

Εργαστήριο 4: Χάρτες Karnaugh, Άλγεβρα Boole, Μνήμη RAM

10 - 15 Νοεμβρίου 2003

[Τα τμήματα της Τρίτης 11/11 (Αγ. Μηνά) μεταφέρονται στο Σάββατο 15/11, ίδιες ώρες, εκτός 7-9 που πάνε 11-1]

[Αναγγελία Διαγωνισμού Προόδου: Σάββατο 29 Νοεμβρίου, 10-12 π.μ.]

[Βιβλίο: *προαιρετικά* μπορείτε να διαβάσετε τις παραγράφους **3.1** έως και το μέσον της **3.3** (σελίδες 93-104), την παράγραφο **3.8** (σελ. 126-129), τις παραγράφους **2.3, 2.4**, και λίγο 2.2 (σελίδες 52-64), και παραγράφους **2.7** και **2.8** (σελ. 75-88)].

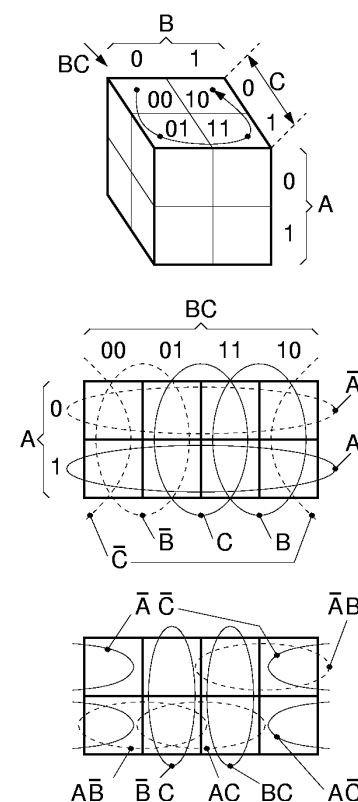
4.1 Διαγράμματα Venn και Χάρτες Karnaugh Τριών και Τεσσάρων Μεταβλητών:

Στην παράγραφο 3.2 (εργαστήριο 3) είδαμε την ιδέα των διαγραμμάτων Venn (αλληλοτεμνόμενες περιοχές όπου κάθε δυαδική μεταβλητή συμβολίζει αν είμαστε μέσα ή έξω από μία από αυτές τις περιοχές), καθώς και τους χάρτες Karnaugh δύο μεταβλητών. Η επιτυχία τους να μας βοηθούν να απλοποιούμε λογικές συναρτήσεις οφείλεται στη διάταξη των τιμών της συνάρτησης στις δύο διαστάσεις με τρόπο τέτοιο που η κάθε μεταβλητή να αντιστοιχεί στη μία διάσταση. Για να πετύχουμε το ίδιο αποτέλεσμα με τρεις μεταβλητές εισόδου, φανταζόμαστε ότι διατάσσουμε τα περιεχόμενα του πίνακα αληθείας στις 3 διαστάσεις, σ' ένα σχήμα κύβου, όπως στο διπλανό σχήμα. Στη συνέχεια, επειδή είναι άβολο να σχεδιάζουμε τριδιάστατα σχήματα στο χαρτί, φανταζόμαστε ότι **ξεδιπλώνουμε** τον κύβο, κόβοντάς τον πίσω στη μέση, και φέρνοντας τις δύο πίσω στήλες στην αριστερή και στη δεξιά άκρη αντίστοιχα, όπως φαίνεται στο σχήμα, δεξιά.

Η συνέπεια είναι ότι οι στήλες του πίνακα αντιστοιχούν στις δύο μεταβλητές εισόδου B και C με τη σειρά που φαίνεται στο σχήμα: 00, 01, 11, 10 --η σειρά αυτή (που λέγεται και "κώδικας Gray") διαφέρει από τη συνηθισμένη σειρά αρίθμησης (μέτρησης) στο δυαδικό (00, 01, 10, 11). Με τη σειρά αυτή πετυχαίνουμε η περιοχή αληθείας της μεταβλητής B να αποτελείται από 4 **γειτονικά** τετράγωνα --τα 4 δεξιά τετράγωνα-- και ταυτόχρονα η περιοχή αληθείας της μεταβλητής C να αποτελείται επίσης από 4 γειτονικά τετράγωνα --τα 4 μεσαία τετράγωνα. Η περιοχή αληθείας του C' (δηλ. του "NOT C") είναι επίσης 4 "γειτονικά" τετράγωνα --δύο στη άκρη αριστερά και δύο στην άκρη δεξιά-- αλλά για να καταλάβουμε ότι αυτά είναι γειτονικά πρέπει να φανταστούμε τον πίνακα σαν ένα ξετυλιγμένο βαρέλι: τα τετράγωνα αυτά ήταν γειτονικά πριν το ξετύλιγμα. Γενικά, στο χάρτη 3 μεταβλητών, οιαδήποτε 4 "γειτονικά" τετράγωνα --δηλαδή 4 τετράγωνα σε σχήμα 2x2 ή 4x1-- αντιστοιχούν σε μία μεταβλητή εισόδου ή στην άρνησή της. Οι δύο οριζόντιες τετράδες αντιστοιχούν στο A και στο A' (NOT A).

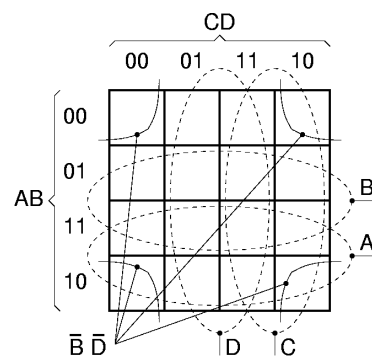
Περιοχές δύο γειτονικών τετραγώνων αντιστοιχούν στο λογικό ΚΑΙ δύο εκ των τριών μεταβλητών εισόδου (ή των αρνήσεών τους). Μερικά τέτοια ζευγάρια φαίνονται στο κάτω μέρος του σχήματος. Παρατηρήστε ότι τα ζευγάρια που περιλαμβάνουν το C' μοιάζουν "κομμένα" --ένα τετράγωνο στην άκρη αριστερά και ένα στην άκρη δεξιά-- όμως αποτελούνται και αυτά από τετράγωνα που ήταν γειτονικά πριν ξετυλίξουμε το χάρτη από τη μορφή βαρελιού που λέγαμε πιο πάνω. Τέλος, φυσικά, όπως και στους χάρτες 2 μεταβλητών, μεμονωμένα τετράγωνα αντιστοιχούν στο λογικό ΚΑΙ όλων των μεταβλητών εισόδου ή των αρνήσεών τους, και περιοχές που προκύπτουν από την ένωση ομάδων τετραγώνων αντιστοιχούν στο λογικό Ή των σχετικών όρων. Όταν καλύπτουμε περιοχές με τέτοιες ενώσεις, επιδιώκουμε η κάθε ομάδα γειτονικών τετραγώνων να είναι όσο μεγαλύτερη γίνεται (πάντα βέβαια μεγέθους δύναμης του 2, διατεταγμένη σε σχήμα ορθογωνίου)· επικαλύψεις περιοχών δεν μας ενοχλούν --αντίθετα βοηθούν στη μεγιστοποίηση της έκτασης της κάθε μεμονωμένης περιοχής.

Ο χάρτης Karnaugh τεσσάρων μεταβλητών σχεδιάζεται όπως φαίνεται στο επόμενο σχήμα. Πρέπει να φανταστούμε ότι αυτός ο πίνακας 4x4 προέρχεται από **διπλό ξετύλιγμα** μιάς παράξενης σφαιροειδούς επιφάνειας και οριζόντια και κατακόρυφα. Η αριστερή και η δεξιά στήλη ήταν γειτονικές πριν το ξετύλιγμα, σαν να προέρχονται από ένα όρθιο βαρέλι, και αντιστοιχούν στη μεταβλητή D' (NOT D). Ταυτόχρονα, η επάνω και η κάτω γραμμή ήταν κι αυτές γειτονικές, πριν το ξετύλιγμα από ένα πλαγιαστό βαρέλι, και αντιστοιχούν στη μεταβλητή B' (NOT B). Τετράδες γειτονικών τετραγώνων αντιστοιχούν στο λογικό ΚΑΙ δύο εκ των τεσσάρων μεταβλητών εισόδου, ή των αρνήσεών τους. Μία τέτοια τετράδα --η πιο πολύ κομμένη απ' όλες--



φαίνεται στο σχήμα στις 4 γωνίες: πρόκειται για την περιοχή B'D', και τα τετράγωνα της ήταν όλα γειτονικά πριν το διπλό ξετύλιγμα. Άλλες κομμένες τετράδες έχουν 2 τετράγωνα αριστερά και 2 δεξιά, ή 2 τετράγωνα επάνω και 2 κάτω. Ζευγάρια γειτονικών τετραγώνων αντιστοιχούν στο λογικό ΚΑΙ τριών εκ των τεσσάρων μεταβλητών εισόδου, ή των αρνήσεών τους. Μεμονωμένα τετράγωνα αντιστοιχούν στο λογικό ΚΑΙ όλων των μεταβλητών εισόδου (ή των αρνήσεών τους).

Μπορούν να οριστούν και χάρτες Karnaugh πέντε ή περισσότερων μεταβλητών, αλλά δεν είναι πρακτικοί. Εξ' άλλου, ας μην ξεχνάμε ότι η απλοποίηση λογικών συναρτήσεων "με το μάτι" και "με το χέρι" ανήκει στο παρελθόν: σήμερα υπάρχουν αποδοτικοί αλγόριθμοι και αντίστοιχα προγράμματα που κάνουν αυτές τις απλοποιήσεις και πολλές άλλες αυτόματα. Το πιο γνωστό τέτοιο πακέτο αυτόματης σύνθεσης υλικού, σήμερα, είναι το "Synopsys".



Άσκηση 4.2: Οι Αριθμοί 0-7 στην Οθόνη 7 Τμημάτων

ABC: 000 001 010 011 100 101 110 111
Οθονη:

Πρίν φτάσετε στο εργαστήριο κάντε αυτή την άσκηση με μολύβι και χαρτί: η άσκηση αυτή δεν έχει τίποτα να φτιάξετε στο εργαστήριο, διότι τα κυκλώματα που προκύπτουν είναι πολύ μεγάλα για να υλοποιηθούν με ηλεκτρονόμεους στην πλακέτα μας --απλώς θα παραδώσετε τη λύση σας μέσα στην αναφορά του εργαστηρίου σας. Ζητείται να βρεθούν οι 7 λογικές συναρτήσεις οδήγησης των 7 τμημάτων του γνωστού μας ενδείκτη, προκειμένου αυτός να εμφανίζει τις 8 ενδείξεις που φαίνονται στο σχήμα, ανάλογα με τον εκάστοτε συνδυασμό τιμών των τριών bits εισόδου A, B, και C. Φτιάξτε τους 7 πίνακες αληθείας σε μορφή 7 χαρτών Karnaugh τριών μεταβλητών, βρείτε τις ομάδες γειτονικών τετραγώνων που χρειάζονται για να καλύψουν τις περιοχές ανάματος της κάθε LED, γράψτε τις αντίστοιχες λογικές συναρτήσεις για τις 7 εξόδους, και σχεδιάστε το αντίστοιχο κύκλωμα με πύλες.

4.3 Συνθήκες Αδιαφορίας (Don't Care Conditions):

Σε όλες τις παραπάνω περιπτώσεις, οι πίνακες αληθείας των επιθυμητών εξόδων ήταν πλήρως προδιαγεγραμμένοι, δηλαδή μας ενδιέφερε η κάθε έξοδος να έχει μία συγκεκριμένη, προκαθορισμένη τιμή για τον κάθε δυνατό συνδυασμό τιμών των εισόδων. Υπάρχουν όμως και περιπτώσεις "μερικώς προδιαγεγραμμένων" συστημάτων. Για παράδειγμα, στην παρακάτω άσκηση, μας ζητείται να οδηγήσουμε τη γνωστή μας οθόνη 7 τμημάτων σε τρόπον ώστε να εμφανίζονται σε αυτήν τα δέκα ψηφία από το 0 ως το 9. Για να πετύχουμε δέκα διαφορετικές εξόδους δεν αρκούν προφανώς 3 bits εισόδου --χρειάζονται τέσσερα. Όμως, τα 4 bits έχουν 16 δυνατούς συνδυασμούς: διαλέγουμε δέκα από αυτούς για να γεννάνε στην οθόνη τα δέκα επιθυμητά ψηφία, ενώ **δεν μας ενδιαφέρει** τι θα κάνει η οθόνη όταν στις εισόδους εμφανίζεται ένας από τους υπόλοιπους έξι συνδυασμούς --π.χ. το κύκλωμα που μας τροφοδοτεί με τα 4 bits εισόδου ποτέ δεν θα μας δίνει έναν από τους υπόλοιπους 6 συνδυασμούς, ή αν μας δώσει, δεν μας ενδιαφέρει τι θα δείξει η οθόνη σε αυτή την περίπτωση.

Όταν ένας πίνακας αληθείας δεν προκαθορίζει την τιμή εξόδου για ορισμένο συνδυασμό τιμών εισόδου, λέμε ότι εκεί έχουμε μία **συνθήκη αδιαφορίας** (don't care condition), και συχνά βάζουμε στον στη θέση εκείνη του πίνακα και του χάρτη Karnaugh σα σύμβολο ένα "x". Στο χάρτη Karnaugh, όταν αναζητούμε την ελάχιστη δυνατή ένωση των μέγιστων δυνατών περιοχών γειτονικών τετραγώνων προκειμένου να καλύψουμε τους άσους του χάρτη, θεωρούμε ότι το κάθε "x" είναι **ό,τι μας βολεύει**. Αν βολεύει να το θεωρήσουμε σαν άσσο, προκειμένου να πετύχουμε μεγαλύτερη περιοχή γειτονικών τετραγώνων, το θεωρούμε σαν άσσο. Αν βολεύει να το θεωρήσουμε σαν μηδενικό, προκειμένου να μην χρειαστούμε μία επιπλέον περιοχή για να το καλύψουμε, το θεωρούμε σαν μηδενικό. Η συμπεριφορά του τελικού κυκλώματος για κάθε αδιάφορη τιμή εισόδων θα καθοριστεί προφανώς από το τι μας βόλεψε και θεωρήσαμε το αντίστοιχο "x" στο χάρτη.

Άσκηση 4.4: Οι Αριθμοί 0-9 στην Οθόνη 7 Τμημάτων

ABCD: 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001
Οθονη:

Πρίν φτάσετε στο εργαστήριο κάντε αυτή την άσκηση με μολύβι και χαρτί: η άσκηση αυτή δεν έχει τίποτα να φτιάξετε στο εργαστήριο, διότι τα κυκλώματα που προκύπτουν είναι πολύ μεγάλα για να υλοποιηθούν με ηλεκτρονόμεους στην πλακέτα μας --απλώς θα παραδώσετε τη λύση σας μέσα στην αναφορά του εργαστηρίου σας. Ζητείται να βρεθούν οι 7 λογικές συναρτήσεις οδήγησης των 7 τμημάτων του γνωστού μας ενδείκτη, προκειμένου αυτός να εμφανίζει τις 10 ενδείξεις που φαίνονται στο σχήμα, ανάλογα με τον εκάστοτε συνδυασμό τιμών των 4 bits εισόδου A, B, C, και D. Φτιάξτε τους 7 πίνακες αληθείας σε μορφή 7 χαρτών Karnaugh τεσσάρων μεταβλητών που περιέχουν και από 6 όρους αδιαφορίας καθένας: βρείτε και σημειώστε τις ομάδες γειτονικών τετραγώνων που βολεύουν και χρειάζονται για να καλύψουν τις περιοχές ανάματος της κάθε LED, και γράψτε τις αντίστοιχες λογικές συναρτήσεις για τις 7 εξόδους. Δεν χρειάζεται να σχεδιάσετε το κύκλωμα με πύλες, εδώ.

4.5 Άλγεβρα Boole:

Οι λογικές πράξεις ΚΑΙ, Ή, ΟΧΙ πάνω σε δυαδικές ψηφιακές μεταβλητές ορίζουν μιάν Άλγεβρα, η οποία ονομάστηκε "**Άλγεβρα Boole**" (Boolean Algebra), προς τιμήν του Μαθηματικού George Boole ο οποίος δημοσίευσε τις πρώτες σχετικές

ιδέες το 1849, καθώς εργάζονταν πάνω στη μαθηματική διατύπωση των κανόνων της λογικής νόησης. Περίπου 90 χρόνια αργότερα, το 1938, ο Claude Shannon έδειξε ότι η Άλγεβρα Boole, όπως είχε εν τω μεταξύ εξελιχθεί, περιγράφει και τη λειτουργία των κυκλωμάτων διακοπών, όπως κάναμε κι εμείς στην αρχή του μαθήματός μας.

Η Άλγεβρα Boole μπορεί να δομηθεί ξεκινώντας από τον ("αξιοματικό") ορισμό των τριών πράξεων, ΚΑΙ, Ή, ΟΧΙ, βάσει του πίνακα αληθείας τους που είδαμε στο εργαστήριο 1, δηλαδή από τον ορισμό τους μέσω της εξαντλητικής απαρίθμησης του αποτελεσματός τους για τον κάθε δυνατό συνδυασμό εισόδων. Για σκοπούς συντομογραφίας, από δω και πέρα, θα συμβολίζουμε τις λογικές αυτές πράξεις με AB [ή και με τελεία στη μέση] (A και B), $A+B$ (A ή B), και A' [ή και με παύλα από πάνω] (όχι A), όπως είπαμε στο εργαστήριο 3. Τις μεταβλητές της Άλγεβρας Boole, δηλαδή τις δυαδικές ψηφιακές μεταβλητές, τις λέμε και "*Μεταβλητές Boole*" (Boolean variables). Όπως έχουμε πεί, οι δύο τιμές μιάς μεταβλητής Boole μπορεί να συμβολίζουν πολλά και διαφορετικά πράγματα, π.χ. αναμένο-σβηστό, ζεστό-κρύο, πάνω-κάτω, μπρός-πίσω, αριστερά-δεξιά, πατημένος-ελεύθερος (διακόπτης), ψηλή-χαμηλή (ηλεκτρική τάση), περνάει - δεν περνάει (ρεύμα), αληθές-ψευδές, ναι-όχι, 1-0, κλπ. Για σκοπούς συντομογραφίας, και πάλι, συνήθως θα χρησιμοποιούμε τα σύμβολα **1** (αληθές, αναμένο, κλπ), και **0** (ψευδές, σβηστό, κλπ).

Ξεκινώντας από τον ορισμό των τριών πράξεων Boole, μπορούμε να αποδείξουμε πολλά θεωρήματα της Άλγεβρας Boole, τα οποία συχνά αντιστοιχούν και σε συνηθισμένες διατυπώσεις της καθημερινής μας λογικής σκέψης. Η απόδειξη μπορεί να γίνει με εξαντλητική επαλήθευση όλων των περιπτώσεων (πίνακας αληθείας), ή με διαγράμματα Venn, ή με τη χρήση άλλων θεωρημάτων. Εκθέτουμε εδώ κάμποσα τέτοια θεωρήματα, κατά σειρά σημαντικότητας.

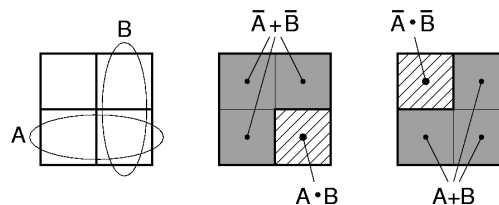
Δύο Αρνήσεις κάνουν Μία Κατάφαση:

Όπως ξέρουμε πολύ καλά και από την καθημερινή μας ζωή, και όπως αποδεικνύεται άμεσα και από τον πίνακα αληθείας, το ΟΧΙ (ΟΧΙ A) είναι το ίδιο με το A , δηλαδή: $(A')' = A$.

Θεώρημα DeMorgan (Άρνηση και Δυσμός):

Όπως παρατηρήσαμε ήδη κάμποσες φορές, από το εργαστήριο 1 και μετά, η άρνηση του ΚΑΙ ισοδυναμεί με το Ή των αρνήσεων, ενώ η άρνηση του Ή ισοδυναμεί με το ΚΑΙ των αρνήσεων. Η ιδιότητα αυτή, γνωστή σαν Αρχή του Δυσμού (Duality Principle) ή "Θεώρημα DeMorgan", διατυπώνεται επίσημα ως εξής:

$$\begin{aligned}(AB)' &= A' + B' \\ (A+B)' &= A'B'\end{aligned}$$



Για να το αποδείξουμε, αρκεί να κατασκευάσουμε τους πίνακες αληθείας των παραπάνω συναρτήσεων και να τους συγκρίνουμε. Αν τους κατασκευάσουμε σε μορφή διαγράμματος Venn / χάρτη Karnaugh, και σημειώσουμε τα τετράγωνα όπου η κάθε συνάρτηση γίνεται 1, θα προκύψει το σχήμα που φαίνεται. Όπως βλέπουμε, η συνάρτηση AB γίνεται 1 στο κάτω δεξιά τετράγωνο, άρα το συμπλήρωμά της (η άρνησή της), δηλ. η συνάρτηση $(AB)'$, θα γίνεται 1 στις υπόλοιπες περιπτώσεις (αριστερά και πάνω "Γ"). Αυτή η τελευταία περιοχή είναι ίδια με την ένωση (λογικό Ή) της περιοχής A' (A bar --δύο επάνω τετράγωνα) με την περιοχή B' (B bar --δύο αριστερά τετράγωνα), αποδεικνύοντας έτσι την πρώτη από τις παραπάνω σχέσεις. Αντίστοιχα μπορεί να αποδειχτεί και η δεύτερη σχέση, όπως φαίνεται στο δεξί μέρος του σχήματος.

Εναλλακτικά, η δεύτερη σχέση μπορεί να προκύψει από την πρώτη και από την ιδιότητα ότι δύο αρνήσεις κάνουν μία κατάφαση. Ξεκινάμε από το δεξί μέλος της ισότητας που θέλουμε να αποδείξουμε, και το μετασχηματίζουμε με δύο αρνήσεις: $A'B' = [(A'B')]'$. Μέσα στις αγκύλες υπάρχει η άρνηση ενός λογικού ΚΑΙ, άρα μπορούμε να εφαρμόσουμε σε αυτήν το πρώτο θεώρημα DeMorgan, και να την μετατρέψουμε στο λογικό Ή των αρνήσεων, οι οποίες στη συνέχεια μπορούν να απλοποιηθούν: $(A'B')' = (A')' + (B')' = A+B$. Βάσει αυτού, η προηγούμενη ισότητα μας δίνει: $A'B' = [(A'B')]'$ $= [A+B]' = (A+B)'$, πράγμα που είναι ακριβώς το δεύτερο θεώρημα DeMorgan. Ο τρόπος αυτός απόδειξης μας οδηγεί σε μία δεύτερη διατύπωση της αρχής του δυσμού: εάν σε μίαν ισότητα της άλγεβρας Boole αλλάξουμε όλα τα ΚΑΙ με Ή, και όλα τα Ή με ΚΑΙ, τότε προκύπτει μία άλλη, επίσης αληθής ισότητα, η "δυϊκή" της πρώτης (όπως θα δούμε πιο κάτω, αν η ισότητα περιέχει και άσους ή μηδενικά, τότε πρέπει και αυτά να τα αλλάξουμε από 0 σε 1 και από 1 σε 0).

Επιμεριστική Ιδιότητα (Distributive Property):

Κατ' ανάλογο τρόπο, μέσω του πίνακα αληθείας / διαγράμματος Venn, μπορεί κανείς να διαπιστώσει ότι:

$$\begin{aligned}A(B+C) &= AB + AC \\ A+(BC) &= (A+B)(A+C)\end{aligned}$$

Ειπωμένο με λόγια, αν ισχύει το A και επίσης ισχύει το B ή το C , τότε θα πρέπει να ισχύει το A και το B ή να ισχύει το A και το C . Αντίστοιχα, η δεύτερη σχέση λέει ότι αν ισχύει το A ή ισχύει το B και το C , τότε θα ισχύει το A ή το B , καθώς επίσης θα ισχύει το A ή το C . Παρατηρήστε ότι όταν χρησιμοποιούμε τα παραπάνω σύμβολα του ΚΑΙ που μοιάζει με το σύμβολο του πολλαπλασιασμού και του Ή που μοιάζει με το σύμβολο της πρόσθεσης, τότε η πρώτη από τις παραπάνω σχέσεις μοιάζει οικεία, αλλά η δεύτερη καθόλου (αφού, φυσικά, δεν μιλάμε για πρόσθεση και πολλαπλασιασμό).

Όπως και με τις δύο μορφές του θεωρήματος DeMorgan, οι δύο παραπάνω σχέσεις είναι *δυϊκές* μεταξύ τους: αν αντικαταστήσουμε τα ΚΑΙ με Ή και τα Ή με ΚΑΙ, τότε προκύπτει η μία από την άλλη. Ο λόγος είναι ότι η δεύτερη μπορεί να

προκύπτει από την πρώτη, εφαρμόζοντας τα θεωρήματα DeMorgan (δηλαδή την αρχή του δυϊσμού), και το ότι δύο αρνήσεις κάνουν μία κατάφαση. Ξεκινώντας με το αριστερό μέλος της δεύτερης σχέσης, το μετασχηματίζουμε ως εξής μέχρι να προκύψει το δεξί: $A+(BC) = \{ [A+(BC)]' \}'$ (δύο αρνήσεις) $= \{ A'(BC)' \}'$ (από DeMorgan) $= \{ A'(B'+C)' \}'$ (από DeMorgan) $= \{ A'B' + A'C' \}'$ (από την πρώτη επιμεριστική ιδιότητα) $= \{ (A+B)' + (A+C)' \}'$ (από DeMorgan) $= \{ [(A+B)(A+C)]' \}'$ (από DeMorgan) $= (A+B)(A+C)$ (δύο αρνήσεις).

Αντιμεταθετική και Προσεταιριστική Ιδιότητα (Commutative and Associative Property):

Όπως ξέρουμε, η σειρά των μεταβλητών δεν παίζει ρόλο στις πράξεις ΚΑΙ και Ή (αντιμεταθετική ιδιότητα), όπως επίσης στα πολλαπλά ΚΑΙ η σειρά των πράξεων δεν παίζει ρόλο, και το ίδιο και στα πολλαπλά Ή (προσεταιριστική ιδιότητα --γι' αυτό και συνήθως τα γράφουμε χωρίς παρενθέσεις). Παρατηρήστε και πάλι τα ζευγάρια δυϊκών σχέσεων:

$$\begin{array}{llll} AB = BA & A+B = B+A & & \text{(αντιμεταθετική)} \\ A(BC) = (AB)C & \text{[συνήθως γράφεται: } ABC \text{]} & & \text{(προσεταιριστική)} \\ A+(B+C) = (A+B)+C & \text{[συνήθως γράφεται: } A+B+C \text{]} & & \end{array}$$

Άλλα Θεωρήματα της Άλγεβρας Boole:

Μπορούν εύκολα να αποδειχτούν τα παρακάτω επίσης θεωρήματα. Τα δύο θεωρήματα σε κάθε γραμμή --αριστερό και δεξί-- είναι *δυϊκά* μεταξύ τους: το ένα προκύπτει από το άλλο ανταλλάζοντας τα ΚΑΙ με τα Ή, και τα 1 με τα 0.

$$\begin{array}{ll} A \cdot 0 = 0 & A+1 = 1 \\ A \cdot 1 = A & A+0 = A \\ A \cdot A = A & A+A = A \\ A \cdot A' = 0 & A+A' = 1 \\ A(A+B) = A & A+AB = A \\ A(A'+B) = AB & A+A'B = A+B \end{array}$$

Άσκηση 4.6: Αποδείξεις Θεωρημάτων

[Προκειμένου να αποφευχθεί ο κίνδυνος υπερφόρτωσής σας με ασκήσεις αυτή τη βδομάδα, την άσκηση αυτή, μαζί με την επόμενη, θα την παραδώσετε (σε χαρτί) μαζί με την αναφορά σας του 5ου εργαστηρίου, την επόμενη βδομάδα, 18-22 Νοεμβρίου. Όμως, κάντε την από τώρα, προκειμένου να μην υπερφορτωθεί στη συνέχεια η επόμενη βδομάδα σας...].

(α) Αποδείξτε όλα τα παραπάνω θεωρήματα μέσω εξαντλητικού ελέγχου της ταυτότητας των δύο μελών τους σε όλες τις περιπτώσεις συνδυασμού τιμών των μεταβλητών τους. Με άλλα λόγια, φτιάξτε τους πίνακες αληθείας των δύο μελών κάθε θεωρήματος, και διαπιστώστε ότι είναι ίδιοι. Γράψτε τον πίνακα αληθείας της κάθε εμπλεκόμενης συνάρτησης σε μορφή στήλης, κατακόρυφα (όχι χάρτη Karnaugh).

(α1) Για τα θεωρήματα που εμπλέκουν μία μόνο μεταβλητή, ο πίνακας αληθείας θα έχει 2 γραμμές· φτιάξτε χωριστές στήλες για το A, το A', το (A)', το A·A, A+A, A·A', A+A', A·0, A·1, A+0, και A+1. Δείξτε με βέλη ποιές στήλες είναι ίσες με ποιές, και σε ποιο θεώρημα αντιστοιχεί κάθε τέτοιο ζευγάρι ίσων στηλών.

(α2) Για τα θεωρήματα που εμπλέκουν δύο μεταβλητές, A και B, ο πίνακας αληθείας θα έχει 4 γραμμές· φτιάξτε χωριστές στήλες για τα A', B', A'+B', A'B', AB, BA, (AB)', A+B, B+A, (A+B)'. Επίσης, φτιάξτε στήλες για τα AB, A+AB, A+B, A(A+B), A', A'B, A+A'B, A'+B, A(A'+B). Δείξτε πάλι τις ίσες στήλες και τα θεωρήματα στα οποία αυτές αντιστοιχούν.

(α3) Για τα θεωρήματα που εμπλέκουν τρεις μεταβλητές, A, B, και C, ο πίνακας αληθείας θα έχει 8 γραμμές· φτιάξτε χωριστές στήλες για τα B+C, A(B+C), AB, AC, AB+AC, BC, A+(BC), A+B, A+C, (A+B)(A+C), καθώς και για τα AB, (AB)C, BC, A(BC), A+B, (A+B)+C, B+C, A+(B+C). Πάλι, δείξτε τις ίσες στήλες.

(β) Αποδείξτε με διαγράμματα Venn την επιμεριστική και την προσεταιριστική ιδιότητα, στις δύο δυϊκές τους μορφές την κάθε μία, καθώς και τα θεωρήματα $A(A+B)=A$, $A+AB=A$, $A(A'+B)=AB$, και $A+A'B=A+B$. Χρησιμοποιήστε 2 ή 3 τεμνόμενες ελλείψεις, που παριστάνουν τα σύνολα A, B, και C. Σημειώστε με κατάλληλο χρώμα ή διαγράμμιση τις περιοχές του επιπέδου που αντιστοιχούν στα B+C, A(B+C), AB, AC, AB+AC, BC, A+(BC), A+B, A+C, (A+B)(A+C)· ποιές περιοχές είναι ίδιες με ποιές; Κάντε το ίδιο για τις περιοχές AB, (AB)C, BC, A(BC), A+B, (A+B)+C, B+C, A+(B+C)s, και τέλος για τις AB, A+AB, A+B, A(A+B), A'B, A+A'B, A'+B, A(A'+B).

Άσκηση 4.7: Αποκλειστικό-Ή και Αποκλειστικό-ΟΥΤΕ (Ισότητα)

[Προκειμένου να αποφευχθεί ο κίνδυνος υπερφόρτωσής σας με ασκήσεις αυτή τη βδομάδα, την άσκηση αυτή, μαζί με την προηγούμενη, θα την παραδώσετε (σε χαρτί) μαζί με την αναφορά σας του 5ου εργαστηρίου, την επόμενη βδομάδα, 18-22 Νοεμβρίου. Όμως, κάντε την από τώρα, προκειμένου να μην υπερφορτωθεί στη συνέχεια η επόμενη βδομάδα σας...].

Είδαμε στο [πείραμα 1.2](#) ότι το *αποκλειστικό-Ή* ("exclusive-OR" ή "XOR") δύο μεταβλητών είναι αληθές τότε και μόνο τότε όταν μία και μόνο μία από τις δύο τους είναι αληθής. Επίσης, η συνάρτηση ισότητας είναι αληθής όποτε και οι δύο μεταβλητές έχουν την ίδια τιμή.

(α) Φτιάξτε τους πίνακες αληθείας των δύο αυτών συναρτήσεων, και αποδείξτε μέσω αυτών ότι η συνάρτηση ισότητας είναι η άρνηση (το "συμπλήρωμα") της συνάρτησης αποκλειστικού-Ή. Για το λόγο αυτό, η συνάρτηση ισότητας ονομάζεται και "*αποκλειστικό-ΟΥΤΕ*" ("exclusive-NOR" ή "XNOR").

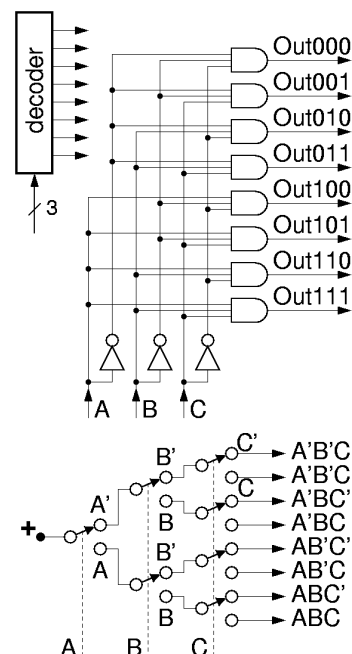
(β) Από τον χάρτη Karnaugh του αποκλειστικού-Ή έχουμε δει ότι αυτό είναι: $A \text{ XOR } B = AB'+A'B$. Αποδείξτε μέσω

αλγεβρικών μετασχηματισμών ότι ισχύει επίσης: $A \text{ XOR } B = (A+B)(A'+B')$. Ξεκινήστε από αυτή τη δεύτερη έκφραση, και εφαρμόστε πάνω της δύο φορές την επιμεριστική ιδιότητα του ΚΑΙ πάνω στο Ή, $A(B+C)=AB+AC$. Στη συνέχεια, απλοποιήστε τους τέσσερεις όρους που προκύπτουν, μέχρι να φτάσετε στην πρώτη έκφραση.

(γ) Αφού η συνάρτηση ισότητας είναι η άρνηση του αποκλειστικού-Ή, θα ισχύει: $A \text{ XNOR } B = [A \text{ XOR } B]' = [AB'+A'B]' = [(A+B)(A'+B')]'$. Απλοποιήστε αυτές τις δύο τελευταίες εκφράσεις, εφαρμόζοντας το θεώρημα DeMorgan, μέχρι να φτάσετε να αποδείξετε ότι: $A \text{ XNOR } B = AB+A'B' = (A+B')(A'+B)$.

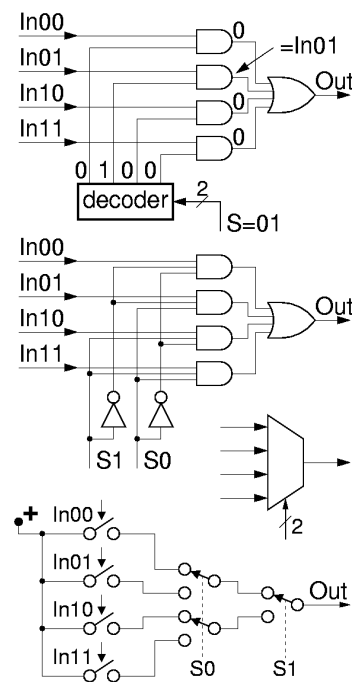
4.8 Ο Αποκωδικοποιητής με Πύλες:

Όπως είπαμε στο εργαστήριο 1, ο αποκωδικοποιητής είναι ένα ψηφιακό κύκλωμα που έχει μία έξοδο για κάθε ένα, χωριστό συνδυασμό τιμών των εισόδων του. Η κάθε έξοδος ανάβει (γίνεται 1) μόνον όταν οι τιμές των εισόδων βρίσκονται στον αντίστοιχο συνδυασμό. Επομένως, κάθε φορά, μία και μόνο μία έξοδος είναι αναμένη --αυτή η οποία αντιστοιχεί στον παρόντα συνδυασμό τιμών των εισόδων. Από τα παραπάνω προκύπτει ότι ο πίνακας αληθείας της κάθε εξόδου, σε μορφή χάρτη Karnaugh, θα είναι παντού 0 εκτός ενός και μόνου τετραγώνου όπου θα είναι 1. Άρα, η συνάρτηση της κάθε εξόδου αντιστοιχεί στο λογικό ΚΑΙ όλων των μεταβλητών εισόδου --είτε αυτών καθεαυτών, είτε των συμπληρωμάτων τους-- κάθε έξοδος θα έχει διαφορετικό συνδυασμό αληθών/συμπληρωμάτων για τις μεταβλητές εισόδου. Έτσι, η υλοποίηση του αποκωδικοποιητή με λογικές πύλες είναι αυτή που φαίνεται στο σχήμα --εδώ για την περίπτωση αποκωδικοποιητή 3-σε-8. Πάνω αριστερά στο σχήμα υπάρχει ένα συνηθισμένο, απλό σύμβολο για τον αποκωδικοποιητή: το σύρμα εισόδου με την πλάγια γραμμούλα και το "3" σημαίνει "τρία (αδελφά) σήματα (bits)", όπως είχαμε δει και στο εργαστήριο 2. Στο κάτω μέρος του σχήματος φαίνεται και η παλαιά υλοποίηση του εργαστηρίου 1, με διακόπτες. Σε αυτήν βλέπουμε, π.χ., ότι ρεύμα θα φτάσει στην επάνω έξοδο όταν και μόνον όταν οι διακόπτες A, B, και C είναι όλοι "όχι πατημένοι", δηλαδή όταν $(A')\text{KAI}(B')\text{KAI}(C')$. Ομοίως, στην δεύτερη έξοδο θα φτάσει ρεύμα όταν και μόνον όταν A και B είναι "όχι πατημένοι" και C είναι πατημένος, δηλαδή όταν $(A')\text{KAI}(B')\text{KAI}(C)$, κ.ο.κ. Παρατηρούμε ότι οι λογικές αυτές συναρτήσεις είναι ίδιες με τις αντίστοιχες του κυκλώματος με πύλες.



4.9 Ο Πολυπλέκτης με Πύλες:

Για πολυπλέκτες μιλήσαμε στο εργαστήριο 2, και είπαμε ότι έχουν μίαν είσοδο επιλογής (με όσα bits χρειάζεται) η οποία με την τιμή της επιλέγει μία από τις εισόδους δεδομένων, και στη συνέχεια η τιμή αυτής της επιλεγμένης εισόδου δεδομένων οδηγείται και δίδεται στην έξοδο· αλλάζοντας την τιμή των bits επιλογής αλλάζει και το ποιά είσοδος δεδομένων οδηγείται στην έξοδο. Επίσης, στο πείραμα 2.1 παρατηρήσαμε ότι ο πολυπλέκτης έχει μια συγγένεια με τον αποκωδικοποιητή, επειδή τα bits επιλογής πρέπει να αποκωδικοποιηθούν, ούτως ώστε για κάθε διαφορετικό συνδυασμό τους να προκληθεί διαφορετική ροή πληροφοριών. Αυτό φαίνεται στο επάνω μέρος του σχήματος: σε κάθε συνδυασμό τιμών των bits επιλογής, δηλαδή σε κάθε έξοδο του αποκωδικοποιητή, αντιστοιχεί κι από μία είσοδος δεδομένων, In00 έως In11, καθώς και από μία πύλη ΚΑΙ. Στο παράδειγμα που δείχνει το σχήμα (πολυπλέκτης 4-σε-1), τα (δύο) bits επιλογής έχουν τιμή 01, και γι' αυτό η δεύτερη έξοδος του αποκωδικοποιητή είναι αναμένη (1), και φυσικά όλες οι άλλες σβηστές (0). Επειδή $A \cdot 0 = 0$ για οιοδήποτε A, όλες οι πύλες ΚΑΙ εκτός από την μία "επιλεγμένη" πύλη δίνουν έξοδο 0. Επειδή $A \cdot 1 = A$ για οιοδήποτε A, η επιλεγμένη πύλη δίνει στην έξοδό της τιμή ίση με αυτήν της εισόδου δεδομένων της --δηλαδή της In01 στο εδώ παράδειγμα με $S=01$. Στη συνέχεια, οι έξοδοι όλων των πυλών ΚΑΙ οδηγούνται στις εισόδους μίας μεγάλης πύλης Ή. Επειδή $A+0 = A$ για οιοδήποτε A, οι εισόδου 0 της πύλης Ή --που αντιστοιχούν σε όλες τις πύλες ΚΑΙ πλην της επιλεγμένης-- δεν επηρεάζουν την τιμή εξόδου της Ή· έτσι, η τελική αυτή έξοδος παίρνει την τιμή της μίας εισόδου της πύλης Ή που δεν είναι αναγκαστικά 0, δηλαδή της εξόδου της επιλεγμένης πύλης ΚΑΙ, η οποία όπως είδαμε ισούται με την τιμή της επιλεγμένης εισόδου δεδομένων (της In01 εδώ, για $S=01$).



Επειδή ο αποκωδικοποιητής αποτελείται και αυτός από πύλες ΚΑΙ, και επειδή $(AB)C = A(BC) = ABC$, οι πύλες ΚΑΙ του αποκωδικοποιητή μπορούν να συνενωθούν με τις πύλες ΚΑΙ που αυτές οδηγούν, δίνοντας το κύκλωμα που φαίνεται στο μέσον του σχήματος. Στο κάτω μέρος του σχήματος υπάρχει το παλαιό κύκλωμα του πολυπλέκτη με διακόπτες, από το πείραμα 2.1· παρατηρούμε ότι ουσιαστικά πρόκειται για την ίδια λογική συνάρτηση: ρεύμα μπορεί να περάσει από την θετική τροφοδοσία προς την έξοδο Out όταν βρεί διέξοδο μέσα από έναν από τέσσερεις εναλλακτικούς δρόμους (4 παράλληλοι δρόμοι αντιστοιχούν στο λογικό Ή 4 εισόδων). Ο πρώτος δρόμος άγει όταν In00 πατημένος και S0 και S1 όχι πατημένοι, δηλαδή όταν $(In00)(S0')(S1')$, που αντιστοιχεί στην πρώτη πύλη ΚΑΙ του νέου κυκλώματος, κ.ο.κ. για τους άλλους τρεις εναλλακτικούς δρόμους.

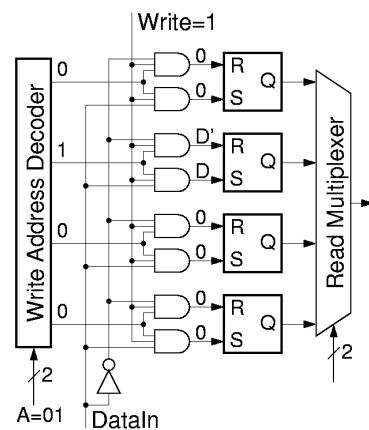
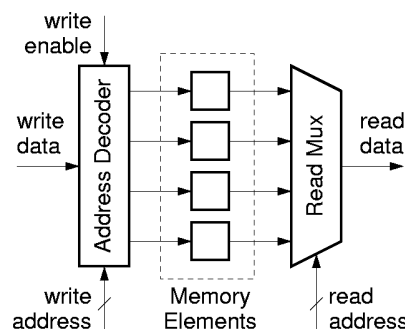
4.10 Η Μνήμη Τυχαίας Προσπέλασης (RAM):

Η *Μνήμη Τυχαίας Προσπέλασης* (Random Access Memory - **RAM**) αποτελεί τη μιά από τις τρεις βασικές συνιστώσες των ψηφιακών συστημάτων --αν θεωρήσουμε ότι οι επεξεργαστές και τα δίκτυα επικοινωνίας είναι οι άλλες δύο. Ρόλος της μνήμης είναι να κρατάει πληροφορίες που είχαν γραφτεί εκεί στο παρελθόν, μέχρις ότου δεν τις χρειαζόμαστε άλλο πιά. Στο πείραμα 2.5 είχαμε δει τη βασική ιδέα για την κατασκευή ενός στοιχείου μνήμης που μπορεί να "θυμάται" 1 bit πληροφορίας. Οι σύγχρονες μνήμες RAM ημιαγωγών (ιδιαίτερα οι "στατικές" RAM - SRAM) χρησιμοποιούν ένα κύκλωμα CMOS βασισμένο στην ίδια ιδέα της θετικής ανάδρασης για να αποθηκεύουν το κάθε ένα bit πληροφορίας, και ως γνωστόν περιλαμβάνουν χιλιάδες ή (συνήθως) εκατομμύρια τέτοια κυκλώματα μέσα σε κάθε ένα chip, προκειμένου να χωρούν χιλιάδες ή εκατομμύρια bits πληροφορίας. Προφανώς όμως, τα chips αυτά δεν μπορούν να έχουν χιλιάδες ή εκατομμύρια σύρματα προκειμένου εμείς απ' έξω να διαβάζουμε ή να γράφουμε όλα αυτά τα bits. Η "προσπέλαση" της μνήμης, λοιπόν, γίνεται επιλεκτικά κάθε φορά, για την ανάγνωση από ή την εγγραφή σε ένα υποσύνολο μόνο των στοιχείων μνήμης. Το κύκλωμα έχει τη δυνατότητα να επιλέγει αυθαίρετα --"τυχαία"-- κάθε φορά το οιοδήποτε στοιχείο μνήμης επιθυμεί να προσπελάσει ο χρήστης, δίνοντας έτσι το όνομα σε αυτόν τον τύπο μνήμης. Το όνομα "τυχαίας προσπέλασης" αντιδιαστέλλει αυτή τη μνήμη από άλλες μνήμες, π.χ. σειριακής προσπέλασης, όπως είναι π.χ. οι μαγνητικές ταινίες, ή --εν μέρει-- οι μαγνητικοί δίσκοι.

Για να επιτευχθεί η τυχαία προσπέλαση, η RAM χρησιμοποιεί έναν αποκωδικοποιητή για να επιλέξει το επιθυμητό υποσύνολο των στοιχείων μνήμης που ο χρήστης θέλει να προσπελάσει. Μπορούμε να φανταστούμε ότι ο αποκωδικοποιητής χρησιμοποιείται για τις εγγραφές (write) στη μνήμη, όπως δείχνει το σχήμα, αν και στην πραγματικότητα, για λόγους οικονομίας, χρησιμοποιείται και για τις αναγνώσεις. Η βασική είσοδος του αποκωδικοποιητή είναι η **διεύθυνση** (address), δηλαδή μία ψηφιακή πληροφορία με τόσα bits όσα χρειάζονται για να επιλεγεί μονοσήμαντα το ένα από τα υποσύνολα στοιχείων μνήμης στο οποίο θα γίνει η προσπέλαση. Οι άλλες εισόδους που χρειάζονται για την εγγραφή είναι μία είσοδος ελέγχου (write enable), που να λέει πότε θέλουμε να γράψουμε και πότε όχι, και η είσοδος δεδομένων (write data), που να λέει τι θέλουμε να γράψουμε στο μέρος που έχει επιλέξει η είσοδος διεύθυνσης. Για την τυχαία προσπέλαση κατά την ανάγνωση, η RAM χρησιμοποιεί έναν πολυπλέκτη για να οδηγήσει στην έξοδο το περιεχόμενο που είναι αποθηκευμένο στο επιθυμητό υποσύνολο των στοιχείων μνήμης. Φυσικά, ο πολυπλέκτης εμπεριέχει κι έναν αποκωδικοποιητή, και για να λειτουργήσει χρειάζεται κι αυτός τη **διεύθυνση** του επιθυμητού υποσυνόλου στοιχείων μνήμης. Έτσι, για κάθε δοσμένη διεύθυνση ανάγνωσης (read address) θα εμφανίζονται στην έξοδο δεδομένων ανάγνωσης (read data) εκείνες οι πληροφορίες που είχαν γραφτεί την τελευταία φορά στην ίδια διεύθυνση εγγραφής, ενώ τα περιεχόμενα (πληροφορίες) των υπολοίπων θέσεων (διευθύνσεων) της μνήμης είναι αυθαίρετα, και δεν αλληλεπιδρούν με τα περιεχόμενα της θέσης που εμείς κοιτάμε αυτή τη στιγμή.

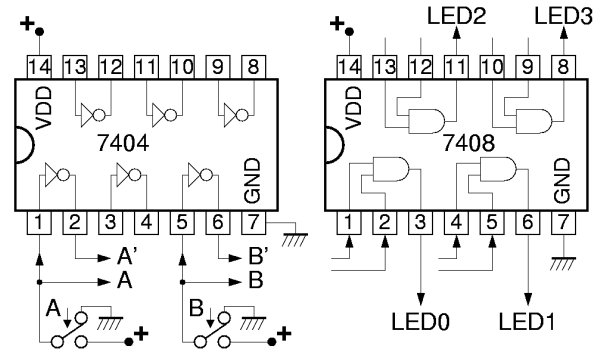
Για να συγκεκριμενοποιήσουμε την παραπάνω γενική εικόνα, ας υποθέσουμε ότι κάθε προσπέλαση στη μνήμη γράφει ή διαβάζει 1 bit, για απλότητα (στις συνηθισμένες μνήμες κάθε προσπέλαση αφορά π.χ. 8 ή 16 ή 32 ή 64 bits). Επίσης, για να μας χωράει το σχήμα, ας υποθέσουμε ότι η μνήμη μας χωράει μόνο 4 bits, συνολικά (αντί των χιλιάδων ή εκατομμυρίων bits των πραγματικών μνημών). Επίσης, κατ' αναλογία προς τις εισόδους Reset και Set του πειράματος 2.5, ας υποθέσουμε ότι κάθε στοιχείο μνήμης έχει δύο εισόδους, R και S, για τον έλεγχο εγγραφής: όταν R=S=0 δεν γίνεται καμία εγγραφή, και το στοιχείο διατηρεί (θυμάται) την προϋπάρχουσα κατάστασή του· όταν R=1 (καθώς S=0), το στοιχείο μηδενίζεται (εγγραφή πληροφορίας 0)· όταν S=1 (καθώς R=0), στο στοιχείο εγγράφεται η πληροφορία 1· και τέλος, θέλουμε ποτέ να μην είναι ταυτόχρονα αναμένο και το R και το S. Q είναι η έξοδος του στοιχείου μνήμης, δηλαδή το σύρμα που έχει πάντα πάνω του την τιμή του αποθηκευμένου bit. Τότε, η μνήμη αυτή, μεγέθους "4x1" (4 "λέξεις", μεγέθους 1 bit καθεμία), θα είναι όπως φαίνεται στο σχήμα. Η ανάγνωση γίνεται με τον πολυπλέκτη 4-σε-1, δηλ. το κύκλωμα της παραγράφου 4.9.

Η εγγραφή χρησιμοποιεί τον αποκωδικοποιητή διευθύνσεων, που είναι παρόμοιος με το κύκλωμα της παραγράφου 4.8, παραπάνω. Οι πύλες KAI που ακολουθούν τον αποκωδικοποιητή εξασφαλίζουν τα εξής. Για όλα τα στοιχεία μνήμης εκτός του "επιλεγμένου" από τη διεύθυνση εγγραφής (εδώ, το δεύτερο στοιχείο, επειδή A=01), R=S=0 επειδή η αντίστοιχη έξοδος του αποκωδικοποιητή είναι 0, άρα δεν γίνεται καμία εγγραφή εκεί. Για το επιλεγμένο στοιχείο μνήμης, όταν Write=0, δηλαδή όταν δεν θέλουμε να κάνουμε καμία εγγραφή, τότε πάλι R=S=0, επομένως ούτε εκεί γίνεται εγγραφή. Εάν τώρα θέλουμε να γίνει εγγραφή, δηλαδή εάν Write=1 όπως στο σχήμα, τότε το επιλεγμένο στοιχείο μνήμης βλέπει R=DataIn' και S=DataIn (επειδή οι άλλες δύο εισόδους των πυλών KAI είναι 1). Αυτό σημαίνει ότι εάν DataIn=0 τότε R=1 και S=0, άρα το στοιχείο μνήμης μηδενίζεται (δηλ. εγγράφεται 0=DataIn), ενώ εάν DataIn=1 τότε R=0 και S=1, άρα στο στοιχείο μνήμης εγγράφεται 1 (=DataIn). Η περίπτωση R=S=1 αποκλείεται, διότι τα DataIn' και DataIn (το ένα συμπλήρωμα του άλλου) δεν είναι ποτέ 1 και τα δύο ταυτόχρονα.



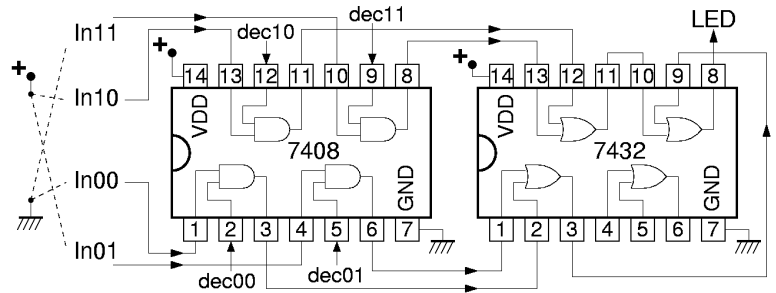
Πείραμα 4.11: Αποκωδικοποιητής 2-σε-4

Κατασκευάστε και ελέγξτε έναν αποκωδικοποιητή 2-σε-4 με πύλες, σύμφωνα με όσα είπαμε παραπάνω στην παράγραφο 4.8. Χρησιμοποιήστε ένα chip 7404 (πύλες NOT) και ένα chip 7408 (πύλες AND) όπως αυτά που είδαμε στο πείραμα 3.11, και των οποίων οι ακροδέκτες επαναλαμβάνονται στο σχήμα εδώ για τη διευκόλυνσή σας. Τροφοδοτήστε τις δύο εισόδους A και B όπως στο σχήμα, δηλαδή όπως και στο πείραμα 3.11. Συνδέστε τις 4 εξόδους σε 4 ενδεικτικές λυχνίες. **Πριν** φτάστε στο εργαστήριο σχεδιάστε την πλήρη συνδεσμολογία του κυκλώματος (τις γραμμές που λείπουν στο σχήμα). Στο εργαστήριο, ελέγξτε τους 4 συνδυασμούς των δύο εισόδων A και B, και διαπιστώστε αν ανάβει πάντα μία και μόνο μία ενδεικτική λυχνία --η σωστή κάθε φορά. Όταν τελειώσετε **μην** χαλάσετε το κύκλωμά σας, διότι θα το χρειαστείτε στο επόμενο πείραμα.



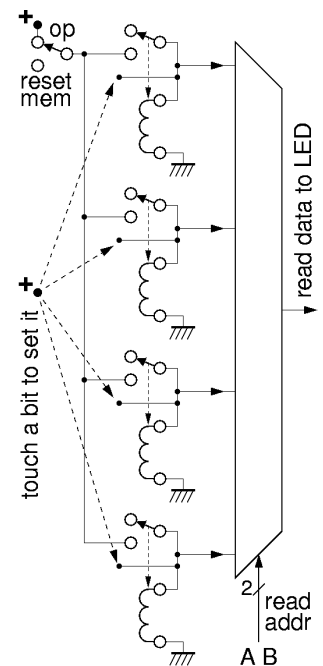
Πείραμα 4.12: Πολυπλέκτης 4-σε-1

Κατασκευάστε και ελέγξτε έναν πολυπλέκτη 4-σε-1 με πύλες, όπως το πρώτο κύκλωμα της παραγράφου 4.9 παραπάνω (χρησιμοποιούμε την πρώτη και όχι τη δεύτερη παραλλαγή του κυκλώματος επειδή δεν έχουμε πύλες AND 3 εισόδων στο εργαστήριό μας). Χρησιμοποιήστε τον αποκωδικοποιητή 2-σε-4 που έχετε έτοιμο από το προηγούμενο πείραμα 4.11, και προσθέστε τα απαιτούμενα κυκλώματα με ένα chip 7408 (πύλες AND) και ένα chip 7432 (πύλες OR) (βλ. πείραμα 3.11) των οποίων οι ακροδέκτες επαναλαμβάνονται στο σχήμα εδώ για τη διευκόλυνσή σας. Επειδή δεν έχουμε πύλες OR 4 εισόδων στο εργαστήριό μας, χρησιμοποιήστε 3 πύλες των 2 εισόδων σε σύνδεση δένδρου, όπως στο σχήμα. Συνδέστε την έξοδο σε μία ενδεικτική λυχνία. Ελέγξτε εν τάχει το κύκλωμά σας, δίνοντας εισόδους δεδομένων (In00 ως In11) μέσω πρόχειρων συνδέσεων στη θετική ή αρνητική τροφοδοσία τις οποίες μπορείτε να αλλάζετε κουνώντας σύρματα κάθε φορά (αν έχετε το χρόνο, μπορείτε να κάνετε σωστότερο έλεγχο συνδέοντας 4 μεταγωγούς διακόπτες με δύο τάσεις τροφοδοσίας καθένας). Η ενδεικτική λυχνία δείχνει πάντα την τιμή της σωστής "επιλεγμένης" εισόδου δεδομένων. Όταν τελειώσετε **μην** χαλάσετε το κύκλωμά σας, διότι θα το χρειαστείτε στο επόμενο πείραμα.



Πείραμα 4.13: Μνήμη Ηλεκτρονόμεν με Πολυπλέκτη Ανάγνωσης

Κατασκευάστε και ελέγξτε την απλοϊκή μνήμη τυχαίας προσπέλασης (RAM) μεγέθους 4x1 με ηλεκτρονόμους που φαίνεται στο σχήμα, κατ' αναλογία της μνήμης που είδαμε παραπάνω στην παράγραφο 4.10. Για λόγους απλοποίησης, παραλείπουμε εντελώς το κύκλωμα εγγραφών, και το αντικαθιστούμε με τις πρόχειρες συνδέσεις που φαίνονται στο σχήμα (κατά συνέπεια, η μνήμη αυτή είναι RAM μόνον όσον αφορά τις αναγνώσεις, και όχι όσον αφορά τις εγγραφές). Τα 4 στοιχεία μνήμης είναι κατασκευασμένα με 4 ηλεκτρονόμους, κατ' αναλογία του πειράματος 2.5. Οι αναγνώσεις γίνονται μέσω του πολυπλέκτη 4-σε-1 με πύλες που έχετε έτοιμο από το προηγούμενο πείραμα 4.12. Η σύνδεση ηλεκτρονόμεν και πυλών ημιαγωγών, εδώ, επιτρέπεται, διότι τα σημεία που τροφοδοτούν τις πύλες δεν είναι ποτέ ανοικτοκυκλωμένα, και η τάση τους είναι πάντα μεταξύ 0 και 5 Volt: όταν ο ηλεκτρονόμος είναι σβηστός, το σημείο που τροφοδοτεί την πύλη ΚΑΙ έχει 0 Volt, δεδομένου ότι η πεπερασμένη αντίσταση του πηνίου το συνδέει με τη γή, ενώ όταν ο ηλεκτρονόμος είναι αναμένος, το σημείο εκείνο συνδέεται με την θετική τροφοδοσία, άρα έχει 5 Volt.



Οι εγγραφές στη μνήμη μας γίνονται με ένα τρόπο διόλου βολικό, λόγω των απλοποιήσεων του κυκλώματος που αναγκαστήκαμε να κάνουμε. Στην αρχή, μηδενίζουμε τα περιεχόμενα ολόκληρης της μνήμης, φέρνοντας το μεταγωγό διακόπτη αριστερά στη θέση "reset memory", δηλαδή διακόπτοντας την τροφοδοσία σε όλα τα στοιχεία μνήμης. Εν συνεχεία, επαναφέρουμε την τροφοδοσία (διακόπτης στη θέση op - operation), και γράφουμε άσους (1) επιλεκτικά, σε όποια στοιχεία μνήμης θέλουμε, ακουμπώντας με το χέρι ένα σύρμα, που η μία του άκρη είναι στη θετική τροφοδοσία, στον ακροδέκτη ενεργοποίησης του πηνίου που θέλουμε να ανάψει: μόλις το πηνίο ανάψει μπορούμε να απομακρύνουμε το σύρμα, αφού το στοιχείο μνήμης "θυμάται" μόνο του ότι το ανάψαμε. Για να αλλάξουμε μερικούς ή όλους τους άσους σε μηδενικά, ο μόνος τρόπος είναι να ξαναμηδενίσουμε όλη τη μνήμη μέσω του διακόπτη "reset memory" και να γράψουμε άσους από την αρχή σε όλα τα μέρη όπου τους θέλουμε.

Ελέγξτε τη σωστή λειτουργία της μνήμης ως εξής. Γράψτε σ' ένα χαρτί 4 bits (π.χ. 1, 0, 0, 1) που θέλετε να αποθηκεύσετε στη

μνήμη, με τη σειρά που θέλετε να τα αποθηκεύσετε. Μετά, γράψτε αυτά τα bits στη μνήμη, με τον χειροκίνητο τρόπο που είπαμε παραπάνω. Στη συνέχεια, δώστε διάφορες διευθύνσεις A και B (00, 01, 10, 11), εναλλάξ, επανειλημμένα, και ανακατωμένα. Για κάθε διεύθυνση, η ενδεικτική λυχνία στην έξοδο της μνήμης (του πολυπλέκτη) σας δίνει το σωστό περιεχόμενο σύμφωνα με το τι είχατε γράψει εκεί; Επαναλάβετε από την αρχή, με διαφορετικές πληροφορίες εγγραφής στη μνήμη (π.χ. 0, 1, 1, 1). Υπάρχουν 16 διαφορετικά σύνολα πληροφοριών που μπορείτε να αποθηκεύσετε σε αυτή τη μνήμη --αν έχετε χρόνο, ελέγξτε τη σωστή λειτουργία της μνήμης με όσα περισσότερα από αυτά γίνεται (φυσικά, δεν θα ελέγχατε με τέτοιο εξαντλητικό τρόπο μία μνήμη π.χ. του 1 Mbit, διότι δεν θα τελειώνατε ποτέ (2 εις την ένα εκατομμύριο συνδυασμοί...) --πρέπει όμως να ελέγξτε ότι δεν είναι καμένο κανένα από τα στοιχεία μνήμης και καμία από τις πύλες του πολυπλέκτη).

[Up to the Home Page of CS-120](#)

© copyright University of Crete, Greece.
Last updated: 5 Nov. 2003, by [M. Katevenis](#).