Προγραμματιζόμενες Συσκευές

Εισαγωγή

Οι προγραμματιζόμενες συσκευές:

- ✓ Είναι ολοκληρωμένα κυκλώματα τα οποία **προγραμματίζονται** από τον σχεδιαστή στο εργαστήριο του.
- ✓ Αποτελούνται από **επαναλαμβανόμενες διατάξεις** κυττάρων τα οποία **διαμορφώνονται** και **διασυνδέονται** για να υλοποιήσουν ένα κύκλωμα.
- ✓ Χρησιμοποιούνται ειδικά εργαλεία για **εισαγωγή σχεδιασμού**, **εξομοίωση** και **προγραμματισμό** της συσκευής.

Πλεονεκτήματα

- *Δεν απαιτείται καμία μάσκα κατασκευής.*
- *Μειωμένο κόστος μονάδος.*
- *Είναι ιδανικές για προτυποποίηση.*
- *Όλη η σχεδίαση γίνεται στο εργαστήριο.*
- *Εύκολος προγραμματισμός.*
- *Γίνονται σχεδιαστικές αλλαγές χωρίς κόστος*

Εισαγωγή

Μεγάλος αριθμός κομματιών ⟺ Το κόστος πρωτοτυποποίησης μεγάλο αλλά εξισταθμίζεται από το κόστος κάθε μονάδας.

↓

Συμφέρουν τα MGAs

Μικρός αριθμός κομματιών

↓

Κόστος κατασκευής του πρωτότυπου, κόστος σχεδιασμού και χρόνος αναμονής → Συμφέρουν τα FPGAs

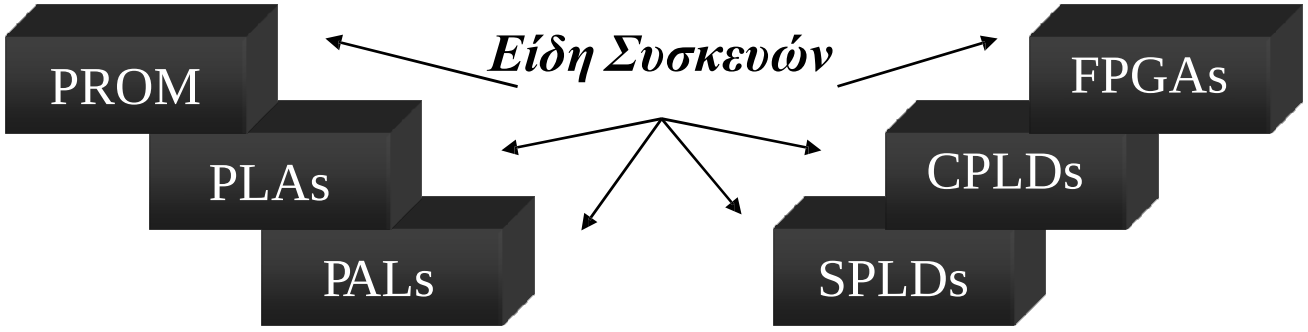
Οι προγραμματιζόμενες συσκευές διατίθενται στην αγορά σε μεγάλη ποικιλία δυνατοτήτων ολοκλήρωσης και ταχύτητας

Εισαγωγή

- ✓ Κάθε οικογένεια προγραμματιζόμενων συσκευών έχει την δική της αρχιτεκτονική.
- ✓ Εξειδικευμένο λογισμικό εκμεταλλεύεται βέλτιστα τα χαρακτηριστικά της για την υλοποίηση σχεδιαζόμενων συστημάτων.
- ✓ Σχεδιάζονται νέες αρχιτεκτονικές διαρκώς για να βελτιώσουν τις επιδόσεις των προγραμματιζόμενων συσκευών.
- ✓ Για την αξιολόγηση χρησιμοποιούνται γνωστά κυκλώματα (*benchmarks*).
- ✓ Παρέχονται σε μεγάλη ποικιλία ανάλογα με το περίβλημα, την απόδοση και φυσικά την τιμή κάθε οικογένειας.

Γενικά Χαρακτηριστικά

- ✓ **Προγραμματιζόμενες Διατάξεις Πυλών (PLDs):** Μπορούν να προγραμματιστούν μετά την κατασκευή ώστε να έχουν διαφορετικές συναρτήσεις σε αντίθεση με τα ASIC που έχουν σταθερές συναρτήσεις για κάθε στοιχείο
- ✓ Είναι διατάξεις από λογικά κύτταρα με δυνατότητες προγραμματισμού, και διασύνδεσης.
- ✓ Υπάρχουν και προγραμματιζόμενα κύτταρα εισόδου/εξόδου.
- ✓ Για την διασύνδεση των κυττάρων χρησιμοποιείται ειδικό πλάνο διασύνδεσης, το οποίο είναι διαφορετικό για την κάθε εταιρία και οικογένεια.
- ✓ Ο προγραμματισμός των συσκευών μπορεί να είναι μόνιμος ή προσωρινός.



Είδη Προγραμματιζόμενων Συσκευών

Μνήμη PROM

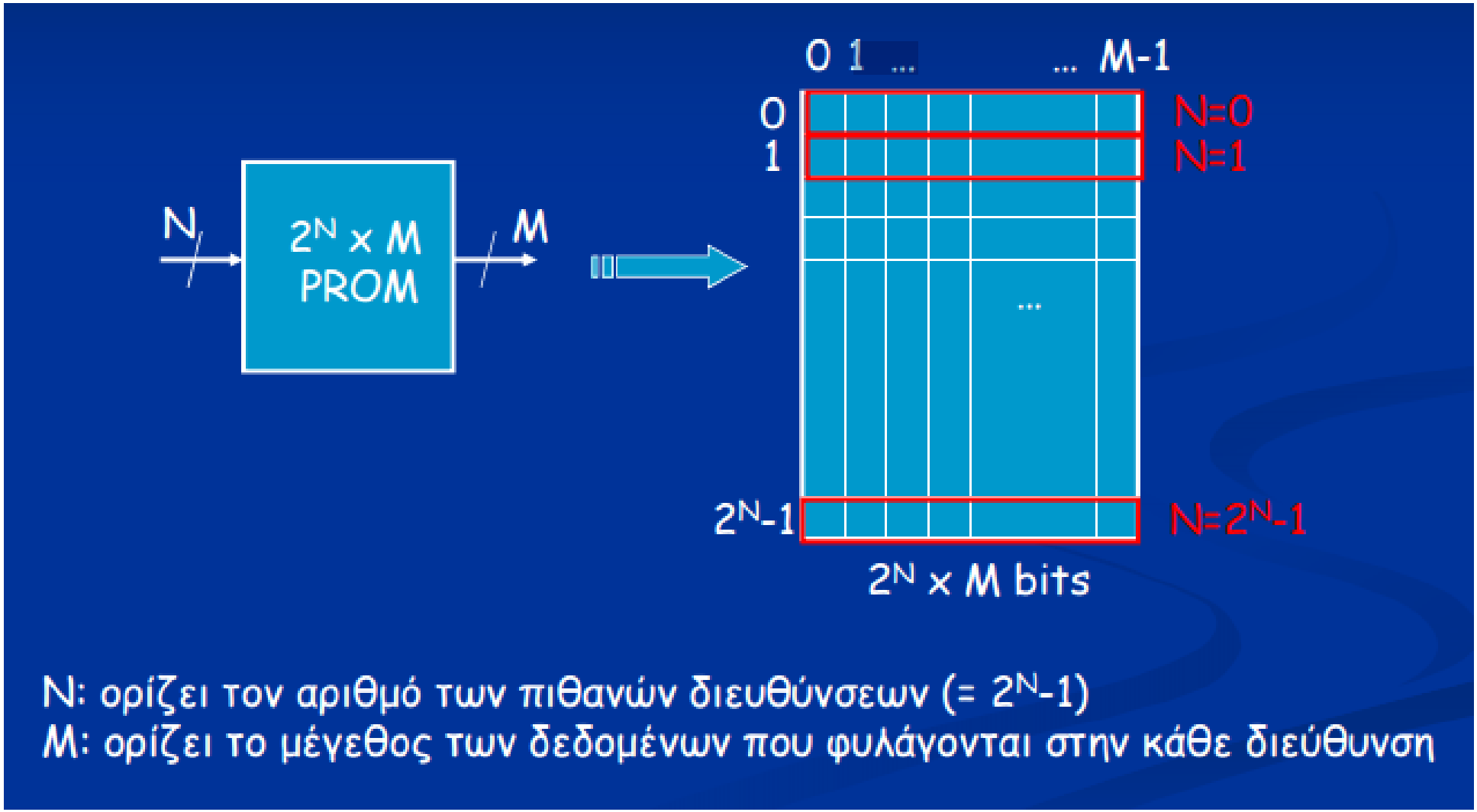
- ✓ Είναι το απλούστερο chip που μπορεί να προγραμματιστεί για να υλοποιεί κάποιο κύκλωμα.
- ✓ Είναι μνήμη που όταν προγραμματιστεί διατηρεί τα περιεχόμενα της και μετά την πτώση της τροφοδοσίας.
- ✓ Κατά την διάρκεια της λειτουργία της επιτρέπει μόνο ανάγνωση.
- ✓ Οι γραμμές διευθύνσεων αποτελούν τις εισόδους του κυκλώματος και οι γραμμές δεδομένων τις εξόδους.
- ✓ Τα περιεχόμενα της PROM είναι ο πίνακας αλήθειας του κυκλώματος.
- ✓ Η μέθοδος αυτή είναι μη-οικονομική αφού το κόστος της PROM είναι μεγάλο ενώ μία λογική συνάρτηση απαιτεί μόνο λίγους όρους συνήθως για να υλοποιηθεί (και έχει πολλούς αδιάφορους όρους X).

Είδη Προγραμματιζόμενων Συσκευών

Μνήμη PROM

- ✓ Οι προγραμματιζόμενες ROM ((PROM)) έχουν:
  - ✓N γραμμές εισόδων,
  - ✓M γραμμές εξόδων, και
  - ✓2^N κωδικοποιημένους ελαχιστόρους (ή διευθύνσεις).
- ✓ Αμετάβλητη διάταξη AND με 2^N εξόδους που υλοποιεί όλους τους ελαχιστόρους (συνήθως με Decoder).
- ✓ Προγραμματιζόμενη διάταξη OR με M εξόδους που σχηματίζει μέχρι και M αθροίσματα ελαχιστόρων.
- ✓ Ένα πρόγραμμα για ένα ROM ή PROM είναι ένας πίνακας αληθείας με πολλαπλό αριθμό εξόδων.
- ✓ Εάν έχουμε είσοδο 1 τότε γίνεται μια ένωση (σύνδεση) στον αντίστοιχο ελαχιστόρο για την αντίστοιχη έξοδο
- ✓ Εάν έχουμε 0, δεν γίνεται καμιά ένωση
- ✓ Μπορούμε να το δούμε σαν μια μνήμη με τις εισόδους ως διευθύνσεις δεδομένων, άρα ROM ή PROM!

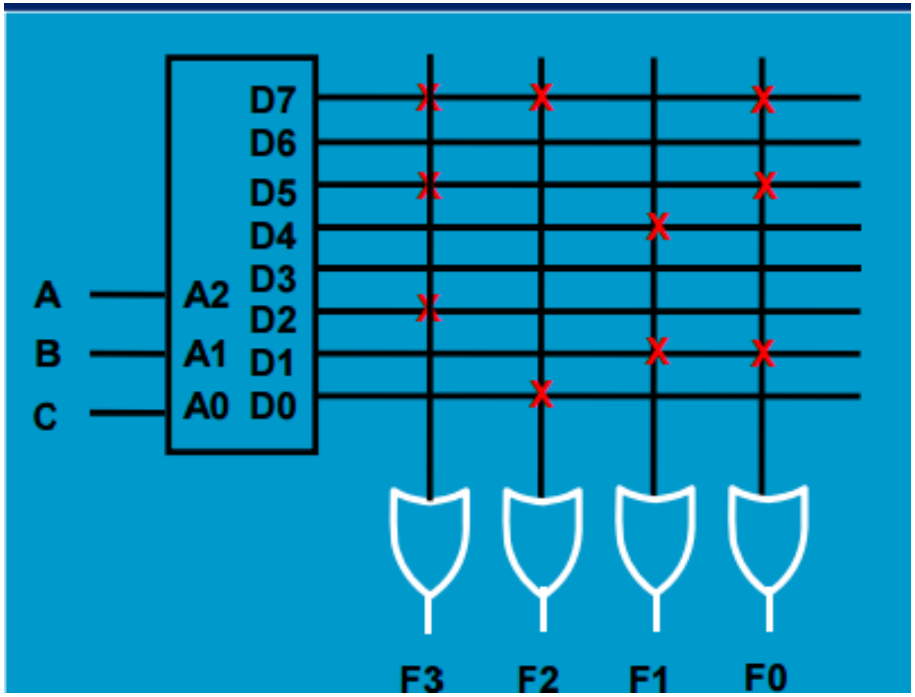
PROM (Παράδειγμα)



N: ορίζει τον αριθμό των πιθανών διευθύνσεων (= 2^N-1)
M: ορίζει το μέγεθος των δεδομένων που φυλάγονται στην κάθε διεύθυνση

PROM (Παράδειγμα)

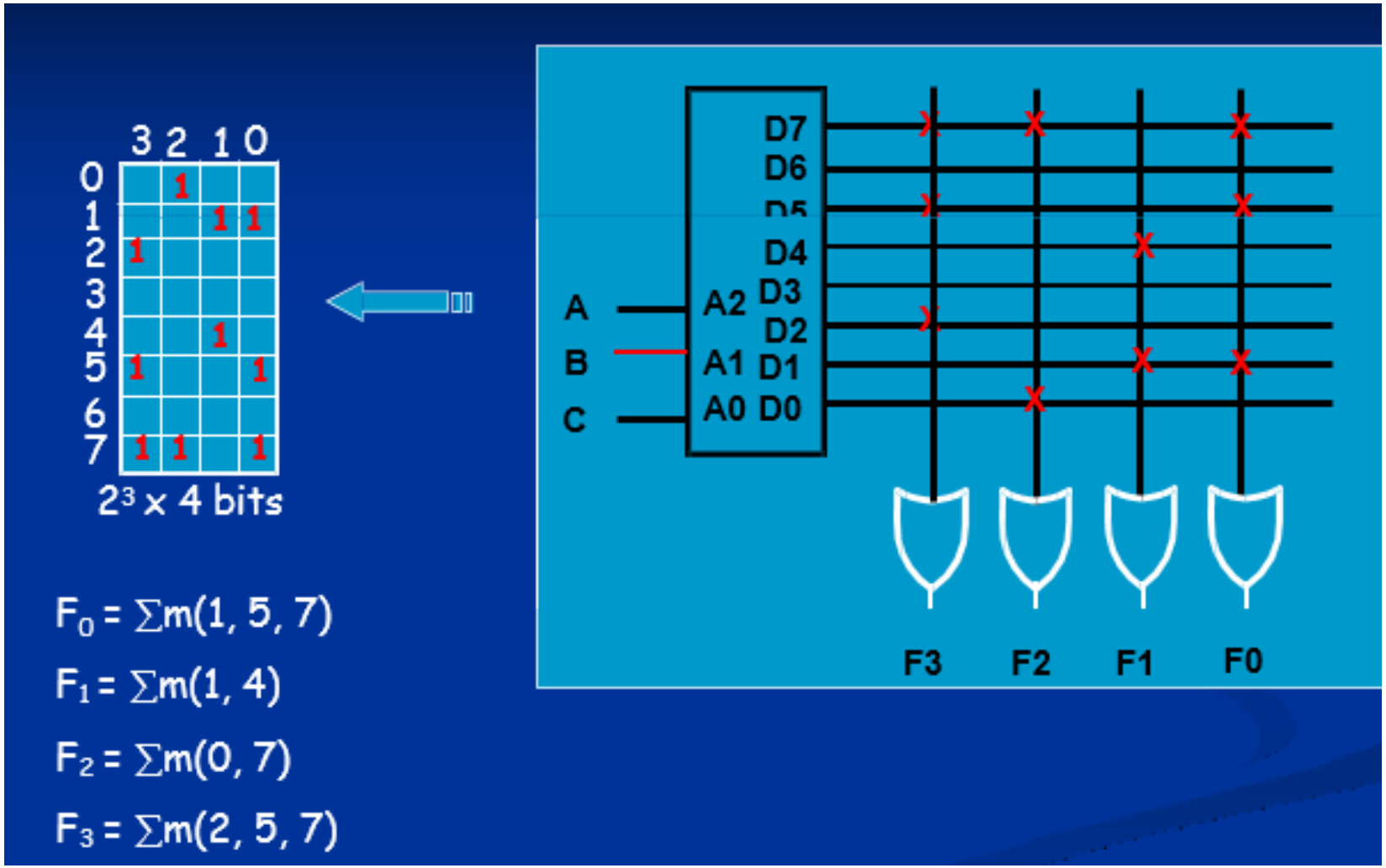
- Παράδειγμα: 8 X 4 ROM (N = 3 είσοδοι, M= 4 έξοδοι)
- Η διάταξη "AND" είναι ένας "αποκωδικοποιητής" με 3 εισόδους και 8 εξόδους που υλοποιεί όλους τους ελαχιστόρους..
- Η προγραμματιζόμενη διάταξη "OR" χρησιμοποιεί μια κάθετη γραμμή για την αναπαράσταση όλων των εισόδων στην πύλη OR. Ένα "x" στην διάταξη αντιστοιχεί στην προσάρτηση του ελαχιστόρου στο OR



- Παράδειγμα: Για είσοδο (A_2,A_1,A_0) ==001„η έξοδος είναι (F_3,F_2,F_1,F_0) = 0011.
- Τι είναι οι συναρτήσεις F_3, F_2 , F_1 και F_0 σε σχέση με τους όρους (A_2, A_1, A_0);;

PROM (Παράδειγμα)

- Παράδειγμα: 8 X 4 ROM (N = 3 είσοδοι, M= 4 έξοδοι)



Είδη Προγραμματιζόμενων Συσκευών

PLA (Programmable LogicArray)

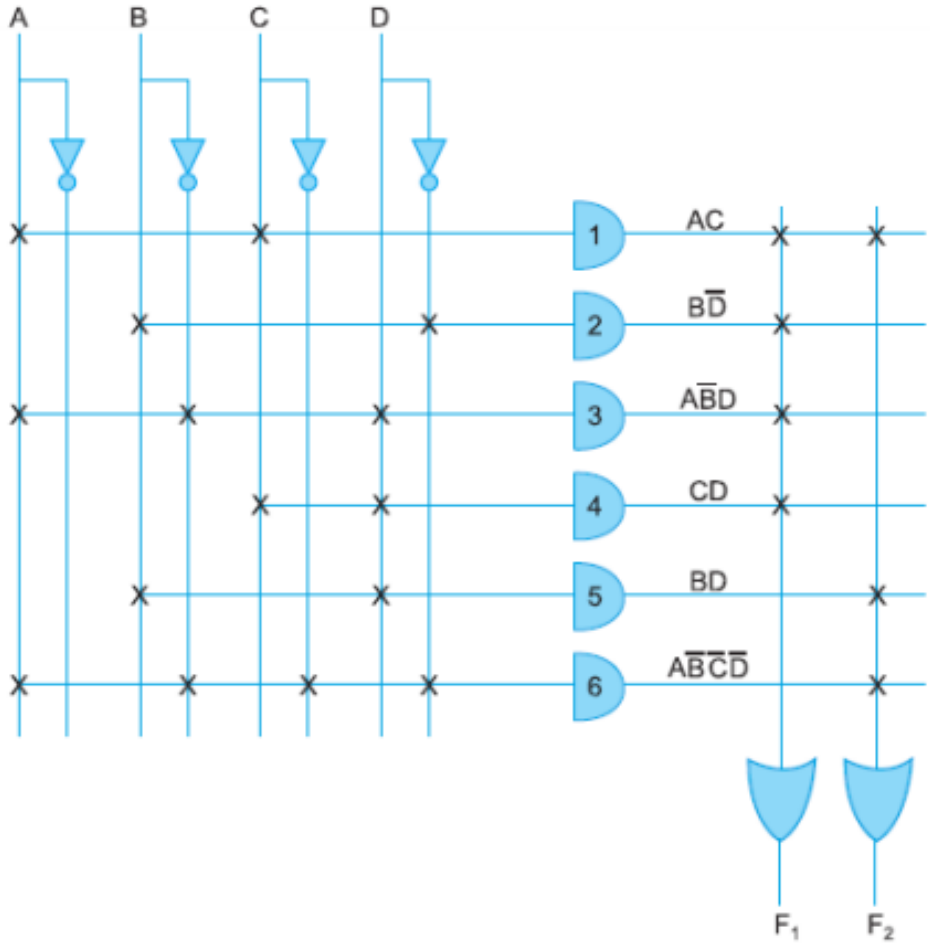
- ✓ Υλοποιεί λογικά κυκλώματα σε διεπίπεδη μορφή AND-OR αθροίσματος γινομένων.
- ✓ Έχει μια προγραμματιζόμενη διάταξη από ANDs συνδυασμένη με μία προγραμματιζόμενη διάταξη από ORs. → Δύο προγραμματιζόμενα επίπεδα λογικής: AND - OR.
- ✓ **Έξοδος πρώτου επιπέδου**: όρος γινομένου οσονδήποτε εισόδων
- ✓ **Έξοδος δευτέρου επιπέδου**: άθροισμα όρων γινομένου πρώτου επίπεδου.
- ✓ Μειονέκτημα: μικρή ταχύτητα λόγω δύο προγραμματιζόμενων επιπέδων

Το πρόβλημα μετριάζεται εάν μόνο το πρώτο επίπεδο (όροι γινομένου) είναι προγραμματιζόμενο.

PLA (παράδειγμα)

$$F_1 = A\,C + B\,\overline{D} + A\,\overline{B}\,D + C\,D$$
$$F_2 = A\,C + B\,D + A\,\overline{B}\,\overline{C}\,\overline{D}$$

Γινόμενα	Είσοδοι				Έξοδοι	
	A	B	C	D	F_1	F_2
$A\,C$	1	–	1	–	1	1
$B\,\overline{D}$	–	1	–	0	1	–
$A\,\overline{B}\,D$	1	0	–	1	1	–
$C\,D$	–	–	1	1	1	–
$B\,D$	–	1	–	1	–	1
$A\,\overline{B}\,\overline{C}\,\overline{D}$	1	0	0	0	–	1

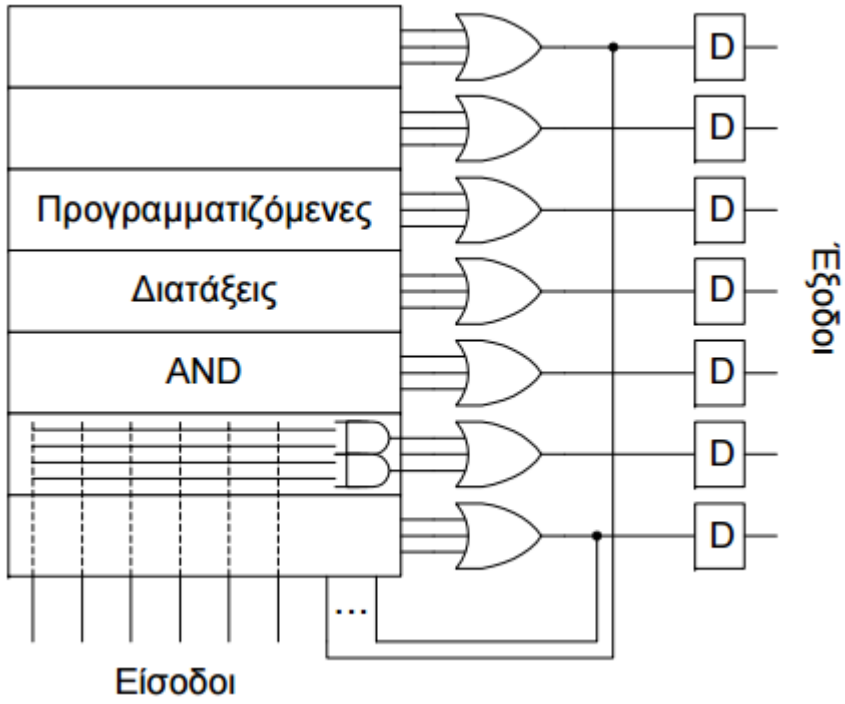


Ο πίνακας προγραμματισμού δείχνει τις επιθυμητές συνδέσεις στις 2 παρατάξεις πυλών

Είδη Προγραμματιζόμενων Συσκευών

PAL (Programmable Array Logic):

- ✓ Μόνο το πρώτο επίπεδο (όροι γινομένου) προγραμματιζόμενο (τα λογικά αθροίσματα είναι σταθερά).
- ✓ Έχουν περιορισμένες δυνατότητες υλοποίησης λογικών κυκλωμάτων
- ✓ Έρχονται σε μεγάλη ποικιλία εισόδων, εξόδων και μεγέθους OR πυλών.
- ✓ Έχουν flip flop στην έξοδο της OR για ακολουθιακά κυκλώματα.



SPLDs (Simple Programmable Logic Devices): όλες οι παρεμφερείς με τις PALs και PLAs συσκευές με μικρό κόστος και μεγάλη ταχύτητα.

PAL (παράδειγμα)

Έστω οι λογικές συναρτήσεις $F_2, F_1,$ και F_0:

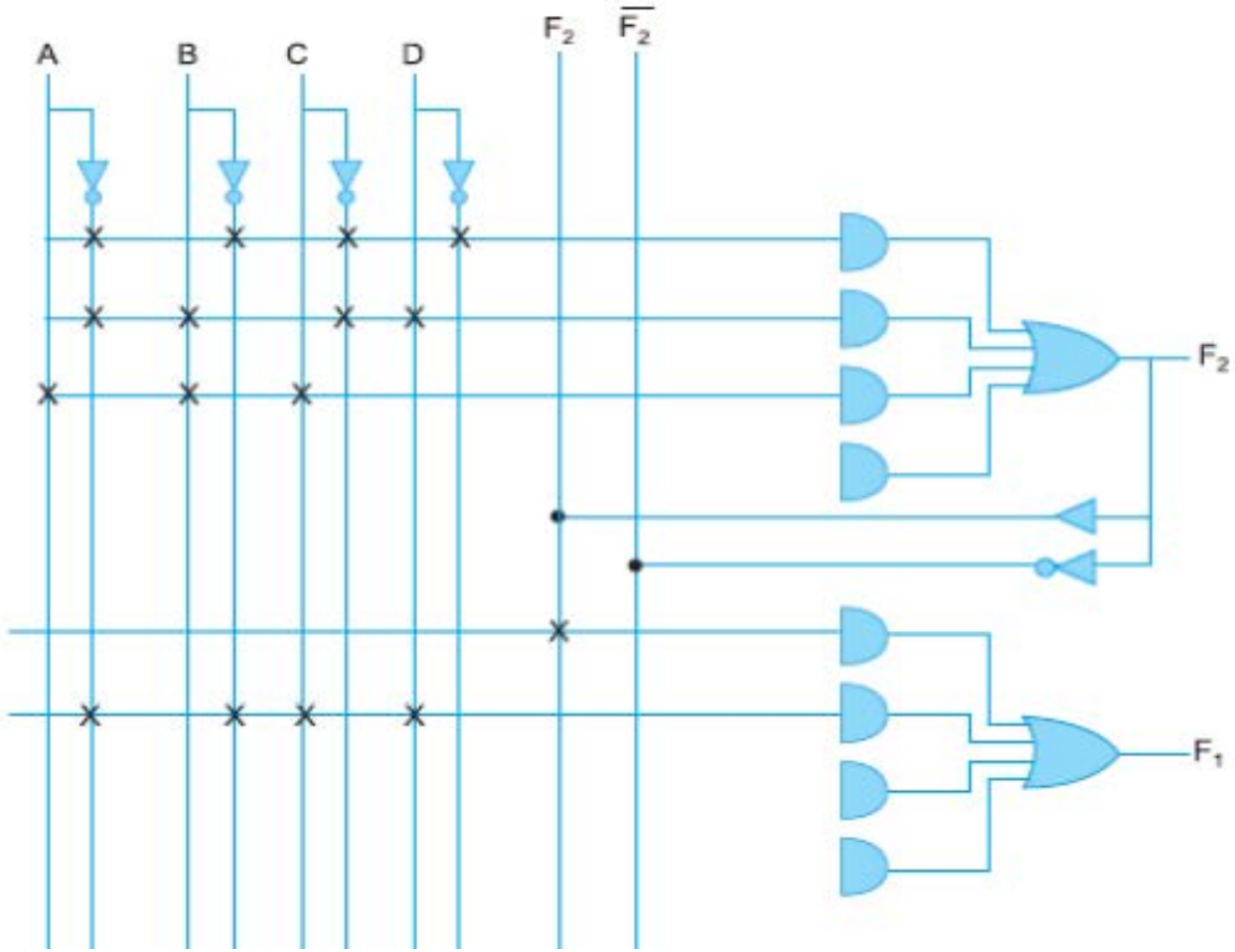
$F_2 = \Sigma(A,B,C,D) = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,B\overline{C}\,D + A\,B\,C$

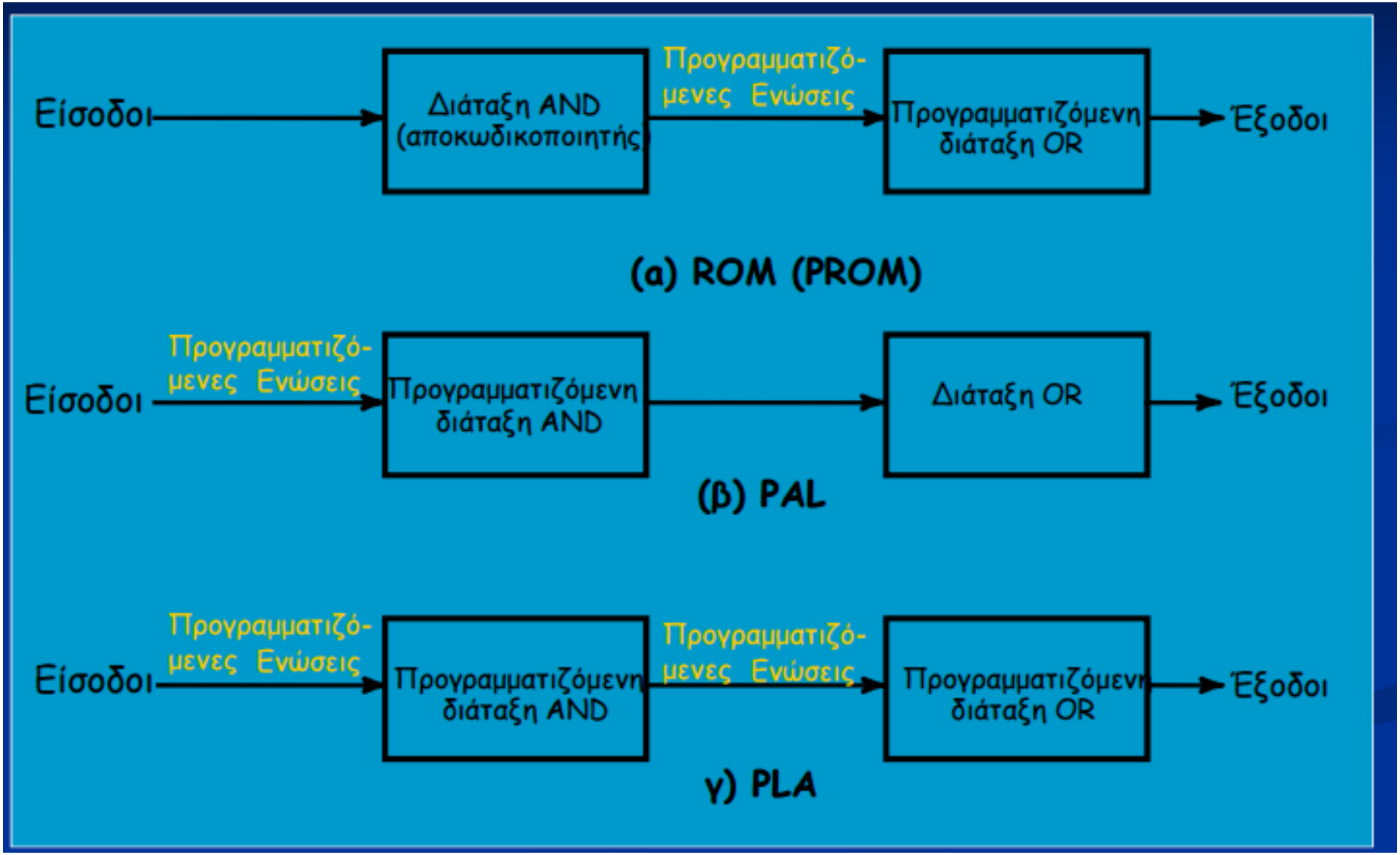
$F_1 = \Sigma(A,B,C,D) = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,B\overline{C}\,D + A\,B\,C + \overline{A}\,\overline{B}\,C\,D$

$F_0 = \Sigma(A,B,C,D) = A\,\overline{C} + \overline{A}\,C + A\,B\,C\,D + A\,\overline{B}\,C\,\overline{D}$

Γινόμενα	Σήματα εισόδου					Σήματα εξόδου
	A	B	C	D	F_2	
$\overline{A}\,\overline{B}\,\overline{C}\,\overline{D}$	0	0	0	0		
$\overline{A}\,B\overline{C}\,D$	0	1	0	1		$F_2 = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,B\overline{C}\,D + A\,B\,C$
$A\,B\,C$	1	1	1	-		
$\overline{A}\,\overline{B}\,C\,D$	0	0	1	1	1	$F_1 = F_2 + \overline{A}\,\overline{B}\,C\,D$
$A\overline{C}$	1	-	0	-		
$\overline{A}\,C$	0	-	1	-		$F_0 = A\,\overline{C} + \overline{A}\,C + A\,B\,C\,D +$
$A\,B\,C\,D$	1	1	1	1		$A\,\overline{B}\,C\,\overline{D}$
$A\,\overline{B}\,C\,\overline{D}$	1	0	1	0		

PAL (παράδειγμα)

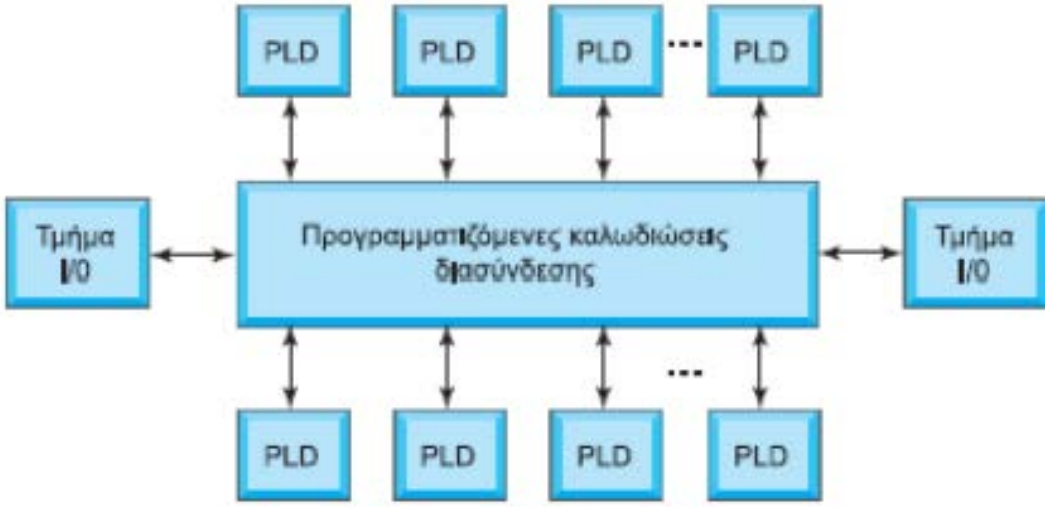



Είσοδοι
Διάταξη AND
(αποκωδικοποιητής)
Προγραμματιζό-
μενες Ενώσεις
Προγραμματιζόμενη
διάταξη OR
Έξοδοι
(α) ROM (PROM)
Προγραμματιζό-
μενες Ενώσεις
Είσοδοι
Προγραμματιζόμενη
διάταξη AND
Διάταξη OR
Έξοδοι
(β) PAL
Προγραμματιζό-
μενες Ενώσεις
Είσοδοι
Προγραμματιζόμενη
διάταξη AND
Προγραμματιζό-
μενες Ενώσεις
Προγραμματιζόμενη
διάταξη OR
Έξοδοι
γ) PLA

CPLDs

Complex Programmable Logic Devices:

- ✓ Σύνολο από μικρές PLDs που συνδέονται με προγραμματιζόμενες καλωδιώσεις
- ✓ Έχουν μεγαλύτερη χωρητικότητα επιφάνεια εφαρμογές.



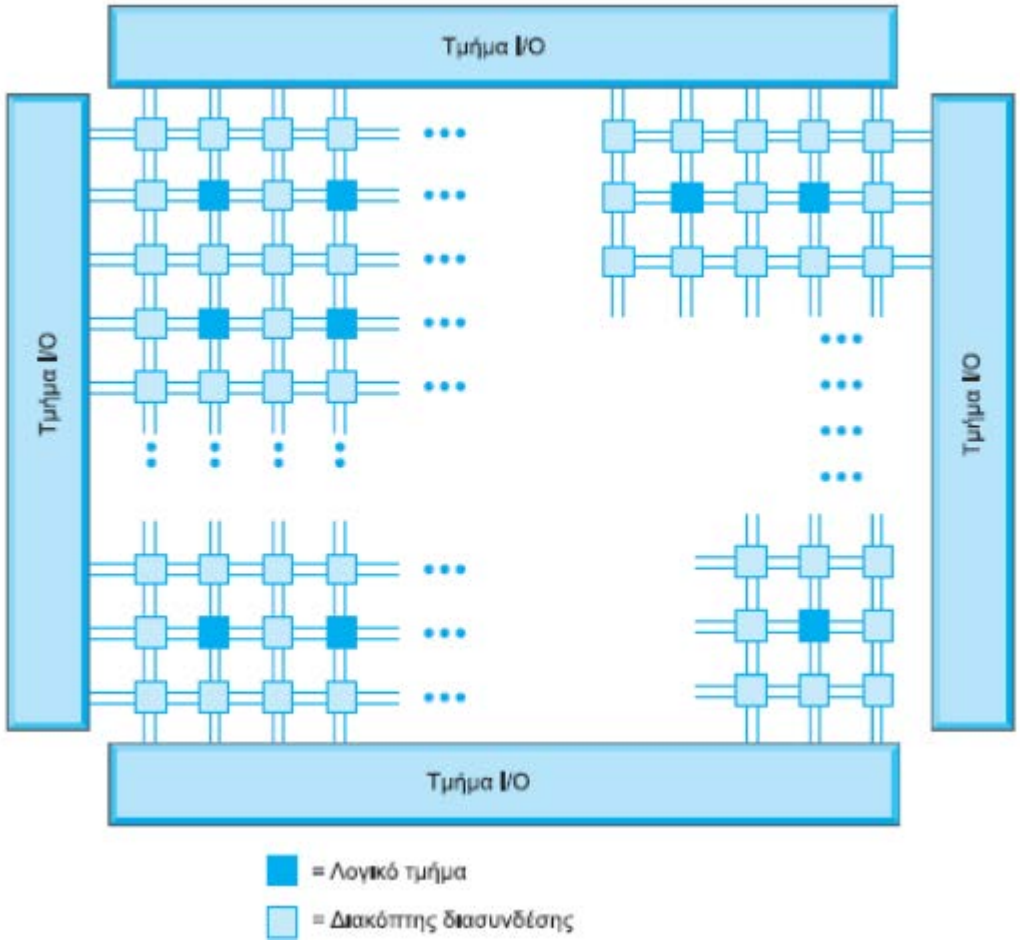
FPGAs

Field Programmable Gate Arrays:

- ✓ Βασίζονται στην αρχιτεκτονική των gate arrays (συστάδες προκατασκευασμένων τρανζίστορ τα οποία διασυνδέονται για να υλοποιήσουν το ζητούμενο κύκλωμα).
- ✓ Τα FPGAs αποτελούνται από μία συστάδα κυκλωματικών στοιχείων (logic blocks) με αρκετές δυνατότητες διασύνδεσης και διαμορφώνονται με προγραμματισμό που γίνεται στο πεδίο εφαρμογής τους.
- ✓ Τα FPGAs είναι οι προγραμματιζόμενες συσκευές που παρέχουν την μεγαλύτερη πυκνότητα ολοκλήρωσης.

FPGAs

Γενική Δομή FPGAs



Προγραμματιζόμενοι Διακόπτες

Όταν προγραμματιστούν καθορίζουν τον τρόπο λειτουργίας του κυκλώματος.

- ✓ Αρχικά χρησιμοποιήθηκαν οι ασφάλειες (fuses).
- ✓ Στα CPLDs χρησιμοποιούνται τα τρανζίστορ ρευστής πύλης (floating gate) που χρησιμοποιούνται και στις μνήμες EPROM και EEPROM
- ✓ Στα FPGAs χρησιμοποιούνται κύτταρα SRAM και (antifuses).

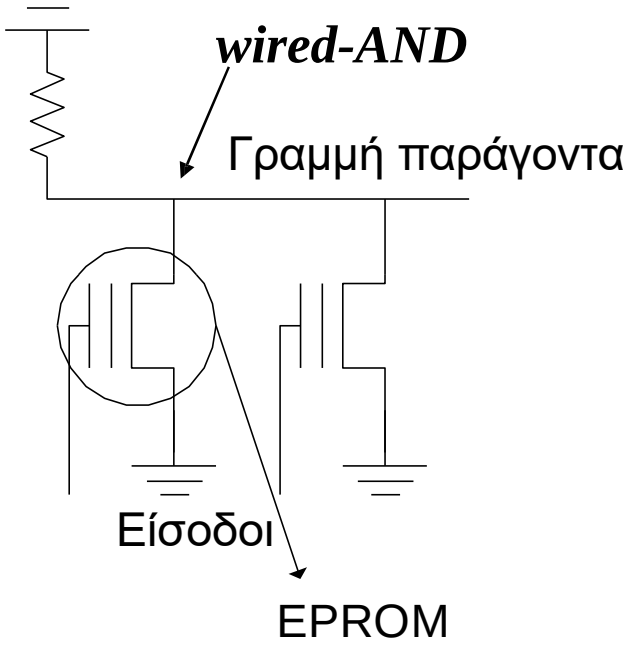
Συστήματα Προγραμματισμού.

- ✓ Χρησιμοποιούνται φθηνά συστήματα, (εξοπλισμός εργαστηρίου).
- ✓ Έχουν ειδικές υποδοχές (*sockets*) στις οποίες τοποθετείται το chip.
- ✓ Επικοινωνούν με σταθμούς εργασίας στους οποίους εκτελείται το κατάλληλο λογισμικό προγραμματισμού.

In System Programming: *Προγραμματισμός μίας συσκευής μετά την τοποθέτηση της στο PCB (για προστασία από πολλαπλές τοποθετήσεις).*

Προγραμματιζόμενοι Διακόπτες

Προγραμματιζόμενοι διακόπτες EPROM, EEPROM.



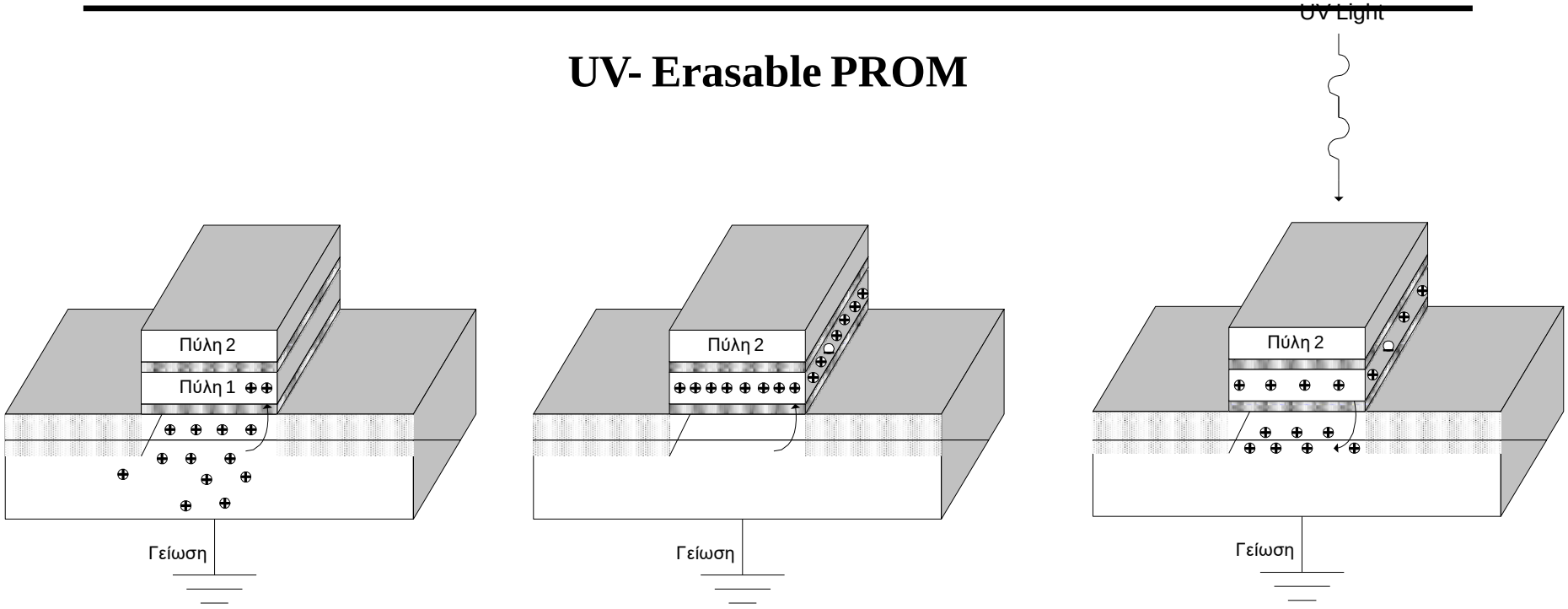
Ανάλογα με τον προγραμματισμό τους βρίσκονται σε δύο καταστάσεις:

1. Ελέγχονται από το δυναμικό της πύλης τους (*gate*).
2. Είναι μονίμως αποκομμένα, ανεξάρτητα από το δυναμικό της πύλης.

Εξαρτάται από τον προγραμματισμό τους.

Προγραμματιζόμενοι Διακόπτες

UV- Erasable PROM



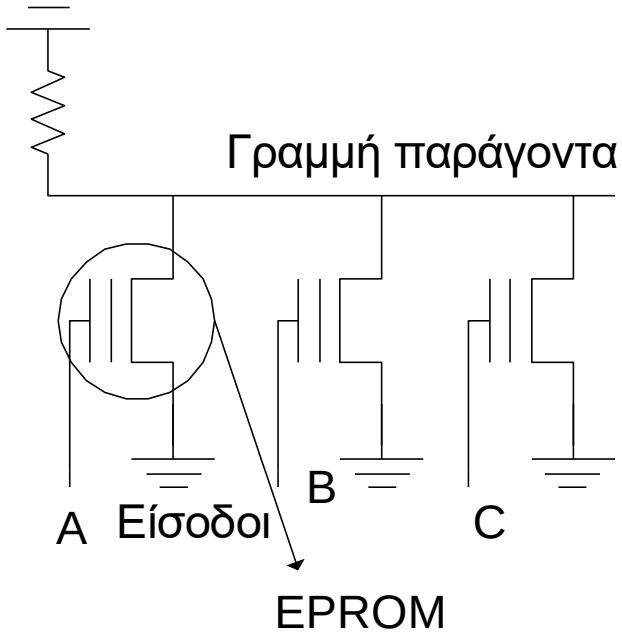
- ✓ Έχουν την δυνατότητα από-προγραμματισμού με υπεριώδη ακτινοβολία.
- ✓ Όταν εφαρμόζουμε τάση μεγαλύτερη από 12V στην υποδοχή του *nmos*, προγραμματίζεται το *transistor*.

Προγραμματιζόμενοι Διακόπτες

UV- Erasable PROM

- ✓ Δημιουργείται υψηλό ηλεκτρικό πεδίο το οποίο ωθεί τα ηλεκτρόνια να υπερπηδήσουν το μονωτικό υλικό και να παγιδευτούν στην κάτω πύλη (*floating*).
- ✓ Ο προγραμματισμός προκαλεί αύξηση της τάσης κατωφλίου πάνω από 5V.
- ✓ Το προγραμματισμένο *transistor* είναι μόνιμα αποκομμένο.
- ✓ Όταν το *transistor* δεν προγραμματιστεί λειτουργεί κανονικά.
- ✓ Με την έκθεση του *transistor* σε υπεριώδη ακτινοβολία τα ηλεκτρόνια επανέρχονται στην θέση τους.
- ✓ Τοποθετείται ένα φωτο-αγώγιμο υλικό στην κορυφή του κεραμικού περιβλήματος.

Προγραμματιζόμενοι Διακόπτες

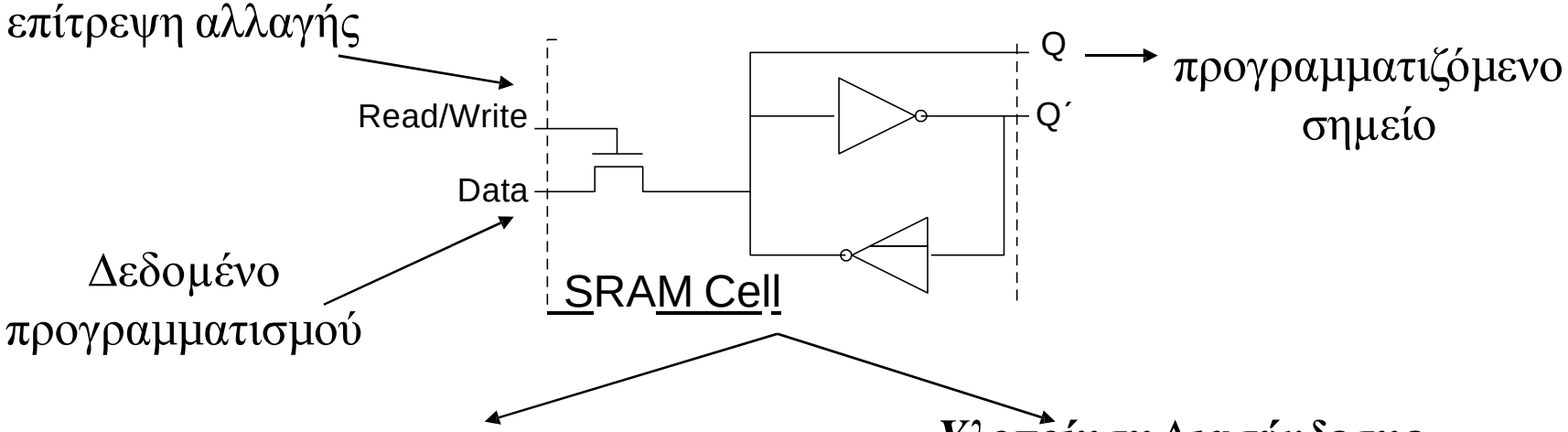


- ✓ Αν όλα τα transistor είναι απρογραμμάτιστα τότε υλοποιείται η συνάρτηση F=(A+B+C)΄.
- ✓ Αν το transistor A είναι προγραμματισμένο τότε υλοποιείται η συνάρτηση F=(B+C)΄. Η είσοδος Α αγνοείται.

Προγραμματιζόμενοι Διακόπτες

Κύτταρα SRAM

Οι συσκευές FPGAs χρησιμοποιούν κύτταρα SRAM και αντιασφάλειες.



Υλοποίηση Λογικής

Θέτουν σε αγωγή *transistor* τα οποία βραχυκυκλώνουν δύο γραμμές.

Υλοποίηση Διασύνδεσης

Γραμμές επιλογής πολυπλεκτών

Μόνιμη διασύνδεση μίας από τις εισόδους του πολυπλέκτη στην έξοδό του .

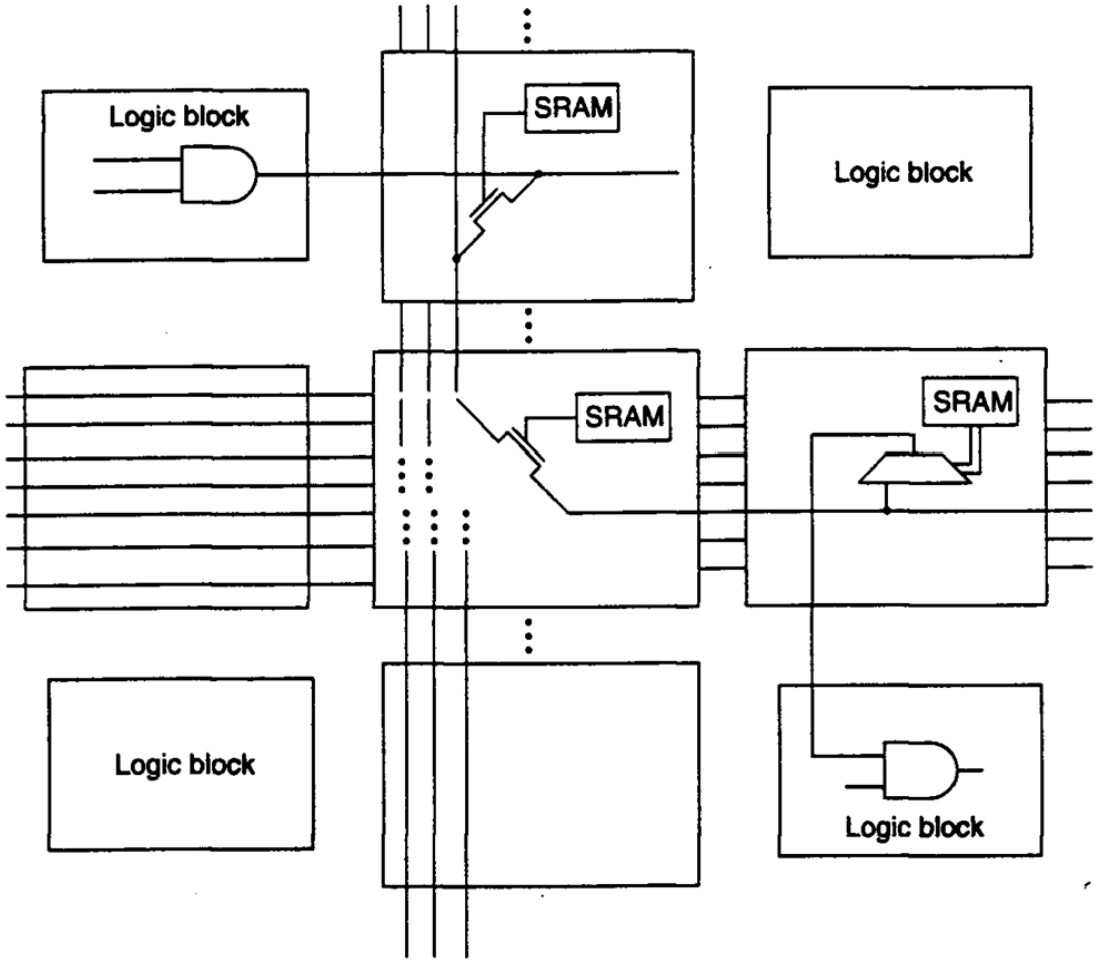
Προγραμματιζόμενοι Διακόπτες

Πλεονέκτημα *SRAM*
Δυνατότητα επαναπρογραμματισμού

Μειονέκτημα SRAM
απώλεια δεδομένων με τροφοδοσία

Downloading από ROM

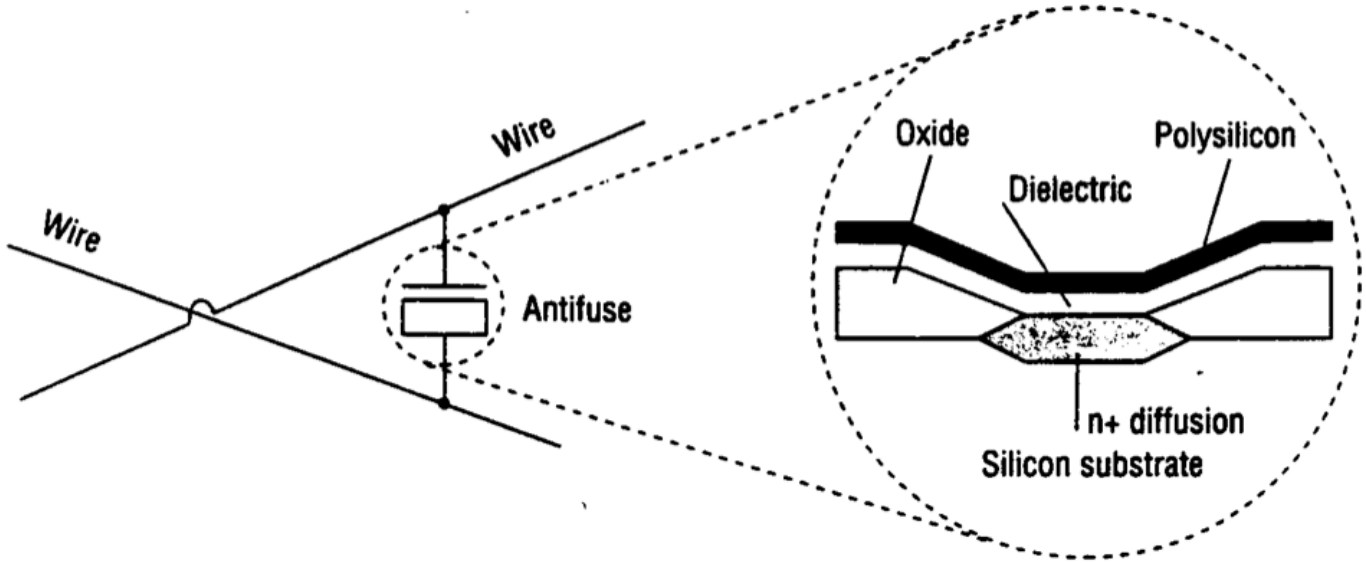
Μπαταρία



Προγραμματιζόμενοι Διακόπτες

Αντιασφάλειες

Οι αντιασφάλειες είναι κυκλώματα τα οποία απρογραμμάτιστα έχουν πολύ υψηλή αντίσταση (ανοικτά κυκλώματα) ενώ όταν προγραμματιστούν αποκτούν πολύ χαμηλή αντίσταση (πχ. αντιασφάλεια PLICE -Programmable Logic Interconnect Circuit Element της Actel)



Η αντίσταση που θα έχει η κάθε αντιασφάλεια μετά τον προγραμματισμό της είναι πολύ σημαντική για την λειτουργία του κυκλώματος .

Αντιασφάλειες

Για τον προγραμματισμό, τροφοδοτούμε με ρεύμα (περίπου 5mA).

Το υψηλό αυτό ρεύμα προκαλεί μεγάλη κατανάλωση ισχύος στην μικρή επιφάνεια της αντιασφάλειας,

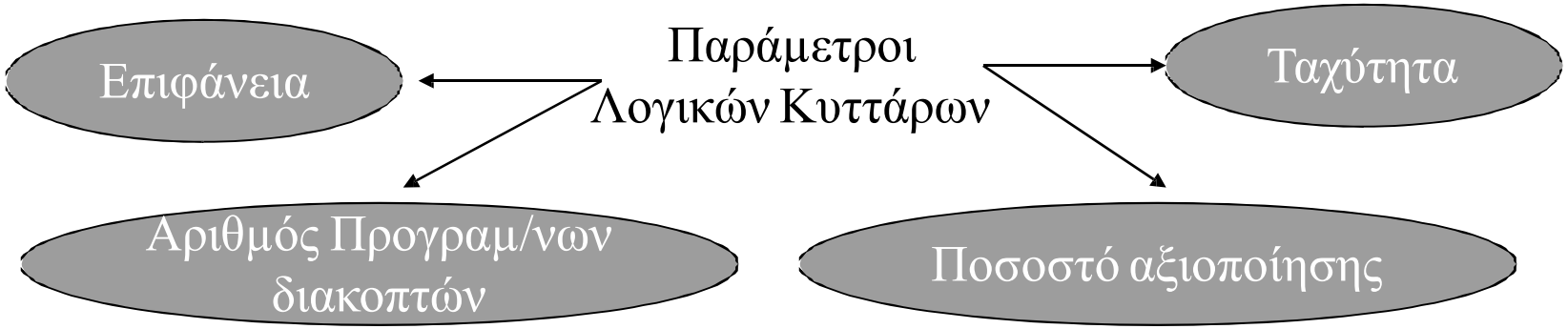
Λιώνει ένα λεπτό μονωτικό υλικό, οπότε δημιουργείται κανάλι αγωγής

- Μία προγραμματιζόμενη συσκευή μπορεί να έχει αρκετές αντιασφάλειες (100.000 – 1.000.000).
- Συνήθως μόνο το 2% από αυτές χρειάζεται να προγραμματιστούν για να προκύψει το τελικό κύκλωμα.
- Οι αντιασφάλειες παρουσιάζουν αρκετά προβλήματα αξιοπιστίας καθώς οι ιδιότητες τους αλλάζουν με την πάροδο του χρόνου..

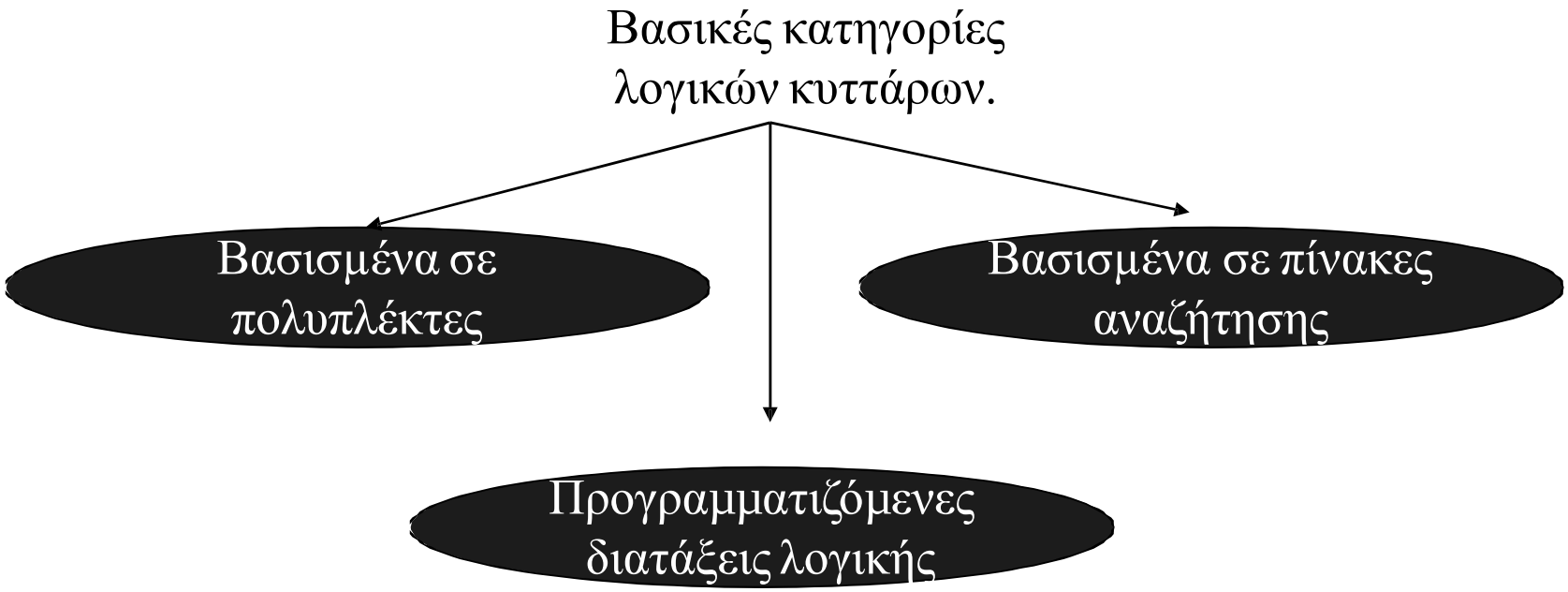
Λογικά Κύτταρα

Βασικό στοιχείο αρχιτεκτονικής μίας προγραμματιζόμενης συσκευής.

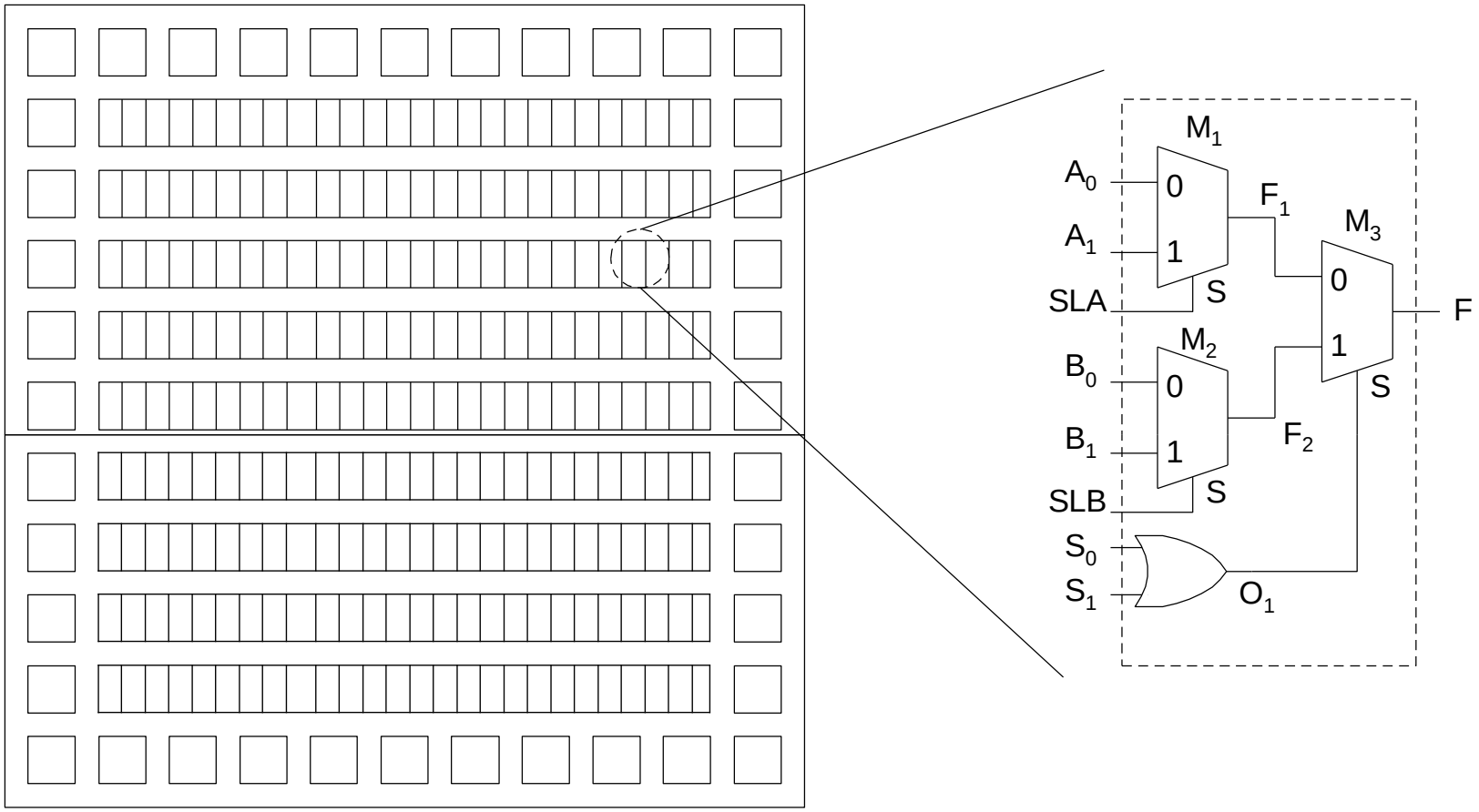
- Το κάθε κύτταρο λογικής αναλαμβάνει να υλοποιήσει ένα τμήμα της λογικής του ζητούμενου κυκλώματος.
- Οι δυνατότητες που έχει το κύτταρο, καθορίζουν πόσο μεγάλο θα είναι το τμήμα λογικής που θα υλοποιήσει.



Κατηγορίες Λογικών Κυττάρων



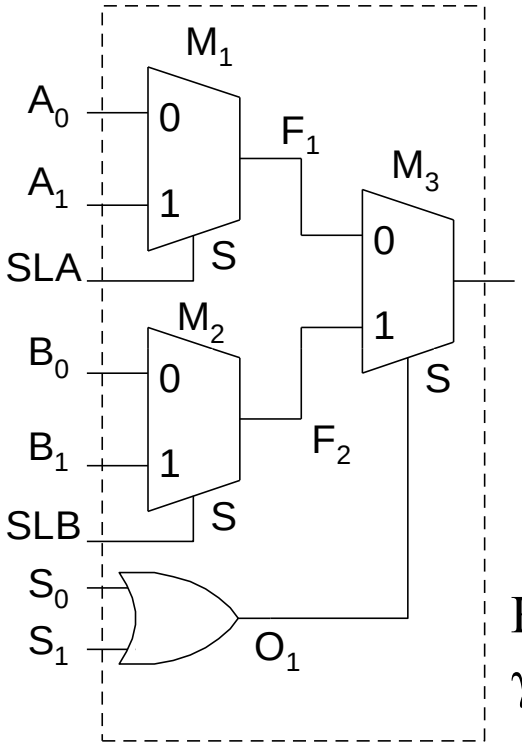
Λογικά Κύτταρα με πολυπλέκτες



Λογικό κύτταρο ACTEL, ACT1

Λογικά Κύτταρα με πολυπλέκτες

Βασική Ιδέα: Διασυνδέουμε τις εισόδους του κυττάρου στα κατάλληλα σήματα, έτσι ώστε η έξοδος να υλοποιεί την επιθυμητή συνάρτηση.



Λογική Συνάρτηση Πολυπλέκτη M_1 :

$F_1=SLA'\cdot A_0+SLA\cdot A_1$

Θεώρημα **Shannon** για συναρτήσεις:

$F(A, B,C, \ldots)=A'\cdot F(A=0, B,C, \ldots) + A\cdot F(A=1, B,C, \ldots)$

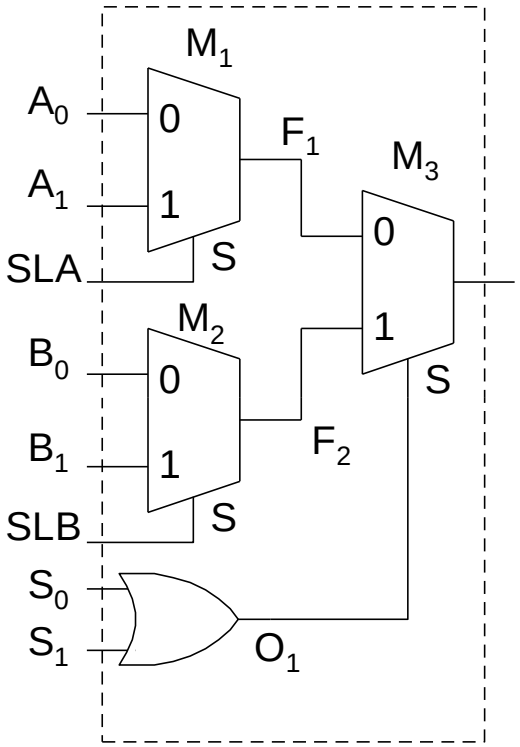
$F=A'\cdot F_{A'}+A\cdot F_A$

Η F υλοποιείται με έναν πολυπλέκτη εάν συνδέσουμε στην γραμμή επιλογής το A, και στις δύο εισόδους τα $F_{A'}$ και F_A.

Αναδρομική εφαρμογή έως ότου φθάσουμε σε τετριμμένες συναρτήσεις εισόδου (0, 1, A, A΄, B, B΄, C, …

Λογικά Κύτταρα με πολυπλέκτες

Παράδειγμα: Υλοποίηση συνάρτησης F=A΄B+AB΄C+ABD με το κύτταρο.



$F=A'\cdot F_{A'}+A\cdot F_A$ με

$F_{A'}=1\cdot B+0\cdot B'\cdot C+0\cdot B\cdot D=B$

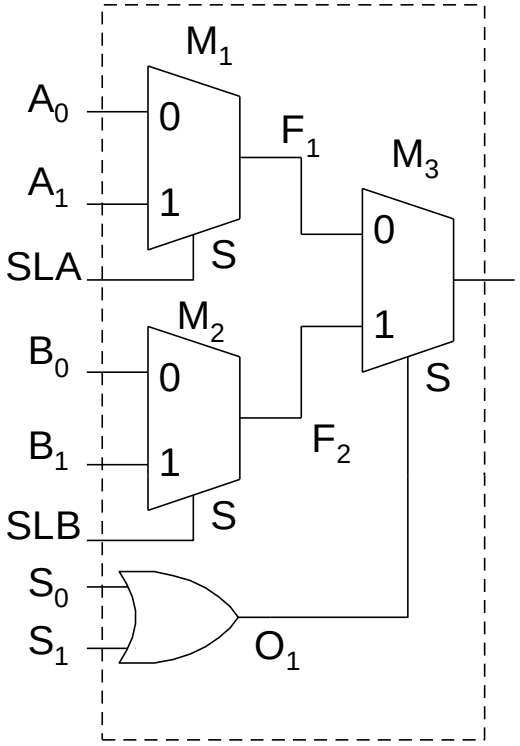
$F_A=0\cdot B+1\cdot B'\cdot C+1\cdot B\cdot D=B'\cdot C+B\cdot D$

Υλοποίηση F με M_3:

- Είσοδος 0: $F_{A'}$
- Είσοδος 1: F_A
- Είσοδος S: A=> $S_1=A$, $S_0=0$

Υλοποίηση $F_{A'}$ με M_1, Υλοποίηση F_A με M_2

Λογικά Κύτταρα με πολυπλέκτες



$F_{A'}=B=B'\cdot 0+B\cdot 1$

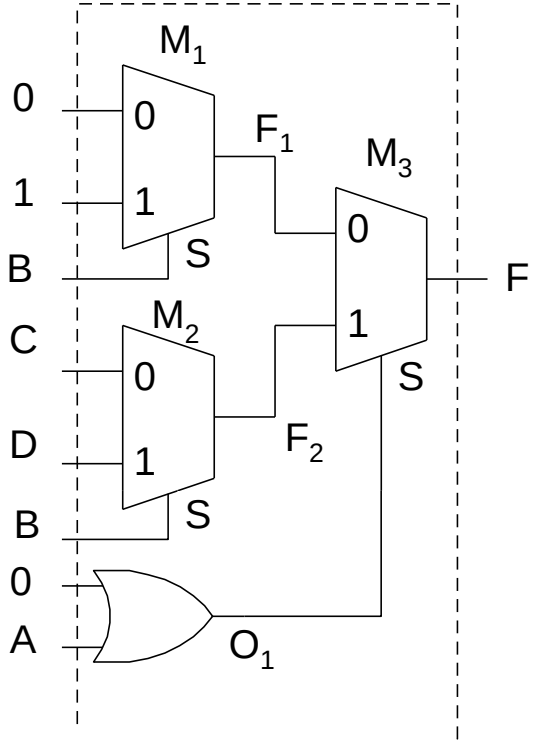
Υλοποίηση $F_{A'}$ με M_1:

- Είσοδος 0: 0
- Είσοδος 1: 1
- Είσοδος S: B

$F_A=B'C+BD$

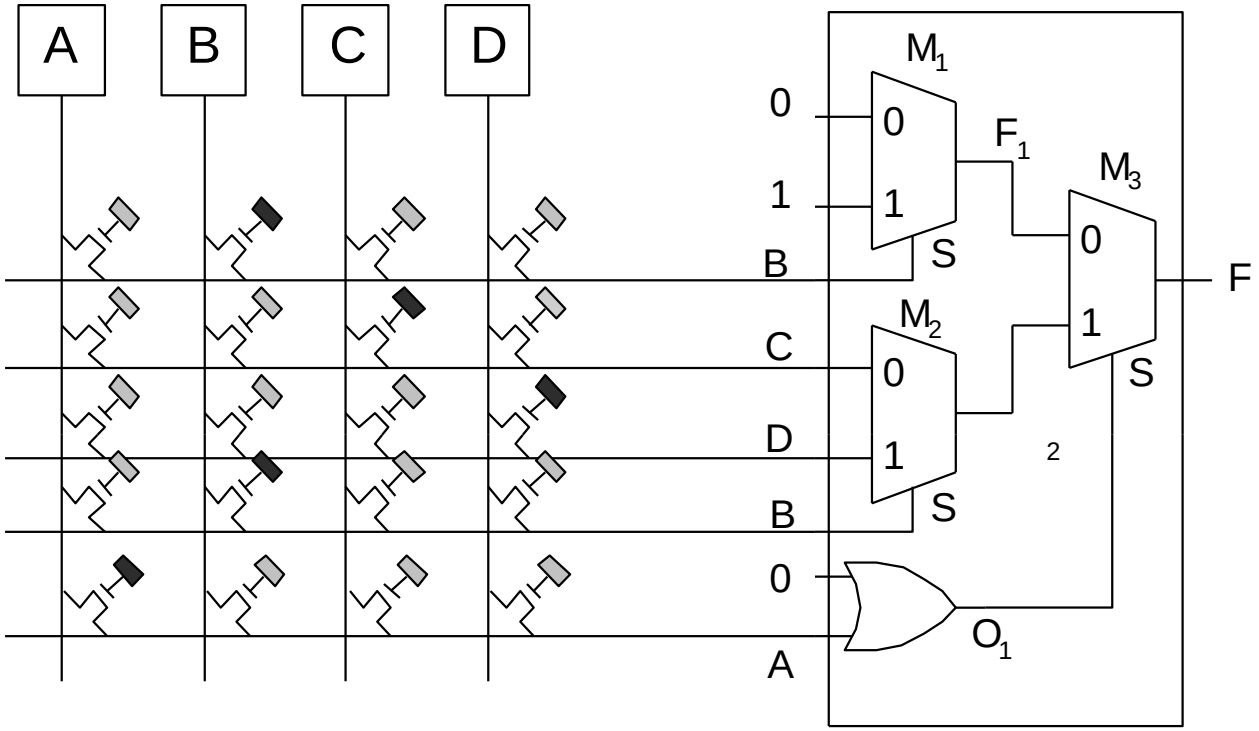
Υλοποίηση F_A με M_2:

- Είσοδος 0: C
- Είσοδος 1: D
- Είσοδος S: B



Μπορεί να υλοποιήσει όλες τις συναρτήσεις δύο εισόδων, τις περισσότερες τριών εισόδων και μερικές τεσσάρων εισόδων

Λογικά Κύτταρα με πολυπλέκτες



Λογικά Κύτταρα με πολυπλέκτες

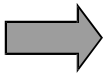
- Η διαδικασία είναι αρκετά χρονοβόρα και δύσκολη για έναν άνθρωπο, αλλά τετριμμένη για έναν υπολογιστή.
- Το λογισμικό συνήθως γνωρίζει πως μπορεί να υλοποιήσει διάφορες συναρτήσεις βάσης, ακολουθώντας πίνακες:

Συνάρτηση	Είσοδος 0	Είσοδος 1	Επιλογή
not(A)	**1**	**0**	**A**
AB	**0**	**B**	**A**
AB´	**A**	**0**	**B**
A´B	**B**	**0**	**A**
A+B	**B**	**1**	**A**
A+B´	**1**	**A**	**B**
A´+B	**1**	**B**	**A**
…	**…**	**…**	**…**

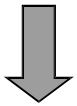
Λογικά Κύτταρα με πολυπλέκτες

- Στόχος κάθε εργαλείου σχεδίασης με προγραμματιζόμενες συσκευές είναι η μεγιστοποίηση της λογικής που υλοποιείται από κάθε κύτταρο.
- Η αρχιτεκτονική επιλογή του κυττάρου, γίνεται με βάση την απόδοση του στην υλοποίηση συναρτήσεων.
- Η απόδοση κάθε κυττάρου στην γενικότερη προγραμματιζόμενη συσκευή μπορούν να μετρηθούν με τα ***benchmarks***.

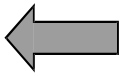
Δύο συνδυαστικά λογικά κύτταρα μπορούν να διασυνδεθούν για να υλοποιήσουν ένα ακολουθιακό κύτταρο.



Επιβάρυνση από την χρήση δικτύου διασύνδεσης για υλοποίηση απλών κυττάρων.

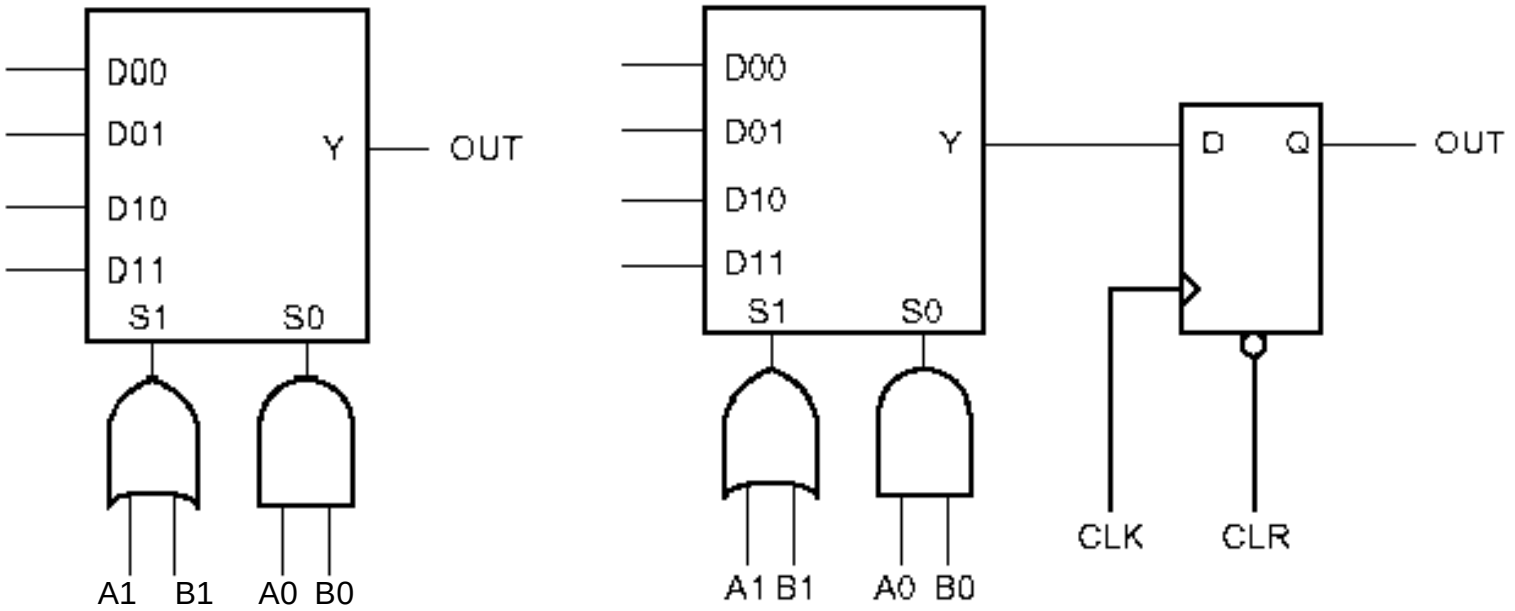


Συμπεριλαμβάνουμε απλά ακολουθιακά στοιχεία μέσα στα λογικά κύτταρα .



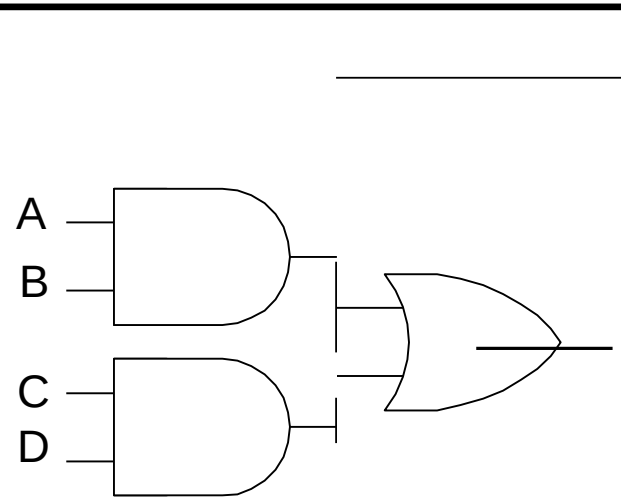
Αύξηση πολυπλοκότητας κυττάρου.

Λογικά Κύτταρα με πολυπλέκτες



- Τέτοιες αρχιτεκτονικές, ονομάζονται *fine-grain*.
- Βοηθούν πολύ στην επίτευξη πολύ υψηλής αξιοποίησης των δυνατοτήτων υλοποίησης λογικής (πάνω από 90%)
- Διευκολύνουν το λογισμικό στην αντιστοίχιση της σχεδιαζόμενης λογική.

Λογικά Κύτταρα με Πίνακες Αναζήτησης



A	B	C	D	F
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

LUT

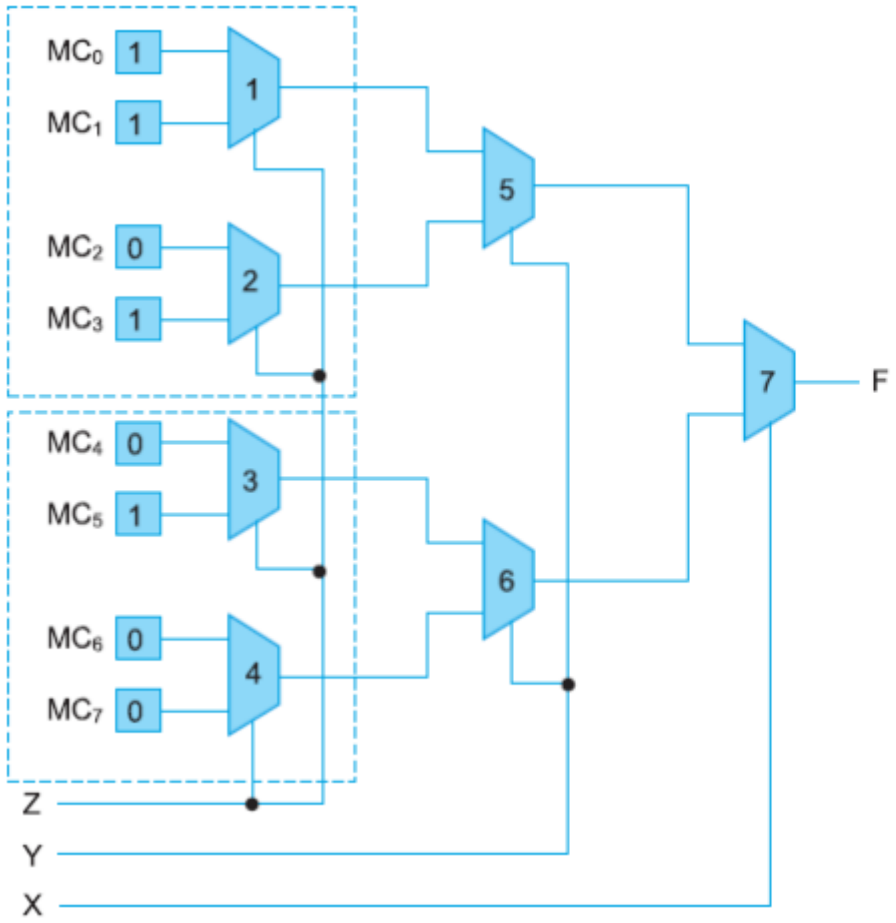
Πλεονέκτημα: Σταθερή καθυστέρηση υπολογισμού μίας συνάρτησης, ανεξάρτητα της πολυπλοκότητας (μειονέκτημα σε απλές συναρτήσεις).

Λογικά Κύτταρα με Πίνακες Αναζήτησης
Παράδειγμα

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

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

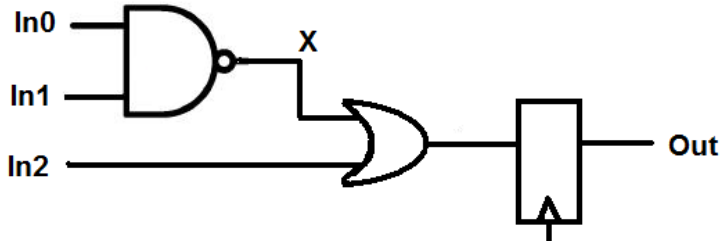
Λογικά Κύτταρα με Πίνακες Αναζήτησης Παράδειγμα

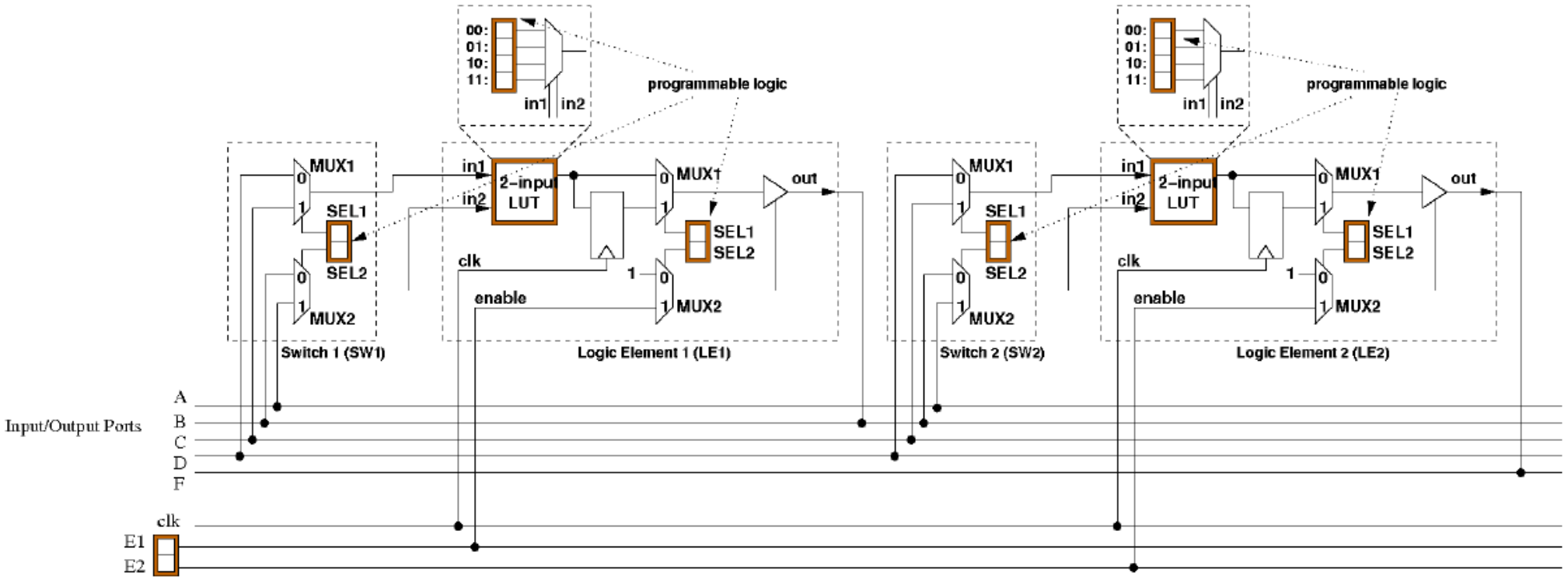


Η z επιλέγει ανάμεσα στο άνω και στο κάτω επίπεδο πολυπλεκτών, ηy επιλέγει μεταξύ των πολυπλεκτών ενός επιπέδου και η x επιλέγει μεταξύ των τιμών κάθε πολυπλέκτη

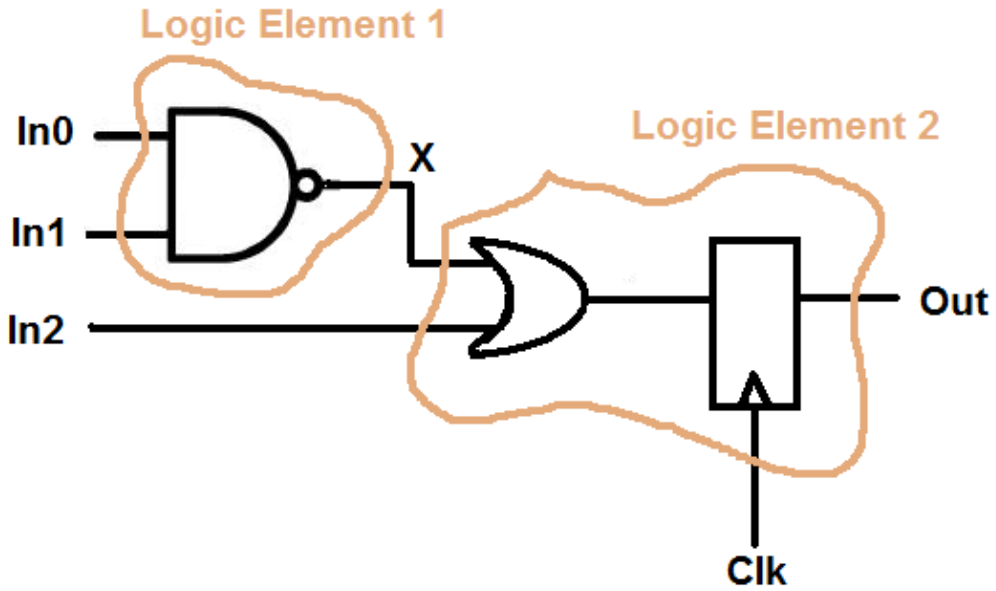
Παράδειγμα αντιστοίχισης κυκλώματος σε FPGA

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





Παράδειγμα αντιστοίχισης κυκλώματος σε FPGA

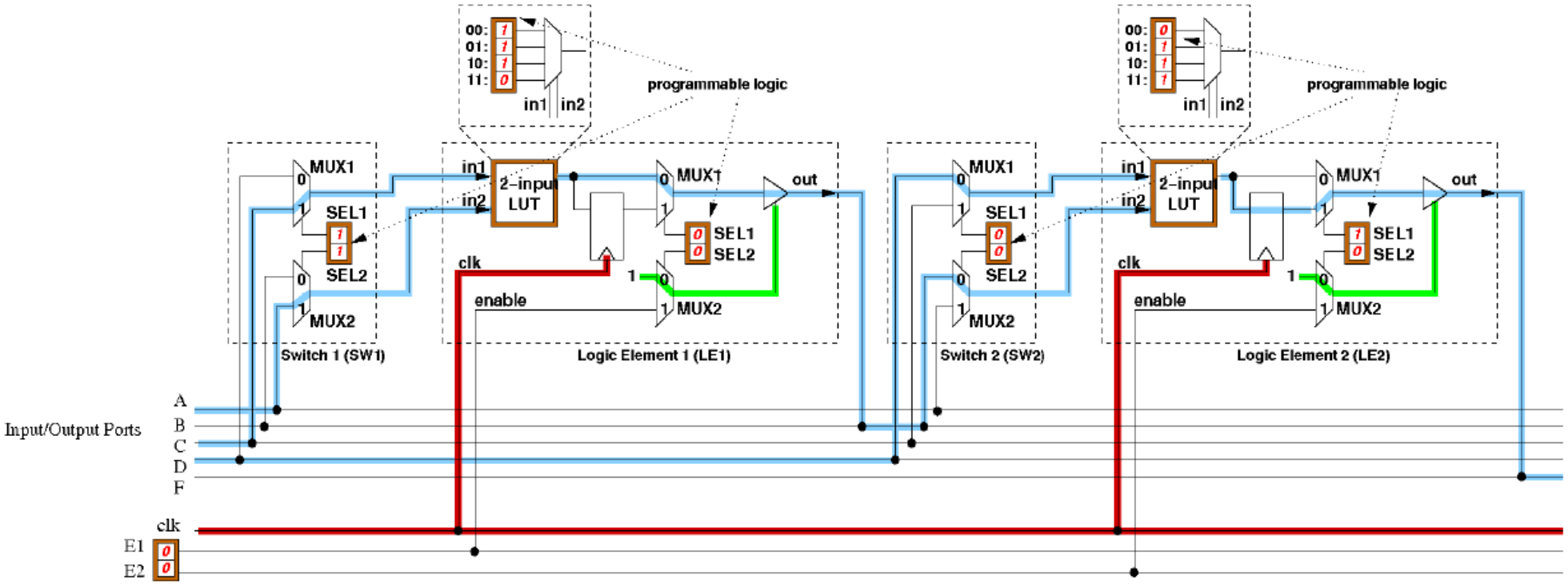


LUTs: Το LUT του logic element 1 προγραμματίζεται ώστε να εκτελεί τη λειτουργία της πύλης NAND. Το LUT του logic element 2 προγραμματίζεται ώστε να εκτελεί τη λειτουργία της πύλης OR.

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

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

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


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