
**ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ**
**ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ**

Ψηφιακή Σχεδίαση

# Εργαστήριο 1:
# Λογική με Διακόπτες, Πολυπλέκτες, Μνήμη ROM

Μανόλης Γ.Η. Κατεβαίνης

Τμήμα Επιστήμης Υπολογιστών

# Άδειες Χρήσης

•Το παρόν εκπαιδευτικό υλικό υπόκειται στην άδεια χρήσης **Creative Commons** και ειδικότερα

***Αναφορά – Μη εμπορική Χρήση – Όχι Παράγωγο Έργο 3.0 Ελλάδα***
***(Attribution – Non Commercial – Non-derivatives 3.0 Greece)***

***CC BY-NC-ND 3.0 GR***

•Εξαιρείται από την ως άνω άδεια υλικό που περιλαμβάνεται στις διαφάνειες του μαθήματος, και υπόκειται σε άλλου τύπου άδεια χρήσης. Η άδεια χρήσης στην οποία υπόκειται το υλικό αυτό αναφέρεται ρητώς.

# Χρηματοδότηση

•Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
•Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Κρήτης**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
•Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο

ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

**Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης**

# Ψηφιακή Σχεδίαση

# Εργαστήριο 1:

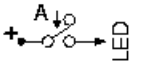
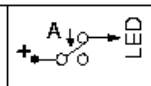
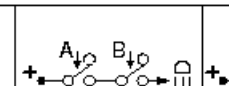
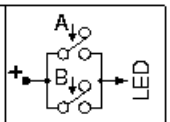
# Λογική με Διακόπτες, Πολυπλέκτες, Μνήμη ROM

[Βιβλία: *προαιρετικά* μπορείτε να διαβάσετε: Wakerly: § 2.15.1 (σελ. 68-70), § 5.7.0 (σελ. 471-472)· ή Mano (4η έκδοση): § 1.7.μέρος (σελ. 25), § 4.11.μέρος (σελ. 152-154)]

## 1.1 Λογικές Πράξεις και Πίνακας Αληθείας

Το εργαστήριο αυτό αποτελεί συνέχεια και εμβάθυνση του εργ. 0, και ιδιαίτερα του πώς κυκλώματα διακοπτών υλοποιούν τις λογικές πράξεις **ΟΧΙ** (§0.9), **ΚΑΙ** (§0.10), **Ή** (§0.11). Αν δεν προλάβατε να κάνετε τα αντίστοιχα πειράματα, ή να τα καταλάβετε σε βάθος, ξεκινήστε σήμερα με εκείνα, ή μελετήστε πρώτα εκείνες τις σημειώσεις γιά να τις καταλάβετε σε βάθος --αποτελούν τη βάση γιά όλο το μάθημα.

Ο πίνακας που ακολουθεί αποτελεί μιά περίληψη των παραπάνω παραγράφων 0.6 - 0.8. Γιά πληρότητα, προσθέσαμε και το κύκλωμα της §0.8 με το όνομα "ταυτότητα" (identity), αν και συνήθως δεν θεωρούμε ενδιαφέρουσα αυτήν την τόσο απλή λογική πράξη. Στον πίνακα αυτόν χρησιμοποιούμε τον όρο "OFF" όταν μεν πρόκειται γιά διακόπτες γιά να σημαίνει "ελεύθερος" (όχι πατημένος), όταν δε πρόκειται γιά LED γιά να σημαίνει "σβηστή"· αντίστροφα, ο όρος "ON" γιά μεν τους διακόπτες σημαίνει "πατημένος" γιά δε τις LED σημαίνει "αναμμένη".

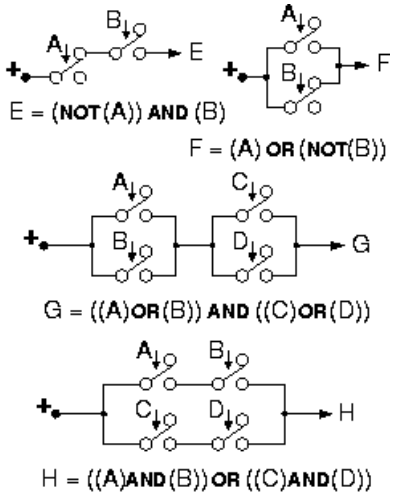
| Διακόπτης **A** | Διακόπτης **B** |  **IDENTITY** (ταυτοτητα) |  **NOT** (οχι) |  **AND** (και) |  **OR** (ή) |
|---|---|---|---|---|---|
| OFF (ελευθερος) | OFF | OFF (σβηστη) | ON (αναμενη) | OFF | OFF |
| | ON | | | OFF | ON |
| ON (πατημενος) | OFF | ON (αναμενη) | OFF (σβηστη) | OFF | ON |
| | ON | | | ON | ON |

Ο πίνακας αυτός δίνει αναλυτικά τη συμπεριφορά των κυκλωμάτων και των αντίστοιχων "Λογικών Συναρτήσεων" (πράξεων) γιά τις διάφορες περιπτώσεις εισόδων τους, δηλαδή κατάστασης των διακοπτών· τέτοιους πίνακες τους λέμε *Πίνακες Αληθείας (truth tables)*. Γιά το κύκλωμα της ταυτότητας, ο πίνακας αληθείας μας λέει ότι η έξοδός του (φωτοβολία της LED) βρίσκεται πάντα σε ευθεία αντιστοιχία με την είσοδό του (κατάσταση του διακόπτη Α): OFF γιά OFF και ON γιά ON. Γιά τη λογική πράξη ΟΧΙ (NOT), ο πίνακας μας δείχνει ότι η έξοδός της είναι πάντα το αντίστροφο (ανάποδο) της εισόδου της Α, εξ' ου και το όνομα του κυκλώματος αυτού που συχνά λέγεται "αντιστροφέας" (inverter). Γιά την λογική πράξη ΚΑΙ (AND), η έξοδός της είναι συνάρτηση των δύο εισόδων της, Α και Β: είναι ON μόνον στον έναν από τους 4 συνδυασμούς τιμών των Α και Β, όταν Α και Β είναι ON· αυτό προκύπτει όταν οι διακόπτες είναι συνδεδεμένοι "εν σειρά". Τέλος, η λογική συνάρτηση Ή (OR) είναι ON όταν Α είναι ON ή Β είναι ON (ή και οι δύο είναι ON)· αυτό προκύπτει όταν οι διακόπτες είναι συνδεδεμένοι "εν παραλλήλω".

Οι λογικές πράξεις **AND** και **OR** είναι ανάλογες με πλήθος παρόμοιων εννοιών της καθημερινής μας ζωής, π.χ.:

- Γιά να τρέξει νερό από τη βρύση πρέπει να ανοίξουμε τη βρύση **και** να είναι ανοικτός και ο γενικός διακόπτης νερού του διαμερίσματος.
- Θα υπάρχει υπερκατανάλωση νερού αν έχουμε διαρροή στο καζανάκι **ή** αν στάζει η βρύση του μπάνιου **ή** της κουζίνας (ή και περισσότερα από ένα από αυτά ταυτόχρονα).
- Γιά να μπώ στο αυτοκίνητό μου που βρίσκεται στο γκαράζ πρέπει να ανοίξω την πόρτα του γκαράζ **και** την πόρτα του αυτοκινήτου.
- Μπορώ να μπώ στο εξοχικό μου σπίτι από την κύρια πόρτα **ή** από την πίσω πόρτα του κήπου.
- Από τους περισσότερους κοινούς λογαριασμούς τράπεζας μπορεί να κάνει ανάληψη ο δικαιούχος Α **ή** ο δικαιούχος Β. Υπάρχουν όμως και λογαριασμοί, π.χ. εταιρειών, όπου γιά να γίνει ανάληψη πρέπει να υπογράψουν π.χ. **και** ο διευθυντής **και** ο ταμίας.

## 1.2 Κυκλώματα γιά Σύνθετες Λογικές Πράξεις




Οι παραπάνω βασικές συνδεσμολογίες διακοπτών μπορούν να συνδυαστούν μεταξύ τους σε οσοδήποτε σύνθετους συνδυασμούς· στο σχήμα δεξιά φαίνονται μερικά παραδείγματα (στα σημεία E, F, G, H, υποτίθεται ότι συνδέονται LED's μεσω αντιστάσεων). Στο πρώτο κύκλωμα, γιά να περάσει ρεύμα προς το E, πρέπει "όχι A πατημένος", επειδή η σύνδεση έγινε στον πάνω δεξιά ακροδέκτη (T0) του A, **ΚΑΙ** (σύνδεση εν σειρά) "B πατημένος", επειδή το E συνδέεται στον κάτω δεξιά ακροδέκτη (T1) του B. Στο δεύτερο κύκλωμα, θα περνάει ρεύμα προς το F όποτε "A πατημένος" (σύνδεση κάτω δεξιά (T1) του A), **Ή** (σύνδεση εν παραλλήλω) "όχι B πατημένος" (σύνδεση πάνω δεξιά (T0) του B). Στο τρίτο κύκλωμα, γιά να περάσει ρεύμα προς το G πρέπει να υπάρχει δρόμος μέσω του αριστερού ζεύγους διακοπτών **ΚΑΙ** μέσω του δεξιού ζεύγους· από το αριστερό ζεύγος μπορεί να περάσει ρεύμα όποτε "A πατημένος" **Ή** "B πατημένος"· ομοίως από το δεξί όποτε "C πατημένος" **Ή** "D πατημένος". Τέλος, στο κύκλωμα που τροφοδοτεί το H, ρεύμα μπορεί να περάσει από το επάνω ζευγάρι διακοπτών **Ή** από το κάτω· γιά να περάσει από επάνω πρέπει "A πατημένος" **ΚΑΙ** "B πατημένος", αντίστοιχα δε από κάτω.

## <u>Πειράματα 1.3:</u> Λογική με Διακόπτες

Σε αυτό εδώ το πείραμα, **πριν** φτάσετε στο εργαστήριο:

i. Σχεδιάστε τα παρακάτω κυκλώματα που σας ζητώνται. Οι διακόπτες A, B, C είναι όπως αυτοί που είδαμε στην § 0.8, και το κάθε κύκλωμα οδηγεί μιά LED μέσω μιάς αντίστασης όπως στην § 0.7.
ii. Φτιάξτε τον πίνακα αληθείας του κάθε κυκλώματος, όπως είδαμε παραπάνω στην § 1.1. Αφού τα κυκλώματα αυτά έχουν 3 εισόδους (A, B, C), οι πίνακες αληθείας τους θα έχουν 8 γραμμές: μία τετράδα γιά A=OFF και μία τετράδα γιά A=ON· σε κάθε τετράδα, θα υπάρχει ένα ζευγάρι γιά B=OFF και ένα γιά B=ON· και σε κάθε ζευγάρι, θα υπάρχει μία γραμμή γιά C=OFF και μία γιά C=ON.

iii. Διατυπώστε γραπτώς τη συνθήκη γιά την άρνηση της κάθε εξόδου, δηλαδή γιά το πότε η φωτοδίοδος θα είναι σβηστή ή το πότε (1) η μπανιέρα δεν γεμίζει, ή (2) δεν μπορώ να μπώ στο σπίτι, ή (3) δεν μπορώ να φύγω με το αυτοκίνητο, ή (4) δεν θα πάω το αυτοκίνητο στο συνεργείο. Διατυπώστε τη συνθήκη με λόγια, χρησιμοποιώντας την καθημερινή μας λογική, και δείτε ότι εφαρμόζεται πάλι η αρχή του δυϊσμού των παραγράφων 0.10 και 0.11· διασταυρώστε την ορθότητα της συνθήκης με τον πίνακα αληθείας.

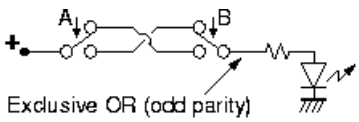
Στο εργαστήριο, κατασκευάστε τα κυκλώματα, και ελέγξτε τα. Γιά να ελεγχθεί πλήρως το κάθε κύκλωμα πρέπει να τού εφαρμόσετε καθέναν από τους 8 συνδυασμούς εισόδων που έχει ο πίνακας αληθείας, και να διαπιστώσετε ότι η LED κάνει το σωστό γιά καθέναν· καθώς τα ελέγχετε, διασταυρώστε την ορθότητα της συνθήκης γιά την άρνηση της κάθε εξόδου.

1. Η LED να ανάβει όταν **θα γεμίσει η μπανιέρα**, δηλαδή όταν:
   - Έχω κλείσει την τάπα (διακόπτης Α πατημένος), **ΚΑΙ**
   - ανοίγω τη βρύση του κρύου (διακόπτης Β πατημένος) **Ή** ανοίγω αυτήν του ζεστού (διακόπτης C πατημένος).
2. Η LED να ανάβει όταν **μπορώ να μπώ στο σπίτι**, δηλαδή όταν:
   - έχω το κλειδί της κεντρικής πόρτας (Α πατημένος), **Ή**
   - έχω το κλειδί της μπαλκονόπορτας (Β πατημένος) **ΚΑΙ** το πατζούρι της μπαλκονόπορτας είναι ανοικτό (C πατημένος).
3. Η LED να ανάβει όταν **μπορώ να φύγω με το αυτοκίνητο**, δηλαδή όταν:
   - έχω το κλειδί του γκαράζ (Α πατημένος), **ΚΑΙ**
   - έχω το κλειδί του αυτοκινήτου (Β πατημένος), **ΚΑΙ**
   - **ΔΕΝ** υπάρχει βλάβη στη μηχανή.

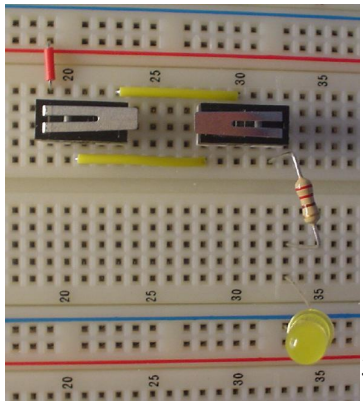
   Όποτε υπάρχει βλάβη στη μηχανή, θα πατιέται ο διακόπτης C.
4. Η LED να ανάβει όταν **πρέπει να πάω το αυτοκίνητο στο συνεργείο**, δηλαδή όταν:
   - υπάρχει βλάβη στη μηχανή (Α πατημένος), **Ή**
   - πέρασαν 12 μήνες από το προηγούμενο service (Β πατημένος) **ΚΑΙ ΔΕΝ** έχω μείνει "πανί-με-πανί".

   Όποτε έχω μείνει "πανί-με-πανί", θα πατιέται ο διακόπτης C.

## <u>Πείραμα 1.4:</u> Διακόπτες "Aller-Retour" - Αποκλειστικό Ή, Έλεγχος Ισότητας




Παρατηρήστε ότι σε όλα τα παραπάνω κυκλώματα υπάρχουν ορισμένες καταστάσεις μερικών από τις εισόδους οι οποίες κάνουν την έξοδο "αναίσθητη" στις (ανεξάρτητη από τις) άλλες εισόδους. Γιά παράδειγμα, στο κύκλωμα ΚΑΙ, όταν ο ένας διακόπτης είναι ελεύθερος, η LED παραμένει σβηστή ό,τι και να κάνει ο άλλος διακόπτης· στο κύκλωμα Ή, όταν ένας διακόπτης είναι πατημένος, η LED ανάβει ό,τι και να κάνει ο άλλος διακόπτης.

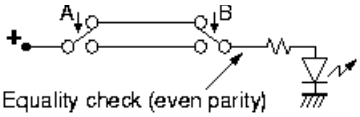


Σκεφτείτε τώρα τις κρεβατοκάμαρες των σπιτιών, όπου οι διακόπτες γιά τα φώτα είναι συνήθως τύπου "aller-retour", δηλαδή σε όποια κατάσταση και να έχει μείνει ο ένας διακόπτης (π.χ. της πόρτας), ο άλλος διακόπτης (π.χ. του κρεβατιού) μπορεί πάντα να αλλάξει την κατάσταση του φωτός (να το ανάψει ή να το σβήσει). Αυτό γίνεται με ένα από τα δύο κυκλώματα που φαίνονται δεξιά.

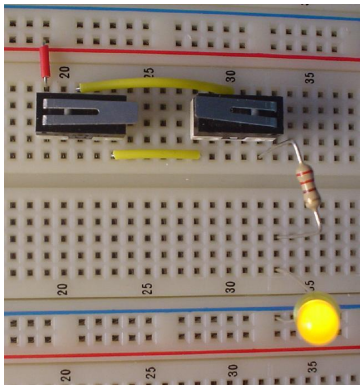
Το πρώτο κύκλωμα υλοποιεί τη λογική πράξη

"**αποκλειστικό Ή**" (exclusive OR - XOR), διότι η LED ανάβει όταν είναι πατημένος *αποκλειστικά ο ένας* από τους δύο διακόπτες και *όχι και οι δύο μαζί*. Παρατηρήστε το κύκλωμα (όπου τα δύο χιαστί σύρματα διασταυρώνονται χωρίς να κάνουν επαφή μεταξύ τους): ρεύμα μπορεί να περάσει προς την LED είτε όταν ο Α είναι πατημένος και ο Β δεν είναι (σύρμα από κάτω αριστερά προς πάνω δεξιά), είτε όταν ο Α δεν είναι πατημένος και ο Β είναι (σύρμα από πάνω αριστερά προς κάτω δεξιά). Επομένως, η LED ανάβει όταν ((Α)**ΚΑΙ**(**ΟΧΙ**(Β))) **Ή** ((**ΟΧΙ**(Α))**ΚΑΙ**(Β)), δηλαδή είναι πατημένος *(μόνο ο Α)* ή *(μόνο ο Β)*.

Το δεύτερο κύκλωμα υλοποιεί τη λογική πράξη **ελέγχου ισότητας** (equality check), διότι η LED ανάβει μόνον όταν οι δύο διακόπτες είναι στην ίδια (ίση) κατάσταση --και οι δύο πατημένοι ή και οι δύο ελεύθεροι. Ρεύμα μπορεί να περάσει προς την LED είτε από το επάνω σύρμα (και οι δύο διακόπτες ελευθεροι) είτε από το κάτω σύρμα (και οι δύο διακόπτες πατημένοι), επομένως η LED ανάβει όταν ((**ΟΧΙ**(Α))**ΚΑΙ**(**ΟΧΙ**(Β))) **Ή** ((Α)**ΚΑΙ**(Β)), δηλαδή όταν Α και Β είναι *(και οι δύο OFF)* ή *(και οι δύο ON)*.




**Πριν φτάσετε στο εργαστήριο**, γράψτε τον πίνακα αληθείας των εξόδων των δύο κυκλωμάτων. Παρατηρήστε ότι, ανεξαρτήτως της κατάστασης του ενός διακόπτη, ο άλλος μπορεί πάντα, ανοιγοκλείνοντας, να αναβοσβήσει το φώς. **Στο εργαστήριο**, φτιάξτε τα δύο κυκλώματα, δείξτε τα στο βοηθό σας, και επαληθεύστε πειραματικά τις παραπάνω ιδιότητες.



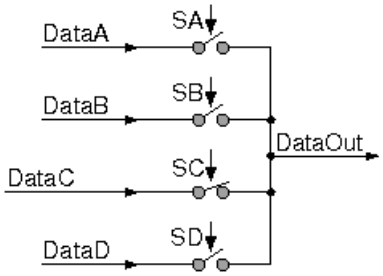
**Περιττή και Άρτια Ισοτιμία** (Odd and Even Parity): η γενίκευση της συνάρτησης αποκλειστικού-Ή σε περισσότερες εισόδους είναι η *περιττή ισοτιμία* (odd parity), η δε γενίκευση της συνάρτησης ελέγχου ισότητας είναι η *άρτια ισοτιμία* (even parity). Όταν έχουμε πολλές εισόδους --πολλούς διακόπτες-- μετράμε το πλήθος τους που είναι πατημένοι (ON) σε δεδομένη στιγμή. Εάν το πλήθος αυτό είναι αριθμός περιττός (μονός - μη ακέραιο πολλαπλάσιο του 2), τότε λέμε ότι έχουμε περιττή ισοτιμία, και η αντίστοιχη συνάρτηση είναι ON (αναμμένη), αλλιώς η συνάρτηση αυτή είναι OFF (σβηστή). Αντίστροφα, όταν το πλήθος των εισόδων που είναι ON είναι αριθμός άρτιος (ζυγός - ακέραιο πολλαπλάσιο του 2), η συνάρτηση άρτιας ισοτιμίας είναι ON, αλλιώς είναι OFF. Επομένως, οι συναρτήσεις περιττής και άρτιας ισοτιμίας είναι πάντα η μία το αντίστροφο (το λογικό ΟΧΙ) της άλλης (άρα αρκεί να ξέρουμε τη μία τους γιά να βρίσκουμε άμεσα και την άλλη).

Εάν ένας από τους διακόπτες αλλάξει κατάσταση, το πλήθος των εισόδων που είναι ON αλλάζει κατά ένα (+1 ή -1): αν ο διακόπτης που άλλαξε ήταν σβηστός (OFF) και άναψε (ON), το πλήθος αυξήθηκε κατά 1, ενώ αν ήταν αναμμένος και έσβησε τότε το πλήθος μειώθηκε κατά 1. Οτιδήποτε από τα δύο και να συμβαίνει, η ισοτιμία αλλάζει από άρτια σε περιττή ή από περιττή σε άρτια! Βλέπουμε λοιπόν ότι διατηρείται η βασική ιδιότητα με την οποία ξεκινήσαμε: σε οιαδήποτε κατάσταση και να βρίσκονται οι διακόπτες, αρκεί οιοσδήποτε ένας από αυτούς να αλλάξει κατάσταση γιά να αλλάξει τιμή η έξοδος (από σβηστή να ανάψει ή από αναμμένη να σβήσει). Η ιδιότητα αυτή αποτελεί τη βάση γιά την κύρια εφαρμογή των συναρτήσεων ισοτιμίας:

Οι συναρτήσεις ισοτιμίας χρησιμοποιούνται σαν η απλούστερη μορφή **κώδικα ανίχνευσης σφαλμάτων** (error detection codes): αφού μεταδώσουμε μέσα από ένα

τηλεπικοινωνιακό δίκτυο κάμποσες πληροφορίες (κάμποσα σύρματα, καθένα "αναμμένο" ή "σβηστό"), μεταδίδουμε στο τέλος ακόμα μία επιπλέον πληροφορία που είναι π.χ. η άρτια ισοτιμία όλων των προηγουμένων. Συνήθως, οι πληροφορίες που μεταδίδουμε φτάνουν στην άλλη άκρη όλες σωστές· σε σπάνιες περιπτώσεις, λόγω θορύβου, μία από τις πληροφορίες μπορεί να φτάσει λάθος (ON αντί OFF, ή OFF αντί ON)· σε πολύ σπανιότερες περιπτώσεις μπορεί δύο η περισσότερες πληροφορίες να φτάσουν λάθος (π.χ. αν η πιθανότητα ενός λάθους είναι μία στις χίλιες, και αν τα λάθη είναι ανεξάρτητα μεταξύ τους, τότε η πιθανότητα δύο λαθών είναι περίπου μία στο εκατομμύριο (περίπου το τετράγωνο της πιθανότητας ενός λάθους)). Εάν συμβεί ένα λάθος στη μετάδοση, τότε θα αλλάξει η ισοτιμία των πληροφοριών που στείλαμε, ακριβώς λόγω της παραπάνω ιδιότητας: οιεσδήποτε και αν είναι οι πληροφορίες, η αλλαγή μίας οιασδήποτε από αυτές αλλάζει την ισοτιμία. Αν όμως αλλάξει η ισοτιμία, θα το καταλάβουμε, διότι η επιπλέον πληροφορία ισοτιμίας που στείλαμε δεν θα συμπίπτει πλέον με την ισοτιμία που βλέπει ο παραλήπτης! Έτσι επιτυγχάνεται ο στόχος της ανίχνευσης σφαλμάτων στις περιπτώσεις που συμβαίνει μόνο ένα σφάλμα, που είναι και οι πιό συχνές. Άπαξ και διαπιστωθεί η ύπαρξη σφάλματος, η διόρθωση του μπορεί να γίνει π.χ. με μία αίτηση αναμετάδοσης ("ξαναπές το --δεν άκουσα καλά"). Ο κύριος περιορισμός των συναρτήσεων ισοτιμίας στην ανίχνευση σφαλμάτων είναι ότι αυτές ανιχνεύουν μόνο 1, 3, 5, κλπ. σφάλματα, ενώ τους διαφεύγουν 2, 4, 6, κλπ. σφάλματα. Εάν τα σφάλματα είναι ανεξάρτητα μεταξύ τους, και εάν η πιθανότητα ενός σφάλματος είναι πολύ μικρή, τότε 2 ή περισσότερα "μαζεμένα" σφάλματα είναι πολύ σπάνια. Εάν όμως τα σφάλματα είναι συχνά ή δεν είναι ανεξάρτητα μεταξύ τους, τότε χρειαζόμαστε άλλους, πιό πολύπλοκους κώδικες γιά την ανίχνευσή τους· τέτοια περίπτωση "ομοβροντίας (εκρηκτικών) σφαλμάτων" (burst errors) έχουμε π.χ. όταν ένα κινητό τηλέφωνο, κινούμενο, περνάει γιά λίγο πίσω από μία "σκιά" της κεραίας του σταθμού βάσης· αρκετά όμως είπαμε γιά τώρα --γιά περισσότερες πληροφορίες θα πρέπει να πάρετε κάποιο μάθημα τηλεπικοινωνιών και κωδικοποίησης....

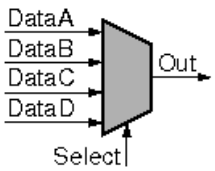
## 1.5 Πολύπλεξη: Επιλογή Πληροφορίας μεταξύ Πολλαπλών Εισόδων



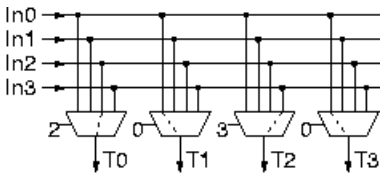

Ένας από τους βασικότερους δομικούς λίθους όλων των ψηφιακών συστημάτων είναι ο *πολυπλέκτης* (*multiplexor*, ή εν συντομία *mux*). Είναι ένα κύκλωμα που επιλέγει μία από τις πολλές εισόδους του, και μεταφέρει (αντιγράφει) την τιμή (κατάσταση) της στην έξοδό του. Δεξιά φαίνεται ένα παράδειγμα πολυπλέκτη *4-σε-1* (4-to-1 mux). Έχει τέσσερεις "Εισόδους Δεδομένων", DataA, DataB, DataC, και DataD, και μία έξοδο (δεδομένων) DataOut. Κάθε φορά, μία και μόνο μία από τις εισόδους μεταφέρεται (αντιγράφεται) και εμφανίζεται στην έξοδο, DataOut. Είναι η ίδια λειτουργία όπως το ραδιόφωνο ή η τηλεόραση, που στη μεν κεραία τους φτάνουν τα σήματα από όλους τους σταθμούς που εκπέμπουν στην περιοχή, αλλά που στη συνέχεια το ραδιόφωνο ή η τηλεόραση, χρησιμοποιώντας τη μέθοδο του συντονισμού, επιλέγει (συντονίζεται σε) έναν από αυτούς τους σταθμούς γιά να περάσει το σήμα του στην "έξοδο" --το μεγάφωνο ή την οθόνη.

Το ποιά είσοδος επιλέγεται μπορεί να αλλάζει "δυναμικά", δηλαδή την ώρα της λειτουργίας του κυκλώματος, και γιά να γίνεται η επιλογή αυτή ο πολυπλέκτης έχει άλλες, επιπλέον εισόδους, τις *εισόδους ελέγχου* (control inputs) ή *εισόδους επιλογής* (selection inputs), που εδώ είναι οι SA, SB, SC, και SD. Στο απλό κύκλωμα πολυπλέκτη που δείχνουμε εδώ, η επιλογή εισόδου γίνεται με τον εξής τρόπο: ανά

πάσα στιγμή, ένας και μόνον ένας από τους διακόπτες SA, SB, SC, SD πρέπει να είναι πατημένος. Αυτός που είναι πατημένος ενώνει την είσοδό δεδομένων του με την έξοδο, "μεταφέροντας" έτσι ό,τι "πληροφορία" υπάρχει σε αυτή την είσοδο προς την έξοδο. Στο σχήμα εδώ, αυτή τη στιγμή, ο διακόπτης SC είναι πατημένος, και γι' αυτό η έξοδος DataOut ισούται αυτή τη στιγμή με την είσοδο DataC. Η κατάσταση των υπολοίπων εισόδων *δεν* μπορεί να επηρεάσει την έξοδο, διότι εκείνες είναι ασύνδετες με την έξοδο. (Η περιγραφή αυτή της λειτουργίας δεν είναι ακριβής από ηλεκτρική άποψη, αλλά θα την κάνουμε ακριβέστερη στην επόμενη παράγραφο).



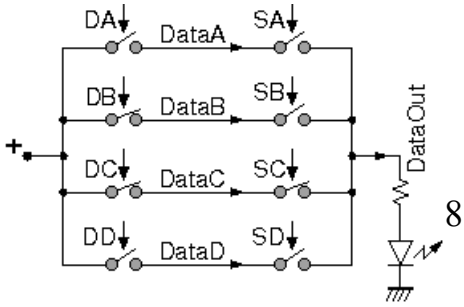

Το σύμβολο του πολυπλέκτη είναι ένα τραπέζιο, όπως φαίνεται στο δεύτερο σχήμα δεξιά γιά το παραπάνω παράδειγμα πολυπλέκτη 4-σε-1. Γιά συντομία, όλες οι είσοδοι επιλογής (ελέγχου) σημειώνονται με μία μόνο γραμμούλα, "Select"· σε επόμενο μάθημα θα πούμε περισσότερα πάνω σε αυτό το θέμα. Μία σημαντικότατη εφαρμογή των πολυπλεκτών είναι στις μνήμες, όπως θα δούμε αμέσως μετά, σε αυτό το εργαστήριο.




Μία δεύτερη, εξ ίσου σημαντική εφαρμογή των πολυπλεκτών είναι στην καρδιά κάθε δικτύου επικοινωνίας, εκεί όπου επιλέγεται το ποιός θα μιλήσει με ποιόν κάθε στιγμή. Αυτό μπορεί να είναι στο εσωτερικό ενός υπολογιστή ή άλλου ψηφιακού συστήματος, ή σ' έναν δρομολογητή (router) ή μεταγωγέα (switch) ενός δικτύου υπολογιστών ή τηλεπικοινωνιών. Στο σχήμα δεξιά φαίνεται ένα απλό παράδειγμα μεταγωγέα 4x4 τύπου *σταυραγωγού (crossbar)* εμπνευσμένου από την κλασσική τηλεφωνία. Είναι κατασκευασμένος από 4 πολυπλέκτες, καθένας μεγέθους 4-σε-1. Μπορούμε να φανταστούμε ότι οι 4 είσοδοι, *In0-In3*, είναι τα μικρόφωνα των τηλεφώνων τεσσάρων σπιτιών, οι δε 4 έξοδοι, *T0-T3*, τροφοδοτούν τα ακουστικά τεσσάρων άλλων τηλεφώνων. Ελέγχοντας κατάλληλα τα σήματα επιλογής των πολυπλεκτών πραγματοποιούμε τις τηλεφωνικές συνδέσεις που μας ζητώνται. Στο παράδειγμα, η είσοδος *In2* έχει ζητήσει να μιλήσει στην έξοδο *T0*, και η *In3* στην *T2*. Η είσοδος *In0* μεταδίδεται σε περισσότερους από έναν ακροατές --στους *T1* και *T3*-- κάτι γνωστό σαν *multicasting* στα δίκτυα. Η είσοδος *In1* είναι αδρανής προς στιγμήν --ή τουλάχιστο, κανείς δεν την ακούει.

Ακόμα μία εφαρμογή των πολυπλεκτών είναι στη *σειριακή* (serial) μετάδοση δεδομένων, στα δίκτυα επικοινωνίας. Στο πρώτο σχήμα παραπάνω, έστω ότι έχουμε τέσσερις πληροφορίες στις τέσσερις εισόδους DataA έως DataD, και θέλουμε να τις στείλουμε σε κάποιον μακρυά, μέσω ενός και μόνο σύρματος, του DataOut (τα 4 σύρματα θα κόστιζαν πολύ, λόγω απόστασης). Γιά να το πετύχουμε, αρκεί να αλλάξουμε διαδοχικά τα σήματα επιλογής του πολυπλέκτη από το SA στο SB, στο SC, και στο SD, οπότε οι 4 πληροφορίες από τις 4 εισόδους θα τοποθετηθούν διαδοχικά πάνω στο σύρμα DataOut και θα μεταδοθούν "σειριακά" (με τη σειρά) στον παραλήπτη. Βεβαίως, γιά να μπορέσει ο παραλήπτης να τα καταλάβει σωστά, πρέπει να υπάρχει μιά πρόσθετη συμφωνία (σύμβαση) *χρονισμού*, δηλαδή να ξέρει σε ποιά χρονική στιγμή μεταδίδεται η καθεμία από τις τέσσερις πληροφορίες.

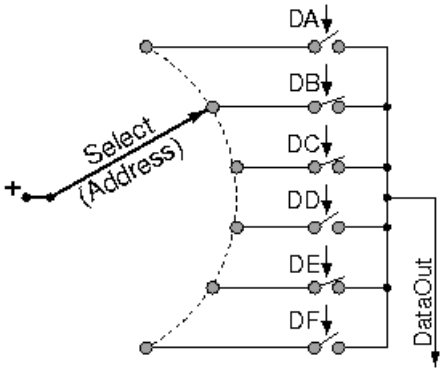
## 1.6 Η Πολύπλεξη σαν το Λογικό Ή πολλαπλών ΚΑΙ




Το κύκλωμα πολυπλέκτη με διακόπτες που είδαμε στο πρώτο σχήμα στην προηγούμενη

παράγραφο, §1.5, ήταν σχεδιασμένο με πολύ αφηρημένο τρόπο από ηλεκτρική άποψη --ίσα-ίσα γιά να δώσει τη γενική ιδέα της πολύπλεξης, αλλά χωρίς να δείχνει τις ηλεκτρικές λεπτομέρειες του τι είναι τα "δεδομένα εισόδου" και τι η "έξοδος". Το σχήμα εδώ δεξιά δείχνει τον ίδιο πολυπλέκτη 4-σε-1, αλλά προσδιορίζοντας αυτή τη φορά το από πού έρχονται οι είσοδοι δεδομένων και πού πηγαίνει η έξοδος· έτσι, εδώ έχουμε ένα κανονικό ηλεκτρικό κύκλωμα, που ξεκινά από το θετικό πόλο της μπαταρίας, αριστερά, και καταλήγει πίσω στη μπαταρία (αρνητικό πόλο) δεξιά-κάτω. *DA* είναι ο διακόπτης που μας τροφοδοτεί τα δεδομένα εισόδου *DataA* που έδειχνε το προηγούμενο σχήμα: όταν ο DA είναι ελεύθερος, όπως στο εδώ σχήμα, τότε η "πληροφορία" DataA λέει "ας μείνει η λάμπα σβηστή". Από την άλλη, ο διακόπτης *DB* που είναι πατημένος μας τροφοδοτεί το "δεδομένο εισόδου" *DataB* που λέει "θέλω η λάμπα να ανάψει". Βέβαια, στην προκειμένη περίπτωση, ούτε ο διακόπτης DA καταφέρνει να κρατήσει τη λάμπα σβηστή, ούτε ο διακόπτης DB είναι αυτός που ανάβει τη λάμπα --απλούστατα διότι τα "σήματα επιλογής" SA και SB **δεν** επιλέγουν τις εισόδους DataA ή DataB γιά να περάσουν στην έξοδο (λάμπα) DataOut --ούτε το SD επιλέγει την DataD-- παρά μόνο το SC επιλέγει την DataC γιά να περάσει.

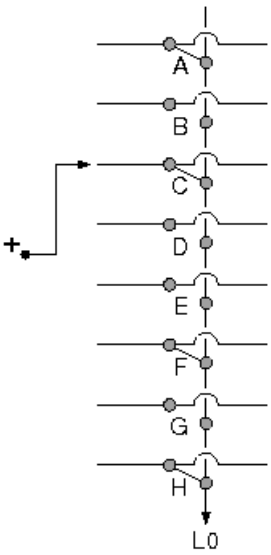
Όπως βλέπουμε, τελικά ο πολυπλέκτης είναι ένα μεγάλο λογικό **Ή** (δηλαδή εν παραλλήλω σύνδεση) από όρους που ο καθένας τους είναι το λογικό **ΚΑΙ** δύο πραγμάτων (δηλαδή δύο διακόπτες εν σειρά): ενός δεδομένου εισόδου, και του αντίστοιχου σήματος επιλογής. *Μία μόνο* από τις εισόδους επιλέγεται κάθε φορά, δηλαδή ένα μόνο από τα σήματα επιλογής είναι αναμμένο (διακόπτης πατημένος). Συνεπώς, ένας μόνο από τους παράλληλα-συνδεδεμένους κλάδους μπορεί να άγει ρεύμα --εκείνος που το σήμα επιλογής του είναι αναμμένο. Αυτός ο επιλεγμένος κλάδος, τελικά, θα άγει ή δεν θα άγει ρεύμα ανάλογα με το τι κάνει το σήμα δεδομένων του: εάν αυτός ο επιλεγμένος διακόπτης δεδομένων πεί να ανάψει η λάμπα, αυτή θα ανάψει, επειδή είναι αναμμένο και το σήμα επιλογής και το ρεύμα μπορεί να περάσει προς τη λάμπα (έξοδο)· εάν ο επιλεγμένος διακόπτης δεδομένων είναι σβηστός, τότε και η λάμπα θα μείνει σβηστή, αφού ο μεν επιλεγμένος κλάδος δεν άγει ρεύμα λόγω των δεδομένων του, οι δε υπόλοιποι κλάδοι δεν άγουν λόγω των σβηστών διακοπτών επιλογής τους. Τελικά λοιπόν, η λάμπα θα κάνει ό,τι κάνει και ο διακόπτης δεδομένων (data) του μοναδικού επιλεγμένου κλάδου --του μοναδικού κλάδου που έχει αναμμένο διακόπτη "Select".




Τα προηγούμενα σχήματα σχεδιάστηκαν θεωρώντας ότι τα δεδομένα (data) "έρχονται από μακρυά" (αριστερά), και η επιλογή γίνεται "τοπικά", δηλαδή δεξιά, κοντά στην έξοδο, όπως π.χ. στα τηλεπικοινωνιακά συστήματα. Η άλλη περίπτωση είναι αυτό που συμβαίνει στη **Μνήμη**: τα δεδομένα είναι εδώ, κοντά μας, αποθηκευμένα μέσα στη μνήμη, και κάποιος έρχεται απ' έξω και ζητάει να διαβάσει *ένα από αυτά* τα δεδομένα (ή μία μικρή ομάδα από αυτά, όπως θα πούμε παρακάτω). Στην περίπτωση αυτή, ταιριάζει να αντιστρέψουμε τη θέση των διακοπτών δεδομένων και επιλογής, όπως φαίνεται στο δεύτερο σχήμα. Φυσικά, η λογική λειτουργία **ΚΑΙ** είναι η ίδια, ανεξαρτήτως θέσης των δύο εν σειρά διακοπτών, αφού γιά να περάσει ρεύμα πρέπει να είναι και οι δύο αναμμένοι, με όποια σειρά και αν τους βάλουμε. Η άλλη αλλαγή που κάναμε σε αυτό το δεύτερο σχήμα είναι να αντικαταστήσουμε, συμβολικά, τους πολλαπλούς διακόπτες επιλογής με κάτι που θυμίζει έναν μόνο μεγάλο διακόπτη με πολλαπλές επαφές, αριστερά: Αφού *ένας και μόνον ένας* από τους διακόπτες επιλογής πρέπει να είναι αναμμένος, σημαίνει ότι η τροφοδοσία ρεύματος από τον θετικό πόλο της πηγής κατευθύνεται σε έναν και μόνον έναν από τους κλάδους. Αυτό δείχνει και

το σχήμα συμβολικά, με το μεγάλο βέλος που κινείται κυκλικά, τροφοδοτώντας τον εκάστοτε επιλεγμένο κλάδο του πολυπλέκτη. Στο σχήμα έχουμε 6 δεδομένα, DA, DB, DC, DD, DE, και DF, με τους αντίστοιχους 6 κλάδους και 6 επιλογές γιά τον μεγάλο κυκλικό διακόπτη αριστερά. Στις μνήμες λέμε ότι η *Διεύθυνση (Address)* είναι αυτή που επιλέγει (select) το ποιό από τα (εδώ 6) δεδομένα θέλουμε εκάστοτε να οδηγηθεί στην έξοδο --δηλαδή να το διαβάσουμε. Στο παράδειγμα του σχήματος, όποτε "διαβάζουμε" (επιλέγουμε) τις πληροφορίες (data) DA, DD, ή DF θα τις βρίσκουμε σβηστές (σβηστοί διακόπτες), δηλαδή η λάμπα που θα συνδέεται στην έξοδο DataOut θα μένει σβηστή· αντίθετα, όποτε διαβάζουμε τις θέσεις DB, DC, ή DE θα παίρνουμε παίρνουμε την απάντηση "αναμμένη" (αναμμένοι διακόπτες), δηλαδή η λάμπα στο DataOut θα ανάβει.

## 1.7 Μνήμη Χειροκίνητης Εγγραφής και Ηλεκτρονικής Ανάγνωσης

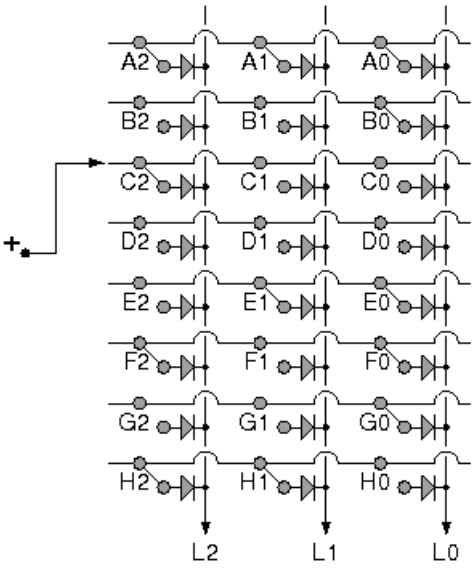



Θα δούμε τώρα πώς φτιάχνονται οι μνήμες, βασικά με μερικούς μεγάλους πολυπλέκτες. Οι πολυπλέκτες είναι γιά την *ανάγνωση* από τις μνήμες. Η εγγραφή στη μνήμη απαιτεί άλλα κυκλώματα, πιό πολύπλοκα, που θα τα δούμε λίγο αργότερα. Η μνήμη που θα δούμε εδώ, επειδή ηλεκτρονικά μόνο να την διαβάσει κανείς μπορεί, λέγεται *ROM* (Read-Only Memory). Παρ' ότι δεν μπορούμε να γράψουμε ηλεκτρονικά στη μνήμη που θα δούμε, εν τούτοις θα μπορούμε να γράψουμε *χειροκίνητα* σε αυτήν --δηλαδή να αλλάξουμε με το χέρι τις πληροφορίες που είναι αποθηκευμένες σε αυτήν.

Το σχήμα δεξιά είναι παρόμοιο με το προηγούμενο, δηλαδή είναι ένας πολυπλέκτης, μεγέθους, εδώ, 8-σε-1. Η κύρια διαφορά από το προηγούμενο --πέρα από το σχήμα των συρμάτων, που όμως δεν επηρεάζει τη λειτουργία τους-- είναι ότι στη θέση των διακοπτών δεδομένων βάλαμε *βραχυκυκλωτήρες* (jumpers): Σε κάθε διασταύρωση οριζόντιου σύρματος με το κατακόρυφο, υπάρχουν δύο επαφές --από μία σε κάθε σύρμα. Σε μερικά από αυτά τα ζευγάρια (και συγκεκριμένα στα A, C, F, και H) τοποθετήσαμε (με το χέρι!) "βραχυκυκλωτήρες", δηλαδή σύρματα (σχεδιασμένα διαγώνια) που ενώνουν ηλεκτρικά μεταξύ τους τις δύο επαφές, άρα το οριζόντιο εκείνο σύρμα με το κατακόρυφο. Στις άλλες διασταυρώσεις (στο παράδειγμά μας, στις B, D, E, και G) δεν τοποθετήσαμε βραχυκυκλωτήρες, άρα τα αντίστοιχα οριζόντια σύρματα *δεν* συνδέονται με το κατακόρυφο. Η διαφορά με το προηγούμενο κύκλωμα είναι ότι ο βραχυκυκλωτήρας *παραμένει* στη θέση του από μόνος του, χωρίς εξωτερική επέμβαση ή πίεση, σε αντίθεση με το διακόπτη που γιά να μείνει αναμμένος πρέπει κάποιος συνεχώς να τον πατάει. Με άλλα λόγια, ο βραχυκυκλωτήρας **έχει μνήμη**, αφού "θυμάται" πού τον βάλαμε, και παραμένει εκεί, σε αντίθεση με τον απλό διακόπτη (bouton) που δεν έχει μνήμη, αφού μόλις τον αφήσεις μόνο του αυτός πάντα σβήνει, ό,τι και να του' χες κάνει πρίν.

Έτσι, το σχήμα εδώ είναι μιά *Μνήμη με 8 αποθηκευμένες πληροφορίες*: στις θέσεις A, C, F, και H η αποθηκευμένη πληροφορία λέει "άναψε λάμπα L0", ενώ στις θέσεις B, D, E, και G η αποθηκευμένη πληροφορία λέει "μείνε σβηστή λάμπα L0". Εάν θέλουμε να αλλάξουμε την πληροφορία που είναι αποθηκευμένη σε κάποια θέση, σε αυτήν εδώ τη μνήμη μόνο χειροκίνητα μπορούμε να το κάνουμε: να βγάλουμε από εκεί τον βραχυκυκλωτήρα που υπήρχε, ή να βάλουμε εκεί έναν αν δεν υπήρχε. Στο σχήμα, ο χρήστης επέλεξε, με το σύρμα αριστερά από την θετική τροφοδοσία, να

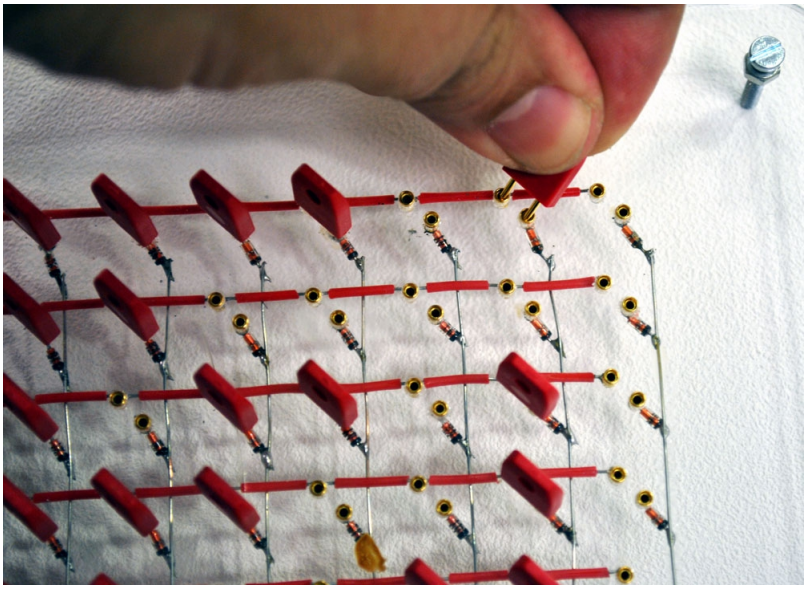
"διαβάσει" τη θέση C της μνήμης, οδηγώντας την αποθηκευμένη πληροφορία της στην έξοδο (λάμπα) L0: επειδή στη θέση C υπάρχει βραχυκυκλωτήρας, η "πληροφορία" αυτή κάνει την λάμπα L0 να ανάψει. Εάν, αντίθετα, ο χρήστης μετακινήσει το σύρμα επιλογής (αριστερά) π.χ. στη θέση G, επειδή στη θέση εκείνη δεν υπάρχει βραχυκυκλωτήρας και τα σύρματα παραμένουν ασύνδετα, η πληροφορία αυτή θα αντικατοπτριστεί στην L0 με σβηστή λάμπα.




Μία μνήμη με ένα μόνο σύρμα εξόδου δεν είναι πολύ χρήσιμη --συνήθως χρειαζόμαστε μεγαλύτερες (φαρδύτερες) μνήμες, που όταν διαβάζουμε από μία θέση τους να διαβάζουμε περισσότερες από μία πληροφορίες. Στο σχήμα εδώ δεξιά "τριπλασιάσαμε" (οριζόντια) την προηγούμενη μνήμη: έχουμε εδώ μία **Μνήμη 8x3**, δηλαδή με 8 ομάδες (γραμμές) πληροφοριών --τις λέμε **8 λέξεις**-- και όπου η κάθε μιά από αυτές τις 8 λέξεις περιέχει 3 "πληροφορίες" μέσα της, οι οποίες ελέγχουν 3 λαμπάκια, L2, L1, και L0 --τα λέμε **3 bits ανά λέξη**. Ο περίφημος όρος **bit** προέρχεται από τις λέξεις **bi**nary digi**t**, και Ελληνικά αποδίδεται πολλές φορές σαν **δυφίο**, από τις λέξεις **δυ**αδικό ψηφ**ίο**. Ένα bit είναι η ποσότητα της πληροφορίας που μπορεί μόλις να επιλέξει μία από **δύο** επιλογές --ας πούμε, εδώ, σβηστή ή αναμμένη λαμπίτσα. Γιά τις μνήμες που είναι φαρδύτερες από ένα bit, όπως αυτή εδώ, απαιτείται η χρήση διόδων στα σημεία διασταύρωσης, όπως δείχνει το σχήμα, προκειμένου αυτές να επιβάλουν το ρεύμα να ρέει πάντα από την είσοδο επιλογής (αριστερά) προς τις εξόδους δεδομένων (κάτω), και ποτέ ανάποδα από ένα κατακόρυφο σύρμα πολυπλέκτη πίσω σε οριζόντια σύρματα άλλα από το μοναδικό επιλεγμένο οριζόντιο σύρμα.

Οι μνήμες είναι βασικά στοιχεία όλων των ψηφιακών συστημάτων και των υπολογιστών. Η βασική οργάνωσή τους, που φαίνεται εδώ, είναι πρωταρχικής σημασίας: *λέξεις*, που αποτελούν οριζόντιες ομάδες από bits, γραμμένες η μία κάτω από την άλλη, όπου κάθε λέξη αποτελείται από *bits*, που γράφονται μέσα στη λέξη, το ένα δίπλα στο άλλο. Όποτε διαβάζουμε κάτι από τη μνήμη, διαβάζουμε πάντα όλα τα bits μίας **ολόκληρης λέξης**. Επομένως, η μνήμη έχει τόσα σύρματα εξόδου δεδομένων (κάτω στο σχήμα) όσα bits έχει η κάθε λέξη της· στο εδώ παράδειγμα έχουμε 3 εξόδους δεδομένων, L2, L1, και L0, και κάθε λέξη έχει 3 bits: η λέξη A έχει τα bits A2, A1, A0, κ.ο.κ. Στο σχήμα, η είσοδος επιλογής (το σύρμα αριστερά από τον θετικό πόλο της μπαταρίας) αυτή τη στιγμή επιλέγει να διαβάσουμε τη λέξη C, η οποία αποτελείται από 3 bits: C2, C1, C0. Στο C2 έχουμε βάλει βραχυκυκλωτήρα, ενώ στα C1 και C0 όχι. Επομένως, η πληροφορία που είναι αποθηκευμένη σε αυτή τη λέξη είναι: "αναμμένη, σβηστή, σβηστή", και όντως, το σύρμα L2 θα πάρει ρεύμα μέσω του βραχυκυκλωτήρα C2, ενώ τα σύρματα L1 και L0 θα παραμείνουν χωρίς ρεύμα (σβηστά) επειδή είναι ασύνδετα με το οριζόντια σύρμα C --που είναι το μοναδικό επιλεγμένο (τροφοδοτημένο από την μπαταρία) οριζόντιο σύρμα. Εάν το σύρμα επιλογής, αριστερά, μετακινηθεί π.χ. στη θέση F, τότε οι έξοδοι δεδομένων θα γίνουν "αναμμένη, σβηστή, αναμμένη", επειδή οι διασταυρώσεις F2 και F0 έχουν βραχυκυκλωτήρες, ενώ η F1 δεν έχει. Έτσι, τότε, θα έχουμε διαβάσει τα 3 bits της λέξης F, αστα οποία είχαμε αποθηκεύσει (στο παρελθόν, χειροκίνητα, μέσω των βραχυκυκλωτήρων) τις πληροφορίες (3 bits) "αναμμένο, σβηστό, αναμμένο".

## <u>Πείραμα 1.8:</u> Μνήμη 8x8 με Jumpers και οδήγηση 7-Segment-Display

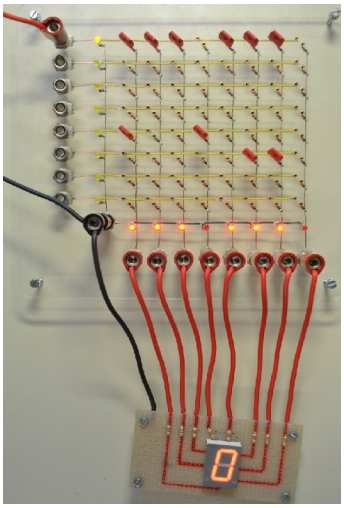
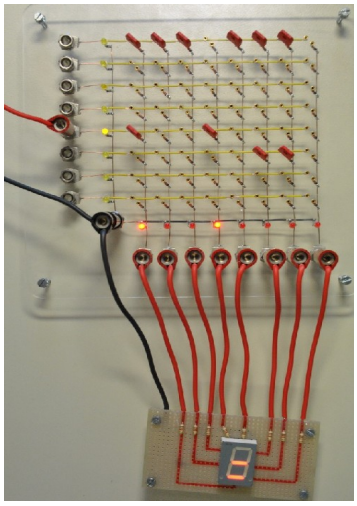
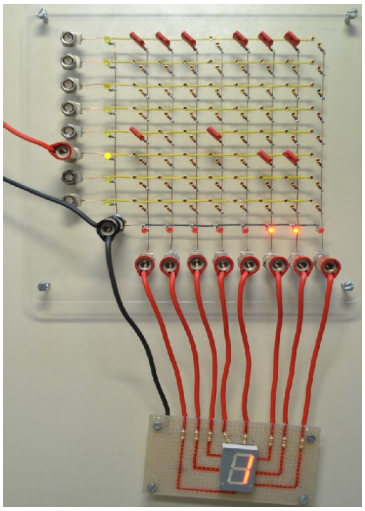


Στο εργαστήριο θα βρείτε πλακέτες (διαφανείς) με το παραπάνω κύκλωμα μνήμης ROM, μεγέθους 8x8, δηλαδή 8 λέξεις (8 γραμμές), η κάθε μία των 8 bits (8 στήλες). Οι βραχυκυκλωτήρες (jumpers) είναι μικρά κόκκινα "πριζάκια", με δύο ποδαράκια καθένας· εσωτερικά, τα δύο ποδαράκια είναι ενωμένα (βραχυκυκλωμένα) μεταξύ τους. Στη φωτογραφία δεξιά βλέπετε ένα κοντινό πλάνο της πάνω δεξιάς γωνίας της μνήμης, και φαίνεται πώς μπαίνουν και βγαίνουν (χειροκίνητα!) τα πριζάκια/jumpers. Παρακαλούμε προσέχετε να **μην χάνετε τους βραχυκυκλωτήρες** αυτούς!

Θα βρείτε την πλακέτα συνδεδεμένη σε έναν ενδείκτη 7 τμημάτων (7-segment display), όπως δείχνουν οι παρακάτω φωτογραφίες. Ο ενδείκτης 7 τμημάτων έχει 7 γραμμικές LED's που σχηματίζουν το γνώριμό μας σχήμα ενός δεκαδικού ψηφίου (και έχει και μιά δεκαδική υποδιαστολή --συνολικά 8 LED's που τροφοδοτούνται από 8 σύρματα). Η πλακέτα μνήμης παίρνει τροφοδοσία 12 Volt όπως φαίνεται στις φωτογραφίες: γείωση (αρνητικό - μαύρο) στην μαύρη υποδοχή, και θετικό (κόκκινο) σε **μία και μόνο μία** γραμμή-λέξη, την "επιλεγμένη" λέξη την οποία θέλετε να διαβάσετε. Επίσης, η πλακέτα μνήμης πρέπει να δίνει γείωση στην πλακέτα 7-segment, μέσω του μαύρου καλωδίου. Η θετική τροφοδοσία φτάνει στις LED του 7-segment display μέσα από τις εξόδους δεδομένων της μνήμης, με 8 κόκκινα καλώδια.

**ΠΡΟΣΟΧΗ:** *<u>μην βραχυκυκλώσετε</u>* (ενώσετε κατευθείαν) ποτέ οιαδήποτε έξοδο δεδομένων της μνήμης (οιαδήποτε από τις κάτω δεξιά οκτώ υποδοχές) με την "γείωση" --την μαύρη υποδοχή: εάν το κάνετε, θα κάψετε την δίοδο που υπάρχει στο bit εκείνο της μνήμης στη γραμμή που έχετε εκείνη τη στιγμή ενεργοποιημένη!

Στις παρακάτω φωτογραφίες έχουμε αποθηκεύσει στη μνήμη (μέσω της θέσης των jumpers) τις πληροφορίες (bits) που χρειάζονται ώστε να εμφανίζονται στον ενδείκτη: το ψηφίο "0" όταν επιλέγεται (διαβάζεται) η πρώτη (επάνω) λέξη (γραμμή) της μνήμης· το σύμβολο "=" όταν διαβάζουμε την πέμπτη γραμμή της μνήμης· και το ψηφίο "1" όταν διαβάζουμε την έκτη γραμμή της μνήμης. Η επιλογή της λέξης που θέλουμε να διαβάσουμε γίνεται μετακινώντας το κόκκινο καλώδιο τροφοδοσίας.

Αποθηκεύστε (γράψτε χειροκίνητα, βγάζοντας/βάζοντας πριζάκια) στη μνήμη σας τις κατάλληλες πληροφορίες γιά να εμφανίζονται στον ενδείκτη σας διάφορα σύμβολα, όπως π.χ.: (α) 0, 1, 2, 3, 4, 5, 6, 7· ή (β) A, b, c, d, E, F, H, J· ή (γ) 0, 2, 4, 6, 8, =, Ξ, ||· κ.ο.κ. Αφού τα γράψετε, μετά διαβάστε τα περιεχόμενα της μνήμης σας, πολλαπλές φορές, με τη σειρά ή ανακατωμένα, και διαπιστώστε ότι η μνήμη σας όντως θυμάται ότι της γράψατε τελευταίο!