Ψηφιακά Κυκλώματα Αριθμητικών Πράξεων

Δυαδικός Αθροιστής

Δεκαδικός Αθροιστής

Συγκριτής Μεγέθους

Δυαδικός Πολλαπλασιαστής

Συστήματα Αρίθμησης

- Ένα σύστημα αριθμών χρησιμοποιεί ένα σύνολο συμβόλων γνωστό ως **ψηφία**.
- Υπάρχουν διάφορα **συστήματα αριθμών** όπως το δεκαδικό, το δυαδικό, το οκταδικό, κλπ.

Στο δεκαδικό σύστημα χρησιμοποιούνται δέκα ψηφία 0, 1, 2, 3, 4, 5, 6, 7, 8 & 9 , ενώ το 10 ορίζεται ως **βάση του συστήματος**.

Παράδειγμα

- $809{,}12 = 8\text{x}10^{2} + 0\text{x}10^{1} + 9\text{x}10^{0} + 1\text{x}10^{-1} + 2\text{x}10^{-2}$

- Η γενική μορφή της απεικόνισης στο δεκαδικό σύστημα είναι:

$$D_{10} = d_n \cdot 10^{n} + d_{n-1} \cdot 10^{n-1} + \ldots + d_1 \cdot 10^{1} + d_0 \cdot 10^{0} + d_{-1} \cdot 10^{-1} + d_{-2} \cdot 10^{-2} + \ldots + d_{-n} \cdot 10^{-n}$$

Συστήματα Αρίθμησης

Οι αριθμοί μπορεί παρασταθούν σε συστήματα που έχουν βάσεις διάφορες του 10

Με βάση 16, δεκαεξαδικό σύστημα

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

Με βάση 8, οκταδικό σύστημα

Ψηφία: 0,1,2,3,4,5,6,7

Με βάση 2, δυαδικό σύστημα

Ψηφία: 0,1

Συστήματα Αρίθμησης

Δεκαδικό Σύστημα

$(\mathbf{215})_{10} = \mathbf{2}x10^2 + \mathbf{1}X10^1 + \mathbf{5}X10^0$

Δεκαεξαδικό Σύστημα

$(\mathbf{D7})_{16} = \mathbf{(13)}x16^1 + \mathbf{7}X16^0 = \mathbf{D}X16^1 + \mathbf{7}X16^0$

Οκταδικό Σύστημα

$(\mathbf{327})_8 = \mathbf{3}x8^2 + \mathbf{2}X8^1 + \mathbf{7}X8^0$

Δυαδικό Σύστημα

$(\mathbf{11010111})_2 = \mathbf{1}x2^7 + \mathbf{1}X2^6 + \mathbf{0}X2^5 + \mathbf{1}X2^4 + \mathbf{0}X2^3 + \mathbf{1}X2^2 + \mathbf{1}X2^1 + \mathbf{1}X2^0$

Πρόσθεση δυαδικών αριθμών

- Η πρόσθεση δύο δυαδικών αριθμών γίνεται, όπως και των δεκαδικών, αρχίζοντας από το τέλος και προσθέτοντας τα ψηφία που βρίσκονται το ένα κάτω από το άλλο.
- Χρησιμοποιούνται μόνο τα ψηφία που ανήκουν στο δυαδικό σύστημα, τα οποία ως γνωστό είναι το "0" και το "1".
- Κρατούμενο (carry) προκύπτει όταν το άθροισμα (sum) των δύο ψηφίων είναι μεγαλύτερο του "1".
- Το κρατούμενο μεταφέρεται και προστίθεται στο αμέσως επόμενο ζευγάρι των υπό πρόσθεση δυαδικών ψηφίων.

Παραδείγματα Πρόσθεσης Δυο Δυαδικών Αριθμών

	Δυαδικό				Δεκαδικό	HEX
κρατουμενο	0	0	1			
	0	1	0	1	(5)	(5)
+	1	0	0	1	(9)	(9)
	1	1	1	0	(14)	(E)

	Δυαδικό					Δεκαδικό	HEX
κρατουμενο	1	1	1	1			
		0	1	1	1	(7)	(7)
+		1	1	0	1	(13)	(D)
	1	0	1	0	0	(20)	(14)

Πρόσθεση δύο δυαδικών ψηφίων

	Κρατούμενο (Carry)	Άθροισμα (Sum)
0+0	0	0
0+1	0	1
1+1	1	0

Ημιαθροιστής (Half adder)

A	B		C	S
0	0	(0+0) =0	0	0
0	1	(0+1) = 1	0	1
1	0	(1+0) =1	0	1
1	1	(1+1) = 2	1	0

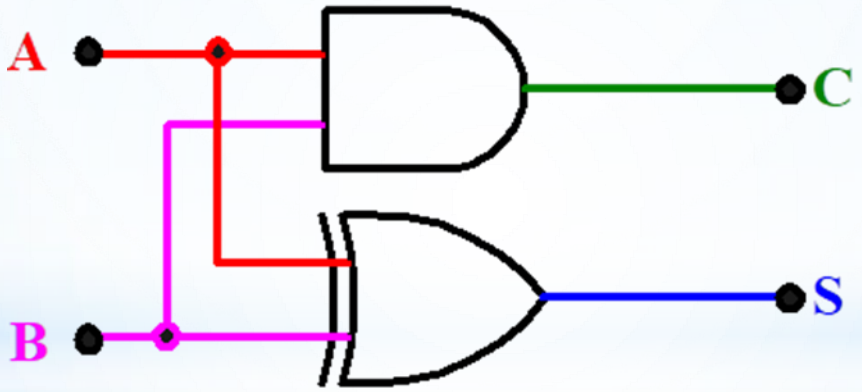
Πίνακας Αληθείας ημιαθροιστή

Ημιαθροιστής (Half adder)

$$C = A \cdot B$$

$$S = \overline{A} \cdot B + A \cdot \overline{B} = (A \oplus B)$$

Λογικές συναρτήσεις ημιαθροιστή

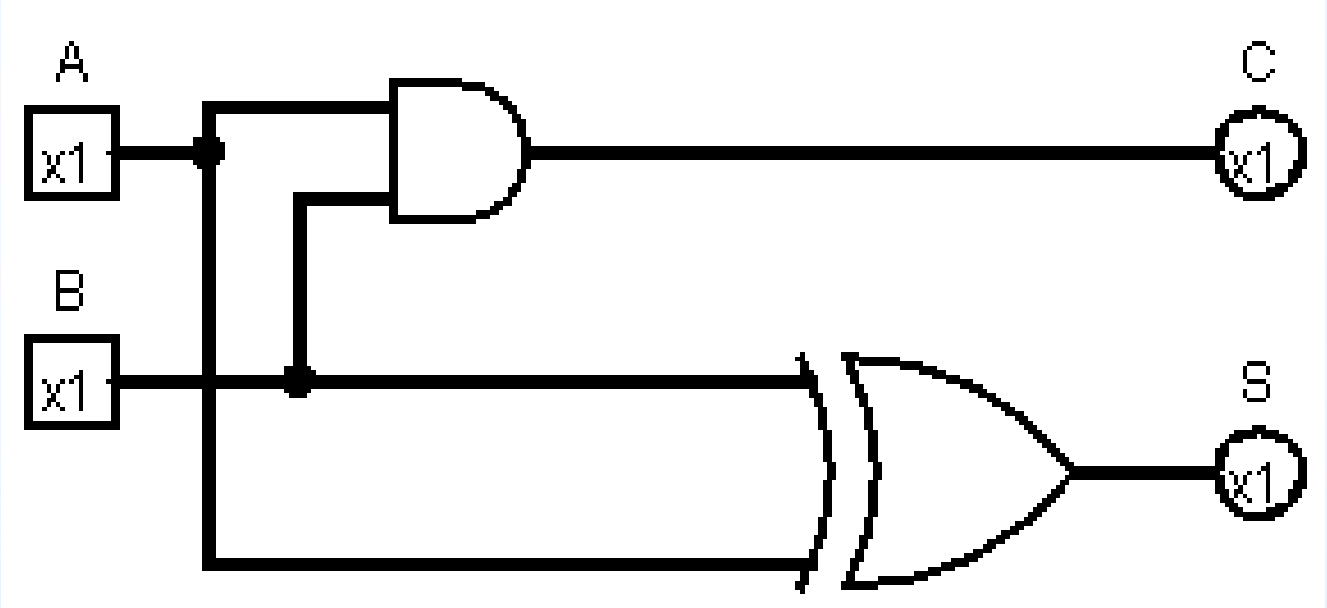


Υλοποίηση ημιαθροιστή

Ημιαθροιστής σε γλώσσα VHDL

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

entity half_adder  is
    port (A : in bit;
          B : in bit;
          S : out   bit;
          C : out  bit
    );
end half_adder;

architecture  dataflow  of half_adder  is
begin
    C <= A and B;
    S <= A xor B;
end;
```

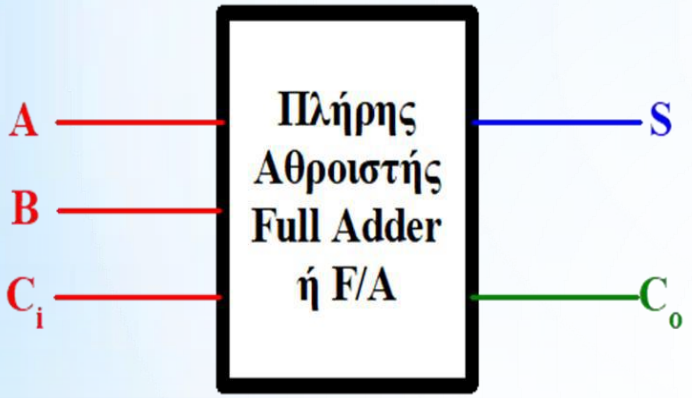
Σχεδίαση Ημιαθροιστή
στο σχεδιαστικό περιβάλλον Logisim



Πρόσθεση δύο δυαδικών ψηφίων και κρατούμενου εισόδου

	Κρατούμενο (Carry)	Άθροισμα (Sum)
(0+0)+0	0	0
(0+0)+1	0	1
(0+1)+0	0	1
(0+1)+1	1	0
(1+1)+0	1	0
(1+1)+1	1	1

Πλήρης αθροιστής (Full adder)



A	B	Ci		Co	S
0	0	0	(0+0+0) =0	0	0
0	0	1	(0+0+1) = 1	0	1
0	1	0	(0+1+0) =1	0	1
0	1	1	(0+1+1) = 2	1	0
1	0	0	(1+0+0) =1	0	1
1	0	1	(1+0+1) = 1	1	0
1	1	0	(1+1+0) =2	1	0
1	1	1	(1+1+1) = 3	1	1

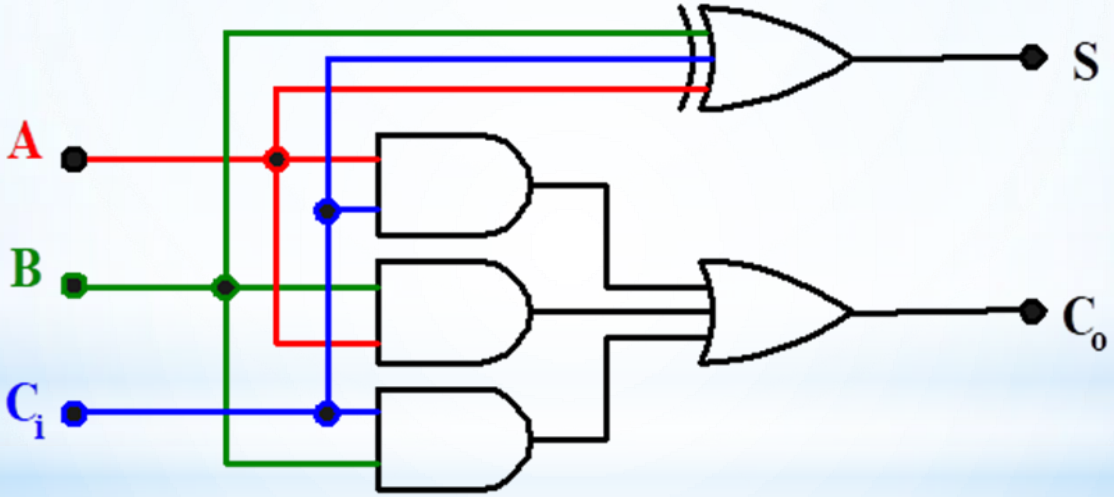
Πίνακας Αληθείας ημιαθροιστή

Πλήρης Αθροιστής (Full adder)

$$Co = A \cdot B + ACi + BCi$$

$$S = \overline{A} \cdot \overline{B} \cdot Ci + \overline{A} \cdot B \cdot \overline{Ci} + A \cdot \overline{B} \cdot \overline{Ci} + A \cdot B \cdot Ci$$

Λογικές συναρτήσεις πλήρους αθροιστή



Υλοποίηση Πλήρους Αθροιστή

Πλήρης Αθροιστής σε γλώσσα VHDL (dataflow)

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

entity full_adder  is
    port (A  : in bit;
          B  : in bit;
          Ci : in bit;
          S  : out  bit;
          Co : out  bit
    );
end full_adder;

architecture  dataflow  of full_adder  is
signal s1, s2, s3, s4 : bit;
begin
    s1 <= A xor B;
    S  <= s1 xor Ci;
    s2 <= A and B;
    s3 <= A and Ci;
    s4 <= B and Ci;
    Co <= s2 or s3 or s4;
end;
```

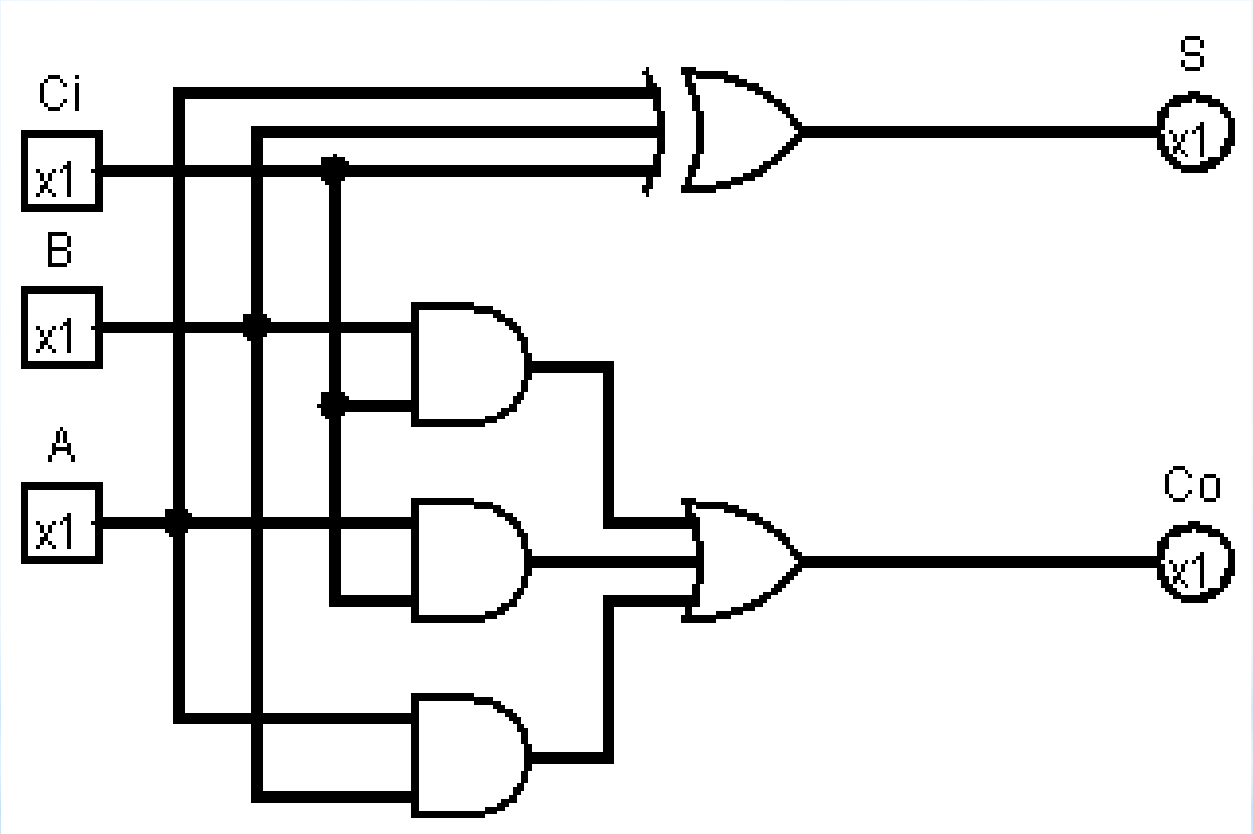
Πλήρης Αθροιστής σε γλώσσα VHDL (procedure)

```
procedure full_adder( signal A,B,Ci: in std_logic;
                      signal S, Co: out std_logic) is

variable v: std_logic_vector(2 downto 0);

begin
    v := A & B & Ci;
    case v is
      when "000"=>  S <='0'; Co<='0';
      when "001"=>  S <='1'; Co<='0';
      when "010"=>  S <='1'; Co<='0';
      when "011"=>  S <='0'; Co<='1';
      when "100"=>  S <='1'; Co<='0';
      when "101"=>  S <='0'; Co<='1';
      when "110"=>  S <='0'; Co<='1';
      when "111"=>  S <='1'; Co<='1';
      when others=> S <='X'; Co<='X';
    end case;
end full_adder;
```

Πλήρης Αθροιστής σε γλώσσα VHDL (function)

```
function  full_adder(A,B,Ci: std_logic) return std_logic_vector is

  variable v : std_logic_vector(2 downto 0);

begin
    v := A & B & Ci;
    case v is
      when "000"=> return "00";
      when "001"=> return "01";
      when "010"=> return "01";
      when "011"=> return "10";
      when "100"=> return "01";
      when "101"=> return "10";
      when "110"=> return "10";
      when "111"=> return "11";
      when others=> return "XX";
    end case;
end function;
```

Σχεδίαση Πλήρους Αθροιστή στο σχεδιαστικό περιβάλλον Logisim

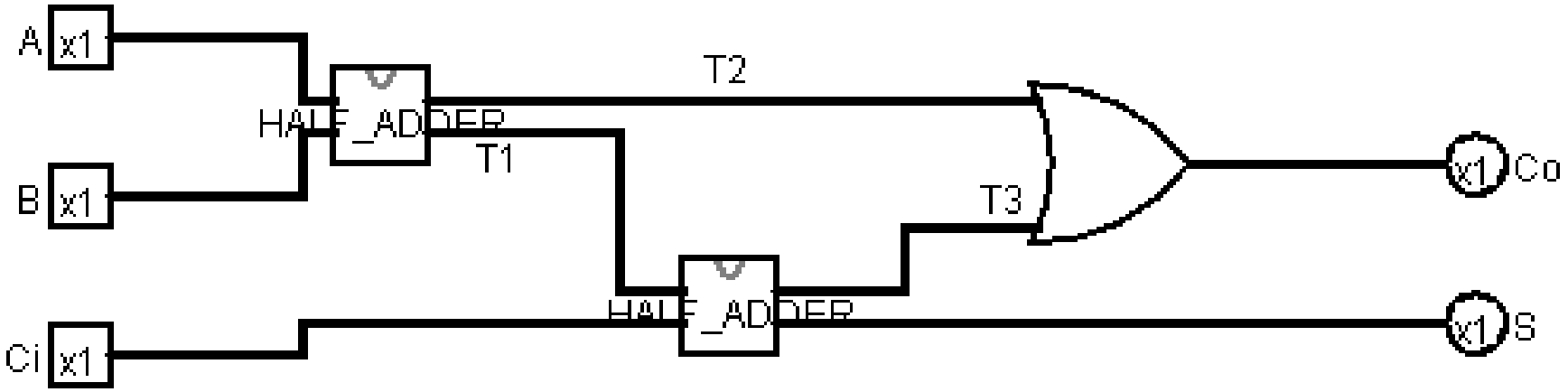


Πλήρης Αθροιστής σε γλώσσα VHDL με χρήση δυο Ημιαθροιστών

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Entity Declaration
entity full_adder is
port (
    A, B, Ci : in bit; -- Inputs
    S, Co: out bit -- Outputs
);
end full_adder;

-- Architecture Definition
architecture structural of full_adder is

-- Component Declaration for Half Adder
component half_adder is
port (
    A, B : in bit;
    S, C : out bit
);
end component;

-- Component Declaration for OR Gate
component or_gate is
port (
    A, B : in bit;
    Y : out bit
);
end component;
```

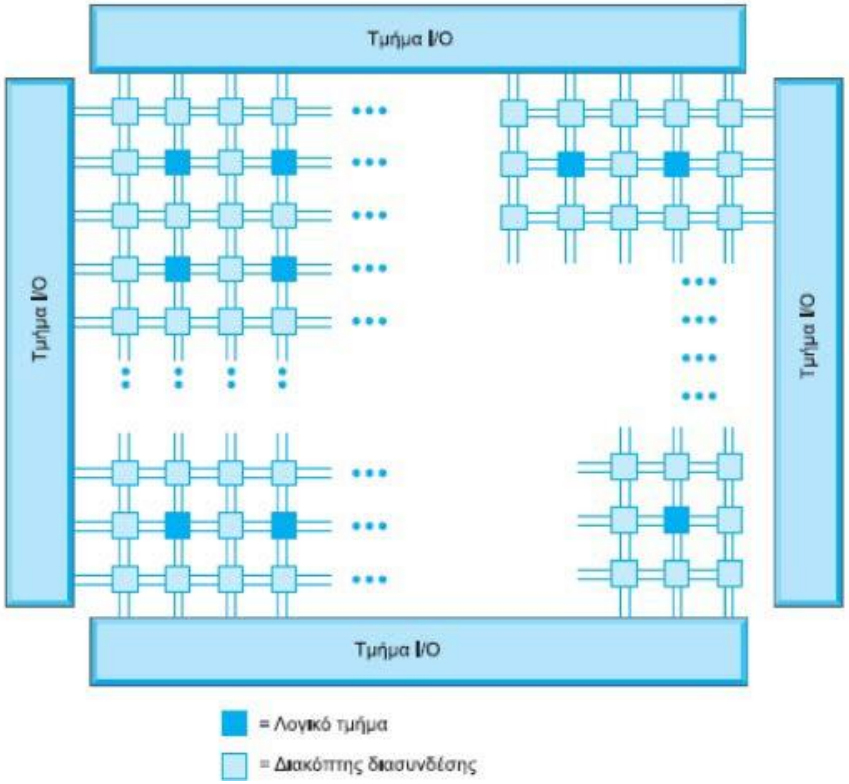
```
-- Internal Signals
signal T1, T2, T3 : bit;

begin

-- Instantiate First Half Adder
half_adder1: half_adder port map (
    A => A, B => B, S => T1, C => T2
);

-- Instantiate Second Half Adder
half_adder2: half_adder port map (
    A => T1, B => Ci, S => S, C => T3
);

-- Instantiate OR Gate for Final Carry-Out
or_gate1: or_gate port map (
    A => T2,
    B => T3,
    Y => Co
);

end structural;
```

Σχεδίαση Πλήρους Αθροιστή με ημιαθροιστές στο Logisim

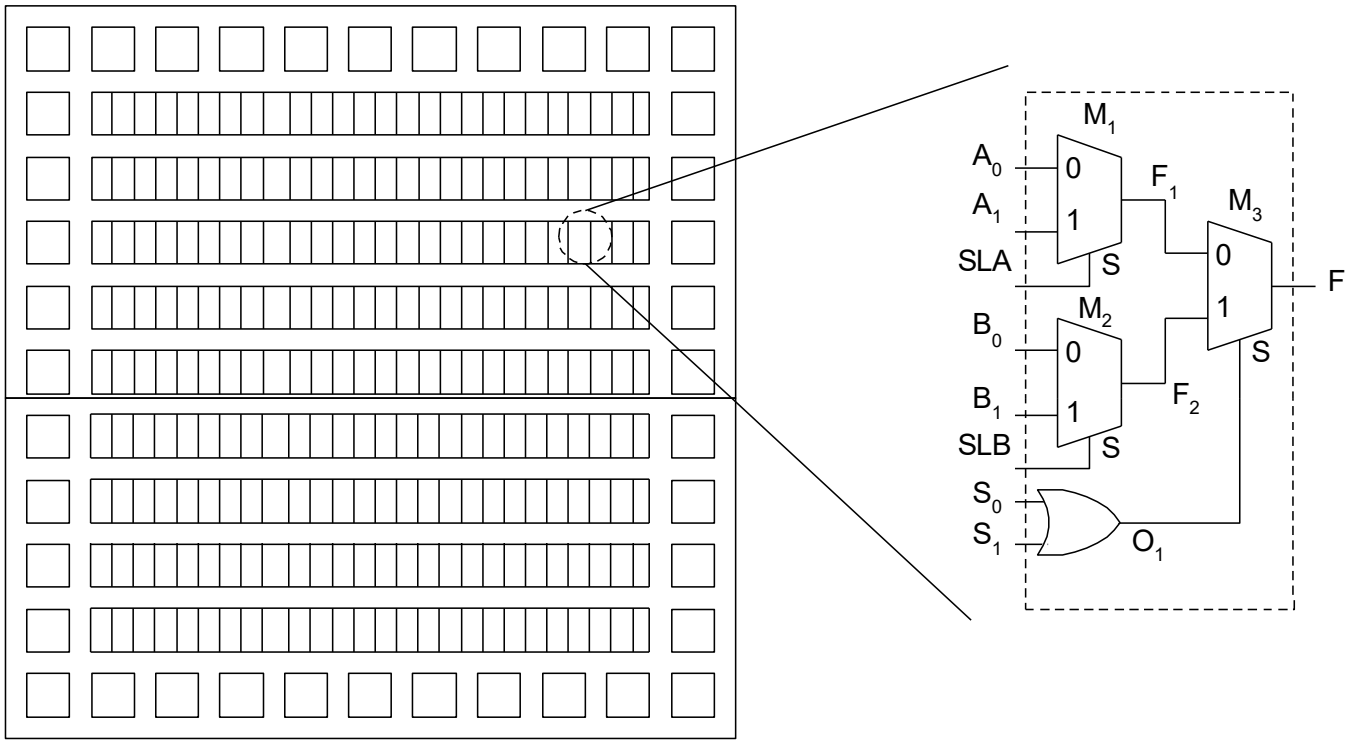


FPGAs

Γενική Δομή FPGAs



Λογικά Κύτταρα με πολυπλέκτες



Λογικό κύτταρο ACTEL,ACT1

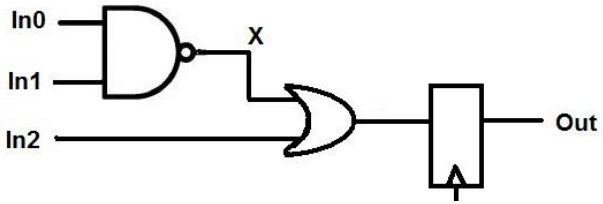
Λογικά Κύτταρα με Πίνακες Αναζήτησης
Παράδειγμα

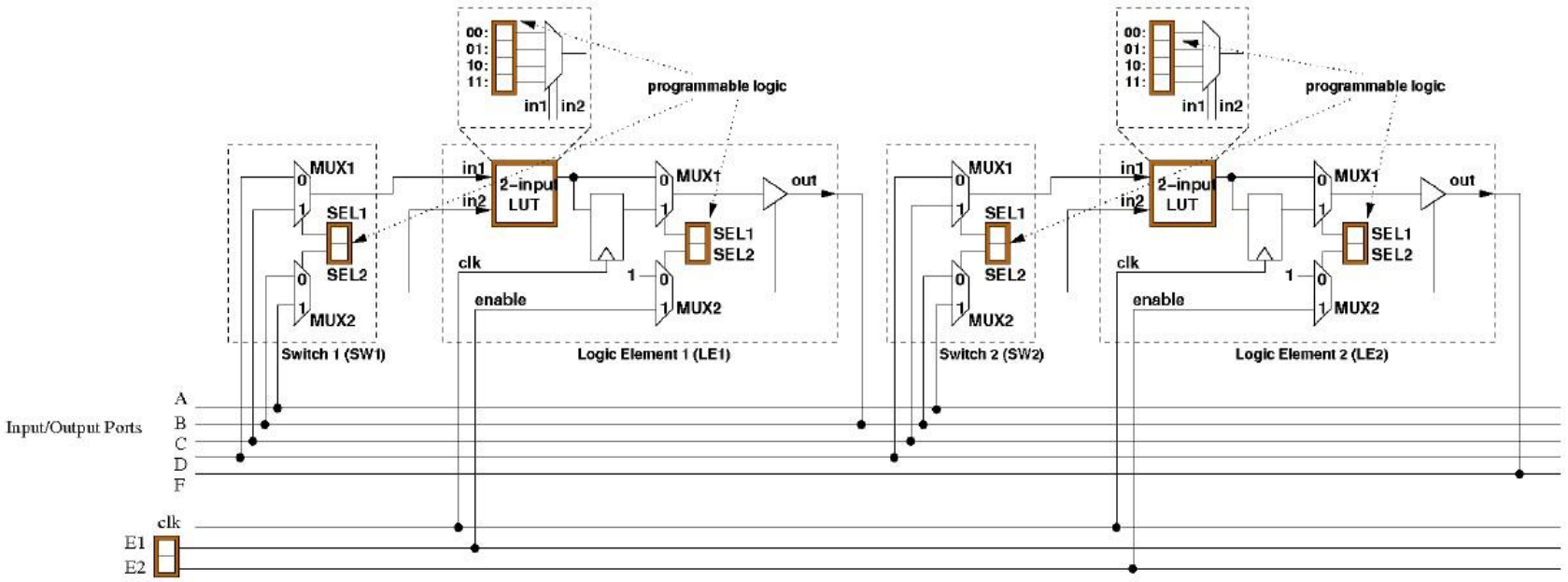
$$F(x, y, z) = \Sigma(0, 1, 3, 5)$$

Σήματα εισόδου			Σήματα εξόδου
x	y	z	F
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

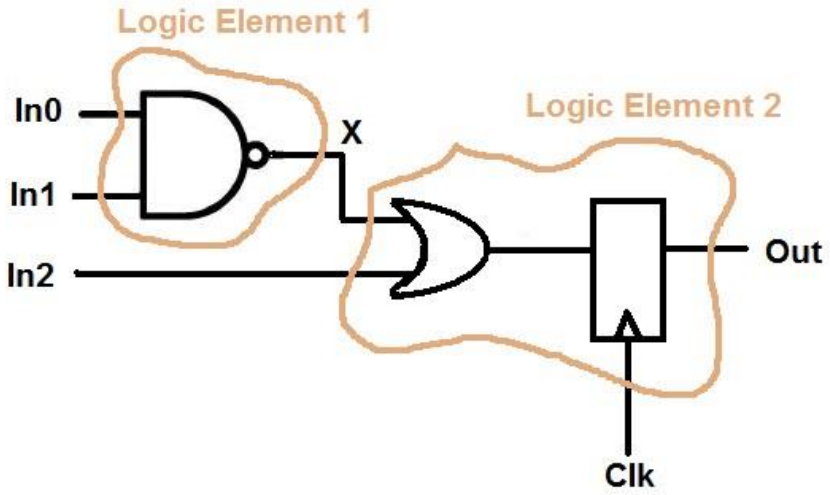
Παράδειγμα αντιστοίχισης κυκλώματος σε FPGA

Να αντιστοιχίσετε τη λειτουργία του κυκλώματος μας στην επαναπρογραμματιζόμενη λογική της FPGA.





Παράδειγμα αντιστοίχισης κυκλώματος σε FPGA

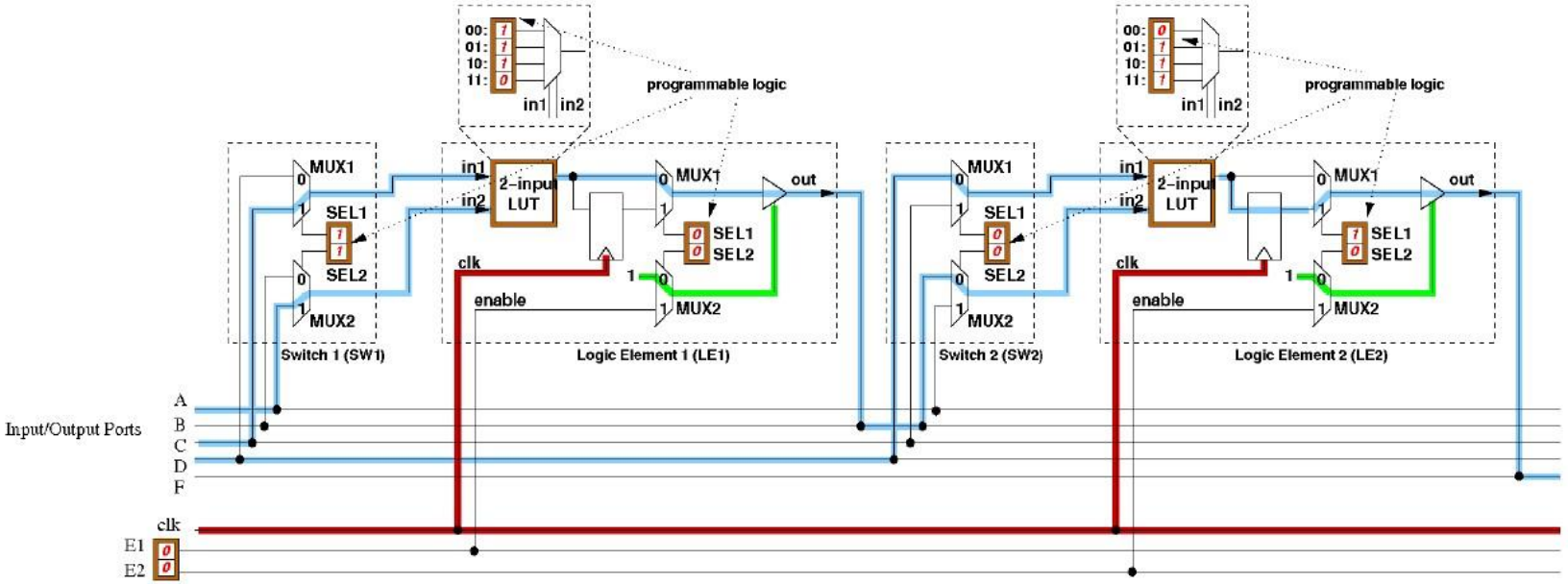


LUTs: Το LUT του logic element 1 προγραμματίζεται ώστε να εκτελεί τη λειτουργία της πύλης NAND. Το LUT του logic element 2 προγραμματίζεται ώστε να εκτελεί τη λειτουργία της πύλης OR.

Σήματα εισόδου: In0 → γραμμή A της FPGA , In1 στη γραμμή C, In2 →γραμμή D και επιλέγεται κατάλληλα σαν είσοδος του Logic Element 2 μέσω του Switch 2.

Σήματα εξόδου: Σήμα εξόδου Out →γραμμή F της FPGA ενώ η ενδιάμεση γραμμή X συνδέεται με τη γραμμή B της FPGA.

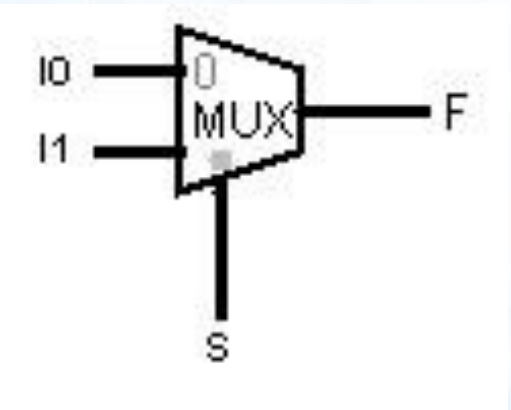
Το SEL1 του MUX 1 των δύο logic elements τίθεται σε διαφορετική τιμή επειδή για το LE1 δε μας ενδιαφέρει η έξοδος του flip-flop ενώ για το LE2 συμβαίνει ακριβώς το αντίθετο.

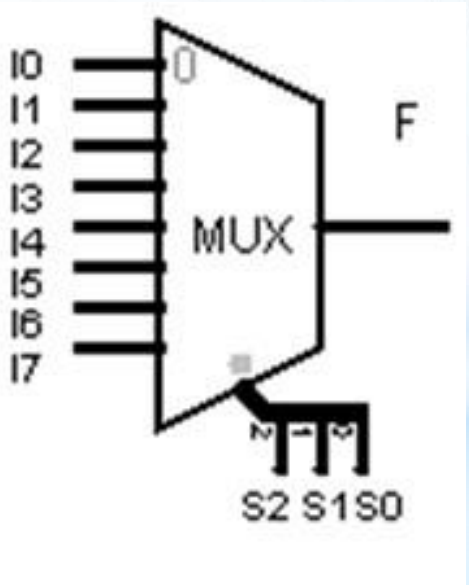

00:
01:
10:
11:
in1
in2
programmable logic
MUX1
SEL1
SEL2
MUX2
2–input LUT
out
clk
enable
Switch 1 (SW1)
Logic Element 1 (LE1)
Switch 2 (SW2)
Logic Element 2 (LE2)
Input/Output Ports
A
B
C
D
F
clk
E1
E2

Πολυπλέκτης (Multiplexer)

Επιλέγει δυαδική πληροφορία που έρχεται σε μία από πολλές γραμμές εισόδου και την μεταφέρει στην μοναδική γραμμή εξόδου

- Η επιλογή της εισόδου γίνεται με ένα σύνολο γραμμών επιλογής.
- Υπάρχουν 2^n είσοδοι και **n** γραμμές επιλογής. Κάθε συνδυασμός των γραμμών επιλογής αντιστοιχεί και σε μία από τις εισόδους.

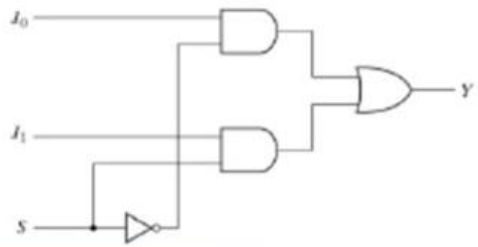




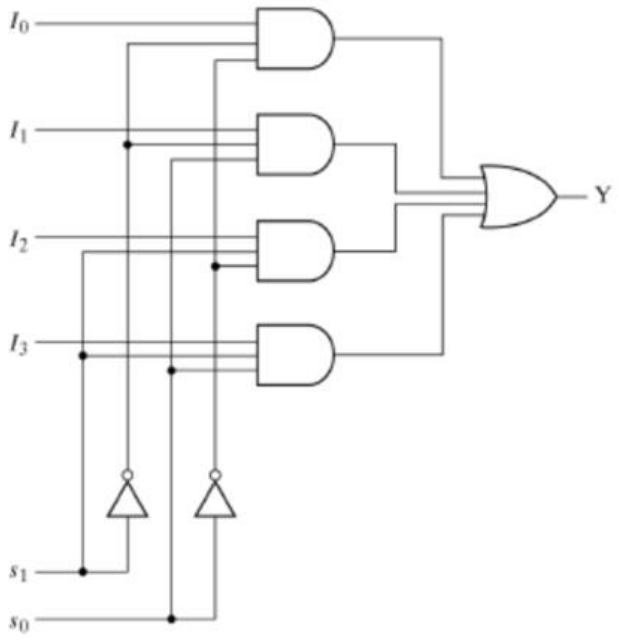
Πολυπλέκτης (Multiplexer)

Λογικά διαγράμματα πολυπλεκτών 2 σε 1 (μια γραμμή επιλογής S) και 4 σε 1 (δύο γραμμές επιλογής S1, S0)

Πολυπλέκτης 2-σε-1



Πολυπλέκτης 4-σε-1



Πολυπλέκτης (Multiplexer)

Υλοποίηση Λογικής Συνάρτησης με ένα Πολυπλέκτη

Μια Συνάρτηση Boole **n** μεταβλητών μπορεί να υλοποιηθεί με ένα Πολυπλέκτη $\mathbf{2^{(n-1)}}$ σε 1.

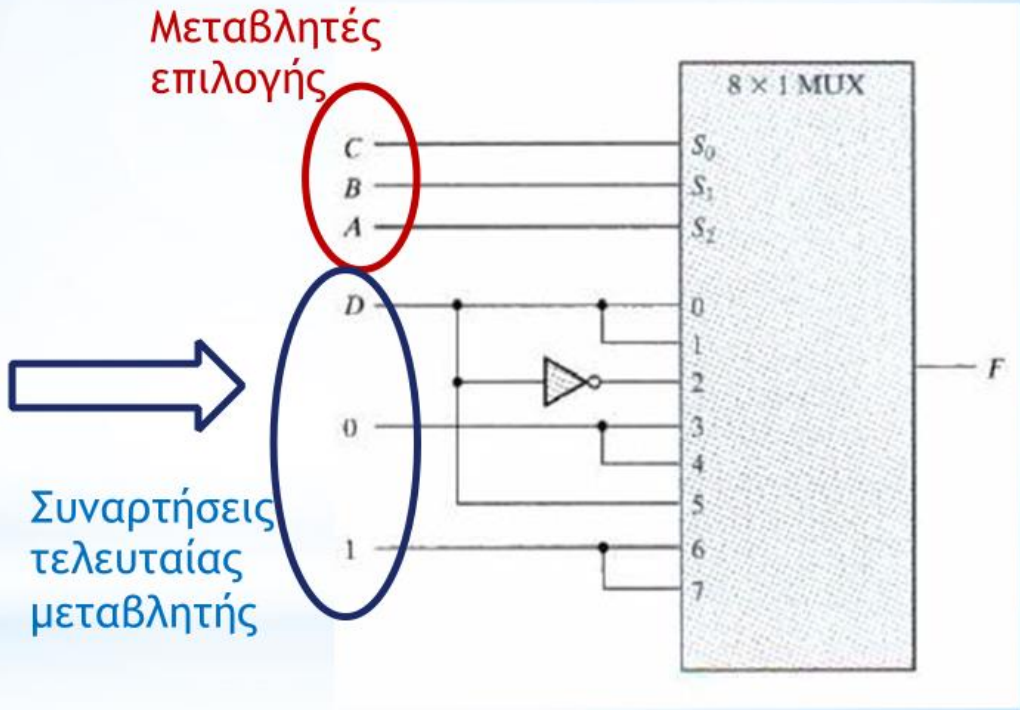
- Δημιουργείται ο Πίνακας Αληθείας της Συνάρτησης.
- Οι **n-1** πρώτες μεταβλητές συνδέονται στις **n-1** γραμμές Επιλογής του Πολυπλέκτη (μεταβλητές επιλογής).
- Για κάθε συνδυασμό των **(n-1)** μεταβλητών επιλογής υπολογίζεται η έξοδος F ως συνάρτηση της τελευταίας μεταβλητής. Προκύπτουν $\mathbf{2^{(n-1)}}$ συναρτήσεις, οι οποίες μπορεί να είναι 0,1, η ίδια η μεταβλητή ή το συμπλήρωμα της.
- Σε κάθε είσοδο του Πολυπλέκτη συνδέεται και μια συνάρτηση που προκύπτει από το προηγούμενο βήμα.

Πολυπλέκτης (Multiplexer)

Παράδειγμα Υλοποίησης Λογικής Συνάρτησης τεσσάρων μεταβλητών με ένα Πολυπλέκτη 8 σε 1

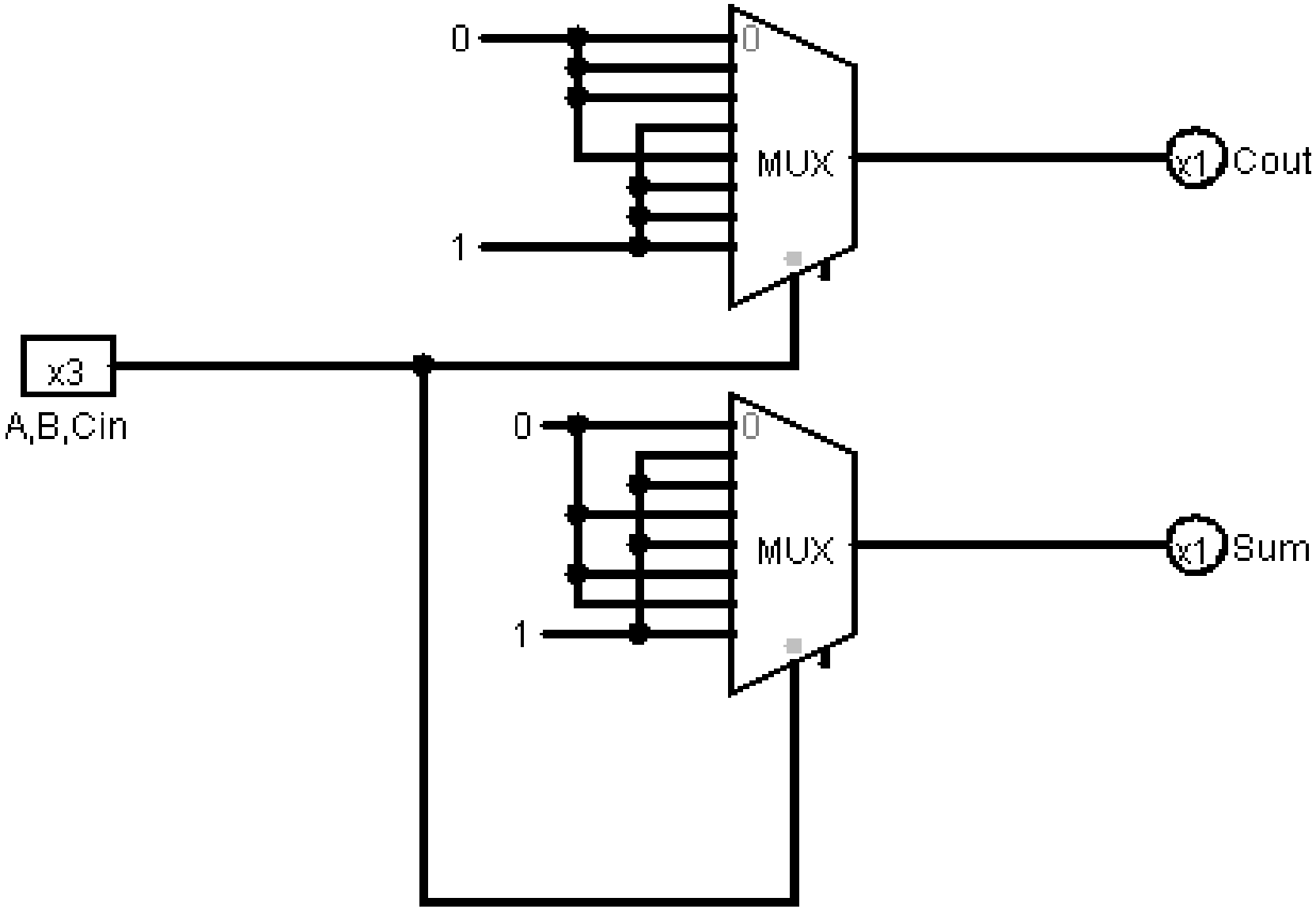
A	B	C	D	F	
0	0	0	0	0	F = D
0	0	0	1	1	
0	0	1	0	0	F = D
0	0	1	1	1	
0	1	0	0	1	F = D'
0	1	0	1	0	
0	1	1	0	0	F = 0
0	1	1	1	0	
1	0	0	0	0	F = 0
1	0	0	1	0	
1	0	1	0	0	F = D
1	0	1	1	1	
1	1	0	0	1	F = 1
1	1	0	1	1	
1	1	1	0	1	F = 1
1	1	1	1	1	

Πίνακας Αληθείας



Σύνδεση Πολυπλέκτη

Υλοποίηση Πλήρους Αθροιστή με Πολυπλέκτες 8Χ1

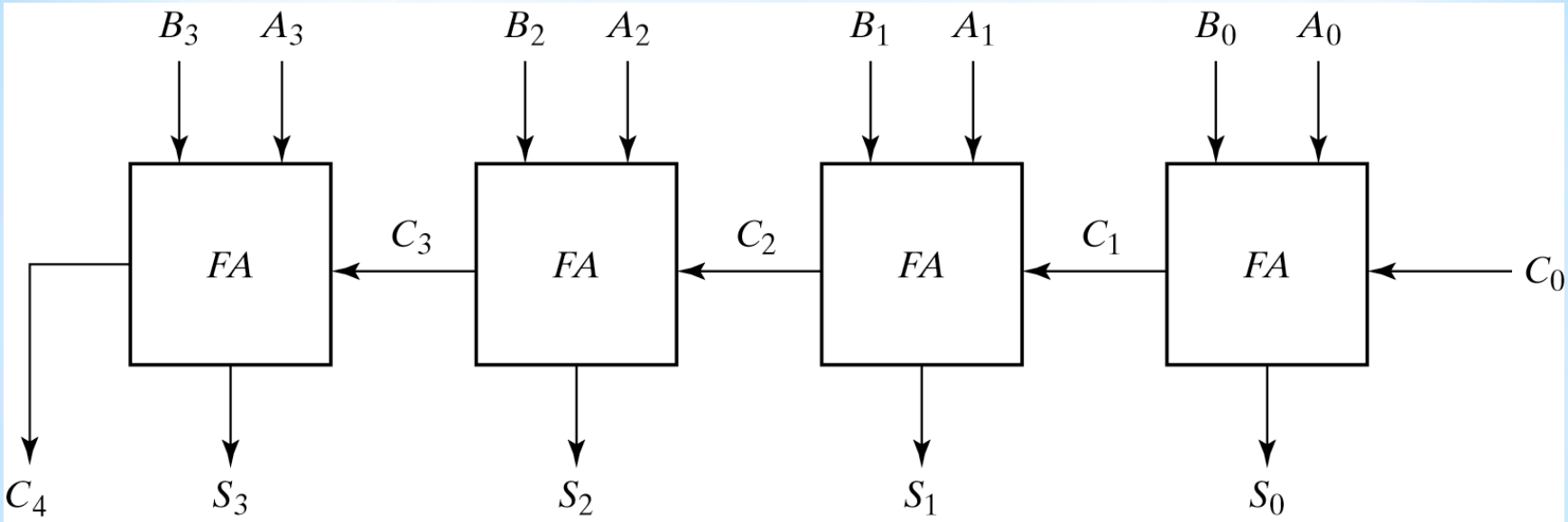


Αθροιστής 4 Ψηφίων

Με την διαδοχική σύνδεση Ν πλήρων αθροιστών μπορούμε να υλοποιήσουμε ένα αθροιστή δύο δυαδικών αριθμών Ν ψηφίων.

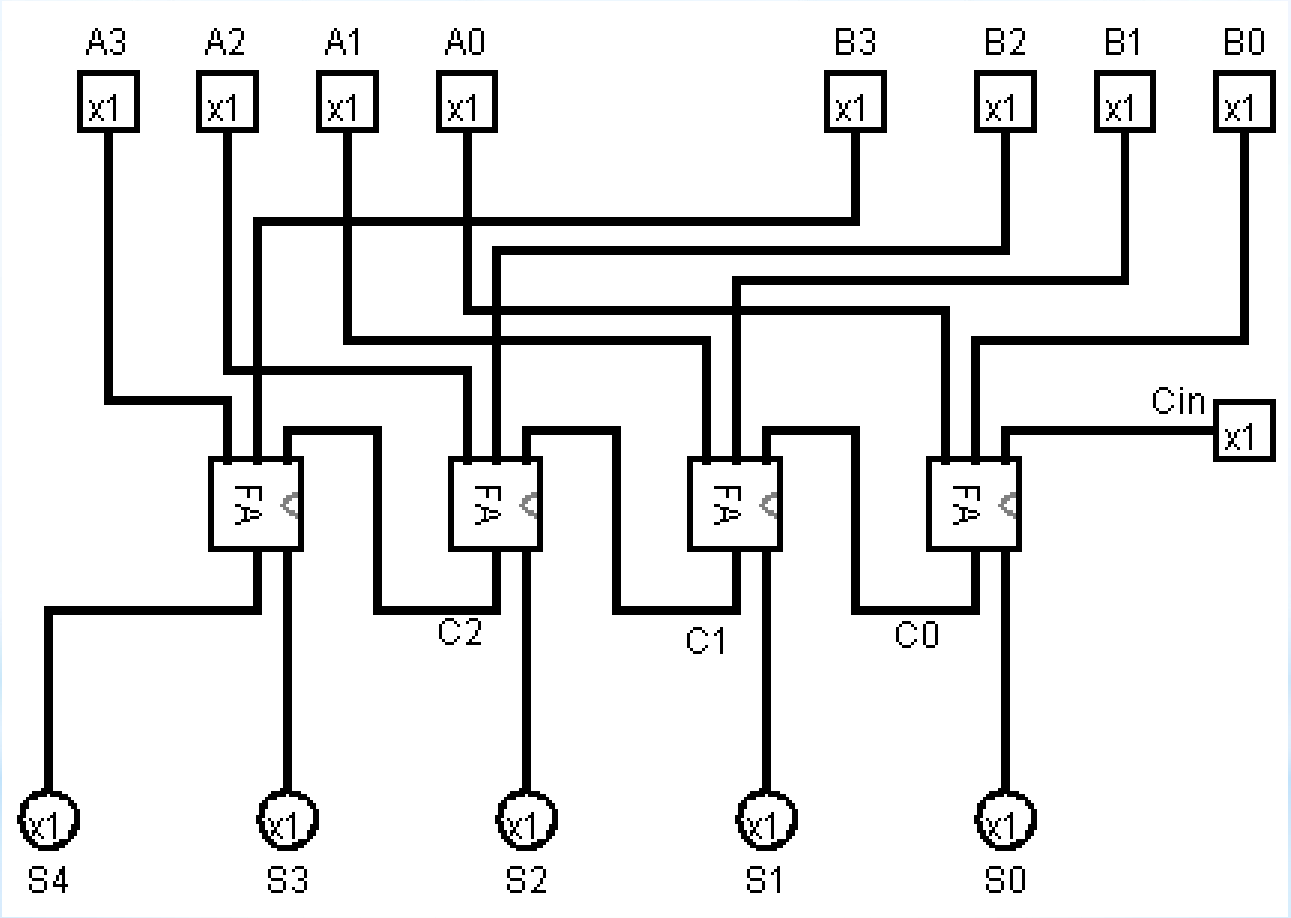
Το άθροισμα μπορεί να χρειάζεται Ν+1 ψηφία για να απεικονιστεί σωστά.

```
Π.Χ.        0 1 1 0   ( 6)
+           1 1 1 0   (14)
          ------------
          1 0 1 0 0   (20)
```



Λογικό Διάγραμμα Αθροιστή 4 ψηφίων με σύνδεση τεσσάρων πλήρων Αθροιστών

Σχεδίαση Αθροιστή 4 ψηφίων με Πλήρεις Αθροιστές στο σχεδιαστικό περιβάλλον Logisim



Δυαδικός Αθροιστής 4 ψηφίων σε γλώσσα VHDL

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity binary_adder_4bit is
port(
     a,b: in bit_vector(3 downto 0);
     cin: in bit;
     sum: out bit_vector(3 downto 0);
     cout: out bit);
end binary_adder_4bit;

architecture struct of binary_adder_4bit is
     signal c : bit_vector(2 downto 0);

component full_adder
port(
     a,b,cin: in bit;
     sum, cout: out bit);
end component;

begin
     FA0: full_adder port map(a(0), b(0), cin,  sum(0), c(0));
     FA1: full_adder port map(a(1), b(1), c(0), sum(1), c(1));
     FA2: full_adder port map(a(2), b(2), c(1), sum(2), c(2));
     FA3: full_adder port map(a(3), b(3), c(2), sum(3), cout);
end struct;
```

Αθροιστής BCD

- Ένας Αθροιστής ο οποίος προσθέτει δύο δεκαδικά ψηφία από το 0 ως το 9 κωδικοποιημένα ως τετραψήφιους δυαδικούς αριθμούς (BCD) και δίνει l το άθροισμα επίσης ως δεκαδικά ψηφία κωδικοποιημένα σε BCD ονομάζεται **Δεκαδικός Αθροιστής ή Αθροιστής BCD.**
- Ο αθροιστής BCD μπορεί να υλοποιηθεί με ένα **Δυαδικό Αθροιστή 4 ψηφίων** και κατάλληλη τροποποίηση της εξόδου του.
- Αν το άθροισμα των δύο δεκαδικών ψηφίων είναι από 0 ως 9 δεν απαιτείται αλλαγή της εξόδου.
- Αν το άθροισμα είναι πάνω από 9 τότε η έξοδος του Δυαδικού Αθροιστή πρέπει να αυξηθεί κατά 6.

Δυαδικό Άθροισμα

Άθροισμα σε κώδικα BCD

C4	S3	S2	S1	S0	HEX	Co	D3	D2	D1	D0	ΔΕΚΑΔΙΚΟ ΑΠΟΤΕΛΕΣΜΑ
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	0	0	0	0	1	1
0	0	0	1	0	2	0	0	0	1	0	2
0	0	0	1	1	3	0	0	0	1	1	3
0	0	1	0	0	4	0	0	1	0	0	4
0	0	1	0	1	5	0	0	1	0	1	5
0	0	1	1	0	6	0	0	1	1	0	6
0	0	1	1	1	7	0	0	1	1	1	7
0	1	0	0	0	8	0	1	0	0	0	8
0	1	0	0	1	9	0	1	0	0	1	9
0	1	0	1	0	A	1	0	0	0	0	10
0	1	0	1	1	B	1	0	0	0	1	11
0	1	1	0	0	C	1	0	0	1	0	12
0	1	1	0	1	D	1	0	0	1	1	13
0	1	1	1	0	E	1	0	1	0	0	14
0	1	1	1	1	F	1	0	1	0	1	15
1	0	0	0	0	10	1	0	1	1	0	16
1	0	0	0	1	11	1	0	1	1	1	17
1	0	0	1	0	12	1	1	0	0	0	18
1	0	0	1	1	13	1	1	0	0	1	19

Αθροιστής BCD

- Συνθήκη για διόρθωση του δυαδικού αθροίσματος

  C4 να είναι 1

  S3 και S2 να είναι 1

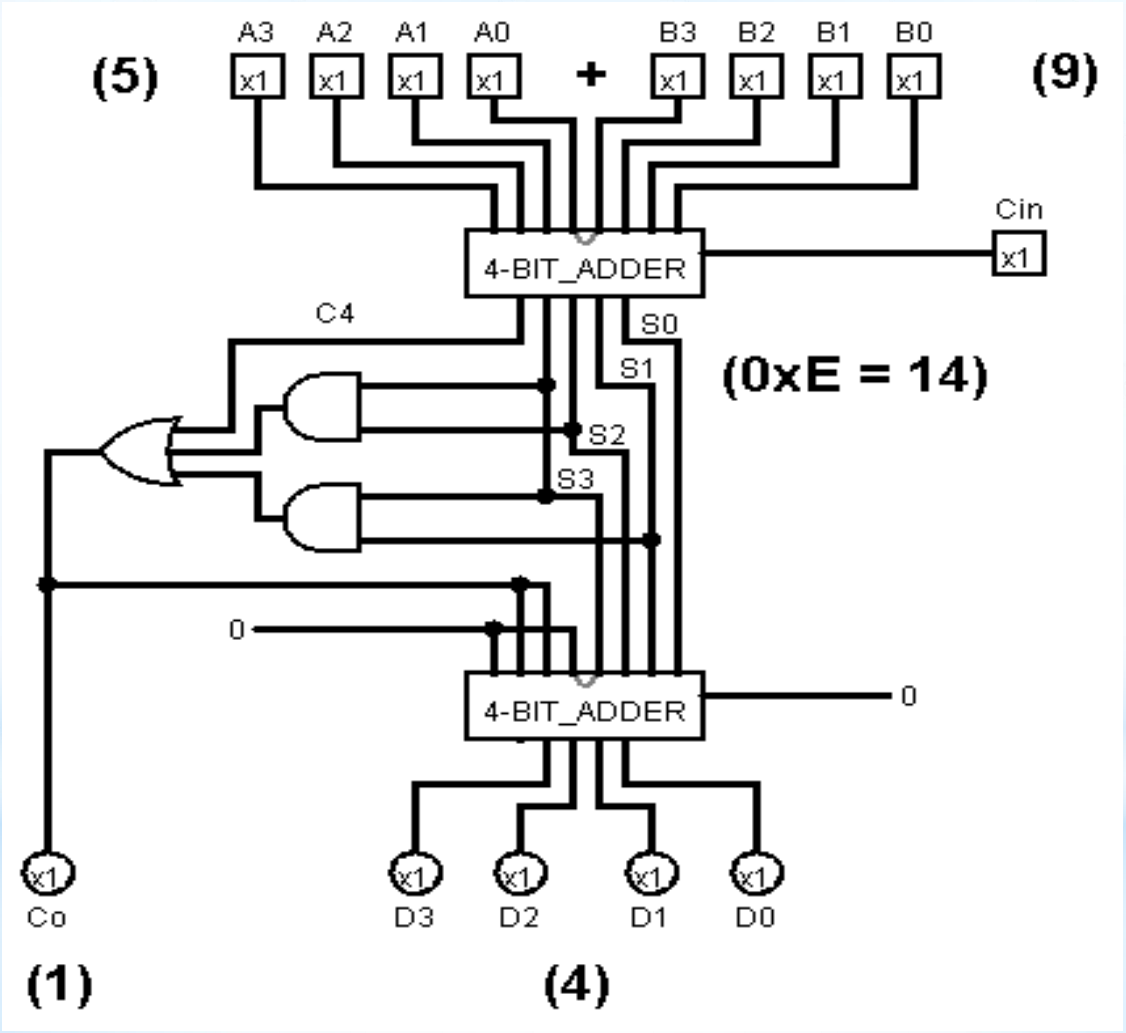
  S3 και S1 να είναι 1

Διόρθωση δυαδικού αθροίσματος

Co = C4 + S3*S2 + S3*S1

(D3, D2,D1,D0) = (S3,S2,S1,S0) συν (0,1,1,0) *Co

Σχεδίαση Αθροιστή BCD
στο σχεδιαστικό περιβάλλον Logisim



Σύγκριση δυο αριθμών

Η σύγκριση δυο αριθμών είναι μια πράξη μεταξύ δυο αριθμών Α,Β που βρίσκει αν ο ένας είναι μεγαλύτερος, μικρότερος ή ίσος με τον άλλο.

Το αποτέλεσμα δίνεται με τρεις δυαδικές μεταβλητές που δείχνουν αν A>B, A<B ή A=B.

Αλγόριθμος σύγκρισης

Θεωρούμε τους δυαδικούς αριθμούς $A=A_3A_2A_1A_0$ και $B=B_3B_2B_1B_0$

Οι δυο αριθμοί είναι ίσοι αν εάν Ai=Bi $\forall i = 0,1,2,3$

Η ισότητα μεταξύ ισοδύναμων ψηφίων αναγνωρίζεται με την πύλη XNOR

$$Xi=AiBi+Ai´Bi´$$

Έχουμε Xi=1 εάν και μόνο εάν Ai=Bi.

Οι δυο αριθμοί είναι ίσοι (A=B) όταν Ai=Bi ∀i Αυτό υπαγορεύει πράξη AND

$$(A=B)=X3\ X2\ X1\ X0$$

Σύγκριση δυο αριθμών (συν.)

Για τον προσδιορισμό της ανισότητας μεταξύ δυο αριθμών εξετάζουμε τα σχετικά μεγέθη των ζευγαριών ψηφίων, αρχίζοντας από την πιο σημαντική θέση. Εάν τα δυο ψηφία είναι ίσα, συγκρίνουμε το επόμενο λιγότερο σημαντικό ζευγάρι ψηφίων κ.ο. Αν τα αντίστοιχα Ai = 1 και Bi = 0 τότε A>B.

Οι διαδοχικές συγκρίσεις εκφράζονται αλγεβρικά ως εξής:

(A>B)= A3B3΄+X3A2B2΄+X3X2A1B1΄+X3X2X1A0B0΄

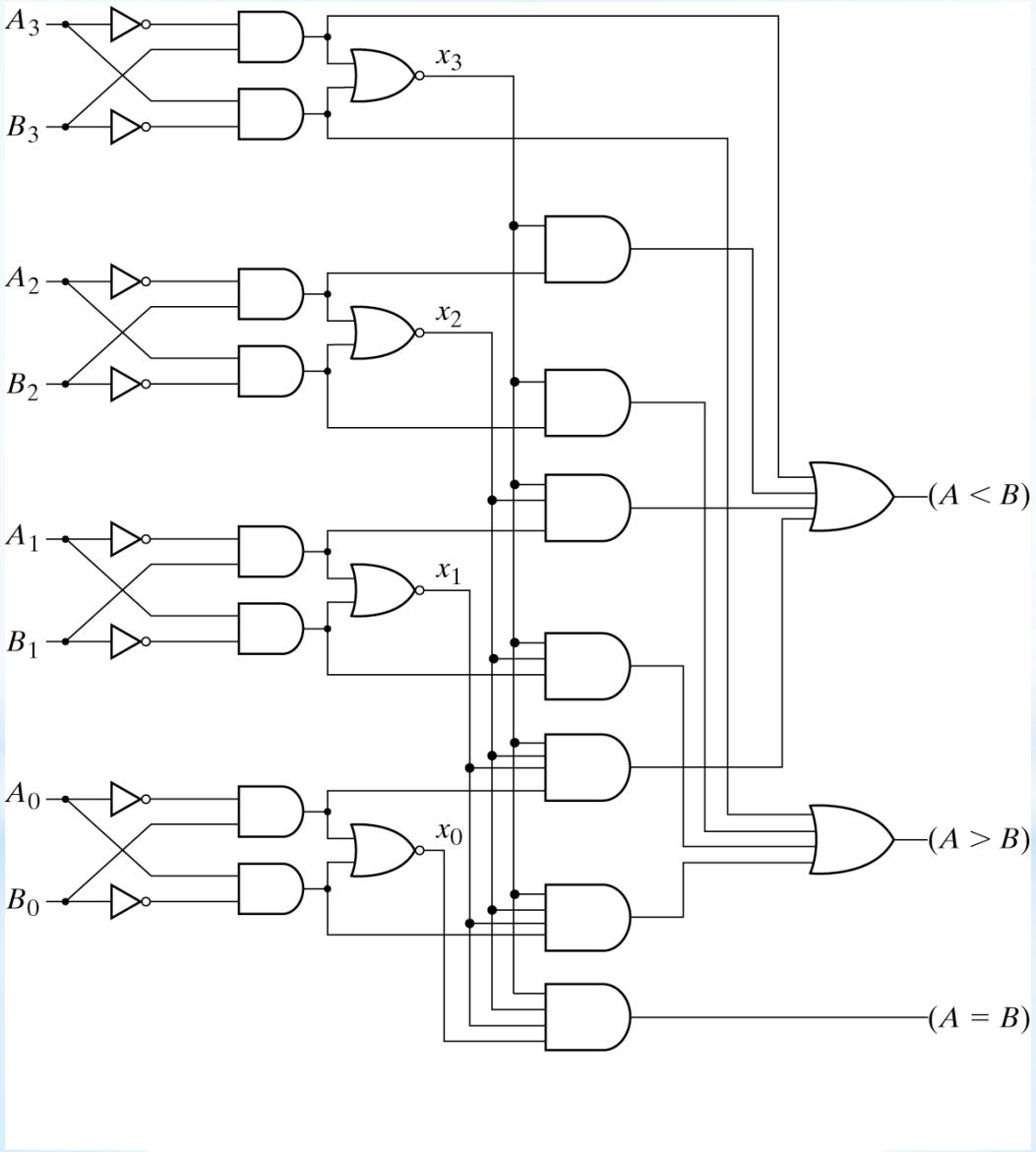
(A<B)= A3΄B3+X3A2΄B2+X3X2A1΄B1+X3X2X1A0΄B0

Η μεταβλητή (A>B) =1 όταν A>B

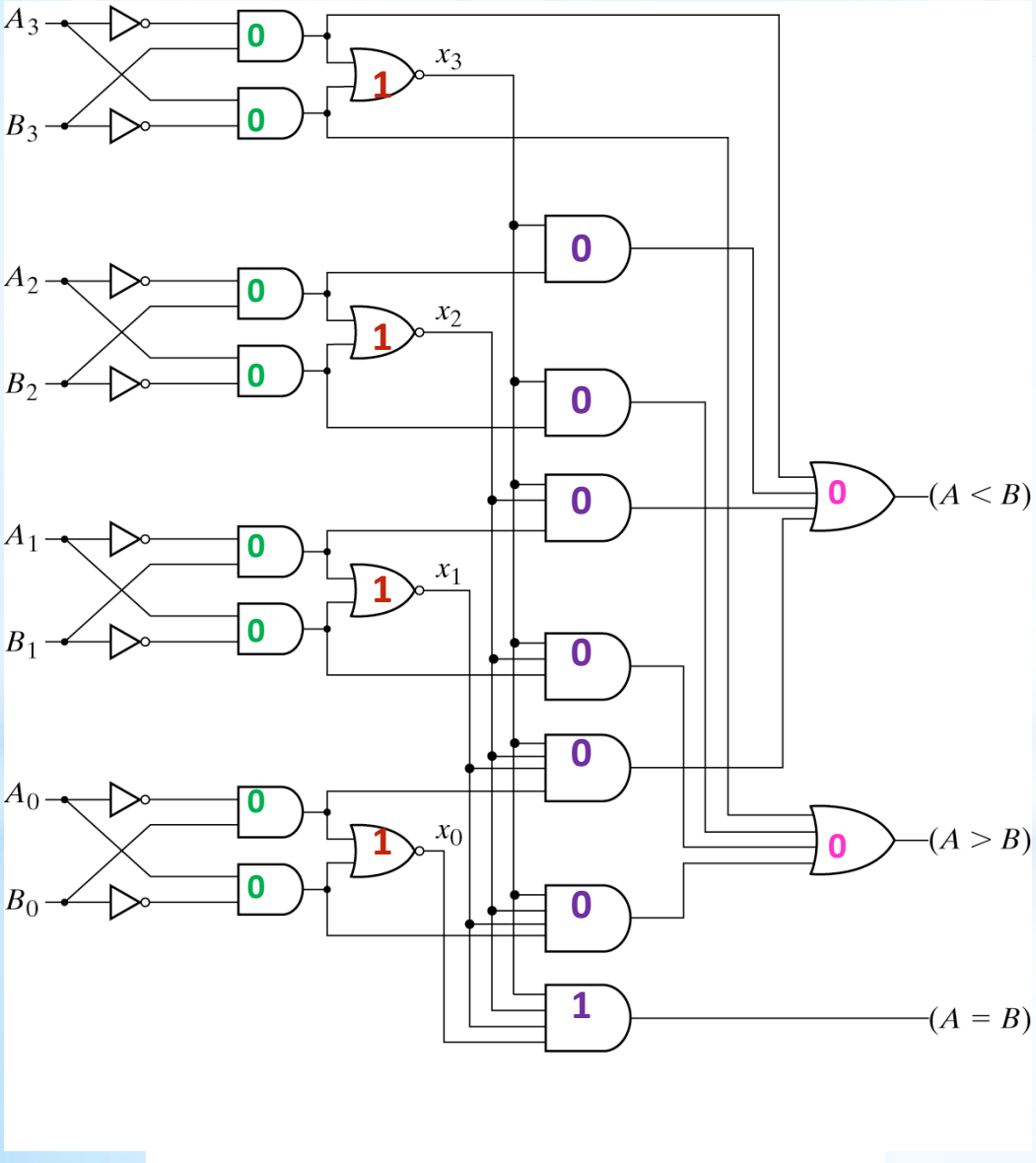
Η μεταβλητή (A<B) =1 όταν A<B

Το κύκλωμα του συγκριτή παρουσιάζει αρκετή κανονικότητα καθώς πραγματοποιεί κάποια αλγοριθμική διαδικασία, που οδηγεί τελικά σε σχετικά εύκολη υλοποίηση

Συγκριτής Μεγέθους 4 ψηφίων



Συγκριτής Μεγέθους 4 ψηφίων



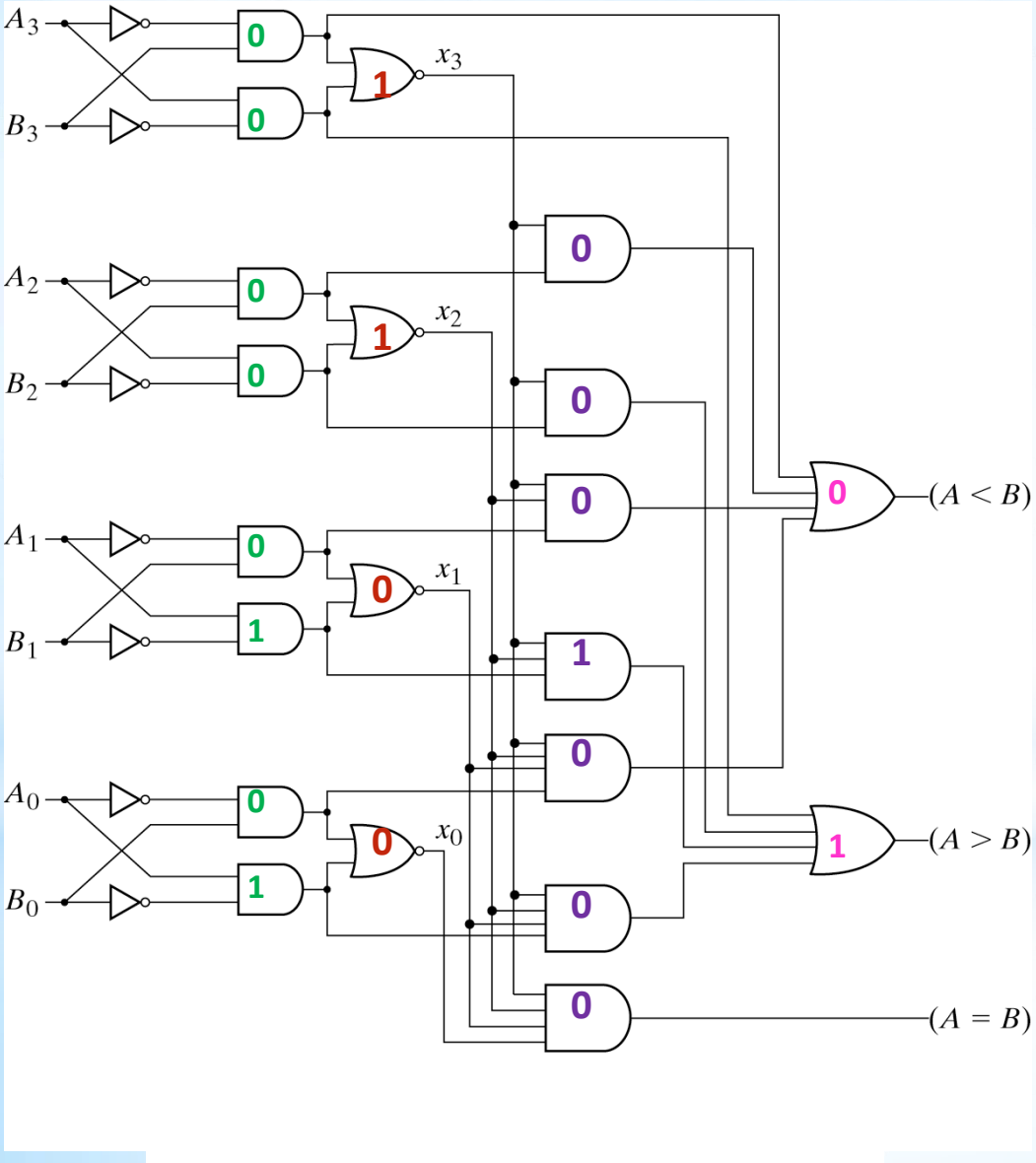
Σύγκριση
1101 με 1101

X3 =1
X2 =1
X1 =1
X0 =1

(A = B) ->1
Ισότητα

Συγκριτής Μεγέθους 4 ψηφίων

1
1
1
1
1
0
1
0

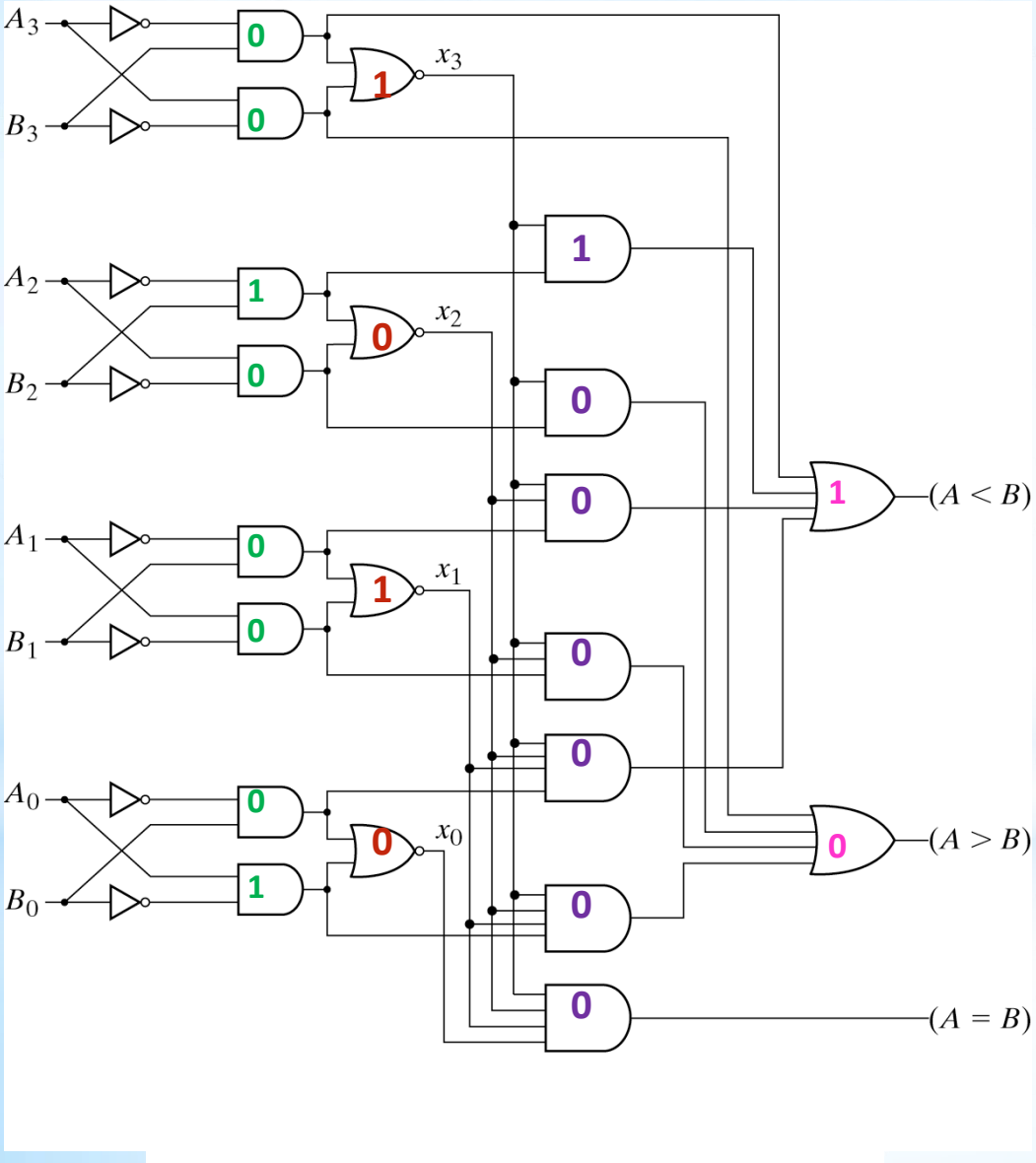


Σύγκριση
1110 με 1101

X3 =1
X2 =1
X1 = 0
X0 =0
Δεν είναι ίσοι

X3*X2*A1*B1’ = 1
(A > B) -> 1

Συγκριτής Μεγέθους 4 ψηφίων



Σύγκριση
1001 με 1100

X3 = 1
X2 = 0
X1 = 1
X0 = 0
Δεν είναι ίσοι

X3*A1'*B1 = 1
(A < B) -> 1

Πολλαπλασιασμός δυαδικών αριθμών

Είναι μια πράξη ακριβώς ίδια με την αντίστοιχη των δεκαδικών και ακόμη ευκολότερη γιατί τα ψηφία πού χρησιμοποιούνται είναι μόνο δύο, το "0" και το "1", οπότε το επιμέρους γινόμενο θα είναι είτε το 0, είτε ο πολλαπλασιαστέος.

Αφού βρεθούν όλα τα επιμέρους γινόμενα, προστίθενται και προκύπτει το τελικό αποτέλεσμα.

Πολλαπλασιασμός δυαδικών αριθμών

B $1001_2 = 9_{10}$

A $\times\ 1011_2 = 11_{10}$

B*A0 1001

B*A1 1001

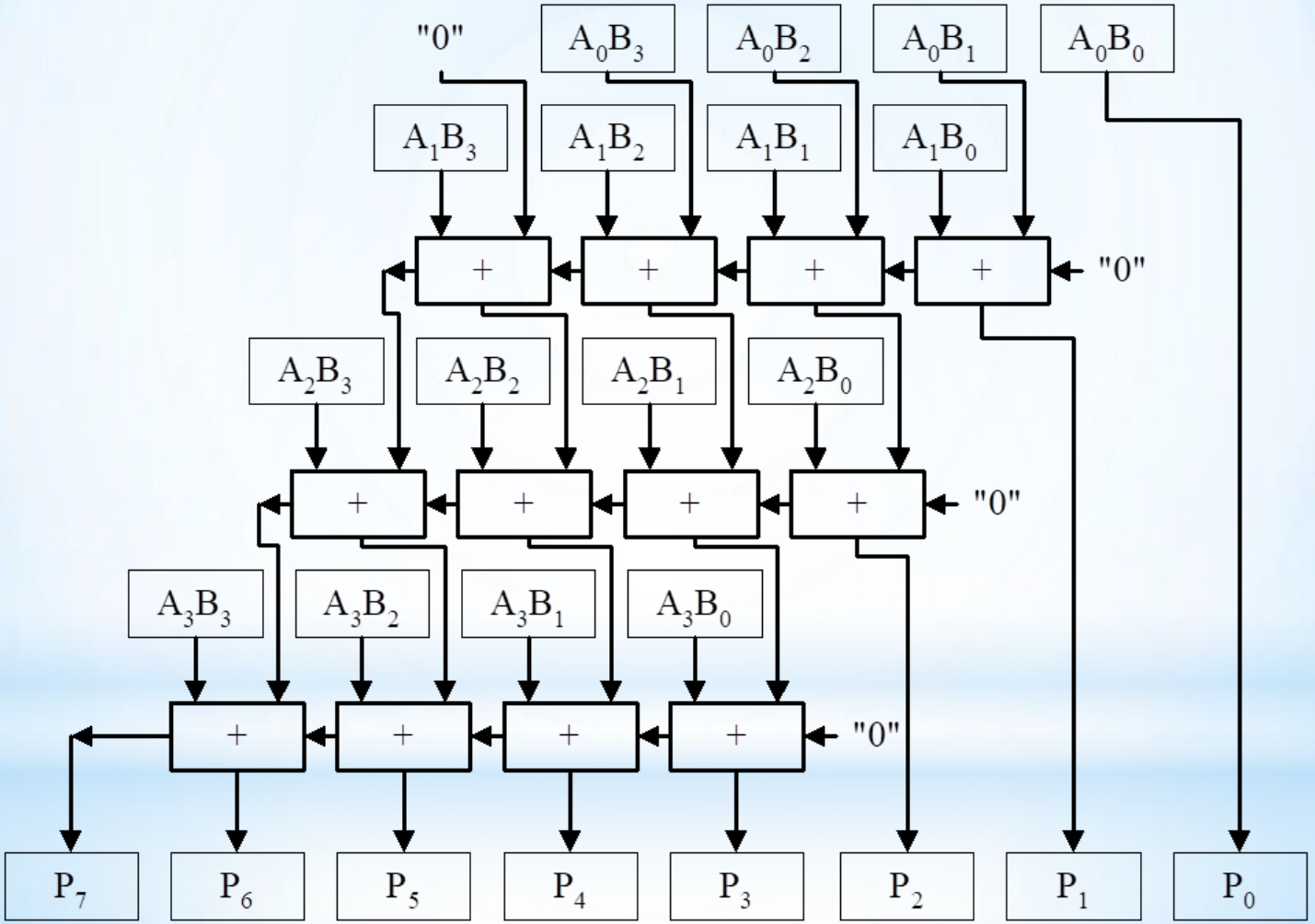
B*A2 0000

B*A3 1001

$1100011_2 = 99_{10}$

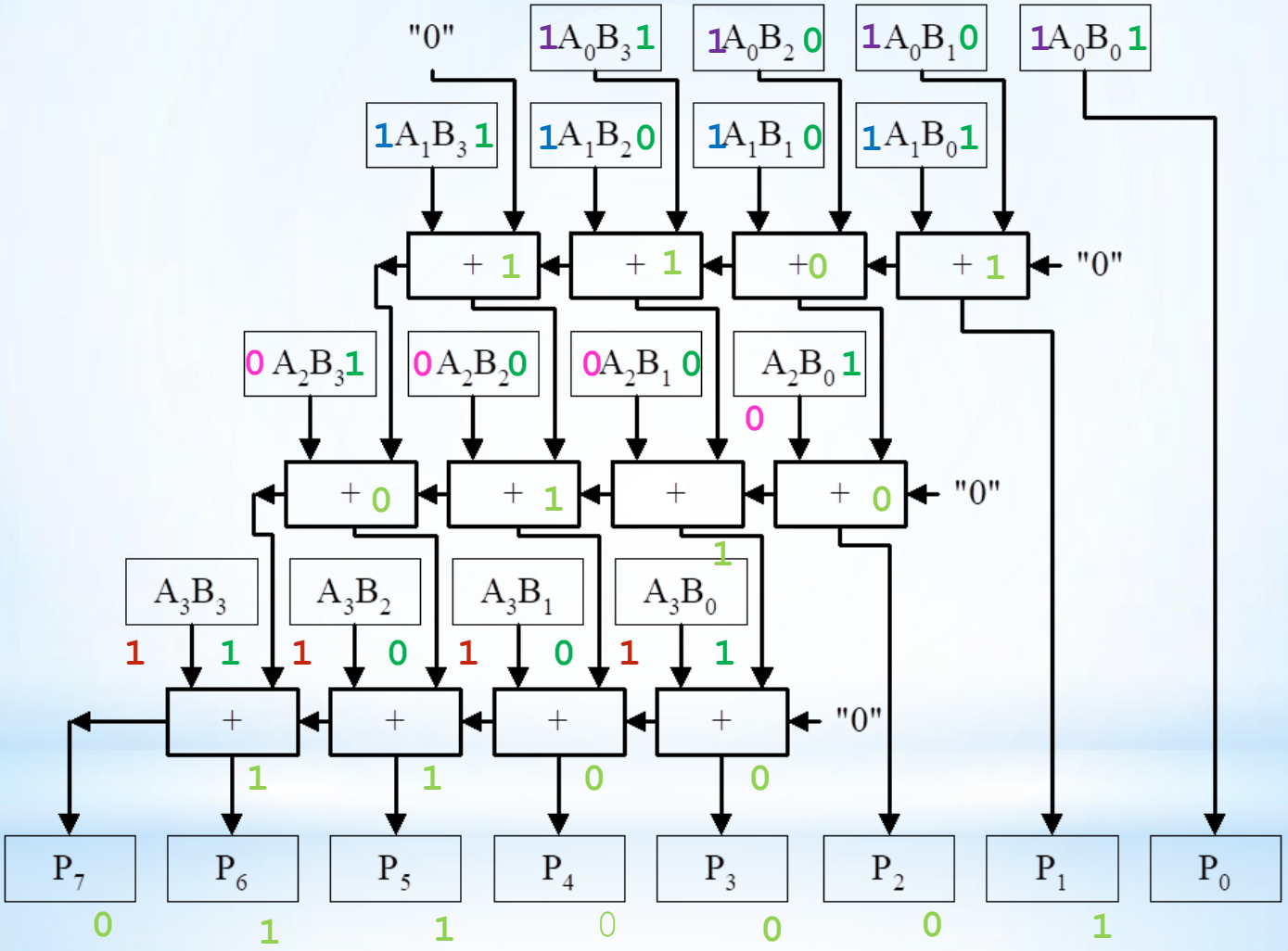
Παράδειγμα πολλαπλασιασμού δύο τετραψήφιων δυαδικών αριθμών

Πολλαπλασιασμός δυαδικών αριθμών



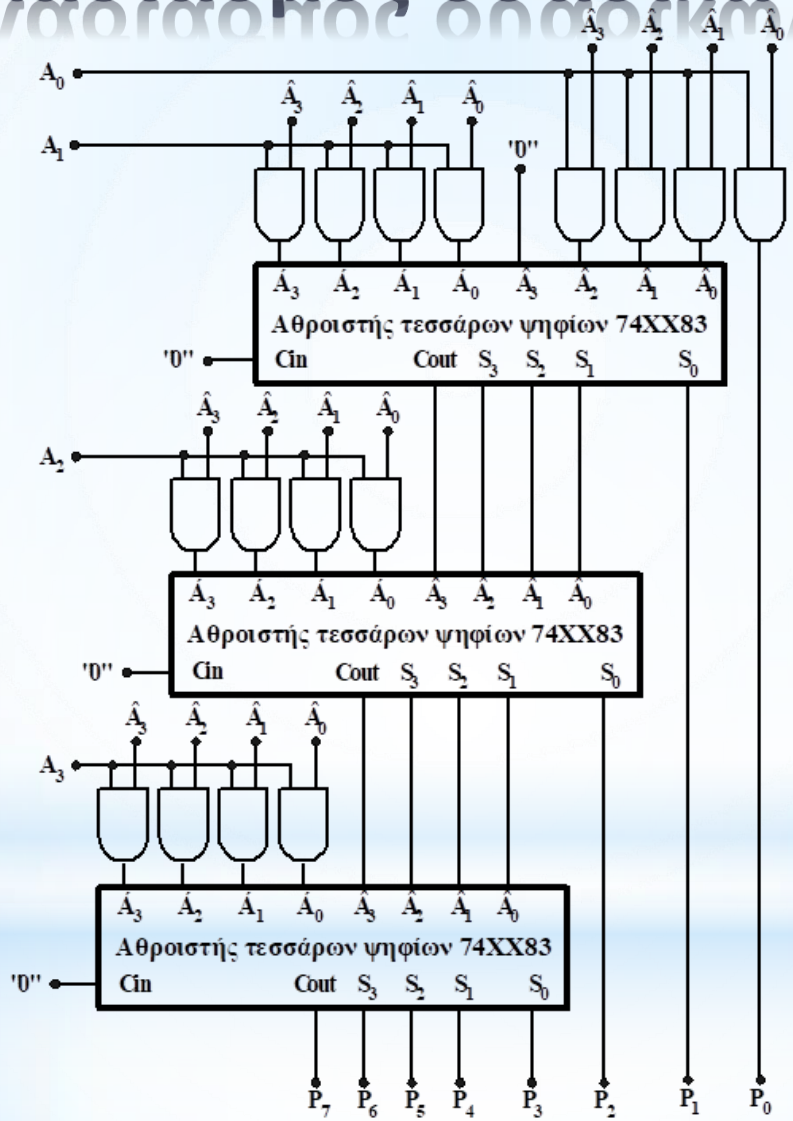
Λογικό Διάγραμμα κυκλώματος
πολλαπλασιασμού δύο τετραψήφιων δυαδικών αριθμών

Πολλαπλασιασμός δυαδικών αριθμών



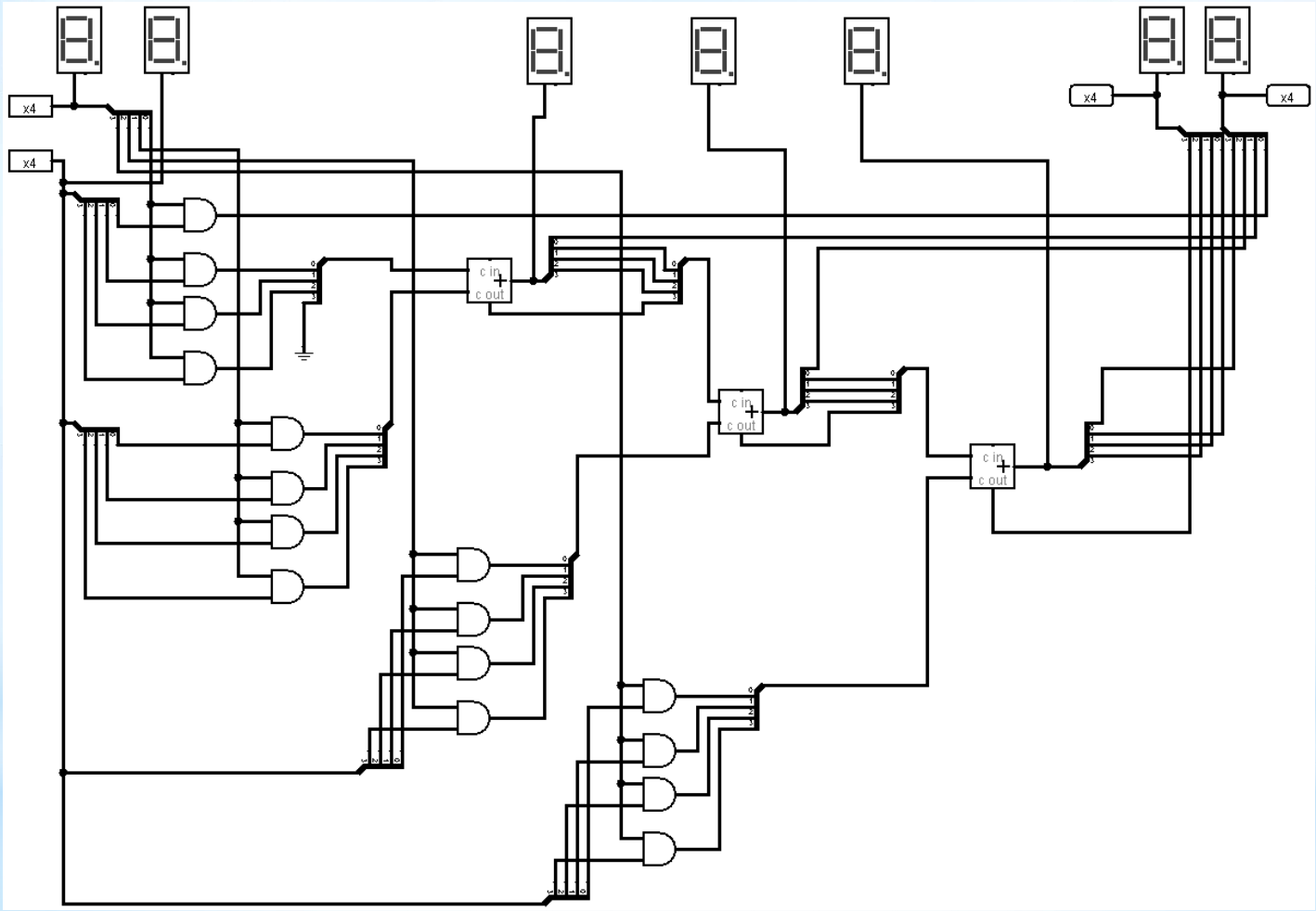
Παράδειγμα εκτέλεσης πολλαπλασιασμού των δυαδικών αριθμών **1001 X 1011**

Πολλαπλασιασμός δυαδικών αριθμών



Σχεδίαση πολλαπλασιαστή δυο τετραψήφιων δυαδικών αριθμών
με τρείς δυαδικούς αθροιστές τεσσάρων ψηφίων

Πολλαπλασιασμός δυαδικών αριθμών



Σχεδίαση πολλαπλασιαστή δυο τετραψήφιων δυαδικών αριθμών στο σχεδιαστικό περιβάλλον Logisim.