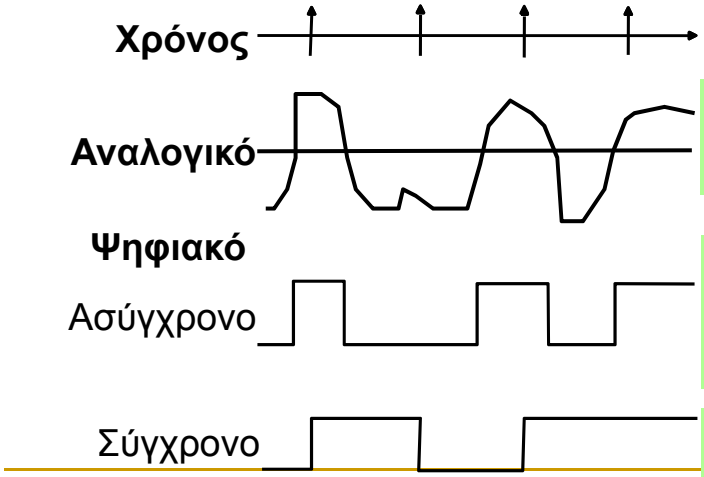
Ενσωματωμένα Ψηφιακά Συστήματα

- Παραδείγματα ΕνσωματωμένωνΣυστημάτων
  - Κυψελοειδής Επικοινωνίες - τηλέφωνα
  - Αυτοκίνητα
  - Video games
  - Φωτοαντιγραφικά
  - Πλυντήρια πιατών
  - TVs
  - Global Positioning Systems

Αναπαράσταση πληροφορίας, Σήματα

- **Στα ψηφιακά συστήματα, οι μεταβλητές παίρνουν διακριτές τιμές.**
- **Οι μεταβλητές δύο επιπέδων ή δυαδικές μεταβλητές είναι οι πιο διαδεδομένες στα ψηφιακά συστήματα.**
- **Οι δυαδικές τιμές αναπαριστάνονται αφηρημένα με:**
  - **τα ψηφία 0 και 1**
  - **τις λέξεις(σύμβολα) False (F) και True (T)**
  - **τις λέξεις(σύμβολα) Low (L) και High (H)**
  - **και τις λέξεις On και Off.**

Παραδείγματα σημάτων στον χρόνο

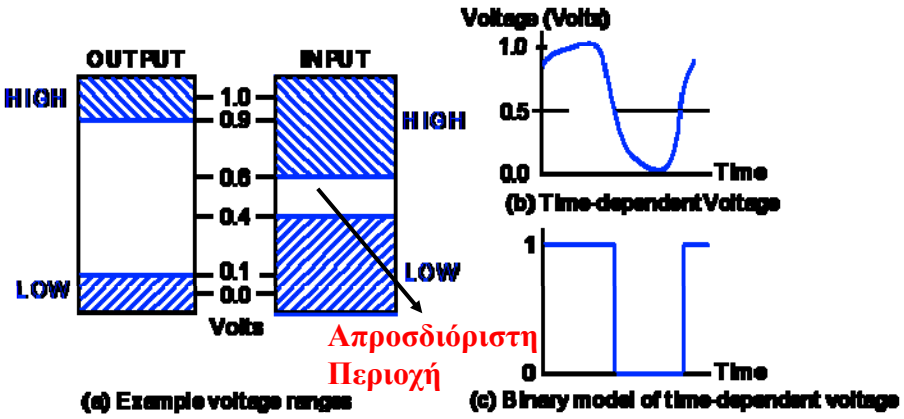


Συνεχές στην τιμή και στο χρόνο

Διακριτό στην τιμή και συνεχές στον χρόνο

Διακριτό στην τιμή και στο χρόνο

Παράδειγμα Σήματος- Φυσική ποσότητα:Τάση



Ψηφιακά Συστήματα - Σήματα

- επεξεργασία *διακριτών στοιχείων* (σύμβολα) πληροφορίας
- αναπαράσταση των διακριτών στοιχείων με φυσικές ποσότητες, τα *σήματα*
- τα σήματα στα σημερινά ψηφιακά συστήματα έχουν συνήθως δύο τιμές, *δυαδικά* (binary)
- η αναπαράσταση των φυσικών σημάτων πραγματικού χρόνου γίνεται με *κβαντισμό* (μετατροπείς αναλογικού-σε-ψηφιακό).

13

Αριθμητικά Συστήματα

- Δεκαδικοί (Decimal)
  - Βάση : **10**, δέκα διαφορετικά ψηφία : (0,1,2,3,4,5,6,7,8,9), LSD και MSD
- Δυαδικοί (Binary)
  - Βάση : **2**, δύο διαφορετικά ψηφία (0, 1), binary digit: 'bit' LSB MSB
- Οκταδικοί (Octal)
  - Βάση : **8**, οκτώ διαφορετικά ψηφία (0,1,2,3,4,5,6,7)
- Δεκαεξαδικοί (Hexadecimal)
  - Βάση : **16**, δεκαέξι διαφορετικά ψηφία : (0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F)

14

Αριθμητικά Συστήματα - Παραδείγματα

	Γενικά	Δεκαδικό	Δυαδικό
Βάση	r	10	2
Ψηφία	0 => r - 1	0 => 9	0 => 1
0	r^0	1	1
1	r^1	10	2
2	r^2	100	4
3	r^3	1000	8
4	r^4	10,000	16
5	r^5	100,000	32
-1	r^{-1}	0.1	0.5
-2	r^{-2}	0.01	0.25
-3	r^{-3}	0.001	0.125
-4	r^{-4}	0.0001	0.0625
-5	r^{-5}	0.00001	0.03125

Ειδικές δυνάμεις του 2

- 2^{10} (1024) - Kilo, συμβολίζεται K
- 2^{20} (1,048,576) - Mega, συμβολίζεται M
- 2^{30} (1,073, 741,824) - Giga, συμβολίζεται G
- 2^{40} (1,099,511,627,776) - Tera, συμβολίζεται T

Αριθμητικά Συστήματα

- Αριθμητικές πράξεις :
  - πρόσθεση, αφαίρεση, πολλαπλασιασμός, διαίρεση
- Μετατροπή βάσης αριθμού
  - μετατροπή από - σε δεκαδικό σύστημα
  - μετατροπή δυαδικό - οκταδικό - δεκαεξαδικό
- Γενικά ένας αριθμός με βάση το r είναι :
  - $a_n \cdot r^n + a_{n-1} \cdot r^{n-1} + ... + a_1 \cdot r + a_0 + a_{-1} \cdot r^{-1} + a_{-2} \cdot r^{-2}$
- Παραδείγματα

17

Δυαδική Πρόσθεση ενός bit με κρατούμενο

- Έστω δύο δυαδικά ψηφία (X,Y), ένα κρατούμενο Z, παίρνουμε τα ακόλουθα, άθροισμα (S) και κρατούμενο (C)

Z	**0**	**0**	**0**	**0**
X	**0**	**0**	**1**	**1**
+ Y	**+ 0**	**+ 1**	**+ 0**	**+ 1**
C S	**0 0**	**0 1**	**0 1**	**1 0**

Z	**1**	**1**	**1**	**1**
X	**0**	**0**	**1**	**1**
+ Y	**+ 0**	**+ 1**	**+ 0**	**+ 1**
C S	**0 1**	**1 0**	**1 0**	**1 1**

Δυαδική Πρόσθεση πολλών bit

- Επέκταση σε δύο αριθμούς με πολλά Bit παραδείγματα:

Κρατούμενα	0	0
	01100	10110
Προσθετέοι	+10001	+10111
Άθροισμα		

- Σημείωση: Το 0 είναι το εξ'ορισμού Κρατούμενο Εισόδου στο λιγότερο σημαντικό bit.

Δυαδική Αφαίρεση ενός bit με δανεικό

- Έστω δύο δυαδικά ψηφία (X,Y), ένα δανεικό Z, παίρνουμε τα ακόλουθα, αποτέλεσμα (S) και δανεικό εξόδου(B)

Z	**0**	**0**	**0**	**0**
X	**0**	**0**	**1**	**1**
- Y	**-0**	**-1**	**-0**	**-1**
BS	**0 0**	**1 1**	**0 1**	**0 0**
Z	**1**	**1**	**1**	**1**
X	**0**	**0**	**1**	**1**
- Y	**-0**	**-1**	**-0**	**-1**
BS	**11**	**1 0**	**0 0**	**1 1**

Δυαδική Αφαίρεση πολλών bit

- Επέκταση σε δύο αριθμούς με πολλά Bit
- παραδείγματα:

Δανεικά	0	0
	10110	10110
Αφαιρετέοι	- 10010	- 10011
Διαφορά		

- Σημείωση: Το 0 είναι το εξ'ορισμού Δανεικό Εισόδου στο λιγότερο σημαντικό bit
- Αν ο αφαιρετέος είναι μεγαλύτερος του μειωτέου τα αντιστρέφουμε και βάζουμε – στο αποτέλεσμα

Δυαδικός Πολλαπλασιασμός

- Ο δυαδικός πολλαπλασιασμός είναι απλός:
- 0*0=0, 0*1=0, 1*0=0, 1*1=1
- Επέκταση πολλαπλασιασμού σε πολλά bit:

	1011
	x 101
	1011
	0000-
	1011--
Γινόμενο:	110111

Μετατροπή βάσης

- Θετικές δυνάμεις του 2

Exponent	Value
0	1
1	2
2	4
3	8
4	16
5	32
6	64
7	128
8	256
9	512
10	1024

Exponent	Value
11	2,048
12	4,096
13	8,192
14	16,384
15	32,768
16	65,536
17	131,072
18	262,144
19	524,288
20	1,048,576
21	2,097,152

Μετατροπή Δυαδικό σε Δεκαδικό

- Για την μετατροπή στο δεκαδικό χρησιμοποιούμε δεκαδική αριθμητική υπολογίζοντας το άθροισμα:
  - Σ (ψηφίο Χ αντίστοιχη δύναμη του 2)
- Παράδειγμα: μετατροπή του 11010_2 σε βάση 10

Μετατροπή Δεκαδικό σε Δυαδικό

- 26_{10} → ???$_2$
- 26_{10} / 2
  - 13 0
  - 6 1
  - 3 0
  - 1 1
  - 0 1

  - 11010_2

Μετατροπή Δεκαδικό σε Δυαδικό

- 0.6875_{10} → ???$_2$
- 0.6875 x 2 → 1 + 0.3750
- 0.3750 x 2 → 0 + 0.750
- 0.7500 x 2 → 1 + 0.5
- 0.5000 x 2 → 1 + 0.0

- Άρα 0.6875_{10} → 0.1011$_2$

Συνήθη Αριθμητικά Συστήματα Διαφόρων Βάσεων

Όνομα	Βάση	Ψηφία
Δυαδικό	2	0,1
Οκταδικό	8	0,1,2,3,4,5,6,7
Δεκαδικό	10	0,1,2,3,4,5,6,7,8,9
Δεκαεξαδικό	16	0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

Αριθμοί σε διαφορετικές βάσεις

Decimal (Base 10)	Binary (Base 2)	Octal (Base 8)	Hexadecimal (Base 16)
00	00000	00	00
01	00001	01	01
02	00010	02	02
03	00011	03	03
04	00100	04	04
05	00101	05	05
06	00110	06	06
07	00111	07	07
08	01000	10	08
09	01001	11	09
10	01010	12	0A
11	01011	13	0B
12	01100	14	0C
13	01101	15	0D
14	01110	16	0E
15	01111	17	0F
16	10000	20	10

Μετατροπή αριθμών μεταξύ συστημάτων με διαφορετική βάση

- Μετατροπή του ακέραιου μέρους
- Μετατροπή του δεκαδικού μέρους
- Συνένωση των δύο αποτελεσμάτων με την υποδιαστολή

Παράδειγμα: μετατροπή του 46.6875_{10} σε Base 2

- Μετατροπή του 46 σε Βάση-2
- Μετατροπή του 0.6875 σε Βάση-2
- Συνένωση των αποτελεσμάτων.

Επαλήθευση της μετατροπής

- Για την μετατροπή πίσω στον αρχικό αριθμό, ακολουθούμε την αντίστροφη διαδικασία.
- Προηγούμενο παράδειγμα: 46.6875_{10}

$101110_2 = 1\cdot32 + 0\cdot16 + 1\cdot8 + 1\cdot4 + 1\cdot2 + 0\cdot1$

$= 32 + 8 + 4 + 2$

$= 46$

$0.1011_2 = 1/2 + 1/8 + 1/16$

$= 0.5000 + 0.1250 + 0.0625$

$= 0.6875$

Οκταδικό σε Δυαδικό – και αντίθετα

- 345_8 → ???
- 011 | 100 | 101 → 011100101_2
- 1101010111_2 → ??? $_8$

Συμπληρώματα : χρησιμοποιούνται για αναπαράσταση αρνητικών αριθμών - αφαίρεση

Γενικά	Βάση : 2	Βάση : 10
συμπλήρωμα ως προς **r-1**	ως προς 1	ως προς 9
συμπλήρωμα ως προς **r**	ως προς 2	ως προς 10

Τυπικοί Ορισμοί :

το συμπλήρωμα ως προς r-1 του αριθμού N : $(r^n - 1) - N$

το συμπλήρωμα ως προς r του αριθμού N : $r^n - N$

Πρακτική Προσέγγιση :

9's compl : αφαίρεση κάθε ψηφίου από το 9
10's compl : 9's compl + 1
1's compl : αντικατάσταση 0 με 1 και 1 με 0
2's compl : 1's compl + 1

33

Αφαίρεση με συμπληρώματα ως προς 2,1

- 23→10111 010111 -23 →10111 101001
- -13 → 1101 110011 13 → 1101 001101
- Αποτέλεσμα 1001010 110110
  01010

- 23→10111 010111 -23 →10111 01000
- -13 → 1101 110010 13 → 1101 1101
- Αποτέλεσμα 1001010 10101
  → 1 01010
  1010

Binary Coded Decimal

- Ο κώδικας BCD έχει βάρη 8,4,2,1.
- Ο πιο απλός κώδικας για δεκαδικά ψηφία, και χρησιμοποιεί τις ίδιες δυνάμεις του 2 όπως ένας δυαδικός αριθμός, αλλά κωδικοποιεί μόνο τις δέκα πρωτες τιμές από το 0 ως το 9.
- Παράδειγμα: 1001 (9) = 1000 (8) + 0001 (1)
- Πόσες "άκυρες" κωδικές λέξεις υπάρχουν?
- Τι είναι οι "άκυρες" κωδικές λέξεις ?

Αριθμητική με Αριθμούς BCD

- Για οποιονδήποτε αριθμό BCD χρησιμοποιούμε δυαδική αριθμητική για πρόσθεση:

8	**1000**	Οκτώ
+5	**+0101**	Συν Πέντε
13	**1101**	Είναι 13 (> 9)

- Το αποτέλεσμα είναι μεγαλύτερο του 9, άρα πρέπει να αναπαριστάνεται με δύο ψηφία.

8	**1000**	Οκτώ
+5	**+0101**	Συν Πέντε
13	**1101**	Είναι 13 (> 9)
	+0110	Άρα προσθέτουμε 6
carry = 1	**0011**	
	0001 | 0011	Τελικό αποτέλεσμα (με δύο ψηφία BCD)

BCD πρόσθεση

- πρόσθεση 2905 BCD με 1897 BCD , με κρατούμενα και διορθώσεις.

				0
	0001	1000	1001	0111
+	0010	1001	0000	0101
	____	____	____	____

Μετατροπή ή Κωδικοποίηση

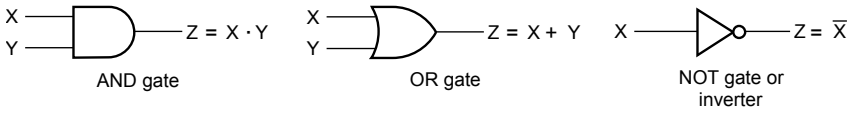
- Προσοχή ΜΗΝ συγχέετε την μετατροπή ενός δεκαδικού αριθμού σε έναν δυαδικό με την κωδικοποίηση ενός δεκαδικού αριθμού χρησιμοποιώντας έναν ΔΥΑΔΙΚΟ ΚΩΔΙΚΑ.
- $13_{10} = 1101_2$ (Αυτό είναι **μετατροπή**)
- $13 \Leftrightarrow 0001|0011$ (Αυτό είναι **κωδικοποίηση**)

Δυαδικοί Αριθμοί και Δυαδική Κωδικοποίηση

- Αναπαράσταση
  - Οτιδήποτε δεδομένο μπορεί να αντιστοιχιστεί σε οποιοδήποτε δυαδικό συνδυασμό (κωδική λέξη) αρκεί να είναι μοναδικά κωδικοποιημένο
- Τύποι πληροφορίας
  - Αριθμητικοί
  - Μη-αριθμητικοί

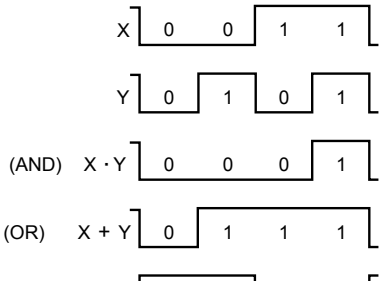
Σύμβολα Λογικών Πυλών - Συμπεριφορά

- Ειδικά σύμβολα για αναπαράσταση λογικών πυλών:



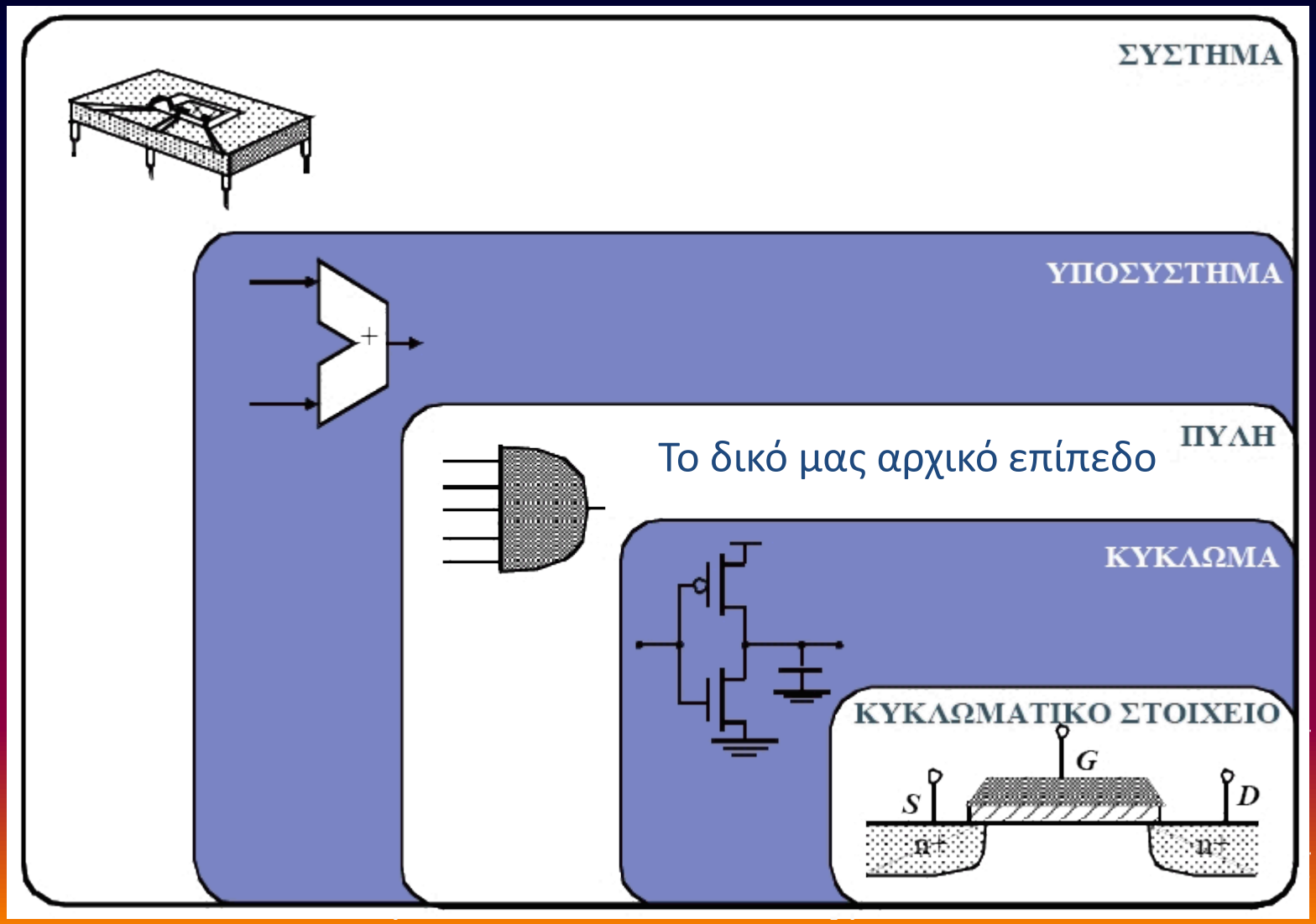
(a) Graphic symbols

- Συμπεριφορά αναπαράσταση στο χρόνο, κυματομορφές - waveform :

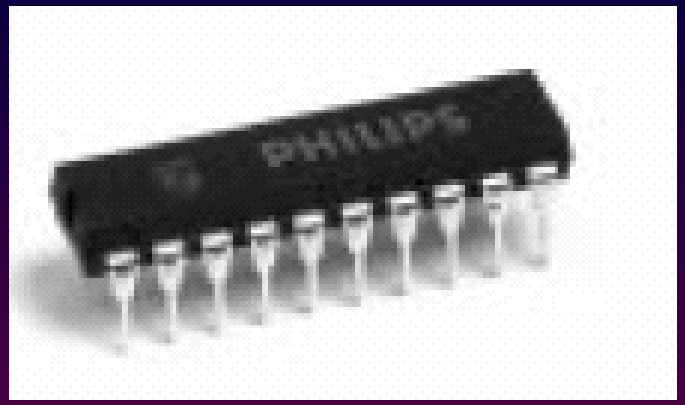


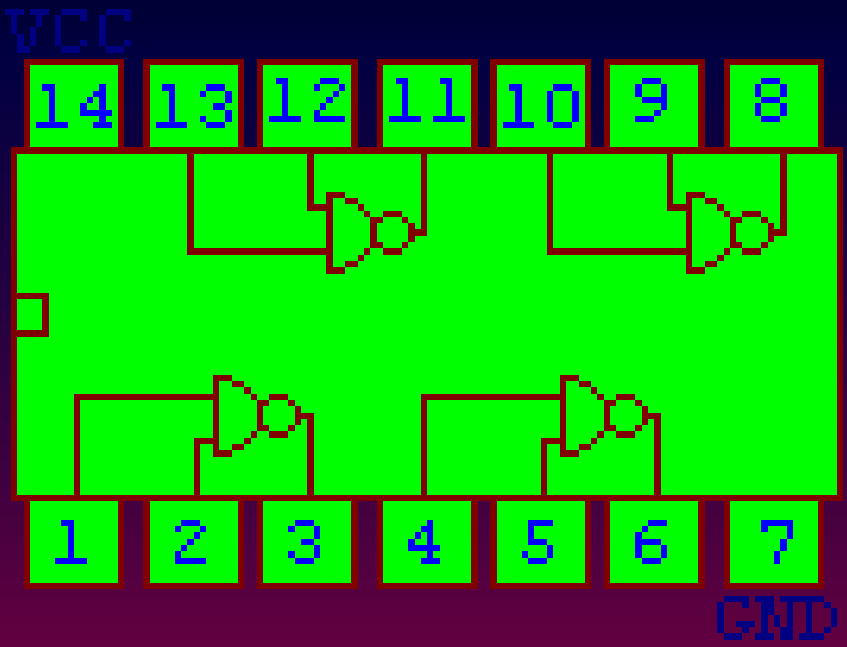
(b) Timing diagram

Οι πολλές μορφές αφαίρεσης και σχεδίασης ενός ψηφιακού κυκλώματος

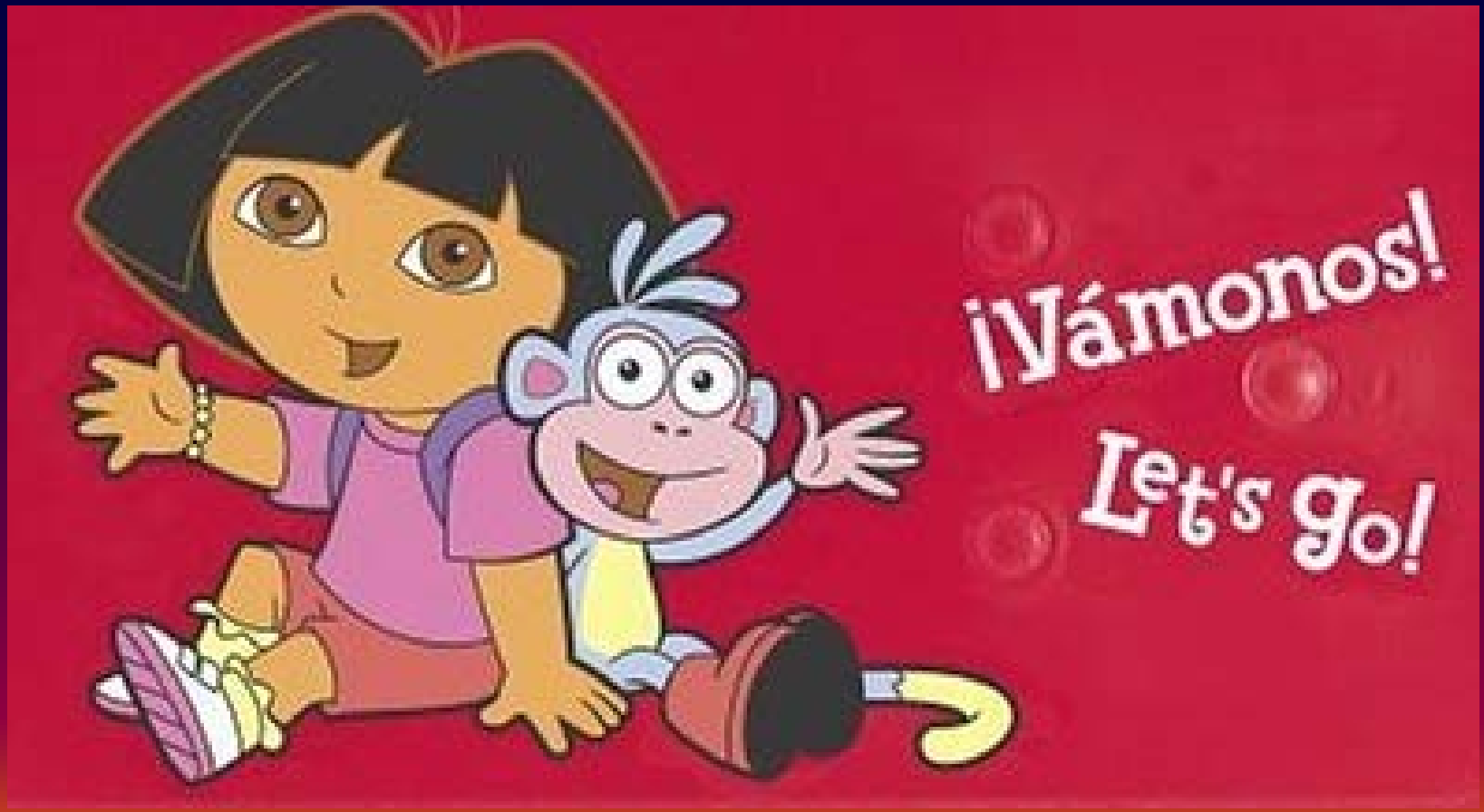


Και ο πραγματικός κόσμος που αντιπροσωπεύει



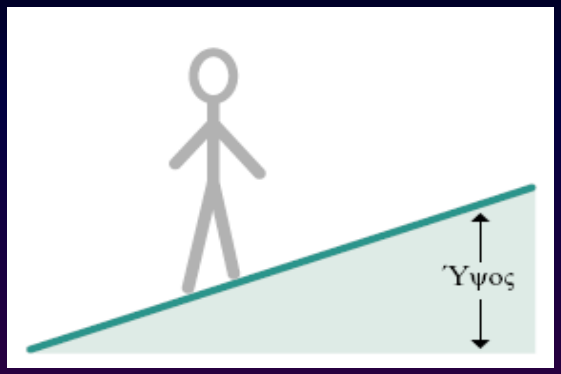


Κάθε πύλη εκτελεί μια "λογική συνάρτηση", συνάρτηση δηλαδή πάνω σε δυαδικές μεταβλητές

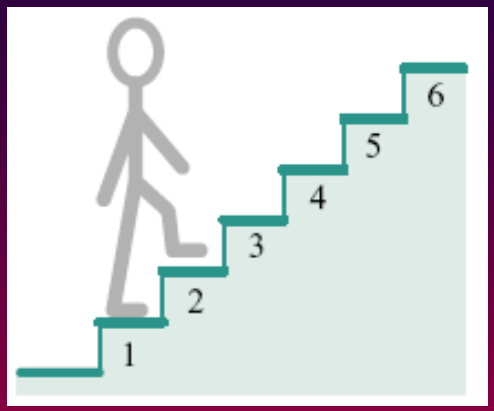

¡Vámonos!
Let's go!

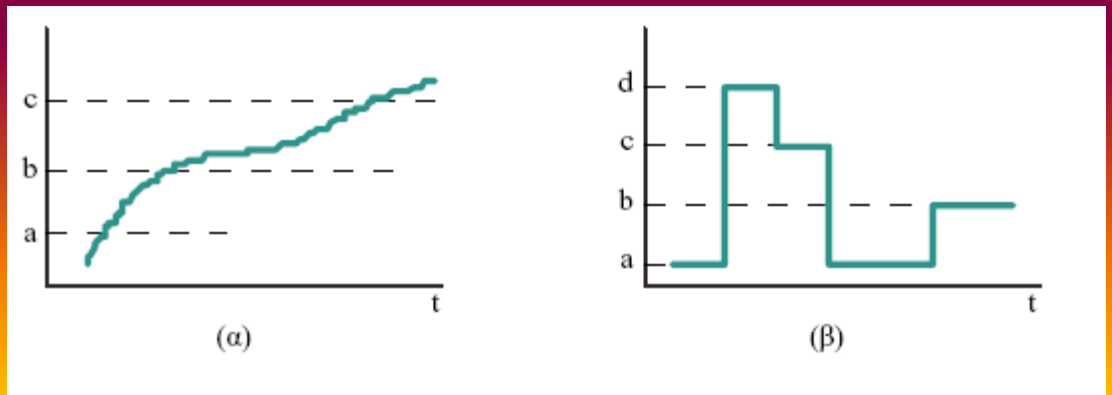
Αναλογικό vs Ψηφιακό

- Αναλογικό είναι ένα μέγεθος (σήμα) όταν μπορεί να πάρει οποιαδήποτε τιμή από ένα συνεχές διάστημα τιμών.
- Η πλειοψηφία των μεγεθών στο κόσμο μας είναι αναλογικά.

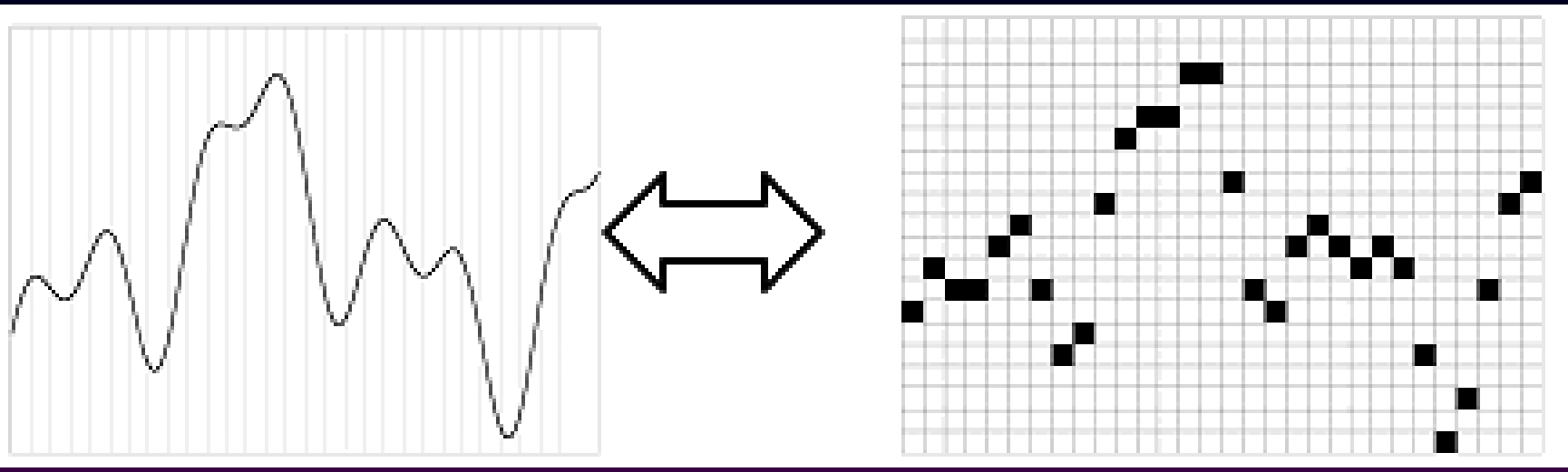


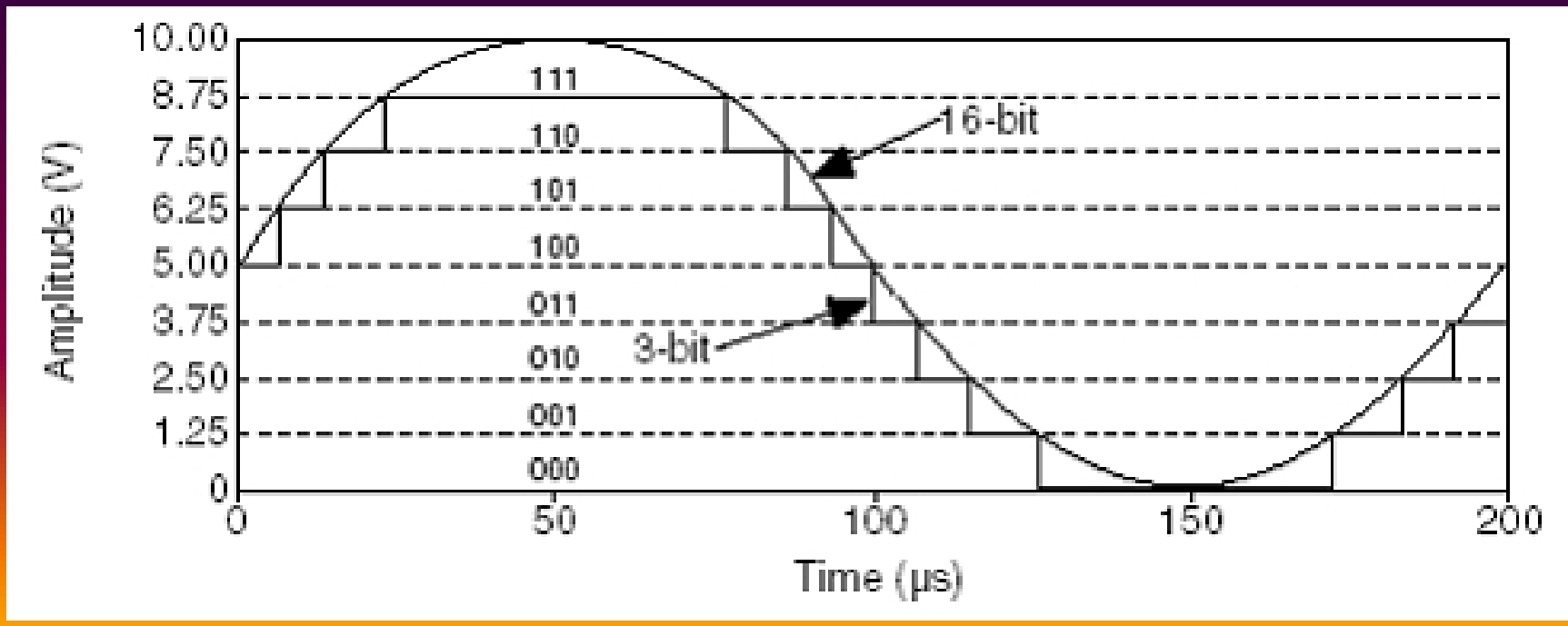
- Ψηφιακό (κβαντισμένο) είναι ένα μέγεθος (σήμα) όταν μπορεί να πάρει διακριτές τιμές.
- Φωτεινοί σηματοδότες, διακόπτες είναι παραδείγματα ψηφιακών μεγεθών.



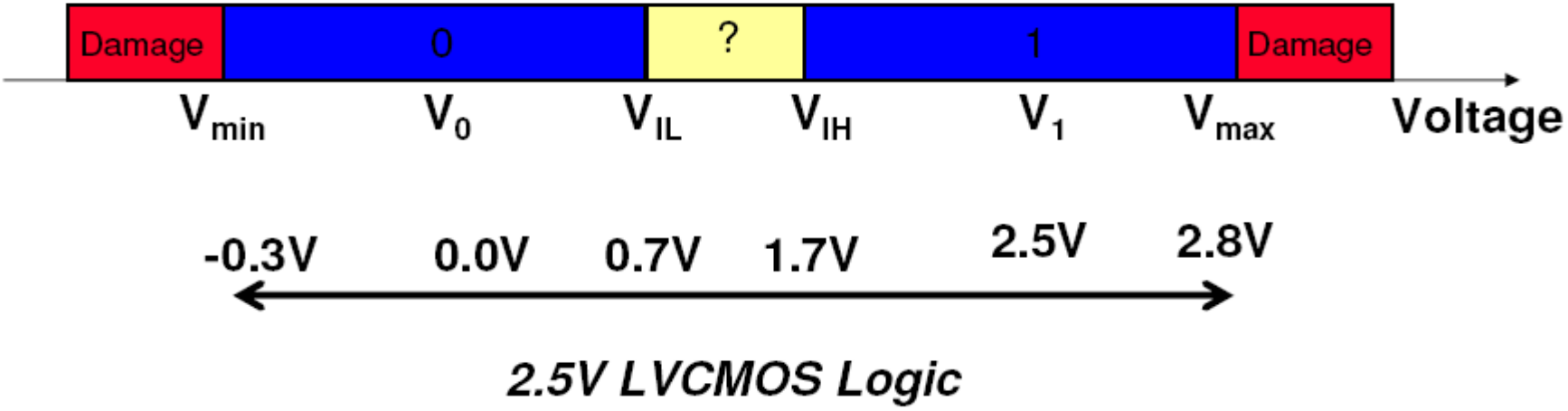


Δειγματοληψία & Κβάντιση





Δυαδικό σήμα : κβάντιση με 2 στάθμες



- Ανά πάσα στιγμή το σήμα μου έχει είτε τη τιμή 0 είτε τη τιμή 1.
- Το ποιο επίπεδο είναι 0 και ποιο είναι 1 είναι καθαρά θέμα σύμβασης.
- Συνεπώς μπορεί να αναπαρασταθεί από μια μεταβλητή 2 τιμών, μια ***δυαδική*** μεταβλητή.
- Η τιμή αυτής της μεταβλητής είναι και ένα ψηφίο του δυαδικού συστήματος αρίθμησης.
- Δεκαδικό ψηφίο : 0, 1, 2, 3, 4, 5,6, 7,8, 9
- Δυαδικό ψηφίο (BInary digiT – BIT) : 0, 1

Προτεραιότητα Τελεστών.

- ✓ Όπως και στην κανονική αριθμητική θα πρέπει να ορίσουμε μία σειρά προτεραιότητας για τους τελεστές της άλγεβρας Boole.
- ✓ Η προτεραιότητα που χρησιμοποιούμε ακολουθεί την εξής σειρά: πρώτες έρχονται οι παρενθέσεις, μετά τα συμπληρώματα (λογικό NOT), μετά το · (λογικό AND) και μετά το + (λογικό OR).
- ✓ Για παράδειγμα αν θέλαμε να υπολογίσουμε το $(x+y)'$ για διάφορους συνδυασμούς x και y θα πρέπει πρώτα να κάνουμε την πράξη OR μέσα στην παρένθεση και μετά να πάρουμε το συμπλήρωμα του αποτελέσματος.
- ✓ Στην $x'\cdot y'$ πρώτα παίρνουμε τα συμπληρώματα των x και y και μετά πραγματοποιούμε το λογικό AND.
- ✓ Στην $x+x\cdot y$ πρώτα πρέπει να πραγματοποιήσουμε το λογικό AND $x\cdot y$ και μετά το λογικό OR.
- ✓ Η προτεραιότητα των τελεστών θυμίζει την προτεραιότητα των πράξεων στην κανονική άλγεβρα αν αντιστοιχήσουμε το · στον πολλαπλασιασμό και το + στην πρόσθεση.

Συναρτήσεις Boole

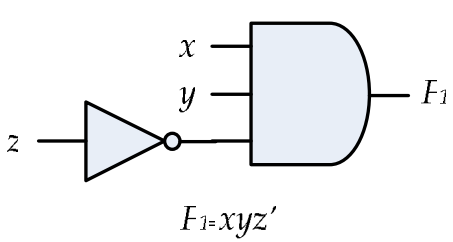
- ✓ Μια συνάρτηση Boole ονομάζεται οποιαδήποτε έκφραση μεταβλητών που ορίζονται μέσα στο B.
- ✓ Μπορεί να σχηματίζεται από παρενθέσεις, τους τελεστές +, · και συμπληρώματα.
- ✓ Παραδείγματα συναρτήσεων είναι η $F_1=xyz'$, η $F_2=x+y'\cdot z$ και η $F_3=x'\cdot y'\cdot z + x'\cdot y\cdot z+x\cdot y'$ και η $F_4=xy+x'z$.
- ✓ Για κάθε συνάρτηση Boole μπορούμε να φτιάξουμε τον πίνακα αλήθειας της που σε κάθε συνδυασμό τιμών των μεταβλητών της αντιστοιχεί μία τιμή στη συνάρτηση. Αν μία συνάρτηση είναι n μεταβλητών τότε έπεται πως ο πίνακας αλήθειας της έχει 2^n καταχωρήσεις.

x	y	z	F_1	F_2	F_3	F_4
0	0	0	0	0	0	0
0	0	1	0	1	1	1
0	1	0	0	0	0	0
0	1	1	0	0	1	1
1	0	0	0	1	1	1
1	0	1	0	1	1	1
1	1	0	1	1	0	0
1	1	1	0	1	0	0

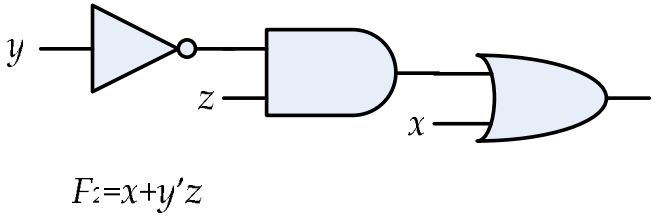
- ✓ Παρατηρούμε πως $F_3=F_4$ και επομένως μία συνάρτηση Boole δεν έχει μοναδικό τρόπο γραφής!

Αναπαράσταση Συναρτήσεων Boole με Πύλες

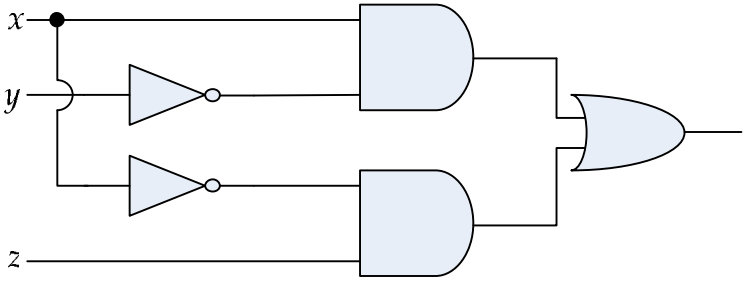
- ✓ Δεδομένης της έκφρασης της συνάρτησης Boole είναι σχετικά απλό να την αναπαραστήσουμε με λογικές πύλες.
- ✓ Ωστόσο θα πρέπει να προσέχουμε την προτεραιότητα των τελεστών.



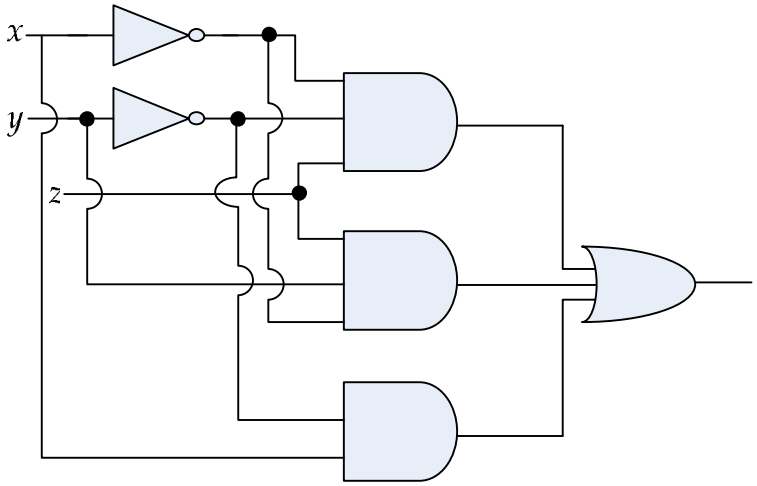
$F_1 = xyz'$



$F_2 = x + y'z$



$F_4 = xy' + x'z$



$F_3 = x'y'z + x'yz + xy'$

- ✓ Παρατηρούμε πόσο πιο απλό είναι το κύκλωμα της F_4 από το κύκλωμα της F_3. Επομένως ο τρόπος έκφρασης μιας συνάρτησης έχει άμεση σχέση με την πολυπλοκότητα του αντίστοιχου ψηφιακού κυκλώματος!

Συμπληρώματα Συναρτήσεων

- ✓ Εξ' ορισμού το συμπλήρωμα F' μιας συνάρτησης F είναι η συνάρτηση εκείνη η οποία ισούται με 1 όταν F=0 ενώ είναι ίση με 0 όταν F=1.
- ✓ Χρησιμοποιώντας το θεώρημα του DeMorgan μπορούμε να υπολογίσουμε το συμπλήρωμα μιας συνάρτησης.
- ✓ Για παράδειγμα αν $F=x+y+z$, έχουμε $F'=(x+y+z)'=x'y'z'$
- ✓ Γενικά οποιοδήποτε άθροισμα της μορφής $a_1 \cdot a_2 \cdot a_3 \cdot \ldots \cdot a_N + b_1 \cdot \ldots \cdot b_N + \ldots$ έχει ως συμπλήρωμα το $(a_1' + a_2' + a_3' + \ldots + a_N') \cdot (b_1' + \ldots + b_N') \cdot \ldots$
- ✓ Επομένως κάθε μεταβλητή εμφανίζεται με το συμπλήρωμα της, τα λογικά OR μετατρέπονται σε λογικά AND και το αντίστροφο.

Κανονικές και Πρότυπες Μορφές

- ✓ Για ένα σύνολο n μεταβλητών $\{a_1,\ldots,a_N\}$ μπορούμε να σχηματίσουμε 2^n διαφορετικά γινόμενα της μορφής $a_1'\cdot\ldots\cdot a_N$ όπου κάθε μεταβλητή μπορεί να συμμετέχει είτε αυτούσια, είτε με το συμπλήρωμα της.
- ✓ Κάθε τέτοιο γινόμενο ονομάζεται ελαχιστόρος ή πρότυπο γινόμενο. Κάθε ελαχιστόρος ισούται με 1 για έναν και μόνο συνδυασμό τιμών των μεταβλητών $a_1,\ldots, a_N$
- ✓ Ο παρακάτω πίνακας δείχνει τους ελαχιστόρους που παράγονται από 3 μεταβλητές (x,y,z). Κάθε όρος ονομάζεται m_i όπου i είναι το δεκαδικό ισοδύναμο του δυαδικού συνδυασμού για τον οποίο ο ελαχιστόρος είναι ίσος με 1.

x	y	z	Ελαχιστόρος	Ονομασία
0	0	0	$x'y'z'$	m_0
0	0	1	$x'y'z$	m_1
0	1	0	$x'yz'$	m_2
0	1	1	$x'yz$	m_3
1	0	0	$xy'z'$	m_4
1	0	1	$xy'z$	m_5
1	1	0	xyz'	m_6
1	1	1	xyz	m_7

Κανονικές και Πρότυπες Μορφές

- ✓ Ομοίως, για ένα σύνολο n μεταβλητών $\{a_1,\ldots,a_N\}$ μπορούμε να σχηματίσουμε 2^n διαφορετικά *αθροίσματα* της μορφής $a_1'+\ldots+a_N$ όπου κάθε μεταβλητή μπορεί να συμμετέχει είτε αυτούσια, είτε με το συμπλήρωμα της.
- ✓ Κάθε τέτοιο άθροισμα ονομάζεται μεγιστόρος ή πρότυπο άθροισμα. Κάθε μεγιστόρος ισούται με 0 για έναν και μόνο συνδυασμό τιμών των μεταβλητών $a_1,\ldots, a_N$
- ✓ Ο παρακάτω πίνακας δείχνει τους μεγιστόρους που παράγονται από 3 μεταβλητές (x,y,z). Κάθε όρος ονομάζεται M_i όπου i είναι το δεκαδικό ισοδύναμο του δυαδικού συνδυασμού για τον οποίο ο μεγιστόρος είναι ίσος με 0.

x	y	z	Μεγιστόρος	Ονομασία
0	0	0	$x+y+z$	M_0
0	0	1	$x+y+z'$	M_1
0	1	0	$x+y'+z$	M_2
0	1	1	$x+y'+z'$	M_3
1	0	0	$x'+y+z$	M_4
1	0	1	$x'+y+z'$	M_5
1	1	0	$x'+y'+z$	M_6
1	1	1	$x'+y'+z'$	M_7

Πίνακες Αλήθειας και Πρότυπες Μορφές

- ✓ Δεδομένου του πίνακα αλήθειας μίας συνάρτησης μπορούμε να την γράψουμε ως γινόμενο μεγιστόρων ή άθροισμα ελαχιστόρων.
- ✓ Για να την γράψουμε ως γινόμενο μεγιστόρων, απλά πολλαπλασιάζουμε τους μεγιστόρους για τους οποίους η συνάρτηση είναι ίση με 0.
- ✓ Για να την γράψουμε ως άθροισμα ελαχιστόρων, απλά αθροίζουμε τους ελαχιστόρους για τους οποίους η συνάρτηση είναι ίση με 1.
- ✓ Για παράδειγμα έστω μία συνάρτηση F τριών μεταβλητών (x,y,z) με τον παρακάτω πίνακα αλήθειας:

x	y	z	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

- ✓ Μπορούμε να γράψουμε την F, ως $F=m_1+m_4+m_7$ ή ως $F=M_0 \cdot M_2 \cdot M_3 \cdot M_5 \cdot M_6$.
- ✓ Για συντομία, μπορούμε να γράψουμε την F και στην εξής μορφή:
- ✓ $F=\Sigma(1,4,7)$ και $F=\Pi(0,2,3,5,6)$. Παρατηρείστε πως αν ο ελαχιστόρος m_j ανήκει στο άθροισμα ελαχιστόρων της F, τότε ο M_j δεν ανήκει στο γινόμενο μεγιστόρων της F και αντίστροφα.
- ✓ Δεδομένου πως $m_j'=M_j$ μπορούμε να συνάγουμε πως αν $F=\Sigma(i_1,i_2,\ldots,i_N)$ τότε $F'=\Pi(i_1,i_2,\ldots,i_N)$
- ✓ Επίσης αν $F=\Pi(i_1,i_2,\ldots,i_N)$ τότε $F'=\Sigma(i_1,i_2,\ldots,i_N)$

Άλλες Λογικές Πράξεις

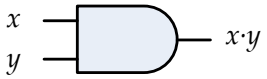
- ✓ Η πράξεις του λογικού AND και του λογικού OR είναι στην ουσία λογικές συναρτήσεις δύο μεταβλητών στις οποίες αντιστοιχούν μία τιμή 0 ή 1 σε κάθε δυνατή δυάδα των μεταβλητών (x,y). Ωστόσο υπάρχουν άλλες 14 πιθανές συναρτήσεις που μπορούν να κατασκευαστούν, όπως δείχνει και ο παρακάτω πίνακας:

x	y	F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
Σύμβολο Τελεστή			·	/		/		⊕	+	↓	⊙	‘	⊂	‘	⊃	↑	

Άλλες Λογικές Πράξεις

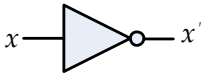
Συναρτήσεις Boole	Σύμβολο Τελεστή	Όνομα	Σχόλια
$F_0=0$		Ουδέτερη	Δυαδική Σταθερά 0
$F_1=x\cdot y$	$x\cdot y$	Λογικό AND	x AND y
$F_2=x\cdot y'$	x/y	Αποτροπή	x αλλά όχι y
$F_3=x$		Μεταφορά	x
$F_4=x'y$	y/x	Αποτροπή	y αλλά όχι x
$F_5=y$		Μεταφορά	y
$F_6=xy'+x'y$	$\oplus$	Αποκλειστικό OR	είτε x είτε y
$F_7=x+y$	+	Λογικό OR	x OR y
$F_8=(x+y)'$	$x\downarrow y$	Λογικό NOR	x NOR y
$F_9=x'y'+xy$	$x\odot y$	Ισοδυναμία	$x = y$
$F_{10}=y'$	y'	Συμπλήρωμα	NOT y
$F_{11}=x+y'$	$x\subset y$	Συνεπαγωγή	Αν ισχύει το y τότε x
$F_{12}=x'$	x'	Συμπλήρωμα	NOT x
$F_{13}=x'+y$	$x\supset y$	Συνεπαγωγή	Αν ισχύει το x τότε y
$F_{14}=(xy)'$	$x\uparrow y$	Λογικό NAND	x NAND y
$F_{15}=1$		Ταυτότητα	Δυαδική Σταθερά 1

Άλλες Λογικές Πύλες

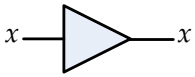


AND

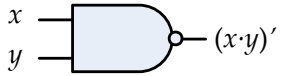
OR



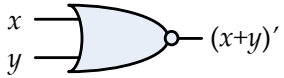
NOT



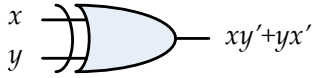
Απομονωτής



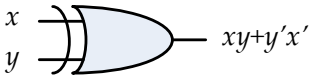
NAND



NOR



XOR

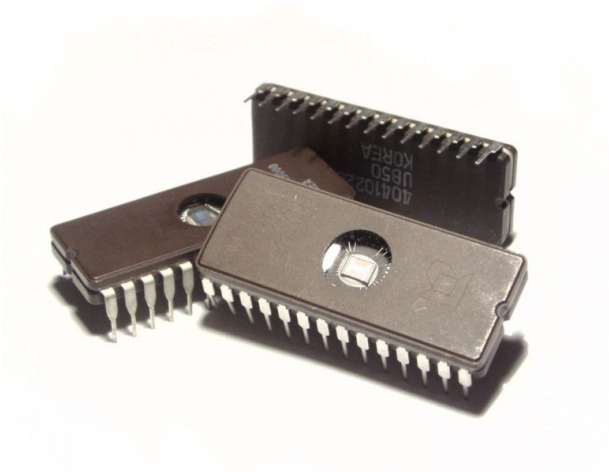


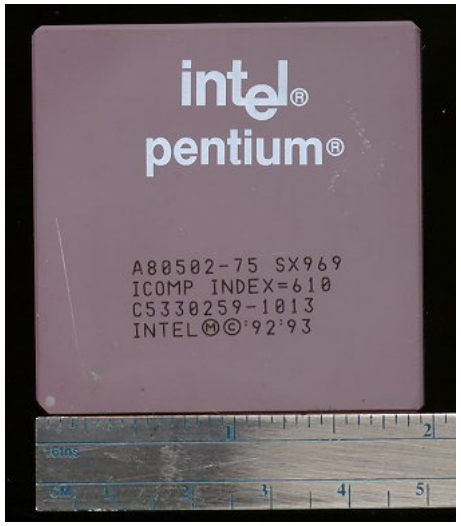
XNOR

- ✓ Πέρα από τις 3 βασικές λογικές πύλες (AND, OR, NOT), ορίζονται και άλλες 3 λογικές πύλες:
- ✓ ο Απομονωτής συνήθως χρησιμοποιείται μόνο για ενίσχυση του σήματος ώστε να μπορεί να οδηγηθεί και σε άλλες λογικές πύλες.
- ✓ Οι πύλες NAND και NOR είναι στην ουσία τα συμπληρώματα των AND και OR.
- ✓ Οι NAND και NOR χρησιμοποιούνται ευρύτατα επειδή είναι εύκολο να υλοποιηθούν με τρανζίστορ και επειδή οι πιο πολύπλοκες συναρτήσεις Boole μπορούν εύκολα να υλοποιηθούνε με αυτές.
- ✓ Οι πύλες XOR χρησιμοποιούνται και αυτές αρκετά συχνά (π.χ. Σε αθροιστές, κτλ κτλ).
- ✓ Εκτός από την πύλη NOT και τον απομονωτή, οι υπόλοιπες πύλες μπορούνε εύκολα να επεκταθούνε για παραπάνω από δύο εισόδους.
- ✓ Στις πύλες AND και OR δεν έχει σημασία η σειρά με την οποία θεωρούμε τις μεταβλητές εισόδου αφού ισχύει η προσαιτεριστική και η αντιμεταθετική ιδιότητα, οπότε για παράδειγμα (x+y)+z=x+(y+z)=x+y+z=x+z+y, κτλ κτλ.
- ✓ Για τις πύλες NAND και NOR χρειάζεται λίγο προσοχή αφού ναι μεν είναι αντιμεταθετικές αλλά δεν είναι προσαιτεριστικές. Επομένως ορίζουμε απευθείας την NOR τριων μεταβλητών ως (x+y+z)' και την NAND ως (xyz)'
- ✓ Οι πύλες XOR και XNOR πολλαπλών εισόδων μπορούν να θεωρηθούν ως κυκλώματα ισοτιμίας!

Ολοκληρωμένα Κυκλώματα

- ✓ Όπως είδαμε και στο προηγούμενο εξάμηνο, τα ηλεκτρονικά κυκλώματα μπορούν να κατασκευασθούν σε ολοκληρωμένη μορφή (integrated circuits – ICs).
- ✓ Στο εσωτερικό των ψηφιακών ICs βρίσκονται πολλές λογικές πύλες ενωμένες μεταξύ τους ώστε να σχηματιστεί το απαιτούμενο ψηφιακό κύκλωμα.
- ✓ Τα chips έχουνε έναν αριθμό από «ποδαράκια» (pins) τα οποία χρησιμοποιούνται με σκοπό την επικοινωνία του ψηφιακού κυκλώματος με τον έξω κόσμο.

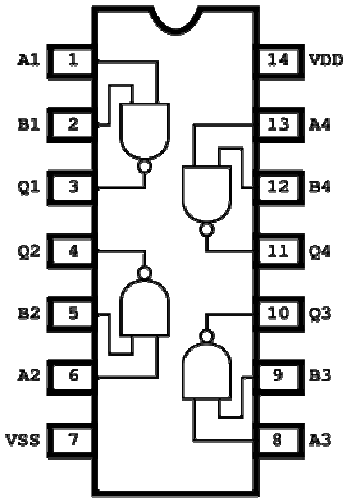




- ✓ Τα ICs επινοήθηκαν για πρώτη φορά από τον Geoffrey William Arnold Dummer, ο οποίος κατάφερε να κατασκευάσει ένα ολοκληρωμένο κύκλωμα για πρώτη φορά το 1952.

Επίπεδα Ολοκλήρωσης

7400



INTEL Core 2 Duo

- ✓ Στο προηγούμενο εξάμηνο είδαμε πως ανάλογα με το επίπεδο ολοκλήρωσης (δηλαδή πόσα τρανζίστορ χωράνε σε ένα chip) υπάρχουν τέσσερις κατηγορίες ολοκληρωμένων κυκλωμάτων:
- ✓ SSI (Small Scale of Integration)
- ✓ MSI (Medium Scale of Integration)
- ✓ LSI (Large Scale of Integration)
- ✓ VLSI (Very Large Scale of Integration)

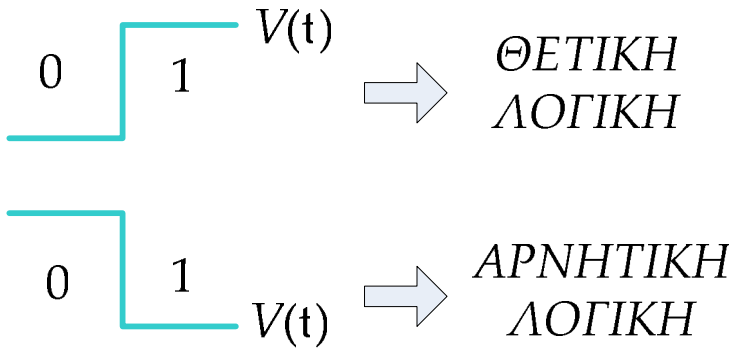
Οικογένειες Ψηφιακής Λογικής

- ✓ Ανάλογα με το πως υλοποιούνται οι διάφορες πύλες (συνήθως μας ενδιαφέρουν οι NAND και NOR) τα ψηφιακά ολοκληρωμένα κυκλώματα χωρίζονται σε οικογένειες ψηφιακής λογικής.
- ✓ TTL: Transistor – Transistor Logic
- ✓ ECL: Emitter Coupled Logic
- ✓ MOS: Metal Oxide Semiconductor
- ✓ CMOS: Complementary MOS

- ✓ Κάθε οικογένεια χαρακτηρίζεται ως προς τις επιδόσεις της στα εξής:
- ✓ *Ικανότητα οδήγησης* (Fan Out): Πόσα φορτία μπορεί να οδηγήσει η έξοδος μιας πύλης χωρίς να κινδυνέψει η κανονική της λειτουργία (ως φορτίο συνήθως ορίζουμε το ρεύμα που χρειάζεται η είσοδος μιας πύλης της ιδίας οικογένειας).
- ✓ *Κατανάλωση Ισχύος* (power dissipation): Πόση ισχύ τροφοδοσίας χρειάζεται η κάθε πύλη.
- ✓ *Καθυστέρηση Διάδοσης* (propagation delay): Ο μέσος χρόνος που χρειάζεται για να διαδοθεί η αλλαγή του σήματος από την είσοδο, στην έξοδο.
- ✓ *Περιθώριο Θορύβου* (Noise Margin): Ελάχιστη τάση εξωτερικού θορύβου που προκαλεί ανεπιθύμητη αλλαγή στην έξοδο.

Θετική και Αρνητική Λογική

- ✓ Ανάλογα με τον τρόπο υλοποίησης μιας λογικής πύλης, το λογικό «1» μπορεί να αντιστοιχείται σε υψηλή τάση (H) ή χαμηλή τάση (L).



- ✓ Στο μάθημα θα ασχοληθούμε με ψηφιακά κυκλώματα στα οποία οι πύλες έχουνε υλοποιηθεί με τη θετική λογική: 0→L και 1→H

ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ
(Τ.Ε.Ι.) ΚΡΗΤΗΣ
Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων

Ψηφιακή Σχεδίαση

Κεφάλαιο 2: Συνδυαστικά Λογικά Κυκλώματα

Γ. Κορνάρος

Περίγραμμα

- Μέρος 1 – Κυκλώματα Πυλών και Boolean Εξισώσεις
  - Δυαδική Λογική και Πύλες
  - Boolean Άλγεβρα
  - Standard Μορφές
- Μέρος 2 – Βελτιστοποίηση Κυκλωμάτων
  - Δι-επίπεδη Βελτιστοποίηση
  - Χειρισμός Χαρτών
  - Practical Optimization (Espresso)
  - Βελτιστοποίηση Κυκλωμάτων πολλών επιπέδων
- Μέρος 3 – Επιπλέον Πύλες και κυκλώματα
  - Άλλοι τύποι πυλών
  - Τελεστής Αποκλειστικού-Ή και πύλες
  - Έξοδοι υψηλής-εμπέδησης

Δυαδική Λογική και Πύλες

- Οι Δυαδικές μεταβλητές παίρνουν μία από τις δύο δυνατές τιμές.
- Οι λογικοί τελεστές λειτουργούν σε δυαδικές τιμές και δυαδικές μεταβλητές.
- Οι βασικοί λογικοί τελεστές είναι οι λογικές συναρτήσεις AND, OR και NOT.
- Οι Λογικές Πύλες υλοποιούν λογικές συναρτήσεις.
- Boolean Algebra: ένα χρήσιμο μαθηματικό σύστημα για περιγραφή και μετασχηματισμούς λογικών συναρτήσεων.
- Μελετούμε την Boolean algebra διότι είναι η βάση για σχεδιασμό και ανάλυση Ψηφιακών Συστημάτων!

Λογικές Λειτουργίες

- Οι τρεις βασικές λογικές πράξεις είναι:
  - AND
  - OR
  - NOT
- AND συμβολίζεται με (·)
- OR συμβολίζεται με πρόσθεση (+)
- NOT συμβολίζεται με μία παύλα (¯), μία απόστροφο (') μετά , ή ένα (~) πριν την μεταβλητή.

Πίνακας Αλήθειας

- *Truth table* :

  μία αναπαράσταση των τιμών μιας συνάρτησης σε πίνακα για όλους του δυνατούς συνδυασμούς των μεταβλητών (ορισμάτων) της συνάρτησης.

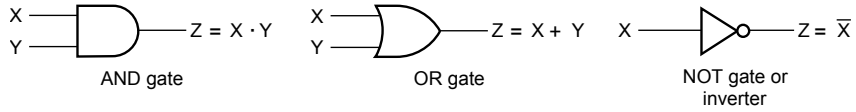
Boolean Άλγεβρα

- Τρεις βασικές λειτουργίες μπορούν να γίνουν σε μεταβλητές Boole
  - και - and (juxtaposition) [•]: 0•0 = 0, 0•1 = 0, 1•0 = 0, 1•1 = 1
  - ή - or [+]: 0+0 = 0, 0+1 = 1, 1+0 = 1, 1+1 = 1
  - συμπλήρωμα - complement (not or negation) x' or x: 0' = 1, 1' = 0
- Οι πράξεις αυτές μπορούν να αναπαρασταθούν σε πίνακα αλήθειας (truth table):

A	B	A+B	A•B	A’	B’
0	0				
0	1				
1	0				
1	1				

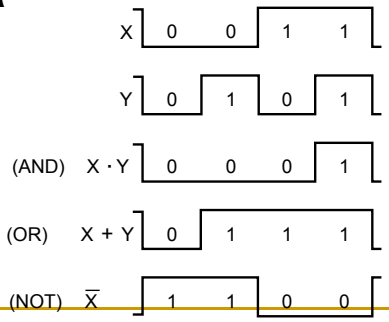
Σύμβολα Λογικών Πυλών - Συμπεριφορά

- Ειδικά σύμβολα για αναπαράσταση λογικών πυλών:



(a) Graphic symbols

- **Συμπεριφορά** αναπαράσταση στο χρόνο, κυματομορφές - waveform :



(b) Timing diagram

Λογικά Διαγράμματα και Εκφράσεις

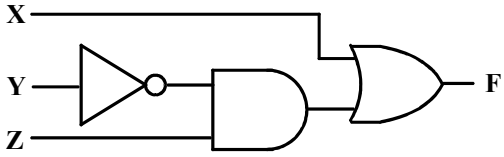
Πίνακας Αλήθειας

X Y Z	F = X + $\overline{Y}$ · Z
0 0 0	0
0 0 1	1
0 1 0	0
0 1 1	0
1 0 0	1
1 0 1	1
1 1 0	1
1 1 1	1

Εξίσωση

$$F = X + \overline{Y}\,Z$$

Logic Diagram



- Η ίδια συνάρτηση μπορεί να περιγράφεται με εξισώσεις Boolean, πίνακες αλήθειας και λογικά διαγράμματα!
  - Οι Πίνακες αλήθειας είναι μοναδικοί.
  - Οι εξισώσεις και τα λογικά διαγράμματα δεν είναι.

Άλγεβρα Boole : αξιωματικός ορισμός

- Η άλγεβρα (switching) Boole είναι ένα κλειστό αλγεβρικό σύστημα S,
  - περιέχει δύο στοιχεία {0,1},
  - δύο τελεστές + (OR) . (AND)
- Κλειστότητα
- Προσεταιριστικός νόμος
- Αντιμεταθετικός νόμος
- Ουδέτερο στοιχείο
- Αντίστροφο
- Επιμεριστικός νόμος

Αξιώματα και Θεωρήματα

• *Ταυτότητα*
– 1a: a+a = a
– 1b: a•a = a
• *Ουδέτερο στοιχείο για τους τελεστές + and •*

– 2a: a+1 = 1
– 2b: a•0 = 0
• *Διπλή άρνηση - Involution*
– 3: (a')' = a

• *Απορρόφηση*
– 4a: a+ab = a
– 4b: a(a+b) = a

– 5a: a+a'b = a+b
– 5b: a(a'+b) = ab
– 6a: ab+ab' = a
– 6b: (a+b)(a+b') = a
– 7a: ab+ab'c = ab+ac
– 7b: (a+b)(a+b'+c) = (a+b)(a+c)

Βασικά Θεωρήματα

• **DeMorgan's (Theorem 5)**
– 8a: (a+b)' = a'•b'
– 8b: (a•b)' = a'+b'

• Consensus (Theorem 9)
– 9a: ab+a'c+bc = ab+a'c
– 9b: (a+b)(a'+c)(b+c) = (a+b)(a'+c)
• Shannon's Expansion (Theorem 10)
–10a: F(x 1 ,x 2 ,…,x n) =
[x 1 •F(1,x 2 ,…,x n)] + [x 1 '•F(0,x 2 ,…,x n)]
–10b: F(x 1 ,x 2 ,…,x n) =
[x 1 +F(1,x 2 ,…,x n)] • [x 1 '+F(0,x 2 ,…,x n)]

Θεώρημα : x + xy = x

Απόδειξη:

x + xy = x . 1 + x y
= x (1 + y)
= x . 1
= x

Λόγω δυϊσμού : x . (x + y) = x

Βασικά Θεωρήματα

- ***n*-Variable Theorems (Generalized)**
- Idempotency
  - T11a: x+x+…+x = x
  - T11b: x•x•…•x = x

- DeMorgan's
  - T12a: (x1 • x2 •…• xn)' = x1'+x2'+…+xn'
  - T12b: (x1+x2+…+xn)' = x1'• x2'•…• xn'

(A+B+C)' = (A+x)' ,όπου x= B+C
= A'x'
= A'(B+C)'
= A'(B'C')
= A'B'C'

Επανάληψη

1.	$X + 0 = X$	2.	$X \cdot 1 = X$
3.	$X + 1 = 1$	4.	$X \cdot 0 = 0$
5.	$X + X = X$	6.	$X \cdot X = X$
7.	$X + \overline{X} = 1$	8.	$X \cdot \overline{X} = 0$
9.	$\overline{\overline{X}} = X$		
10.	$X + Y = Y + X$	11.	$XY = YX$ Αντιμεταθετική
12.	$(X + Y) + Z = X + (Y + Z)$	13.	$(XY)Z = X(YZ)$ Προσεταιριστική
14.	$X(Y + Z) = XY + XZ$	15.	$X + YZ = (X + Y)(X + Z)$ Επιμεριστική
16.	$\overline{X + Y} = \overline{X} \cdot \overline{Y}$	17.	$\overline{X \cdot Y} = \overline{X} + \overline{Y}$ DeMorgan's

Υπολογισμός συνάρτησης Boolean

$F1 = xy\bar{z}$
$F2 = x + \bar{y}z$
$F3 = \bar{x}\bar{y}\bar{z} + \bar{x}\,y\,z + x\bar{y}$
$F4 = x\bar{y} + \bar{x}\,z$

x	y	z	F1	F2	F3	F4
0	0	0	0	0		
0	0	1	0	1		
0	1	0	0	0		
0	1	1	0	0		
1	0	0	0	1		
1	0	1	0	1		
1	1	0	1	1		
1	1	1	0	1		

Συμπλήρωμα συνάρτησης ?

F1' =

Απλοποίηση συναρτήσεων

- x(x´+y) = x´x + xy = 0 +xy = xy
- x+x´y = (x + x´)(x + y) = 1 (x + y) = x + y
- (x + y)(x + y´) = xx + xy + xy´ + yy´ =
  = x (1 + y + y´) = x
- xy + x´z + yz = xy + x´z + yz(x+x´)
  = xy + x´z + xyz + x´yz
  =xy(1 + z) + x´z (1 + y)
  = xy + x´z

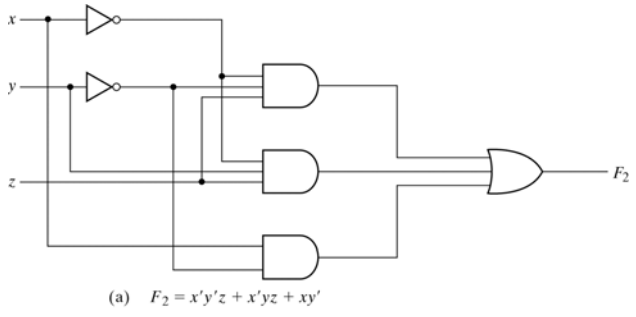
Απλοποίηση Έκφρασης

- Εφαρμογή της άλγεβρας Boole.
- Απλοποίηση ώστε να περιέχεται ο μικρότερος αριθμός μεταβλητών (είτε στην κανονική είτε ως συμπλήρωμα):

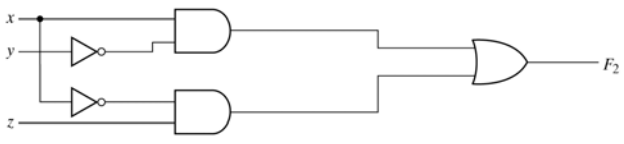
$$AB + \overline{A}CD + \overline{A}BD + \overline{A}C\overline{D} + ABCD$$

- = AB + ABCD + A´C D + A´C D´ + A´ B D
- = AB + AB(CD) + A´C (D + D´) + A´B D
- = AB + A´C + A´B D
- = B(A + A´D) +A´C
- = B(A + D) + A´C

Έκφραση συνάρτησης Boole με πύλες



(a) $F_2 = x'y'z + x'yz + xy'$



(b) $F_2 = xy' + x'z$

Fig. 2-2 Implementation of Boolean function F_2 with gates

Κανονικές και Πρότυπες Μορφές

	Ελαχιστόροι		Μεγιστόροι	
x y z	Terms	Name	Terms	Name
0 0 0	x' y' z'	m_0	x+y+z	M_0
0 0 1	x' y' z	m_1	x+y+z'	M_1
0 1 0	x' y z'	m_2	x+y'+z	M_2
0 1 1	x' y z	m_3	x+y'+z'	M_3
1 0 0	x y' z'	m_4	x'+y+z	M_4
1 0 1	x y' z	m_5	x'+y+z'	M_5
1 1 0	x y z'	m_6	x'+y'+z	M_6
1 1 1	x y z	m_7	x'+y'+z'	M_7

Ελαχιστόροι

x	y	z	Corresponding minterm	Designation
0	0	0	x'y'z'	m_0
0	0	1	x'y'z	m_1
0	1	0	x'yz'	m_2
0	1	1	x'yz	m_3
1	0	0	xy'z'	m_4
1	0	1	xy'z	m_5
1	1	0	xyz'	m_6
1	1	1	xyz	m_7

- Μία μεθοδος αναπαράστασης συναρτήσεων Boole είναι η κανονική μορφή ελαχιστόρων (άθροισμα γινομένων ή SOP),
- F = x'y'z +xy'z + xyz' = m1 + m5 + m6 = Σ(1,5,6)

Ελαχιστόροι - παραδείγματα

x	y	z	F2 (Given)	Designation
0	0	0	1	m_0
0	0	1	1	m_1
0	1	0	1	m_2
0	1	1	1	m_3
1	0	0	0	
1	0	1	1	m_5
1	1	0	0	
1	1	1	0	

- F2 = Σ(0,1,2,3,5)
- = x'y'z' + x'y'z + x'yz' + x'yz + xy'z

Ελαχιστόροι – παραδείγματα – F'

x	y	z	F2 (Given)	Designation
0	0	0	1	m_0
0	0	1	1	m_1
0	1	0	1	m_2
0	1	1	1	m_3
1	0	0	0	
1	0	1	1	m_5
1	1	0	0	
1	1	1	0	

- (F2)' = Σ(all minterms not in F2) = Σ(4,6,7)
- = xy'z' + xyz' + xyz

Μεγιστόροι

x	y	z	Corresponding maxterm	Designation
0	0	0	x +y +z	M_0
0	0	1	x +y +z'	M_1
0	1	0	x +y' +z	M_2
0	1	1	x +y' +z'	M_3
1	0	0	x' +y +z	M_4
1	0	1	x' +y +z'	M_5
1	1	0	x' +y' +z	M_6
1	1	1	x' +y' +z'	M_7

- Ένας Μεγιστόρος είναι ένας όρος από OR
- Κάθε Μεγιστόρος έχει τιμή 0 για ένα συνδυασμό τιμών των N μεταβλητών

Μετατροπή μεταξύ Ελαχιστόρων - Μεγιστόρων

- m0 = x'y'z' = (x+y+z)' = (M0)'
- Γενικά, mi = (Mi)'

Παράδειγμα : Μετατροπή σε γινόμενο αθροισμάτων

- F=xy+x'z
  - = (xy + x')(xy + z)
  - = (x + x')(y + x')(x + z)(y + z)
  - = (x' + y)(x + z)(y + z)

x'+y = x' + y + zz' = (x'+y+z)(x'+y+z')

.

.

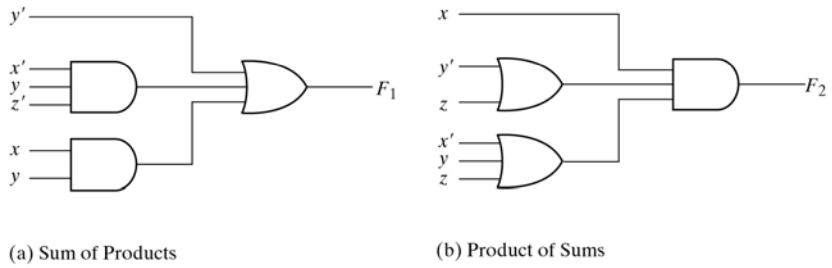
F = $M_0M_2M_4M_5$

Μεγιστόροι

x	y	z	F3 (Given)	Designation
0	0	0	0	M_0
0	0	1	1	
0	1	0	0	M_2
0	1	1	0	M_3
1	0	0	0	M_4
1	0	1	1	
1	1	0	1	
1	1	1	0	M_7

- F3 = (x+y+z)(x+y'+z)(x+y'+z')(x'+y+z)(x'+y'+z')
- = π(0,2,3,4,7)
- (F3)' = π(all maxterm not in F3)

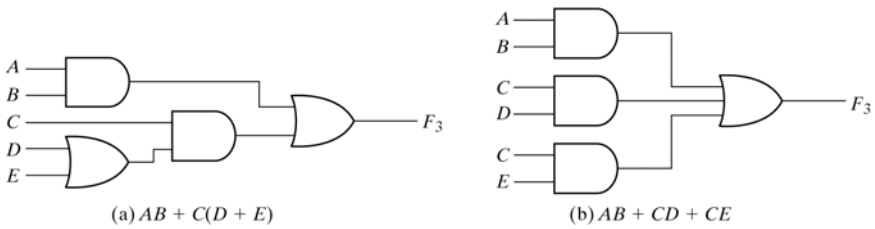
Υλοποίηση δύο επιπέδων



(a) Sum of Products (b) Product of Sums

Fig. 2-3 Two-level implementation

Υλοποίηση τριών επιπέδων και δύο επιπέδων



(a) $AB + C(D + E)$ (b) $AB + CD + CE$

Fig. 2-4 Three- and Two-Level implementation

Άλλες λογικές πράξεις

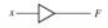
Name	Graphic symbol	Algebraic function	Truth table
AND		$F = xy$	$x\ y \mid F$ $0\ 0 \mid 0$ $0\ 1 \mid 0$ $1\ 0 \mid 0$ $1\ 1 \mid 1$
OR		$F = x + y$	$x\ y \mid F$ $0\ 0 \mid 0$ $0\ 1 \mid 1$ $1\ 0 \mid 1$ $1\ 1 \mid 1$
Inverter		$F = x'$	$x \mid F$ $0 \mid 1$ $1 \mid 0$
Buffer		$F = x$	$x \mid F$ $0 \mid 0$ $1 \mid 1$
NAND		$F = (xy)'$	$x\ y \mid F$ $0\ 0 \mid 1$ $0\ 1 \mid 1$ $1\ 0 \mid 1$ $1\ 1 \mid 0$
NOR		$F = (x + y)'$	$x\ y \mid F$ $0\ 0 \mid 1$ $0\ 1 \mid 0$ $1\ 0 \mid 0$ $1\ 1 \mid 0$
Exclusive-OR (XOR)		$F = xy' + x'y$ $= x \oplus y$	$x\ y \mid F$ $0\ 0 \mid 0$ $0\ 1 \mid 1$ $1\ 0 \mid 1$ $1\ 1 \mid 0$
Exclusive-NOR or equivalence		$F = xy + x'y'$ $= (x \oplus y)'$	$x\ y \mid F$ $0\ 0 \mid 1$ $0\ 1 \mid 0$ $1\ 0 \mid 0$ $1\ 1 \mid 1$

Fig. 2-5 Digital logic gates

Προσοχή! Προσεταιριστική ιδιότητα με NOR

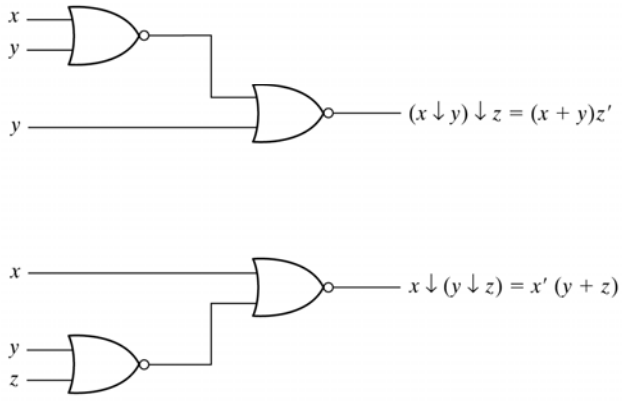
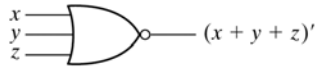
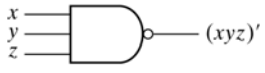


Fig. 2-6 Demonstrating the nonassociativity of the NOR operator; $(x \downarrow y) \downarrow z \neq x \downarrow (y \downarrow z)$

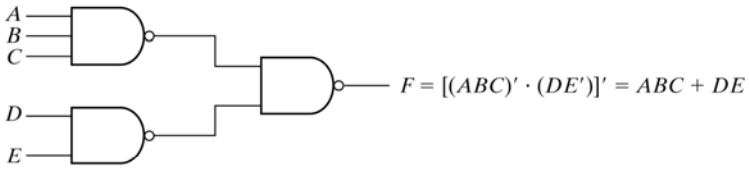
Πύλες πολλαπλών εισόδων



(a) 3-input NOR gate



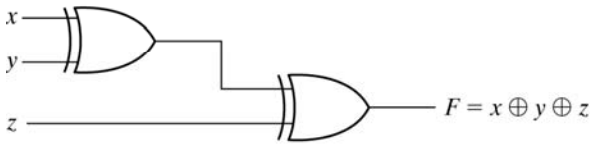
(b) 3-input NAND gate



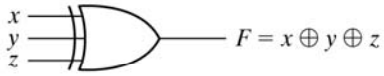
(c) Cascaded NAND gates

Fig. 2-7 Multiple-input and cascated NOR and NAND gates

Πύλη αποκλειστικού-Ή



(a) Using 2-input gates



(b) 3-input gate

x	y	z	F
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

(c) Truth table

Fig. 2-8 3-input exclusive-OR gate

Θετική και Αρνητική Λογική

- Θετική λογική: αντιστοίχιση του υψηλού δυναμικού στο λογικό “1”

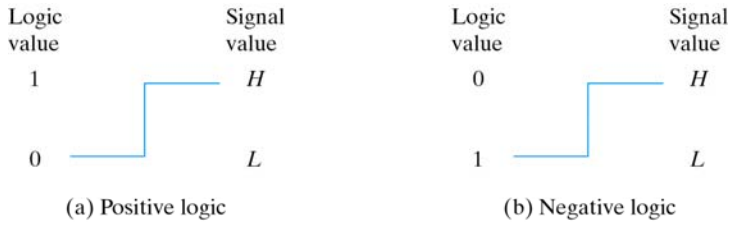
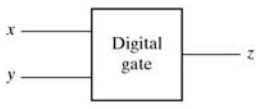


Fig. 2-9 signal assignment and logic polarity

Θετική και Αρνητική Λογική

x	y	F
L	L	L
L	H	L
H	L	L
H	H	H

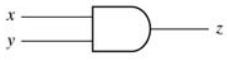
(a) Truth table with *H* and *L*



(b) Gate block diagram

x	y	z
0	0	0
0	1	0
1	0	0
1	1	1

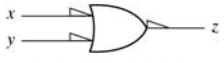
(c) Truth table for positive logic



(d) Positive logic AND gate

x	y	z
1	1	1
1	0	1
0	1	1
0	0	0

(e) Truth table for negative logic



(f) Negative logic OR gate

Fig. 2-10 Demonstration of positive and negative logic

Ολοκληρωμένα Κυκλώματα

- *Επίπεδα ολοκλήρωσης*
  - SSI
  - MSI
  - LSI
  - VLSI
- *Οικογένειες ψηφιακής λογικής*
  - TTL
  - ECL
  - CMOS
- *Χαρακτηριστικά*
  - ικανότητα οδήγησης (Fan-Out)
  - κατανάλωση ισχύος (power dissipation)
  - καθυστέρηση διάδοσης (propagation delay)
  - περιθώριο θορύβου (noise margin)

Παράρτημα - Απλοποίηση Λογικής

- Δεδομένης μιας έκφρασης Boole θέλουμε να ελαχιστοποιήσουμε :
  - αριθμό όρων
  - αριθμό μεταβλητών
  - αριθμό πυλών
- πχ η έκφραση :
  - f = B'C'D' + B'D + ABC + BC'D + ABD' + (A+B+D')'
- έχει 6 όρους, 4 μεταβλητές, 17 σύμβολα
- μπορεί να ελαχιστοποιηθεί ως :
  - f = B'C' + B'D + AB + C'D
- όπου έχουμε 4 όρους, 4 μεταβλητές και 8 σύμβολα

Παράδειγμα Ελαχιστοποίησης

f = B'C'D' + B'D + ABC + BC'D + ABD' + (A+B+D')'

σημαντικό θεώρημα : x + xy = x

f = B'C'D' + B'D + ABC + BC'D + ABD' + A'B'D

= B'C'D' + ABC + BC'D + ABD' + B'D

= B'C'D' + ABC + BC'D + ABD' + B'D + B'DC'

= B'C'(D' + D) + ABC + BC'D + ABD' + B'D

= B'C' + B'C'D + ABC + BC'D + ABD' + B'D

= B'C' + C'D (B + B') + ABC + ABD' + B'D

= B'C' + B'D + C'D + ABC + ABD'

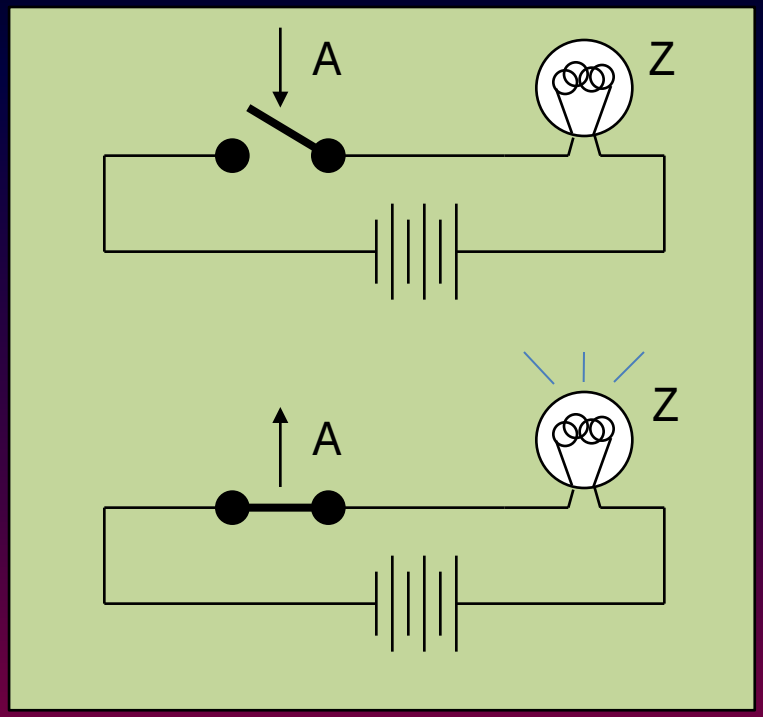
= B'C' + B'D + C'D + ABC'D + ABCD + ABCD' + ABCD' + ABC'D'

= B'C' + B'D + C'D + ABD(C+C') + ABD'(C+C')

= B'C' + B'D + C'D + ABD + ABD'

= B'C' + B'D + C'D + AB

Λογικές (Δυαδικές) Συναρτήσεις

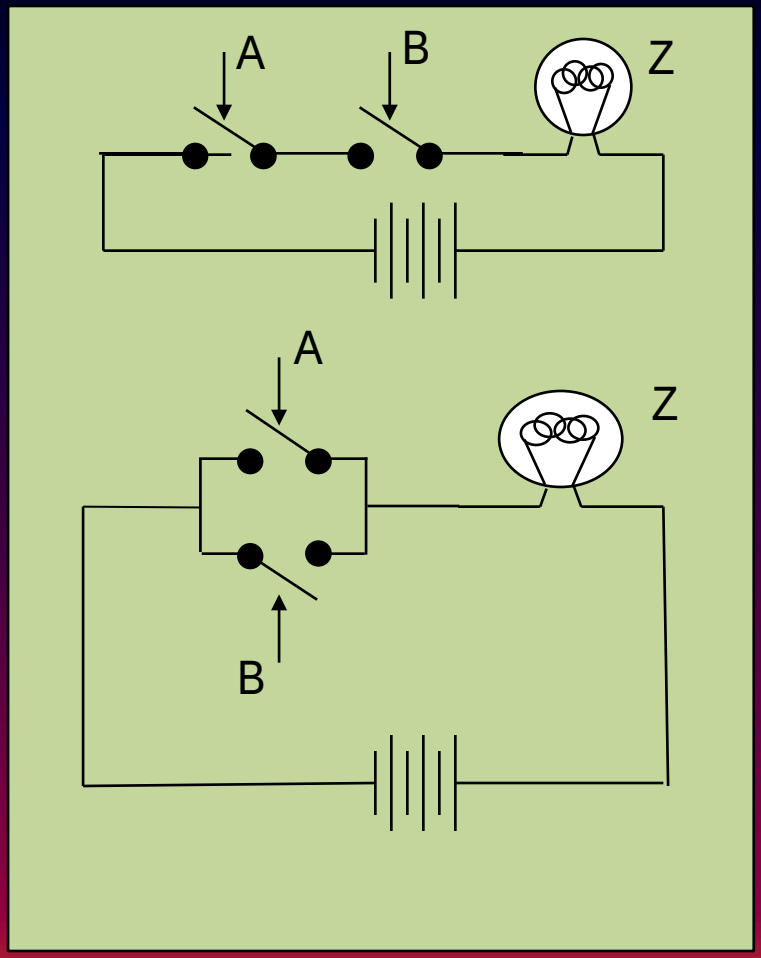


- Υποθέστε
  - τη δυαδική μεταβλητή A
    - A=0 => διακόπτης ανοικτός
    - A=1=> διακόπτης κλειστός
  - και τη Z
    - Z=0 => Οχι φώς
    - Z=1 => Φως

Η Z είναι ανεξάρτητη από τη A ή όχι ?

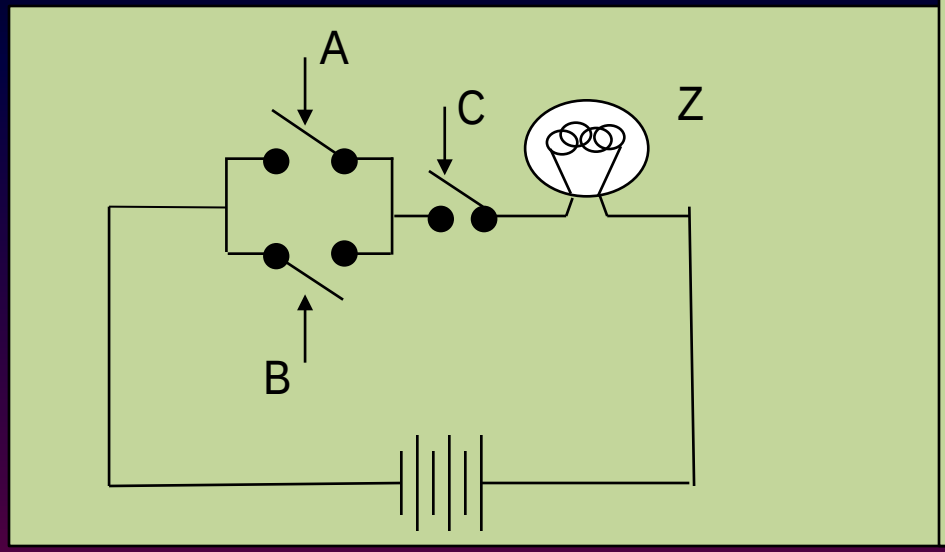
Η Z είναι μια συνάρτηση της A, ισχύει Z = f(A) = A!

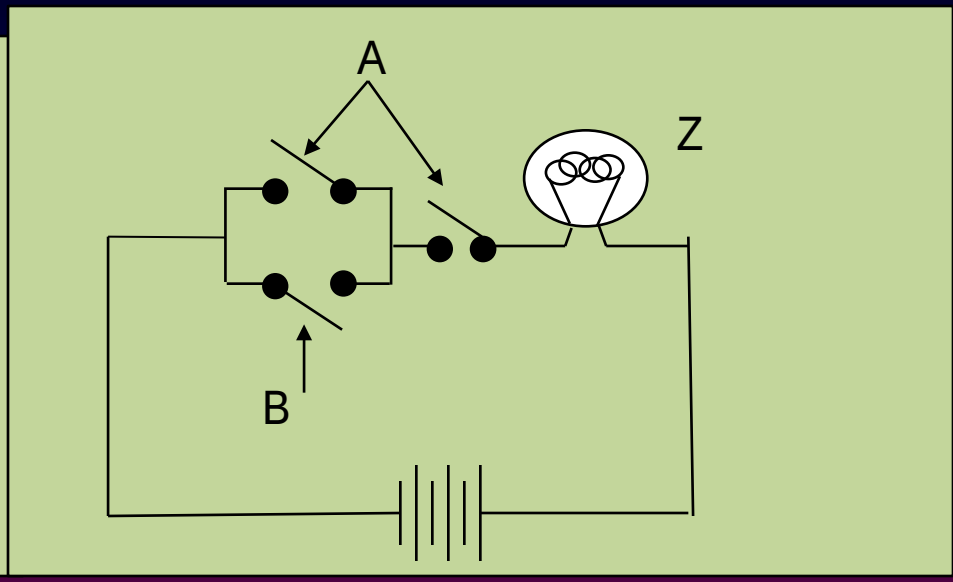
Απλές λογικές συναρτήσεις 2 δυαδικών μεταβλητών



Οι συναρτήσεις αυτές ονομάζονται και λογικές γιατί τις χρησιμοποιούμε στη λογική των προτάσεων που συντάσσουμε καθημερινά.

Σύνθεση συναρτήσεων





Οσο πιο πολύπλοκο γίνεται το σχέδιο, τόσο πιο προφανές γίνεται ότι χρειαζόμαστε κάποιο τυπικό τρόπο, για να βρούμε πότε το Z θα γίνει 1.

Χρειαζόμαστε συνεπώς μια άλγεβρα !

Τι είναι μια άλγεβρα ?

- Μια μαθηματική δομή :
  - Σύνολο ψηφίων & αναπαραστάσεων
  - Σύνολο τελεστών & προτεραιότητες
  - Αξιώματα
  - Θεωρήματα
  - Συναρτήσεις

- Αλγεβρα των φυσικών αριθμών
  - Ψηφίο $\in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, Αναπαραστάσεις = ∞
  - Τελεστές : +, -, *, / Προτεραιότητες : {,[,(, /, *, -, +
  - Αξιώματα : ουδετερότητα του 0 στη +, του 1 στον *, προσεταιριστικότητα του +, επιμεριστικότητα του * ως προς το +, ...
  - Θεωρήματα : x+x = 2x
  - Συναρτήσεις f(x,y) = 3x+5y

Αλγεβρα Boole

- Το 1854 (!!!) ο Αγγλος μαθηματικός George Boole εισήγαγε μια άλγεβρα δύο τιμών : αλήθεια και ψέμα.
- Προτάθηκε για το λογισμό της αλήθειας ή του ψέμματος προτάσεων
- Π.χ.
  - Δεχόμαστε τη πρόταση "Οσοι κάνουν μπάνιο κάθε μέρα έχουν πολλά λεφτά ή δε τους αρέσει η βρωμιά" ως αληθή.
  - Δεχόμαστε και τη πρόταση "Κανένας μηχανικός υπολογιστών δεν έχει πολλά λεφτά" ως αληθή.
  - Εστω η πρόταση "Αν είσαι μηχανικός υπολογιστών και σου αρέσει η βρωμιά, κάνεις μπάνιο κάθε μέρα".
  - Τι μπορούμε να αποφανθούμε για την αλήθεια ή το ψέμμα της ?

Shannon

- Σχεδόν ένα αιώνα αργότερα (1938), ο Claude Shannon διαπιστώνει ότι απλά αντιστοιχίζοντας τις "αλήθεια" και "ψέμμα" στο "κλειστός διακόπτης" και "ανοικτός διακόπτης", μπορεί να εφαρμόσει τα όσα ανέπτυξε ο Boole και σε κυκλώματα διακοπτών.
- Ετσι η άλγεβρα Boole έγινε "switching algebra".
- Εμείς θα πάμε απλά ένα βήμα πιο πέρα. Αντί για "κλειστός διακόπτης" και "ανοικτός διακόπτης" θα χρησιμοποιούμε τις τιμές δυαδικών μεταβλητών "1" και "0" αντίστοιχα.
- Ετσι έχουμε τη δυαδική άλγεβρα (λογική άλγεβρα).

Κι ο Huntington ?

Το 1904 (ενδιάμεσα δηλαδή από το Boole & το Shannon) ένας άλλος μαθηματικός, ο Edward Huntington

(

1919 : president of the Mathematical Association of America

1941 : vice-president of the American Association for the Advancement of Science

1913 : member of the American Academy of Arts and Sciences

1933 : member of the American Philosophical Society in 1933.

)

διατύπωσε μια σειρά αξιωμάτων που ισχύουν όχι μόνο για την άλγεβρα του Boole, αλλά για κάθε *αλγεβρική δομή*.

Διάφορα θεωρήματα αποδεικνύονται συνεπώς με τον ίδιο τρόπο όπως για κάθε άλγεβρα λόγω αυτής της ιδιότητας.

Αξιώματα

- Δίτιμη άλγεβρα. Κάθε στοιχείο της $X \in \{0,1\}$
- Υπαρξη συμπληρώματος (α')
  - Αν $\alpha = 0 \Rightarrow \alpha' = 1$, Αν $\alpha = 1 \Rightarrow \alpha' = 0$
- Δύο λογικές πράξεις :
  - Λογική σύζευξη (and / και) : • (το σύμβολο μπορεί να παραλείπεται σε απλή παράταξη μεταβλητών)
  - Λογική διάζευξη (or / ή) : +
- Οι λογικές πράξεις ακολουθούν τους εξής κανόνες :
  - $0 \bullet 0 = 0$, $1 + 1 = 1$
  - $1 \bullet 1 = 1$, $0 + 0 = 0$
  - $0 \bullet 1 = 1 \bullet 0 = 0$, $1 + 0 = 0 + 1 = 1$
- Σε μία σύνθετη συνάρτηση των δύο λογικών πράξεων, οι προτεραιότητες είναι : {, [, (, ', •, +

Θεωρήματα μιας μεταβλητής

(Απόδειξη με εξέταση όλων των δυνατών τιμών της μεταβλητής)

• X + 0 = X,	X • 1 = X	(Ουδέτερα στοιχεία)
• X + 1 = 1,	X • 0 = 0	(Απορροφητικά στοιχεία)
• X + X = X,	X • X = X	(Αυτοαπορρόφησης)
• (X')' = X		(Διπλοαντιστροφής)
• X + X' = 1	X • X' = 0	(Συμπληρωματικά στοιχεία)

Θεωρήματα δύο και τριών μεταβλητών (1/2)

(Απόδειξη με εξέταση όλων των δυνατών τιμών των δύο μερών)

- $X + Y = Y + X$, $X \bullet Y = Y \bullet X$ (Αντιμεταθετική)
- $(X + Y) + Z = X + (Y + Z)$, $(X \bullet Y) \bullet Z = X \bullet (Y \bullet Z)$ (Προσεταιριστική)
  - Συμπέρασμα : *Σε λογικά γινόμενα ή λογικά αθροίσματα, η χρήση παρενθέσεων είναι προαιρετική. Για παράδειγμα μπορώ να γράφω w + x + y + z αφού όπως κι αν υπολογιστεί αυτή η έκφραση θα πάρω το ίδιο λογικό αποτέλεσμα.*
- $X \bullet Y + X \bullet Z = X \bullet (Y + Z)$, $(X + Y) \bullet (X + Z) = X + (Y \bullet Z)$ (Επιμεριστική)
  - Προσοχή :
    1. Η έκφραση $X \bullet Y + X \bullet Z$ είναι ισοδύναμη με την $(X \bullet Y) + (X \bullet Z)$. Θυμηθείτε τη προτεραιότητα τελεστών.
    2. Ισχύουν και οι 2 επιμερισμοί. Στην άλγεβρα των αριθμών ισχύει μόνο του x ως προς το +.
  - Παράδειγμα :
    - $(W + Y) \bullet (X + Z) \bullet (V + Y) \bullet (W + Z) \bullet (X + Y) \bullet (V + Z) =$
      $[W + (Y \bullet Z)] \bullet [X + (Y \bullet Z)] \bullet [V + (Y \bullet Z)] = (Y \bullet Z) + (V \bullet X \bullet W)$

Θεωρήματα δύο και τριών μεταβλητών (2/2)

(Απόδειξη με εξέταση όλων των δυνατών τιμών των δύο μερών)

- $X + X \bullet Y = X$, $X \bullet (X + Y) = X$ (Κάλυψης)
- $X \bullet Y + X \bullet Y' = X$ $(X + Y) \bullet (X + Y') = X$ (Συνδυασμών)
  - Παράδειγμα : $X \bullet Y \bullet Z' + X \bullet Y' \bullet Z' + X \bullet Y \bullet Z + X \bullet Y' \bullet Z = X \bullet Z' + X \bullet Z = X$
  - *Συμπέρασμα : Σε ένα άθροισμα γινομένων, που κάθε γινόμενο έχει κ μεταβλητές, αν υπάρχουν όλοι οι συνδυασμοί τιμών των κ-1 μεταβλητών τότε μπορούν να διαγραφούν από την έκφραση.*
- $X \bullet Y + X' \bullet Z + Y \bullet Z = X \bullet Y + X' \bullet Z$,
- $(X + Y) \bullet (X' + Z) \bullet (Y + Z) = (X + Y) \bullet (X' + Z)$ (Συναίνεσης)

Θεωρήματα γενικευμένου αριθμού μεταβλητών

- $X + X + \ldots + X = X$, $X \bullet X \bullet \ldots \bullet X = X$ (Αυτοαπορρόφησης)

- $(X_1 + X_2 + \ldots + X_n)' = X_1' \bullet X_2' \bullet \ldots \bullet X_n'$
- $(X_1 \bullet X_2 \bullet \ldots \bullet X_n)' = X'_1 + X'_2 + \ldots + X'_n$ (Θεωρήματα De Morgan)

- $[F(X_1, X_2, \ldots, X_n, +, \bullet)]' = F(X'_1, X'_2, \ldots, X'_n, \bullet, +)$ (Γενικευμένο θεώρημα De Morgan)
  - Παράδειγμα : Δίδεται η $F = (W' \bullet X) + (X \bullet Y) + [W \bullet (X' + Z')]$.
  - Τότε $F' = ((W')' + X') \bullet (X' + Y') \bullet [W' + (X \bullet Z)] =$
    $= (W + X') \bullet (X \bullet Y)' \bullet [W' + (X \bullet Z)]$

- $F(X_1, X_2, \ldots, X_n) = X_1 \bullet F(1, X_2, \ldots, X_n) + X_1' \bullet F(0, X_2, \ldots, X_n)$
- $F(X_1, X_2, \ldots, X_n) = [X_1 + F(0, X_2, \ldots, X_n)] \bullet [X_1' + F(1, X_2, \ldots, X_n)]$ (Θεωρήματα Shannon)
  - Παράδειγμα : Δίδεται η $F(X, W, Z) = X + W \bullet Z$.
  - Τότε $F(0, W, Z) = W \bullet Z$ και $F(1, W, Z) = 1$.
  - Αρα $F = X \bullet 1 + X' \bullet W \bullet Z$ καθώς και $F = (X + W \bullet Z) \bullet (X' + 1)$

Το μεταθεώρημα του δυϊσμού

- Ενα μεταθεώρημα είναι ένα θεώρημα για άλλα θεωρήματα. Είναι δηλαδή ένα πολύ βασικό και γενικευμένο θεώρημα.

- Μέχρι ώρας είδαμε διάφορα θεωρήματα. Ολα μα όλα παρουσιάστηκαν σε ζεύγη. Τι υποκρύπτει αυτή η δυαδική συμπεριφορά ?

- **Δυισμός :**

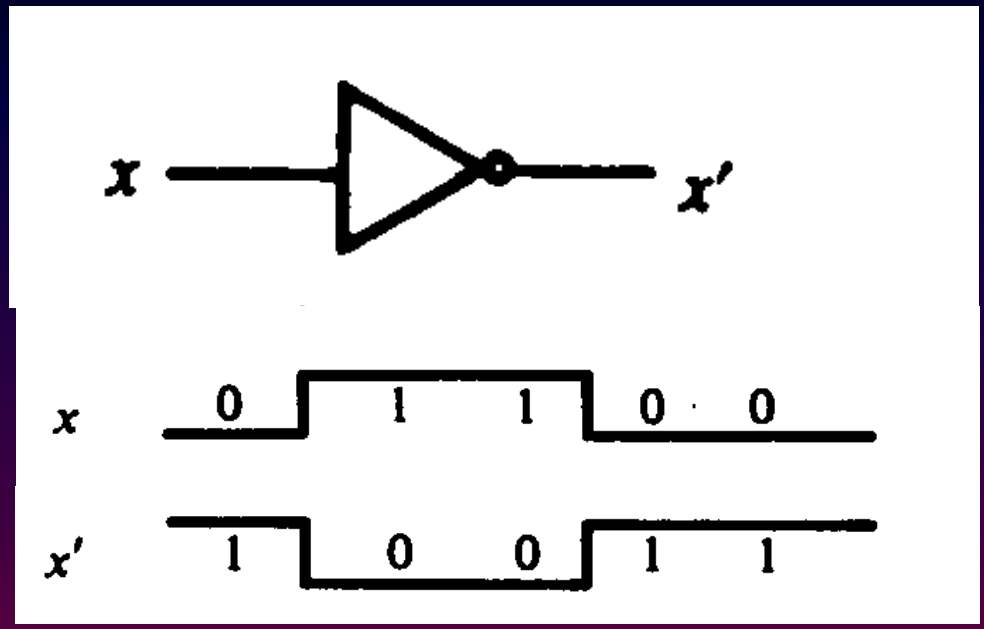
  Η αλήθεια ή το ψέμμα μιας ταυτότητας ή ενός θεωρήματος που ασχολείται με δυαδικές μεταβλητές και λογικές συναρτήσεις δεν αλλάζει αν ανταλλάξουμε το 0 με το 1 και το + με το •.

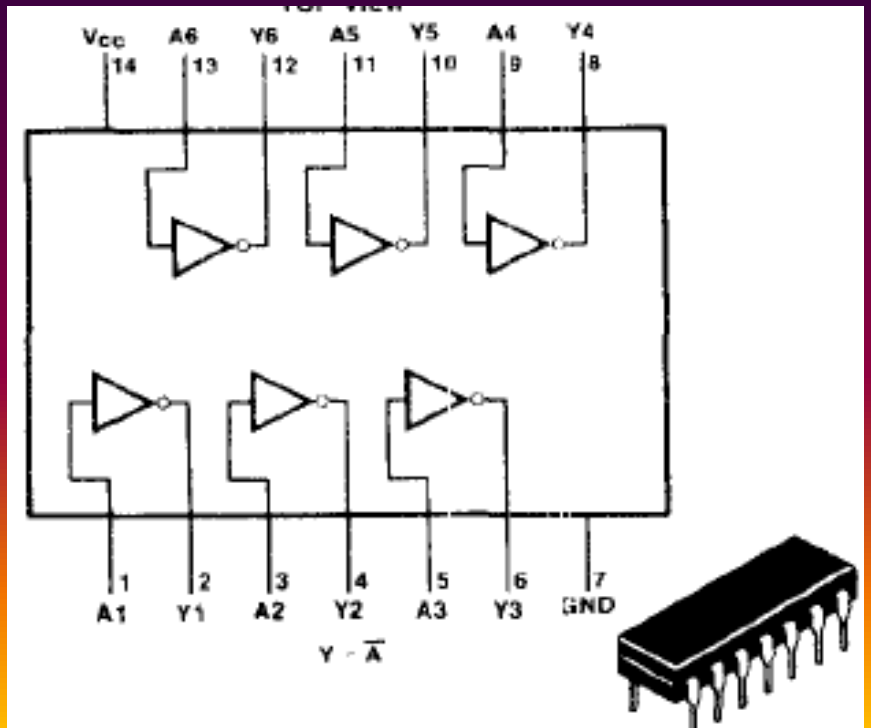
Γιατί τόση θεωρία ?

- Οσα είδαμε έχουν άμεση εφαρμογή στο πραγματικό κόσμο.
- Υπάρχουν έτοιμοι ψηφιακοί σχεδιασμοί οι οποίοι υλοποιούν τις βασικές λογικές συναρτήσεις (φυσικά υπάρχουν και άλλοι που υλοποιούν άλλες ενδεχόμενα πιο πολύπλοκες συναρτήσεις).
- Οι σχεδιασμοί αυτοί ονομάζονται ψηφιακές ***πύλες***.
- Θα εισάγουμε :
  - κάποια σχηματικά για την απεικόνιση αυτών.
  - Ενα πίνακα που δείχνει τη λογική συνάρτηση που επιτελείται από το συγκεκριμένο ψηφιακό κύκλωμα. Ο πίνακας αυτός είναι ο ***πίνακας αληθείας*** της συνάρτησης, άρα και του ψηφιακού σχεδιασμού.

Ο αντιστροφέας

- Παράγει το συμπλήρωμα μιας δυαδικής μεταβλητής.
- Μπορούμε αντί για το πλήρες σχηματικό του αντιστροφέα, να χρησιμοποιούμε μόνο το κύκλο.
- Είναι διαθέσιμο ως ψηφιακό κύκλωμα σε 6άδες, εντός ενός ολοκληρωμένου με κωδικό 7404.

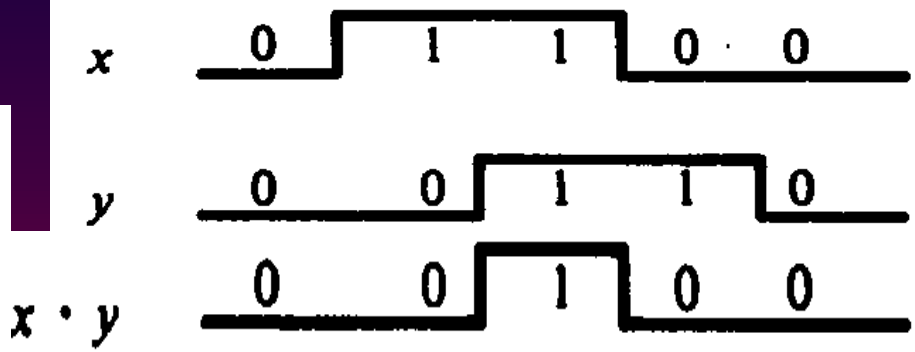


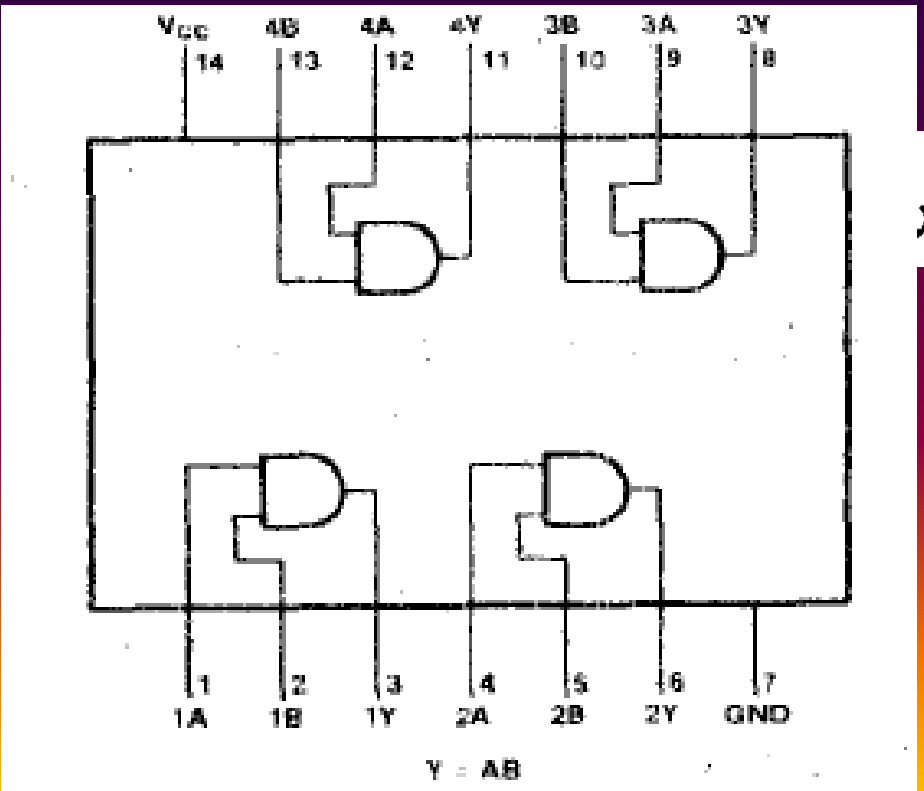


x	x′
0	1
1	0

Η πύλη AND δύο μεταβλητών

- Εκτελεί το λογικό ΚΑΙ των δύο μεταβλητών.
- Είναι διαθέσιμο ως ψηφιακό κύκλωμα σε 4άδες, εντός ενός ολοκληρωμένου με κωδικό 7408.

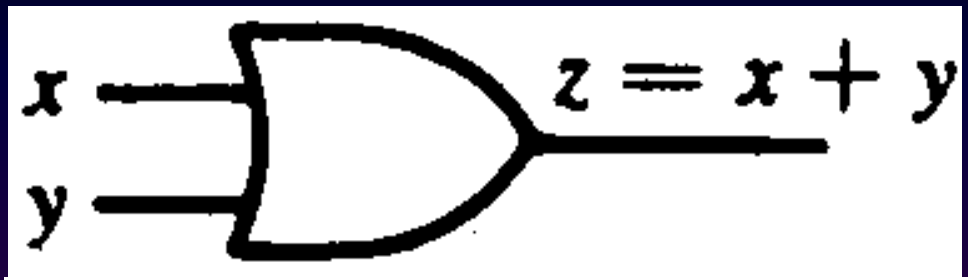


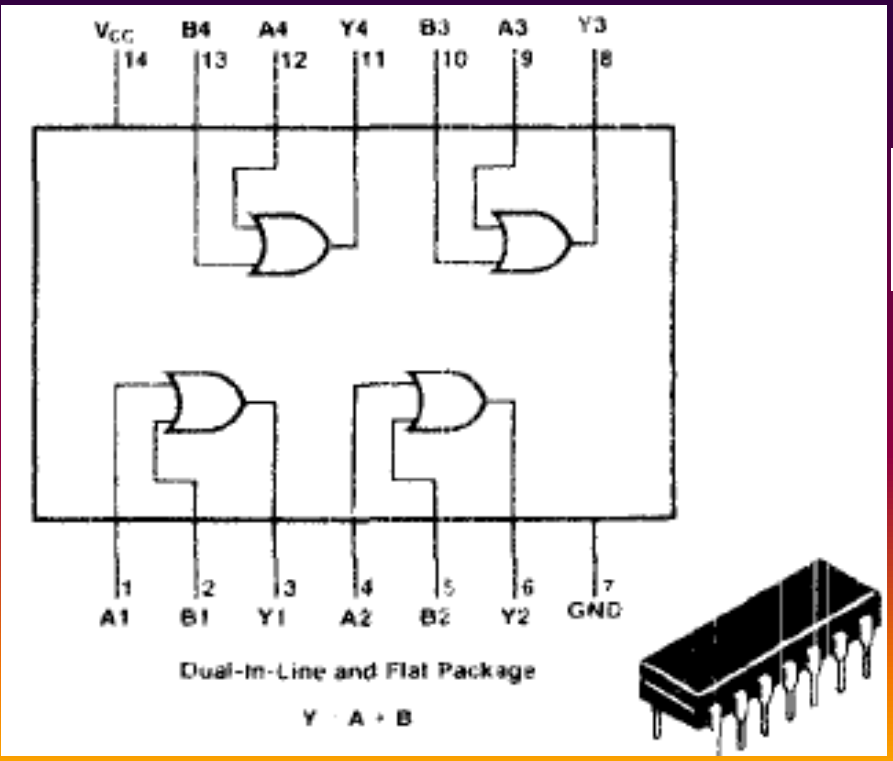


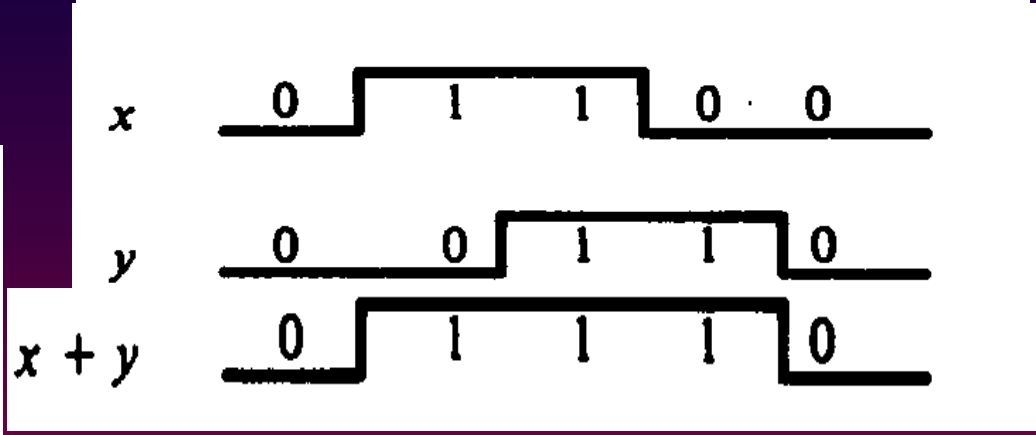
x	y	x · y
0	0	0
0	1	0
1	0	0
1	1	1

Η πύλη OR δύο μεταβλητών

- Εκτελεί το λογικό Η των δύο εισόδων.
- Είναι διαθέσιμο ως ψηφιακό κύκλωμα σε 4άδες, εντός ενός ολοκληρωμένου με κωδικό 7432.



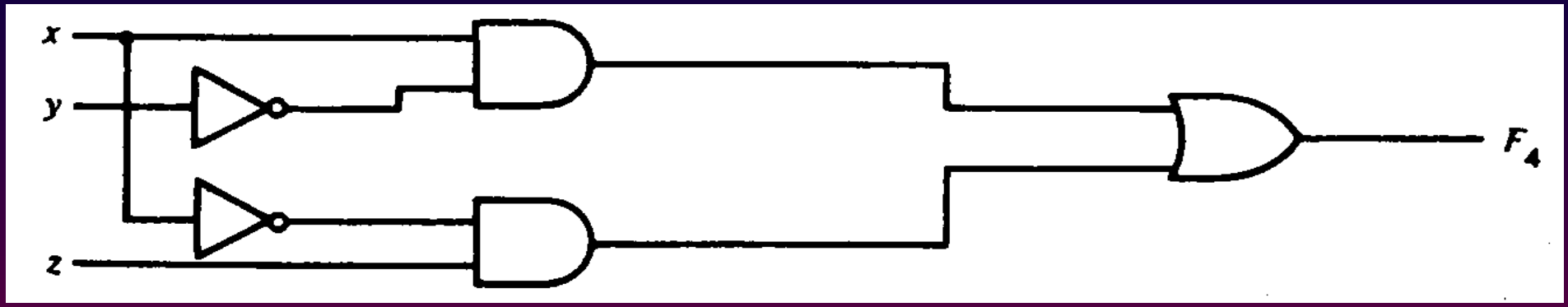




x	y	x + y
0	0	0
0	1	1
1	0	1
1	1	1

Λογικό διάγραμμα

- Η χρήση των σχηματικών των διαφόρων πυλών και η διασύνδεσή τους οδηγεί σε ένα **λογικό διάγραμμα** που περιγράφει κάποια πιο σύνθετη συνάρτηση.
- Για παράδειγμα :



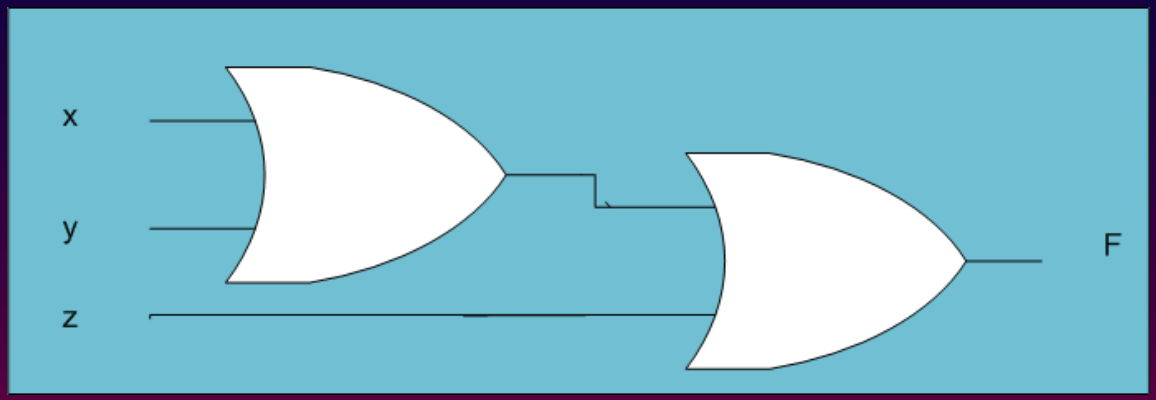
- Το παραπάνω είναι ένα διάγραμμα που περιγράφει τη συνάρτηση :

$$F_4(x, y, z) = x' \bullet z + y' \bullet x$$

- Ποιο είναι το λογικό διάγραμμα της G(X, W, Y, Z) = [X' • Y + X] • (Z + W') ;

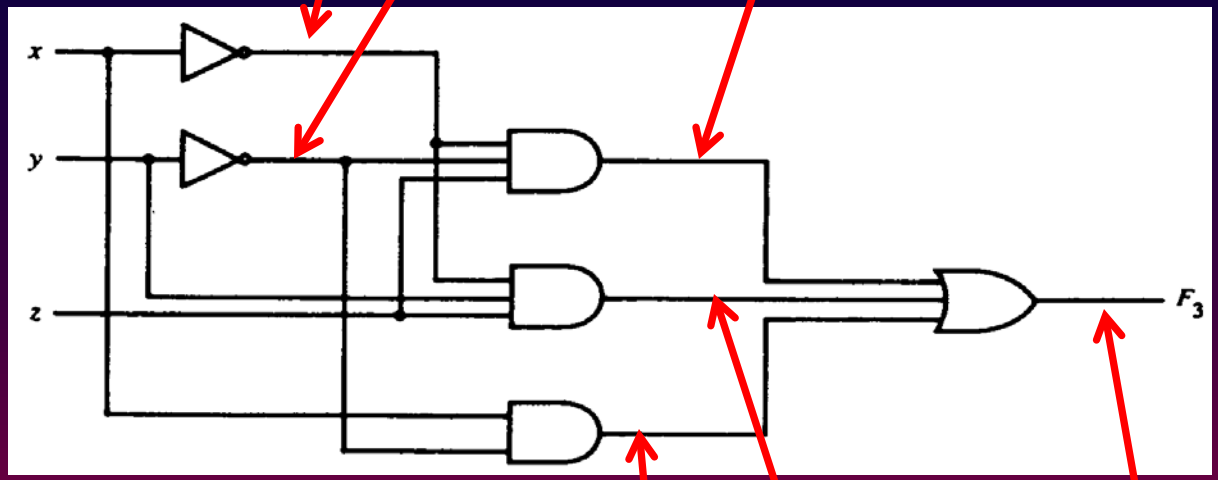
Επέκταση σε περισσότερες είσοδες

- Αν ήθελα το λογικό διάγραμμα της F(x, y, z) = x + y + z πως θα το έφτιαχνα ?
- Από τη θεωρία μου γνωρίζω ότι F(x, y, z) = x + y + z = (x + y) + z και συνεπώς θα μπορούσα να χρησιμοποιήσω κάτι τέτοιο :



- Μήπως υπάρχουν έτοιμες και πύλες περισσότερων εισόδων ?
- Φυσικά ! Αυτό όμως δε σημαίνει ότι υπάρχουν οσωνδήποτε εισόδων.
- Κι εδώ πρέπει να διαχωρίσουμε το πραγματικό από τον ιδεατό κόσμο :
  - Στον ιδεατό κόσμο μπορούμε να φτιάχνουμε λογικά διαγράμματα με πύλες όσων εισόδων θέλουμε.
  - Στη πράξη αυτά θα υλοποιηθούν με όσα πραγματικά υπάρχουν.

Ποια συνάρτηση επιτελεί αυτό το κύκλωμα ?



X'

Y'

X' • Y' • Z

X • Y'

X' • Y • Z

X • Y' + X' • Y • Z + X' • Y' • Z

Ποιες άλλες συναρτήσεις και πύλες υπάρχουν ?

- Πόσες διαφορετικές συναρτήσεις των ν μεταβλητών υπάρχουν ?
  - Μια συνάρτηση ν μεταβλητών, έχει 2^{ν} πιθανές τιμές εισόδου.
  - Διαφορετική συνάρτηση ⇔ διαφορετική έξοδος έστω και για κάποιον συνδυασμό εισόδου. Αφού η έξοδός μου για κάθε συνδυασμό εισόδου μπορεί να είναι 0 ή 1 θα υπάρχουν $2^{2^{\nu}}$ συναρτήσεις.
- Για ν =2 ποιες είναι οι συναρτήσεις που υπάρχουν και ποιες από αυτές είναι χρήσιμες ?

x	y	F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
Σύμβολο τελεστή			·	/		/		⊕	+	↓	⊙	′	⊂	′	⊃	↑	

1. Δυο σταθερές 0, 1.
2. Τέσσερις unary συμπληρώματος/μεταφοράς.
3. Δέκα συναρτήσεις με δυαδικούς τελεστές.

Μας ενδιαφέρουν επίσης οι :

x	y	F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
Σύμβολο τελεστή			$\cdot$	/		/		$\oplus$	+	$\downarrow$	$\odot$	$'$	$\subset$	$'$	$\supset$	$\uparrow$	

- F_{14} που είναι συμπληρωματική της AND. Θα τη λέμε NAND (not – AND).
- F_8 που είναι συμπληρωματική της OR. Θα τη λέμε NOR (not – OR).
- F_6 που μας δίνει 1 μόνο όταν μόνο 1 είσοδος είναι στο λογικό 1 (για περισσότερες εισόδους, όταν ο αριθμός των 1 στις εισόδους είναι περιττός). Θα την ονομάζουμε αποκλειστικό-OR (eXclusive-OR) – XOR.
- F_9 που μας δίνει 1 μόνο όταν μόνο 0 ή 2 είσοδοι είναι στο λογικό 1 (για περισσότερες εισόδους όταν ο αριθμός των 1 στις εισόδους είναι άρτιος). Θα την ονομάζουμε συνάρτηση ισοδυναμίας (not eXclusive-OR) – XNOR.

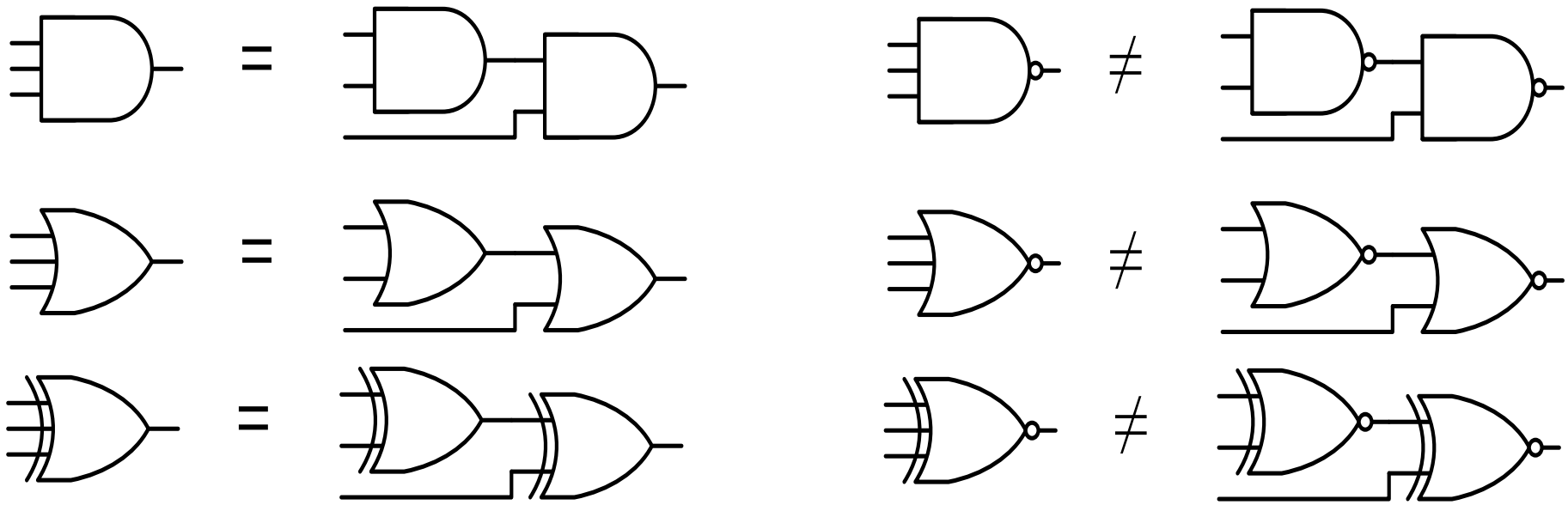
Ετσι έχουμε για τα λογικά μας διαγράμματα : (1/2)

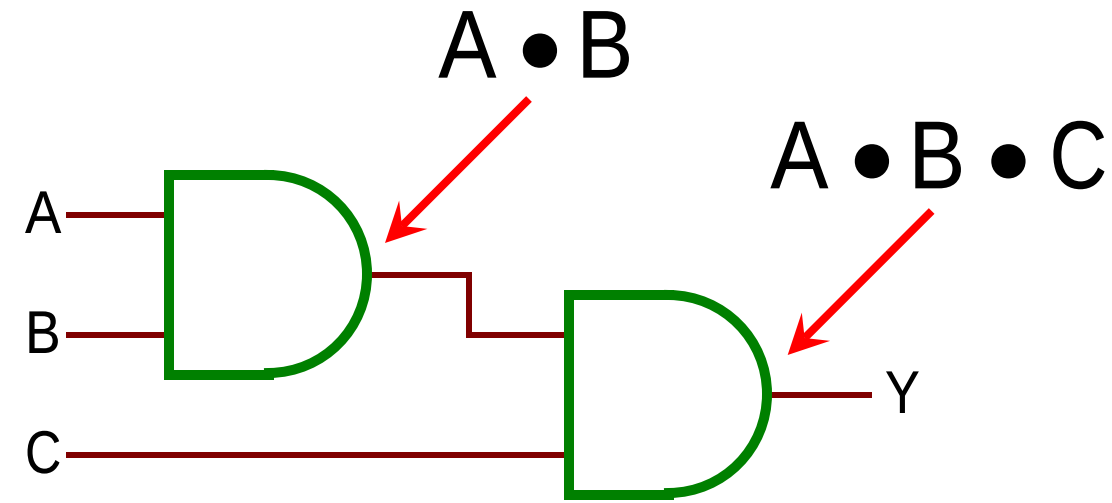
Όνομα	Γραφικό Σύμβολο	Αλγεβρική Συνάρτηση	Πίνακας Αληθείας
AND ΚΑΙ		$F = xy$	x y | F 0 0 | 0 0 1 | 0 1 0 | 0 1 1 | 1
OR Ή		$F = x + y$	x y | F 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 1
Αντιστροφέας		$F = x'$	x | F 0 | 1 1 | 0
Απομονωτής		$F = x$	x | F 0 | 0 1 | 1

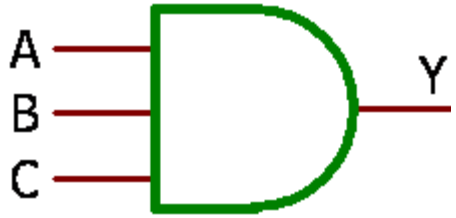
Ετσι έχουμε για τα λογικά μας διαγράμματα : (2/2)

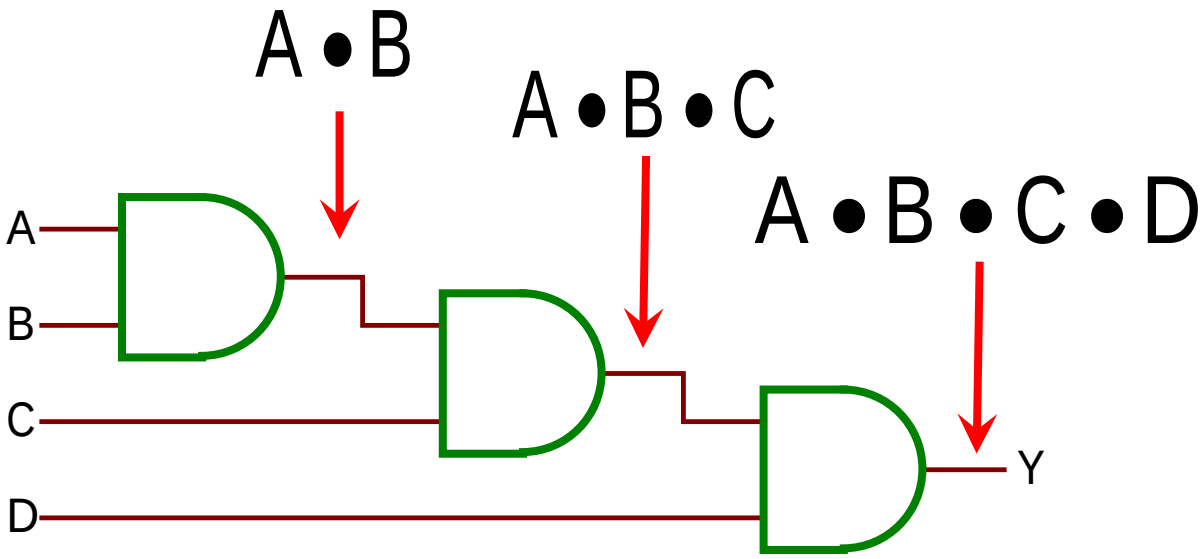
Όνομα	Γραφικό Σύμβολο	Αλγεβρική Συνάρτηση	Πίνακας Αληθείας
NAND ΟΧΙ-ΚΑΙ		$F = (xy)'$	x y | F 0 0 | 1 0 1 | 1 1 0 | 1 1 1 | 0
NOR ΟΥΤΕ		$F = (x + y)'$	x y | F 0 0 | 1 0 1 | 0 1 0 | 0 1 1 | 0
XOR Αποκλειστό - Ή		$F = xy' + x'y$ $= x \oplus y$	x y | F 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 0
Ισοδυναμία ή Αποκλειστικό -ΟΥΤΕ		$F = xy + x'y'$ $= x \odot y$	x y | F 0 0 | 1 0 1 | 0 1 0 | 0 1 1 | 1

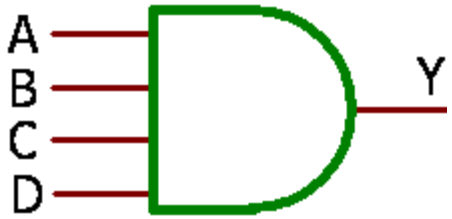
Επέκταση σε περισσότερες εισόδους ? (2)

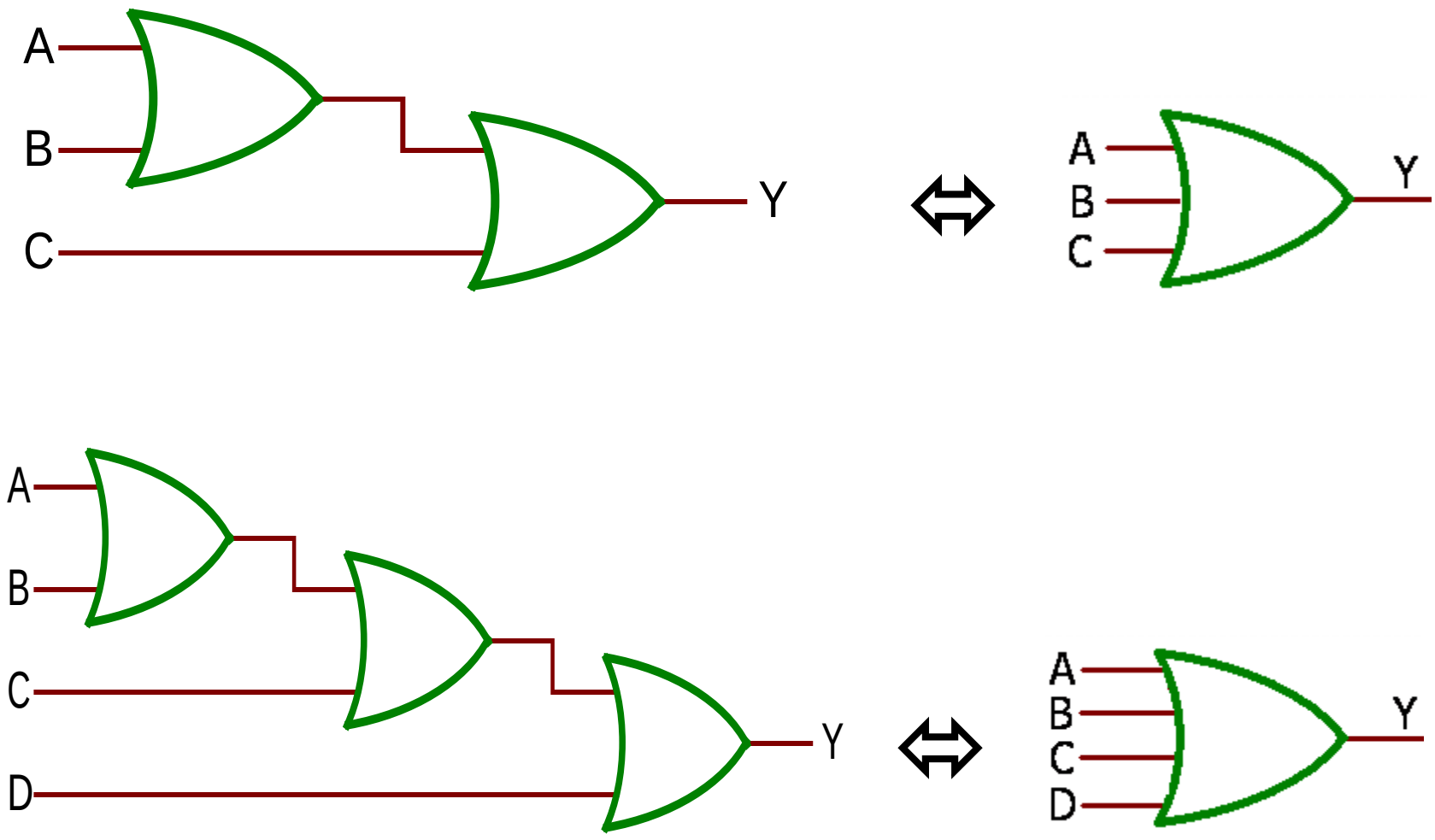


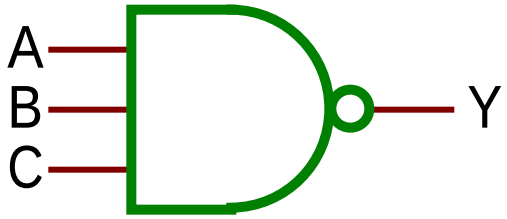

A • B
A • B • C
A
B
C
Y


A
B
C
Y


A • B
A • B • C
A • B • C • D
A
B
C
D
Y

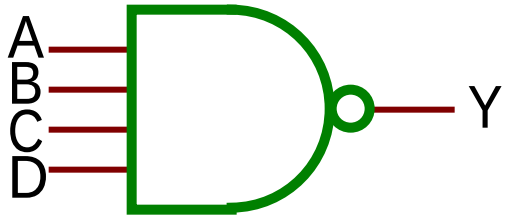

A
B
C
D
Y


A
B
C
Y
A
B
C
Y
A
B
C
D
Y
A
B
C
D
Y



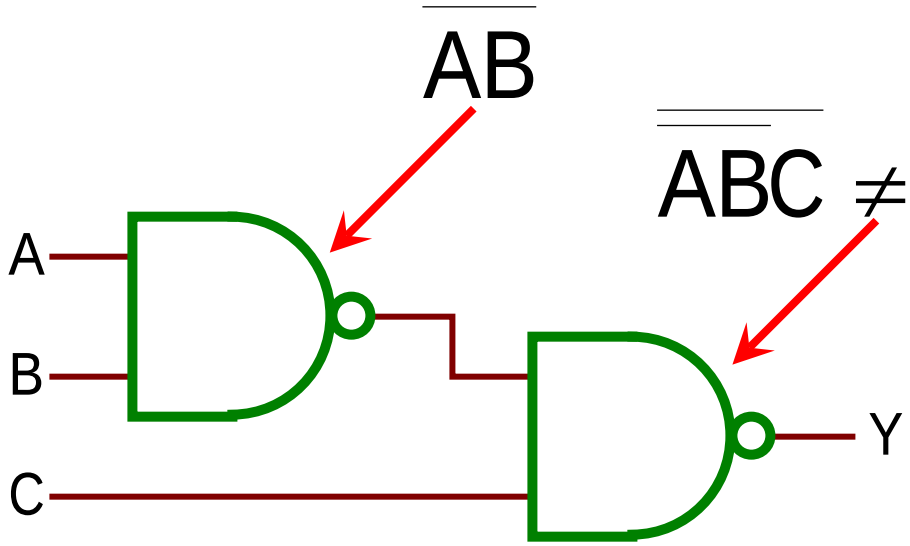
$$Y = \overline{A \bullet B \bullet C}$$

A	B	C	Y
X	X	0	1
X	0	X	1
0	X	X	1
1	1	1	0



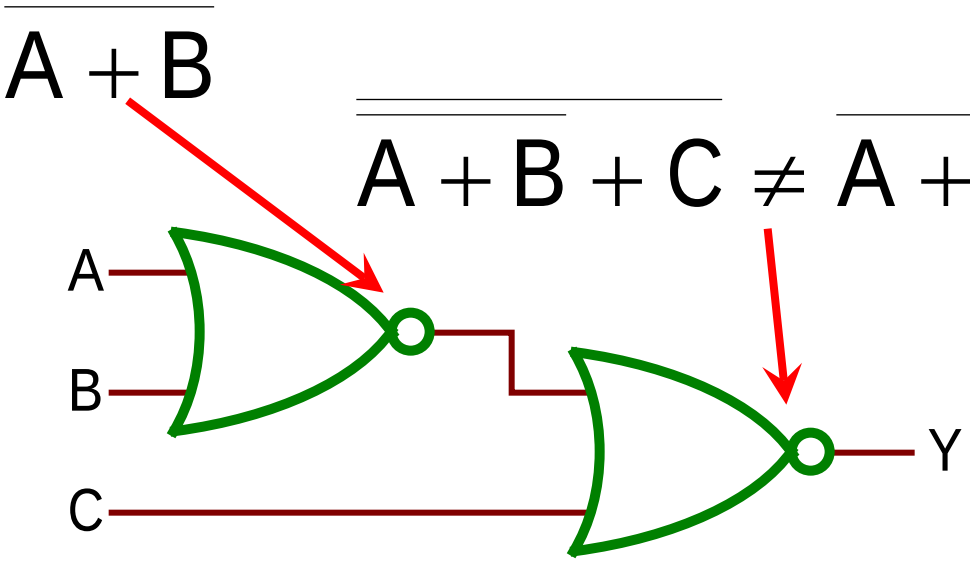
$$Y = \overline{A \bullet B \bullet C \bullet D}$$

A	B	C	D	Y
X	X	X	0	1
X	X	0	X	1
X	0	X	X	1
0	X	X	X	1
1	1	1	1	0

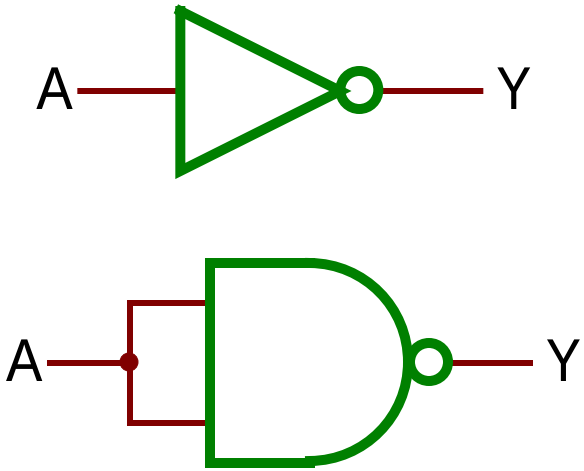


$$\overline{\overline{ABC}} \neq \overline{ABC}$$

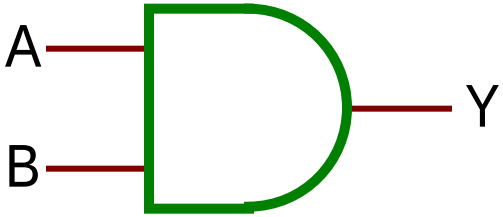
A	B	C	Y
X	X	0	1
0	X	1	0
X	0	1	0
1	1	1	1

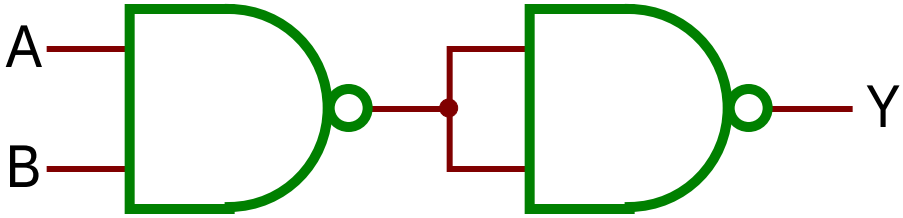


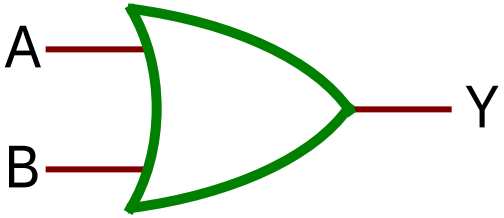
$$\overline{\overline{A+B}+C} \neq \overline{A+B+C}$$

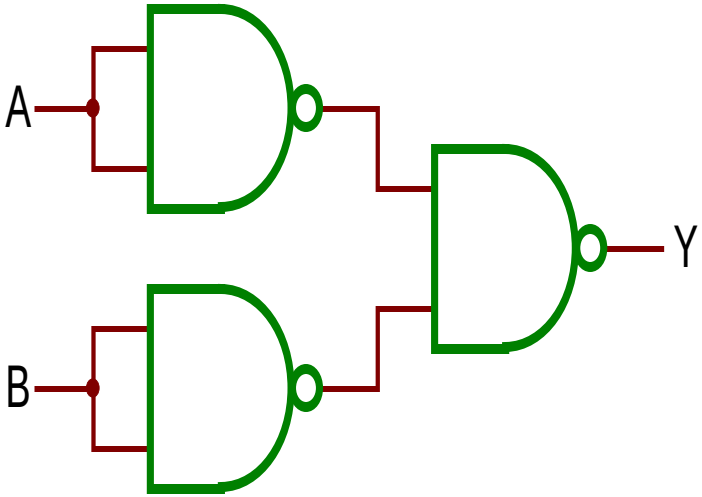

A
Y

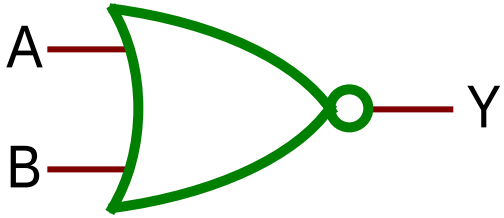
A
Y

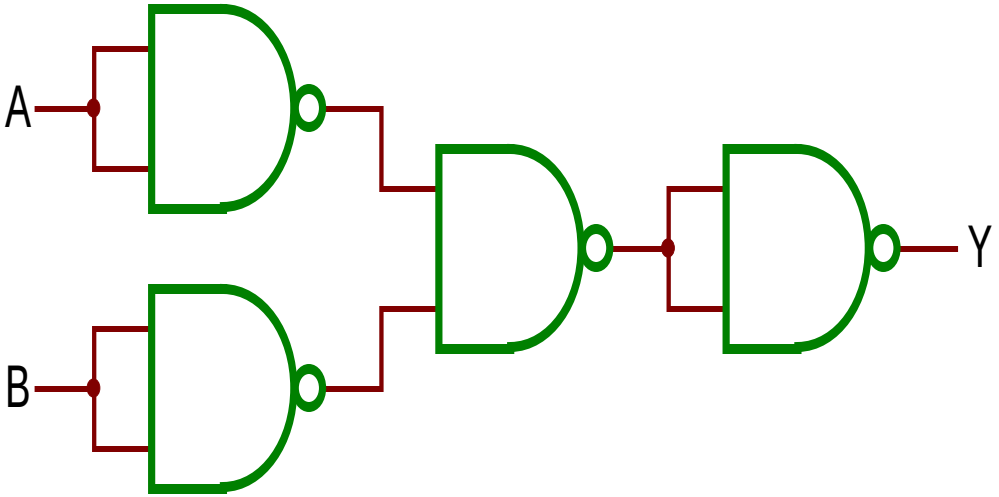

A
B
Y

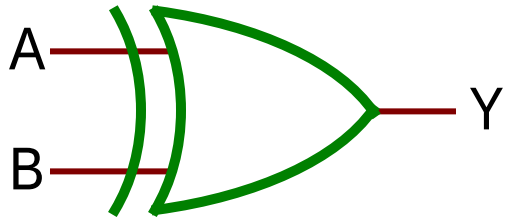

A
B
Y


A
B
Y

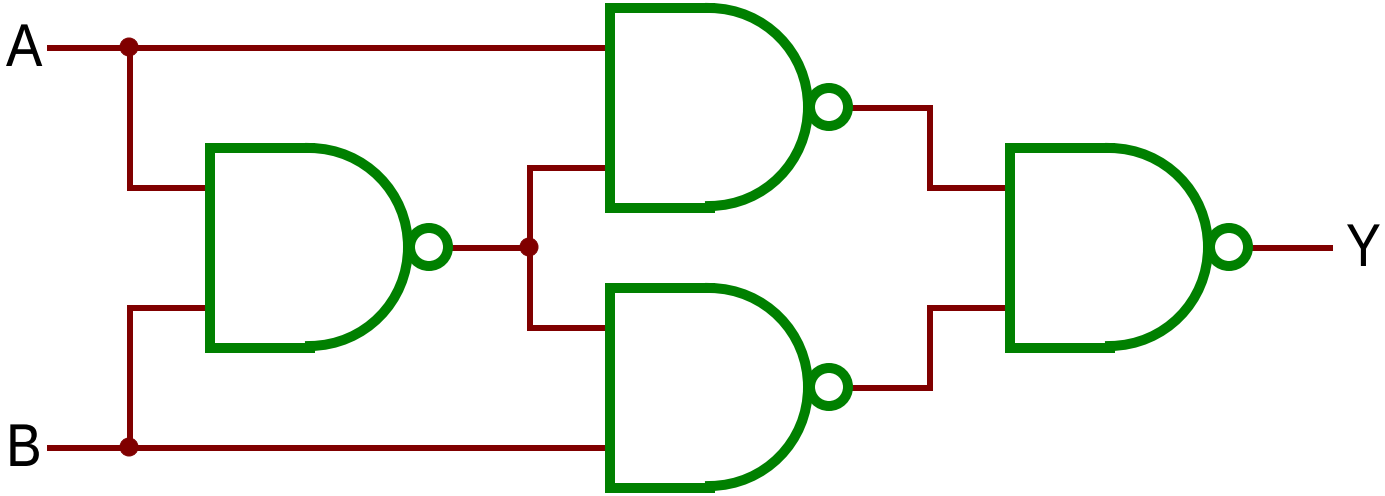

A
B
Y


A
B
Y


A
B
Y



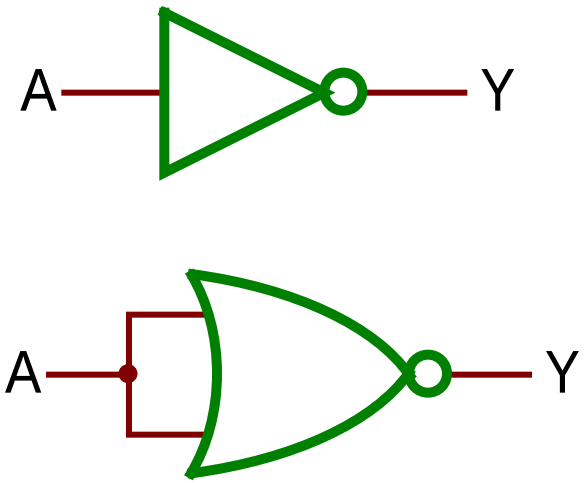
$$Y = A \oplus B$$

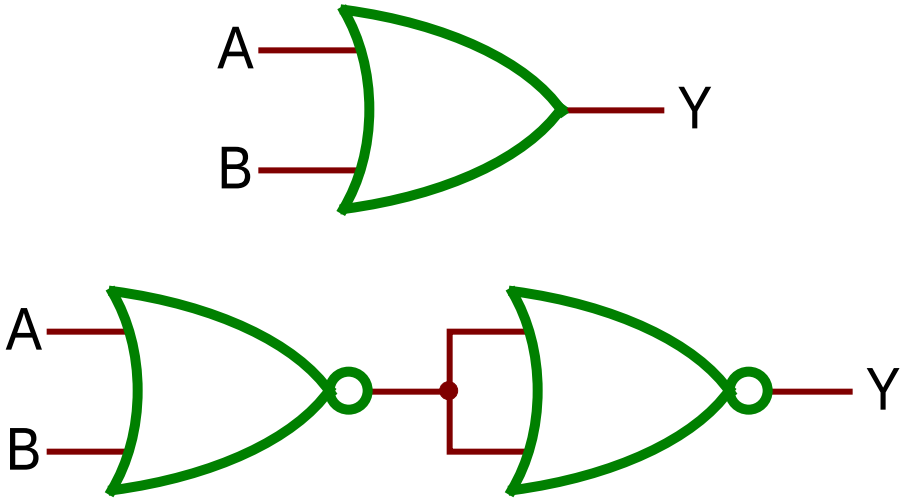


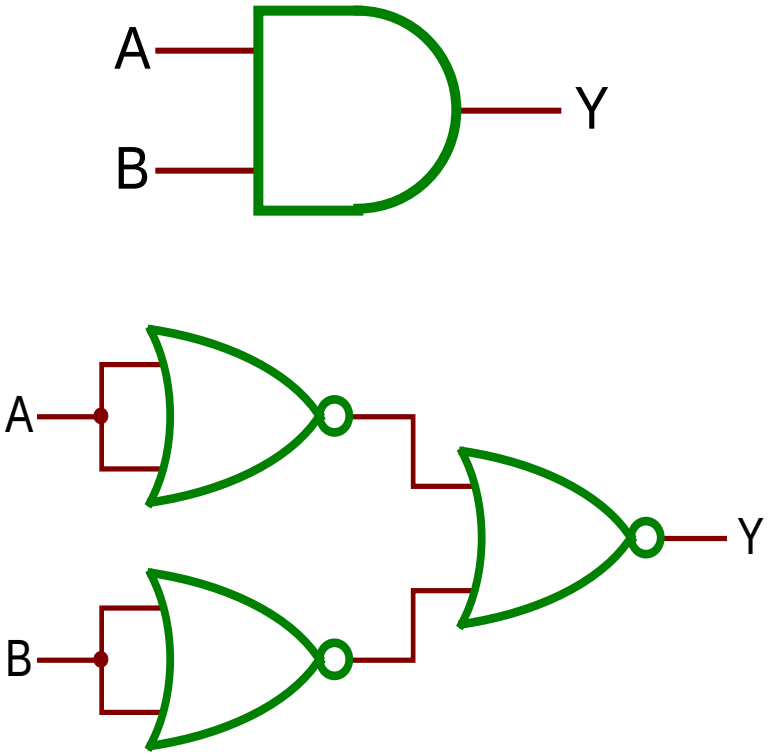
$$Y = \overline{\overline{A\overline{AB}}\,\overline{B\overline{AB}}} = A\overline{AB} + B\overline{AB} =$$

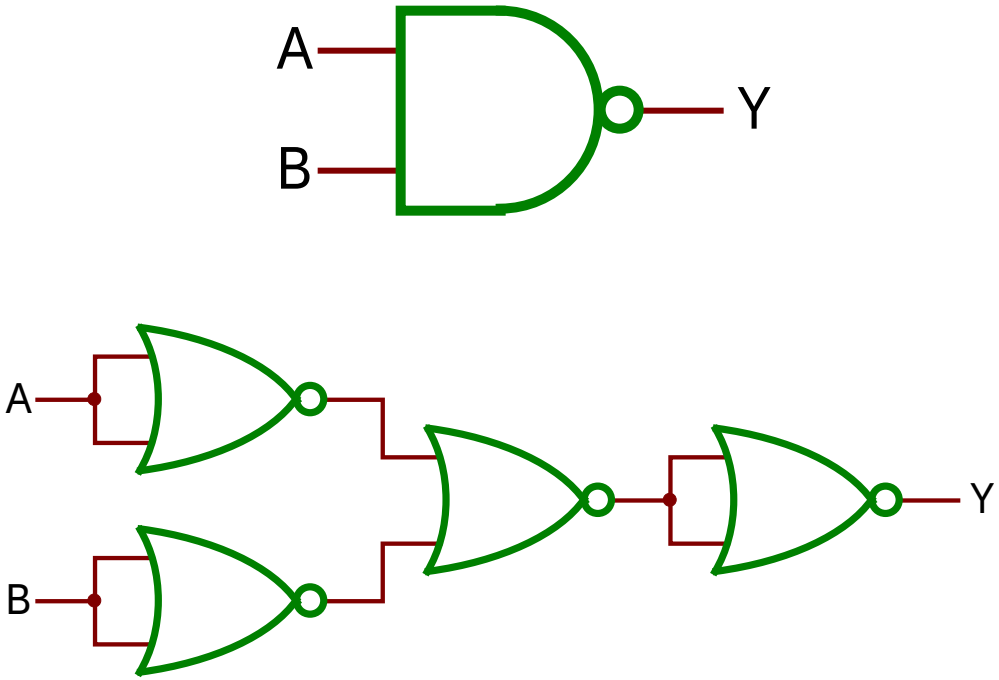
$$= A\left(\overline{A} + \overline{B}\right) + B\left(\overline{A} + \overline{B}\right) =$$

$$= \overline{A}B + A\overline{B} = A \oplus B$$

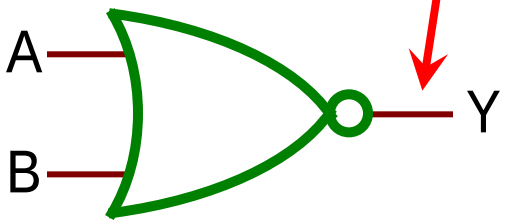

A
Y
A
Y

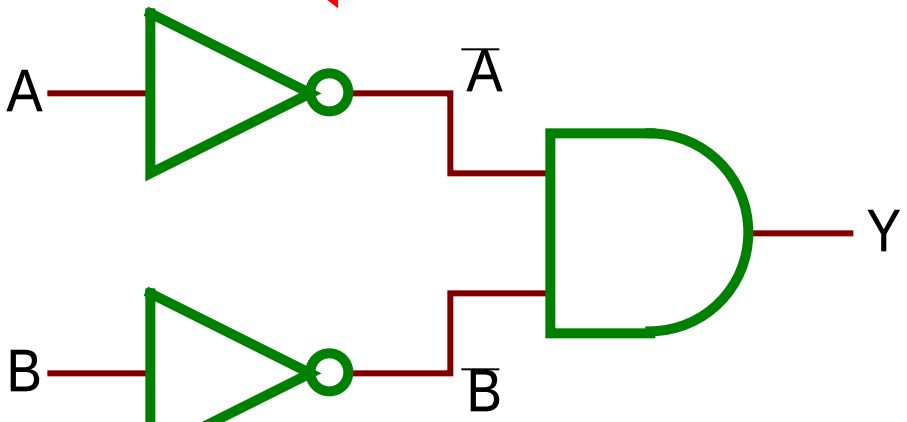

A
B
Y
A
B
Y

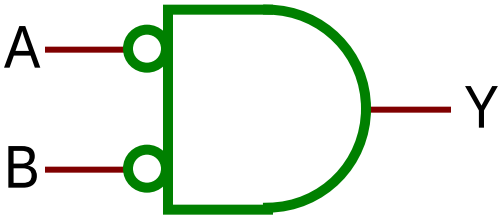

A
B
Y
A
B
Y


A
B
Y
A
B
Y

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







Κόστος πυλών

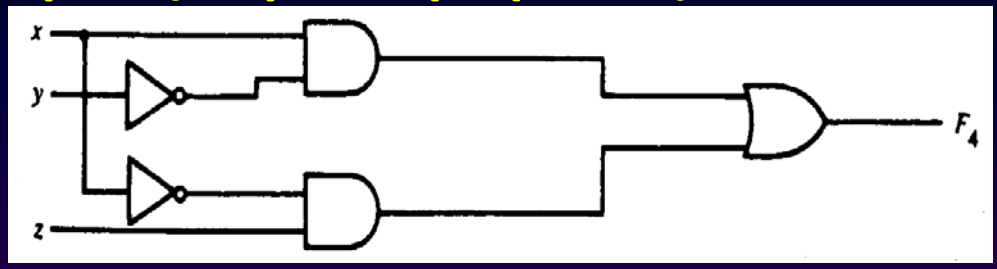
- Κοστίζουν όλες οι πύλες το ίδιο ?
- Οχι. Αλλωστε είναι χαρακτηριστικό ότι σε ένα ολοκληρωμένο κύκλωμα έχουμε 6 inverters και μόνο 4 OR, AND, …
- Το κόστος μιας πύλης εξαρτάται :
  - Από τον αριθμό των εισόδων της. Περισσότεροι είσοδοι => ↑ κόστος.
  - Από τη συνάρτηση που επιτελεί :
    - NAND, NOR οι πιο απλές.
    - AND, OR, ελάχιστα πιο πολύπλοκες (σε επίπεδο transistor).
    - XOR, XNOR αρκετά πιο πολύπλοκες
    - Μεγαλύτερη πολυπλοκότητα => ↑ κόστος

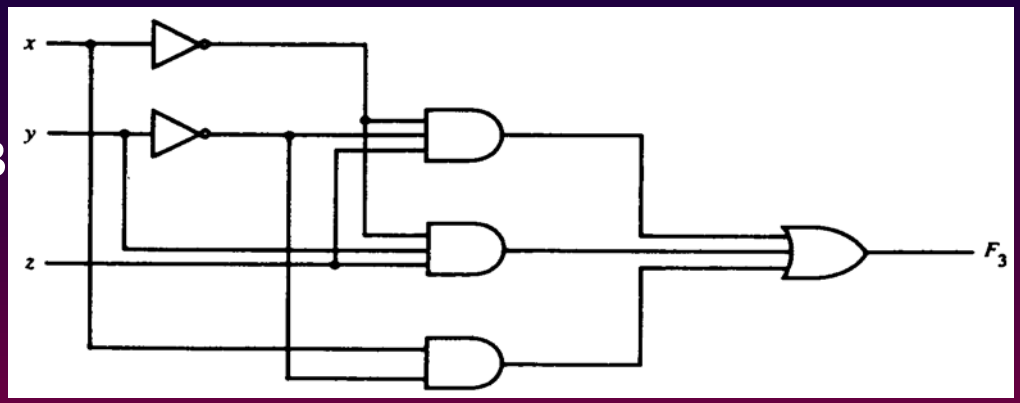
Ισοδύναμες και συμπληρωματικές συναρτήσεις

- Δύο συναρτήσεις των κ μεταβλητών είναι ισοδύναμες, αν για κάθε συνδυασμό τιμών των μεταβλητών εισόδου, οι συναρτήσεις οδηγούν στην ίδια έξοδο.
  - ***Συμπέρασμα 1 : Ισοδύναμες συναρτήσεις μπορεί να έχουν διαφορετικές αλγεβρικές εκφράσεις.***
  - ***Συμπέρασμα 2 : Ισοδύναμες συναρτήσεις μπορεί να έχουν διαφορετικά λογικά διαγράμματα.***
  - ***Συμπέρασμα 3 : Ισοδύναμες συναρτήσεις έχουν τον ίδιο πίνακα αληθείας.***
- Δύο συναρτήσεις των κ μεταβλητών είναι συμπληρωματικές, αν για κάθε συνδυασμό των μεταβήτών εισόδου, οι συναρτήσεις παράγουν συμπληρωματικές τιμές.

Μεταξύ ισοδύναμων συναρτήσεων κάποιες είναι σαφώς προτιμητέες !

- Η F_3 είναι σαφώς πιο πολύπλοκη συνάρτηση από την F_4.
- Χρησιμοποιεί περισσότερες πύλες, μα ταυτόχρονα και πύλες με περισσότερες εισόδους.
- Παράλληλα χρησιμοποιεί 4 διαφορετικά ολοκληρωμένα έναντι 3 της F_4.
- Το μεγάλο συνεπώς ερώτημα που προκύπτει είναι **πως θα βρω τη συνάρτηση με το ελάχιστο κόστος ανάμεσα στις ισοδύναμες ?**
- **Αυτό είναι το πιο ενδιαφέρον σημείο του μαθήματος : *απλοποίηση συναρτήσεων***





x	y	z	F_3	F_4
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	1	1
1	0	0	1	1
1	0	1	1	1
1	1	0	0	0
1	1	1	0	0

Η θεωρία παρέχει τη μέθοδο της αλγεβρικής απλοποίησης

- Θυμηθείτε ότι για την F_3 είχαμε ότι : $F_3 (X, Y, Z) = X \bullet Y' + X' \bullet Y \bullet Z + X' \bullet Y' \bullet Z$
- Από τη θεωρία βάσει του θεωρήματος των συνδυασμών γνωρίζω ότι :
  - $X' \bullet Y \bullet Z + X' \bullet Y' \bullet Z = X' \bullet Z$
  - Αρα $F_3 = X \bullet Y' + X' \bullet Z = F_4$.
- Η αλγεβρική απλοποίηση δεν είναι πάντα τόσο εύκολη !
- Ακόμα χειρότερα, ποτέ δε γνωρίζω αν έχοντας κάνει κάποια στάδια απλοποίησης έχω βρει την απολύτως ελάχιστη ισοδύναμη συνάρτηση.
- Η αλγεβρική απλοποίηση πρέπει να χρησιμοποιείται για συναρτήσεις πολύ λίγων μεταβλητών από έμπειρους σχεδιαστές.

Παραδείγματα αλγεβρικής απλοποίησης

- $F(X, Y, Z) = X \bullet Y' \bullet Z + X' \bullet Y \bullet Z + Y \bullet Z =$
  $= X \bullet Y' \bullet Z + Y \bullet Z$ (Απορρόφηση)
  $= Z \bullet (X \bullet Y' + Y)$ (Επιμεριστική)
  $= Z \bullet (X + Y)$ ($1^{ο}$ αποδειχθέν θεώρημα)

- Υλοποιείστε με τον ελάχιστο αριθμό πυλών τη $F(X, Y, Z) = X \bullet Y' \bullet Z + X \bullet Y' \bullet Z + X \bullet Y \bullet Z'$
  - $F(X, Y, Z) = X \bullet Y' \bullet Z + X \bullet Y' \bullet Z + X \bullet Y \bullet Z' =$
    $= X \bullet Y' \bullet Z + X \bullet Y \bullet Z'$ (Αυτοαπορρόφηση)
    $= X \bullet (Y' \bullet Z + Y \bullet Z')$ (Επιμεριστική)
    $= X \bullet (Y \oplus Z)$ (συνάρτηση XOR)

Προβλήματα & Αλγόριθμοι

- Η αλγεβρική απλοποίηση
  - Είναι δύσκολη
  - Μη ντετερμινιστική
  - Χωρίς σίγουρο αποτέλεσμα
- Θα ήθελα συνεπώς μια ντετερμινιστική μεθοδολογία.
- Στη γλώσσα των υπολογιστών μια μεθοδολογία που αποτελείται από μικρά κατανοητά βήματα και η οποία αν ακολουθηθεί παράγει τη λύση σε κάποιο πρόβλημα ονομάζεται ***αλγόριθμος***.
- Για να εφαρμοστεί κάποιος αλγόριθμος όμως απαιτείται να υπάρχει μια σταθερή αρχική μορφή του προβλήματος.
- Πριν λοιπόν δώσουμε αλγόριθμο απλοποίησης, πρέπει πρώτα να εισάγουμε πρότυπες μορφές έκφρασης μιας συνάρτησης.
- Σε αυτό θα μας βοηθήσουν οι ελαχιστόροι και οι μεγιστόροι.

Πίνακας αλήθειας

- Ισοδύναμες συναρτήσεις έχουν ποικίλες αλγεβρικές αναπαραστάσεις, ποικίλα λογικά διαγράμματα, *αλλά κοινό πίνακα αλήθειας.*
- Ο πίνακας αλήθειας μας δίνει τη τιμή μιας συνάρτησης για κάθε πιθανό συνδυασμό εισόδων.
- Συνήθως διατάσσουμε τους συνδυασμούς εισόδων σαν αύξοντες δυαδικούς αριθμούς.
- Ο πίνακας αλήθειας μιας λογικής συνάρτησης ν μεταβλητών έχει 2^{ν} γραμμές.

Οροι, αθροίσματα, γινόμενα

- Μια μεταβλητή στη κανονική ή τη συμπληρωματική της μορφή είναι ένας ***όρος***
  - Παραδείγματα όρων : X, X', Y, Z'
- Ενα ***γινόμενο*** είναι είτε ένας όρος είτε το λογικό AND δύο ή περισσότερων όρων
  - Παραδείγματα γινομένων : X, X • Z', Y• X', X • Z' • Y'
- Ενα ***άθροισμα*** είναι είτε ένας όρος είτε το λογικό OR δύο ή περισσοτέρων όρων
  - Παραδείγματα αθροισμάτων : X, X + Z', Y+ X', X + Z' + Y'
- ***Αθροισμα γινομένων (sum of products – SOP)*** είναι κάθε άθροισμα του οποίου οι όροι είναι γινόμενα
  - Παράδειγμα SOP : X + X • Z' + Y• X' + X • Z' • Y'
- ***Γινόμενο αθροισμάτων (product of sums – POS)*** είναι κάθε γινόμενο του οποίου οι όροι είναι αθροίσματα.
  - Παράδειγμα POS : X • (X + Z') • (Y+ X') • (X + Z' + Y')
- ***Κανονικός όρος*** είναι ένα άθροισμα ή ένα γινόμενο, στο οποίο κάθε μεταβλητή (κανονική ή συμπληρωμένη) εμφανίζεται μόνο 1 φορά. Κάθε μη κανονικός όρος μπορεί να μετατραπεί μέσω απλοποίησης σε κανονικό.
  - Κανονικοί όροι : X • Z' • Y', X + Z' + Y'
  - Μη κανονικοί όροι : X • Z' • Y' • X, X + Z' + Y' + Z

Ελαχιστόροι - Μεγιστόροι

- Κάθε κανονικός όρος - γινόμενο μιας συνάρτησης κ μεταβλητών, είναι ***ελαχιστόρος*** αν περιέχει κ όρους. Κάθε συνάρτηση κ μεταβλητών έχει $2^κ$ ελαχιστόρους.
  - Η F(X, Y) έχει τους ελαχιστόρους : X' • Y΄, X΄ • Y, X • Y' και X • Y
  - *Κάθε ελαχιστόρος αντιπροσωπεύει μία γραμμή του πίνακα αληθείας*

- Κάθε κανονικός όρος – άθροισμα μιας συνάρτησης κ μεταβλητών, είναι ***μεγιστόρος*** αν περιέχει κ όρους. Κάθε συνάρτηση κ μεταβλητών έχει $2^κ$ μεγιστόρους.
  - Η F(X, Y) έχει τους μεγιστόρους : X' + Y΄, X΄ + Y, X + Y' και X + Y
  - *Κάθε μεγιστόρος αντιπροσωπεύει μία γραμμή του πίνακα αληθείας*

Ελαχιστόροι – Μεγιστόροι και πίνακας αλήθειας

- Μεταξύ πίνακα αλήθειας και ελαχιστόρων (μεγιστόρων) υπάρχει μια στενή σχέση. Ο ελαχιστόρος (μεγιστόρος) μπορεί να οριστεί σα το γινόμενο (άθροισμα) που μπορεί να γίνει 1 (0) μόνο για τις τιμές εισόδου μιας και μόνο γραμμής του πίνακα αλήθειας.

			Ελαχιστόροι		Μεγιστόροι	
x	y	z	Όρος	Ονομασία	Όρος	Ονομασία
0	0	0	$x'y'z'$	m_0	$x + y + z$	M_0
0	0	1	$x'y'z$	m_1	$x + y + z'$	M_1
0	1	0	$x'yz'$	m_2	$x + y' + z$	M_2
0	1	1	$x'yz$	m_3	$x + y' + z'$	M_3
1	0	0	$xy'z'$	m_4	$x' + y + z$	M_4
1	0	1	$xy'z$	m_5	$x' + y + z'$	M_5
1	1	0	xyz'	m_6	$x' + y' + z$	M_6
1	1	1	xyz	m_7	$x' + y' + z'$	M_7

- Μπορούμε να ορίσουμε τη δυαδική τιμή που επαληθεύει τον όρο (κάνει 1 τον ελαχιστόρο ή 0 τον μεγιστόρο) σα το διακριτικό του αντίστοιχου όρου.
  - Π.χ. Ελαχιστόρος 4 = x y' z'. Μεγιστόρος 5 = x' + y + z'

Συνάρτηση σα κανονικό άθροισμα

- Αφού κάθε γραμμή του πίνακα αληθείας αντιστοιχεί σε ένα ελαχιστόρο μπορούμε κοιτώντας το πίνακα αληθείας να δούμε ποιοι ελαχιστόροι επαληθεύουν τη συνάρτηση.
- Το λογικό άθροισμα αυτών μας δίνει την έκφραση της συνάρτησης σα κανονικό άθροισμα.

x	y	z	F_3
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

- Η συνάρτηση επαληθεύεται όταν επαληθεύεται ο ελαχιστόρος 1 ή ο
ελαχιστόρος 3 ή ο
ελαχιστόρος 4 ή ο
ελαχιστόρος 5

Ισοδύναμα είναι $F_3(x, y, z) = m_1 + m_3 + m_4 + m_5 =$
$x'y'z + x'yz + xy'z' + xy'z =$
$\Sigma(1, 3, 4, 5)$

- Αφού ισοδύναμες συναρτήσεις έχουν κοινό πίνακα αλήθειας, θα έχουν και κοινά κανονικά αθροίσματα. ***Συνεπώς το κανονικό άθροισμα είναι μια πρότυπη μορφή πάνω στην οποία μπορώ να εφαρμόσω κάποιον αλγόριθμο.***

Συνάρτηση σα κανονικό γινόμενο

- Αφού κάθε γραμμή του πίνακα αληθείας αντιστοιχεί σε ένα μεγιστόρο μπορούμε κοιτώντας το πίνακα αληθείας να δούμε ποιοί μεγιστόροι **δεν** επαληθεύουν τη συνάρτηση.
- Το λογικό γινόμενο αυτών μας δίνει την έκφραση της συνάρτησης σα κανονικό γινόμενο.

x	y	z	F_3
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

- Η συνάρτηση δεν επαληθεύεται αν επαληθεύεται (παίρνει δηλαδή τη τιμή 0) ο

μεγιστόρος 0 ή ο

μεγιστόρος 2 ή ο

μεγιστόρος 6 ή ο

μεγιστόρος 7

Ισοδύναμα είναι $F'_3(x, y, z) = M'_0 + M'_2 + M'_6 + M'_7 \Rightarrow$

$$F_3(x, y, z) = M_0 \bullet M_2 \bullet M_6 \bullet M_7 =$$

$$(x + y + z) \bullet (x + y' + z) \bullet (x' + y' + z) \bullet (x' + y' + z') =$$

$$\Pi(0, 2, 6, 7)$$

- Αφού ισοδύναμες συναρτήσεις έχουν κοινό πίνακα αλήθειας, θα έχουν και κοινά κανονικά γινόμενα. ***Συνεπώς το κανονικό γινόμενο είναι μια πρότυπη μορφή πάνω στην οποία μπορώ να εφαρμόσω κάποιον αλγόριθμο.***

Μετατροπές μεταξύ κανονικών μορφών

- Αφού κάθε γραμμή του πίνακα αληθείας είναι είτε 0 είτε 1 αν γνωρίζω τη μορφή της συνάρτησης στη μία κανονική μορφή η άλλη προκύπτει από τους όρους που λείπουν.
  - Π.χ. $F(A, B, C) = \Sigma (1,4,6) \Rightarrow F = \Pi (0, 2, 3, 5, 7)$

    $G(W, X, Y, Z) = \Pi (1, 8, 11, 14, 15) \Rightarrow$

    $G = \Sigma (0, 2, 3, 4, 5, 6, 7, 9, 10, 12, 13)$
- Είναι επίσης πολύ εύκολο για μια συνάρτηση F να βρώ τη F'
- Ισχύει ότι $m'_i = M_i$ και φυσικά $M'_i = m_i$
  - Για παράδειγμα είναι $m'_0 = (x'\ y'\ z')' = x + y + z = M_0$
- Αρα αν $F(x, y, z) = \Sigma (1, 3,4) \Rightarrow F = m_1 + m_3 + m_4 \Rightarrow$

  $F' = (m_1 + m_3 + m_4)' = M_1 \bullet M_3 \bullet M_4 = \Pi(1, 3, 4) = \Sigma (0, 2, 5, 6, 7)$
- ***Δηλαδή η συμπληρωματική μιας συνάρτησης προκύπτει σα κανονικό άθροισμα των ελαχιστόρων που λείπουν από το κανονικό της άθροισμα***

Να θυμάστε

Πίνακας αλήθειας
Κανονικό άθροισμα
Κανονικό γινόμενο

} πρότυπες μορφές

Αλγεβρική μορφή
Λογικό διάγραμμα
Λεκτική περιγραφή

} μή πρότυπες μορφές

Πρέπει επίσης να μπορείτε να πηγαίνετε από μη πρότυπες μορφές σε πρότυπες.

Απλοποίηση Συναρτήσεων

- Σκοποί της απλοποίησης
  - Λιγότεροι όροι
  - Απλούστεροι όροι
- Θέλουμε απλές και συστηματικές μεθόδους
- Υπάρχουν :
  - Η μέθοδος του χάρτη (μέθοδος Karnaugh / k-map) : γραφική μέθοδος για συναρτήσεις έως 5 μεταβλητών.
  - Η μέθοδος Quine-McClauskey : αλγεβρική μέθοδος
  - Η μέθοδος Espresso : αλγεβρική μέθοδος
- ***Οι μέθοδοι αυτοί δε μας δίνουν τις υλοποιήσεις με τις λιγότερες πύλες, αλλά τις απλούστερες υλοποιήσεις με NOT, AND & OR.***

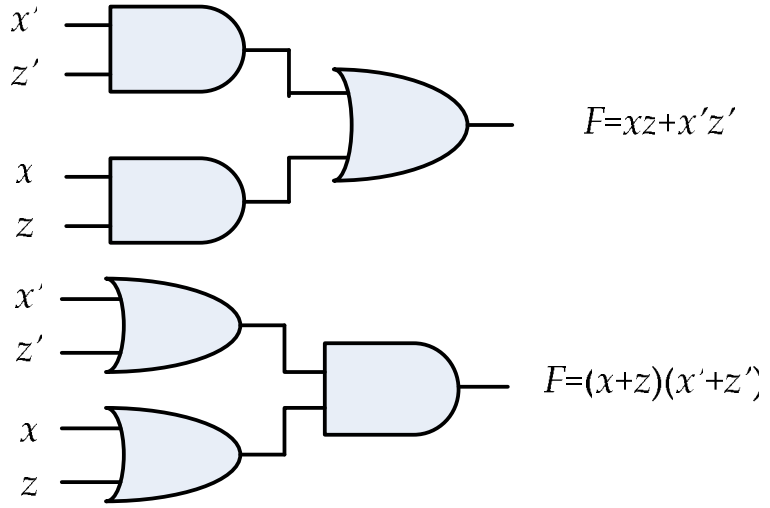
Πίνακας αλήθειας
Κανονικό άθροισμα
Κανονικό γινόμενο
} πρότυπες μορφές

Αλγεβρική μορφή
Λογικό διάγραμμα
Λεκτική περιγραφή
} μή πρότυπες μορφές

Υλοποίηση με AND και OR

- ✓ Σημειώνοντας τους άσσους ή τα μηδενικά μιας συνάρτησης πάνω στον χάρτη μπορούμε να γράψουμε μια συνάρτηση ως άθροισμα γινομένων ή ως γινόμενο αθροισμάτων.
- ✓ Με τον τρόπο αυτό μπορούμε να υλοποιήσουμε την συνάρτηση χρησιμοποιώντας μια λογική δύο επιπέδων όπου τα γινόμενα υλοποιούνται με πύλες AND ενώ τα αθροίσματα με πύλες OR.
- ✓ Για παράδειγμα η F=Σ(1,3,4,6) έχει τον παρακάτω χάρτη

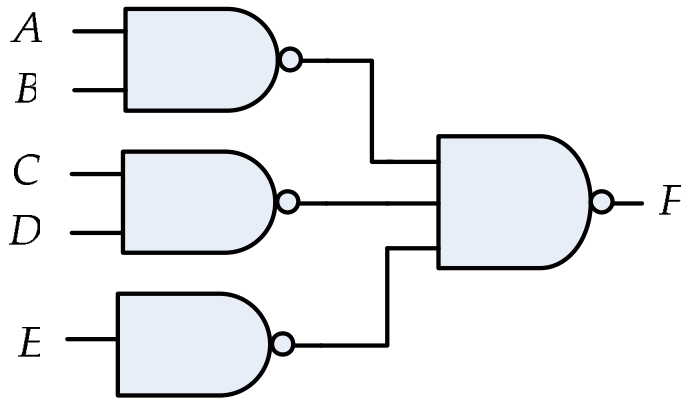
x \ yz	00	01	11	10
0	0	1	1	0
1	1	0	0	1



- ✓ Κάνοντας τις απλοποιήσεις μπορούμε να βρούμε ότι η F γράφεται $F=xz+x'z'$ και $F=(x'+z')(x+z)$
- ✓ Ο συνολικός αριθμός πυλών μπορεί να διαφέρει στις δύο υλοποιήσεις!

Υλοποίηση με NAND και NOR

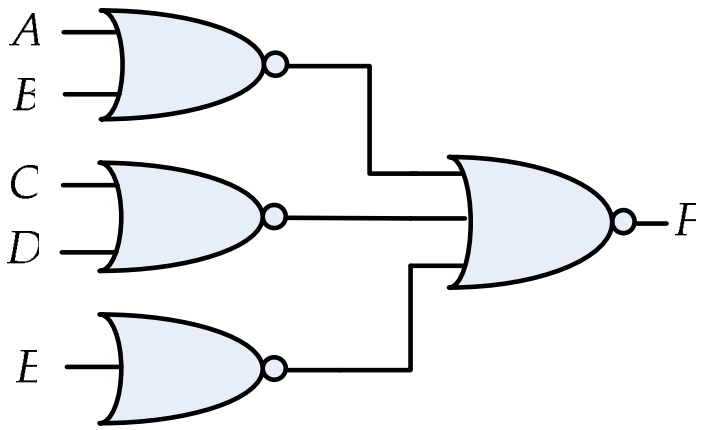
- ✓ Πολλές φορές οι πύλες NAND και NOR προτιμούνται στην υλοποίηση ψηφιακών κυκλωμάτων.
- ✓ Αυτό γίνεται επειδή απαιτούν μικρότερο αριθμό τρανζίστορ και είναι πιο γρήγορες.
- ✓ Για να υλοποιήσουμε μία ψηφιακή συνάρτηση με την βοήθεια πυλών NAND και NOR χρησιμοποιούμε το θεώρημα του DeMorgan.
- ✓ Στην περίπτωση υλοποίησης με πύλες NAND πρέπει να μετατρέψουμε την έκφραση της F από άθροισμα γινομένων σε μια μορφή που να περιέχει όρους της μορφής $(xy)'$.
- ✓ Για παράδειγμα αν $F=AB+CD+E$, τότε χρησιμοποιώντας το γεγονός πως $(x'y')'=x+y$ μπορούμε να γράψουμε την F ως $F=((AB)'(CD)'E')'$.



✓Παρατηρείστε πως το E' υλοποιείται με την μία NAND μίας εισόδου. Αυτό οφείλεται στο γεγονός πως αν κανείς δει το σχεδιαγραμμα της πύλης NAND σε επίπεδο τρανζίστορ προκύπτει πως μια NAND με μία είσοδο μπορεί να θεωρηθεί ως μία πύλη NOT!

Υλοποίηση με NAND και NOR

- ✓ Παρόμοια είναι και η διαδικασία υλοποίησης με πύλες NOR.
- ✓ Στην περίπτωση αυτή ξεκινάμε από την έκφραση της συνάρτησης σε γινόμενο αθροισμάτων και χρησιμοποιούμε πάλι τον νόμο του DeMorgan.
- ✓ Για παράδειγμα αν $F=(A+B)(C+D)E$, τότε θα έχουμε $F=((A+B)'+(C+D)'+E')'$
- ✓ Όπως και στις πύλες NAND, μία πύλη NOR με μία είσοδο είναι μία πύλη NOT.



ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ
(Τ.Ε.Ι.) ΚΡΗΤΗΣ
Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων

Ψηφιακή Σχεδίαση

Κεφάλαιο 3: Ελαχιστοποίηση σε Επίπεδο Πυλών

Γ. Κορνάρος

Φεβρουάριος 2012

Εισαγωγή – Χάρτες Karnaugh

- Υλοποίηση λογικής 2-Επιπέδων χρησιμοποιώντας SOP ή POS δεν είναι ο πιο οικονομικός τρόπος όσο αφορά #πυλών και #εισόδων
- Ένας χάρτης Karnaugh είναι μια γραφική αναπαράσταση ενός πίνακα αλήθειας
  - Ο πίνακας περιέχει ένα κελί για κάθε δυνατό **ελαχιστόρο**
  - Γειτονικά κελιά διαφέρουν σε ένα μόνο literal; Λόγου χάρη στο x (ή στο x')
  - Η συνάρτηση σχηματίζεται βάζοντας 1 στα κελιά που αντιστοιχούν στους ελαχιστόρους της συνάρτησης
  - Βάζουμε 0 σε όλα τα υπόλοιπα κελιά

Karnaugh με δύο μεταβλητές

- F=f(x,y)

x	y	F
0	0	m0
0	1	m1
1	0	m2
1	1	m3

x \ y	0	1
0	m0	m1
1	m2	m3

- Παράδειγμα, F1=x'y

x \ y	0	1
0		
1		

Χάρτες Karnaugh δύο-τριών μεταβλητών

	x′y′	x′y
x	xy′	xy

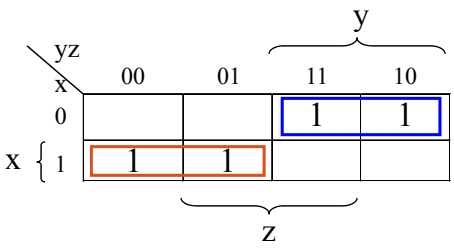
	m0	m1
	m2	m3

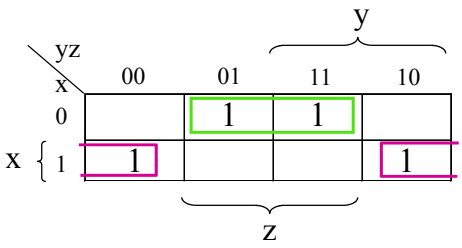
	yz = 00	01	11	10
x = 0	x′y′z′	x′y′z	x′yz	x′yz′
x = 1	xy′z′	xy′z	xyz	xyz′

m0	m1	m3	m2
m4	m5	m7	m6

- Οι ελαχιστόροι τοποθετούνται σε σειρά όχι δυαδικών αριθμών , αλλά σύμφωνα με τον ανακλαστικό κώδικα (gray).
- Δύο γειτονικά τετράγωνα στο χάρτη διαφέρουν κατά μία μόνο μεταβλητή

Χάρτες Karnaugh δύο-τριών μεταβλητών (2)





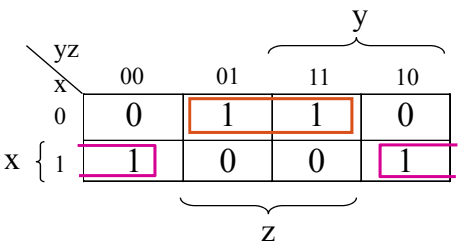
$F = x'yz + x'yz' + xy'z' + xy'z$

$= x'y + xy'$

$F = x'y'z + x'yz + xy'z' + xyz'$

$= x'z + xz'$

Χάρτες Karnaugh δύο-τριών μεταβλητών (3)



Απλοποίηση σε μορφή αθροίσματος ελαχιστόρων

$F = x'y'z + x'yz + xy'z' + xyz'$

$= x'z + xz'$

Απλοποίηση σε μορφή γινομένου μεγιστόρων

$F' = x'y'z' + x'yz' + xy'z + xyz$

$= x'z + xz'$

$\Rightarrow (F')' = (x'z + xz')' = (x+z')(x'+z)$

Karnaugh δύο-τριών μεταβλητών

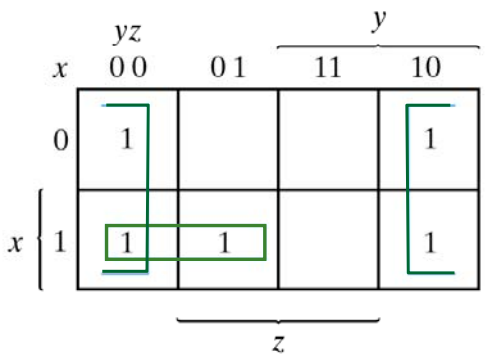


Fig. 3-6 Map for Example 3-3; $F(x, y, z) = \Sigma(0, 2, 4, 5, 6) = z' + xy'$

Karnaugh δύο-τριών μεταβλητών

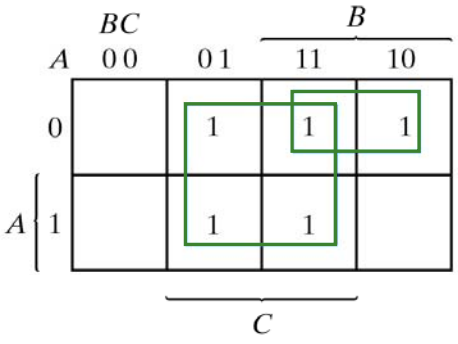


Fig. 3-7 Map for Example 3-4; $A'C + A'B + AB'C + BC = C + A'B$

Χάρτες Karnaugh τεσσάρων μεταβλητών

wx \ yz	00	01	11	10
00	w′x′y′z′	w′x′y′z	w′x′yz	w′x′yz′
01	w′xy′z′	w′xy′z	w′xyz	w′xyz′
11	wxy′z′	wxy′z	wxyz	wxyz′
10	wx′y′z′	wx′y′z	wx′yz	wx′yz′

y (yz: 00, 01, 11, 10) — x (wx: 00, 01, 11, 10) — w — z

m_0	m_1	m_3	m_2
m_4	m_5	m_7	m_6
m_{12}	m_{13}	m_{15}	m_{14}
m_8	m_9	m_{11}	m_{10}

Χάρτες Karnaugh τεσσάρων μεταβλητών

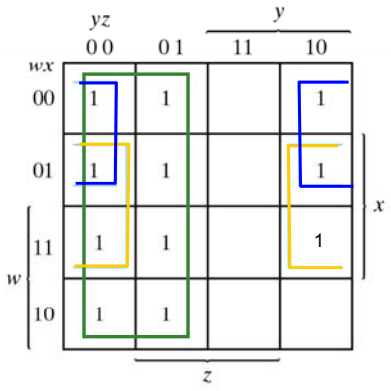


Fig. 3-9 Map for Example 3-5; $F(w, x, y, z)$
$= \Sigma\,(0, 1, 2, 4, 5, 6, 8, 9, 12, 13, 14) = y' + w'z' + xz'$

Πρωτεύοντες όροι - Prime Implicants

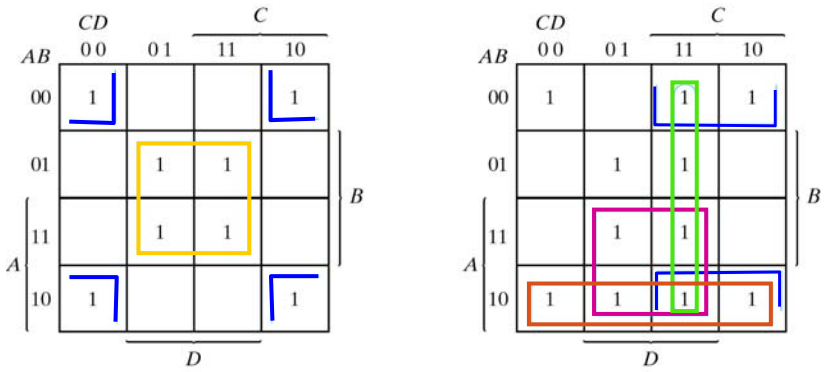


Fig. 3-11 Simplification Using Prime Implicants

Χάρτης 5 μεταβλητών

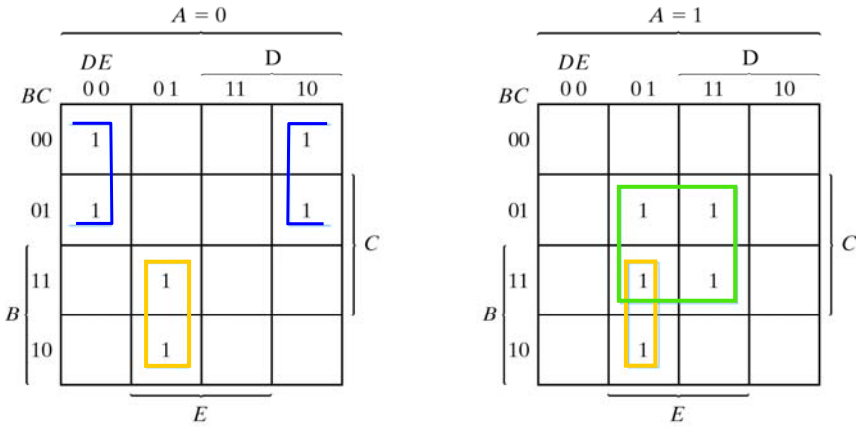
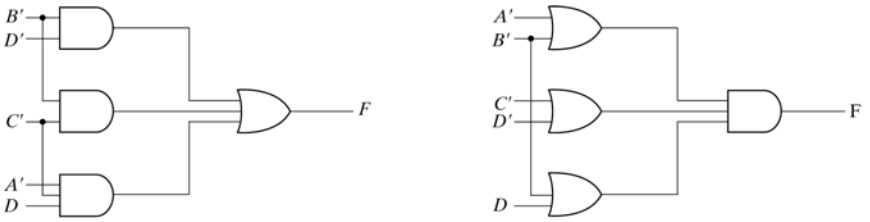


Fig. 3-13 Map for Example 3-7; $F = A'B'E' + BD'E + ACE$

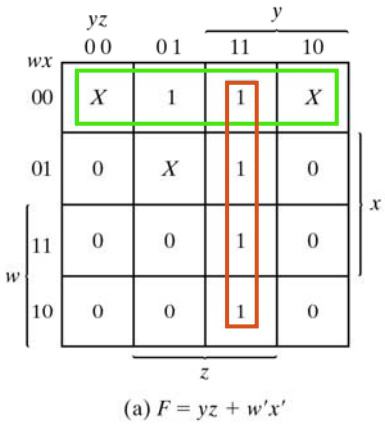
Υλοποίηση με πύλες



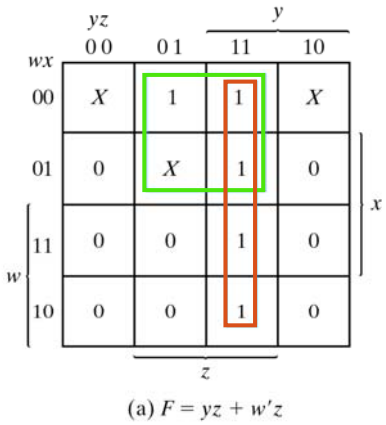
(a) $F = B'D' + B'C' + A'C'D$ (b) $F = (A' + B')(C' + D')(B' + D)$

Fig. 3-15 Gate Implementation of the Function of Example 3-8

Υλοποίηση με αδιάφορους όρους



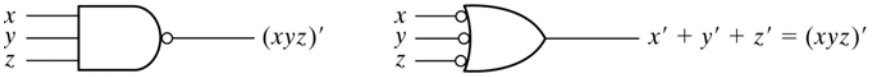
(a) $F = yz + w'x'$



(a) $F = yz + w'z$

Fig. 3-17 Example with don't-care Conditions

Πύλες NAND



(a) AND–invert (b) Invert–OR

Fig. 3-19 Two Graphic Symbols for NAND Gate

Διαφορετικοί τρόποι υλοποίησης

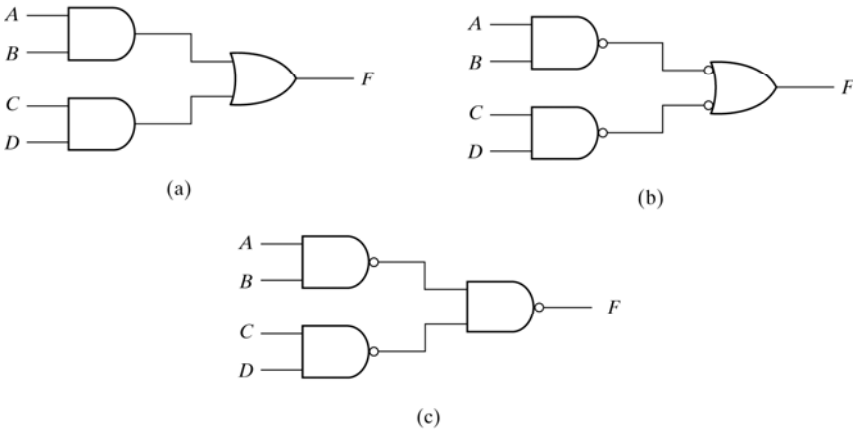
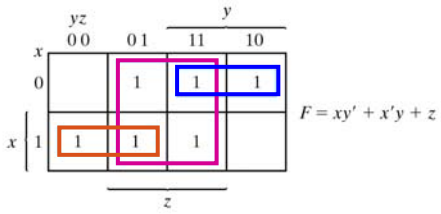
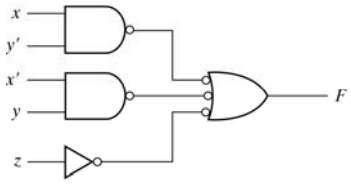


Fig. 3-20 Three Ways to Implement $F = AB + CD$

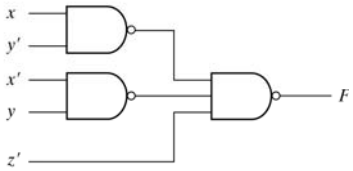
Υλοποίηση με NAND



(a)



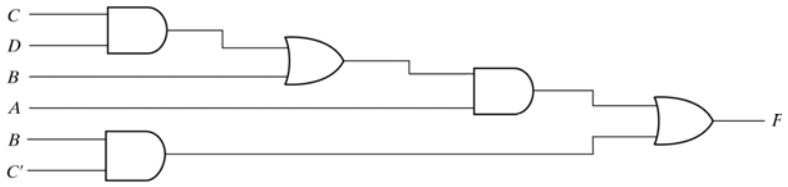
(b)



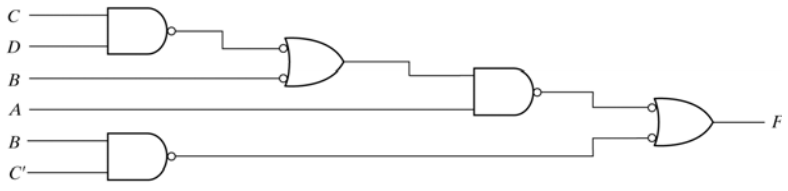
(c)

Fig. 3-21 Solution to Example 3-10

Κυκλώματα ΟΧΙ-ΚΑΙ πολλαπλών επιπέδων



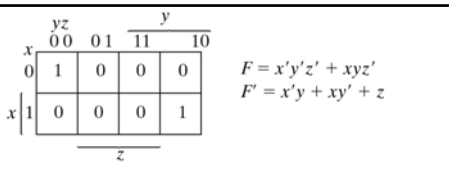
(a) AND-OR gates



(a) NAND gates

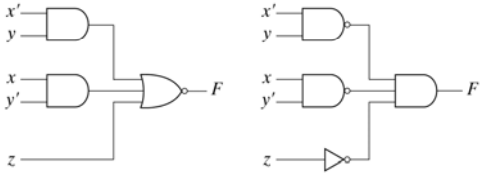
Fig. 3-22 Implementing $F = A(CD + B) + BC$

Διεπίπεδες Υλοποιήσεις



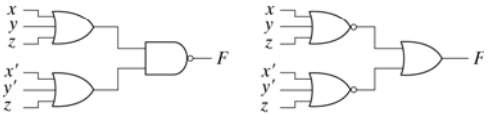
$F = x'y'z' + xyz'$
$F' = x'y + xy' + z$

(a) Map simplification in sum of products.



AND-NOR NAND-AND

(b) $F = (x'y + xy' + z)'$

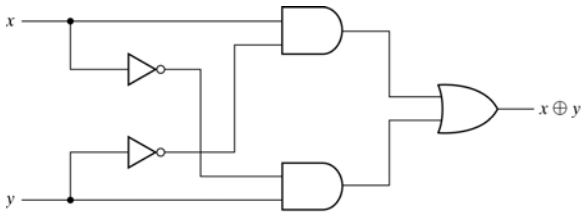


OR-NAND NOR-OR

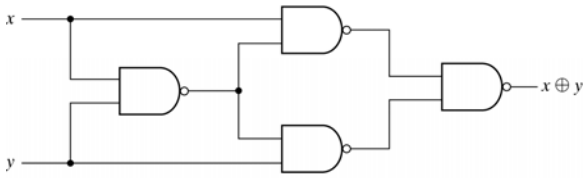
(c) $F = [(x + y + z)(x' + y' + z)]'$

Fig. 3-31 Other Two-level Implementations

Υλοποίηση της συνάρτησης XOR



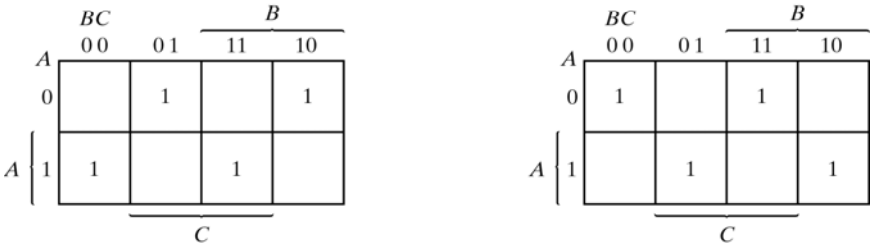
(a) With AND-OR-NOT gates



(b) With NAND gates

Fig. 3-32 Exclusive-OR Implementations

Χάρτης Karnaugh της περιττής και της άρτιας ισοτιμίας - συνάρτησης

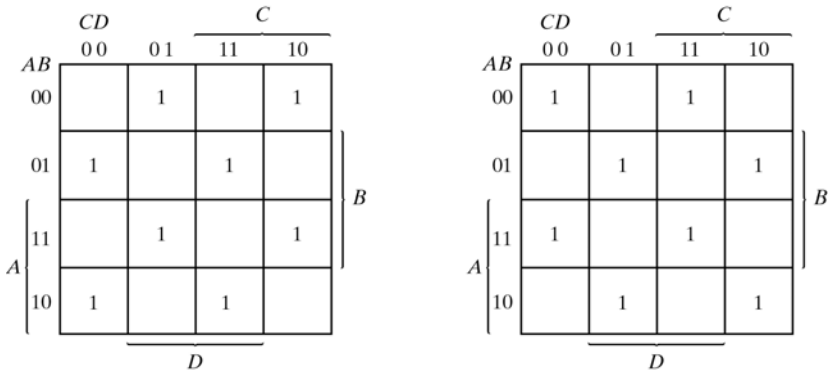


(a) Odd function
$F = A \oplus B \oplus C$

(a) Even function
$F = (A \oplus B \oplus C)'$

Fig. 3-33 Map for a Three-variable Exclusive-OR Function

Χάρτης Karnaugh της περιττής και της άρτιας ισοτιμίας – συνάρτησης 4 μεταβλητών

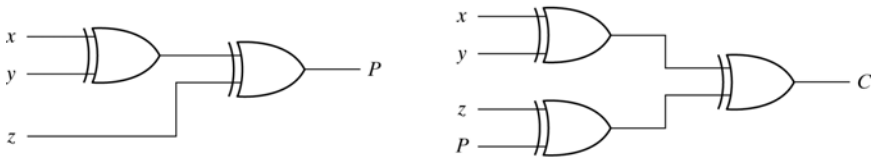


(a) Odd function
$F = A \oplus B \oplus C \oplus D$

(b) Even function
$F = (A \oplus B \oplus C \oplus D)'$

Fig. 3-35 Map for a Four-variable Exclusive-OR Function

Γεννήτρια άρτιας ισοτιμίας 3 bit
Ελεγκτής άρτιας ισοτιμίας 4 bit



(a) 3-bit even parity generator (a) 4-bit even parity checker

Fig. 3-36 Logic Diagram of a Parity Generator and Checker