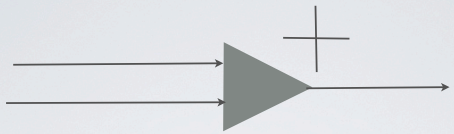


SUMADORES

Iván Arozamena Balbín
Eric Alvarado Salgueiro
Adrián Alonso Sangrones

QUE SON LOS SUMADORES

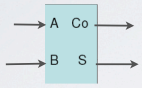
En electrónica un **sumador** es un circuito lógico que calcula la operación suma. En los computadores modernos se encuentra en lo que se denomina Unidad aritmético lógica (ALU).
Generalmente realizan las operaciones aritméticas en código binario decimal o BCD exceso 3, por regla general los sumadores emplean el sistema binario. En los casos en los que se esté empleando un complemento a dos para representar números negativos el sumador se convertirá en un sumador- substractor (*Adder-subtracter*).

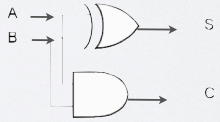
Tipos de sumadores:
Half-adder.
Full-Adder.
Metodo Ripple
Carry-Look-Ahead.
Carry-select.

SEMISUMADOR

- Se denomina **semisumador** al circuito combinacional capaz de realizar la suma aritmética binaria de dos únicos bits A y B, proporcionando a su salida un bit resultado de suma S y un bit de acarreo C. En la siguiente figura se muestra la tabla de verdad de este circuito con sus funciones, acompañado de un esquema del Half-Adder.

A + B	Co	S
0 0	0	0
0 1	0	1
1 0	0	1
1 1	1	0





VHDL SEMISUMADOR

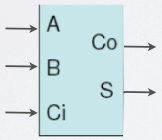
```
entity parasum is
generic (N: integer := 4);
port (A,B:  in bit_vector(N downto 1);
      S:    out bit_vector(N downto 1);
      Cout: out bit);
end parasum;

architecture genera of parasum is
signal C: bit_vector(N downto 1);
begin
G1: for i in 1 to N generate
  G2: if ( i = 1 ) generate
    S(i) <= A(i) xor B(i);
    C(i) <= A(i) and B(i);
  end generate G2;
  G3: if ( i /= 1 ) generate
    S(i) <= A(i) xor B(i) xor C(i-1);
    C(i) <= (A(i) and B(i)) or (A(i) and C(i-1)) or (B(i) and C(i-1));
  end generate G3;
end generate G1;
Cout <= C(4);
end genera;
```

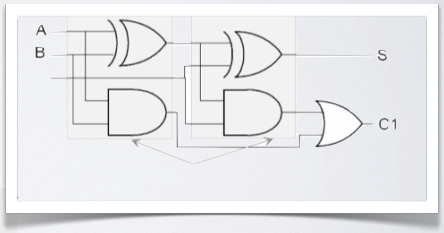
FULL ADDER

- Este dispositivo nos ofrece una mejora del semisumador al cual se le añade un acarreo de entrada.

De está manera podemos afrontar sumas de mas de un bit para las cuales utilizaremos el acarreo de salida del anterior en el acarreo de entrada del siguiente. Así completamos la suma correctamente.
A continuación vemos la tabla de verdad y un esquema.

A	+	B	+	Ci	Co	S
0		0		0	0	0
0		0		1	0	1
0		1		0	0	1
0		1		1	1	0
1		0		0	0	1
1		0		1	1	0
1		1		0	1	0
1		1		1	1	1





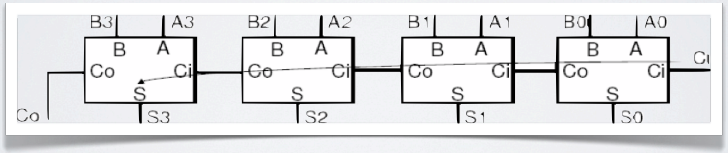
$S = A \oplus B \oplus Ci$

$Co = A\,B + A\,Ci + B\,Ci$

METODO RIPPLE

- Un sumador de dos informaciones binarias A+B de ***n*** bits necesita realizar ***n*** sumas

parciales, empleando para ello ***n*** sumadores completos. Esto nos hace conectar el acarreo de salida con el siguiente acarreo de entrada de
manera que podamos realizar la suma del siguiente bit con acarreo.
Es un circuito muy simple e intuitivo pero presenta el serio inconveniente de tener que esperar un tiempo igual a ***n*** **tiempos de propagación** antes de obtener un resultado estable.



CARRY LOOK AHEAD

- Este sumador, llamado también sumador paralelo con acarreo anticipado, realiza la suma aumentando la velocidad de proceso sobre la conexión en serie. Lo logra mediante la generación de todos los bits de acarreo en el mismo proceso de calculo de las sumas parciales.

Al sumar dos informaciones se obtendrá el acarreo por dos posibilidades: • Se genera acarreo en la propia etapa del sumador.
Generado (A=B=1)
Gj = Aj * Bj •
Proviene de la etapa anterior. **Propagado**
Pj = Aj ⊕ Bj

Por tanto el acarreo producido en la etapa i-esima C_i será porque se genera o propaga y se expresará: $C_i = G_i + P_i C_{i-1} = A_i B_i + (A_i + B_i) C_{i-1}$

CARRY LOOK AHEAD

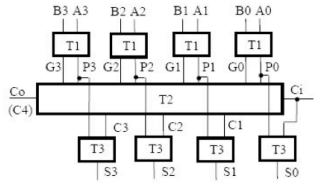
Para un sumador de cuatro bits (i=0, … , 3) se tiene que:

- Para 4 bits se generan los acarreos en paralelo:

T2

C1 = G0 + P0Ci
C2 = G1 + P1 C1 = G1 + P1 (G0 + P0 Ci) = G1 + P1G2 + P1P0Ci
C3 = G2 + P2 C2 = G2 + P2G1 + P2P1G0 + P2P1P0Ci
Co = C4 = G3 + P3 C3 = G3 + P3G2 + P3P2G1 + P3P2P1G0 +
+ P4P3P2P1Ci

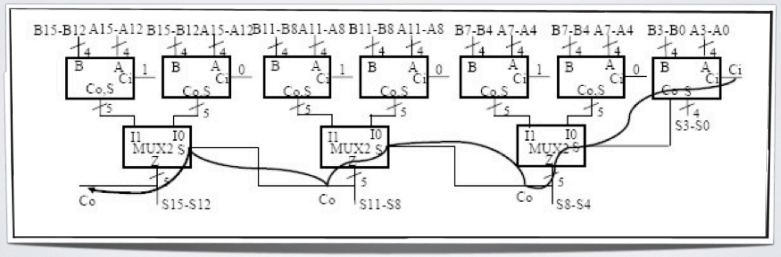
$S_j = A_i \oplus B_j \oplus C_i = P_j \oplus C_i$



CARRY SELECT

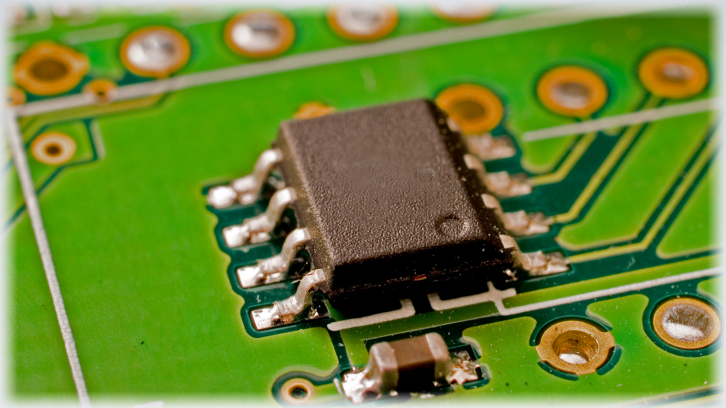
- En este tipo de sumador se realiza un acarreo mixto basado en sumadores y multiplexores, donde la generación de acarreo en cada sumador se realiza en paralelo y la propagación en cada multiplexor en serie.

El tiempo de propagación de este sumador depende del tiempo de propagación de la primera etapa, más el tiempo de propagación de los (M/N-1) multiplexores para propagación del acarreo. A cambio el circuito es bastante más grande que la estructura “ripple”.



VHDL

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity sumauns is
generic(N: integer:=4);
port (A, B: in std_logic_vector(N downto 1);
      Sum: out std_logic_vector(N+1 downto 1));
end sumauns;

architecture uno of sumauns is
begin
Sum <= ('0' & A) + ('0' & B);
end uno;
```



CIRCUITO COMERCIAL

74283

74283

- Circuito sumador
- 8 bit 25nseg
- 16 bit 45nseg

54LS283/DM54LS283/DM74LS283
4-Bit Binary Adders with Fast Carry

General Description

These full adders perform the addition of two 4-bit binary numbers. The sum (Σ) outputs are provided for each bit and the resultant carry (C4) is obtained from the fourth bit. These adders feature full internal look ahead across all four bits. This provides the system designer with partial look-ahead performance at the economy and reduced package count of a ripple-carry implementation.

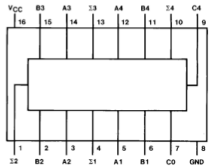
The adder logic, including the carry, is implemented in its true form meaning that the end-around carry can be accomplished without the need for logic or level inversion.

Features

- Full-carry look-ahead across the four bits
- Systems achieve partial look-ahead performance with the economy of ripple carry
- Typical add times
  Two 8-bit words 25 ns
  Two 16-bit words 45 ns
- Typical power dissipation per 4-bit adder 95 mW
- Alternate Military/Aerospace device (54LS283) is available. Contact a National Semiconductor Sales Office/Distributor for specifications.

Connection Diagram

Dual-In-Line Package



TL/F/6421–1

Order Number 54LS283DMQB, 54LS283FMQB, 54LS283LMQB, DM54LS283J, DM54LS283W, DM74LS283M or DM74LS283N
See NS Package Number E20A, J16A, M16A, N16E or W16A

74283

- Condiciones operativas
- VCC 5
- VIH 2
- VIL 0.7
- VOH 3.4
- VOL 0.25

Recommended Operating Conditions

Symbol	Parameter	DM54LS283			DM74LS283			Units
		Min	Nom	Max	Min	Nom	Max	
V_{CC}	Supply Voltage	4.5	5	5.5	4.75	5	5.25	V
V_{IH}	High Level Input Voltage	2			2			V
V_{IL}	Low Level Input Voltage			0.7			0.8	V
I_{OH}	High Level Output Current			−0.4			−0.4	mA
I_{OL}	Low Level Output Current			4			8	mA
T_A	Free Air Operating Temperature	−55		125	0		70	°C

Electrical Characteristics over recommended operating free air temperature range (unless otherwise noted)

Symbol	Parameter	Conditions		Min	Typ (Note 1)	Max	Units
V_I	Input Clamp Voltage	V_{CC} = Min, I_I = −18 mA				−1.5	V
V_{OH}	High Level Output Voltage	V_{CC} = Min, I_{OH} = Max, V_{IL} = Max, V_{IH} = Min	DM54	2.5	3.4		V
			DM74	2.7	3.4		
V_{OL}	Low Level Output Voltage	V_{CC} = Min, I_{OL} = Max, V_{IL} = Max, V_{IH} = Min	DM54		0.25	0.4	V
			DM74		0.35	0.5	
		I_{OL} = 4 mA, V_{CC} = Min	DM74		0.25	0.4	
I_I	Input Current @ Max Input Voltage	V_{CC} = Max, V_I = 7V	A, B			0.2	mA
			C0			0.1	
I_{IH}	High Level Input Current	V_{CC} = Max, V_I = 2.7V	A, B			40	µA
			C0			20	
I_{IL}	Low Level Input Current	V_{CC} = Max, V_I = 0.4V	A, B			−0.8	mA
			C0			−0.4	
I_{OS}	Short Circuit Output Current	V_{CC} = Max (Note 2)	DM54	−20		−100	mA
			DM74	−20		−100	
I_{CC1}	Supply Current	V_{CC} = Max (Note 3)			19	34	mA
I_{CC2}	Supply Current	V_{CC} = Max (Note 4)			22	39	mA

74283

Switching Characteristics at V_{CC} = 5V and T_A = 25°C (See Section 1 for Test Waveforms and Output Load)

Symbol	Parameter	From (Input) To (Output)	R_L = 2 kΩ				Units
			C_L = 15 pF		C_L = 50 pF		
			Min	Max	Min	Max	
t_{PLH}	Propagation Delay Time Low to High Level Output	C0 to Σ1, Σ2		24		28	ns
t_{PHL}	Propagation Delay Time High to Low Level Output	C0 to Σ1, Σ2		24		30	ns
t_{PLH}	Propagation Delay Time Low to High Level Output	C0 to Σ3		24		28	ns
t_{PHL}	Propagation Delay Time High to Low Level Output	C0 to Σ3		24		30	ns
t_{PLH}	Propagation Delay Time Low to High Level Output	C0 to Σ4		24		28	ns
t_{PHL}	Propagation Delay Time High to Low Level Output	C0 to Σ4		24		30	ns
t_{PLH}	Propagation Delay Time Low to High Level Output	A_i or B_i to Σ_i		24		28	ns
t_{PHL}	Propagation Delay Time High to Low Level Output	A_i or B_i to Σ_i		24		30	ns
t_{PLH}	Propagation Delay Time Low to High Level Output	C0 to C4		17		24	ns
t_{PHL}	Propagation Delay Time High to Low Level Output	C0 to C4		17		25	ns
t_{PLH}	Propagation Delay Time Low to High Level Output	A_i or B_i to C4		17		24	ns
t_{PHL}	Propagation Delay Time High to Low Level Output	A_i or B_i to C4		17		26	ns

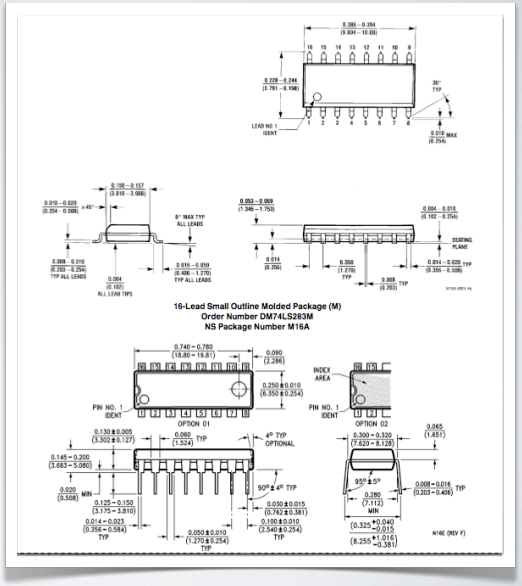
74283

Function Table

Input				Outputs					
				When C0 = L			When C0 = H		
				When C2 = L			When C2 = H		
A1 A3	B1 B3	A2 A4	B2 B4	Σ1 Σ3	Σ2 Σ4	C2 C4	Σ1 Σ3	Σ2 Σ4	C2 C4
L	L	L	L	L	L	L	H	L	L
H	L	L	L	H	L	L	L	H	L
L	H	L	L	H	L	L	L	H	L
H	H	L	L	L	H	L	H	H	L
L	L	H	L	L	H	L	H	H	L
H	L	H	L	H	H	L	L	L	H
L	H	H	L	H	H	L	L	L	H
H	H	H	L	L	L	H	H	L	H
L	L	L	H	L	H	L	H	H	L
H	L	L	H	H	H	L	L	L	H
L	H	L	H	H	H	L	L	L	H
H	H	L	H	L	L	H	H	L	H
L	L	H	H	L	L	H	H	L	H
H	L	H	H	H	L	H	L	H	H
L	H	H	H	H	L	H	L	H	H
H	H	H	H	L	H	H	H	H	H

74283

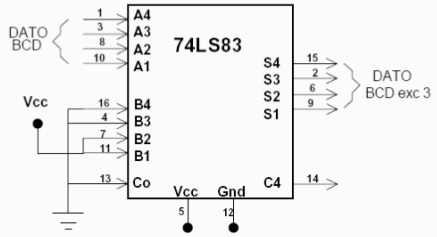
Diagrama logico
y fisico
del microchip



16-Lead Small Outline Molded Package (M)
Order Number DM74LS283M
NS Package Number M16A

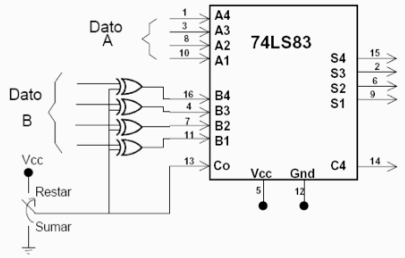
PROBLEMAS

Convertidor BCD - BCD exceso tres.- Una aplicación directa de un sumador de cuatro bits como el 7483 es un convertidor de BCD a BCD exceso tres que se puede realizar sumado al dato de entrada A, una constante $B = 3 = (0011)_2$ como se muestra en la figura siguiente



PROBLEMAS

Sumador - Sustractor de 4 bits.- Utilizando el método expuesto en el segundo capítulo para realizar la resta A-B usando la suma de A + complemento a dos de B podemos realizar un sumador/restador binario de cuatro bits como sigue



Obsérvese que el bloque de cuatro puertas EXOR realiza el complemento a uno del dato B cuando el switch está en la posición de restar y Co le suma 1 a este complemento a uno de B para obtener su complemento a dos.

PROBLEMAS

Sumador BCD.- El problema de sumar dos datos BCD usando un sumador binario (como el 7483) ocurre cuando el resultado de la suma es mayor que 9, ya que entonces el sumador binario producirá un resultado erróneo en BCD.

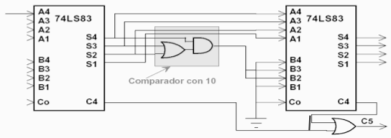
Por ejemplo, al sumar $4+7$ el resultado binario será $15=(1111)_2$ mientras que el resultado esperado en BCD es $15=(1\ 0101)_{BCD}$. Obsérvese que si al 15 producido por el sumador binario le sumáramos un 6: $15+6=21\ =(10101)_2$ ¡El resultado sería correcto en BCD!

Lo ilustrado en el caso de la suma $4+7$ se cumple en general, de manera que para realizar una suma de dos datos BCD se procederá de la siguiente manera:

i. Si el resultado es menor que 10 es correcto tanto en binario como en BCD
ii. Si el resultado es mayor o igual que 10, el resultado correcto en BCD es el resultado en binario más 6

Lo anterior se puede resolver usando un par de sumadores binarios: unao para realizar la primera suma y otro para realizar la corrección (sumar 6) en el caso necesario. Además se requiere un circuito lógico comparador para que active un indicador de que el resultado es mayor o igual que 10.

En la siguiente figura se muestra la implementación del sumador de dos dígitos BCD



PROBLEMAS PROPUESTOS

1• Diseñar utilizando únicamente semisumadores y sumadores completos un circuito digital que realice la multiplicación de un números binario de dos bits por otro de tres bits

2•Diseñar un circuito que realice la operación aritmética:
O = 6X+Y

3•Realizar la suma de 2 números de 16 bits
utilizando el menor número posible de sumadores completos ("74LS83").

4•Diseñar un circuito que realice la operación aritmética:
O = 5 X + 2 Y + Z
para operandos X (x1x0), Y (y1y0) y Z (z1z0) de dos bits, utilizando el
menor número posible de semisumadores de dos bits de operandos
de entradas A (a1a0) y B (b1b0)