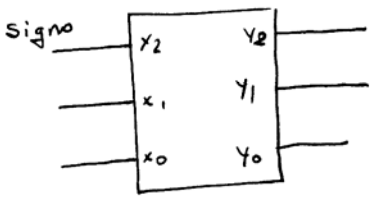
- Binario puro ⇒ 0 → 0000 al 1111 → 15

- Magnitud y signo ⇒ { 0 → positivo; 1 → negativo } ⇒
  - 0111 → +7
  - 0000 → +0 } *
  - ∅000 → −0 } *
  - 1111 → −7

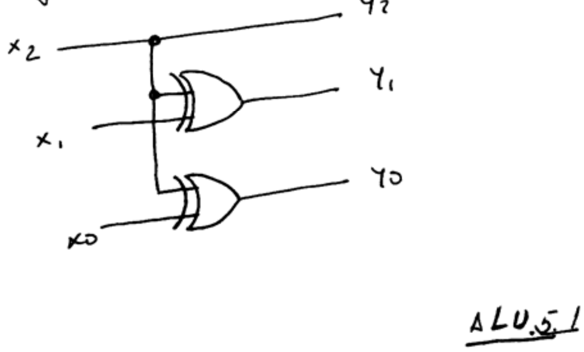
- Complemen. A1 ⇒ { 0xxx → positivo; kkkk → INVERTIR; 1--- → negativo }
  - 0111 → +7
  - 0000 → +0 } *
  - 1111 → −0 } *
  - 1000 → −7

- Complemen A2 ⇒ {
  - 7 − 0111
  - 1 − 0001
  - ∅ − 0000 → −∅ → 1111 +1 = 0000
  - −1 → 0001 → 1110 +1 = 1111
  - −8 → 1000

Codificador M+S → C.a1

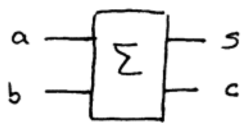


Signo (−) ⇒ (Y_1 e Y_0 = invertidos)



2.- Sumadores y restadores

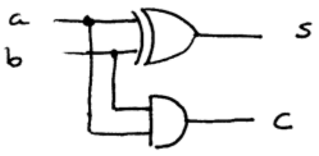
Semisumador



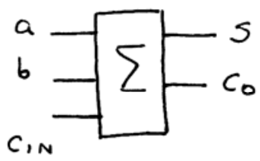
a	b	s	c
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$S = \bar{a}b + a\bar{b} \Rightarrow a \oplus b$

$C = a \cdot b$



Sumador

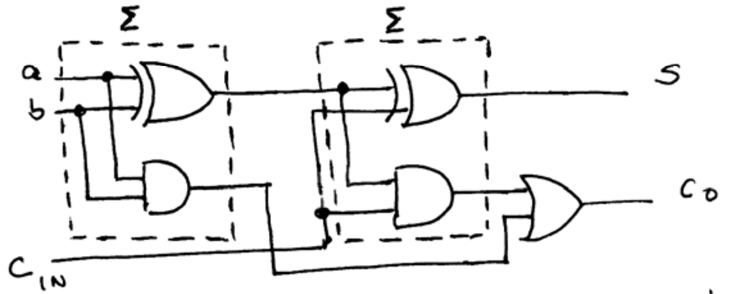


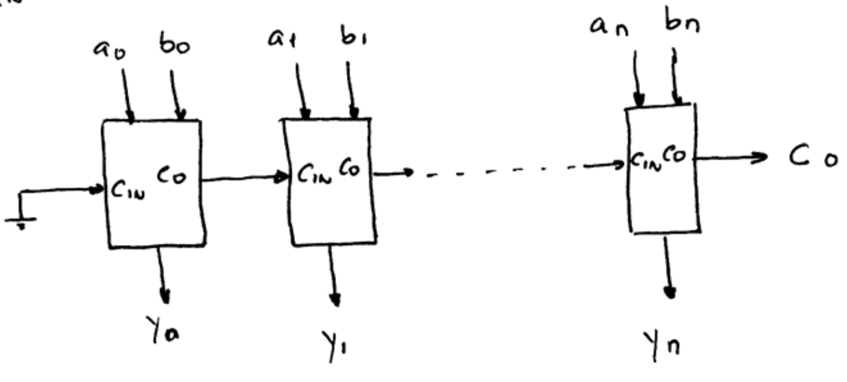
a	b	C_{IN}	S	C_o
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$s = \bar{a}\bar{b}c + \bar{a}b\bar{c} + a\bar{b}\bar{c} + abc = c\underbrace{(\bar{a}\bar{b} + ab)}_{\underbrace{\overline{a \oplus b}}_{\bar{m}}} + \bar{c}\underbrace{(\bar{a}b + a\bar{b})}_{\underbrace{a \oplus b}_{m}}$$

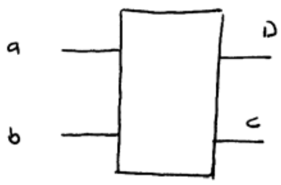
$$S = c\bar{m} + \bar{c}m = c \oplus m = c \oplus (a \oplus b)$$

$$C_o = \underbrace{\bar{a}bc + a\bar{b}c}_{c(a \oplus b)} + \underbrace{ab\bar{c} + abc}_{ab} = ab + c(a \oplus b)$$





Semirestadores

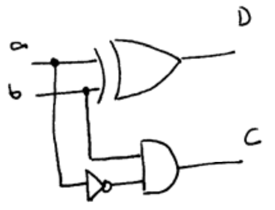


a - b

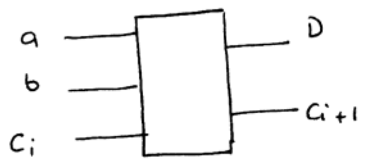
a	b	D	C
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

$D = \bar{a}b + a\bar{b}$

$C = \bar{a}b$



Restador completo



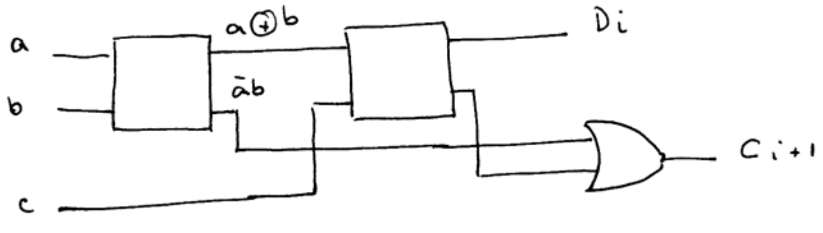
a	b	C_i	D	C_{i+1}
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

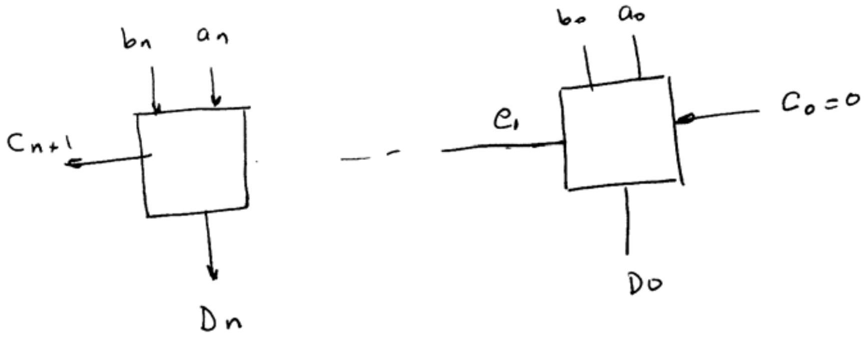
$$D = \underbrace{\underbrace{\bar{a}\bar{b}c + \bar{a}b\bar{c}}_{\bar{a}(b \oplus c)} + \underbrace{a\bar{b}\bar{c} + abc}_{a\overline{(b \oplus c)}}}_{a \oplus b \oplus c}$$

$$C_{i+1} = \bar{a}\bar{b}c + \bar{a}b\bar{c} + \bar{a}bc + abc \qquad \bar{a}b + C_i\overline{(a \oplus b)}$$

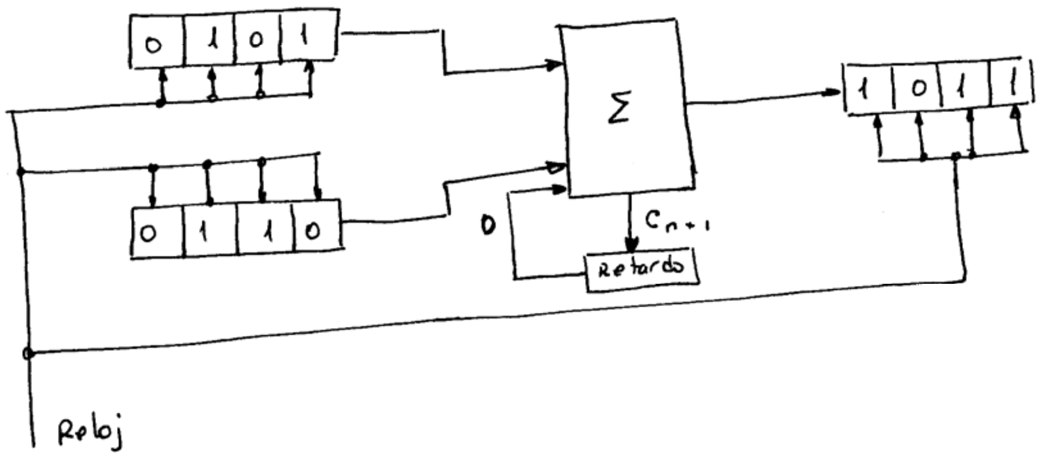
$\bar{a}b$

$c\overline{(a \oplus b)}$





Sumador serie



Acarreo adelantado

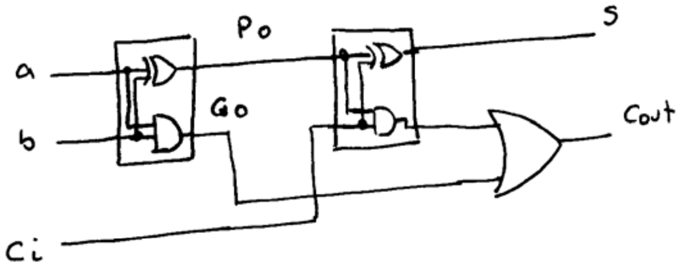
$P_i = a_i \oplus b_i$

$G_i = a_i \cdot b_i$

$S_i = P_i \oplus C_i$

$C_i = G_{i-1} + P_{i-1} C_{i-1}$

sumador



$C_1 = G_0 + P_0 C_0$

$C_2 = G_1 + P_1 \cdot C_1 = G_1 + P_1 (G_0 + P_0 C_0) = G_1 + P_1 G_0 + P_0 P_1 C_0$

$C_3 = G_2 + P_2 \cdot C_2 = G_2 + P_2 (G_1 + P_1 G_0 + P_0 P_1 C_0) = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0$

$C_4 = G_3 + P_3 \cdot C_3 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0 + P_3 P_2 P_1 P_0 C_0$

$\Downarrow$

Gestionar el acarreo desde el inicio

5.3 Sumador complemento a 1 : Rebose

Rebose ⇒ suma resultado más bit que los nºs

↕

Los 2 nºs mismo signo

Comprobar rebose ⇒ Comprobar resultado de la suma

⇓

1º Sumar en bin. puro sin signo ni magnitud

2º (Acarreo = ∅ y no rebose) ⇒ Resultado válido

3º (Acarreo = 1 y no rebose) ⇒ Sumar 1 a resultado

4º (si rebose) ⇒ dar error

No → problema

$$\begin{array}{cc} 00 & 0 \\ 01 & +1 \\ \hline 01 & +1 \end{array} \qquad \begin{array}{cc} 01 & 1 \\ 10 & -1 \\ \hline 11 & -0 \end{array}$$

Sumar 1 a resulta →

$$\begin{array}{cc} 10 & -1 \\ 11 & -0 \\ \hline {}^{1}\,01 & -1 \\ +\ \ 1 & \\ \hline 10 & \leftarrow -1 \end{array}$$

Rebose

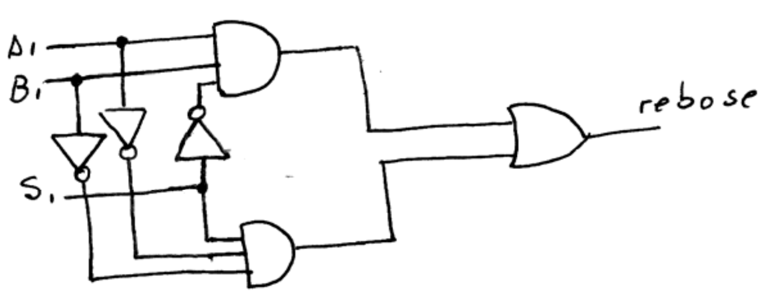
$$\begin{array}{ccc} \boxed{A_1}\ \boxed{A_0} & & \\ 0\ 1 & & +1 \\ \boxed{B_1}\ 0\ 1\ \boxed{B_0} & & +1 \\ \hline \boxed{S_1}\ 1\ 0\ \boxed{S_0} & & +2 \\ 0 & & \\ \Downarrow & & \\ -1 & & \end{array}$$

Rebose

$$\begin{array}{cc} 10 & -1 \\ 10 & -1 \\ \hline 1\ 00 & -2 \\ \Downarrow & \\ 0 & \end{array}$$

⇓

$$rebose = S_1\,\bar{A}_1\,\bar{B}_1 + \bar{S}_1\,A_1\,B_1$$

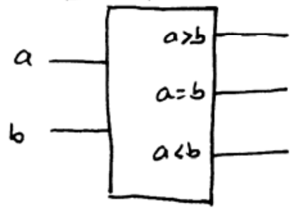


4. Comparadores

Compara dos palabras de n bits y da unas salidas en función de que $A > B$, $A = B$ o $A < B$

El nº de bits de cada palabra da nombre al comparador.

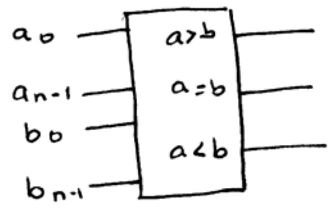
Comp. 1 bit



a	b	a > b	a = b	a < b
0	0	0	1	0
0	1	0	0	1
1	0	1	0	0
1	1	0	1	0

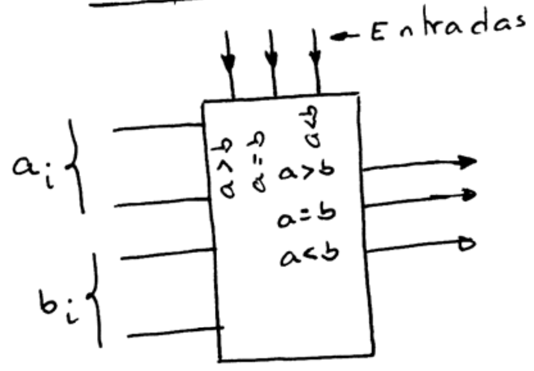
$\boxed{(a > b) = a\bar{b}}$ $\boxed{(a = b) = \bar{a}\bar{b} + ab = \overline{a \oplus b}}$ $\boxed{(a < b) = \bar{a}b}$

Comparador n bits



$a_0 \ldots a_{n_1}$	$b_0 \ldots b_{n_1}$	a>b	a=b	a<b	
0 . . . 0	0 . . . 0	1	0	0	$\leftrightarrow$ a>b
⋮	⋮	0	1	0	$\leftrightarrow$ a=b
		0	0	1	$\leftrightarrow$ a<b

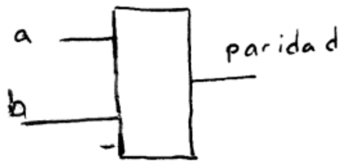
Comparador con entradas para configuración en CASCADA



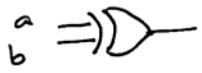
En las entradas se meten las salidas de comparación de los bits inmediatamente interiores. De esta manera siempre que haya una diferencia entre a_i y b_i la salida se posicionará en función de ello. Pero si $a_i = b_i$, la salida tomará el valor de las entradas de la comparación en cascada. Que resulta ser el valor de la comparación de a_{i-1} y b_{i-1}.

Generadores / detectores de paridad

Los generadores de paridad PAR son aquellos circuitos que me generan un "0" cuando el nº de "1" a la entrada es par y un "1" cuando es impar



a	b	paridad
0	0	0
0	1	1
1	0	1
1	1	0



$$paridad = \bar{a}b + a\bar{b} = a \oplus b$$

Para el caso de 3 bits

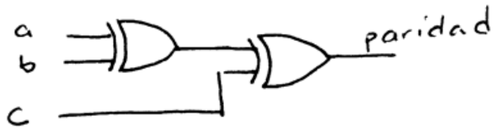
a	b	c	paridad
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

$$paridad = \bar{a}\bar{b}c + \bar{a}b\bar{c} + a\bar{b}\bar{c} + abc =$$

$$= \bar{c}\underbrace{\underbrace{(\bar{a}b + a\bar{b})}_{a \oplus b}}_{m} + c\underbrace{\underbrace{(\bar{a}\bar{b} + ab)}_{\overline{a \oplus b}}}_{\bar{m}} =$$

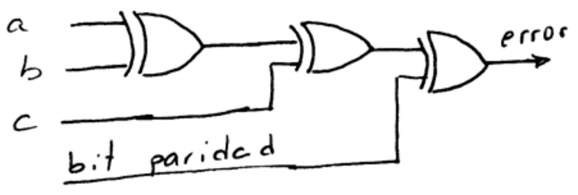
$$= \bar{c}m + c\bar{m} = c \oplus m =$$

$$= c \oplus (a \oplus b)$$

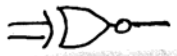


se puede observar que lo único que hay que hacer para ampliar el nº de bits es ir aumentando el nº de puertas.

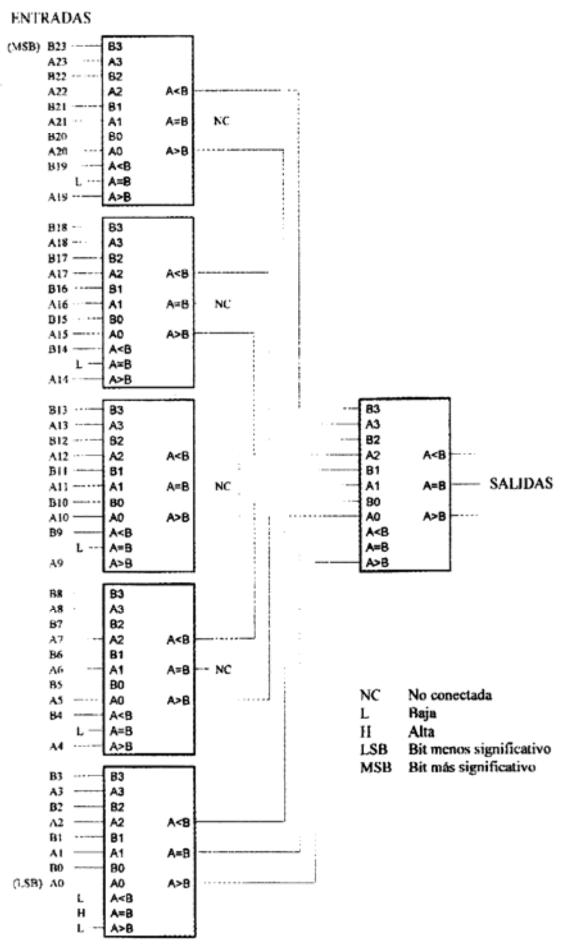
- Para el caso del detector el circuito es identico, solo que el bit de paridad forma parte de la entrada y salida es el detector de error



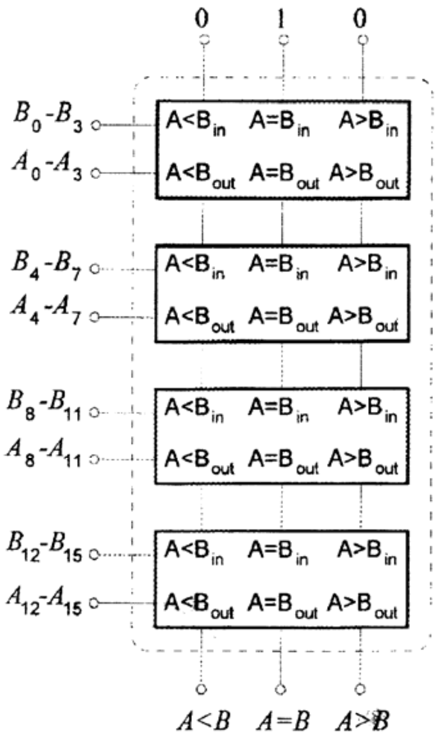
PARA EL CASO DE DETECTOR/GENERADOR DE PARIDAD IMPAR LO UNICO QUE HAY QUE HACER ES SUSTITUIR LA ULTIMA PUERTA POR UNA NOR-EXCLUSIVE



Comparador de 24 bits



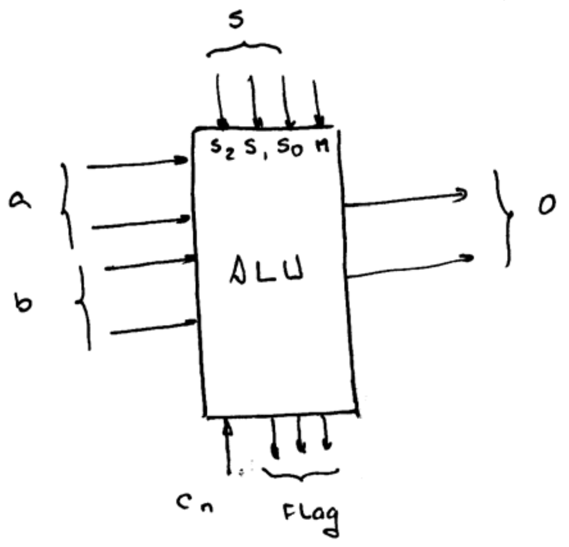
Comparador en cascada de 16 bits



5.- ALU's

Basadas en multiplexores que seleccionan las funciones implementadas en las entradas.

Tabla 5.23 (Pg 292)



Arithmetic logic unit 74F181

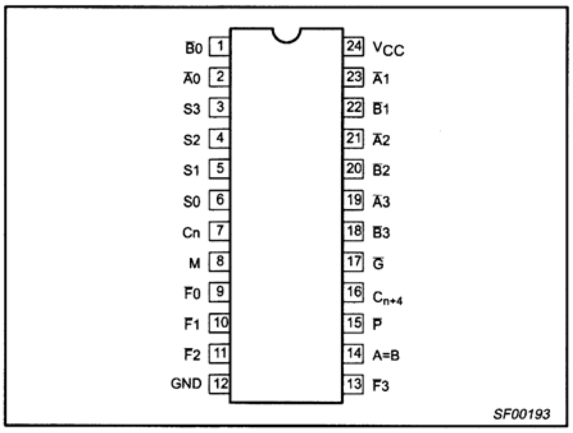
FEATURES

- Provides 16 arithmetic operation: add, subtract, compare, and double; plus 12 other arithmetic operations
- Provides all 16 logic operations of two variables: Exclusive-OR, Compare, AND, NAND, NOR, OR, plus 10 other logic operations
- Full look-ahead carry for high speed arithmetic operation on long words
- 40% faster than 'S181 with only 30% 'S181 power consumption
- Available in 300mil-wide Slim 24-pin Dual In-Line package

DESCRIPTION

The 74F181 is a 4-bit high-speed parallel Arithmetic Logic Unit (ALU). Controlled by the four Function Select inputs (S0–S3) and the Mode Control input (M), it can perform all the 16 possible logic operations or 16 different arithmetic operations on active-High or active-Low operands. The Function Table lists these operations.

PIN CONFIGURATION



TYPE	TYPICAL PROPAGATION DELAY	TYPICAL SUPPLY CURRENT (TOTAL)
74F181	7.0ns	43mA

ORDERING INFORMATION

DESCRIPTION	COMMERCIAL RANGE V_{CC} = 5V ±10%, T_{amb} = 0°C to +70°C
24-Pin Plastic Slim DIP (300 mil)	N74F181N
24-Pin Plastic SOL	N74F181D

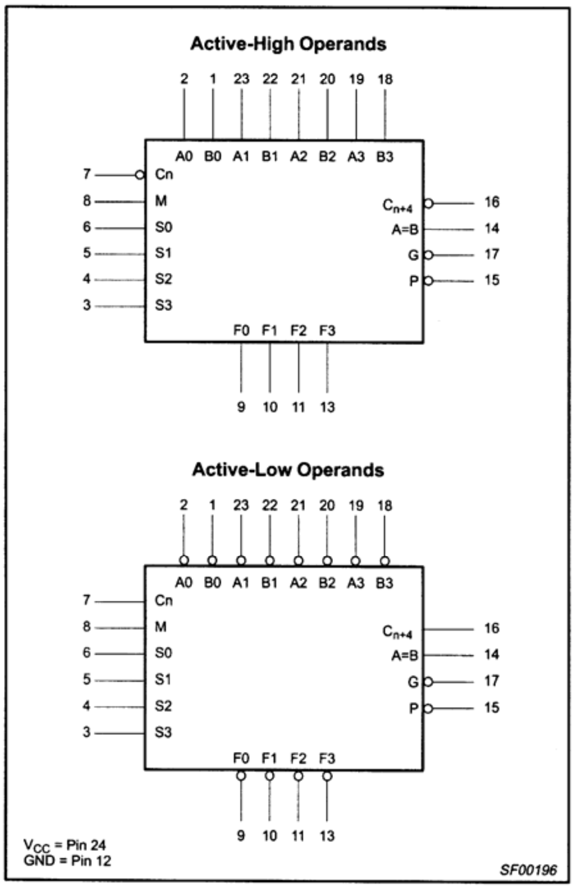
INPUT AND OUTPUT LOADING AND FAN-OUT TABLE

PINS	DESCRIPTION	74F (U.L.) HIGH/LOW	LOAD VALUE HIGH/LOW
$\overline{A}0$–$\overline{A}3$	A operand inputs	1.0/3.0	20μA/1.8mA
$\overline{B}0$–$\overline{B}3$	B operand inputs	1.0/3.0	20μA/1.8mA
M	Mode control input	1.0/1.0	20μA/0.6mA
S0–S3	Function select input	1.0/4.0	20μA/2.4mA
Cn	Carry input	1.0/5.0	20μA/3.0mA
C_{n+4}	Carry output	50/33	1.0mA/20mA
$\overline{P}$	Carry Propagate output	50/33	1.0mA/20mA
$\overline{G}$	Carry Generate output	50/33	1.0mA/20mA
A=B	Compare output	OC/33	OC/20mA
$\overline{F}0$–$\overline{F}3$	Outputs	50/33	1.0mA/20mA

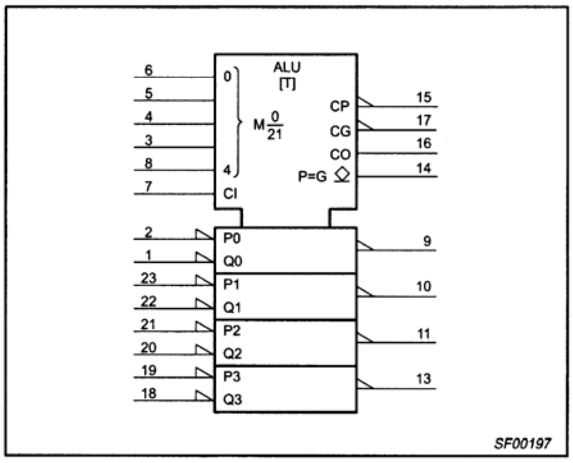
NOTE: One (1.0) FAST unit load is defined as: 20μA in the High state and 0.6mA in the Low state.
OC = Open Collector

ALU.5.b

LOGIC SYMBOL

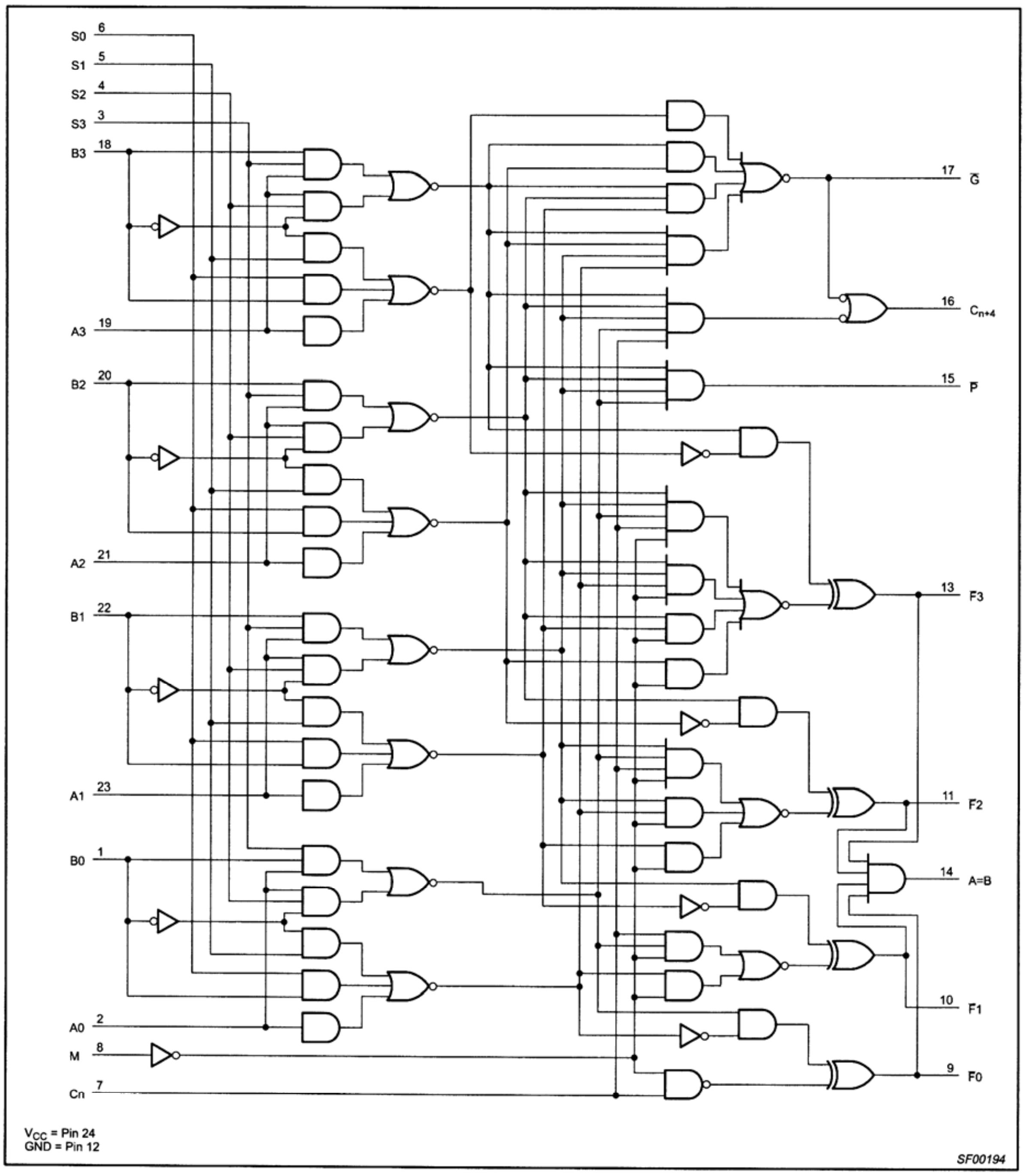


IEC/IEEE SYMBOL



ΔLy.5.11

LOGIC DIAGRAM



BLU.5.12

When the Mode Control input (M) is High, all internal carries are inhibited and the device performs logic operations on the individual bits as listed. When the Mode control input is Low, the carries are enabled and the device performs arithmetic operations on the two 4-bit words. The device incorporates full internal carry look-ahead and provides for either ripple carry between device using the C_{n+4} output, or for carry look-ahead between packages using the signals $\overline{P}$ (Carry Propagate) and $\overline{G}$ (Carry Generate). $\overline{P}$ and $\overline{G}$ are not affected by carry in. When speed requirements are not stringent, it can be used in a simple ripple carry mode by connecting the Carry output (C_{n+4}) signal to the Carry input (Cn) of the next unit. For high-speed operation, the device is used in conjunction with the 74F182 carry look-ahead circuit. One carry look-ahead package is required for each group of four 74F181 devices. Carry look-ahead can be provided at various levels and offers high speed capability over extremely long word lengths.

The A=B output from the device goes High when all four F outputs are High and can be used to indicate logic equivalence over 4-bits when the unit is in the subtract mode. The A=B output is open-collector and can be wired-AND with other A=B outputs to give a comparison for more than 4 bits. The A=B signal can also be used with the C_{n+4} signal to indicate A>B and A<B. The Function Table lists the arithmetic operations that are performed without a carry in. An incoming carry adds a one to each operation. Thus select code LHHL generates A minus B minus 1 (two's complement notation) without a carry in and generates A minus B when a carry is applied. Because subtraction is actually performed by complementary addition (one's complement), a carry out means borrow; thus, a carry is generated when there is no underflow and no carry is generated when there is underflow. As indicated, this device can be used with either active-Low inputs producing active-Low outputs or with active-High inputs producing active-High outputs. For either case, the table lists the operations that are performed to the operands labeled inside the logic symbol.

MODE-SELECT FUNCTION TABLE

MODE SELECT INPUTS				ACTIVE HIGH INPUTS & OUTPUTS		ACTIVE LOW INPUTS & OUTPUTS	
S3	S2	S1	S0	Logic (M=H)	Arithmetic** (M=L) (Cn=H)	Logic (M=H)	Arithmetic** (M=L) (Cn=L)
L	L	L	L	$\overline{A}$	A	$\overline{A}$	A minus 1
L	L	L	H	$\overline{A+B}$	A+B	$\overline{AB}$	AB minus 1
L	L	H	L	$\overline{A}B$	$A+\overline{B}$	$\overline{A}+B$	$A\overline{B}$ minus 1
L	L	H	H	Logical 0	minus 1	Logical 1	minus 1
L	H	L	L	$\overline{AB}$	A plus $A\overline{B}$	$\overline{A+B}$	A plus $(A+\overline{B})$
L	H	L	H	$\overline{B}$	(A+B) plus $A\overline{B}$	$\overline{B}$	AB plus $(A+\overline{B})$
L	H	H	L	$A\oplus B$	A minus B minus 1	$\overline{A\oplus B}$	A minus B minus 1
L	H	H	H	$A\overline{B}$	AB minus 1	$A+\overline{B}$	$A+\overline{B}$
H	L	L	L	$\overline{A}+B$	A plus AB	$\overline{A}B$	A plus (A+B)
H	L	L	H	$\overline{A\oplus B}$	A plus B	$A\oplus B$	A plus B
H	L	H	L	B	$(A+\overline{B})$ plus AB	B	$A\overline{B}$ plus (A+B)
H	L	H	H	AB	AB minus 1	A+B	A+B
H	H	L	L	Logical 1	A plus A*	Logical 0	A plus A*
H	H	L	H	$A+\overline{B}$	(A+B) plus A	$A\overline{B}$	AB plus A
H	H	H	L	A+B	$(A+\overline{B})$ plus A	AB	$A\overline{B}$ plus A
H	H	H	H	A	A minus 1	A	A

H = High voltage level
L = Low voltage level
* = Each bit is shifted to the next more significant position.
** = Arithmetic operations expressed in two's complement notation.

ALU.5.13