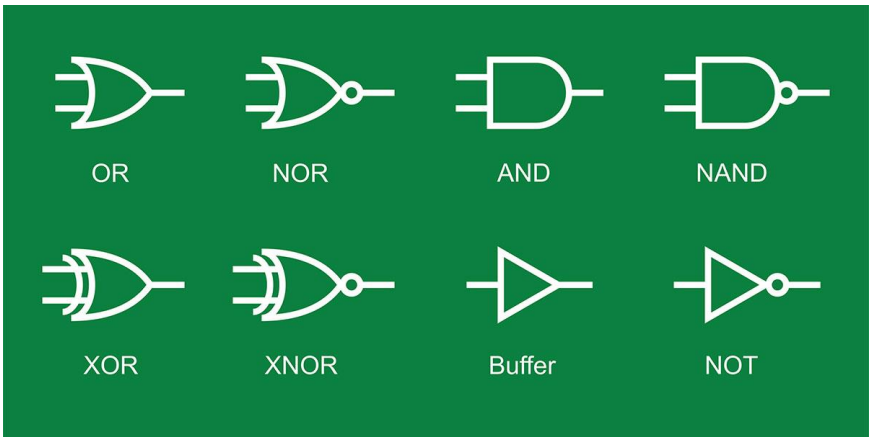
ΕΛ.ΜΕ.ΠΑ.

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ ΗΜΜΥ

ΛΟΓΙΚΗ ΣΧΕΔΙΑΣΗ



Ν.Κορνήλιος

Καθηγητής

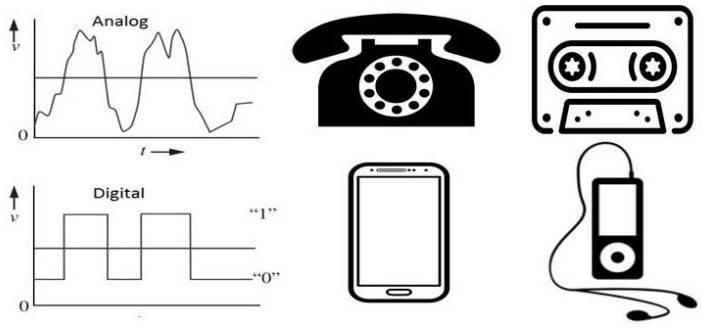
2021

ΠΕΡΙΕΧΟΜΕΝΑ

Εισαγωγή

Ένα αναλογικό ηλεκτρονικό κύκλωμα είναι ένα σύνολο από αλληλοεπιδρώντα εξαρτήματα τα οποία επικοινωνούν μεταξύ τους με αναλογικά ηλεκτρικά σήματα και εκτελούν μία συγκεκριμένη εργασία.

Ένα αναλογικό σήμα είναι ένα σήμα το οποίο μπορεί να πάρει οποιαδήποτε τιμή από ένα συνεχές εύρος τιμών τάσης, ρεύματος ή ενός άλλου φυσικού μεγέθους. Σε αντίθεση με το αναλογικό, ένα ψηφιακό σήμα χρησιμοποιεί διακριτά στοιχεία πληροφορίας για να λειτουργήσει. Στα ψηφιακά ηλεκτρονικά το στοιχειώδες κύτταρο πληροφορίας ονομάζεται BIT και μπορεί να πάρει δύο δυνατές τιμές, 0 ή 1. Έτσι ένα ψηφιακό σήμα μπορεί να πάρει για κάθε χρονική στιγμή μία και μόνο μία από δύο διαθέσιμες διακριτές τιμές που είναι το μηδέν 0 και το 1. Τίποτα παραπάνω. Οι δύο αυτές τιμές 0 ή 1 μπορούν να λεχθούν ψευδές ή αληθές γεγονός, δηλαδή FALSE ή TRUE στα αγγλικά, LOW ή HIGH, απενεργοποιημένο ή ενεργοποιημένο ή οιανδήποτε άλλη κωδικοποίηση εμείς επινοήσουμε αρκεί να υπάρχει αντιστοιχία με το λογικό 0 και το λογικό 1. Στο παρακάτω σχέδιο βλέπουμε ένα αναλογικό σήμα με το οποίο λειτουργούσαν τα παλιά τηλέφωνα και ένα ψηφιακό της σύγχρονης κινητής τηλεφωνίας.



Στα Ψηφιακά Ηλεκτρονικά ή στα Λογικά Κυκλώματα, ή στη Ψηφιακή Σχεδίαση όπως συνηθίζουμε να ονομάζουμε το αντικείμενο, η κυκλωματική λειτουργία τους στηρίζεται σε ένα αυστηρό μαθηματικό υπόβαθρο που ονομάζουμε άλγεβρα Boole ή θεωρία των δισταθών ψηφιακών συστημάτων, έτσι ονομάζονται επειδή έχουν δύο και μόνο δύο σταθερές καταστάσεις. Η ανάπτυξη αυτής της θεωρίας οφείλεται στον Άγγλο Μαθηματικό George Boole (1815-64).

Η θεωρία του βασίζεται στις ιδέες του Αριστοτέλη όπου ένα γεγονός αληθές παρουσιάζεται με το ένα 1 και ένα ψευδές με το μηδέν 0. Πολλά χρόνια αργότερα ο Shannon ανέπτυξε τη θεωρία διακοπτών "SWITCHING THEORY" βασιζόμενος στην Άλγεβρα του Boole και έθεσε τις βάσεις για την ανάπτυξη και διαχείριση της ψηφιακής πληροφορίας.

Οι σημειώσεις αυτές έχουν γραφεί για το μάθημα Λογική Σχεδίαση του τμήματος ΗΜΜΥ του ΕΛ.ΜΕ.ΠΑ. Είναι ένα απλό βοήθημα και αποτελούν ένα ξεκίνημα για την κατανόηση και σχεδίαση ψηφιακών κυκλωμάτων. Επιπλέον βοήθεια θα λάβετε από το σύγγραμμα που θα σας διατεθεί μέσα από τον Εύδοξο αλλά και από την εκτενή βιβλιογραφία που θα βρείτε για κάθε θέμα που σας αφορά στο διαδίκτυο. Στο διαδίκτυο σήμερα υπάρχουν άπειρες δυνατότητες που μας επιτρέπουν να εμβαθύνουμε ή να κατανοήσουμε διάφορες έννοιες και κυκλώματα. Είναι τόσο πυκνή η διάθεση γνώσης που μπορεί ο καθένας να βρει τον τρόπο μέσα από τον οποίο θα έχει την πλήρη κατανόηση σε κάθε τι που τον ενδιαφέρει να καταλάβει σε βάθος και να εμπλουτίσει τις γνώσεις του.

Επί μέρους ανάπτυξη θεμάτων με ένα ιδιαίτερο προσωπικό ενδιαφέρον μπορεί να γίνει από την βιβλιογραφία που δίνεται στο τέλος των σημειώσεων.

Το μάθημα χωρίζεται σε δύο μεγάλες ενότητες.

Η πρώτη ενότητα αφορά τα συνδυαστικά κυκλώματα και γενικά τα κυκλώματα των οποίων η λειτουργία δεν εξαρτάται από τον χρόνο. Σε αυτά τα κυκλώματα οι έξοδοι είναι συνάρτηση αποκλειστικά των εισόδων. Αν οι είσοδοι παραμένουν αναλλοίωτες και οι έξοδοι επίσης.

Η δεύτερη ενότητα του μαθήματος περιλαμβάνει τα ακολουθιακά κυκλώματα τα οποία χρονίζονται από ένα κεντρικό ρολόι. Η λειτουργία τους συναρτάται άμεσα με τον χρόνο. Οι έξοδοι τους είναι συναρτήσεις των εισόδων και του ρολογιού χρονισμού.

Θα δούμε όμως ότι ένα περίπλοκο όχι αναγκαστικά λογικό κύκλωμα συνήθως περιλαμβάνει ένα μείγμα και των δύο κατηγοριών κυκλωμάτων, δηλαδή και συνδυαστικά και ακολουθιακά. Στις σημειώσεις αυτές θα δείτε μία περιγραφή των βασικών οικογενειών ολοκληρωμένων κυκλωμάτων με

ιδιαίτερη έμφαση στην οικογένεια TTL λόγω της μεγάλης διάδοσης αυτής της οικογένειας και του χαμηλού κόστους που έχει.

Παράλληλα με τη θεωρία το μάθημα για να ολοκληρωθεί απαιτεί την παρουσία και την υλοποίηση 5 εργαστηριακών ασκήσεων στο εργαστήριο. Για την υλοποίηση των ασκήσεων θα σας διανεμηθεί ηλεκτρονικά το αντίστοιχο φυλλάδιο ασκήσεων. Για να έχει επιτυχία η υλοποίηση των ασκήσεων στον εργαστηριακό χώρο, η οποία βοηθά στην εμπέδωση της θεωρίας απαιτείται καλή προετοιμασία της άσκησης πριν την προσέλευση στο εργαστήριο.

Απαιτείται επίσης καλή γνώση των διαθέσιμων στον πάγκο εργασίας οργάνων, της πλακέτας breadboard υλοποίησης του κυκλώματος της άσκησης όπως επίσης και το pin-out του ολοκληρωμένου κυκλώματος, το οποίο δίνει ο κατασκευαστής του ολοκληρωμένου κυκλώματος και δείχνει την λειτουργία που έχει ο κάθε ακροδέκτης του. Μας δίνει τον ακροδέκτη τροφοδοσίας του κυκλώματος, τον ακροδέκτη γείωσης, τις εισόδους και τις εξόδους των κυκλωμάτων που περιέχει και των πυλών.

Η καλή προετοιμασία είναι προϋπόθεση για μια επιτυχημένη εκτέλεση της άσκησης η οποία δίνει θετική αξιολόγηση στον φοιτητή και συμβάλει θετικά στην τελική βαθμολογία του μαθήματος.

Κεφάλαιο 1. Συστήματα Αρίθμησης & Κώδικες

Στην καθημερινή μας ζωή χρησιμοποιούμε τα γράμματα του αλφαβήτου και τους αριθμούς του δεκαδικού συστήματος αρίθμησης για να μπορέσουμε να επικοινωνήσουμε με ότι μας περιβάλλει. Τα στοιχεία αυτά τα ονομάζουμε διακριτά στοιχεία πληροφορίας.

Σε ένα ψηφιακό σύστημα τα διακριτά στοιχεία πληροφορίας είναι δύο όπως είδαμε προηγουμένως και γι' αυτό το λόγο τα ονομάζουμε δυαδικά. Το στοιχειώδες δομικό στοιχείο ή στοιχειώδες κύτταρο πληροφορίας ονομάζεται BIT και μπορεί να πάρει δύο δυνατές τιμές, 0 ή 1. Οι δύο αυτές λογικές τιμές αντιστοιχούν σε δύο επίπεδα τάσης. Το λογικό 0 σε αντιστοιχεί σε (0 ± ΔV) Volts ενώ το λογικό 1 σε μία τάση (V ± ΔV) Volts. Η τάση V είναι μια μικρή τάση, της τάξης των 2 έως 10 Volt. Η ακριβής τιμή καθορίζεται από την οικογένεια των ολοκληρωμένων κυκλωμάτων που χρησιμοποιούμε όπως θα δούμε αναλυτικότερα παρακάτω. Βλέπουμε στον παρακάτω πίνακα μερικά παραδείγματα με διάφορες υποτιθέμενες τεχνολογίες λογικής όπου μπορούμε να αντιστοιχίσουμε διάφορες καταστάσεις όπως αυτές περιγράφονται στον παρακάτω πίνακα με τα δυαδικά λογικά σήματα 0 και 1.

ΤΕΧΝΟΛΟΓΙΑ	**ΛΟΓΙΚΟ 0**	**ΛΟΓΙΚΟ 1**
Λογική ρελέ	Ανοικτό	Κλειστό
Λογική TTL	0-0,8V	2,0-5,0V
Λογική CMOS	0-0,5V	1,5-2,0V
Δυναμική Μνήμη	Πυκνωτής εκφορτισμένος	Πυκνωτής φορτισμένος
Οπτικές ίνες	Όχι φως	Φως
Δίοδος LED	Όχι φως	Φως
Οπτικός Δίσκος CD	Υπάρχουν εσοχές	Δεν υπάρχουν εσοχές

Σε ένα αναλογικό κύκλωμα οι καταστάσεις στην έξοδο του περιγράφονται από ένα μεγάλο εύρος τιμών τάσης και ρεύματος. Σε ένα λογικό κύκλωμα έχουμε μόνο δύο καταστάσεις. Η θετικότερη τάση ονομάζεται επίσης υψηλή τιμή ή High, και συμβολίζεται απλά με το γράμμα **H** της αγγλικής γλώσσας, ενώ η χαμηλή τιμή ονομάζεται Low και συμβολίζεται με το

γράμμα **L** επίσης της αγγλικής. Έτσι στη λογική σχεδίαση θα έχουμε ότι H=1 και L=0. Η παραπάνω επιλογή, δηλαδή H=1 και L=0 ονομάζεται θετική λογική και χρησιμοποιείται από την πλειονότητα των ψηφιακών κυκλωμάτων.Εδώ μπορούμε να πούμε ότι υπάρχει και η δυνατότητα επιλογής H=0 και L=1 την οποία ονομάζουμε αρνητική λογική, η οποία χρησιμοποιείται σε σπάνιες περιπτώσεις. Με την εξέλιξη της ψηφιακής τεχνολογίας πολλά αντικείμενα του αναλογικού μας κόσμου πέρασαν στον αναδυόμενο ψηφιακό. Παραθέτουμε παρακάτω μερικά παραδείγματα:

Φωτογραφία: Πριν από μερικά χρόνια οι φωτογραφικές μηχανές χρησιμοποιούσαν ειδικά φιλμ για την καταγραφή διαφόρων εικόνων. Σήμερα έχουν αντικατασταθεί με φθηνές ψηφιακές φωτογραφικές συσκευές που περιέχονται και σε όλα σχεδόν τα κινητά τηλέφωνα.Η απεικόνιση γίνεται σε pixels διαστάσεων 1920x1080 ή και περισσότερο. Τα ψηφιακά αρχεία που προκύπτουν μπορούν να δεχθούν του κάθε είδους επεξεργασία και τροποποίηση όπως και συμπίεση για την αποθήκευση τους στην μορφή JPEG.

Το JPEG είναι ένα πρότυπο συμπίεσης εικόνων. Δημιουργήθηκε από την ομάδα Joint Photographic Experts Group-JPEG από την οποία πήρε και το όνομα. Λόγω του μικρού μεγέθους αρχείου που μπορεί να προκύψει με αυτή τη μέθοδο συμπίεσης, χρησιμοποιείται κυρίως σε ιστοσελίδες και σε υψηλής ανάλυσης σύγχρονες φωτογραφικές μηχανές. Μια υψηλής ανάλυσης εικόνα η οποία δεν έχει συμπιεστεί μπορεί να χρησιμοποιεί έως και 40MB χώρου ενώ σε μορφή JPEG χρησιμοποιεί περίπου 3MB.

Βίντεο-Ταινίες: Οι ταινίες έχουν πάψει πιά να αποθηκεύονται σε φιλμ. Αποθηκεύονται σε μια εξαιρετικά συμπιεσμένη μορφή την οποία γνωρίζουμε ως MPEG4. Το MPEG-4 είναι μια μέθοδος ορισμού της συμπίεσης ψηφιακών δεδομένων ήχου και εικόνας. Εισήχθη στα τέλη του 1998 και δημιούργησε ένα πρότυπο και μια ομάδα μορφών κωδικοποίησης ήχου και εικόνας και γενικότερα οπτικοακουστική κωδικοποίηση.

Ήχος: Η καταγραφή του ήχου από το βινύλιο και την μαγνητική ταινία πέρασε στην ψηφιακή καταγραφή. Όπως ισχύει για τις φωτογραφίες έτσι και ο ήχος μπορεί να καταγραφεί, να συμπιεστεί και να αποδοθεί σε μορφή MP3.Το MP3, είναι ένα δημοφιλές πρότυπο ψηφιακής κωδικοποίησης του

ήχου, το οποίο βασίζεται στην αποτελεσματική συμπίεση αρχείων μέσω ενός αλγορίθμου σχεδιασμένου να μειώνει δραστικά το πλήθος των ψηφιακών δεδομένων που απαιτούνται για την αποθήκευση και ορθή αναπαραγωγή του ήχου.

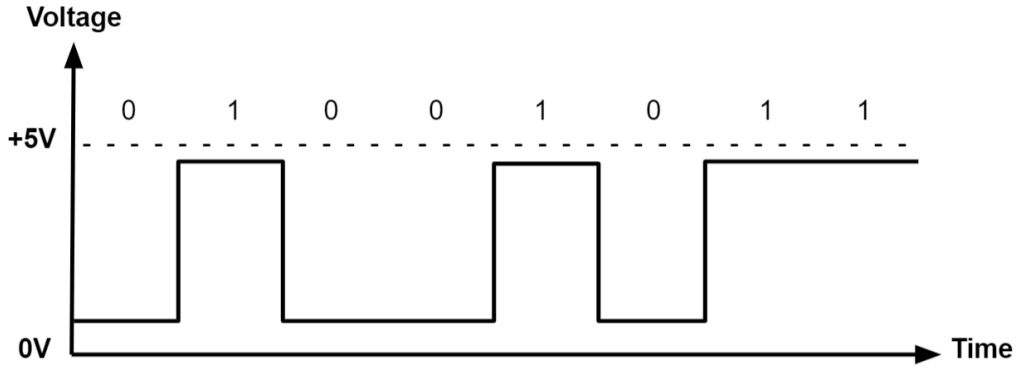
Τηλεφωνία: Με τα σύγχρονα ψηφιακά κυκλώματα πολύπλεξης, απόπλεξης και τη χρήση οπτικών ινών οι σύγχρονες τηλεπικοινωνίες είναι ψηφιακές. Όλες οι ψηφιακές διαδικασίες επικοινωνίας βασίζονται σε ένα πρωτόκολλο διαδικτύου (Internet Protocol IP). Έτσι μέσω μιας απλής ασύρματης γραμμής επικοινωνίας μπορούν ταυτόχρονα να ομιλούν χιλιάδες χρήστες, χωρίς κανένα πρόβλημα παρεμβολών, διακοπών ή ότιδήποτε άλλο συνέβαινε κατά κόρο στις παλιές αναλογικές επικοινωνίες.

Αυτοκίνηση: Η λειτουργία των κινητήρων εσωτερικής καύσης ελέγχεται πλέον μέσα από ένα μικροεπεξεργαστή. Ο έλεγχος αυτός γίνεται με τα δεδομένα που δίνουν στον μικροεπεξεργαστή διάφοροι αισθητήρες όπως πίεσης, θερμοκρασίας, ποιότητας καυσίμου, …. έτσι ώστε να βελτιστοποιείται η λειτουργία του κινητήρα και να αυξάνεται η απόδοση στο μέγιστο.Με αυτό τον τρόπο τα σύγχρονα αυτοκίνητα παρουσιάζουν πολύ μικρές καταναλώσεις και επιδόσεις που θα φάνταζαν ανήκουστες πριν από μερικά χρόνια.

Φωτεινοί σηματοδότες: Οι φωτεινοί σηματοδότες με τη σειρά τους ελέγχονται από μικροελεγκτές σε συνεργασία με διάφορους αισθητήρες που καταγράφουν την ροή των οχημάτων και αλλάζουν συνεχώς τους χρόνους διέλευσης έτσι ώστε να επιτυγχάνεται η μέγιστη κυκλοφορία. Μπορούμε πέρα από αυτά τα βασικά παραδείγματα να δώσουμε και άλλα πολλά, όπως στη **Βιομηχανία** με τον αυτόματο έλεγχο και τη ρομποτική, στην **Ιατρική** με την ψηφιακή απεικόνηση τα επεμβατικά λέιζερ τις γρήγορες εξετάσεις, στην **Αεροδιαστημική**, με τις δορυφορικές επικοινωνίες, στις **Αμυντικές βιομηχανίες** όπου η ταχύτητα και η τεχνολογία αιχμής βρίσκει δυστυχώς πάντα τις σημαντικότερες εφαρμογές, γιατί όποιος διαθέτει τα πιο σύγχρονα συστήματα σε όλα τα επίπεδα υπερέχει, στη **Βιοτεχνολογία** και σε πάρα πολλούς άλλους κλάδους και περιοχές του συγχρονου κόσμου που μας περιβάλλει όπου πλέον η ψηφιακή τεχνολογία έχει διεισδύσει και έχει προσφέρει άπειρα πλεονεκτήματα.

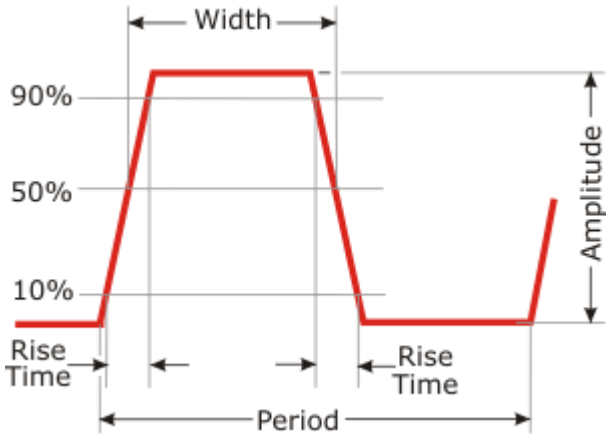
Ψηφιακά σήματα

Τα ψηφιακά σήματα που διαχειρίζονται τα λογικά κυκλώματα είναι τετραγωνικοί παλμοί της παρακάτω μορφής. Η μεταγωγή από την κατάσταση Η=1 στην κατάσταση L=0 ή αντίστροφα από την κατάσταση L=0 στην κατάσταση Η= 1 φαίνεται από το διάγραμμα να γίνεται ακαριαία. Η αλλαγή αυτή είναι μια κάθετη γραμμή στον άξονα Χ των χρόνων.

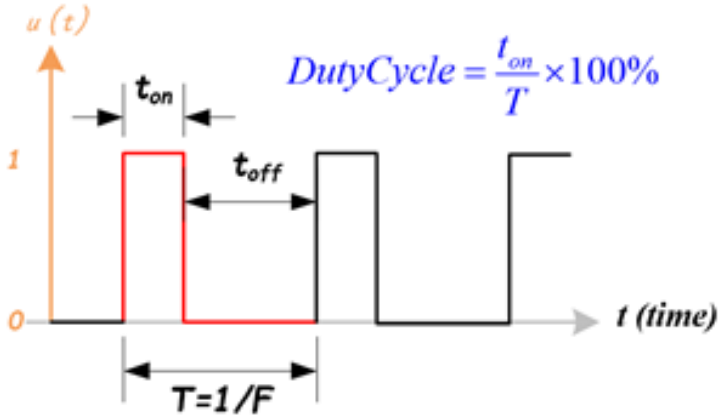


Στην πραγματικότητα η μεταγωγή δεν είναι ακαριαία λόγω της πεπερασμένης ταχύτητας των φορτίων στα κυκλώματα και στους αγωγούς σύνδεσης. Οι αναγκαίοι χρόνοι για να γίνουν οι μεταβάσεις Η=1 σε L=0 και L=0 σε Η=1 σε συνήθη κοινά ολοκληρωμένα είναι της τάξης των μερικών ns (10^{-9} sec). Ο χρόνος ανόδου του παλμού ονομάζεται rise-time, συμβολίζεται με t_r και είναι ο χρόνος που χρειάζεται ο παλμός για να πάει από 0,1Η στο 0,9Η. Αν δηλαδή το Η=1 αντιστοιχεί σε 5 Volts ο χρόνος ανόδου είναι αυτός που χρειάζεται για να γίνει η μετάβαση τάσης από 0,1Η=0,1x5V=0,5V σε 0,9Η=0,9x5V=4,5V. Ο ίδιος ορισμός ισχύει και ανάστροφα. Ο χρόνος καθόδου του παλμού ονομάζεται fall-time, συμβολίζεται με t_f και είναι ο χρόνος που χρειάζεται ο παλμός για να πάει από 0,9Η στο 0,1Η. Αν δηλαδή το Η=1 αντιστοιχεί σε 5 Volts ο χρόνος καθόδου είναι αυτός που χρειάζεται για να γίνει η μετάβαση τάσης από 0,9Η=0,9x5V=4,5V σε 0,1Η=0,1x5V=0,5V. Στο παρακάτω διάγραμμα βλέπουμε ένα πραγματικό διάγραμμα τετραγωνικού παλμού όπως επίσης το χρόνο ανόδου t_r και τον χρόνο καθόδου t_f σύμφωνα με τον ορισμό που δώσαμε. Εδώ πρέπει να πούμε ότι οι παραπάνω χρόνοι είναι εντελώς διαφορετικοί, διότι συναρτώνται με διαφορετικά φαινόμενα. Στο διάγραμμα

που παρουσιάζουμε παρακάτω τους έχουμε πάρει ίδιους και ίσους με $t_r = t_f$ για λόγους συντομίας.



Ένα άλλο χαρακτηριστικό στους τετραγωνικούς παλμούς που χρονίζουν τα λογικά κυκλώματα είναι ο κύκλος απόδοσης ή **Duty cycle** στα αγγλικά. Για να τον καταλάβουμε ας δούμε ένα χαρακτηριστικό παράδειγμα τετραγωνικού παλμού στο παρακάτω διάγραμμα.



Βλέπουμε καθαρά την περίοδο του τετραγωνικού σήματος $T = \frac{1}{F}$ όπου F είναι η συχνότητα. Βλέπουμε επίσης ότι ο χρόνος παραμονής του τετραγωνικού παλμού στην κατάσταση High=1 $\mathbf{t_{on}}$ και ο χρόνος παραμονής στην κατάσταση 0 $\mathbf{t_{off}}$. Έχουμε επίσης συνολικά ότι $T = t_{on} + t_{off}$.

Ο κύκλος απόδοσης ή Duty cycle είναι στην ουσία ο χρόνος παραμονής του παλμού στην κατάσταση 1 πάντα σε σχέση με τον χρόνο της περιόδου, δηλαδή:

$$\boldsymbol{Duty\_cycle} = \frac{\boldsymbol{t_{on}}}{\boldsymbol{T}} \boldsymbol{x} 100\%$$

Αν έχουμε ένα συμμετρικό σήμα δηλαδή $\boldsymbol{t_{on}} = \boldsymbol{t_{off}}$ τότε ο κύκλος απόδοσης θα είναι:

$$\boldsymbol{Duty\_cycle} = \frac{\boldsymbol{t_{on}}}{\boldsymbol{T}} \boldsymbol{x} 100\% = \frac{\frac{\boldsymbol{T}}{2}}{\boldsymbol{T}} \boldsymbol{x} 100\% = \frac{1}{2} 100\% = 50\%$$

Τα βασικά εργαλεία που θα χρησιμοποιήσουμε για να σχεδιάσουμε ένα ψηφιακό κύκλωμα ανεξάρτητα από την δυσκολία του δίνονται παρακάτω. Κατά τη διάρκεια αυτού του μαθήματος θα προσπαθήσουμε να αναλύσουμε όσο γίνεται πιο απλά το ένα μετά το άλλο ένα ένα τα κεφάλαια.

Ένα αριθμητικό σύστημα αποτελείται από ένα διατεταγμένο σύνολο συμβόλων τα οποία ονομάζουμε ψηφία και σχέσεις που ορίζουν τις βασικές πράξεις που όλοι γνωρίζουμε, δηλαδή την πρόσθεση, την αφαίρεση, τον πολλαπλασιασμό**,** και την διαίρεση.

Ένα αριθμητικό σύστημα αρίθμησης μπορεί να χρησιμοποιήσει οιανδήποτε αριθμό σαν βάση. Δηλαδή μπορούμε να χρησιμοποιήσουμε ένα οιανδήποτε αριθμό σαν βάση για να καθορίζουμε ένα διαφορετικό σύστημα αρίθμησης. Το απλούστερο σύστημα αρίθμησης που χρησιμοποιούμε ως γνωστό σήμερα είναι το δεκαδικό. Το σύστημα αυτό είναι το απλούστερο στη χρήση του. Εύκολα λέμε για παράδειγμα ότι **10^5=100.000** όχι όμως και **8^5=32.768**. Όταν μιλάμε για ένα σύστημα αρίθμησης με μία βάση τυχαία r σημαίνει ότι όλοι οι αριθμοί θα είναι ένας συνδυασμός των δυνάμεων αυτής της βάσης. Το **8^5=32.768** θα το χρειαζόμασταν αν είχαμε χρησιμοποιήθει σύμφωνα με τα παραπάνω **r=8** σαν βάση.

Η βάση του αριθμητικού συστήματος είναι ο συνολικός αριθμός των ψηφίων που χρησιμοποιεί. Αν έχουμε δηλαδή την βάση r χρειαζόμαστε r σύμβολα για να παραστήσουμε όλους του αριθμούς σε αυτή τη βάση. Στον παρακάτω πίνακα φαίνονται διάφορα συστήματα αρίθμησης με την αντίστοιχη βάση. Αν πάρουμε το δικό μας σύστημα αρίθμησης με βάση το 10, τότε θέλουμε 10 ψηφία για να αναπαράγουμε όλους τους αριθμούς. Τα ψηφία ή σύμβολα είναι τα γνωστά σε εμάς 0, 1, 2,3,4,5,6,7,8,9.

Αν πάρουμε τη βάση 3 τότε θέλουμε 3 ψηφία και χρησιμοποιούμε αποκλειστικά το 0, 1, 2, για να γράψουμε όλους τους αριθμούς, για τη βάση 2 θέλουμε 2 ψηφία το 0 και το 1 ενώ για τη βάση 16 θέλουμε 16 ψηφία. Σε αυτή την περίπτωση χρησιμοποιούμε τα γνωστά πάλι 0 ,1, 2, 3, 4, 5, 6, 7, 8, 9, και συμπληρώνουμε με τα πρώτα 6 κεφαλαία γράμματα του Λατινικού Αλφάβητου A, B, C, D, E, F.

Βάση 10	**Βάση 2**	**Βάση 3**	**Βάση 8**	**Βάση 16**
0	0	0	0	0
1	1	1	1	1
2	10	2	2	2
3	11	10	3	3
4	100	11	4	4
5	101	12	5	5
6	110	20	6	6
7	111	21	7	7
8	1000	22	10	8
9	1001	100	11	9
10	1010	101	12	A
11	1011	102	13	B
12	1100	110	14	C
13	1101	111	15	D
14	1110	112	16	E
15	1111	120	17	F
16	10000	121	20	10
17	10001	122	21	11
18	10010	200	22	12
19	10011	201	23	13

Συστήματα αρίθμησης με την αντίστοιχη βάση

Το σύστημα αριθμών που χρησιμοποιούμε ονομάζεται δεκαδικό. Αυτό σημαίνει ότι η βάση των αριθμών είναι το δέκα **(10)** και ότι όλοι οι αριθμοί του αριθμητικού συστήματος εκφράζονται σαν συνδυασμοί των δυνάμεων

του δέκα. Χρησιμοποιούνται επίσης τα δέκα γνωστά μας ψηφία 0,1,2,3,4,5,6,7,8,9. Για παράδειγμα ο δεκαδικός αριθμός 1990 γράφεται:

$$\mathbf{1990= 1\cdot 10^3 + 9\cdot 10^2+9\cdot 10^1+0\cdot 10^0}$$

Βλέπουμε ότι για ένα ακέραιο αριθμό, γραμμένο στη βάση 10 το τελευταίο ψηφίο είναι το βάρος του $\mathbf{10^0}$, το δεύτερο από το τέλος είναι το βάρος του $\mathbf{10^1}$, το τρίτο από το τέλος είναι το βάρος του $\mathbf{10^2}$ κ.λ.π.

επίσης ο αριθμός 195,33 γράφεται

$$\mathbf{195,33=1\cdot 10^2+9\cdot 10^1+5\cdot 10^0+3\cdot 10^{-1}+3\cdot 10^{-2}}$$

Το δυαδικό σύστημα όπως είδαμε και προηγουμένως είναι ένα σύστημα αρίθμησης με βάση το 2. Αυτό σημαίνει ότι τα δύο ψηφία 0 και 1 μπορούν να ορίσουν όλους τους αριθμούς εννοείται δυαδικούς. Για παράδειγμα ο αριθμός 11011 είναι σύμφωνα με τον ορισμό που δώσαμε είναι:

$$\mathbf{11011=1\cdot 2^4+1\cdot 2^3+0\cdot 2^2+1\cdot 2^1+1\cdot 2^0=(27)_{10}}$$

Γενικά ένας αριθμός σε μία βάση r εκφράζεται με συνδυασμούς των N ψηφίων από το 0 μέχρι το r-1 και των δυνάμεων του r όπως φαίνεται παρακάτω.

$$\mathbf{(ΑΡΙΘΜΟΣ)_r= a_n r^n+a_{n-1}r^{n-1}+.....a_0r^0+a_{-1}r^{-1}+.....}$$

Για να μπορούμε να διακρίνουμε τους αριθμούς στην βάση που είναι γραμμένοι τους τοποθετούμε μέσα σε παρένθεση και γράφουμε την αντίστοιχη βάση σαν δείκτη.

έτσι έχουμε $\mathbf{(111)_{2=}(7)_{10}}$

Σε περίπτωση που η βάση είναι μεγαλύτερη του δέκα χρειαζόμαστε περισσότερα σύμβολα από τα δέκα ψηφία του δεκαδικού συστήματος αρίθμησης. Τότε τα γράμματα του αλφαβήτου συμπληρώνουν τα υπόλοιπα ψηφία. Για παράδειγμα στο δεκαεξαδικό σύστημα (βάση το 16) τα γράμματα A,B,C,D,E,F χρησιμοποιούνται για τα ψηφία 10, 11, 12, 13, 14, και 15 αντίστοιχα.

Παράδειγμα δεκαεξαδικού αριθμού είναι.

$$\mathbf{AB8=10\cdot 16^2+11\cdot 16^1+8\cdot 16^0=(2744)_{10}}$$

Στο οκταδικό (βάση το 8) σύστημα αρίθμησης τα ψηφία που χρησιμοποιούνται είναι τα πρώτα 8 του δεκαδικού συστήματος αρίθμησης δηλαδή τα 0,1,2,3,4,5,6,7.

Έτσι $\mathbf{(111)_8=1\cdot8^2+1\cdot8^1+1\cdot8^0=(73)_{10}}$

ΜΕΤΑΤΡΟΠΗ ΒΑΣΗΣ

Οι μετατροπές βάσης είναι δύο τύπων και έχουν διαφορετική δυσκολία η κάθε μία. Στην πρώτη περίπτωση έχουμε ένα αριθμό σε μία βάση τυχαία και θέλουμε να τον εκφράσουμε στο δεκαδικό σύστημα. Σε αυτήν την περίπτωση αρκεί να γράψουμε τις αντίστοιχες δυνάμεις της βάσης και έχουμε το αποτέλεσμα.

ΠΑΡΑΔΕΙΓΜΑΤΑ

- $(153)_8=1\cdot8^2+5\cdot8^1+3\cdot8^0=(107)_{10}$
- $(101,1)_2=1\cdot2^2+0\cdot2^1+1\cdot2^0+1\cdot2^{-1}=(5,5)_{10}$
- $(A9)_{16}=10\cdot16^1+9\cdot16^0=(169)_{10}$
- $(1334)_5=1\cdot5^3+3\cdot5^2+3\cdot5^1+4\cdot5^0=(219)_{10}$

Αντίστροφα τώρα, έχουμε δηλαδή ένα αριθμό στο δεκαδικό και θέλουμε να τον εκφράσουμε σε μία τυχαία βάση r. Η μετατροπή γίνεται χωριστά στο ακέραιο και στο δεκαδικό μέρος όπως θα δούμε παρακάτω σε παραδείγματα. Μετατροπή του δεκαδικού αριθμού $\mathbf{(35,54)_{10}}$ σε δυαδικό. Για να γίνει αυτή η μετατροπή παίρνουμε πρώτα το ακέραιο μέρος και πραγματοποιούμε διαδοχικές διαιρέσεις με τη βάση του συστήματος που θέλουμε να κάνουμε τη μετατροπή όπως φαίνεται παρακάτω. Το υπόλοιπο θα σχηματίσει το ζητούμενο αριθμό. Έχουμε λοιπόν :

35/2=17	και υπόλοιπο 1
17/2=8	και υπόλοιπο 1
8/2=4	και υπόλοιπο 0
4/2=2	και υπόλοιπο 0
2/2=1	και υπόλοιπο 0
1/2=0	και υπόλοιπο 1

Άρα $(35)_{10}=(100011)_2$

Ας δούμε τώρα το κλασματικό μέρος. Για να γίνει η μετατροπή του δεκαδικού μέρους το πολλαπλασιάζουμε με τη βάση, κρατάμε το ακέραιο ψηφίο και συνεχίζουμε με το κλασματικό. Στο παρακάτω παράδειγμα αποτυπώνεται η διαδικασία. Έχουμε λοιπόν ότι:

	Ακέραιο μέρος	**Κλασματικό**
0.54x2=1.08	ακέραιο μέρος = 1	κλασματικό =0.08
0.08x2=0.16	ακέραιο μέρος = 0	κλασματικό= 0.16
0.16x2=0.32	ακέραιο μέρος = 0	κλασματικό= 0.32
0.32x2=0.64	ακέραιο μέρος = 0	κλασματικό= 0.64
0.64x2=1.28	ακέραιο μέρος = 1	κλασματικό= 0.28
0.28x2=0.56	ακέραιο μέρος = 0	κλασματικό= 0.56
0.56x2=1.12	ακέραιο μέρος = 1	κλασματικό= 0.12

Άρα έχουμε τελικά **$(35,54)_{10}=(100011,1000101)_2$**

Με τον ίδιο ακριβώς τρόπο γίνεται οιανδήποτε μετατροπή από το δεκαδικό σύστημα σε ένα αριθμητικό σύστημα με μία τυχαία βάση r.

Ένα BIT είδαμε ότι παίρνει 2 διαφορετικές τιμές. Ένα σύνολο από BIT φτιάχνουν ένα καταχωρητή. Αν έχουμε 2 BIT έχουμε ένα δίμπιτο καταχωρητή, τέσσερα BIT τετράμπιτο καταχωρητή, οκτώ BIT οκτάμπιτος καταχωρητής κ.λ.π. Οι διαφορετικές καταστάσεις που μπορεί να πάρει ένας καταχωρητής με **n** BIT είναι **2^n**. Σε αυτό τον καταχωρητή μπορούμε να καταχωρήσουμε τους αριθμούς μέτρησης από το 0 μέχρι το (**2^n** -1). Έτσι αν έχουμε 2 BIT έχουμε **$2^2=4$** καταστάσεις και αριθμούμε από το 0 μέχρι το 3. Αν έχουμε 4 BIT έχουμε **$2^4=16$** καταστάσεις και αριθμούμε από το 0 μέχρι το 15, αν έχουμε 6 BIT έχουμε **$2^6=64$** καταστάσεις και αριθμούμε από το 0 μέχρι το 63 κ.λ.π.

Μετατροπές από βάση 2 στη βάση 4, 8 και 16. Σε αυτή την περίπτωση κόβουμε τον αριθμό σε δυάδες, σε τριάδες και σε τετράδες αντίστοιχα ξεκινώντας πάντα από το τέλος. Πηγαίνοντας αριστερά του αριθμού αν τα

ψηφία δεν αρκούν συμπληρώνουμε με μηδενικά. Ας πάρουμε ένα τυχαίο αριθμό γραμμένο στη βάση 2. **$(110101011101)_2$.**

Αν τον κόψουμε σε δυάδες θα έχουμε :

11-01- 01- 01- 11- 01

3 1 1 1 3 1 επειδή 11=3 και 01=1 και 10=2 και 00=0

Άρα $(110101011101)_2=(311131)_4$

Η μετατροπή στη βάση 8 θα γίνει με αντίστοιχο τρόπο.

110-101-011-101

6 5 3 5 επειδή 110=6 101=5 και 011=3 άρα

$(110101011101)_2=(6535)_8$

Η μετατροπή στη βάση 16 θα γίνει με αντίστοιχο τρόπο.

1101 0101 1101

D 5 D όμως 1101=13=D και 0101=5 και κατά συνέπεια ο αριθμός $(110101011101)_2=(D5D)_{16}$.

Αριθμητικές Πράξεις

Οι αριθμητικές πράξεις οι γνωστές σε εμάς δηλαδή η πρόσθεση, η αφαίρεση ο πολλαπλασιασμός και η διαίρεση εκτελούνται ακριβώς όπως γνωρίζουμε από το δεκαδικό σύστημα όπως θα δούμε με τα παρακάτω παραδείγματα.

Πρόσθεση 111101

\+ 01101

1001010

Ξεκινώντας από το τέλος προσθέτοντας τα τελευταία ψηφία βλέπουμε ότι 1+1=10 στο δυαδικό, έτσι γράφουμε το **0** και κρατάμε το 1, στην συνέχεια έχουμε 0+0+1 το κρατούμενο=1 και γράφουμε το **1** χωρίς κρατούμενο. Στη συνέχεια έχουμε 1+1=10, γράφουμε το **0** και κρατάμε το 1, οπότε με κρατούμενο 1 έχουμε στη συνέχεια 1+1+1=11 τότε γράφουμε το **1** και κρατάμε το 1, στη συνέχεια έχουμε 1 το κρατούμενο 1+1+0=10, γράφουμε το **0** και κρατάμε το 1, και τέλος 1+1=**10** το γράφουμε και

τελειώνει η πρόσθεση. Υπενθυμίζουμε ότι στο δυαδικό σύστημα αρίθμησης έχουμε:

- 0+0=0
- 0+1=1
- 1+0=1
- 1+1=10

Αφαίρεση 11110

- 1101

= 10001

Με τον ίδιο τρόπο ξεκινώντας από το τέλος λειτουργούμε όπως μια αφαίρεση στο δεκαδικό με τη διαφορά ότι τώρα η βάση είναι το 2. Ας δούμε λοιπόν όπως και προηγουμένως και ξεκινώντας από το τέλος. Λέμε 1 από 0 δεν βγαίνει και δανειζόμαστε μία δυάδα δηλαδή θα πούμε 1 από 10, μας δίνει **1** και 1 υπόλοιπο. Στο επόμενο βήμα θα έχουμε το κρατούμενο στο οποίο προσθέτουμε το 0, δηλαδή 1+0=1 από 1 μας δίνει **0**. Συνεχίζοντας βλέπουμε ότι 1 από 1 κάνει **0**, 1 από 1 κάνει **0** και 0 από 1 κάνει **1**. Το αποτέλεσμα είναι το 10001.

Πολλαπλασιασμός 10101011

x 111

=10101011

10101011

+10101011

= 10010101101

Ο πολλαπλασιασμός είναι και αυτός ανάλογος με αυτόν που κάνουμε στο δεκαδικό όπως φαίνεται στο παραπάνω παράδειγμα. Παρόμοια διαδικασία με αυτή που όλοι γνωρίζουμε γίνεται και στη διαίρεση όπως φαίνεται στον παρακάτω πίνακα. Υπάρχει κανονικά διαιρετέος, διαιρέτης, υπόλοιπο και πηλίκον.

1111	**11**
11	101
-11	
0011	
- 0011	
00	

Το ίδιο και η διαίρεση. Γίνεται ακριβώς με τον ίδιο τρόπο όπως στο δεκαδικό. Έστω ότι θέλουμε να διαιρέσουμε το 1111 με το 11. Στον προηγούμενο πίνακα βλέπουμε όλα τα βήματα που θα κάναμε για μία διαίρεση στο δεκαδικό. Βλέπουμε ότι η διαίρεση 1111:11=101.

Συμπληρώματα

Τα συμπληρώματα στα ψηφιακά κυκλώματα είναι πάρα πολύ σημαντικά διότι όπως θα δούμε παρακάτω με τη χρήση τους καταργείται στην ουσία η πράξη της αφαίρεσης και τα ηλεκτρονικά κυκλώματα στην ουσία εκτελούν πρόσθεση αντί για αφαίρεση. Για να δούμε πως λειτουργεί το παραπάνω διαδικασία ας δούμε πρώτα κάποιους κανόνες.

Κανόνας: για ένα δεδομένο θετικό αριθμό Ν γραμμένο σε ένα αριθμητικό σύστημα με τη βάση r με n ακέραια ψηφία το συμπλήρωμα του ως προς r είναι $\mathbf{r^n-N}$ όταν το Ν είναι διαφορετικό του μηδενός και 0 όταν το Ν είναι 0. Επίσης ορίζουμε το συμπλήρωμα ως προς τη (βάση **r– 1**) για τον ίδιο παραπάνω αριθμό Ν με n ψηφία $\mathbf{r^n-N-1}$ όταν το Ν είναι διαφορετικό του μηδενός και 0 όταν το Ν είναι 0.

Ας πάρουμε για παράδειγμα ένα πενταψήφιο αριθμό τον 45782. Ο παραπάνω αριθμός έχει 5 ψηφία άρα n=5. Εφαρμόζοντας τον ορισμό θα έχουμε ότι το συμπλήρωμα αυτού του αριθμού ως προς τη βάση 10 είναι 10^5-45782 δηλαδή 100000-45782, ενώ το συμπλήρωμα ως τη βάση 10 -1 δηλαδή ως προς 9 είναι 10^5-45782-1=(10^5-1)-45782=99999-45782. Βλέπουμε ότι μπορούμε να το βρούμε πολύ εύκολα αρκεί να αφαιρέσουμε όλα τα ψηφία του αριθμού από 9. Τότε θα έχουμε ξεκινώντας από το τέλος 2 από 9 ίσον **7**, 8 από 9 ίσον **1**, 7 από 9 ίσον **2**, 5 από 9 ίσον **4**, και 4 από 9 ίσον **5**. Δηλαδή το συμπλήρωμα ως προς (10-1)=9 του αριθμού 45782

είναι το **54217** και αν προσθέσουμε 1 μονάδα θα πάρουμε το συμπλήρωμα ως προς 10, δηλαδή **54128**.

Άρα $\mathbf{\Sigma(45782)_{10}=54128}$ και $\mathbf{\Sigma(45782)_{9}=54127}$

Παράδειγμα: Έστω ο αριθμός 1992. Όπως βλέπουμε έχει τέσσερα ακέραια ψηφία άρα το συμπλήρωμα του ως προς τη βάση 10 που είναι γραμμένος είναι 10^4-1992=8008.

Στους δυαδικούς αριθμούς έχουμε το συμπλήρωμα ενός δυαδικού αριθμού ως προς τη βάση 2, δηλαδή ως προς 2 ή ως προς τη βάση 2 μείον 1, δηλαδή ως προς 1. Αν χρησιμοποιήσουμε τον ίδιο ορισμό για ένα δυαδικό με n ψηφία θα έχουμε ότι το συμπλήρωμα

ΔΥΑΔΙΚΟΙ ΚΩΔΙΚΕΣ

Στα ψηφιακά συστήματα κάθε στοιχείο για επεξεργασία είναι σε κωδικοποιημένη μορφή. Έτσι κωδικοποιούνται τα μαθηματικά σύμβολα για να μπορέσουμε να κωδικοποιήσουμε τις πράξεις, τα γράμματα του αλφαβήτου για τις μεταβλητές, οι αριθμοί κ.λ.π. Η κωδικοποίηση γίνεται με ένα συγκεκριμένο αριθμό δυαδικών κυττάρων συνδυάζοντας τα μεταξύ τους. Για να κωδικοποιήσουμε με αυτό τον τρόπο 2^N διαφορετικά στοιχεία χρειαζόμαστε N δυαδικά κύτταρα. Έτσι με τα N δυαδικά κύτταρα μπορώ να φτιάξω 2^N συνδυασμούς με 0 και 1 έτσι ώστε κάθε στοιχείο του αρχικού συνόλου να αντιστοιχεί σε ένα και μοναδικό συνδυασμό.

Παράδειγμα: Έστω ότι έχουμε ένα σύνολο τεσσάρων στοιχείων όπως φαίνεται παρακάτω **a,b,c,d.** Τα τέσσερα αυτά στοιχεία θα μπορούσαν να αντιστοιχούν σε τέσσερα κουτιά με διαφορετικό χρώμα ή μέγεθος για παράδειγμα. Για να τα ταυτοποιήσω χρειάζεται να φτιάξω ένα δυαδικό κώδικα έτσι ώστε να αντιστοιχίσω τα τέσσερα αυτά στοιχεία. Δηλαδή ένα συνδυασμό από αριθμούς δυαδικούς 0 και 1 όπου το κάθε κουτί ή χρώμα θα αντιστοιχεί σε ένα και μόνο ένα συνδυασμό. Για να συμβεί αυτό χρειάζομαι τέσσερεις διαφορετικούς συνδυασμούς. Όμως έχω ότι $\mathbf{4=2^2}$. Άρα χρειάζομαι δύο δυαδικά κύτταρα και θα έχω την παραπάνω αντιστοιχία αν πρόκειται για κουτιά με αυξανόμενες διαστάσεις από το a στο d δηλαδή διαστάσεις **a>b>c>d**

- Το a θα αντιστοιχεί στον συνδυασμό [00]

- Το b θα αντιστοιχεί στον συνδυασμό [01]
- Το c θα αντιστοιχεί στον συνδυασμό [10]
- Το d θα αντιστοιχεί στον συνδυασμό [11]

Παράδειγμα: Τα δέκα ψηφία του δεκαδικού συστήματος αρίθμησης 0,1,2,3,4,5,6,7,8,9 για να κωδικοποιηθούν χρειάζονται τέσσερα δυαδικά κύτταρα τα οποία μπορούν να δώσουν 16 δυνατούς συνδυασμούς. Από αυτούς τους δεκάξι συνδυασμούς μόνο οι δέκα χρειάζονται και οι υπόλοιποι θα μείνουν αχρησιμοποίητοι και θα ονομάζονται όπως θα δούμε παρακάτω στις λογικές συναρτήσεις αδιάφοροι όροι.

Ο ΚΩΔΙΚΑΣ BCD

Ο κώδικας BCD "BINARY CODED DECIMAL" είναι άμεση αντιστοιχία του δεκαδικού ψηφίου με το δυαδικό του ισοδύναμο. Ο κώδικας BCD αποτελείται από τέσσερα bits. Κάθε bit έχει ένα βάρος το οποίο αντιστοιχεί σε μία δύναμη του 2. Τι λιγότερο σημαντικό ψηφίο (LSB) είναι το δεξιό το οποίο αντιστοιχεί στο $\mathbf{2^0}$ και τέλος το σημαντικό bit (MSB) είναι το αριστερό το οποίο αντιστοιχεί στο $\mathbf{2^3}$. Το βάρος του κώδικα BCD ορίζεται σαν **(8421)** πράγμα που αντιστοιχεί στον ορισμό $\mathbf{2^3 2^2 2^1 2^0}$.

Παράδειγμα: Κωδικοποιήστε τον αριθμό N=(1982) στον κώδικα BCD. Κωδικοποιούμε ένα ένα τα ψηφία του δεκαδικού αριθμού και έχουμε:

Το 1 στο BCD = $\mathbf{0\cdot 2^3+0\cdot 2^2+0\cdot 2^1+1\cdot 2^0=0001}$

Το 9 στο BCD = $\mathbf{1\cdot 2^3+0\cdot 2^2+0\cdot 2^1+1\cdot 2^0=1001}$

Το 8 στο BCD = $\mathbf{1\cdot 2^3+0\cdot 2^2+0\cdot 2^1+0\cdot 2^0=1000}$

Το 2 στο BCD = $\mathbf{0\cdot 2^3+0\cdot 2^2+1\cdot 2^1+0\cdot 2^0=0010}$

Έτσι ο αριθμός N=(1982)= [0001100110000010] στον κώδικα BCD. Η κωδικοποίηση κάθε δεκαδικού ψηφίου στο BCD φαίνεται παρακάτω:

Δεκαδικός αριθμός	**Κώδικας BCD**
0	0000
1	0001
2	0010

3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

Ο ΚΩΔΙΚΑΣ 4221

Ο κώδικας 4221 είναι ένας άλλος κώδικας με αντίστοιχα βάρη των ψηφίων 4221. Έτσι ο αριθμός 9 γράφεται 1001 στο BCD και με [1111] στον κώδικα 4221. $\mathbf{9=1\cdot2^2+1\cdot2^1+1\cdot2^1+1\cdot2^0=4+2+2+1=9}$

Παράδειγμα:

Γράψετε τον αριθμό Ν=(1982) στον κώδικα 4221.

ΤΟ 1 στον κώδικα 4221 γράφεται $\mathbf{0\cdot2^2+0\cdot2^1+0\cdot2^1+1\cdot2^0=0001}$

ΤΟ 9 στον κώδικα 4221 γράφεται $\mathbf{1\cdot2^2+1\cdot2^1+1\cdot2^1+1\cdot2^0=1111}$

ΤΟ 8 στον κώδικα 4221 γράφεται $\mathbf{1\cdot2^2+1\cdot2^1+1\cdot2^1+0\cdot2^0=1110}$

ΤΟ 2 στον κώδικα 4221 γράφεται $\mathbf{0\cdot2^2+0\cdot2^1+1\cdot2^1+0\cdot2^0=0010}$

Άρα το 1982 είναι =[0001111111100010] στον κώδικα 4221. Ο κώδικας 4221 είναι αυτοσυμπληρούμενος και γι' αυτό το λόγο χρησιμοποιείται από πολλά ψηφιακά συστήματα για πράξεις. Το αυτοσυμπληρούμενος σημαίνει ότι για να βρούμε το συμπλήρωμα ενός κωδικοποιημένου αριθμού στον κώδικα 4221 αρκεί να αλλάξουμε τα 1 σε 0 και τα 0 σε 1.

Δεκαδικός αριθμός	**Κώδικας 4221**
0	0000
1	0001

2	0010
3	0011
4	1000
5	0111
6	1100
7	1101
8	1110
9	1111

Παράδειγμα: Βρείτε το συμπλήρωμα ως προς 9 του αριθμού N=(9750)10 και συγκρίνετε το αποτέλεσμα χρησιμοποιώντας τον κώδικα 4221.

Το συμπλήρωμα του [9750] ως προς 9 είναι βάση του ορισμού 10^4-N-1= (0249). Όμως ο αριθμός 9750 γράφεται στον κώδικα 4221 σαν N=[1111110101110000]. Αν αλλάξουμε τα 1 σε 0 και τα 0 σε 1 θα έχουμε N=[0000001010001111]=0249. Βλέπουμε ότι έχουμε το ίδιο αποτέλεσμα και με τους δύο τρόπους.

Η κωδικοποίηση των δέκα δεκαδικών ψηφίων στον κώδικα 4221 είναι αυτή που είδαμε παραπάνω.

Ο ΚΩΔΙΚΑΣ GRAY

Ο κώδικας GRAY είναι ένας κυκλικός κώδικας. Αυτό σημαίνει ότι μόνο ένα ψηφίο (BIT) αλλάζει στον κώδικα όταν προχωράει από τον ένα κωδικοποιημένο αριθμό στον αμέσως επόμενο του. Η κωδικοποίηση γίνεται όπως στον παρακάτω στο πίνακα.

Δεκαδικός αριθμός	Δυαδικός αριθμός	Κώδικας Gray
0	0000	0000
1	0001	0001
2	0010	0011

3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

Ο ΚΩΔΙΚΑΣ EXCES 3

Ο κώδικας EXCESS-3 είναι ένας άλλος κώδικας και παράγεται προσθέτοντας τον αριθμό 3 στον κώδικα BCD.

EXCESS-3 = BCD + 3

Η κωδικοποίηση του EXCESS-3 φαίνεται παρακάτω όπως επίσης ο κώδικας BCD και ο αντίστοιχος δεκαδικός.

Δεκαδικός αριθμός	**Κώδικας BCD**	**Κώδικας EXCESS-3**
0	0000	0011
1	0001	0100
2	0010	0101
3	0011	0110
4	0100	0111

5	0101	1000
6	0110	1001
7	0111	1010
8	1000	1011
9	1001	1100

Βλέπουμε καθαρά στον παραπάνω πίνακα ότι για την κωδικοποίηση EXCESS-3 αρκεί να προσθέσουμε σε κάθε συνδυασμό BCD το 0011=3. Αν έχουμε να μετατρέψουμε ένα δεκαδικό αριθμό ο οποίος έχει γραφεί στον κώδικα BCD στον Excess-3 εκτελούμε την παρακάτω διαδικασία.

Π.χ. ο αριθμός 12 στο BCD είναι 0001 0010. Προσθέτουμε το 3 σε κάθε ψηφίο του δεκαδικού, δηλαδή 1+3=4 και 2+3=5 και στη συνέχεια γράφουμε το αποτέλεσμα 45 στο BCD, δηλαδή 0100 0101 που είναι στην πραγματικότητα ο αρχικός δεκαδικός αριθμός 12 γραμμένος στον κώδικα Excess-3. Μπορούμε επίσης να γράψουμε απ' ευθείας την παρακάτω αντιστοιχία $(00010010)_{BCD}=(01000101)_{Excess-3}$ αρκεί να θυμόμαστε ότι και οι δύο κώδικες είναι τετράμπιτοι και κωδικοποιούν τα δέκα ψηφία του δεκαδικού συστήματος αρίθμησης.

ΑΛΦΑΡΙΘΜΗΤΙΚΟΙ ΚΩΔΙΚΕΣ

Οι αλφαριθμητικοί κώδικες χρησιμοποιούνται όταν το ψηφιακό κύκλωμα χρησιμοποιεί εκτός των αριθμών σύμβολα και γράμματα. Λόγω του μεγάλου αριθμού συμβόλων οι αλφαριθμητικοί κώδικες είναι μεγαλύτεροι και μπορεί να έχουν από 6 μέχρι 8 bits. Στον παρακάτω πίνακα μπορούμε να δούμε δύο βασικούς κώδικες τον ASCI II και τον EBCDIC. Οι δύο αυτοί κώδικες κωδικοποιούν τα γράμματα του Λατινικού και του Ελληνικού αλφαβήτου τους αριθμούς και ένα μεγάλο αριθμό από βασικά σύμβολα που χρησιμοποιούμε πολύ συχνά. Ο κώδικας **EBCDIC** είναι ένας κώδικας 8 BIT ενώ ο κώδικας **ASCII** είναι ένας κώδικας 7 BIT. Το P είναι το BIT ισοτιμίας το οποίο μπορεί να είναι 0 ή 1 ανάλογα αν έχουμε περιττή ή άρτια ισοτιμία. Το BIT ισοτιμίας θα το δούμε και παρακάτω, είναι ένα BIT που προσθέτουμε σε ένα δυαδικό μήνυμα κατά την μεταφορά του από ένα κύκλωμα επεξεργασίας σε ένα άλλο και το οποίο μας επιτρέπει να ελέγξουμε αν η μεταφορά έγινε σωστά, δηλαδή αν

το ψηφιακό μήνυμα έφθασε στον προορισμό του σωστό. Το BIT ισοτιμίας μπορεί όπως είναι λογικό να πάρει δυο διαφορετικές τιμές, 0 ή 1 ανάλογα με την ισοτιμία που έχουμε καθορίσει από την αρχή. Αν είναι άρτια η ισοτιμία θα πρέπει ο συνολικός αριθμός των 1 να είναι άρτιος αριθμός ενώ αν είναι περιττή περιττός.

ΣΥΜΒΟΛΟ	**EBCDIC**	**ASCII**
A	11000001	P1000001
B	11000010	P1000010
C	11000011	P1000011
D	11000100	P1000100
E	11000101	P1000101
F	11000110	P1000110
G	11000111	P1001000
H	11001000	P1001000
I	11001001	P1001001
J	11010001	P1001011
K	11010010	P1001011
L	11010011	P1001100
M	11010100	P1001101
N	11010101	P1001110
O	11010110	P1001111
P	11010111	P1010000
Q	11011000	P1010001
R	11011001	P1010010
S	11100010	P1010011
T	11100011	P1010100
U	11100100	P1010101
V	11100110	P1010110
W	11100110	P1010111
X	11100111	P1011000
Y	11101000	P1011001
Z	11101001	P1011010
0	11110000	P0110000
1	11110001	P0110001
2	11110010	P0110010
3	11110011	P0110011

4	11110100	P0110100
5	11110101	P0110101
6	11110110	P0110110
7	11110111	P0110111
8	11111000	P0111000
9	11111001	P0111001
.	01001011	P0101110
+	01001110	P0101011
$	01011011	P0100100
*	01011100	P0101010
(	01001101	P0101000
)	01011101	P0101010
=	01111110	P0111101

Ανίχνευση σφαλμάτων

Κατά τη λειτουργία ενός ψηφιακού συστήματος υπάρχει μεταφορά πληροφορίας ανάμεσα στα επιμέρους κομμάτια που το αποτελούν. Η διαδικασία αυτή της μεταφοράς συνοδεύεται συχνά από σφάλματα. Γι' αυτό το λόγο είναι αναγκαία η ανεύρεση κάποιου τρόπου ελέγχου του μεταδιδόμενου κωδικοποιημένου σήματος.

Η απλούστερη μέθοδος ελέγχου που υπάρχει είναι ο έλεγχος της ισοτιμίας. Για να είναι εύκολη η αναγνώριση λάθους έχει προστεθεί στους κωδικοποιημένους χαρακτήρες επί πλέον ένα bit (parity bit). Πριν από την αποστολή του μηνύματος ένα κύκλωμα το οποίο ονομάζεται γεννήτρια ισοτιμίας παράγει το BIT ισοτιμίας το οποίο προστίθεται στο μήνυμα. Κατά την άφιξη στον προορισμό του ένα άλλο κύκλωμα ελέγχει την ακεραιότητα του μηνύματος και αν όλα είναι εντάξει αφαιρεί το BIT και αν όχι η διαδικασία αποστολής επαναλαμβάνεται. Επαναλαμβάνουμε ότι στο μεταφερόμενο δυαδικό σήμα προστίθεται ένα ψηφίο 0 η 1 ανάλογα με τον τύπο της ισοτιμίας που έχουμε έτσι ώστε ο συνολικός αριθμός των 1 να είναι άρτιος στην άρτια ισοτιμία και περιττός στην περιττή ισοτιμία. Στον παρακάτω πίνακα δίνονται οι αριθμοί από 0 μέχρι 9 με τα bit των δύο ισοτιμιών.

Δεκαδικός	**Δυαδικός**	**Άρτια Ισοτιμία**	**Περιττή Ισοτιμία**

0	0000	00000	10000
1	0001	10001	00001
2	0010	10010	00010
3	0011	00011	10011
4	0100	10100	01000
5	0101	00101	10101
6	0110	00110	10110
7	0111	10111	00111
8	1000	11000	01000
9	1001	01001	11001

Παράσταση δυαδικών ψηφίων με bit ισοτιμίας.

Έστω ότι θέλουμε να μεταφέρουμε το C στον κώδικα ASCII το οποίο είναι 1000011. Στην άρτια ισοτιμία θα προσθέσουμε το 1 μπροστά από το 11000011 για να γίνει ο συνολικός αριθμός των άσσων 1 άρτιος. Ενώ στην περίπτωση της περιττής ισοτιμίας το προστιθέμενο ψηφίο θα ήταν το 0 για να διατηρηθεί ο συνολικός αριθμός των άσσων περιττός

Ο ρόλος του ψηφίου ισοτιμίας είναι η ανίχνευση σφαλμάτων που μπορεί να εμφανιστούν κατά την μεταφορά σημάτων. Έστω ότι έχουμε να μεταφέρουμε το Α στον κώδικα ASCII το οποίο είναι 1000001. Στην περίπτωση περιττής ισοτιμίας το bit που προσθέτουμε είναι το ένα και το 1 και το Α γίνεται 11000001. Το παραπάνω BIT το οποίο παίρνει τη θέση MSB δηλαδή τη αριστερή θέση στον αριθμό παράγεται από ένα κύκλωμα το οποίο ονομάζουμε γεννήτρια ισοτιμίας. Όπως βλέπουμε η μέθοδος της ισοτιμίας είναι ο απλούστερος τρόπος να ελέγξουμε ένα σήμα και αναφέρεται στην περίπτωση που έχουμε σφάλμα, ή αλλαγή ενός μόνο BIT ή ενός μόνο ψηφίου κατά τη μεταφορά μιας πληροφορίας. Η μέθοδος αυτή δεν καλύπτει την περίπτωση που έχουμε αλλαγή δύο ψηφίων συγχρόνως. Σε αυτή τη περίπτωση η ισοτιμία δεν αλλάζει και με αυτό τον τρόπο ο δέκτης λαμβάνει το σήμα θεωρώντας το σωστό ενώ στην πραγματικότητα δεν είναι. Ένας πιο περίπλοκος τρόπος για την ανίχνευση αλλά και τη διόρθωση σφαλμάτων είναι δυνατόν αν χρησιμοποιήσουμε τον κώδικα του HAMMING.

Πίνακας με τον κώδικα **ASCII** ο οποίος όπως είδαμε είναι ένας κώδικας 7 BIT. Βλέπουμε στον προηγούμενο πίνακα να αναφέρεται πάνω από τις

κολώνες στη μέση το στοιχείο $\mathbf{b_7b_6b_5}$ και αριστερά την κορυφή της 1ης κολώνας $\mathbf{b_4b_3b_2b_1}$. Ένα στοιχείο του κώδικα **ASCII** κωδικοποιείται με τα BIT όπως $\mathbf{b_7b_6b_5b_4b_3b_2b_1}$. Το πιο σημαντικό ψηφίο MSB είναι το $\mathbf{b_7}$ και το λιγότερο σημαντικό LSB το $\mathbf{b_1}$. Με βάση τα παραπάνω ας δούμε λοιπόν μερικούς χαρακτήρες και σύμβολα πως κωδικοποιούνται. Το NULL σαν 0000000, το σύμβολο ! σαν 0100001, το σύμβολο # σαν 0100011, το σύμβολο (σαν 0101000, το σύμβολο = σαν 0111101, το γράμμα B κεφαλαίο σαν 1000010, το b μικρό σαν 1100010 κ.λ.π.

	$b_7b_6b_5$							
$b_4b_3b_2b_1$	000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	‘	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	“	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	‘	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L		l	|
1101	CR	GS	−	=	M	]	m	}
1110	SO	RS	.	>	N	∧	n	~
1111	SI	US	/	?	O	−	o	DEL

Control Characters

NUL	Null	DLE	Data-link escape
SOH	Start of heading	DC1	Device control 1
STX	Start of text	DC2	Device control 2
ETX	End of text	DC3	Device control 3
EOT	End of transmission	DC4	Device control 4
ENQ	Enquiry	NAK	Negative acknowledge
ACK	Acknowledge	SYN	Synchronous idle
BEL	Bell	ETB	End-of-transmission block
BS	Backspace	CAN	Cancel
HT	Horizontal tab	EM	End of medium
LF	Line feed	SUB	Substitute
VT	Vertical tab	ESC	Escape
FF	Form feed	FS	File separator
CR	Carriage return	GS	Group separator
SO	Shift out	RS	Record separator
SI	Shift in	US	Unit separator
SP	Space	DEL	Delete

Αριθμητικές πράξεις και κυκλώματα

Είδαμε προηγουμένως πως εκτελούνται οι στοιχειώδεις πράξεις στο δυαδικό σύστημα αρίθμησης. Στην συνέχεια θα εμβαθύνουμε λίγο παραπάνω διαδικασίες και τεχνικές που χρησιμοποιούνται από τα ψηφιακά συστήματα για να εκτελέσουν αριθμητικές πράξεις.

Ας πάρουμε για παράδειγμα τους δύο παρακάτω οκτάμπιτους αριθμούς $A_7A_6A_5A_4A_3A_2A_1A_0$ και $B_7B_6B_5B_4B_3B_2B_1B_0$. Ονομάζουμε MSB (More

Significant Bit) περισσότερο σημαντικό ψηφίο το αριστερότερο ψηφίο που είναι το A_7 και το B_7 για τους αριθμούς Α και Β αντίστοιχα. Ονομάζουμε επίσης LSB (Less Significant Bit) λιγότερο σημαντικό ψηφίο το δεξιότερο ψηφίο που είναι το A_0 και το B_0 για τους αριθμούς Α και Β αντίστοιχα. Για να τους προσθέσουμε ξεκινούμε από το LSB και πηγαίνουμε προς το MSB. Αν οι αριθμοί είναι Α=01010111 και Β=00110101 θα έχουμε

$$\begin{array}{r} 01010111 \\ +00110101 \\ \hline 10001100 \end{array}$$

Αν γράψουμε στη βάση 16 τους ίδιους αριθμούς και κάνουμε τις πράξεις θα έχουμε:

$$\begin{array}{r} 57 \\ +35 \\ \hline 8C \end{array}$$

Συνήθως αναφέρουμε ότι οι παραπάνω αριθμοί είναι γραμμένοι στο δεκαεξαδικό προσθέτοντας το γράμμα Η στο τέλος κάθε ψηφίου όπως φαίνεται παρακάτω.

$$\begin{array}{r} 57H \\ +35H \\ \hline 8CH \end{array}$$

Ας υποθέσουμε ότι έχουμε να προσθέσουμε τους παρακάτω δεκαεξάμπιτους αριθμούς Α=0000 1111 1010 1100 και τον επίσης δεκαεξάμπιτο Β=0011 1000 0111 1111. Θα έχουμε:

$$\begin{array}{r} 0000\ 1111\ 1010\ 1100 \\ +0011\ 1000\ 0111\ 1111 \\ \hline 0100\ 1000\ 0010\ 1011 \end{array}$$

Αν γράψουμε τους αριθμούς στη βάση 16 θα έχουμε ότι:

0FAC

+387F

482B

Ή με το Η στο τέλος για να διακρίνουμε το δεκαεξαδικό.

0FACH

+387FH

482BH

Βλέπουμε προηγουμένως στους δεκαεξαδικούς αριθμούς ότι ξεκινώντας από την πρόσθεση των δύο LSB έχουμε να προσθέσουμε $(F+C)_{16}=(15+12)_{10}=(27)_{10}$. Ποιος είναι όμως ο δεκαδικός αριθμός 27 στο σύστημα αρίθμησης με βάση το 16. Για να απαντήσουμε πρέπει να δούμε τον παρακάτω πίνακα.

Βάση 10	**Βάση 2**	**Βάση 8**	**Βάση 16**
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	12	9
10	1010	13	A
11	1011	14	B
12	1100	15	C
13	1101	16	D
14	1110	17	E
15	1111	20	F
16	10000	21	10
17	10001	22	11
18	10010	23	12
19	10011	24	13

20	10100	25	14
21	10101	26	15
22	10110	27	16
23	10111	30	17
24	11000	31	18
25	11001	32	19
26	11010	33	1A
27	11011	34	1B
28	11100	35	1C
29	11101	36	1D
30	11110	37	1E
31	11111	40	1F

Βλέπουμε ότι το 27 στη βάση 10 είναι 1Β στη βάση 16, γράφουμε κατά συνέπεια το Β και κρατάμε το 1, οπότε στη συνέχεια θα έχουμε 1 το κρατούμενο $(1+7+A)_{16}=(18)_{10}=(12)_{16}$ οπότε γράφουμε το 2 και κρατάμε το 1 στη συνέχεια θα έχουμε $(1+8+F)_{16}=(1+8+15)_{10}=(24)_{10}=(18)_{16}$ οπότε γράφουμε το 8 και κρατάμε το 1 και τέλος $(1+3+0)_{16}=(4)_{16}$.

<u>Δυαδική αφαίρεση</u>

Οι τέσσερεις βασικές περιπτώσεις δυαδικής αφαίρεσης είναι:

0-0=0
1-0=1
1-1=0
10-1=1

Αν τώρα προσπαθήσουμε να αφαιρέσουμε 2 οκτάμπιτους αριθμούς θα έχουμε:

1100 1000

\- 0111 1101

0100 1011

Και σε δεκαεξαδικό συμβολισμό θα έχουμε:

C8H

\- 7DH

4DH

Ας εξηγήσουμε τη διαδικασία. Έχουμε D από 8 δεν βγαίνει και δανειζόμαστε μια δεκαεξάδα, δηλαδή θα αφαιρέσουμε D από 18 στο δεκαεξαδικό και οι δύο αριθμοί $(18)_{16}$-$(D)_{16}$=$(24)_{10}$-$(13)_{10}$=$(11)_{10}$=$(B)_{16}$. Οπότε γράφουμε το Β, στη συνέχεια έχουμε 1 το κρατούμενο 1+ 7 =8 από C ίσον 4.

Αν οι αριθμοί είναι θετικοί ή αρνητικοί αγνοούμε τα πρόσημα και μπορούμε να επικεντρωθούμε αποκλειστικά στα νούμερα. Σε αυτή την περίπτωση οι οκτάμπιτοι αριθμοί μέγιστοι και ελάχιστοι είναι:

	Βάση 2	**Βάση 16**
Ελάχιστος αριθμός	0000 0000	00H
Μέγιστος αριθμός	1111 1111	FFH

Οι παραπάνω αριθμοί ισοδυναμούν με τους δεκαδικούς από 0 μέχρι 255. Αν το ψηφιακό μας σύστημα διαχειρίζεται δεδομένα σε καταχωρητές 16 Bit τότε ο προηγούμενος πίνακας γίνεται:

	Βάση 2	**Βάση 16**
Ελάχιστος αριθμός	0000 0000 0000 0000	0000H
Μέγιστος αριθμός	1111 1111 1111 1111	FFFFH

Οι παραπάνω αριθμοί ισοδυναμούν με τους δεκαδικούς από 0 μέχρι 65535.

Υπερχείλιση

Όταν κάνουμε πράξεις με δεδομένα καταχωρημένα σε οκτάμπιτους καταχωρητές υπάρχει ενδεχόμενο μία πρόσθεση να υπερβαίνει το 255 οπότε λέμε ότι έχουμε υπερχείλιση. Για να παρουσιάσουμε και να εγγράψουμε το αποτέλεσμα χρειαζόμαστε ένα επιπλέον Bit.

Αριθμοί με πρόσημο

Στα ψηφιακά κυκλώματα που χρησιμοποιούν την δυαδική λογική προφανώς και χρησιμοποιούνται για τις αριθμητικές πράξεις αριθμοί με θετικό ή αρνητικό πρόσημο. Τα πρόσημα δεν μπορεί να είναι – ή + και πρέπει αναγκαστικά να κωδικοποιηθούν σε 0 ή 1. Έχουμε λοιπόν ότι το πρόσημο + αναφέρεται με το 0 ενώ το πρόσημο – με το 1. Τα παραπάνω Bit τοποθετούνται μπροστά από τον αριθμό. Έτσι για παράδειγμα ο αριθμός 00001111=+15 ενώ ο 10001111=-15. Επίσης το +100=0100 και το -100=1100.

Παραδείγματα: Καταχωρήστε τους δεκαδικούς αριθμούς -7, +25, -32 σε προσημασμένη, δυαδική μορφή σε οκτάμπιτους καταχωρητές και επίσης τους -17, +25, -64, -128 σε δεκαεξάμπιτους καταχωρητές .

Προσημασμένος δεκαδικός	**Προσημασμένος δυαδικός**
-7	1000 0111
+25	0001 1001
-32	1010 0000
-17	1001 0001
-25	1001 1001
-64	1010 0000
-128	1100 0000

Ας υποθέσουμε ότι έχουμε να κάνουμε την δεκαδική αφαίρεση των παρακάτω αριθμών 67573-16151. Το συμπλήρωμα ως προς τη βάση 10 του αριθμού 16151 είναι 83849 Θα έχουμε κατά συνέπεια να κάνουμε την παρακάτω πρόσθεση.

$$\begin{array}{r} 67573 \\ +83849 \\ \hline 151422 \end{array}$$

Με την απόρριψη του τελικού κρατουμένου δηλαδή της μονάδας στη θέση MSB είναι σαν να αφαιρούμε το 10^5=100000 από το 151422 οπότε μένει το 51422 που είναι και το τελικό αποτέλεσμα.

67573

+83849

151422

Ας δούμε ένα άλλο παράδειγμα. Έστω ότι έχουμε να κάνουμε την παρακάτω αφαίρεση 12567-68951 δεκαδικών αριθμών. Το συμπλήρωμα ως προς 10 του 68951 είναι 31049 και αντί για αφαίρεση κάνουμε πρόσθεση.

12567

+31049

43616

Επειδή η πρόσθεση δίνει μηδενικό κρατούμενο το πραγματικό αποτέλεσμα της αρχικής αφαίρεσης είναι το συμπλήρωμα ως προς 10 του 43616 είναι (99999-43616+1)=56384 προφανώς με το πρόσημο – και τελικά το αποτέλεσμα της αφαίρεσης θα είναι το -56384. Στην παραπάνω περίπτωση επειδή ο αφαιρετέος (12567) είναι μικρότερος από τον αφαιρέτη (68951) το αποτέλεσμα θα είναι αρνητικό. Όταν δεν έχουμε τελικό κρατούμενο σημαίνει ότι έχουμε αρνητικό αποτέλεσμα και θα πρέπει να πάρουμε το συμπλήρωμα ως προς 10 για να βρούμε το τελικό αποτέλεσμα. Το πρόσημο δεν μπορεί να εξασφαλισθεί από τις πράξεις αφού οι αριθμοί που χρησιμοποιούμε δεν είναι προσημασμένοι. Η αφαίρεση δυαδικών αριθμών γίνεται με τον ίδιο τρόπο όπως και προηγουμένως. Ας δούμε δύο όμοια παραδείγματα.

Δίνονται οι δυαδικοί αριθμοί X=110001 και Y=101010. Χρησιμοποιώντας τα συμπληρώματα ως προς 2 εκτελέστε τις πράξεις X-Y και Y-X. Βλέπουμε ότι ο X>Y. Θα έχουμε

110001

-101010

?

Το συμπλήρωμα ως προς 2 του Y είναι 010101+1=010110. Κατά συνέπεια μπορούμε να προχωρήσουμε στην παρακάτω πρόσθεση.

110001

-010110

1000111

Με την απόρριψη του τελικού κρατουμένου το τελικό αποτέλεσμα είναι το $(000111)_2=(7)_{10}$ Ας επιβεβαιώσουμε το αποτέλεσμα της αφαίρεσης. Το $X=(110001)_2=(49)_{10}$. Το $Y=(101010)_2=(42)_{10}$. Οπότε έχουμε 49-42=7 σωστή η αφαίρεση.

Ας δούμε τώρα και το Y-X. Θα έχουμε λοιπόν:

101010

-110001

?

Το συμπλήρωμα ως προς 2 του X είναι 001110+1=001111. Κατά συνέπεια μπορούμε να προχωρήσουμε στην παρακάτω πρόσθεση.

101010

+001111

111001

Επειδή η πρόσθεση δίνει μηδενικό κρατούμενο το πραγματικό αποτέλεσμα της αρχικής αφαίρεσης είναι το συμπλήρωμα ως προς 2 του 111001 είναι 000110+1=000111 προφανώς με το πρόσημο – και τελικά το αποτέλεσμα της αφαίρεσης θα είναι το -000111 δηλαδή το -111=-7. Οι πράξεις είναι σωστές.

Σχετικά με τους προσημασμένους αριθμούς έχουμε δει ότι το πρόσημο τοποθετείται στο αριστερότερο bit. Αν είναι 0 τότε ο αριθμός είναι θετικός ενώ αν είναι 1 ο αριθμός είναι αρνητικός. Σε αυτή την περίπτωση το MSB είναι το δεύτερο bit από αριστερά.

Ας δούμε μερικούς ορισμούς στη περίπτωση των προσημασμένων αριθμών. Ας υποθέσουμε ότι οι δυαδικοί αριθμοί που χρησιμοποιήσουμε είναι 8 bit μαζί με το bit προσήμου. Ας πάρουμε τον αριθμό +8 στο δυαδικό η παρακάτω αναπαράσταση ονομάζεται **προσημασμένο μέτρο**. Θα είναι:

0000	1000

Αν έχουμε τον αριθμό -8 τότε η αναπαράσταση προσημασμένου μέτρου θα είναι:

1000	1000

Το **προσημασμένο συμπλήρωμα ως προς 1** είναι το παρακάτω για τον ίδιο αριθμό -8. Για να το βρούμε κρατάμε το bit προσήμου αναλλοίωτο και μετατρέπουμε τα ο σε 1 και τα 1 σε 0 όπως φαίνεται παρακάτω.

1111	0111

Για να βρούμε το **προσημασμένο συμπλήρωμα ως προς 2** αρκεί να προσθέσουμε στο προσημασμένο συμπλήρωμα ως προς 1 που βρήκαμε προηγουμένως το 1.

1111	1000

Για να συνεχίσουμε παρακάτω θα κωδικοποιήσουμε τα παρακάτω όπως: **ΠΜ** θα είναι το **προσημασμένο μέτρο** ενός αριθμού με δεδομένο αριθμό bits**, ΠΣ/1** θα είναι το **προσημασμένο συμπλήρωμα ως προς 1,** και **ΠΣ/2** θα είναι το **προσημασμένο συμπλήρωμα ως προς 2.** Ας δούμε με ένα παράδειγμα τα παραπάνω. Ας γράψουμε το **ΠΜ** του προσημασμένου δυαδικού αριθμού +15, το **ΠΜ** του -15, το **ΠΣ/1** του -15 και το **ΠΣ/2** του -15**.**

0000	1111

Είναι το **ΠΜ** του +1111

1000	1111

Είναι το **ΠΜ** του -1111

1111	0000

Είναι το **ΠΣ/1** του -1111

1111	0001

Είναι το **ΠΣ/2** του -1111

Ας κάνουμε μερικές πράξεις για να καταλάβουμε τη χρησιμότητα των παραπάνω. Ας υποθέσουμε ότι έχουμε να προσθέσουμε τον προσημασμένο αριθμό +8 (00001000) με το -15 (10001111). Για να γίνει αυτό αρκεί να προσθέσουμε στο ΠΜ του +8 το ΠΣ/2 του -15. Δηλαδή:

0000 1000

+1111 0001

1111 1001

Για να βρούμε το σωστό αποτέλεσμα θα πρέπει να πάρουμε το **ΠΣ/2** του παραπάνω αριθμού. Για να συμβεί αυτό αρκεί να κάνουμε τα 1 σε 0 και τα 0 σε 1 χωρίς να αγγίξουμε το προσήμου και να προσθέσουμε μία μονάδα. Δηλαδή 1000 0110+1=1000 0111 δηλαδή -7. Σωστό αποτέλεσμα.

Ας κάνουμε πράξεις με τους παρακάτω αριθμούς +12=0000 1100 και με το +17= 0001 0001

ΠΜ +12=0000 1100

ΠΜ +17=0001 0001

Ας κάνουμε

0000 1100

+ 0001 0001

0001 1101

Το αποτέλεσμα 0001 1101=+29 είναι σωστό.

Ας κάνουμε τώρα την πρόσθεση του +12 με το -19. Το **ΠΜ** +12=0000 1100 και το **ΠΜ** -19=1001 0011 και το **ΠΣ/2** του -19 θα είναι το παρακάτω 1110 1101. Άρα θα κάνουμε πρόσθεση όπως παρακάτω:

0000 1100

+ 1110 1101

1111 1001

Το αποτέλεσμα είναι το **ΠΣ/2** του 1111 1001. Για να το βρούμε αρκεί να κάνουμε τα 0 σε 1 και τα 1 σε 0 αφήνοντας το bit προσήμου. Δηλαδή θα έχουμε 1000 0110+1=1000 0111 που είναι το δεκαδικό -7 και είναι το σωστό αποτέλεσμα.

Ας υποθέσουμε ότι θέλουμε να προσθέσουμε τους αριθμούς το -16 με το -7. Θα έχουμε για το -16=1001 0000 και το -7=1000 0111 . Τα **ΠΣ/2** των δύο παραπάνω αριθμών θα είναι **ΠΣ/2** του -16 θα είναι 1111 0000 και το **ΠΣ/2** του -7 θα είναι 1111 1001. Θα έχουμε για την πρόσθεση:

1111 0000

+ 1111 1001

11110 1001

Αν απορρίψουμε το τελικό κρατούμενο τότε το αποτέλεσμα θα είναι το **ΠΣ/2** του 1110 1001. Για να το βρούμε αρκεί να διατηρήσουμε το bit προσήμου να αλλάξουμε τα 0 σε 1 και τα 1 σε 0 και προσθέτουμε 1. Θα έχουμε δηλαδή 1001 0110+1=1001 0111=-23 που είναι και το σωστό αποτέλεσμα.

Πρόσθεση στο BCD. Ας υποθέσουμε ότι έχουμε δύο αριθμούς X και Y που ανήκουν στον κώδικα BCD και ας υποθέσουμε ότι τους προσθέτουμε και το αποτέλεσμα είναι το παρακάτω:

$$Z=X+Y$$

Αν $Z \leq 9$ τότε το άθροισμα S είναι ίσο με Z δηλαδή S=Z και το κρατούμενο C=0, ενώ αν Z>9 το άθροισμα είναι S=Z+6 και C=1. Αν X=0111 =7 και το Y=0101 =5 τότε το αποτέλεσμα θα είναι:

0111

+ 0101

1100

Όμως ο παραπάνω συνδυασμός 1100 δεν υπάρχει στον κώδικα BCD. Για να συμβεί αυτό θα πρέπει να προσθέσουμε το 6 δηλαδή το 0110. Θα έχουμε λοιπόν:

1100

+ 0110

10010

Το 10010 γράφετε 0001 0010 δηλαδή στο 12 που είναι και το σωστό αποτέλεσμα.

Ας δούμε άλλο ένα παράδειγμα. Το +23 είναι 0010 0011 στο BCD και επίσης το +99=1001 1001 επίσης στον κώδικα BCD. Ας κάνουμε την πρόσθεση:

0010 0011

+ 1001 1001

……1100

Προσθέτοντας τα δύο αριστερά ψηφία κωδικοποιημένα στον κώδικα BCD βλέπουμε ότι το αποτέλεσμα δεν είναι συνδυασμός του παραπάνω κώδικα και προσθέτουμε πριν συνεχίσουμε στο επόμενο ψηφίο το 6=0110.Οπότε θα έχουμε πριν πάμε στο επόμενο ψηφίο:

$$\begin{array}{r} 1100 \\ +\ 0110 \\ \hline 1\ 0010 \end{array}$$

Οπότε συνεχίζουμε την πρόσθεση:

$$\begin{array}{r} 0010\ 0011 \\ +\ 1001\ 1001 \\ \hline 1100\ 0010 \end{array}$$

Πάλι όμως για το δεύτερο ψηφίο ο συνδυασμός που προκύπτει δεν ανήκει στον κώδικα και θα πρέπει να προσθέσουμε το 0110 όπως:

$$\begin{array}{r} 1100\ 0010 \\ +\ \ 0110\ 0000 \\ \hline 10010\ 0010 \\ \hline \end{array}$$

Άρα το τελικό αποτέλεσμα θα είναι 0001 0010 0010= 122 που είναι και το σωστό αποτέλεσμα.

Άσκηση

Προσθέστε τους παραπάνω αριθμούς γραμμένους στον κώδικα BCD. 153+798. Δηλαδή 0001 0101 0011 + 0111 1001 1000 το τελικό αποτέλεσμα θα είναι 1001 0101 0001= 951 σωστό. Προσπαθήστε να το βρείτε.

Άσκηση

Προσθέστε τους παραπάνω αριθμούς γραμμένους στον κώδικα BCD. 153+798. Δηλαδή 0001 0101 0011+0111 1001 1000 το τελικό αποτέλεσμα θα είναι 1001 0101 0001= 951 σωστό. Προσπαθήστε να το βρείτε.

Ασκήσεις

1. Δίνονται οι παρακάτω αριθμοί A,B,C,D.Υπολογίστε A+B, A-B, και C·D.

A=1011010, B=101111, C=1010101, D=110

2. Υπολογίστε το γινόμενο A.B και δώστε το αποτέλεσμα στην βάση 5.

$A=[101011]_2$ $B=[110001]_2$

3. Κάνετε τις παρακάτω μετατροπές

α) $(250)_8$ $(?)_{10}$

β) $(250)_8$ $(?)_{16}$

γ) $(250)_8$ $(?)_2$

δ) $(182.25)_{10}$ $(?)_8$

ε) $(1023)_4$ $(?)_5$

ζ) $(1011001111001001)_{\text{Excess 3}}$ $(?)_{\text{BCD}}$

η) $(1011001111001001)_{\text{Excess 3}}$ $(?)_{10}$

θ) $(0011100001111100)_{\text{Excess 3}}$ $(?)_{4221}$

4. Μετατρέψτε τους παρακάτω δεκαδικούς 197, 2056 στην βάση, 2, 4, 5, 8 και 16

5. Γράψετε στον κώδικα BCD και στον κώδικα EXCESS3 τους παρακάτω δεκαδικούς αριθμούς 1821, 45678, 400789

6. Γράψετε στον κώδικα ASCII το 1821, το 7d7b και το 45899

7. Γράψετε στον κώδικα ASCII τη φράση Time is $

Κεφάλαιο 2. Άλγεβρα BOOLE

Σε αυτό το κεφάλαιο θα δούμε το μαθηματικό υπόβαθρο που στηρίζει τη λειτουργία των λογικών κυκλωμάτων. Ονομάζεται άλγεβρα Boole και μας παρέχει τα βασικά εργαλεία ανάλυσης και σχεδίασης ενός ψηφιακού κυκλώματος. Η ανάλυση η σχεδίαση και η λειτουργία ενός ψηφιακού ή λογικού κυκλώματος είναι μια αυστηρή μαθηματική διαδικασία η οποία μας επιτρέπει την μετατροπή μαθηματικών εργαλείων που περιγράφουν το προς επίλυση πρόβλημα και το μετατρέπουν σε ένα λογικό κύκλωμα.

Άλγεβρα Boole

Η άλγεβρα του Boole βασίζεται σε επτά αξιώματα τα οποία θεωρούνται δεδομένα, δεν υπάρχουν αποδείξεις αφού είναι αξιώματα, και παρουσιάζονται παρακάτω:

1. Μία **ΑΛΓΕΒΡΑ BOOLE** είναι ένα κλειστό σύστημα το οποίο περιέχει Κ η περισσότερα στοιχεία και δύο τελεστές · και + Για κάθε στοιχείο a, b που ανήκει στο Κ το a+b ανήκει Κ όπως επίσης και το a.b

Ο τελεστής **+** ονομάζεται Η **(OR)**

Ο τελεστής **·** ονομάζεται ΚΑΙ **(AND)**

2. Στο κλειστό αλγεβρικό σύστημα δύο εκφράσεις είναι ίσες (=) όταν η μία μπορεί να αντικαταστήσει την άλλη.

3. Υπάρχουν δύο μοναδικά στοιχεία το 1 και το 0 τα οποία ανήκουν στο Κ και έχουμε για κάθε a που ανήκει στο Κ ότι.

$$a+0=a \quad a.1=a$$

4. Για κάθε a,b στοιχεία του Κ έχουμε.

$$a+b=b+a$$

$$a.b=b.a$$

5. Για κάθε a,b,c στοιχεία του Κ έχουμε.

$$a+(b+c)=(a+b)+c$$

$$a.(b.c)=(a.b).c$$

6. Για κάθε a,b,c στοιχεία του Κ έχουμε.

a+(b.c)=(a+b).(a+c)

a.(b+c)=a.b+a.c

7. Για κάθε στοιχείο a του K υπάρχει ένα μοναδικό στοιχείο στο K (συμπλήρωμα του a) έτσι ώστε.

$$a + \bar{a} = 1 \qquad a.\bar{a} = 0$$

Χρησιμοποιώντας τα παραπάνω επτά αξιώματα μπορούμε να αποδείξουμε τα παρακάτω θεωρήματα της άλγεβρας του BOOLE.

Θεώρημα 8

a+a=a a.a=a

Θεώρημα 9

1. $a + 1 = 1$
2. $a \cdot 0 = 0$

Απόδειξη του 1.

$$a + 1 = 1 + a = [a + 1] \cdot 1$$
$$= 1 \cdot [a + 1] = [a + \bar{a}][a + 1]$$
$$= a + \bar{a} = 1$$

Θεώρημα 10

1. $a + a \cdot b = a$
2. $a \cdot [a + b] = a$

Απόδειξη του 1.

$$a + a \cdot b = a \cdot 1 + a \cdot b =$$
$$= a\,[1 + b] = a[b + 1] =$$
$$= a \cdot 1 = a$$

Θεώρημα 11

1. $a + \bar{a} \cdot b = a + b$
2. $a \cdot [\bar{a} + b] = a \cdot b$

Απόδειξη του 2.

$$a\cdot[\bar{a}+b]=a\cdot[\bar{a}+b]=a\bar{a}+a\cdot b$$
$$=0+a\cdot b=a\cdot b+0=a\cdot b$$

Θεώρημα 12:

1. $\overline{a+b}=\bar{a}\cdot\bar{b}$
2. $\overline{a\cdot b}=\bar{a}+\bar{b}$

Τα δύο αυτά θεωρήματα είναι του De Morgan και η απόδειξη τους πολύ εύκολη με τη βοήθεια των πινάκων αληθείας και μπορούν πολύ εύκολα να γενικευτούν στην περίπτωση που έχουμε πολλές έως και άπειρες μεταβλητές όπως φαίνεται παρακάτω.

$$\overline{A_1+A_2+A_3+A_4\ldots\ldots+A_{n-1}+A_n}=\bar{A}_1\cdot\bar{A}_2\cdot\bar{A}_3\cdot\bar{A}_4\ldots\ldots\bar{A}_{n-1}\cdot\bar{A}_n$$
$$\overline{A_1\cdot A_2\cdot A_3\cdot A_4\ldots\ldots\ldots\ldots A_{n-1}\cdot A_n}=\bar{A}_1+\bar{A}_2+\bar{A}_3+\ldots.+\bar{A}_{n-1}+\bar{A}_n$$

Θεώρημα 13

1. $ab+\bar{a}c+bc=ab+\bar{a}c$
2. $[a+b]\cdot[\bar{a}+c]\cdot[b+c]=[a+b]\cdot[\bar{a}+c]$

Απόδειξη του 1.

$$ab+\bar{a}c+bc=ab+\bar{a}c+bc1=$$
$$=ab+\bar{a}c+[a+\bar{a}]bc=$$
$$=ab+\bar{a}c+abc+\bar{a}bc=$$
$$=a[b+bc]+\bar{a}[c+bc]$$
$$=ab+\bar{a}c$$

Θεώρημα 14

1. $ab+a\bar{b}=a$
2. $[a+b][a+\bar{b}]=a$

Απόδειξη του 1.

$$ab + a\bar{b} = ab + a\bar{b} =$$
$$= a[b + \bar{b}] = a \cdot 1 = a$$

Θεώρημα 15

1. $ab + a\bar{b}c = ab + ac$
2. $[a + b] \cdot [a + \bar{b} + c] = [a + b] \cdot [a + c]$

Απόδειξη του 1

$$ab + a\bar{b}c = a[b + \bar{b}c]$$
$$= a[b + c]$$
$$= ab + ac$$

Θεώρημα 16

1. $ab + \bar{a}c = [a + c][\bar{a} + b]$
2. $[a + b][\bar{a} + c] = ac + \bar{a}b$

Απόδειξη του 1

$$ab + \bar{a}c = [ab + \bar{a}][ab + c]$$
$$= [a + \bar{a}][b + \bar{a}][a + c][b + c]$$
$$= 1[\bar{a} + b][a + c][b + c]$$
$$= [\bar{a} + b][a + c][b + c]$$
$$= [\bar{a} + b][a + c]$$

Το θεώρημα αυτό είναι χρήσιμο όταν θέλουμε να αλλάξουμε την παρουσίαση μίας λογικής συνάρτησης.

Από την εφαρμογή των σχέσεων της Άλγεβρας Boole προκύπτει ο παρακάτω πίνακας για τη δίτιμη Άλγεβρα.

$\bar{1} = 0$	$\bar{0} = 1$
$0 + 0 = 0$	$1 \cdot 1 = 1$
$0 + 1 = 1$	$0 \cdot 1 = 0$
$1 + 0 = 1$	$1 \cdot 0 = 0$
$1 + 1 = 1$	$0 \cdot 0 = 0$

Τα αξιώματα και τα θεωρήματα της άλγεβρας του BOOLE που παρουσιάσαμε έχουν ισχύ και στην περίπτωση που μας ενδιαφέρει όπου τα κυκλώματα είναι δισταθή και το σύνολο Κ έχει μόνο δύο στοιχεία το μηδέν και το ένα. Κ=[0,1] Έτσι μία λογική συνάρτηση πολλών μεταβλητών Χ1,Χ2, Χ3,.....Χn θα παίρνει την τιμή 0 η 1 ανάλογα με τις τιμές των παραπάνω μεταβλητών και τους τελεστές που τις συνδέουν μεταξύ τους.

Λογικές Συναρτήσεις

Οι λογικές συναρτήσεις ορίζονται όπως οι κλασσικές συναρτήσεις που έχουμε όλοι δει στην ανάλυση. Έχουμε δηλαδή ένα πεδίο ορισμού, ένα πεδίο τιμών και μια αναλυτική έκφραση της κάθε συνάρτησης. Στην περίπτωση των λογικών συναρτήσεων το πεδίο ορισμού και το πεδίο τιμών είναι ένα και το αυτό και περιέχει 2 στοιχεία, το 0 και το 1. Είναι το [0,1]. Η αναλυτική έκφραση είναι ένας συνδυασμός μεταβλητών με τους τελεστές AND OR. Ας δούμε παρακάτω ένα παράδειγμα.

Παράδειγμα

Έστω η λογική συνάρτηση F(A,B,C)=A.B+B.C+A.C. Βλέπουμε ότι έχουμε συνδυασμούς των μεταβλητών A,B,C και των τελεστών + και . Μπορούμε με βάση αυτά που έχουμε ήδη δει από την άλγεβρα Boole να αναλύσουμε την παραπάνω σχέση και να πούμε ότι η συνάρτηση F(A,B,C) με τον τρόπο που δίνεται από την αναλυτική σχέση ότι θα ισούται με το λογικό 1 ή θα είναι αληθής (είναι το ίδιο πράγμα) όταν ό ένας τουλάχιστον από τους τρεις όρους είναι ίσο με 1, δηλαδή όταν A=B=1 η όταν B=C=1 η όταν A=C=1. Σε όλες τις άλλες περιπτώσεις η παραπάνω συνάρτηση θα μηδενίζεται. Πρέπει να δώσουμε μεγάλη προσοχή στις παραπάνω σχέσεις που γράψαμε γιατί το **ή** όπως θα δούμε παρακάτω σημαίνει μία πύλη και το **και** μία άλλη εντελώς διαφορετική. Τις πύλες αυτές θα τις χρησιμοποιήσουμε για να υλοποιήσουμε τα λογικά κυκλώματα σαν κατασκευή πιά.

Η αναλυτική έκφραση είναι ένας τρόπος παρουσίασης μιας λογικής συνάρτησης. Ένας άλλος τρόπος παρουσίασης μίας λογικής συνάρτησης γίνεται με τον πίνακα αληθείας. Ο πίνακας αληθείας μιας συνάρτησης περιέχει όλους τους δυνατούς συνδυασμούς μεταξύ των μεταβλητών που την ορίζουν και τις αντίστοιχες τιμές της συνάρτησης F. Δηλαδή αν η

συνάρτηση είναι Ν μεταβλητών τότε όλοι οι δυνατοί συνδυασμοί μεταξύ των μεταβλητών είναι 2^N. Ο πίνακας αλήθειας θα έχει λοιπόν Ν+1 κολόνες και 2^N γραμμές.

Οι Ν κολόνες θα δείχνουν τις τιμές των Ν μεταβλητών όπως επίσης και όλων των δυνατών συνδυασμών και η κολόνα Ν+1 θα δίνει τη λογική τιμή της συνάρτησης που ορίζουν οι μεταβλητές.

Παράδειγμα: έστω η συνάρτηση $F(A,B) = \overline{A}B + A\overline{B}$ έχουμε ότι

F(0,0)=1·0+0·1=0

F(0,1)=1·1+0·0=1

F(1,0)=0·0+1·1=1

F(1,1)=0·1+1·0=0

Οι παραπάνω σχέσεις μας λένε ότι η συνάρτηση F(A,B) είναι ίση με 1 όταν το Α=0 και το Β=1 ή όταν το Α=1 και το Β=0. Ο πίνακας αλήθειας της παραπάνω συνάρτησης θα είναι σύμφωνα με τον ορισμό που δώσαμε προηγουμένως ο ακόλουθος:

A	B	F(A,B)
0	0	0
0	1	1
1	0	1
1	1	0

Για μία συνάρτηση Ν μεταβλητών ο πίνακας αλήθειας έχει την ακόλουθη μορφή και περιέχει όλους τους δυνατούς συνδυασμούς μεταξύ των μεταβλητών που την ορίζουν. Οι συνδυασμοί είναι **2^N** και μπορούμε να αριθμήσουμε από το 0 μέχρι το **$(2^N - 1)$.**

X_1	X_2	X_3	.	.	.	X_n	$F(X_1,..X_n)$
0	0	0	.	.	.	0	a_0
0	.	.	.	.	.	1	a_1
.	.	.	.	.	.	.	.

.	.	.	.	.	.	.	.
1	1	.	.	.	.	0	.
1	1	1	.	.	.	1	a_n

Έστω η λογική συνάρτηση F(A,B)=A·B Η παραπάνω συνάρτηση παίρνει τη λογική τιμή 1 όταν το Α=1 και το Β=1 Ο πίνακας αλήθειας φαίνεται παρακάτω.

A	B	A·B	F(A,B)
0	0	0	0
0	1	0	0
1	0	0	0
1	1	1	1

Παράδειγμα Βρείτε τον πίνακα αλήθειας της παρακάτω λογικής συνάρτησης τριών μεταβλητών.

$$F(A,B,C) = A \cdot B \cdot C + A \cdot \overline{B} \cdot \overline{C}$$

Βλέπουμε ότι η παραπάνω συνάρτηση θα πάρει τη λογική τιμή 1 όταν θα έχουμε το Α=1 και το Β=1 και το C=1 ή το Α=1 και το Β=0 και το C=0. Μπορούμε με αυτό τον τρόπο εύκολα να συμπληρώσουμε τον πίνακα αλήθειας που φαίνεται παρακάτω. Στις 3 αριστερές κολώνες βλέπουμε τους 8 συνδυασμούς των μεταβλητών Α, Β, C, στις επόμενες 2 κολώνες βλέπουμε τους 2 όρους που ορίζουν την συνάρτηση αναλυτική και τέλος στις 2 τελευταίες βλέπουμε την ένωση των δύο όπως με τελεστή ή και την ίδια τη συνάρτηση.

A	B	C	A·B·C	$A.\overline{B}.\overline{C}$	ABC+ $A.\overline{B}.\overline{C}$	F
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0

0	1	1	0	0	0	0
1	0	0	0	1	1	1
1	0	1	0	0	0	0
1	1	0	0	0	0	0
1	1	1	1	0	1	1

Υπάρχει το ενδεχόμενο αναζήτησης του πίνακα αλήθειας μίας συνάρτησης Ν μεταβλητών όπου κάθε όρος περιέχει λιγότερες από το σύνολο των μεταβλητών. Τότε για απλοποίηση των πράξεων καλό είναι να εισάγω στους όρους που λείπουν τις αντίστοιχες μεταβλητές χρησιμοποιώντας το ουδέτερο στοιχείο του πολλαπλασιασμού 1 το οποίο γράφουμε ίσον με μία μεταβλητή συν το συμπλήρωμα της.

Παράδειγμα Βρείτε τον πίνακα αλήθειας της παρακάτω λογικής συνάρτησης F(A,B,C)=AB+BC. Βλέπουμε ότι ενώ η συνάρτηση είναι μια συνάρτηση 3 μεταβλητών οι δύο όροι που την καθορίζουν αναλυτικά περιέχουν μόνο τις 2 μεταβλητές. Μπορούμε χρησιμοποιώντας το ουδέτερο στοιχείο του πολλαπλασιασμού να εισάγουμε την μεταβλητή που λείπει από κάθε όρο όπως φαίνεται παρακάτω:

$$F(A,B,C) = AB + BC = AB(C + \overline{C}) + (A + \overline{A})BC \Rightarrow$$
$$F(A,B,C) = ABC + AB\overline{C} + ABC + \overline{A}BC \Rightarrow$$
$$F(A,B,C) = ABC + AB\overline{C} + \overline{A}BC$$

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

1	1	1	1

Συμπλήρωμα μίας συνάρτησης

Ο πίνακας αλήθειας που αποτελεί το συμπλήρωμα μίας λογικής συνάρτησης μπορεί να ευρεθεί πολύ εύκολα από τον πίνακα αλήθειας της κανονικής συνάρτησης. Πρέπει να υπενθυμίσουμε ότι η συμπληρωματική συνάρτηση είναι αληθής ίση δηλαδή με **1** όταν η κανονική είναι **0** και ίση με **0** όταν η κανονική είναι ίση με **1** όπως φαίνεται καθαρά στο παρακάτω παράδειγμα. Η διαδικασία λοιπόν είναι η ακόλουθη. Σε μια διπλανή κολώνα από την συνάρτηση F ορίζουμε την συμπληρωματική συνάρτηση και συμπληρώνουμε την κολώνα όπου είχαμε 0 με 1 και όπου είχαμε 1 βάζουμε 0.

Παράδειγμα

A	**B**	**C**	**F**	**ΣΥΜΠΛΗΡΩΜΑ της F**
0	0	0	0	1
0	0	1	1	0
0	1	0	1	0
0	1	1	1	0
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

ΛΟΓΙΚΕΣ ΠΥΛΕΣ

Οι ηλεκτρονικοί υπολογιστές και γενικά τα ψηφιακά συστήματα αποτελούνται από ένα μεγάλο αριθμό κυκλωμάτων τα οποία είναι συνδυαστικά η ακολουθιακά ή στην γενικότερη περίπτωση ένας συνδυασμός και των δύο.

Η πρώτη ενότητα του μαθήματος που είναι τα συνδυαστικά κυκλώματα ξεκινούν από την ανάλυση των βασικών λογικών πράξεων και των

βασικών λογικών πυλών που σχηματίζουν αυτά τα κυκλώματα και μας επιτρέπουν να υλοποιήσουμε κυκλωματικά τις λογικές συναρτήσεις μέσα από τις σχέσεις της άλγεβρας Boole. Η δυαδική λογική είναι το μαθηματικό υπόβαθρο που περιγράφει με ακρίβεια και αυστηρότητα τις μαθηματικές διεργασίες που επιτρέπουν την επεξεργασία των δυαδικών πληροφοριών. Η βασική δυαδική πληροφορία ονομάζεται δυαδικό κύτταρο BIT και μπορεί να πάρει δύο τιμές, 0 ή 1. Ομοίως μεταβλητές που χρησιμοποιούνται μπορούν να πάρουν μόνο δύο διακριτές τιμές το μηδέν η το ένα. Πέντε βασικές λογικές πύλες "ΚΑΙ" [AND], "Η" [OR] ,"ΟΧΙ" [NOT], "ΟΧΙ ΚΑΙ" NAND,και "ΟΧΙ Η" NOR καθορίζουν τη δυαδική λογική. Οι πίνακες αληθείας των παραπάνω λογικών πυλών φαίνονται παρακάτω.

A	NOT
0	1
1	0

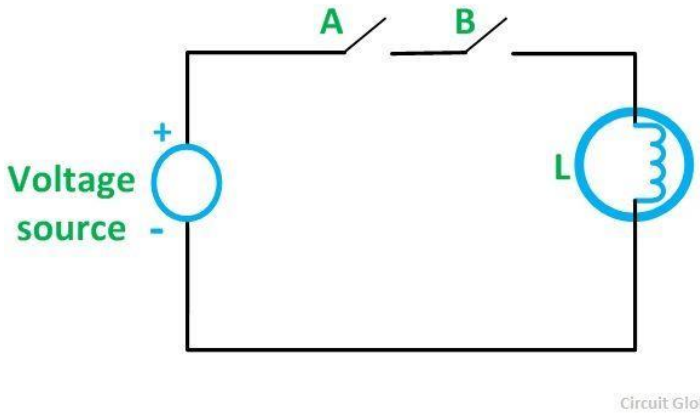
A	B	AND	OR	NAND	XOR	XNOR
0	0	0	0	1	0	1
0	1	0	1	1	1	0
1	0	0	1	1	1	0
1	1	1	1	0	0	1

Παρακάτω θα δούμε ξανά τις παραπάνω πύλες αναλυτικά όπως επίσης και τις συναρτήσεις που αναπαράγουν στην έξοδο του.

ΑΝΑΛΥΣΗ ΤΩΝ ΒΑΣΙΚΩΝ ΛΟΓΙΚΩΝ ΠΡΑΞΕΩΝ

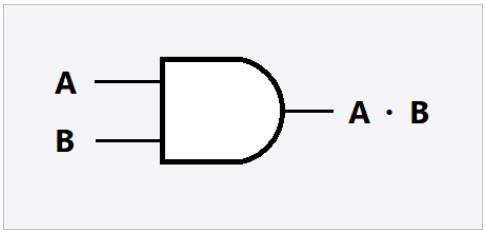
1. Τελεστής **ΚΑΙ (AND):** Η πράξη και σημειώνεται με μία τελεία $X \cdot Y = Z$ και σημαίνει ότι το Z είναι ίσο με 1 όταν και το X και το Y είναι ίσα με ένα. Είναι πιο εύκολο να καταλάβουμε τη λογική αυτής της πράξης αν την παρομοιάσουμε με ένα ηλεκτρικό κύκλωμα, ή αν δούμε το ηλεκτρικό ισοδύναμο όπου ένας λαμπτήρας συνδέεται με μία πηγή τάσης μέσω δύο διακοπτών Α, Β οι οποίοι είναι εν σειρά όπως φαίνεται παρακάτω. Αυτό

σημαίνει ότι για να ανάψει ο λαμπτήρας θα πρέπει και οι δύο διακόπτες να είναι κλειστοί όπως φαίνεται στο παρακάτω κύκλωμα και στον πίνακα.



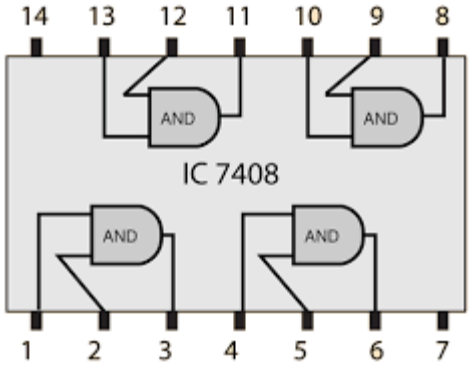
Διακόπτης Α	Διακόπτης Β	Λαμπτήρας
ανοικτός	ανοικτός	κλειστός
ανοικτός	κλειστός	κλειστός
κλειστός	ανοικτός	κλειστός
κλειστός	κλειστός	ανοικτός

Μπορούμε να δούμε την αντιστοιχία του παραπάνω πίνακα που περιγράφει τις διαφορετικές καταστάσεις των διακοπτών και του λαμπτήρα με αυτό που δώσαμε προηγουμένως για τη λογική πράξη ΚΑΙ. Δηλαδή ο διακόπτης Α υποκαθιστά την μεταβλητή Α και ο διακόπτης Β υποκαθιστά την μεταβλητή Β. Βλέπουμε από το κύκλωμα ότι ο λαμπτήρας είναι ανοικτός μόνο και μόνο όταν και ο διακόπτης Α είναι κλειστός και ο διακόπτης Β επίσης.



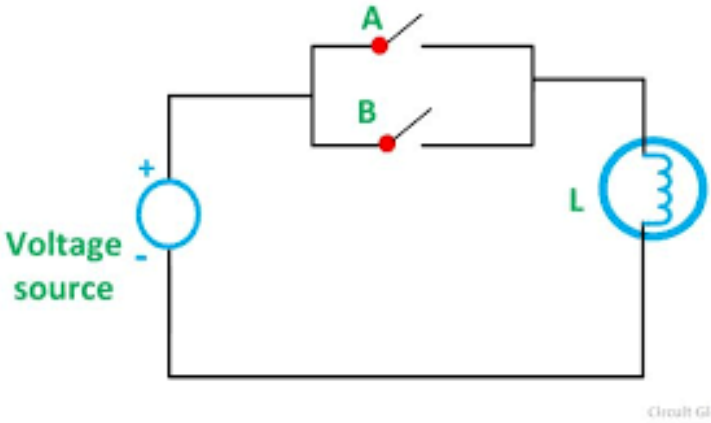
Βλέπουμε στο παραπάνω διάγραμμα τον συμβολισμό σαν κύκλωμα πιά μιας πύλης AND δύο εισόδων. Στα αριστερά φαίνονται οι δύο είδοδοι με τις μεταβλητές Α και Β αντίστοιχα και δεξιά φαίνεται η έξοδος με το αντίστοιχο

αποτέλεσμα δηλαδή **AB** ή **A AND B.** Παρακάτω βλέπουμε το ολοκληρωμένο κύκλωμα της σειράς TTL που θα δούμε παρακάτω αναλυτικά το οποίο περιέχει τέσσερεις πύλες AND δύο εισόδων η κάθε μία. Βλέπουμε ότι το παραπάνω ολοκληρωμένο έχει 14 PIN ή ακροδέκτες από τους οποίους το 7 είναι η γείωση ή GND και το PIN 14 η τροφοδοσία ή Vcc. Με την σύνδεση των παραπάνω PIN όλες οι πύλες τροφοδοτούνται και είναι κατάλληλες να λειτουργήσουν αρκεί να συνδέσουμε σωστά όπως αναφέρονται οι είσοδοι και οι έξοδοι των πυλών.



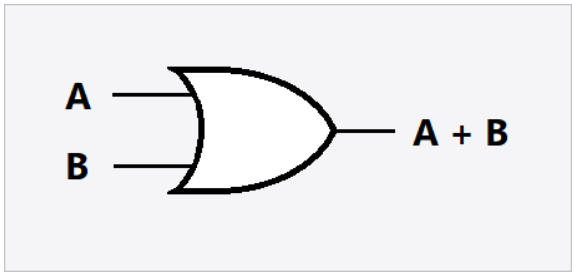
Βλέπουμε παραπάνω το ολοκληρωμένο κύκλωμα **7408** που θα χρησιμοποιήσουμε στο εργαστήριο για να επιβεβαιώσουμε την λειτουργία της πύλης.

2. Τελεστής **Η (OR):** Η λογική πράξη **ή** συμβολίζεται με το **+,** γράφουμε για 2 μεταβλητές X,Y που συνδέονται με αυτή την πράξη ότι X+Y=Z και δηλώνει ότι με το Z=1 όταν το X=1 η όταν το Y=1 ή όταν και τα δύο είναι ίσον με ένα. Το αντίστοιχο ηλεκτρικό ισοδύναμο όπως και προηγουμένως θα είναι ένας λαμπτήρας ο οποίος θα συνδέεται με μία πηγή τάσης μέσω πάλι δύο διακοπτών οι οποίοι αυτή τη φορά είναι παράλληλα συνδεδεμένοι μεταξύ τους. Αυτό σημαίνει ότι για να ανάψει ο λαμπτήρας θα πρέπει τουλάχιστον ένας διακόπτης να είναι ανοικτός, ή και οι δύο.



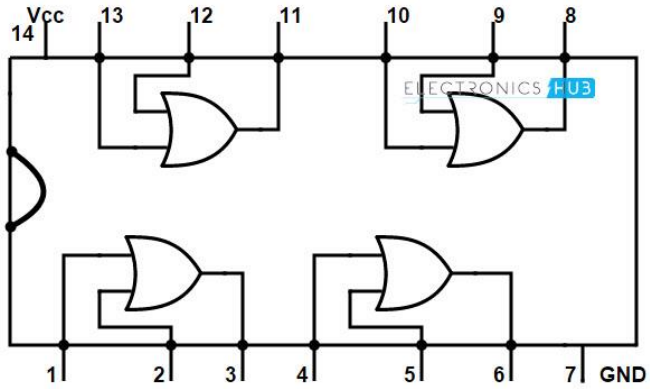
Διακόπτης Α	Διακόπτης Β	Λαμπτήρας
ανοικτός	ανοικτός	κλειστός
ανοικτός	κλειστός	ανοικτός
κλειστός	ανοικτός	ανοικτός
κλειστός	κλειστός	ανοικτός

Είναι φανερή η αντιστοιχία του παραπάνω πίνακα με αυτόν που δώσαμε προηγουμένως για τη λογική πράξη Η. Οι διακόπτες πάλι αντικαθιστούν τις μεταβλητές Χ και Υ. Μπορούμε να δούμε την αντιστοιχία του παραπάνω πίνακα που περιγράφει τις διαφορετικές καταστάσεις των διακοπτών και του λαμπτήρα με αυτό που δώσαμε προηγουμένως για τη λογική πράξη **Ή**. Δηλαδή ο διακόπτης Α υποκαθιστά την μεταβλητή Α και ο διακόπτης Β υποκαθιστά την μεταβλητή Β. Βλέπουμε από το κύκλωμα ότι ο λαμπτήρας είναι ανοικτός μόνο και μόνο όταν και ο διακόπτης Α είναι κλειστός ή και ο διακόπτης Β επίσης.



Βλέπουμε στο παραπάνω διάγραμμα τον συμβολισμό σαν κύκλωμα πιά μιας πύλης **OR** δύο εισόδων. Στα αριστερά φαίνονται οι δύο είδοδοι με τις

μεταβλητές Α και Β αντίστοιχα και δεξιά φαίνεται η έξοδος με το αντίστοιχο αποτέλεσμα δηλαδή **A+B** ή **A OR B.** Παρακάτω βλέπουμε το ολοκληρωμένο κύκλωμα της σειράς TTL που θα δούμε παρακάτω αναλυτικά το οποίο περιέχει τέσσερεις πύλες **OR** δύο εισόδων η κάθε μία. Βλέπουμε ότι το παραπάνω ολοκληρωμένο έχει 14 PIN ή ακροδέκτες από τους οποίους το 7 είναι η γείωση ή GND και το PIN 14 η τροφοδοσία ή Vcc. Με την σύνδεση των παραπάνω PIN όλες οι πύλες τροφοδοτούνται και είναι κατάλληλες να λειτουργήσουν αρκεί να συνδέσουμε σωστά όπως αναφέρονται οι είσοδοι και οι έξοδοι των πυλών.

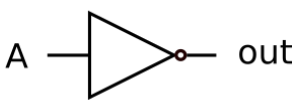


Βλέπουμε παραπάνω το ολοκληρωμένο κύκλωμα **7432** που θα χρησιμοποιήσουμε στο εργαστήριο για να επιβεβαιώσουμε την λειτουργία της πύλης OR.

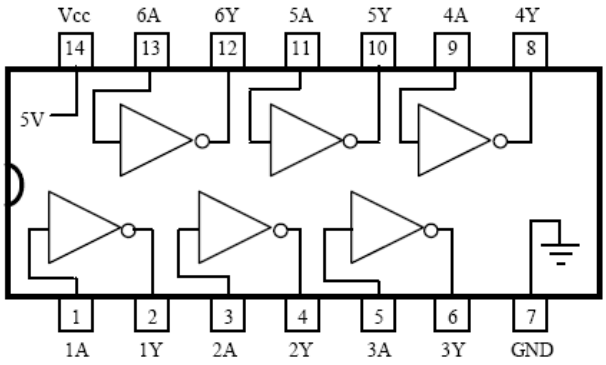
Η πύλη **NOT .** Η λογική πράξη όχι, συμβολίζεται με μία μπάρα πάνω από τη μεταβλητή την οποία και αντιστρέφει όπως βλέπουμε παρακάτω στον κυκλωματικό συμβολισμό της.

NOT:

A	out
0	1
1	0



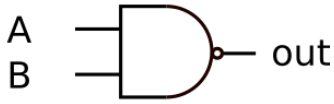
Παρακάτω βλέπουμε το ολοκληρωμένο κύκλωμα **7404** της σειράς TTL το περιέχει έξη πύλες **NOT** . Βλέπουμε ότι το παραπάνω ολοκληρωμένο έχει 14 PIN ή ακροδέκτες από τους οποίους το 7 είναι η γείωση ή GND και το PIN 14 η τροφοδοσία ή Vcc.



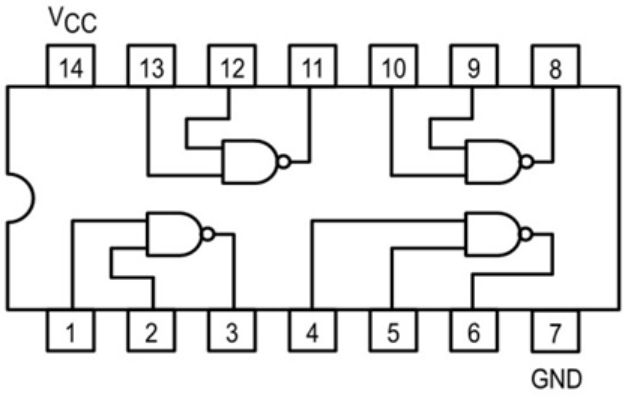
Η πύλη **NAND.** Είναι το συμπλήρωμα της λογικής πράξης **AND**. Βλέπουμε παρακάτω τον συμβολισμό της πύλης όπως επίσης και τον πίνακα αλήθειας της. Βλέπουμε ότι αν οι είσοδοι είναι A και B η συνάρτηση στην έξοδο θα είναι $\overline{AB}$ όπως φαίνεται καθαρά στον πίνακα αλήθειας.

NAND:

A	B	out
0	0	1
0	1	1
1	0	1
1	1	0



Παρακάτω βλέπουμε το ολοκληρωμένο κύκλωμα της σειράς **7400 TTL** που θα δούμε παρακάτω αναλυτικά το οποίο περιέχει τέσσερεις πύλες **NAND** δύο εισόδων η κάθε μία. Βλέπουμε ότι το παραπάνω ολοκληρωμένο έχει 14 PIN ή ακροδέκτες από τους οποίους το 7 είναι η γείωση ή GND και το PIN 14 η τροφοδοσία ή Vcc. Με την σύνδεση των παραπάνω PIN όλες οι πύλες τροφοδοτούνται και είναι κατάλληλες να λειτουργήσουν αρκεί να συνδεθούν σωστά.

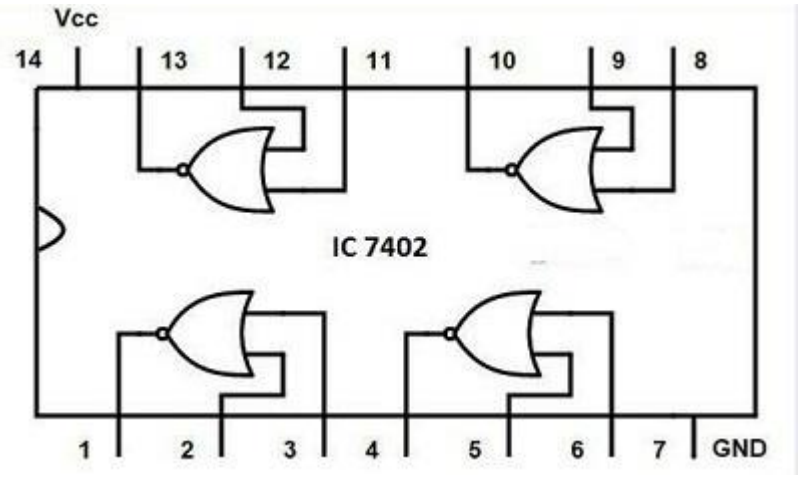


Η πύλη **NOR.** Είναι το συμπλήρωμα της λογικής πράξης **Η**. Βλέπουμε παρακάτω τον συμβολισμό της πύλης όπως επίσης και τον πίνακα αλήθειας της. Βλέπουμε ότι αν οι είσοδοι είναι Α και Β η συνάρτηση στην έξοδο θα είναι $\overline{A+B}$ όπως φαίνεται καθαρά στον πίνακα αλήθειας παρακάτω.

Inputs		Output
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

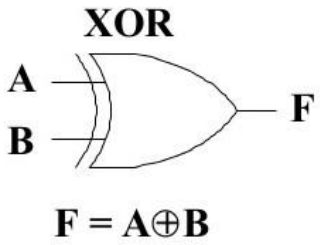
$Y = \overline{A+B}$

Βλέπουμε παρακάτω το ολοκληρωμένο κύκλωμα της σειράς **7402 TTL** που θα το οποίο περιέχει τέσσερεις πύλες **NOR** δύο εισόδων η κάθε μία. Βλέπουμε ότι το παραπάνω ολοκληρωμένο έχει 14 PIN ή ακροδέκτες από τους οποίους το 7 είναι η γείωση ή GND και το PIN 14 η τροφοδοσία ή Vcc. Με την σύνδεση των PIN Vcc και GND όλες οι πύλες τροφοδοτούνται και είναι κατάλληλες να λειτουργήσουν αρκεί να συνδεθούν σωστά.



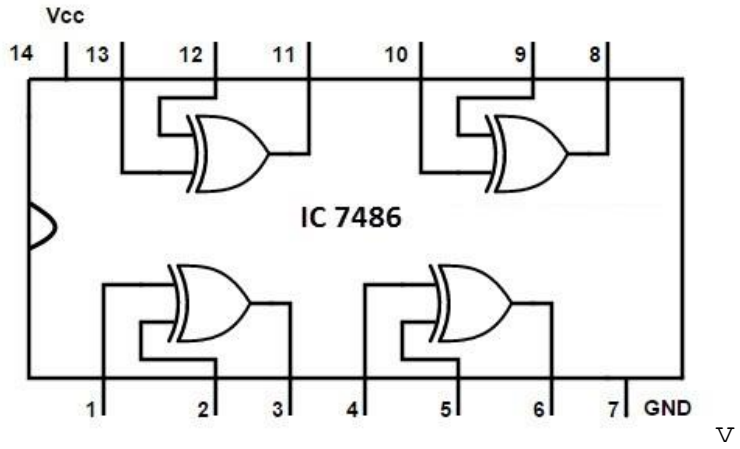
Η πύλη **XOR.** Αν το Α=1 και Β=0 ή αν Α=0 και Β=1 τότε η έξοδος της πύλης XOR είναι ίση με 1. Μπορούμε να πούμε απλούστερα ότι όταν οι είσοδοι μιας XOR είναι διαφορετικές τότε δίνει λογικό 1 στην έξοδο της. Βλέπουμε παρακάτω τον συμβολισμό της πύλης όπως επίσης και τον πίνακα αλήθειας της. Βλέπουμε ότι αν οι είσοδοι είναι Α και Β η συνάρτηση

στην έξοδο θα είναι $\overline{A}B + A\overline{B}$ όπως φαίνεται καθαρά στον πίνακα αλήθειας παρακάτω. Η πύλη XOR συμβολίζεται όπως $F(A,B) = A \oplus B = \overline{A}B + A\overline{B}$.



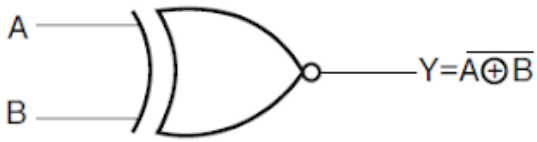
A	B	F=A⊕B
0	0	0
0	1	1
1	0	1
1	1	0

Βλέπουμε παρακάτω το ολοκληρωμένο κύκλωμα της σειράς **7486 TTL** το οποίο περιέχει τέσσερεις πύλες **XOR** δύο εισόδων η κάθε μία. Βλέπουμε ότι το παραπάνω ολοκληρωμένο έχει 14 PIN ή ακροδέκτες από τους οποίους το 7 είναι η γείωση ή GND και το PIN 14 η τροφοδοσία ή Vcc. Με την σύνδεση των PIN Vcc και GND όλες οι πύλες τροφοδοτούνται και είναι κατάλληλες να λειτουργήσουν αρκεί να συνδεθούν σωστά.



Η πύλη **XNOR.** Η παραπάνω πύλη **XNOR** είναι το συμπλήρωμα της πύλης **XOR** που σημαίνει ότι η έξοδος της παίρνει τη λογική τιμή 1 αν το A=0 και το B=0 ή αν A=1 και B=1. Δηλαδή όπως και προηγουμένως η **XNOR** δίνει στην έξοδο της λογικό 1 όταν και μόνο όταν οι είσοδοι είναι ίδιες, δηλαδή αν το A=B=0 ή αν το A=B=1. Αυτό επιβεβαιώνεται όπως βλέπουμε παρακάτω στον συμβολισμό της πύλης και στον πίνακα αλήθειας που την συνοδεύει. Εδώ θα πρέπει επίσης να πούμε ότι η λογική πύλη

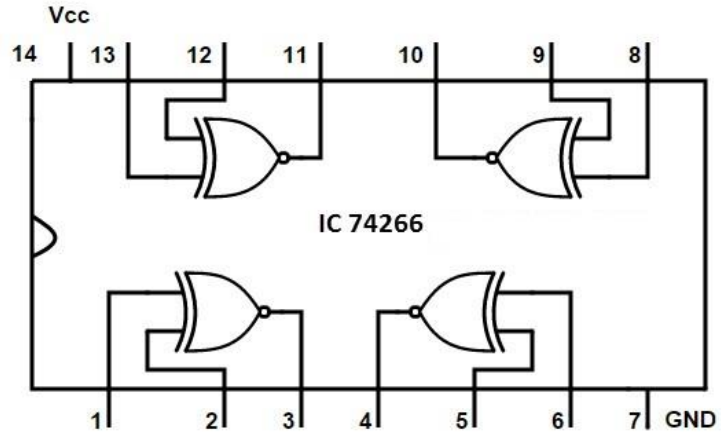
XNOR έχει την παρακάτω έκφραση επίσης αφού είναι το συμπλήρωμα της **XOR** $F(A,B) = A \odot B = \overline{A \oplus B} = \overline{A}\overline{B} + AB$.



$$Y = (\overline{A \oplus B}) = (A.B + \overline{A}.\overline{B})$$

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

Βλέπουμε παρακάτω το ολοκληρωμένο κύκλωμα της σειράς **74266 TTL** το οποίο περιέχει τέσσερεις πύλες **XNOR** δύο εισόδων η κάθε μία. Βλέπουμε ότι το παραπάνω ολοκληρωμένο έχει 14 PIN ή ακροδέκτες από τους οποίους το 7 είναι η γείωση ή GND και το PIN 14 η τροφοδοσία ή Vcc. Με την σύνδεση των PIN Vcc και GND όλες οι πύλες τροφοδοτούνται και είναι κατάλληλες να λειτουργήσουν αρκεί να συνδεθούν σωστά.



Τα λογικά κυκλώματα που υλοποιούν τις παραπάνω λογικές πράξεις λέγονται λογικές πύλες ή Gates στα αγγλικά. Δέχονται στην είσοδο τους λογικά σήματα και παράγουν ανάλογα σήματα στην έξοδο όταν ικανοποιούνται οι κατάλληλες συνθήκες τροφοδοσίας και γείωσης κάθε πύλης. Στην παρακάτω εικόνα φαίνονται οι βασικές πύλες που μόλις περιγράψαμε. Ο αριθμός εισόδων σε κάθε πύλη είναι συνάρτηση του τύπου της πύλης και της λογικής οικογένειας που ανήκει. Υπάρχουν πύλες με περισσότερες των 2 εισόδους όπως για παράδειγμα πύλες NAND

και πύλες NOR με περισσότερες των 2 , δηλαδή 3, 4, ή ακόμη και 5 εισόδων ή και περισσότερων.

Θα δούμε παρακάτω ότι αν έχουμε 2 δυαδικές μεταβλητές x, y μπορούμε να ορίσουμε 16 λογικές συναρτήσεις, από την F0 μέχρι την F15. Αυτό προκύπτει από το $\mathbf{4^2}$ επειδή με δύο μεταβλητές x και y έχουμε 4 διαφορετικούς συνδυασμούς όπως φαίνεται και παρακάτω.

X	Y	F0	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

$F_0 = 0$ Είναι η μηδενική συνάρτηση ή η συνάρτηση μηδέν.

$F_1 = xy$ Είναι η συνάρτηση **AND**.

$F_2 = x\bar{y}$ Είναι η συνάρτηση αποτροπή, δηλαδή x όχι y.

$F_3 = xy + x\bar{y} = x(y + \bar{y}) = x$ Είναι η συνάρτησης μεταφορά επειδή η είσοδος x μεταφέρεται ως είναι και στην έξοδο.

$F_4 = \bar{x}y$ Είναι η συνάρτηση αποτροπή, δηλαδή y όχι x.

$F_5 = \bar{x}y + xy = (\bar{x} + x)y = y$ Είναι η συνάρτησης μεταφορά επειδή η είσοδος y μεταφέρεται ως είναι και στην έξοδο.

$F_6 = \bar{x}y + x\bar{y} = x \oplus y$ Είναι η συνάρτηση **XOR**.

$F_7 = \bar{x}y + x\bar{y} + xy = \bar{x}y + x\bar{y} + xy + xy = x(y + \bar{y}) + y(x + \bar{x}) = x + y$

Είναι η συνάρτηση **OR.** Βλέπουμε για να κάνουμε την απλοποίηση χρησιμοποιώντας την επιμεριστική ιδιότητα προσθέσαμε ένα παράγοντα άλλη μία φορά. Αυτό είναι εφικτό επειδή έχουμε τη σχέση **X+X=X.**

$F_8 = \bar{x}\bar{y} = \overline{x + y}$ Είναι η συνάρτηση **NOR.**

$F_9 = \bar{x}\bar{y} + xy$ Είναι η συνάρτηση **XNOR** ή συνάρτηση x ίσον y.

$F_{10} = \bar{x}\bar{y} + x\bar{y} = (\bar{x} + x)\bar{y} = \bar{y}$ Είναι η συνάρτηση συμπλήρωμα ή διαφορετικά η συνάρτηση όχι y.

$F_{11} = \bar{x}\bar{y} + x\bar{y} + xy = \bar{x}\bar{y} + x\bar{y} + xy + x\bar{y} = (\bar{x} + x)\bar{y} + x(y + \bar{y}) = x + \bar{y}$

$F_{12} = \bar{x}\bar{y} + \bar{x}y = \bar{x}(y + \bar{y}) = \bar{x}$ Είναι η συνάρτηση συμπλήρωμα ή διαφορετικά η συνάρτηση όχι x.

$F_{13} = \bar{x}\bar{y} + \bar{x}y + xy = \bar{x}\bar{y} + \bar{x}y + \bar{x}y + xy = \bar{x}(y + \bar{y}) + y(x + \bar{x}) = \bar{x} + y$

$F_{14} = \bar{x}\bar{y} + \bar{x}y + x\bar{y} = \bar{x}\bar{y} + \bar{x}\bar{y} + \bar{x}y + x\bar{y} = \bar{x}(y + \bar{y}) + y(x + \bar{x}) = \bar{x} + \bar{y} = \overline{xy}$
Είναι η συνάρτηση NAND.

$F_{15} = \bar{x}\bar{y} + \bar{x}y + x\bar{y} + xy = 1$ Είναι η συνάρτηση 1 δηλαδή για κάθε τιμή του x και του y η συνάρτησης είναι ίση με 1.

ΛΟΓΙΚΕΣ ΠΥΛΕΣ ΚΑΙ ΛΟΓΙΚΑ ΚΥΚΛΩΜΑΤΑ

Το σύνολο των σχέσεων που θα χρησιμοποιήσουμε για τη σχεδίαση ψηφιακών λογικών κυκλωμάτων δίνεται παρακάτω:

1) $\bar{0} = 1$	**12) A+A=A**
2) $\bar{1}$**=0**	**13)** $A+\bar{A}=1$
3) A=0 $\bar{A} = 1$	**14) A+B=B+A**
4) A=1 $\bar{A} = 0$	**15) A·B=B.A**
5) $A=\bar{\bar{A}}$	**16) A+(B+C)=(A+B)+C**
6) A·0=0	**17) A·(B·C)=(A·B)·C**
7) A·1=A	**18) A·(B+C)=A·B+A·C**
8) A·A=A	**19) A+B·C=(A+B)·(A+C)**
9) $A\cdot\bar{A}$**=0**	**20) A+A·B=A**
10) A+0=A	**21)** $\overline{A \cdot B} = \bar{A} + \bar{B}$
11) A+1=1	**22)** $\bar{A} \cdot \bar{B} = \overline{A + B}$

Μερικές από τις παραπάνω σχέσεις αποτελούν θεωρήματα (οι άλλες είναι αξιώματα) και μπορούμε να τις αποδείξουμε εύκολα με τη βοήθεια των πινάκων αληθείας.

Παράδειγμα: Το θεώρημα x+xy=x

x	**y**	**x·y**	**x+xy**
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	1

βλέπουμε εύκολα ότι η συνάρτηση x+xy και η συνάρτηση =x είναι ίσες γιατί και οι δύο συναρτήσεις έχουν τον ίδιο πίνακα αλήθειας. Με τον ίδιο τρόπο μπορούμε να αποδείξουμε ότι δύο οιανδήποτε λογικές συναρτήσεις είναι ίσες αντί να κάνουμε αλγεβρικές πράξεις να δείξουμε απλά ότι έχουν ίδιο πίνακα αλήθειας. Είναι μια πολύ σημαντική ιδιότητα η οποία χρησιμοποιείται συχνά.

Κεφάλαιο 3. Πύλες και Λογικά Κυκλώματα

Τα ολοκληρωμένα κυκλώματα ταξινομούνται με δύο διαφορετικούς τρόπους ανάλογα με την πυκνότητα ολοκλήρωσης και ανάλογα με τη βασική ηλεκτρονική διάταξη που χρησιμοποιείται για την υλοποίηση τους.

Α. Η πυκνότητα ολοκλήρωσης διαχωρίζει τα ολοκληρωμένα κυκλώματα σε IC (INTEGRATED CIRCUITS) όπως παρακάτω:

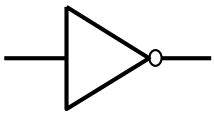
- **SSI** Small Scale Integration μέχρι 10 πύλες το τσιπ.
- **MSI** Medium Scale Integration μέχρι 100 πύλες το τσιπ.
- **LSI** Large Scale Integration μέχρι μερικές χιλιάδες πύλες το τσιπ.
- **VLSI** Very Large Scale Integration περισσότερες από μερικές εκατοντάδες χιλιάδες πύλες το τσιπ.
- **UVLSI** Ultra Very Large Scale Integration περισσότερες από μερικά εκατομμύρια πύλες εκατοντάδες χιλιάδες πύλες το τσιπ.

Β. Κάθε οικογένεια ψηφιακών κυκλωμάτων χρησιμοποιεί μία βασική ηλεκτρονική διάταξη. Ανάλογα με αυτή τη βασική ηλεκτρονική διάταξη διαχωρίζονται όπως παρακάτω.

- **DTL (DIODE TRANSISTOR LOGIC)** Βασική ηλεκτρονική διάταξη σε αυτή την οικογένεια θεωρείται η δίοδος.
- **TTL (TRANSISTOR TRANSISTOR LOGIC)** Βασική ηλεκτρονική διάταξη το διπολικό ΤΡΑΝΖΙΣΤΟΡ.
- **MOS (METAL OXYDE FET)** Βασική ηλεκτρονική διάταξη το τρανζίστορ πεδίου MOSFET. Ανάλογα με τον τύπο του καναλιού που χρησιμοποιείται έχουμε την οικογένεια PMOS και την NMOS, ανάλογα με το αν το κανάλι είναι p τύπου η n τύπου αντίστοιχα.
- **CMOS (COMPLEMENTARY MOS)** Σε αυτή την οικογένεια ολοκληρωμένων κυκλωμάτων συνυπάρχουν στο ίδιο κομμάτι ημιαγωγού και διατάξεις PMOS όπως και διατάξεις NMOS.

Όλες οι συναρτήσεις BOOLE μπορούν να υλοποιηθούν με τη βοήθεια των τριών βασικών πυλών NOT, AND και OR.

NOT

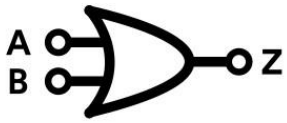


74LS04

AND gate

A	B	Z
0	0	0
0	1	0
1	0	0
1	1	1

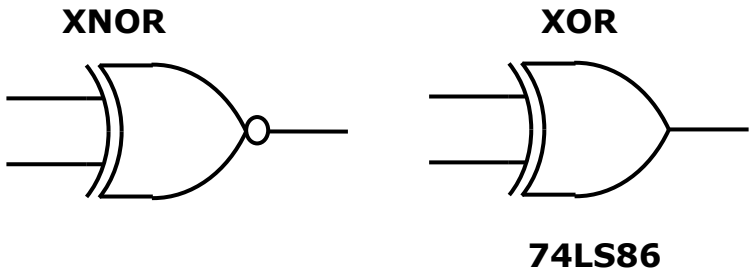
OR gate



A	B	Z
0	0	0
0	1	1
1	0	1
1	1	1

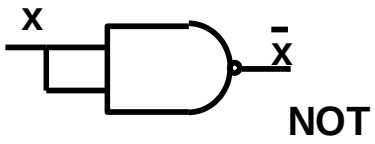
Το σύστημα των τριών αυτών πυλών ονομάζεται λογικό συμπληρωμένο σύστημα. Η υλοποίηση όλων των λογικών συναρτήσεων είναι δυνατή με τις προηγούμενες τρεις πύλες, η με ένα άλλο σύνολο πυλών που υλοποιούν τις προηγούμενες υποχρεωτικά. Σε αντίθεση με τα παραπάνω η πύλη **NAND** ή **ΟΧΙ-ΚΑΙ** αποτελεί από μόνη της ένα λογικό συμπληρωμένο σύστημα γιατί όπως θα δούμε παρακάτω μπορεί να υλοποιήσει όλες τις βασικές πύλες. Το ίδιο ισχύει και για την λογική πύλη **NOR** ή **ΟΧΙ-Η** δηλαδή αποτελεί μόνη της ένα λογικό συμπληρωμένο σύστημα και υλοποιεί όλες τις βασικές λογικές πύλες. Κατά συνέπεια αφού υλοποιούνται όλες οι λογικές πύλες μπορούν να υλοποιηθούν όλες οι συναρτήσεις BOOLE δηλαδή όλα τα ψηφιακά κυκλώματα. Μπορούμε να δούμε παρακάτω πώς σχηματίζονται όλες οι λογικές πύλες αποκλειστικά με πύλες NAND και στη συνέχεια αποκλειστικά με πύλες NOR. Δηλαδή χρησιμοποιώντας λογικές πύλες NAND και μετά NOR θα υλοποιήσουμε τον αντιστροφέα ή την πύλη NOT, την πύλη AND, την OR, την πύλη NOR και την πύλη XOR όπως επίσης και την πύλη XNOR. Οι δύο τελευταίες πύλες είναι συμπληρωματικές, η μία δηλαδή είναι συμπλήρωμα της άλλης Θα λέμε σε

αυτή τη περίπτωση ότι η πύλη NAND και η πύλη NOR είναι δύο πύλες οικουμενικές.

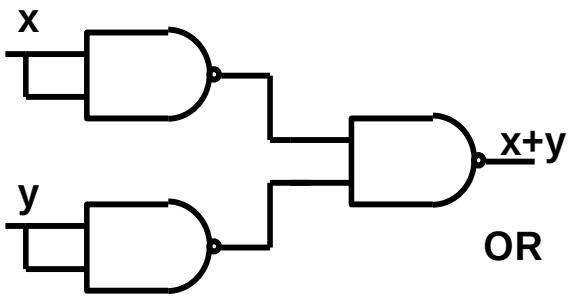


Οικουμενικότητα της πύλης NAND

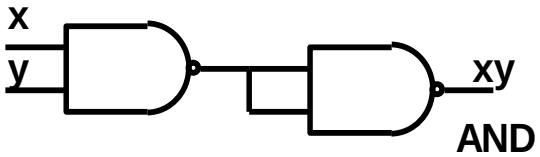
Θα δούμε παρακάτω ότι αποκλειστικά με πύλες NAND μπορούμε να φτιάξουμε όλες τις άλλες πύλες.



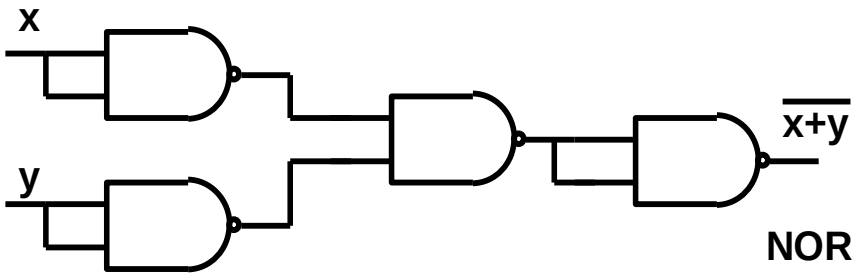
Βραχυκυκλώνοντας τις 2 εισόδους μιας πύλης NAND υλοποιούμε ένα αντιστροφέα.



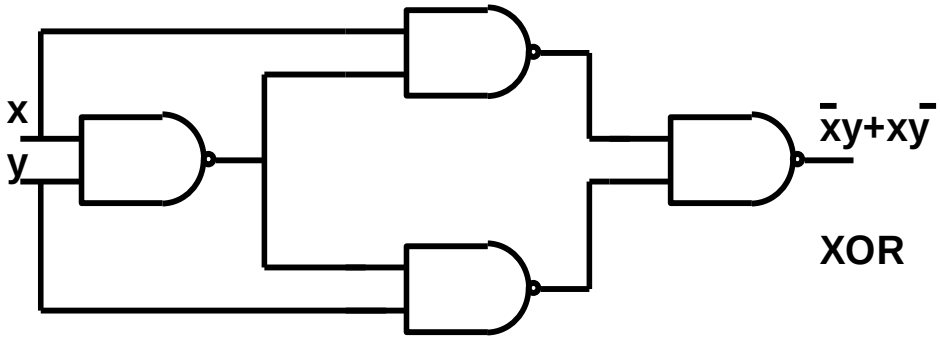
Υλοποίηση της λογικής πράξης OR με πύλες NAND



Υλοποίηση της λογικής πράξης AND με πύλες NAND



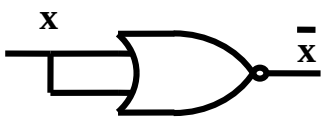
Υλοποίηση της λογικής πράξης NOR με πύλες NAND



Υλοποίηση της λογικής πράξης XOR με πύλες NAND. Στο παραπάνω κύκλωμα αν συνδέσουμε στην έξοδο μια βραχυκυκλωμένη NOR θα πάρουμε μια πύλη XNOR.

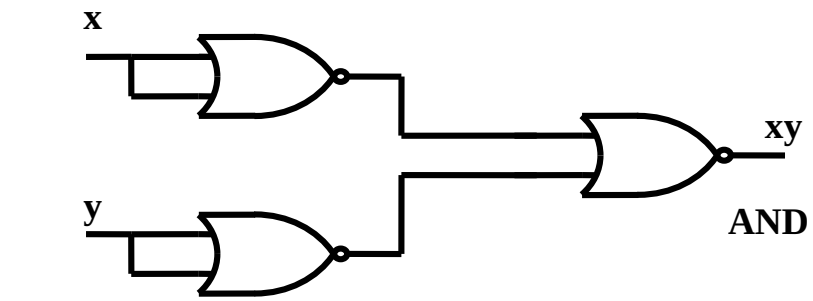
Υλοποίηση των βασικών πυλών αποκλειστικά με πύλες NOR

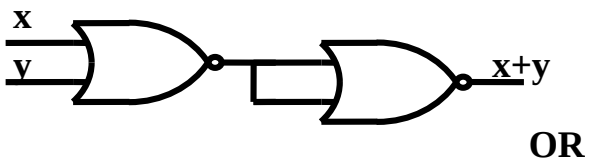
Ομοίως όπως φαίνεται στα παρακάτω κυκλώματα μπορούμε με ανάλογο τρόπο να υλοποιήσουμε τις λογικές πύλες στο σύνολο τους αποκλειστικά με πύλες NOR.

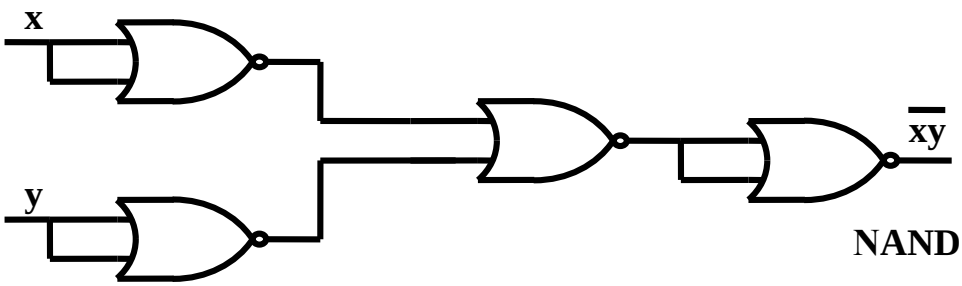


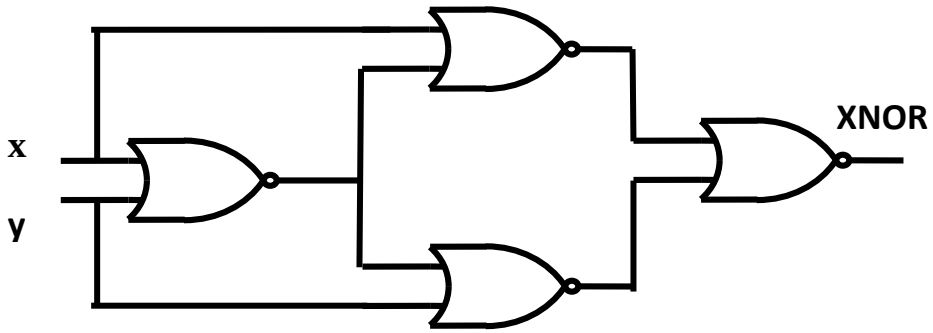
NOT

Στο παραπάνω κύκλωμα αν συνδέσουμε στην έξοδο μια βραχυκυκλωμένη NOR θα πάρουμε μια πύλη XOR.









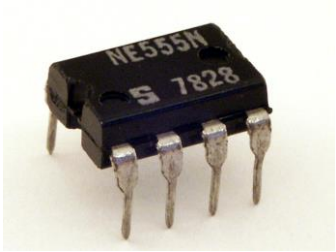
Ισοδύναμες πύλες. Θα συναντήσετε πολλές φορές τις παρακάτω πύλες οι οποίες παράγουν ισοδύναμα αποτελέσματα.

Τεχνικά χαρακτηριστικά των πυλών

Τα κυκλώματα τα οποία θα χρησιμοποιήσουμε για το εργαστηριακό κομμάτι του μαθήματος ονομάζονται ολοκληρωμένα κυκλώματα. Η συντομογραφία στα Αγγλικά είναι IC (Integrated Circuit) ενώ στα Ελληνικά το λέμε τσιπ. Ένα ολοκληρωμένο κύκλωμα είναι ένα κύκλωμα με πολλές προδιαγραφές με πάρα πολύ μικρές διαστάσεις και κατανάλωση. Κατασκευάζεται πάνω σε ένα κομμάτι ημιαγώγιμου υλικού, πάχους (200-300) μm με πάρα πολύ περίπλοκες τεχνικές οι οποίες βελτιώνονται συνεχώς με την πάροδο του χρόνου με αποτέλεσμα όλο μικρότερα κυκλώματα με βελτιωμένα χαρακτηριστικά.

Πάνω λοιπόν σε ένα κομμάτι ημιαγωγού κατασκευάζονται και ενεργά και παθητικά στοιχεία. Τα ενεργά στοιχεία μπορεί να είναι τρανζίστορ, δίοδοι, και τα παθητικά, αντιστάσεις πυκνωτές, πηνία κ.λ.π. Οι επιφάνεια του ημιαγωγού μπορεί να έχει διάμετρο από 8 μέχρι 15 εκατοστά και πάνω σε αυτή την επιφάνεια κατασκευάζονται ταυτόχρονα πολλά ίδια κυκλώματα. Στην συνέχεια τεμαχίζονται, τοποθετούνται σε διάφορες συσκευασίες, γίνονται οι συνδέσεις και είναι έτοιμα για χρήση αφού περάσουν ένα σύντομο ποιοτικό έλεγχο. Μερικές από τις συσκευασίες που χρησιμοποιούνται φαίνονται παρακάτω:

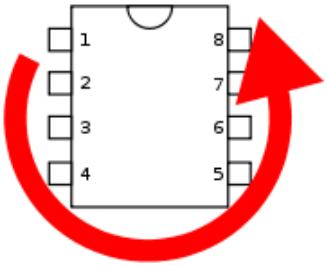
1. Συσκευασία DIP (Dual In Line Package). Σε αυτή τη συσκευασία που βλέπουμε παρακάτω υπάρχει μια εγκοπή η οποία φαίνεται αριστερά στο παρακάτω τσιπ και μας επιτρέπει να γνωρίζουμε την θέση του κάθε ακροδέκτη.



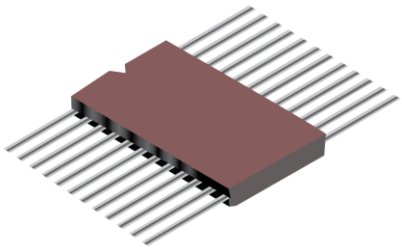
Συσκευασία DIP (Dual In Line Package)

Αν τοποθετήσουμε το τσιπ όπως φαίνεται στο παρακάτω διάγραμμα η αρίθμηση των ακροδεκτών γίνεται με τη φορά που δείχνει το βέλος. Κάθε

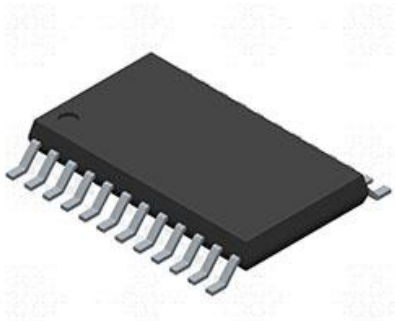
ακροδέκτης έχει συγκεκριμένο προορισμό και λειτουργία και δίνεται αναλυτικά από τον κατασκευαστή.



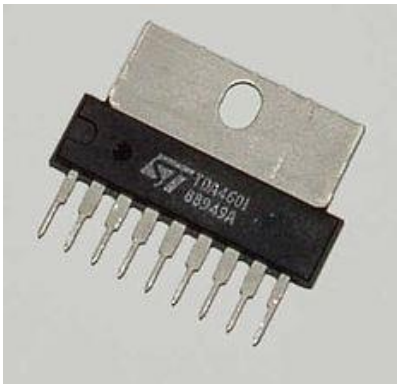
2. Επίπεδη συσκευασία. Σε αυτή τη συσκευασία που βλέπουμε παρακάτω το τσιπ και οι ακροδέκτες είναι στο ίδιο επίπεδο. Μια εγκοπή πάλι μας επιτρέπει να προσανατολίσουμε το παραπάνω ηλεκτρονικό εξάρτημα για να γνωρίζουμε με ακρίβεια την θέση των ακροδεκτών. Ο ακροδέκτης ή PIN που είναι ακριβώς κάτω από την εγκοπή είναι ο 1, δίπλα ο 2 κ.λ.π. όπως και προηγουμένως.



3. Συσκευασία επιφανείας SMT(Surface Mount Technology). Βλέπουμε ένα βαθούλωμα που επιτρέπει τον προσανατολισμό του τσιπ. Ακριβώς κάτω από αυτό το σημείο είναι το PIN 1.



4. Συσκευασία SIL (Single In Line) ένα δείγμα φαίνεται παρακάτω.



Υπάρχουν και άλλες περίπλοκες συσκευασίες που αφορούν εξελιγμένες δομές κατασκευής πολυσύνθετων ολοκληρωμένων κυκλωμάτων όπως είναι αυτές των επεξεργαστών, με εκατομμύρια παθητικά και ενεργά στοιχεία.

Σε κάθε περίπτωση για την επιλογή της συσκευασίας απαιτούνται ειδικές τεχνικές σχεδιασμού και είναι σημαντικό τα υλικά που χρησιμοποιούνται ως ηλεκτρικές επαφές και καλωδιώσεις να παρουσιάζουν άριστα χαρακτηριστικά όπως χαμηλή αντίσταση, χαμηλή χωρητικότητα και χαμηλή αυτεπαγωγή. Η δομή όσο και τα υλικά της συσκευασίας πρέπει να δίνουν προτεραιότητα στη σωστή μεταφορά των ηλεκτρικών σημάτων, ελαχιστοποιώντας παράλληλα τυχόν παρασιτικά στοιχεία που θα μπορούσαν να επηρεάσουν αρνητικά τα μεταφερόμενα στον έξω κόσμο σήματα. Ο έλεγχος αυτών των χαρακτηριστικών γίνεται όλο και πιο σημαντικός καθώς η υπόλοιπη τεχνολογία επιταχύνεται, βελτιώνεται και αλλάζει. Η συσκευασία ενός κυκλώματος είναι υπεύθυνη για τη διατήρηση της ασφάλειας του από κάθε είδους πιθανή ζημιά. Η συσκευασία πρέπει να αντιστέκεται στη φυσική θραύση, να παρέχει αεροστεγή σφράγιση για να διατηρεί την υγρασία και επίσης να παρέχει αποτελεσματική απαγωγή θερμότητας από το τσιπ. Τα υλικά που χρησιμοποιούνται για το σώμα της συσκευασίας είναι συνήθως είτε πλαστικά (θερμοσκληρυνόμενα είτε θερμοπλαστικά) ή κεραμικά. Μόλις εγκατασταθεί, το πλαστικό σκληρύνεται και σε συγκεκριμένη θερμοκρασία. Και οι δύο επιλογές μπορούν να προσφέρουν υψηλή θερμική αγωγιμότητα και αξιοπρεπή μηχανική αντοχή. Το κεραμικό έχει γενικά προτιμώμενα χαρακτηριστικά, αλλά είναι πιο ακριβό. Η αύξηση της επιφάνειας της συσκευασίας επιτρέπει καλύτερη

μεταφορά θερμότητας, έτσι καμιά φορά γίνεται χρήση μεταλλικών πτερυγίων για τη βελτίωση της μεταφοράς θερμότητας.

Το κόστος είναι ένας σημαντικός περιοριστικός παράγοντας. Επιλογές του υλικού συσκευασίας και το επίπεδο λειτουργίας πρέπει να εξισορροπούνται από την οικονομική βιωσιμότητα του τελικού προϊόντος. Ανάλογα με τις ανάγκες, η επιλογή υλικών χαμηλότερου κόστους είναι συχνά μια αποδεκτή λύση μπροστά στους οικονομικούς περιορισμούς. Συνήθως, μια φθηνή πλαστική συσκευασία μπορεί να αντέξει θερμότητα έως και 2W, η οποία επαρκεί για πολλές απλές εφαρμογές, αν και μια παρόμοια κεραμική συσκευασία μπορεί να αντέξει έως και 50W. Στόχος πάντα παραμένει η αποτελεσματικότερη απαγωγή θερμότητας, με συνέπεια το κόστος της συσκευασίας να αυξάνεται μαζί με αυτό.

Σημαντικά Χαρακτηριστικά των ICs

Τα σημαντικότερα τεχνικά χαρακτηριστικά μιας οικογένειας ολοκληρωμένων κυκλωμάτων τα οποία θα μας οδηγήσουν να κάνουμε μία συγκεκριμένη επιλογή όταν σχεδιάσουμε ένα ψηφιακό κύκλωμα είναι τα παρακάτω:

- **ταχύτητα**
- **κατανάλωση**
- **Fan-out**

Ταχύτητα

Τα λογικά ολοκληρωμένα κυκλώματα λειτουργούν σε μία μεγάλη κλίμακα συχνοτήτων. Η συχνότητα αυτή είναι πεπερασμένη. Τα στάνταρ TTL ολοκληρωμένα κυκλώματα μπορούν να λειτουργήσουν μέχρι μερικά MHz (μέγιστη συχνότητα), ενώ τα ολοκληρωμένα ECL που θα περιγράψουμε παρακάτω αναλυτικά μπορούν να λειτουργήσουν μέχρι και συχνότητες της τάξης του GHz.

Η καθυστέρηση κάθε πύλης καθορίζει τη μέγιστη συχνότητα λειτουργίας των ολοκληρωμένων κυκλωμάτων. Έτσι θα πρέπει σε ένα περίπλοκο ψηφιακό κύκλωμα η περίοδος του ρολογιού να είναι αρκετά μεγάλη για να προλαβαίνουν όλα τα σήματα να φθάσουν στον προορισμό τους. Το περίπλοκο αυτό κύκλωμα πρέπει να δεχθεί τις μεταβολές που εμείς έχουμε

προγραμματίσει στην διάρκεια ενός παλμού πριν έρθει ο επόμενος. Αυτό συμβαίνει όταν το κύκλωμα χρονίζεται από ένα εξωτερικό ρολόι που λειτουργεί σε μια δεδομένη συχνότητα.

Αυτό που πρέπει επίσης να ξέρουμε είναι ότι: όταν κερδίζουμε σε συχνότητα χάνουμε σε κατανάλωση, και έτσι τα κυκλώματα με τις μέγιστες συχνότητες λειτουργίας μίας δεδομένης οικογένειας έχουν τη μέγιστη κατανάλωση.

Κατανάλωση

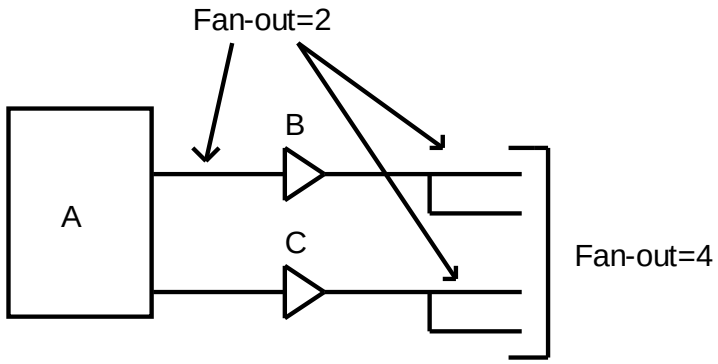
Σε μερικές περιπτώσεις η κατανάλωση ισχύος είναι ένας πολύ σημαντικός παράγοντας. Λόγω της κατανάλωσης ηλεκτρικής ενέργειας από κάθε επιμέρους ηλεκτρονικό κύκλωμα έχουμε αύξηση της θερμοκρασίας. Η κατανάλωση μίας πύλης χωρίζεται σε δύο κατηγορίες :

- στατική κατανάλωση
- δυναμική κατανάλωση

Η στατική κατανάλωση είναι η αναγκαία κατανάλωση για να παραμείνει σε μία συγκεκριμένη λογική κατάσταση, ενώ η δυναμική κατανάλωση είναι αυτή που χρειάζεται για να αλλάξει κατάσταση. Αν θεωρήσουμε όλα τα κυκλώματα των λογικών πυλών σαν μικρούς ηλεκτρικούς καταναλωτές η συνολική ισχύς που καταναλώνει ένα περίπλοκο λογικό κύκλωμα θα είναι το άθροισμα των καταναλώσεων όλων των επιμέρους κυκλωμάτων που το συνθέτουν.

Fan-Out

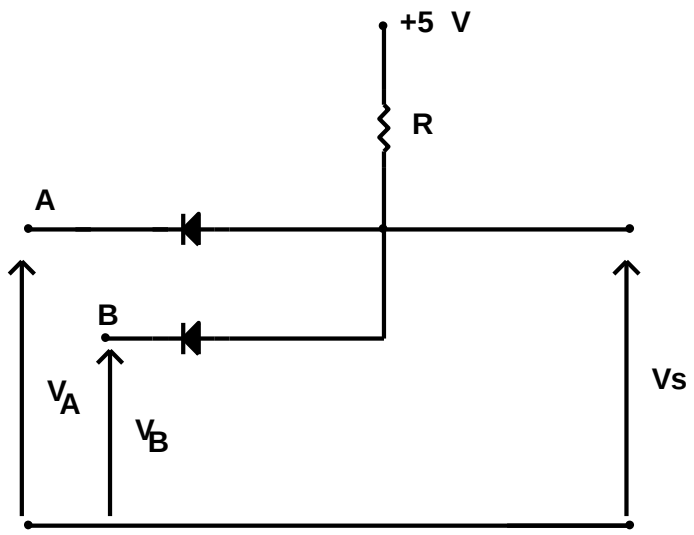
Η έξοδος μίας πύλης μπορεί να δώσει ένα περιορισμένο ρεύμα, και κατά συνέπεια να αποδώσει μια περιορισμένη ηλεκτρική ισχύ. Έτσι μία έξοδος μπορεί να οδηγήσει ένα περιορισμένο, αριθμό πυλών, ή ένα συγκεκριμένο αριθμό καταναλωτών. Ο μέγιστος αριθμός πυλών που μπορεί να οδηγήσει μία έξοδος της ίδιας οικογένειας λέγεται Fan-Out. Αν έχουμε μία οικογένεια με περιορισμένο Fan-Out συνδέουμε την έξοδο σε bufers όπως φαίνεται στο παρακάτω σχήμα και πολλαπλασιάζεται το Fan-Out.



Η πύλη Α έχει Fan-Out 2. Έτσι τη συνδέουμε σε δύο BuFlip-Flopers (B,C) και αυξάνουμε το Fan-Out σε 4 στην έξοδο των BuFlip-Flopers.

Λογικά κυκλώματα διόδων

Υποθέτουμε ότι έχουμε δύο διόδους τις οποίες θεωρούμε ιδανικές (δηλαδή Rd=0 Ω δυναμική αντίσταση της διόδου ίση με 0Ω) και οι οποίες τροφοδοτούνται μέσω μιας μικρής αντίστασης για να περιορίσουμε το ρεύμα με μία τάση 5 V όπως φαίνεται στο παρακάτω κύκλωμα. Βλέπουμε ότι και οι δύο δίοδοι είναι ορθά πολωμένες. Δηλαδή το δυναμικό στην περιοχή P είναι μεγαλύτερο από αυτό της περιοχής N. Η διάταξη είναι δίοδος και έχουμε πολικότητα. Στους άλλους δύο πόλους των διόδων εφαρμόζουμε δύο διαφορετικές τάσεις V_A και V_B.



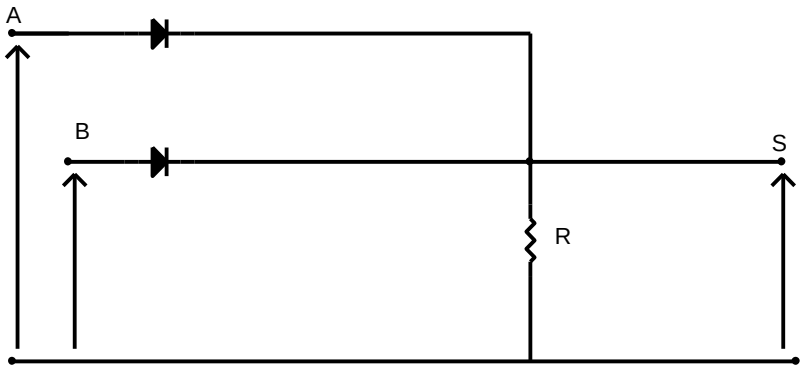
Εάν μία από τις δύο εισόδους είναι στο δυναμικό μηδέν τότε η έξοδος είναι στο δυναμικό μηδέν Vs=0. Όταν συμβεί αυτό το αντίστοιχο δικτύωμα

διόδου άγει και αφού η αντίσταση της είναι μηδέν η δίοδος είναι ιδανική τα 0 Volts μεταφέρονται και στην έξοδο.

Η έξοδος του κυκλώματος είναι στο δυναμικό 5V μόνο και μόνο όταν και οι δύο είσοδοι είναι στο δυναμικό 5V. Είναι η λογική πράξη AND όπως επιβεβαιώνει ο παρακάτω πίνακας αλήθειας που προκύπτει από τη λειτουργία του κυκλώματος.

V_A	V_B	V_S
0 V	0 V	0 V
0 V	5 V	0 V
5 V	0 V	0 V
5 V	5 V	5 V

Το παρακάτω κύκλωμα μας δίνει τη συνάρτηση OR, γιατί αρκεί μία είσοδος να είναι στο δυναμικό 5V,το οποίο αντιστοιχεί στο λογικό 1 (δηλαδή στο HIGH) για να έχω την έξοδο στο HIGH



Η παραπάνω οικογένεια λογικών κυκλωμάτων δεν είναι ένα συμπληρωμένο λογικό σύστημα γιατί μόνο με διόδους είναι αδύνατο να υλοποιηθεί η αντιστροφή δηλαδή η πύλη NOT.

Ας θεωρήσουμε μία πύλη AND με διόδους όπως αυτές που είδαμε προηγουμένως. Εάν μία είσοδος είναι στο λογικό 1 κανένα ρεύμα δεν περνάει σε αυτή την είσοδο. Αντίθετα με το λογικό 0 στην είσοδο ένα ρεύμα υποχρεωτικά πρέπει να κυκλοφορεί από αυτή την είσοδο λόγω της διαφοράς δυναμικού. Βλέπουμε λοιπόν ότι κατά την υλοποίηση μιας συγκεκριμένης συναρτήσεως **BOOLE** πολλές πύλες μπορούν να συνδεθούν στην σειρά και υπάρχει περίπτωση κάποια στιγμή να αλλοιωθούν οι

χαρακτηριστικές τιμές τάσης πού αντιστοιχούν στο λογικό 0 και στο λογικό 1. Έτσι όπως θα δούμε και παρακάτω κάθε οικογένεια ολοκληρωμένων κυκλωμάτων χαρακτηρίζεται από ένα αριθμό τον οποίο ονομάζουμε Fan-out και ο οποίος δηλώνει το μέγιστο αριθμό πυλών που μπορεί να οδηγήσει έξοδος ενός IC χωρίς να δημιουργείτε πρόβλημα στη λειτουργία του κυκλώματος.

Η οικογένεια RTL (Resistor- Transistor-Logic)

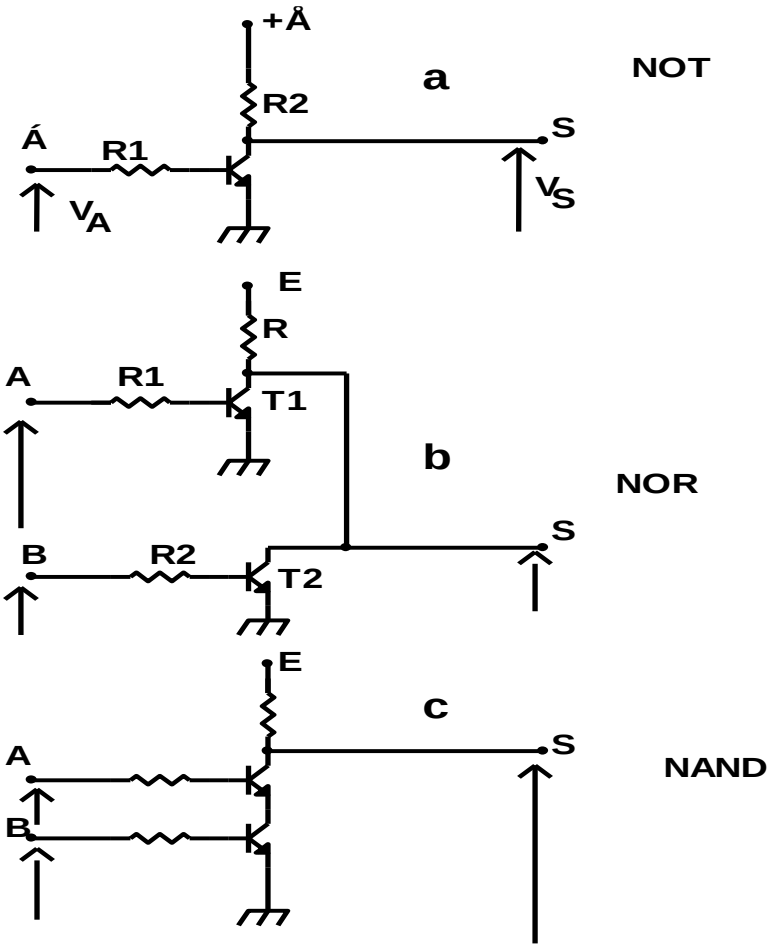
Το τρανζίστορ επιτρέπει πολύ εύκολα να φτιάξουμε την πύλη NOT όπως φαίνεται παρακάτω (κύκλωμα a). Αν V_A=0 το τρανζίστορ Τ είναι στην αποκοπή δηλαδή δεν οδηγεί και κατά συνέπεια η έξοδος θα είναι στα 5 Volts Vs=+E και η έξοδος S είναι στο λογικό 1. Αν V_A= E με την προϋπόθεση ότι η συνθήκη κόρου του τρανζίστορ $R_2>R_1/\beta$ ισχύει το τρανζίστορ είναι στον κόρο. Vs=0 και η έξοδος S=0. Βλέπουμε ότι η έξοδος είναι το συμπλήρωμα της εισόδου, δηλαδή είναι η πύλη NOT.

Στο κύκλωμα b βλέπουμε ότι αν $\mathbf{V_A=V_B=0}$ τα δύο τρανζίστορ είναι στην αποκοπή και κατά συνέπεια η έξοδος S δίνει το λογικό 1. Αντίθετα αν μία από τις εισόδους πάρει την τιμή +E δηλαδή το λογικό 1 το αντίστοιχο τρανζίστορ πηγαίνει στον κόρο και έχουμε Vs=0.

Ο πίνακας αλήθειας του παραπάνω κυκλώματος είναι η πύλη NOR.

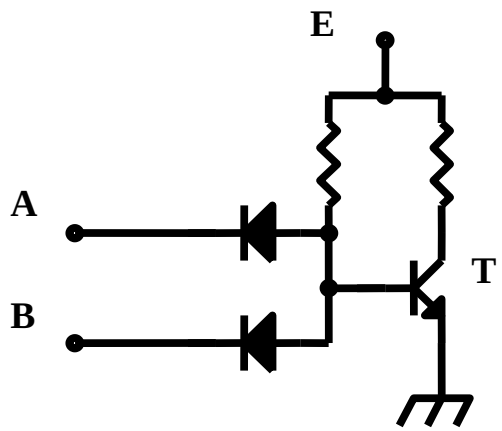
A	B	S
0	0	1
0	1	0
1	0	0
1	1	0

Το κύκλωμα c είναι ένα κύκλωμα το οποίο υλοποιεί τη συνάρτηση NAND αλλά χρησιμοποιείται ελάχιστα λόγω της κακής σύζευξης που έχουν τα δύο τρανζίστορ στην είσοδο. Έτσι η βασική πύλη της οικογένειας RTL παραμένει η πύλη NOR που είδαμε προηγουμένως αφού μόνη της σχηματίζει ένα συμπληρωμένο λογικό σύστημα.



Η οικογένεια DTL (Diode Transistor Logic)

Η οικογένεια αυτή σχηματίζεται από τη λογική οικογένεια διόδων και το διπολικό τρανζίστορ για την αντιστροφή. Το βασικό λογικό κύκλωμα είναι η πύλη NAND και σχηματίζεται όπως φαίνεται στο παρακάτω σχήμα.

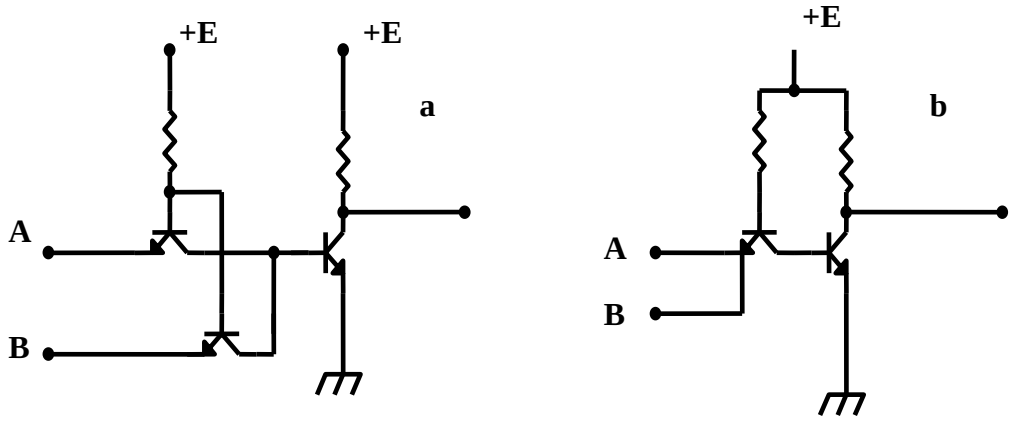


NAND DTL πύλη

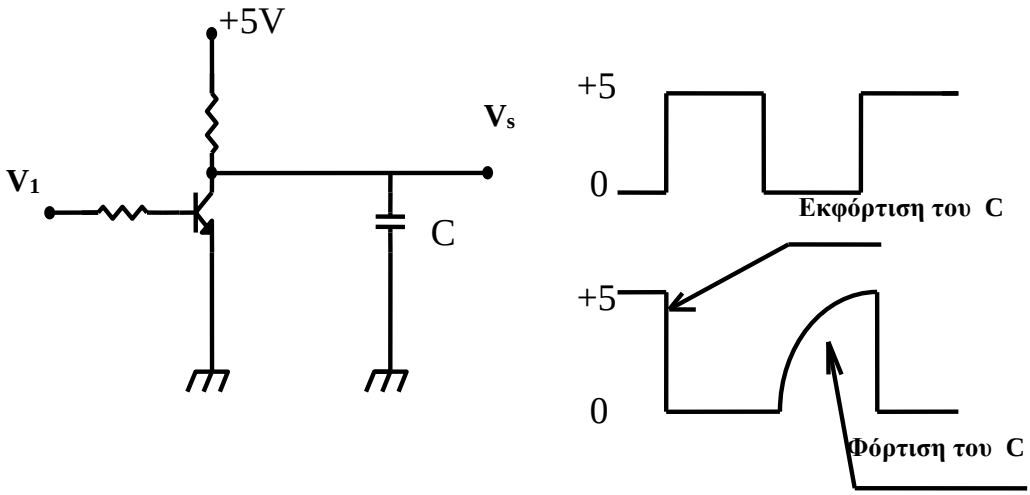
Αν $V_A=V_B=E$ τότε οι δύο δίοδοι είναι στην αποκοπή και το τρανζίστορ Τ είναι στον κόρο δηλαδή Vs=0. Αν $V_A=0$ η δίοδος στην είσοδο μπλοκάρει το τρανζίστορ και έχουμε στην έξοδο τάση μηδέν.

Οικογένεια TTL (Transistor Transistor Logic)

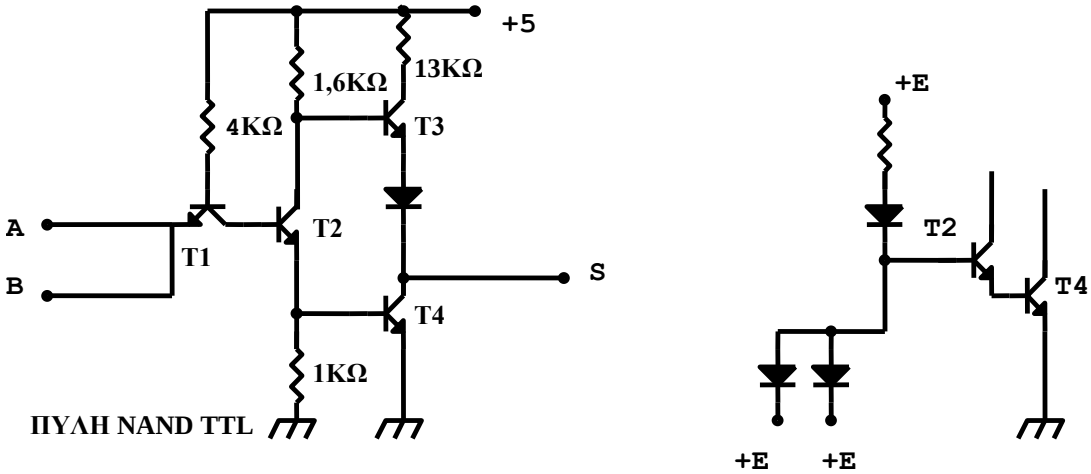
Η οικογένεια TTL είναι μία πολύ σημαντική οικογένεια ολοκληρωμένων κυκλωμάτων και θα προσπαθήσουμε να την αναλύσουμε. Δε διαφέρει πολύ από την DTL. Απλά οι δίοδοι στο κύκλωμα εισόδου έχουν αντικατασταθεί από ένα διπολικό τρανζίστορ με πολλούς εκπομπούς. Η οικογένεια TTL με τις σειρές 54 & 74 είναι η πιο διαδεδομένη. Βασική της πύλη η NAND. Το παρακάτω σχήμα δείχνει την αντικατάσταση των διόδων με τις διόδους εκπομπού βάσης του τρανζίστορ και στην συνέχεια το επόμενο σχήμα πως τα δύο τρανζίστορ της εισόδου μπορούν να αντικατασταθούν με ένα τρανζίστορ με πολλούς εκπομπούς. Το πρόβλημα είναι καθαρά κατασκευαστικό. Για να αυξήσουμε τις επιδόσεις του παραπάνω κυκλώματος κοιτάζουμε το κύκλωμα στην έξοδο το οποίο δεν έχει καλή προσαρμογή στην έξοδο αν το φορτίο είναι χωρητικότητα.



Για να καταλάβουμε τι γίνεται στην έξοδο της πύλης NAND αν για φορτίο συνδέσουμε μία χωρητικότητα ας παρατηρήσουμε τα παρακάτω σχήματα. Όταν η έξοδος πηγαίνει από +Ε στο 0 το ρεύμα εκφόρτωσης του πυκνωτή διασχίζει το διπολικό τρανζίστορ.



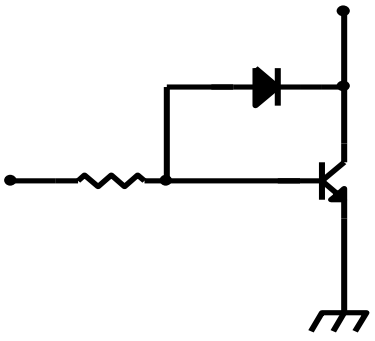
Όταν η έξοδος περνάει από το 0 στο +Ε το τρανζίστορ είναι στην αποκοπή και κατά συνέπεια η φόρτιση του πυκνωτή πρέπει να γίνει διά μέσου της αντίστασης R άρα με μία σταθερά χρόνου RC σημαντική. Για να μειώσουμε την σταθερά χρόνου αρκεί να μειώσουμε την αντίσταση R τότε όμως η κατανάλωση του κυκλώματος αυξάνεται ανάλογα. Έτσι για να αυξήσουμε την ταχύτητα ένα δεύτερο τρανζίστορ παίρνει τη θέση της R. Έτσι όταν το τρανζίστορ αυτό βρεθεί στον κόρο συνδέει απ' ευθείας τη χωρητικότητα στο +Ε. Το κύκλωμα έχει μοιάζει με το κύκλωμα push-pull όπως φαίνεται στο παρακάτω σχήμα.



Αν $V_A=V_B=+5V$, οι δίοδοι εκπομπού βάσης του τρανζίστορ Τ1 είναι στην αποκοπή και αντίθετα η δίοδος βάσης συλλέκτη οδηγεί. Ένα ρεύμα περνάει όπως φαίνεται στο παρακάτω σχήμα και τα τρανζίστορ Τ2 και Τ4

είναι στον κόρο. Όταν το τρανζίστορ Τ2 είναι στον κόρο η τάση του συλλέκτη είναι ίδια με αυτή του εκπομπού δηλαδή περίπου 0,6V. Το τρανζίστορ Τ3 δεν άγει εξ αιτίας της διόδου που υπάρχει μετά τον εκπομπό του. Μόνο αν η βάση του πολωθεί με τάση μεγαλύτερη από 1.2 Volts. Έχουμε λοιπόν ότι Vs=0. Αν τώρα μηδενίσουμε μία από τις εισόδους Α η Β η και τις δύο η αντίσταση των 4 ΚΩ εγγυάται ότι το τρανζίστορ Τ1 θα είναι στον κόρο και έτσι το δυναμικό της βάσης του Τ2 είναι μηδέν. Αυτό σημαίνει ότι το τρανζίστορ Τ2 όπως επίσης και το Τ4 θα είναι στην αποκοπή και ότι το Τ3 θα είναι στον κόρο εξ αιτίας της αντίστασης των 1.6ΚΩ η οποία συνδέει τη βάση με την τάση +Ε. Η έξοδος θα είναι στο επίπεδο High δηλαδή το παραπάνω κύκλωμα είναι η πύλη NAND. $S= \overline{AB}$

Το προηγούμενο κύκλωμα με δύο τρανζίστορ στην έξοδο Τ3 και Τ4 ονομάζεται **"totem pole"** και επιτρέπει γρήγορες μεταβάσεις στην έξοδο ακόμα και πάνω σε ένα φορτίο τύπου χωρητικότητας. Ο χρόνος μετάβασης είναι της τάξης των 10ns. Μέσα στην οικογένεια TTL υπάρχουν διάφορες υποοικογένειες και κάθε μία έχει μία ιδιαιτερότητα. Στην πύλη NAND που μόλις περιγράψαμε τα τρανζίστορ δουλεύουν σαν διακόπτες. Αυτό σημαίνει ότι μερικές φορές είναι στον κόρο. Ένα τρανζίστορ όταν είναι στον κόρο μαζεύει φορτία στην βάση του τα οποία πρέπει να φύγουν στην συνέχεια και αυτή είναι η πρωταρχική αιτία που περιορίζει την ταχύτητα του τρανζίστορ στο να αλλάζει κατάσταση σαν διακόπτης δηλαδή να οδηγεί και η να είναι στην αποκοπή. Για να αυξήσουμε την ταχύτητα πρέπει να αποφύγουμε τον κόρο και αυτό είναι δυνατόν αν βάλουμε μία δίοδο παράλληλα στην βάση-συλλέκτη του τρανζίστορ έτσι ώστε το δυναμικό του συλλέκτη να είναι κατά λίγο μικρότερο από αυτό της βάσης.



Στην πραγματικότητα η αύξηση της ταχύτητας δεν είναι και πολύ μεγάλη γιατί και η δίοδος συγκεντρώνει φορτία. Η λύση δίνεται με την αντικατάσταση της κλασσικής διόδου με μία δίοδο Schottky η οποία δε συγκεντρώνει φορτία και είναι πολύ γρήγορη. Με αυτό τον τρόπο φτιάξαμε την υποοικογένεια TTL-S την TTL Schottky. Αν τώρα αντί για μικρές αντιστάσεις στο βασικό κύκλωμα NAND χρησιμοποιήσουμε μεγαλύτερες της τάξης των 20ΚΩ θα έχουμε μία σημαντική μείωση της κατανάλωσης και θα έχουμε την TTL Schottky με χαμηλή κατανάλωση η την Low Power Schottky TTL-LS. Ο παρακάτω πίνακας συνοψίζει τα βασικά χαρακτηριστικά των δύο υποοικογενειών TTL.

Οικογένεια και Κύκλωμα	**Τυπικός χρόνος καθυστέρησης**	**Κατανάλωση ανά πύλη**	**Ρεύμα εισόδου στο LOW**	**Ρεύμα max εξόδου στο LOW**
TTL 7400	22 ns	2mA	-1.6mA	16mA
TTL-S 7400	4.5 ns	3.75mA	-2mA	20mA
TTL-LS 74LS00	20 ns	0.4mA	0.4mA	4mA

Βασικές οικογένειες TTL

Βλέπουμε ότι η TTL Schottky έχει τη μεγαλύτερη κατανάλωση είναι όμως η ταχύτερη οικογένεια και δε διαφέρει κατά πολύ από την αρχική TTL την κανονική.

Έξοδος **"open collector"**. Είναι μία παραλλαγή της TTL μόνο ως προς την έξοδο επειδή η αντίσταση του συλλέκτη δεν έχει ολοκληρωθεί στο αρχικό τσιπ και πρέπει να τη βάλει ο ίδιος ο χρήστης.

Με αυτό τον τρόπο ορισμένα τσιπ έχουν τη δυνατότητα να περιέχουν από την κατασκευή τους τρανζίστορ στην έξοδο τα οποία μπορεί να αντέξουν πολλά Volts(30-40) και είναι πολύ χρήσιμα όταν θέλουμε να έχουμε γεννήτριες σημάτων μεγάλου εύρους.

Οικογένεια I^2L

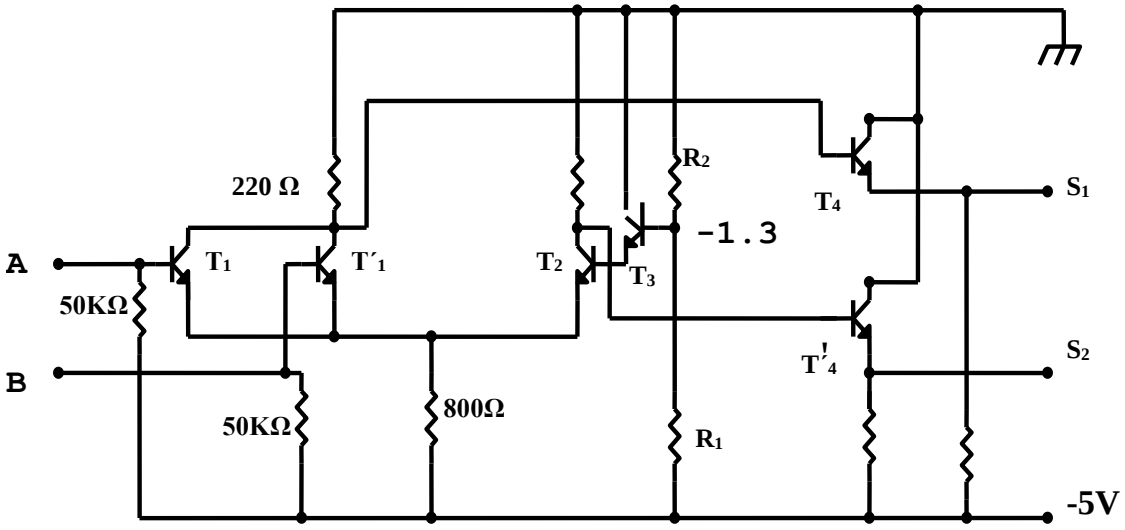
Η οικογένεια αυτή χρησιμοποιεί επίσης το διπολικό τρανζίστορ και χρησιμοποιείται στην περίπτωση που το ολοκληρωμένο κύκλωμα είναι πολύ πολύπλοκο όπως στην περίπτωση ενός επεξεργαστή.

Οικογένεια ECL (Emitter Coupled Logic)

Στα προηγούμενα κυκλώματα που είδαμε, τα τρανζίστορ δουλεύουν σαν διακόπτες είναι δηλαδή στην αποκοπή η είναι στον κόρο και είδαμε τη συνέπεια στην αύξηση της καθυστέρησης. Για να αυξηθεί η ταχύτητα των ολοκληρωμένων κυκλωμάτων η MOTOROLA κατασκεύασε πρώτη μία νέα οικογένεια όπου τα τρανζίστορ δεν είναι ποτέ στον κόρο και την ονόμασε λογική οικογένεια με σύζευξη εκπομπών. Το βασικό κύκλωμα ECL φαίνεται στην παρακάτω εικόνα. Για να είναι δυνατή η γείωση των συλλεκτών των τρανζίστορ στο κύκλωμα εξόδου η τροφοδοσία είναι αρνητική δηλαδή -5V, για να είναι όμως δυνατή η συνδεσμολογία και με άλλες οικογένειες υπάρχει μία τροφοδοσία με θετική τάση. Τα λογικά επίπεδα τάσεων είναι τα παρακάτω :

Επίπεδο High (λογικό 1) → -0.8 Volt

Επίπεδο Low (λογικό 0) → -1.8Volt



Βασικό κύκλωμα ECL

Το παραπάνω κύκλωμα είναι το κύκλωμα ενός διαφορικού ενισχυτή (T1 T'1, T2) ο οποίος συγκρίνει την τάση εισόδου V_A V_B με μία τάση αναφοράς

(-1.3 Volts) η οποία υπάρχει πάνω στην βάση του Τ2 και διατηρείται από το τρανζίστορ Τ3 του οποίου η βάση είναι πολωμένη από το διαιρέτη τάσης R1 R2.

Αν μία από τις δύο εισόδους είναι στο λογικό 1 (-0.8 V), ας πάρουμε την είσοδο Α το τρανζίστορ Τ2 είναι στην αποκοπή, ο συλλέκτης του πηγαίνει στο δυναμικό της βάσης και χάρη στην τάση βάσης εκπομπού του Τ'4 η έξοδος S2 πηγαίνει σε δυναμικό -0.7 μέχρι -0.8 V που αντιστοιχεί στο λογικό 1.

Αντίθετα το ρεύμα στο Τ2 έχει την τιμή :

$$\mathbf{I} = \frac{-0{,}8 - \mathbf{V_{BE}} + 5}{800\Omega} = 4{,}3\text{mA}$$

δίνοντας στον συλλέκτη του Τ1 ένα δυναμικό της τάξης του $-4.3\cdot10^{-3}$x200=-0.96Volt, και στην έξοδο S1 εξ' αιτίας της μετατόπισης εισόδου του Τ4, περίπου -1.7 Volt.

Οι είσοδοι Α και Β παίζουν συμμετρικό ρόλο και το κύκλωμα πραγματοποιεί στην έξοδο S2 τη συνάρτηση NOR και στην S1 τη συνάρτηση NAND.

Για τα παραπάνω κυκλώματα τα οποία είναι αρκετά γρήγορα οι χρόνοι καθυστέρησης είναι της τάξης των 3ns για την οικογένεια MECLII και 1ns για την οικογένεια MECL III. Πρέπει να σημειώσουμε ότι η διαφορά τάσεων ανάμεσα στο λογικό 1 και το λογικό 0 είναι πολύ μικρή και τα κυκλώματα ECL είναι κατά συνέπεια πολύ ευαίσθητα στον θόρυβο.

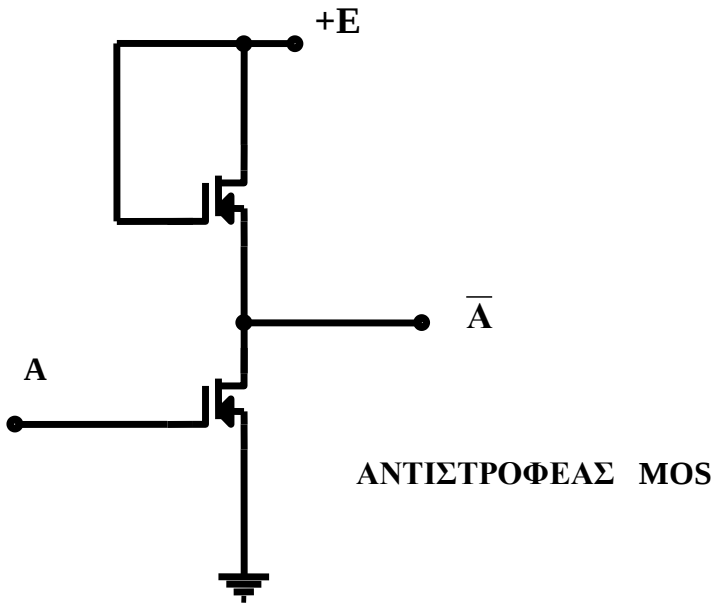
Οικογένεια MOS

Το τρανζίστορ MOS του οποίου η δομή είναι πολύ πιο απλή από αυτή του διπολικού χρησιμοποιείται σαν βασική ηλεκτρονική διάταξη για την κατασκευή ολοκληρωμένων κυκλωμάτων. Μπορούμε να καταχωρήσουμε τα ολοκληρωμένα κυκλώματα που χρησιμοποιούν το τρανζίστορ MOS σαν βασική ηλεκτρονική διάταξη σε τρεις κατηγορίες.

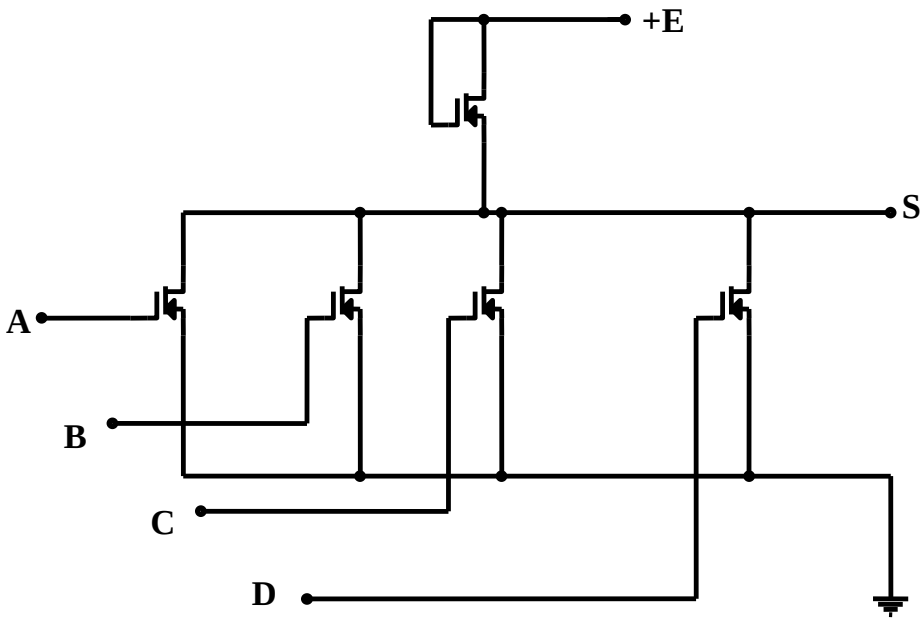
- την PMOS όπου βασική διάταξη είναι το τρανζίστορ με προσμίξεις στο κανάλι τύπου p
- την NMOS όπου το κανάλι έχει προσμίξεις τύπου n,

- και την CMOS όπου πάνω στο ίδιο κομμάτι πυριτίου συνυπάρχουν και τρανζίστορ των δύο παραπάνω τύπων

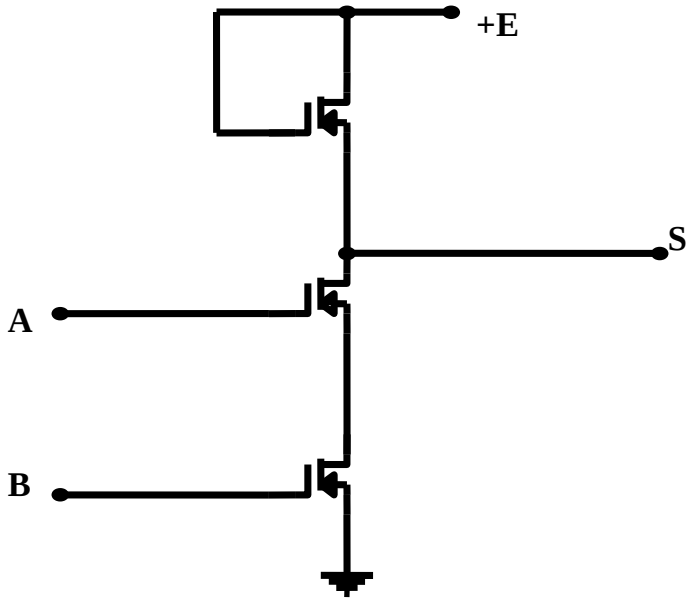
Τα ολοκληρωμένα κυκλώματα PMOS και NMOS έχουν μία πυκνότητα ολοκλήρωσης μεγαλύτερη από την CMOS και είναι κατά συνέπεια περισσότερο οικονομικά. Η οικογένεια ολοκληρωμένων κυκλωμάτων MOS συγκρινόμενη με την οικογένεια TTL του διπολικού τρανζίστορ είναι πολύ πιο αργή, καταναλώνει όμως λιγότερο, έχει μεγαλύτερα περιθώρια ανοχής στον θόρυβο, λειτουργεί κάτω από μία μεγάλη γκάμα τάσεων πόλωσης, έχει μεγαλύτερο fan-out, και τέλος μία πυκνότητα ολοκλήρωσης πολύ πιο μεγάλη. Μπορούμε να δούμε στο παρακάτω σχήμα τον αντιστροφέα MOS.



αντιστροφέας MOS

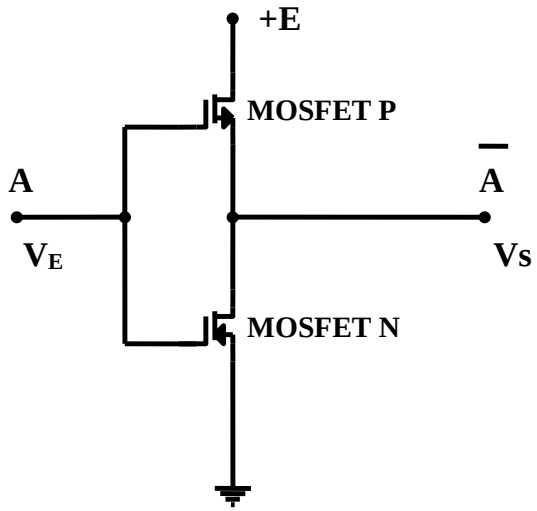


Πύλη NOR τεσσάρων εισόδων με MOS.



Πύλη NAND δύο εισόδων με MOS

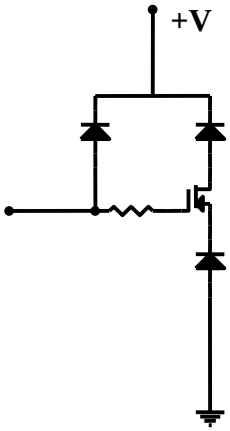
Όπως είδαμε και προηγουμένως η χρήση ταυτοχρόνως και των δύο τύπων τρανζίστορ MOS πάνω στο ίδιο κομμάτι πυριτίου δίνει τη δυνατότητα κατασκευής ολοκληρωμένων κυκλωμάτων με πολύ μικρή κατανάλωση την οποία ονομάζουμε **CMOS.** Η παραπάνω οικογένεια χρησιμοποιείται πολύ στις εφαρμογές MSI όπως η οικογένεια TTL και έχει σαν βασική πύλη τον αντιστροφέα που βλέπουμε στο σχήμα.

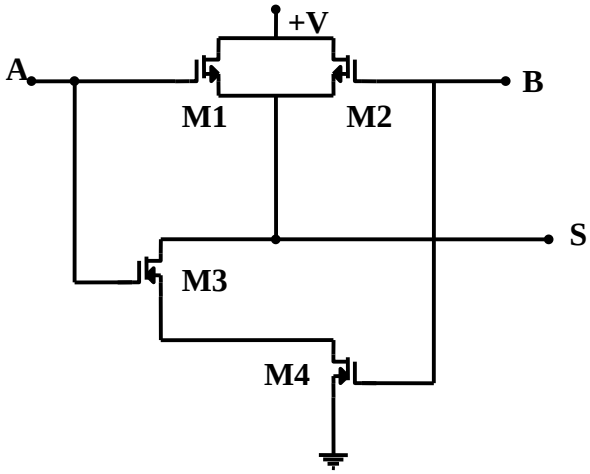


Αντιστροφέας CMOS

Όταν V_E είναι περίπου ίσον με +E, κατάσταση HIGH, το **NMOS** με θετική πόλωση στην πύλη οδηγεί ενώ αντίθετα το **PMOS** δεν οδηγεί. Έτσι η τάση εξόδου Vs είναι πολύ μικρή και το ρεύμα κατανάλωσης μηδέν. Αντίθετα όταν η τάση στην είσοδο είναι σχεδόν μηδέν LOW το τρανζίστορ NMOS δεν οδηγεί και η τάση στην έξοδο είναι μηδέν όπως επίσης και το ρεύμα που καταναλώνεται. Από τα παραπάνω βλέπουμε ότι ποτέ και τα δύο τρανζίστορ δεν οδηγούν συγχρόνως και κατά συνέπεια η κατανάλωση του κυκλώματος όταν βρίσκεται σε μία σταθερή κατάσταση είναι μηδέν. Η κατανάλωση φαίνεται μόνο όταν υπάρχει μεταβατική κατάσταση. Η κατανάλωση για μία μέτρια ταχύτητα της πύλης μπορεί να είναι και εκατό φορές μικρότερη από αυτή της TTL έχουμε όμως χαμηλές συχνότητες λειτουργίας περίπου 10 MHz σε αντίθεση με τα 500MHz για μία πύλη ECL.

Ένα σοβαρό πρόβλημα των ολοκληρωμένων **CMOS** είναι η ευαισθησία που έχουν σε στατικά φορτία και εκφορτίσεις και μπορεί να χαλάσουν πολύ εύκολα ακόμα και όταν τα κρατάμε με γυμνά χέρια. Το πρόβλημα αυτό εντοπίζεται στην μεγάλη ευαισθησία που έχει το λεπτό στρώμα οξειδίου του πυριτίου. Το σοβαρό αυτό πρόβλημα έχει λυθεί κατά ένα μέρος ολοκληρώνοντας διόδους στις εισόδους του κυκλώματος. Στο παρακάτω σχήμα βλέπουμε το κύκλωμα προστασίας στην είσοδο δουλεύει αρκετά καλά, και έχει απαλλάξει τους χρήστες των **CMOS** από αρκετά προβλήματα. Στο δίπλα σχήμα βλέπουμε μία πύλη NAND δύο εισόδων CMOS.





Κύκλωμα προστασίας NAND δύο εισόδων

Στο παρακάτω σχήμα βλέπουμε μία NAND δύο εισόδων. Εάν μία από τις από τις εισόδους είναι στο δυναμικό μηδέν τότε το αντίστοιχο MOS το οποίο είναι τύπου p οδηγεί και δίνει στην έξοδο το δυναμικό +V. Αν αντίθετα M1 και M2 δεν οδηγούν τότε M3 και M4 οδηγούν και η έξοδος S είναι στο δυναμικό μηδέν.

Τα βασικά κυκλώματα της σειράς **4000 CMOS** είναι τα παρακάτω:

4001 τέσσερις NOR δύο εισόδων

4011 τέσσερις NAND δύο εισόδων.

4009 έξη αντιστροφείς.

4013 διπλό Flip Flop D.

4027 "--" JK.

4042 τέσσερα Flip-Flop D.

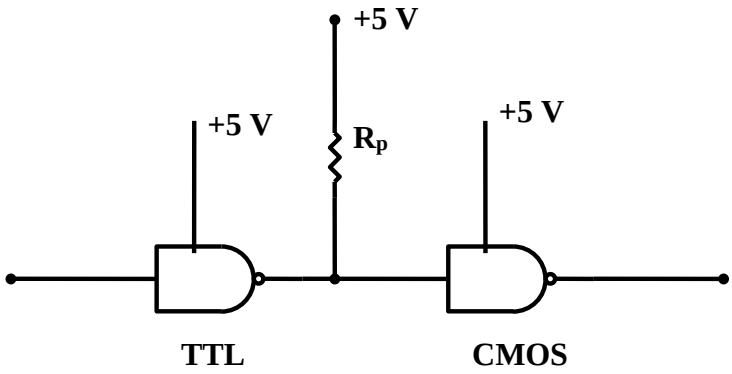
Τα ολοκληρωμένα **CMOS** χρησιμοποιούνται σε ευρεία κλίμακα σήμερα εξ' αιτίας της μικρής κατανάλωσης που έχουν. Αρκετά περίπλοκα κυκλώματα με TTL τα οποία απαιτούν μία σταθεροποιημένη τροφοδοσία μπορεί να λειτουργήσουν με μπαταρίες αν κατασκευαστούν με ολοκληρωμένα CMOS.

INTERFACES

Είναι κυκλώματα τα οποία επιτρέπουν στις διαφορετικές οικογένειες να επικοινωνήσουν μεταξύ τους. Με αυτό τον τρόπο μπορούμε να σχεδιάσουμε ένα ψηφιακό κύκλωμα χρησιμοποιώντας διαφορετικές οικογένειες ολοκληρωμένων κυκλωμάτων. Τις περισσότερες περιπτώσεις έχουμε σύζευξη ECL και TTL η TTL με CMOS.

1. Σύζευξη CMOS με την έξοδο μίας πύλης TTL

Μία πύλη TTL " totem pole " δίνει στην έξοδο το πολύ 0.4 V στην λογική κατάσταση LOW και το λιγότερο 3.6 V στην λογική κατάσταση HIGH. Άρα κάτω από τα 5 V χρειάζεται για την CMOS το πολύ 1.5 V στην λογική κατάσταση 0, και το λιγότερο 3.5 για το λογικό 1. Κατά συνέπεια μία πύλη TTL οδηγεί χωρίς πρόβλημα μία CMOS όσον αφορά το λογικό μηδέν, είναι όμως στα όρια για το λογικό 1. Για το λόγο αυτό χρησιμοποιούμε μία αντίσταση που ονομάζουμε αντίσταση pull-up Rp η οποία ανεβάζει το επίπεδο τάσης στο λογικό 1 όπως φαίνεται στο παρακάτω σχήμα.



Πως η έξοδος μίας TTL οδηγεί μία πύλη CMOS

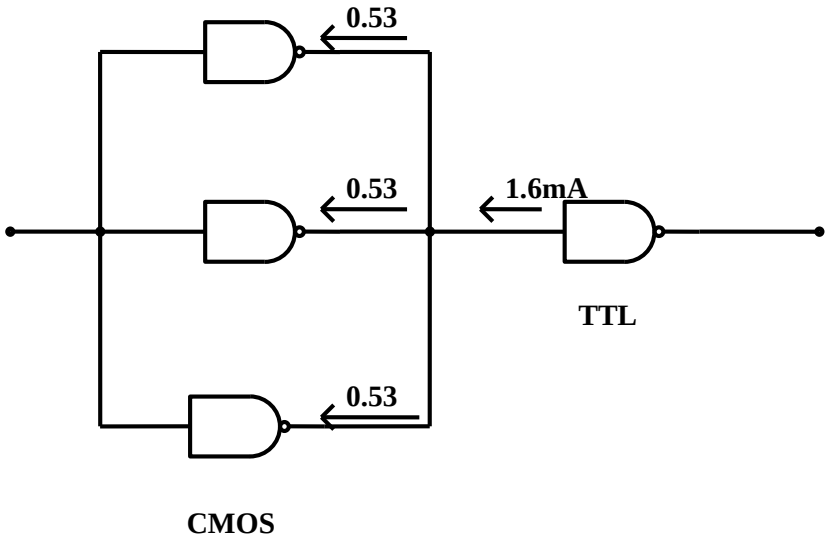
2. Σύζευξη μίας TTL με την έξοδο μίας πύλης CMOS

Στη λογική κατάσταση 1 δεν υπάρχει πρόβλημα γιατί αρκεί ένα πολύ μικρό ρεύμα στην είσοδο της TTL. Δεν ισχύει όμως το ίδιο για το επίπεδο 0 όπου η απευθείας σύνδεση είναι αδύνατη. Πρέπει ή:

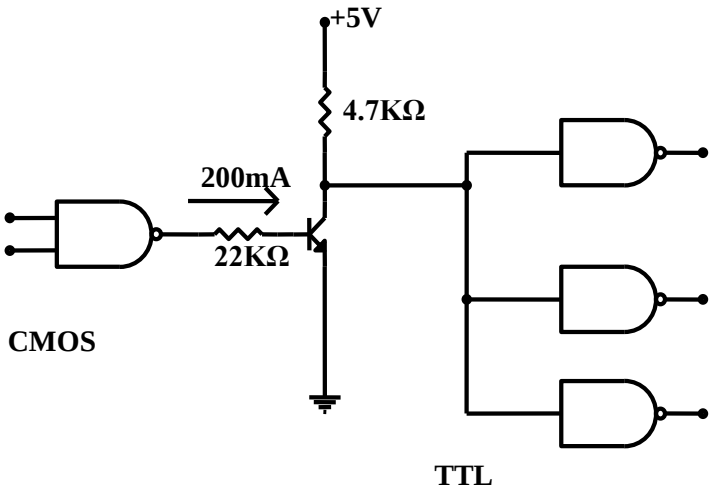
i) να χρησιμοποιήσουμε πολλές CMOS παράλληλα όπως φαίνεται στο παρακάτω σχήμα.

ii) ή να χρησιμοποιήσουμε CMOS ισχύος τύπου BuFlip-Floper, 4009 και 4010 τα οποία μπορούν να τραβήξουν μέχρι 3mA στα 0.4Volts ή

iii) να παρεμβάλουμε ένα τρανζίστορ ανάμεσα το οποίο εισάγει μία νέα αντιστροφή όπως βλέπουμε στο παρακάτω σχήμα.



Συνδεσμολογία πολλών CMOS παράλληλα.

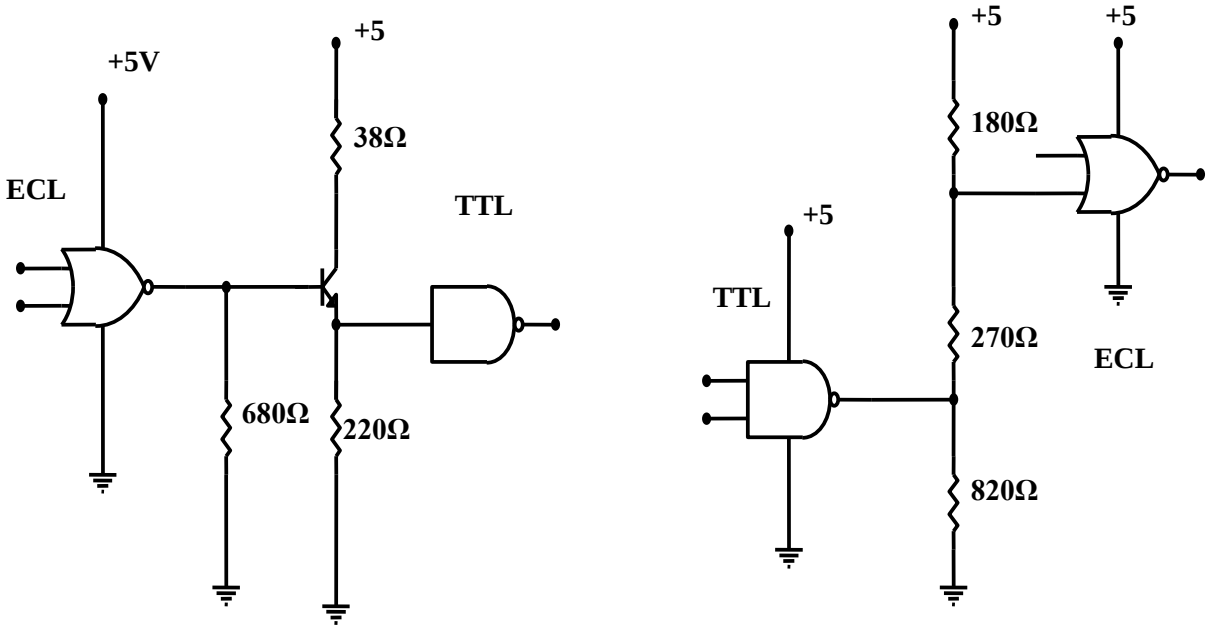


Σύζευξη CMOS - TTL με τρανζίστορ.

3. Σύζευξη ECL με TTL

Η δυσκολία είναι διαφορετική ανάλογα με το αν η ECL τροφοδοτείται από 0 μέχρι 5V η από 0 μέχρι -5V. Αν ξέρουμε από την αρχή ότι θα γίνει μία τέτοια σύζευξη προτείνουμε την πρώτη λύση. Ο κατασκευαστής δίνει

τα παρακάτω κυκλώματα τα οποία φαίνονται στο παρακάτω σχήμα τα οποία επιτρέπουν σύζευξη TTL-ECL και ανάστροφα. Υπάρχουν όμως ειδικά κυκλώματα (MC 1026) τα οποία εξασφαλίζουν τη σύζευξη μεταξύ ECL και TTL χωρίς να έχουμε σημαντικές απώλειες των επιδόσεων των δύο παραπάνω οικογενειών λογικών κυκλωμάτων.



Κυκλώματα σύζευξης ECL - TTL και αντίστροφα.

Κεφάλαιο 4. Κανονικές Μορφές - Πίνακες Karnaugh

Μία δυαδική μεταβλητή παίρνει τις λογικές τιμές 0 ή 1 και με το τρόπο αυτό εμφανίζεται στην κανονική της μορφή η με το συμπλήρωμα της. Ας πάρουμε για παράδειγμα τη λογική πράξη AND και δύο μεταβλητές. Όλοι οι δυνατοί τρόποι που μπορούν να συνδυαστούν οι δύο μεταβλητές μεταξύ τους με την πράξη **και** είναι οι ακόλουθοι τέσσερις τους οποίους ονομάζουμε **ελαχιστόρους**. $\bar{x}\bar{y}, \bar{x}y, x\bar{y}, xy$

Με το ίδιο τρόπο για τη λογική πράξη OR και για δύο μεταβλητές έχουμε πάλι 4 δυνατούς συνδυασμούς οι οποίοι φαίνονται παρακάτω.

$$x+y, x+\bar{y}, \bar{x}+y, \bar{x}+\bar{y}$$

Τους όρους αυτούς ονομάζω **μεγιστόρους.** Αν έχουμε τρεις μεταβλητές όπως και προηγουμένως μπορούμε με τον ίδιο τρόπο να ορίσουμε τους ελαχιστόρους, δηλαδή όλους τους δυνατούς συνδυασμούς για την πράξη **και** όπως:

$$\bar{x}\bar{y}x, x\bar{y}z, \bar{x}yz, xyz, x\bar{y}\bar{z}, \bar{x}y\bar{z}, \bar{x}\bar{y}z, xyz$$

Εδώ σημειώνουμε ότι οι δυνατοί συνδυασμοί είναι $\mathbf{2^3=8}$ όπως και στην περίπτωση των μεγιστόρων που θα είναι 8 και δίνονται παρακάτω:

$$x+y+z, x+y+\bar{z}, x+\bar{y}+z, x+\bar{y}+\bar{z}$$
$$\bar{x}+y+z, \bar{x}+y+\bar{z}, \bar{x}+\bar{y}+z, \bar{x}+\bar{y}+\bar{z}$$

Θα δούμε παρακάτω ότι οι μεγιστόροι και οι ελαχιστόροι καθορίζουν τις δύο κανονικές μορφές μιας λογικής συνάρτησης Boole. Με τον ίδιο τρόπο που ορίσαμε τους ελαχιστόρους και τους μεγιστόρους στην περίπτωση των δύο μεταβλητών κάνουμε και αν έχουμε περισσότερες μεταβλητές. Αρκεί να θυμόμαστε ότι για τους ελαχιστόρους και την πράξη **και** όλοι οι όροι θα πρέπει να είναι ίσοι με 1 ενώ για τους μεγιστόρους και την πράξη **ή** όλοι οι όροι θα είναι ίσοι με μηδέν. Ο συνολικός αριθμός ελαχιστόρων η μεγιστόρων είναι πάντα στην περίπτωση που έχουμε n μεταβλητές 2^n. Έστω ότι μου δίνουν μία συνάρτηση τριών μεταβλητών F(A,B,C) και μας λένε να την ορίσουμε. Να βρούμε δηλαδή πότε η F(A,B,C) είναι ίση με 1 η ότι πότε είναι αληθής με τις παρακάτω προϋποθέσεις:

- A=0, B=1 και C=0 η όταν
- A=0, B=1 και C=1, η όταν
- A=1, B=0 και C=0 η όταν
- A=1, B=1 και C=1

αν υποθέσουμε ότι η κανονική μορφή μία μεταβλητής είναι 1, και η συμπληρωματική της 0 τότε σύμφωνα με την εκφώνηση προηγουμένως μπορούμε να γράψουμε:

$$F = \overline{A}B\overline{C} + \overline{A}BC + A\overline{B}\overline{C} + ABC$$

Αυτή η μορφή της παραπάνω συνάρτησης λέγεται κανονική. Σε αυτή τη περίπτωση όλοι οι όροι περιέχουν όλες τις μεταβλητές. Από την προηγούμενη έκφραση της F βλέπουμε ότι είναι ένα άθροισμα γινομένων γιατί συνδέει με OR τους όρους που προκύπτουν από την πράξη ΚΑΙ. Η αναλυτική έκφραση της συνάρτησης F αποτυπώνεται πολύ εύκολα από τον πίνακα αλήθειας που φαίνεται παρακάτω.

A	B	C	F	Ελαχιστόρος	Μεγιστόρος
0	0	0	0	m_0	M_0
0	0	1	0	m_1	M_1
0	1	0	1	m_2	M_2
0	1	1	1	m_3	M_3
1	0	0	1	m_4	M_4
1	0	1	0	m_5	M_5
1	1	0	0	m_6	M_6
1	1	1	1	m_7	M_7

Για κάθε συγκεκριμένο συνδυασμό των μεταβλητών A,B,C και ανάλογα με τη λογική πράξη που συνδέει αυτές τις μεταβλητές μεταξύ τους (AND η OR) έχουμε αντίστοιχα ένα ελαχιστόρο η ένα μεγιστόρο και μόνο ένα. Οι ελαχιστόροι και οι μεγιστόροι ορίζονται όπως φαίνεται παρακάτω για μία συνάρτηση τριών μεταβλητών.

$m_0 = \bar{x}\,\bar{y}\,\bar{z}$	$M_0 = x+y+z$
$m_1 = \bar{x}\,\bar{y}\,z$	$M_1 = x+y+\bar{z}$
$m_2 = \bar{x}\,y\,\bar{z}$	$M_2 = x+\bar{y}+z$
$m_3 = \bar{x}\,y\,z$	$M_3 = x+\bar{y}+\bar{z}$
$m_4 = x\,\bar{y}\,\bar{z}$	$M_4 = \bar{x}+y+z$
$m_5 = x\,\bar{y}\,z$	$M_5 = \bar{x}+y+\bar{z}$
$m_6 = x\,y\,\bar{z}$	$M_6 = \bar{x}+\bar{y}+z$
$m_7 = x\,y\,z$	$M_7 = \bar{x}+\bar{y}+\bar{z}$

Βλέπουμε ότι η παραπάνω συνάρτηση ορίζεται από τους ελαχιστόρους m2,m3,m4, και τον m7 και γράφουμε F(x,y,z)=Σ(2,3,4,7). Το Σ δηλώνει το άθροισμα των ελαχιστόρων 2,3,4,7 (χρησιμοποιούμε μόνο τους δείκτες). Μπορούμε να αποδείξουμε πολύ εύκολα ότι το συμπλήρωμα ενός ελαχιστόρου δίνει μεγιστόρο με τον ίδιο δείκτη όπως επίσης ότι από το συμπλήρωμα ενός μεγιστόρου προκύπτει ο ελαχιστόρος με τον ίδιο δείκτη.

$$m_0 = \bar{x}\bar{y}\bar{z}$$
$$\overline{m_0} = \overline{\bar{x}\bar{y}\bar{z}} = \bar{\bar{x}} + \bar{\bar{y}} + \bar{\bar{z}} = x + y + z = M_0$$

Σημαντική ιδιότητα: κάθε συνάρτηση Boole μπορεί να εκφραστεί σαν άθροισμα ελαχιστόρων η σαν γινόμενο μεγιστόρων. Παρακάτω θα δούμε πως αλλάζουμε κανονική μορφή δηλαδή πως αν ξέρουμε το άθροισμα των ελαχιστόρων μιας συνάρτησης πηγαίνουμε σε γινόμενο μεγιστόρων και αντίστροφα. Θα δούμε επίσης πως πηγαίνουμε στις κανονικές μορφές του συμπληρώματος μίας συνάρτησης F γνωρίζοντας την κανονική της μορφή συνάρτηση.

Παράδειγμα: Έστω η λογική συνάρτηση τριών μεταβλητών

$$F(A,B,C) = \bar{A}B\bar{C} + ABC + \bar{A}\bar{B}\bar{C}$$

Βλέπουμε ότι τη συνάρτηση F(A,B,C) ορίζουν οι ελαχιστόροι m0,m2,m7. Οι ελαχιστόροι που λείπουν είναι οι m1,m3,m4,m5, και ο m6 οι οποίοι όμως είναι οι ελαχιστόροι που ορίζουν τη συνάρτηση συμπλήρωμα $\overline{F}(A,B,C)$. Το παραπάνω φαίνεται πολύ εύκολα από τον πίνακα αλήθειας των συναρτήσεων πίνακας 10.

A	**B**	**C**	**F**	$\overline{F}$	**Ελαχιστόροι**
0	0	0	1	0	m_0
0	0	1	0	1	m_1
0	1	0	1	0	m_2
0	1	1	0	1	m_3
1	0	0	0	1	m_4
1	0	1	0	1	m_5
1	1	0	0	1	m_6
1	1	1	1	0	m_7

Ελαχιστόροι μίας συνάρτησης και του συμπληρώματος της.

Έχουμε λοιπόν ότι **F(A,B,C)=m₀+m₂+m₇**. Αν συμπληρώσουμε την παραπάνω σχέση και εφαρμόσουμε τις σχέσεις του DE MORGAN θα πάρουμε ότι:

$$\overline{F(A,B,C)} = \overline{m_0 + m_2 + m_7} \Rightarrow$$
$$\overline{F(A,B,C)} = m_0 + m_2 + m_7$$
$$F(A,B,C) = M_0 \cdot M_2 \cdot M_7 \equiv\rightarrow$$

Βλέπουμε ότι από την τελευταία σχέση η συνάρτηση F(A,B,C) θα ισούται με το γινόμενο των μεγιστόρων M1·M3·M4·M5·M6 Οπότε συνοψίζοντας έχουμε ότι:

F(A,B,C)=m₀+m₂+m₇=Σ(0,2,7)

επίσης ότι **F(A,B,C)=M₁·M₃·M₄·M₅·M₆ =Π(1,3,4,5,6)**. Δηλαδή για να αλλάξουμε κανονική μορφή αρκεί να πάρουμε τους δείκτες των ελαχιστόρων η των μεγιστόρων που λείπουν, έτσι πηγαίνουμε με ευκολία από αθροίσματα γινομένων σε γινόμενα αθροισμάτων. Όπως θα δούμε

παρακάτω είναι πολύ χρήσιμες αυτές οι μετατροπές για τις υλοποιήσεις των λογικών συναρτήσεων. Είδαμε επίσης με αυτό το παράδειγμα πως από μία κανονική μορφή της F μπορούμε να βρούμε τις κανονικές μορφές του συμπληρώματος της.

Παράδειγμα: Έστω η συνάρτηση **F(A,B,C,D)=Σ(0,1,5,6,7,13,15)**. Να βρεθούν οι κανονικές μορφές της συνάρτησης όπως επίσης και του συμπληρώματος της.

Σύμφωνα με το προηγούμενο παράδειγμα θα έχουμε ακριβώς τα ίδια δηλαδή: ότι αν:

F(A,B,C,D)=Σ(0,1,5,6,7,13,15)=Π(2,3,4,8,9,10,11,12,14)

δηλαδή από το άθροισμα των ελαχιστόρων για να βρούμε το γινόμενο των μεγιστόρων αρκεί να πάρουμε τους δείκτες των όρων που λείπουν από την πρώτη κανονική μορφή για να πάμε στην δεύτερη μορφή. Τότε η συνάρτηση συμπλήρωμα γίνεται:

$$\overline{F}(A,B,C,D) = \Sigma(2,3,4,8,9,10,11,12,14) = \Pi(0,1,5,6,7,13,15)$$

Βλέπουμε ότι μπορούμε να γνωρίζουμε τις κανονικές μορφές χωρίς καν να χρησιμοποιήσουμε το πίνακα αλήθειας της συνάρτησης.

Μπορεί σε ορισμένες περιπτώσεις να έχουμε να κάνουμε με μία συνάρτηση η οποία μας έχει δοθεί σε μια μορφή η οποία είναι τυχαία. Δηλαδή ενώ η συνάρτηση είναι τεσσάρων μεταβλητών υπάρχουν όροι τριών δύο η ακόμα και μίας μεταβλητής. Σε αυτή την περίπτωση για να μπορέσω να βρω τις κανονικές μορφές της συνάρτησης και να προχωρήσω στην συνέχεια στην απλοποίηση της και στην υλοποίηση της είναι αναγκαίο να συμπληρώσω τους όρους με τις μεταβλητές που λείπουν. Ο τρόπος εισαγωγής μιας μεταβλητής σε ένα όρο που λείπει είναι να πολλαπλασιάσω τον όρο με το ουδέτερο στοιχείο του πολλαπλασιασμού που είναι : $[X + \overline{X}] = 1$ θα το δούμε όμως παρακάτω σε παραδείγματα.

Παράδειγμα: Βρείτε τις κανονικές μορφές της παρακάτω λογικής συνάρτησης **F(A,B,C)=AB+BC**

Βλέπουμε ότι η συνάρτηση είναι δύο μεταβλητών και οι όροι που την ορίζουν περιέχουν μόνο δύο μεταβλητές. Από τον πρώτο όρο λείπει η

μεταβλητή C και από το δεύτερο η μεταβλητή A. Η διαδικασία εύρεσης της κανονικής μορφής της συνάρτησης φαίνεται παρακάτω:

$$F(A,B,C) = AB + BC$$
$$F(A,B,C) = AB(\overline{C} + C) + (\overline{A} + A)BC$$
$$F(A,B,C) = AB\overline{C} + ABC + \overline{A}BC + ABC$$

Ο όρος ABC είναι δύο φορές και κατά συνέπεια μπορούμε να τον παραλείψουμε και να το λάβουμε υπ' όψιν μας μόνο μία φορά σύμφωνα με την παρακάτω ιδιότητα.

$$F(A,B,C) = ABC + \overline{A}BC + AB\overline{C}$$

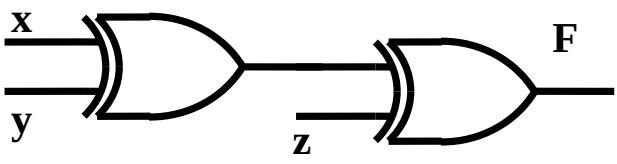
Η τελευταία σχέση λέει ότι

F(A,B,C)=m3+m6+m7 = Σ(3,6,7) = Π(0,1,2,4,5)

Παράδειγμα: Δίνεται η συνάρτηση Boole F(A,B,C)=A+BC. Σε αυτή τη συνάρτηση από το πρώτο όρο λείπουν δύο μεταβλητές και από τον δεύτερο μία. Η διαδικασία φαίνεται παρακάτω :

$$F(A,B,C) = A + BC$$
$$F(A,B,C) = A(\overline{B} + B)(\overline{C} + C) + (\overline{A} + A)BC$$
$$F(A,B,C) = (A\overline{B} + AB)(\overline{C} + C) + \overline{A}BC + ABC$$
$$F(A,B,C) = A\overline{B}\overline{C} + A\overline{B}C + AB\overline{C} + ABC + \overline{A}BC + ABC$$
$$F(A,B,C) = A\overline{B}\overline{C} + A\overline{B}C + AB\overline{C} + ABC + \overline{A}BC$$
$$F(A,B,C) = m4 + m5 + m6 + m7 + m3$$
$$F(A,B,C) = \Sigma(3,4,5,6,7) \quad = \Pi(0,1,2)$$

Παράδειγμα: Δίνεται το παρακάτω ψηφιακό κύκλωμα. Βρείτε τη λογική συνάρτηση που δίνει στην έξοδο. Στην συνέχεια δώσετε τον πίνακα αλήθειας και τις κανονικές μορφές της συνάρτησης.



$$
\begin{aligned}
F(x,y,z) &= x \oplus y \oplus z = \left(\bar{x}y + x\bar{y}\right) \oplus z \\
&= \overline{\left(\bar{x}y + x\bar{y}\right)} \bullet z + \left(\bar{x}y + x\bar{y}\right) \bullet \bar{z} \\
&= \left(\overline{\bar{x}y}\right) \bullet \left(\overline{x\bar{y}}\right) \bullet z + \bar{x}y\bar{z} + x\bar{y}z \\
&= (x + \bar{y}) \bullet (\bar{x} + y) \bullet z + \bar{x}y\bar{z} + x\bar{y}z \\
&= xyz + \bar{x}\bar{y}z + \bar{x}y\bar{z} + x\bar{y}\bar{z}
\end{aligned}
$$

Άρα η συνάρτηση στην έξοδο είναι η F(x,y,z)=Σ(1,2,4,7)=Π(0,3,5,6)

<u>Άσκηση:</u> Δίνονται δύο συναρτήσεις Τ1 και Τ2. Δείξατε ότι η συνάρτηση γινόμενο Η1=Τ1·Τ2 ορίζεται από τους κοινούς ελαχιστόρους της Τ1 και της Τ2 (δηλαδή από την τομή) και ότι η συνάρτηση Η2=Τ1+Τ2 ορίζεται από όλους τους ελαχιστόρους της Τ1 και της Τ2 (δηλαδή από την ένωση).

Όταν ορίσαμε τους ελαχιστόρους είπαμε ότι ο αριθμός τους είναι συνάρτηση του αριθμού των μεταβλητών της συνάρτησης και ότι οι ελαχιστόροι περιέχουν όλους τους δυνατούς συνδυασμούς για τη λογική πράξη AND των μεταβλητών. Κατά συνέπεια δεν υπάρχουν δύο ελαχιστόροι ίδιοι. Αν πολλαπλασιάσουμε δύο ελαχιστόρους mi.mj το αποτέλεσμα θα είναι ίσο με μηδέν αν i≠j και ίσον με mi αν i=j. Αυτό σημαίνει ότι αν πολλαπλασιάσω τους ελαχιστόρους της συνάρτησης Τ1 με αυτούς της Τ2 το αποτέλεσμα θα είναι διαφορετικό του μηδενός μόνο για τους κοινούς ελαχιστόρους. Για τη συνάρτηση Η2 το αποτέλεσμα θα είναι όλοι οι κοινοί ελαχιστόροι των Τ1 και Τ2 από μία φορά. Είναι κάτι ανάλογο σαν την τομή δύο συνόλων η περίπτωση του πολλαπλασιασμού συναρτήσεων και κάτι ανάλογο με την ένωση το άθροισμα λογικών συναρτήσεων.

Παράδειγμα: Δίνονται οι λογικές συναρτήσεις F1(x,y,z)=Σ(1,5,7) και F2(x,y,z)=Σ(0,1,5,6,7) Δώστε τις κανονικές μορφές των παρακάτω λογικών συναρτήσεων. F(x,y,z)=F1+F2 και G(x,y,z)= F1·F2 Έχουμε ότι F(x,y,z)=Σ(0,1,5,6,7)=Π(2,3,4) και G(x,y,z)=Σ(1,5)=Π(0,2,3,4,6,7). Βλέπουμε πόσο γρήγορα βρήκαμε το αποτέλεσμα του γινομένου και του αθροίσματος λογικών συναρτήσεων χωρίς καν να κάνουμε πράξεις.

Υλοποίηση Συναρτήσεων & Απλοποίηση

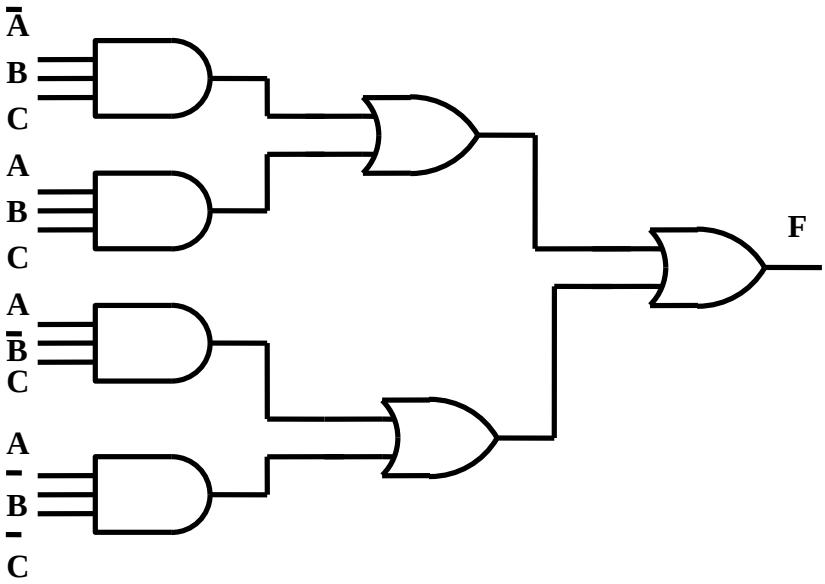
Για την υλοποίηση των λογικών συναρτήσεων είναι ευνόητο ότι αναζητούμε την πιο απλή μορφή της συγκεκριμένης συνάρτησης που έχουμε να υλοποιήσουμε. Η πιο απλή μορφή, μεταφράζεται με λιγότερους μαθηματικούς όρους και θα μας επιτρέψει να χρησιμοποιήσουμε τον ελάχιστο αριθμό πυλών και κατά συνέπεια και καλωδιώσεων.

Ο απλούστερος τρόπος ελαχιστοποίησης μίας συνάρτησης είναι η χρησιμοποίηση όλων των σχέσεων που είδαμε ορίζοντας την Άλγεβρα Boole. Χρησιμοποιούμε μία πληθώρα σχέσεων και κάνουμε πράξεις όπως στην κλασσική Άλγεβρα. Δεν υπάρχουν νόμοι και οι βασικές σχέσεις που χρησιμοποιούμε είναι το ουδέτερο στοιχείο του πολλαπλασιασμού (AND) και της πρόσθεσης (OR). Σε αυτή την περίπτωση ειδικά όταν οι συναρτήσεις Boole είναι πολλών μεταβλητών και υπάρχουν πολλοί όροι η δυνατότητα συνδυασμών των όρων είναι μεγάλη και δεν μπορούμε να είμαστε ποτέ σίγουροι ότι η απλοποιημένη μορφή που θα πετύχουμε με αυτό τον τρόπο είναι η ελάχιστη. Ας δούμε όμως παρακάτω μερικά παραδείγματα.

Παράδειγμα

Δίνεται η συνάρτηση $F(A,B,C) = \overline{A}BC + ABC + A\overline{B}C + A\overline{B}\overline{C}$

1. Να υλοποιηθεί όπως είναι
2. Να απλοποιηθεί
3. Να υλοποιηθεί το απλοποιημένο κύκλωμα και να συγκριθεί με το αρχικό. Για την υλοποίηση υποθέτω ότι οι αντιστροφές είναι διαθέσιμες στην είσοδο και θα χρησιμοποιήσω πύλες AND και OR του παρακάτω σχήματος.

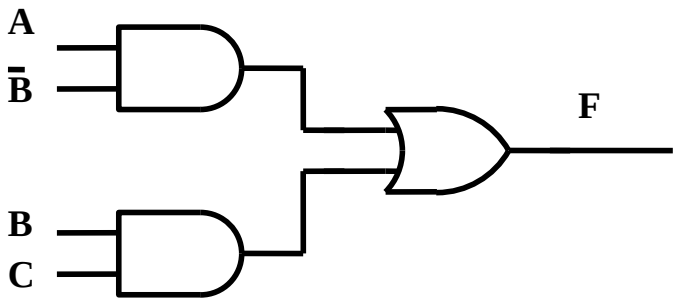


Υλοποίηση της μη απλοποιημένης μορφής της F(A,B,C)

Αν τώρα προσπαθήσω να απλοποιήσω τη λογική συνάρτηση θα έχω ότι:

$$\begin{aligned} F(A,B,C) &= \bar{A}BC + ABC + A\bar{B}C + A\bar{B}\bar{C} \\ &= (\bar{A} + A)\cdot BC + A\bar{B}\cdot(C + \bar{C}) \\ &= A\bar{B} + BC \end{aligned}$$

Και κατά συνέπεια το κύκλωμα σε αυτή την περίπτωση θα είναι αυτό που φαίνεται στο παρακάτω σχήμα:



Υλοποίηση της απλοποιημένης μορφής της F(A,B,C).

Παρατηρώντας τα παρακάτω σχήματα βλέπουμε πόσο πιο απλό είναι το απλοποιημένο κύκλωμα και την οικονομία σε υλικό που κάνουμε. Δεν είναι μόνο η οικονομία στην κατασκευή. Το απλοποιημένο κύκλωμα εκτός των άλλων θα έχει μικρότερη κατανάλωση και μικρότερη καθυστέρηση. Θα είναι με άλλα λόγια πιο γρήγορο βασικός στόχος σε αυτά τα κυκλώματα.

Άσκηση: Δίνεται ο παρακάτω πίνακας αληθείας. Βρείτε τις συναρτήσεις Τ1 και Τ2 και απλοποιήστε τις στον ελάχιστο αριθμό παραγόντων.

A	B	C	T1	T2
0	0	0	1	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	0	1

Από τον πίνακα αλήθειας βλέπουμε ότι $T1=m_0+m_1+m_2$ και ότι $T2=m_3+m_4+m_5+m_6+m_7$

$$
\begin{aligned}
\mathbf{T1} &= \overline{\mathbf{A}}\,\overline{\mathbf{B}}\,\overline{\mathbf{C}} + \overline{\mathbf{A}}\,\overline{\mathbf{B}}\mathbf{C} + \overline{\mathbf{A}}\mathbf{B}\,\overline{\mathbf{C}} \\
&= \overline{\mathbf{A}}\,\overline{\mathbf{B}}(\overline{\mathbf{C}} + \mathbf{C}) + \overline{\mathbf{A}}\mathbf{B}\,\overline{\mathbf{C}} \\
&= \overline{\mathbf{A}}\,\overline{\mathbf{B}} + \overline{\mathbf{A}}\mathbf{B}\,\overline{\mathbf{C}} \\
&= \overline{\mathbf{A}}(\overline{\mathbf{B}} + \mathbf{B}\,\overline{\mathbf{C}}) \\
&= \overline{\mathbf{A}}(\mathbf{B} + \overline{\mathbf{B}})(\overline{\mathbf{B}} + \overline{\mathbf{C}}) \\
&= \overline{\mathbf{A}}(\overline{\mathbf{B}} + \overline{\mathbf{C}})
\end{aligned}
$$

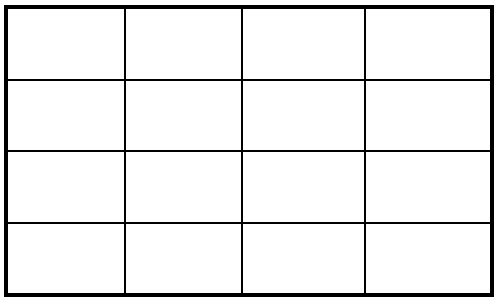
Από τον πίνακα αλήθειας βλέπω ότι η συνάρτηση Τ1 είναι το συμπλήρωμα της Τ2 και αντίστροφα. Κατά συνέπεια για να αποφύγω τις πράξεις αρκεί να συμπληρώσω την απλοποιημένη σχέση που βρήκα για την Τ1 και το αποτέλεσμα θα είναι η απλοποίηση της Τ2. Δηλαδή $\mathbf{T_2=A+BC}$.

Μέχρι τώρα είδαμε στα περισσότερα παραδείγματα συναρτήσεις τριών μεταβλητών οι οποίες είναι σχετικά απλές και στην απλοποίηση τους. Αν έχω όμως να κάνω με περισσότερες μεταβλητές η πιθανότητα σφάλματος στην απλοποίηση είναι μεγάλη και γι' αυτό το λόγο καταφεύγουμε σε μία άλλη μέθοδο σχετικά απλή η οποία δίνει εύκολα την ελάχιστη μορφή μιας λογικής συνάρτησης. Η μέθοδος αυτή λέγεται μέθοδος του **χάρτη** η πίνακας **Karnaugh**.

Ο πίνακας Karnaugh είναι ένας πίνακας του οποίου οι διαστάσεις ποικίλουν ανάλογα με τη συνάρτηση που έχω να απλοποιήσω. Ο αριθμός των συνδυασμών που έχω για κάθε συνάρτηση είναι ίσος με $\mathbf{2^N}$ αν Ν είναι ο αριθμός μεταβλητών) καθορίζει τον αριθμό των τετραγώνων του πίνακα. Έτσι για μία συνάρτηση δύο μεταβλητών ο πίνακας Karnaugh θα έχει τέσσερα τετράγωνα,

για μία συνάρτηση τριών μεταβλητών οκτώ τετράγωνα,

για μία συνάρτηση τεσσάρων μεταβλητών δεκάξι τετράγωνα κ.λ.π.



Ο πίνακας Karnaugh είναι ένας πίνακας διπλής εισόδου και κάθε ένα τετράγωνο περιέχει ένα συνδυασμό των μεταβλητών για την πράξη AND δηλαδή περιέχει ένα ελαχιστόρο. Η διάταξη των τετραγώνων δεν είναι τυχαία αλλά ακολουθεί την παρακάτω λογική. Τα τετράγωνα είναι γειτονικά. Αυτό σημαίνει ότι πηγαίνοντας από το ένα τετράγωνο στο άλλο (όχι όμως διαγώνια) μόνο μία μεταβλητή αλλάζει. Αυτό σημαίνει ότι αν στο ένα τετράγωνο μία μεταβλητή είναι στην κανονική της μορφή στο γειτονικό τετράγωνο θα είναι με τη συμπληρωματική της. Μπορεί δηλαδή να έχω **xyz** στο ένα τετράγωνο οπότε στο γειτονικό του θα έχω $\mathbf{xy\bar{z}}$. Μόνο η μεταβλητή z αλλάζει από το ένα τετράγωνο στο άλλο. Αν

συνδυάσω δύο τετράγωνα γειτονικά μεταξύ τους η μεταβλητή που αλλάζει απλοποιείται σύμφωνα με την ιδιότητα

$$\overline{x} + x = 1$$

Ο πίνακας **Karnaugh** είναι ένα εργαλείο το οποίο μας επιτρέπει να απλοποιήσουμε μία συνάρτηση στην ελάχιστη της μορφή. Η ελαχιστοποίηση γίνεται είναι κατά κάποιο τρόπο μπορούμε να πούμε κωδικοποιημένη και συνίσταται σε συνδυασμούς απλά γειτονικών τετραγώνων. Παρακάτω δίνουμε τη διάταξη των ελαχιστόρων στους πίνακες K**arnaugh** για τις περιπτώσεις των δύο μεταβλητών A,B, τριών μεταβλητών A,B,C και τεσσάρων μεταβλητών A,B,C,D .

	$\overline{B}$	B
$\overline{A}$	$\overline{A}\,\overline{B}=m_0$	$\overline{A}\,B=m_1$
A	$A\,\overline{B}=m_2$	$AB=m_3$

Χάρτης μίας συνάρτησης δύο μεταβλητών.

A \ B C	$\overline{B}\,\overline{C}$	$\overline{B}\,C$	BC	$B\overline{C}$
$\overline{A}$	m0	m1	m3	m2
A	m4	m5	m7	m6

Χάρτης μίας συνάρτησης τριών μεταβλητών και χωροθέτηση των 8 αντιστοίχων ελαχιστόρων στα τετράγωνα τους σύμφωνα με τον οδηγό που μας δίνει ο πίνακας διπλής εισόδου

	$\bar{C}\,\bar{D}$	$\bar{C}\,D$	$C\,D$	$C\,\bar{D}$
$\bar{A}\,\bar{B}$	m0	m1	m3	m2
$\bar{A}\,B$	m4	m5	m7	m6
$A\,B$	m12	m13	m15	m14
$A\,\bar{B}$	m8	m9	m11	m10

Χάρτης μίας συνάρτησης τεσσάρων μεταβλητών και χωροθέτηση των 16 αντιστοίχων ελαχιστόρων.

Όπως βλέπουμε στους παραπάνω χάρτες η κατανομή των ελαχιστόρων έχει γίνει με τέτοιο τρόπο έτσι ώστε όλα τα τετράγωνα να είναι γειτονικά. Δηλαδή πηγαίνοντας από το ένα στο άλλο μόνο μία μεταβλητή αλλάζει. Εδώ πρέπει να πούμε επίσης ότι ο πίνακας Karnaugh είναι σαν ένα γεωμετρικό σχήμα που κλείνει γύρω από τον εαυτό του, έτσι στον πίνακα Karnaugh που βρίσκεται στο παρακάτω σχήμα το τετράγωνο με το ελαχιστόρο $\mathbf{m_0}$ είναι γειτονικό του $\mathbf{m_8}$ όπως επίσης και του $\mathbf{m_2}$. Είναι σαν να έχουμε σχεδιάσει τον πίνακα Karnaugh πάνω σε μια ελαστική μεμβράνη με την οποία τυλίγουμε ένα μπαλόνι. Τότε η γραμμή του πίνακα θα ακουμπήσει την τελευταία και η πρώτη κολώνα την τελευταία.

Έτσι το m0 γίνεται γειτονικό του m8 και του m2, το m4 του m6, το m12 του m14, το m8 του m10, επίσης είναι γειτονικά τα τέσσερα γωνιακά τετράγωνα m0, m2, m8, m10. Όταν έχουμε μία συνάρτηση και θέλουμε να την απλοποιήσουμε με τη βοήθεια του χάρτη χρειαζόμαστε την κανονική μορφή της συνάρτησης για να μπορέσουμε να συμπληρώσουμε τον αντίστοιχο χάρτη ή πίνακα Karnaugh. Όταν την έχουμε πηγαίνω και τοποθετώ ένα 1 στα τετράγωνα των ελαχιστόρων που ορίζουν τη συνάρτηση και ένα 0 στα υπόλοιπα τετράγωνα. Με αυτό τον τρόπο έχω αυτόματα έτοιμο και τον πίνακα Karnaugh η χάρτη και του συμπληρώματος

της συνάρτησης αρκεί να αλλάξω τα 1 σε 0 και τα 0 σε 1 πράγμα που θα μου χρειαστεί πολύ παρακάτω όπως θα δούμε.

Η ιδιότητα που χρησιμοποιώ για να απλοποιήσω είναι

$$\bar{\mathbf{x}} + \mathbf{x} = 1$$

Αντί όμως να προχωρήσω σε πράξεις για να απλοποιήσω την συνάρτηση συνδυάζω τα τετράγωνα που είναι γειτονικά υπενθυμίζοντας ότι είναι με τέτοιο τρόπο τοποθετημένα ώστε να μπορώ να εφαρμόζω την παραπάνω ιδιότητα και να απλοποιήσω χωρίς πράξεις. Τα τετράγωνα που συνδυάζω τα περικλείω με μία γραμμή κλειστή γιατί τις περισσότερες φορές έχω πολλούς συνδυασμούς να κάνω πριν την τελική απλοποίηση. Έτσι

- αν συνδυάσω **δύο τετράγωνα απλοποιώ μία μεταβλητή,**
- αν συνδυάσω **τέσσερα απλοποιώ δύο μεταβλητές**
- αν συνδυάσω **οκτώ τρεις μεταβλητές**
- και τέλος αν συνδυάσω δεκάξι τετράγωνα σημαίνει ότι η συνάρτηση μου είναι παντού 1 και κατά συνέπεια απλοποιούνται όλες οι μεταβλητές.

Για συναρτήσεις με περισσότερες από 4 μεταβλητές τα πράγματα δυσκολεύουν στη παρουσίαση των αντίστοιχων πινάκων **Karnaugh**.

Τα τετράγωνα με τους ελαχιστόρους γίνονται πολλά και η γεωμετρία τοποθέτησης και γειτονίας περιπλέκεται. Ας δούμε όμως πως είναι ο πίνακας **Karnaugh** μιας συνάρτησης πέντε μεταβλητών στο παρακάτω σχήμα και στη συνέχεια έξη.

AB \ CDE	000	001	011	010	110	111	101	100
00								
01								
11								
10								

Πίνακας **Karnaugh**

Ο αριθμός των ελαχιστόρων και κατά συνέπεια των τετραγώνων του πίνακα είναι 2^N όπου Ν αριθμός των μεταβλητών. Στη περίπτωση των 5 έχουμε 32 ελαχιστόρους και κατά συνέπεια 32 τετράγωνα στο πίνακα **Karnaugh.** Μπορούμε σύμφωνα με τη διάταξη των τετραγώνων να πούμε ότι ο πίνακας **Karnaugh** μιας συνάρτησης 5 μεταβλητών αποτελείται από δύο πίνακες τεσσάρων μεταβλητών διατεταγμένοι όπως οι σελίδες ενός βιβλίου. Όταν το βιβλίο κλείσει κατά μήκος των διπλών γραμμών οι πίνακες τεσσάρων μεταβλητών θα έλθουν ο ένας πάνω στο άλλο. Έτσι τα τετράγωνα που πέφτουν το ένα πάνω στο άλλο θα θεωρούνται γειτονικά.

ABC \ DEF	000	001	011	010	100	101	111	110
000								
001								
011								
010								
100								
101								
111								
110								

Πίνακας **Karnaugh**

Στη περίπτωση που έχουμε έξη μεταβλητές ο αριθμός των ελαχιστόρων και κατά συνέπεια των τετραγώνων του πίνακα θα είναι πάλι 2^N όπου Ν αριθμός των μεταβλητών.

Στη περίπτωση των 6 μεταβλητών θα έχουμε 64 ελαχιστόρους και κατά συνέπεια 64 τετράγωνα στο πίνακα **Karnaugh.** Μπορούμε σύμφωνα με τη διάταξη των τετραγώνων να πούμε ότι ο πίνακας **Karnaugh** μιας συνάρτησης 6 μεταβλητών αποτελείται από τέσσερεις πίνακες τεσσάρων μεταβλητών διατεταγμένοι όπως οι σελίδες ενός βιβλίου.

Όταν το βιβλίο κλείσει κατά μήκος των διπλών γραμμών δύο φορές οι πίνακες τεσσάρων μεταβλητών θα έλθουν ο ένας πάνω στο άλλο. Θα έχουμε 4 δεκαεξάδες η μία πάνω στην άλλη, δηλαδή ένα πίνακα Karnaugh τρισδιάστατο, ή ένα πίνακα τεσσάρων ορόφων με 16 τετράγωνα για τον κάθε όροφο. Έτσι τα τετράγωνα που πέφτουν το ένα πάνω στο άλλο θα θεωρούνται γειτονικά όπως και προηγουμένως. Βλέπουμε παρακάτω ένα πίνακα Karnaught 6 μεταβλητών με τους αντίστοιχους ελαχιστόρους σε κάθε ένα από τα 64 τετράγωνα.

$\overline{A}$, $\overline{B}$

CD \ EF	00	01	11	10
00	m_0	m_1	m_3	m_2
01	m_4	m_5	m_7	m_6
11	m_{12}	m_{13}	m_{15}	m_{14}
10	m_8	m_9	m_{11}	m_{10}

$\overline{A}$, B

CD \ EF	00	01	11	10
00	m_{16}	m_{17}	m_{19}	m_{18}
01	m_{20}	m_{21}	m_{23}	m_{22}
11	m_{28}	m_{29}	m_{31}	m_{30}
10	m_{24}	m_{25}	m_{27}	m_{26}

A, $\overline{B}$

CD \ EF	00	01	11	10
00	m_{32}	m_{33}	m_{35}	m_{34}
01	m_{36}	m_{37}	m_{39}	m_{38}
11	m_{44}	m_{45}	m_{47}	m_{46}
10	m_{40}	m_{41}	m_{43}	m_{42}

A, B

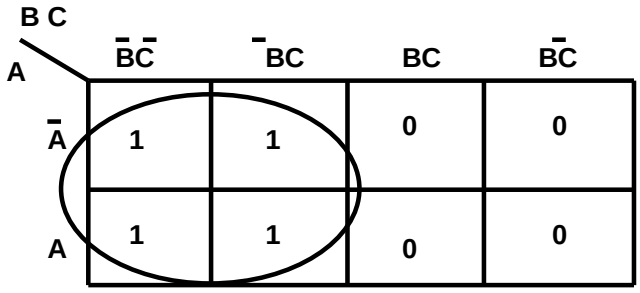
CD \ EF	00	01	11	10
00	m_{48}	m_{49}	m_{51}	m_{50}
01	m_{52}	m_{53}	m_{55}	m_{54}
11	m_{60}	m_{61}	m_{63}	m_{62}
10	m_{56}	m_{57}	m_{59}	m_{58}

Άσκηση

Να απλοποιηθεί η συνάρτηση F(A,B,C)=Σ(0,1,4,5). Όπως βλέπουμε η παραπάνω συνάρτηση αν απλοποιηθεί με τον κλασσικό τρόπο θα πάρουμε:

$$\begin{aligned} F(A,B,C) &= \overline{A}\overline{B}\overline{C} + \overline{A}\overline{B}C + A\overline{B}\overline{C} + A\overline{B}C \\ &= \overline{A}\overline{B}\cdot(\overline{C} + C) + A\overline{B}\cdot(\overline{C} + C) \\ &= \overline{A}\overline{B} + A\overline{B} \\ &= (\overline{A} + A)\cdot\overline{B} \\ &= \overline{B} \end{aligned}$$

Για να φτάσουμε σε αυτό το αποτέλεσμα χρησιμοποιήσαμε τη σχέση του ουδέτερου στοιχείου του πολλαπλασιασμού $X + \overline{X} = 1$ δύο φορές. Ας δούμε τι γίνεται με τον πίνακα Karnaugh.

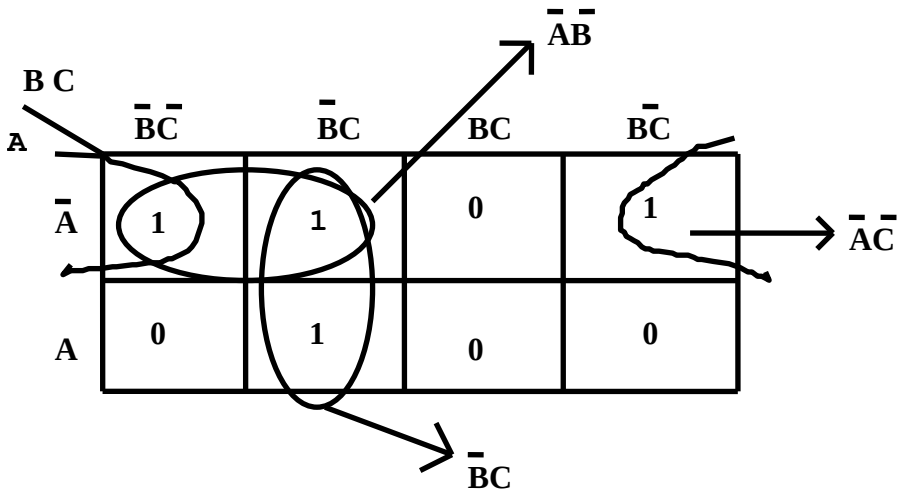


Τα τέσσερα τετράγωνα που περιέχουν τους ελαχιστόρους m0, m1, m4, m5 είναι γειτονικά. Κατά συνέπεια όλες οι μεταβλητές που αλλάζουν τιμή πηγαίνοντας από το ένα τετράγωνο του πίνακα στο άλλο θα απλοποιηθούν, δηλαδή θα απαλειφθούν.

Θα παραμείνει μόνο η μεταβλητή $\overline{B}$ γιατί είναι η μόνη που δεν αλλάζει μέσα στα τέσσερα αυτά τετράγωνα. Βλέπουμε ότι το αποτέλεσμα είναι το ίδιο που θα προέκυπτε αν κάναμε πράξεις. Το πρόβλημα που παρουσιάζεται είναι ότι σε περίπτωση συναρτήσεων πολλών μεταβλητών με πολλούς όρους είναι εύκολο να κάνουμε λάθη ενώ με το χάρτη η κατάσταση είναι υπό έλεγχο. Σε κάθε περίπτωση η διαδικασία με τη μέθοδο του χάρτη είναι πιο σύντομη και πιο σίγουρη.

Άσκηση

Να απλοποιηθεί με τη βοήθεια του χάρτη η παρακάτω λογική συνάρτηση **F(A,B,C)=Σ(0,1,2,5)**. Όπως βλέπουμε η παραπάνω συνάρτηση ορίζεται από τους ελαχιστόρους $\mathbf{m_0+m_1+m_2+m_5}$. Ο χάρτης αυτής της συνάρτησης είναι ο παρακάτω.



Πίνακας Karnaugh

Βλέπουμε πώς τα ακραία τετράγωνα είναι γειτονικά και μπορούν εύκολα να συνδυαστούν. Έτσι η αρχική συνάρτηση από τέσσερις όρους δύο μεταβλητών που είχε στην αρχή μετά την απλοποίηση έγινε

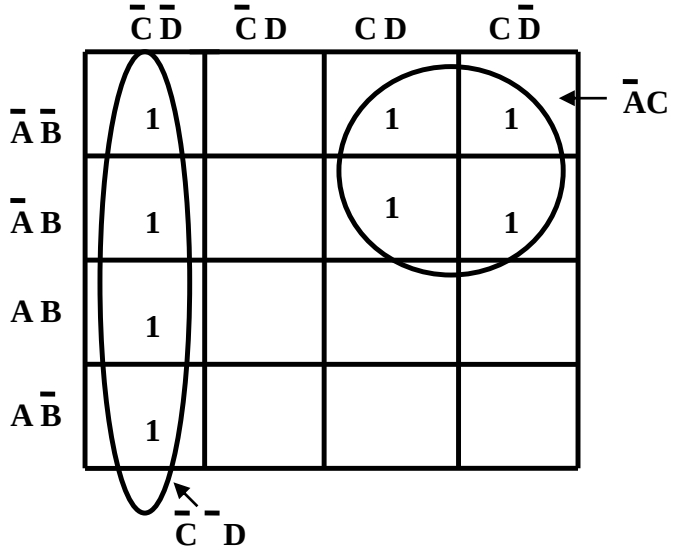
$$F(A,B,C) = \bar{A}\bar{B} + \bar{A}\bar{C} + \bar{B}C$$

πολύ πιο απλή για οιανδήποτε υλοποίηση με πύλες. Αυτό που δεν πρέπει να ξεχνάμε είναι ότι ο αριθμός των τετραγώνων που συνδυάζουμε είναι πάντα δύναμη του δύο δηλαδή δύο, τέσσερα, οκτώ, δεκάξι κ.λ.π.

Άσκηση

Δίνεται η συνάρτηση Boole F(A,B,C,D)=Σ(0,2,3,4,6,7,8,12) να απλοποιηθεί με τη βοήθεια του πίνακα Karnaugh.

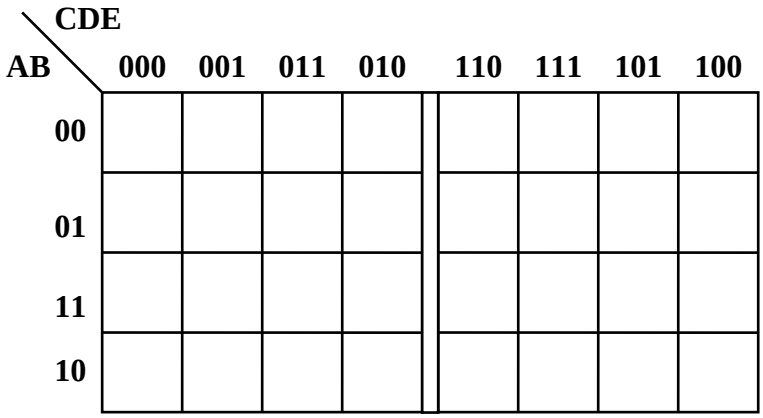
Έχοντας σαν δεδομένο την κανονική μορφή μιας λογικής συνάρτησης μπορούμε εύκολα να συμπληρώσουμε τον χάρτη της ή τον πίνακα **Karnaugh** όπως φαίνεται παρακάτω.



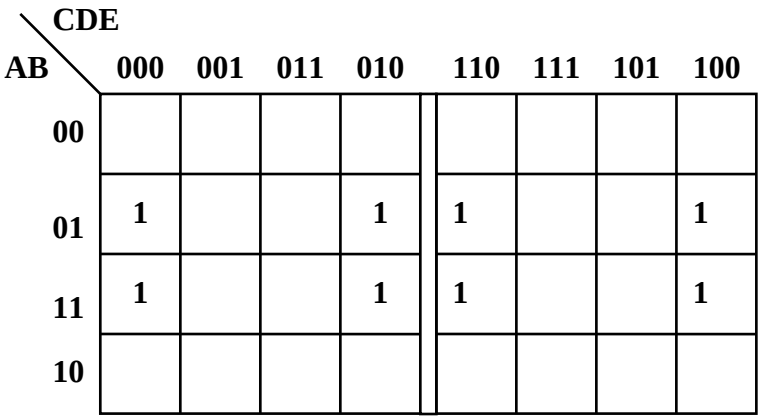
Άρα $F(A,B,C,D)=\Sigma(0,2,3,4,6,7,8,12)=\bar{A}C+\bar{C}\bar{D}$

Άσκηση: Να απλοποιηθεί η παρακάτω συνάρτηση πέντε μεταβλητών F(A,B,C,D,E)=Σ[8,10,12,14,24,26,28,30].

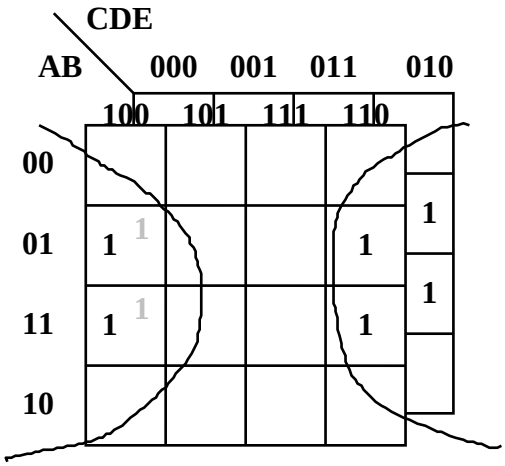
Ο πίνακας Karnaugh για μια συνάρτηση 5 μεταβλητών είναι ο παρακάτω:



Είναι ένας πίνακας διπλής εισόδου όπως και τους άλλους, σε κάθε τετράγωνο αντιστοιχεί ένας δυαδικός αριθμός που είναι τελικά ο αντίστοιχος ελαχιστόρος. Αν τον συμπληρώσουμε με τους ελαχιστόρους και στη συνέχεια με αυτούς της συνάρτησης θα πάρουμε:



Αν θυμηθούμε αυτά που είπαμε προηγουμένως ότι δηλαδή όταν διπλώσει ο πίνακας γύρω από τη διπλή γραμμή θα φτιάξουμε κατά κάποιο τρόπο ένα διώροφο πίνακα τεσσάρων μεταβλητών και τα τετράγωνα που ακουμπάνε μεταξύ τους είναι γειτονικά μπορούμε να δούμε ότι θα συνδυάσουμε τελικά 8 τετράγωνα και θα απλοποιήσουμε 3 μεταβλητές. Την Α τη C, και τη D. Θα μείνει τελικά:



$$F[A, B, C, D, E] = B\bar{E}$$

Βλέπουμε τη σημασία καλής χρήσης του πίνακα Karnaugh. Μία συνάρτηση 8 όρων των 5 μεταβλητών κάθε ένας μετά από μία σωστή απλοποίηση έγινε ένας όρος δύο μεταβλητών με πολύ εύκολη υλοποίηση.

1η ΣΕΙΡΑ ΑΣΚΗΣΕΩΝ και ΛΥΣΕΙΣ

1. Βρείτε τις κανονικές μορφές των παρακάτω συναρτήσεων:

$$F1(x,y,z) = xy + y\bar{z}$$

$$F2(x,y,z) = xy + xy\bar{z} + yz$$

$$F_3(A,B,C,D) = \bar{A}BC\bar{D} + ABC + BC\bar{D} + BC$$

Απαντήσεις

Βλέπουμε ότι η συνάρτηση F1 ενώ είναι μια συνάρτηση τριών μεταβλητών ορίζεται αναλυτικά από ένα δύο όρους των δύο μεταβλητών. Για να εισάγουμε σε κάθε όρο την μεταβλητή που λείπει αρκεί να πολλαπλασιάσουμε με 1 κάθε όρο βάση της ιδιότητας $a + \bar{a} = 1$. Θα έχουμε λοιπόν.

$$F1(x,y,z) = xy + y\bar{z} = xy(z + \bar{z}) + (x + \bar{x})y\bar{z}$$

$$F1(x,y,z) = xyz + xy\bar{z} + xy\bar{z} + \bar{x}y\bar{z}$$

Βλέπουμε ότι ο 2ος και ο 3ος όρος είναι ίδιοι και σε αυτή την περίπτωση τον χρησιμοποιούμε μόνο μία φορά. Άρα

$$F1(x,y,z) = \Sigma(2,6,7)$$

Για τους ίδιους λόγους θα έχουμε για την συνάρτηση F2(x,y,z) ότι

$$F2(x,y,z) = xy(z + \bar{z}) + xy\bar{z} + (x + \bar{x})yz$$

$$F2(x,y,z) = xyz + xy\bar{z} + xy\bar{z} + xyz + \bar{x}yz$$

$$F2(x,y,z) = xyz + xy\bar{z} + \bar{x}yz$$

$$F2(x,y,z) = \Sigma(3,6,7)$$

Εδώ έχουμε 2 όρους από 2 φορές οπότε οι 5 όροι γίνονται 3 για την F2.

Για την συνάρτηση F3(A,B,C,D) θα έχουμε αυτή τη φορά ότι:

$$F3(A,B,C,D) = \bar{A}BC\bar{D} + ABC(D + \bar{D}) + (A + \bar{A})BC\bar{D} + (A + \bar{A})BC(D + \bar{D})$$

$$F3(A,B,C,D) = \overline{A}BC\overline{D} + ABCD + ABC\overline{D} + ABC\overline{D} + \overline{A}BC\overline{D} + ABCD + ABC\overline{D} + \overline{A}BCD + \overline{A}BC\overline{D}$$

$$F3(A,B,C,D) = \overline{A}BC\overline{D} + ABCD + ABC\overline{D} + ABCD + \overline{A}BCD$$

$$F3(A,B,C,D) = \Sigma(6,7,14,15)$$

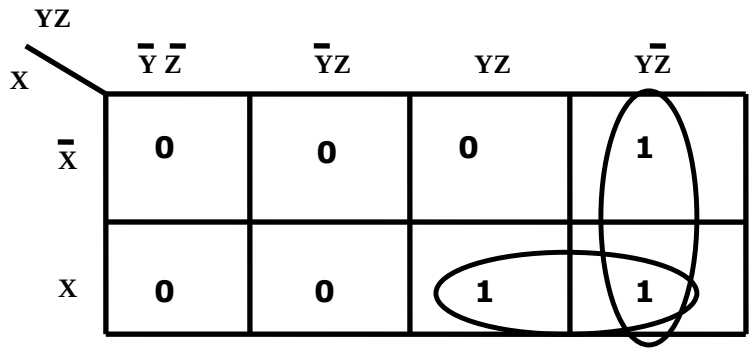
2. Απλοποιήστε όπου αυτό είναι εφικτό με τη βοήθεια του πίνακα Karnaugh τις προηγούμενες συναρτήσεις F1,F2, F3.

$F1(x,y,z) = \Sigma(2,6,7)$

$F2(x,y,z) = \Sigma(3,6,7)$

$F3(A,B,C,D) = \Sigma(6,7,14,15)$

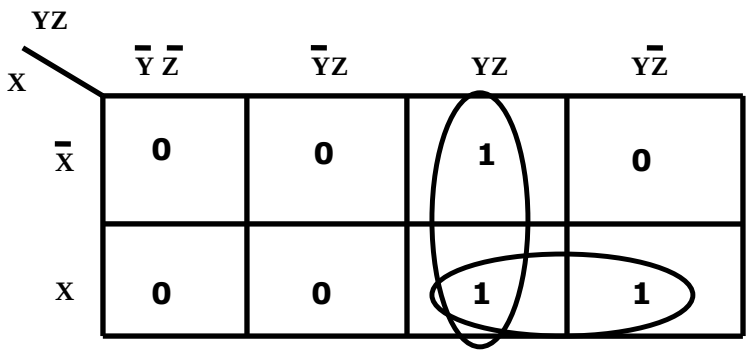
Ο πίνακας Karnaugh της συνάρτησης F1(x,y,z)



Η συνάρτηση απλοποιείται όπως φαίνεται στον προηγούμενο πίνακα Karnaugh της F1(x,y,z) και γίνεται:

$$F1(x,y,z) = xy + y\overline{z}$$

Ο πίνακας Karnaugh της συνάρτησης F2(x,y,y) θα γίνει:



Και μετά την απλοποίηση $F2(x,y,z) = xy + yz$

Ο πίνακας Karnaugh της συνάρτησης F3(A,B,C,D)

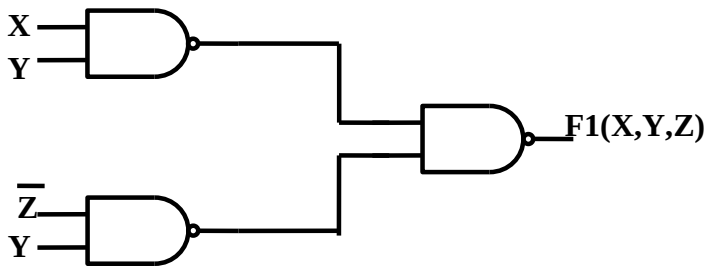
	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	0	0	0	0
$\bar{A}B$	0	0	1	1
AB	0	0	1	1
$A\bar{B}$	0	0	0	0

Και μετά την απλοποίηση $F3(A,B,C,D) = BC$

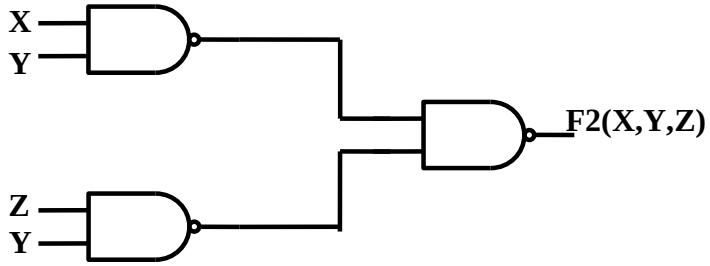
3. Σχεδιάστε τις απλοποιημένες συναρτήσεις F1,F2, F3 αποκλειστικά με πύλες NAND.

Με δεδομένη την απλοποίηση των τριών προηγούμενων συναρτήσεων η υλοποίηση τους αποκλειστικά με πύλες NAND είναι εύκολη, με την προϋπόθεση ότι θα θεωρούμε δεδομένες τις αντιστροφές στην είσοδο.

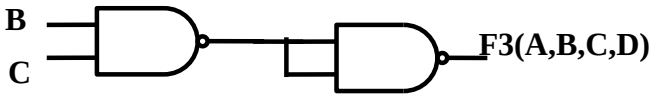
$F1(x,y,z) = xy + y\bar{z}$



$F2(x,y,z) = xy + yz$

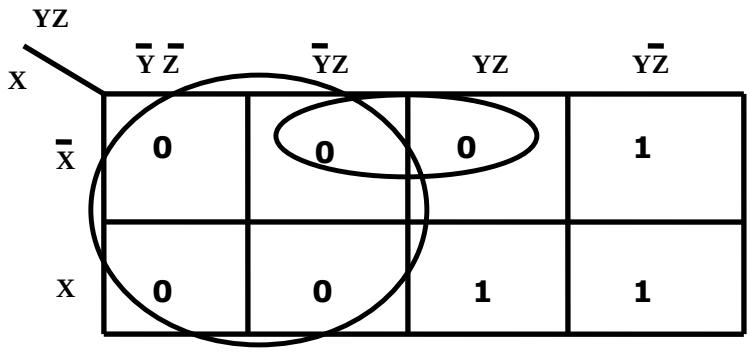


$F3(A,B,C,D) = BC$



4. Σχεδιάστε τις απλοποιημένες συναρτήσεις F1,F2, F3 αποκλειστικά με πύλες NOR

Είχαμε δει ότι για να σχεδιάσουμε μία συνάρτηση αποκλειστικά με πύλες NOR πρέπει στον πίνακα Karnaugh της αρχικής συνάρτησης να συνδυάσουμε τα 0 και να απλοποιήσουμε κατά συνέπεια την συμπληρωματικά της συνάρτησης $\overline{F}$. Αφού απλοποιήσουμε την συμπληρωματική συνάρτησης σαν άθροισμα γινομένων συμπληρώνουμε ξανά, οπότε πέφτουμε ξανά πάλι πάνω στην F αλλά στην μορφή γινομένου αθροισμάτων, μορφή που χρειαζόμαστε για την υλοποίηση με πύλες NOR αποκλειστικά. Ας δούμε όμως για κάθε συνάρτηση τι θα συμβεί. Ο πίνακας Karnaugh της συνάρτησης F1(x,y,z)



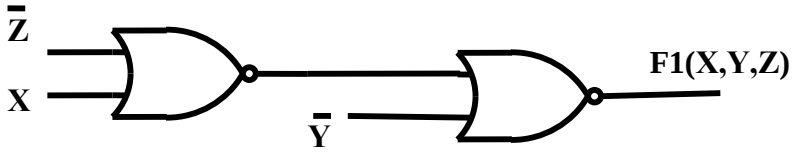
Με τον συνδυασμό των 0 θα πάρουμε

$$\overline{F1} = \overline{y} + \overline{x}z$$

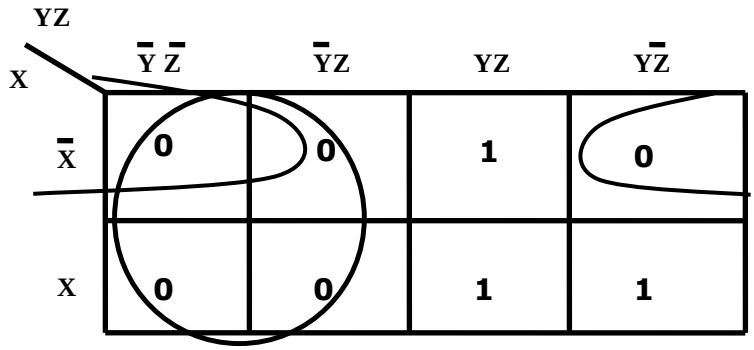
και αν συμπληρώσουμε την παραπάνω σχέση θα έχουμε

$$\overline{F1} = \bar{y} + \bar{x}z \Rightarrow \overline{\overline{F1}} = \overline{\bar{y} + \bar{x}z} = \bar{\bar{y}}\,\bar{\bar{x}}z = y(\bar{\bar{x}} + \bar{z}) = y(x + \bar{z})$$

Δηλαδή $F1 = y(x + \bar{z})$



Ο πίνακας Karnaugh της συνάρτησης F2(x,y,y) θα είναι:

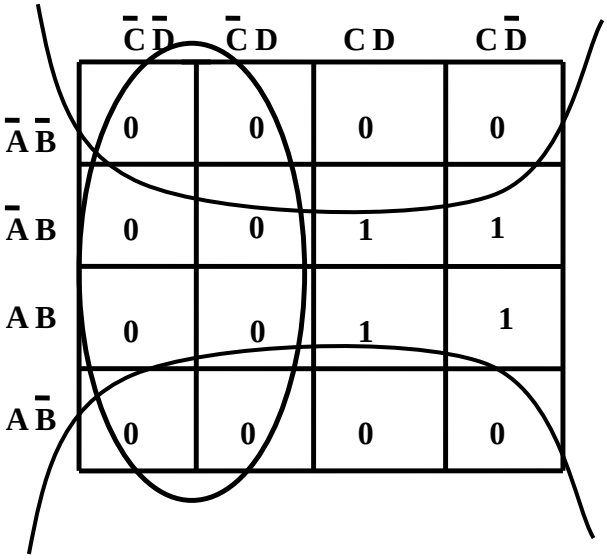


Και μετά την απλοποίηση $\overline{F2}(x,y,z) = \bar{y} + \bar{x}\bar{z}$ Αν συμπληρώσουμε την παραπάνω σχέση θα βρούμε την F2 σε γινόμενο αθροισμάτων.

$$\overline{F2}(x,y,z) = \bar{y} + \bar{x}\bar{z} \Rightarrow \overline{\overline{F2}}(x,y,z) = \overline{\bar{y} + \bar{x}\bar{z}} \Rightarrow$$

$$\overline{\overline{F2}}(x,y,z) = \bar{\bar{y}}(\bar{\bar{x}}\bar{\bar{z}}) = \bar{\bar{y}}(\bar{\bar{x}} + \bar{\bar{z}}) = y(x + z) \Rightarrow F2(x,y,z) = y(x + z)$$

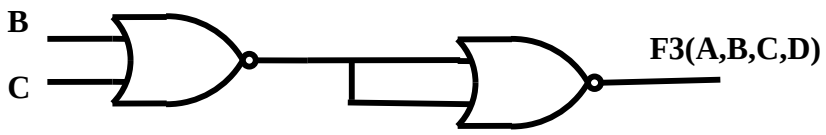
Ο πίνακας Karnaugh της συνάρτησης F3(A,B,C,D) είναι:



Συνδυάζοντας τα 0 βλέπουμε ότι έχουμε 2 συνδυασμούς των 8 BIT οπότε απλοποιούνται 3 μεταβλητές και θα έχουμε τελικά $\overline{F}3 = \overline{B} + \overline{C}$. Και θα έχουμε :

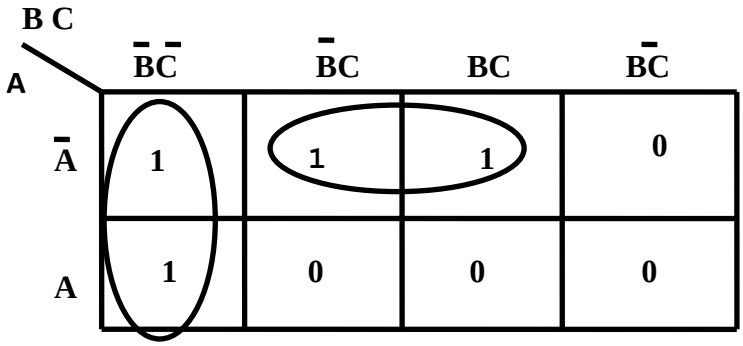
$$\overline{F}3 = \overline{B} + \overline{C} \Rightarrow \overline{\overline{F}}3 = \overline{\overline{B} + \overline{C}} \Rightarrow F3 = \overline{\overline{B}}\,\overline{\overline{C}} \Rightarrow F3 = \overline{\overline{B}} + \overline{\overline{C}} \Rightarrow F3 = B + C$$

Και το κύκλωμα θα είναι



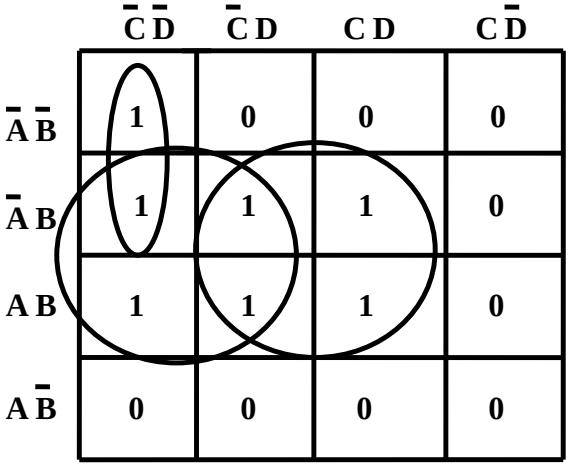
5. Απλοποιήστε με τη βοήθεια του πίνακα Karnaugh τις παρακάτω συναρτήσεις F(A,B,C)=Σ(0,1,3,4) , G(A,B,C,D)=Σ(0,4,5,7,12,13,15) , H(A,B,C,D)=Σ(0,2,4,6,10,14)

Ας δούμε μία μία τις συναρτήσεις και τον πίνακα Karnaugh της κάθε συνάρτησης παρακάτω και τους συνδυασμούς γειτονικών τετραγώνων F(A,B,C)=Σ(0,1,3,4).

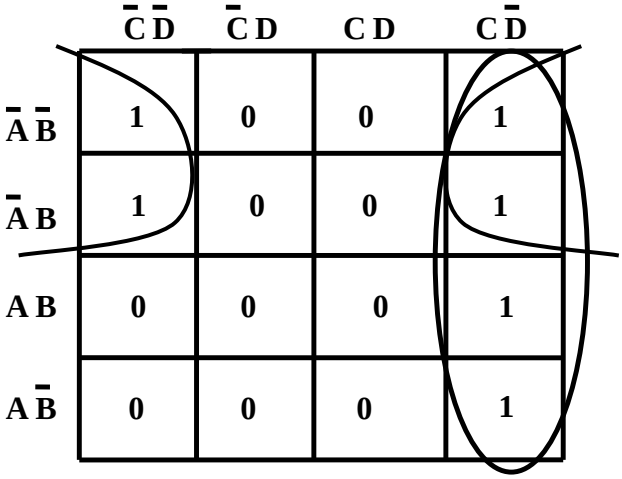


Με την απλοποίηση θα πάρουμε: $F(A,B,C) = \bar{B}\bar{C} + \bar{A}C$

Για τη συνάρτηση G(A,B,C,D)=Σ(0,4,5,7,12,13,15) θα έχουμε τον παρακάτω πίνακα Karnaugh και μετά την απλοποίηση θα έχουμε $G(A,B,C,D) = \bar{A}C\bar{D} + B\bar{C} + BD$



Για τη συνάρτηση H(A,B,C,D)=Σ(0,2,4,6,10,14)

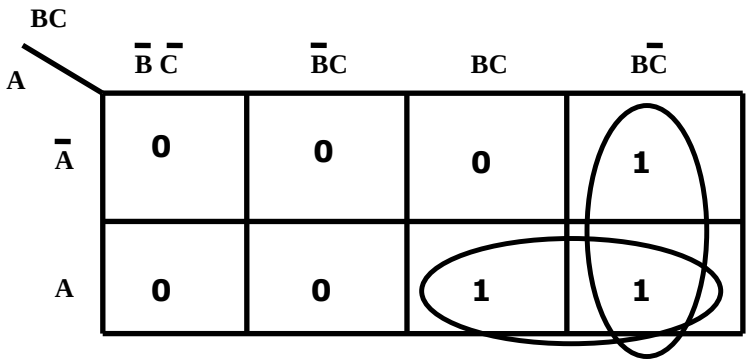


Από τον πίνακα Karnaugh και μετά την απλοποίηση θα έχουμε:

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

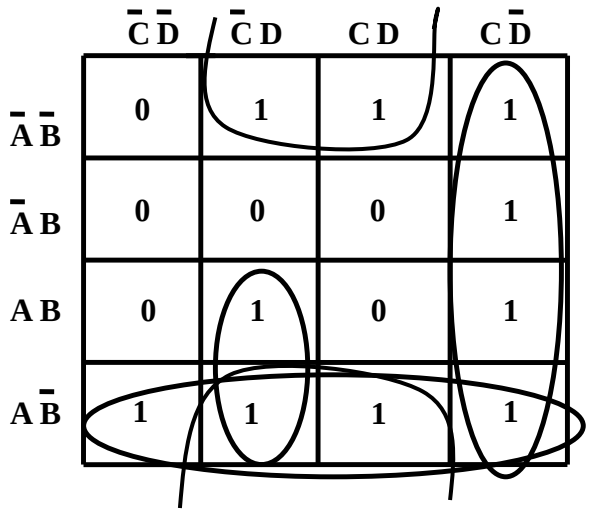
6. Απλοποιήστε με τη βοήθεια του πίνακα Karnaugh τις παρακάτω συναρτήσεις F_1(A,B,C)=Π(0,1,3,4,5) , G_1(A,B,C,D)=Π(0,4,5,7,12,15), H_1(A,B,C,D)=Π(0,2,4,6,10,11,12, 14).

Η συνάρτηση F1(A,B,C)=Π(0,1,3,4,5)=Σ(2,6,7). Υπενθυμίζουμε ότι για να πάμε από την μια κανονική μορφή στην άλλη αρκεί να πάρουμε τους δείκτες που λείπουν. Ο πίνακας Karnaugh της παραπάνω συνάρτησης φαίνεται παρακάτω. Οι συνδυασμοί των άσσων θα είναι δύο με δύο τετράγωνα ο καθένας. Βλέπουμε παρακάτω την απλοποίηση.



$$F1(A,B,C) = AB + B\overline{C}$$

Η επόμενη G_1(A,B,C,D)=Π(0,4,5,7,12,15)=Σ(1,2,3,6,8,9,10,11,13,14) συνάρτηση έχει τον παρακάτω πίνακα Karnaugh.

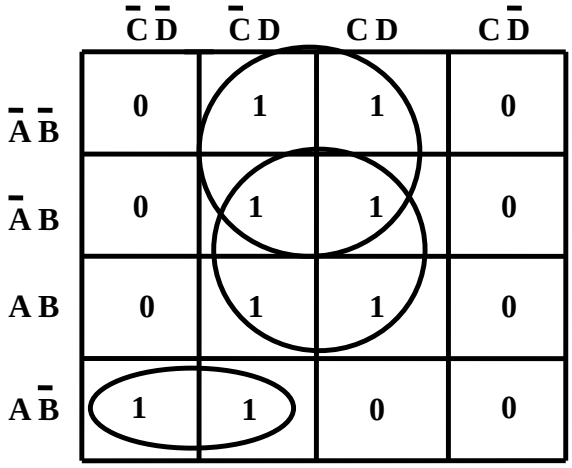


Μετά την απλοποίηση

$$G1(A,B,C,D) = A\overline{B} + A\overline{C}D + \overline{B}D + C\overline{D}$$

Η τελευταία συνάρτηση

$H_1(A,B,C,D)=\Pi(0,2,4,6,10,11,12,14)=\Sigma(1,3,5,7,8,9,13,15)$



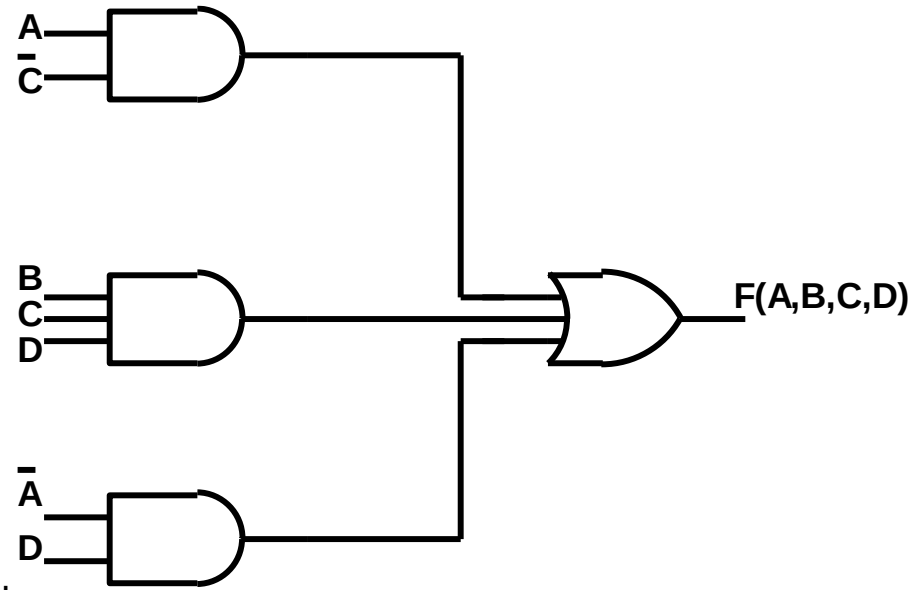
Μετά την απλοποίηση $H1(A,B,C,D) = \overline{A}D + BD + A\overline{B}\overline{C}$

Υλοποίηση τύπου AND-OR και NAND

Για την υλοποίηση των συναρτήσεων Boole είπαμε ότι πρέπει να έχουμε την ελάχιστη μορφή της συνάρτησης. Η ελάχιστη αυτή μορφή είναι συνάρτηση του τρόπου που θέλουμε να υλοποιήσουμε το λογικό κύκλωμα. Αν δηλαδή θέλουμε μία υλοποίηση **AND-OR** η **NAND** τότε η ελάχιστη μορφή πρέπει να είναι άθροισμα γινομένων. Αν θέλουμε μία υλοποίηση **OR-AND** η **NOR** τότε η απλοποιημένη μορφή της συνάρτησης πρέπει να είναι γινόμενα αθροισμάτων. Τα παραπάνω πηγάζουν από τον τρόπο παρουσίασης των λογικών συναρτήσεων δηλαδή από τις κανονικές μορφές οι οποίες είναι δύο όπως έχουμε δει, αθροίσματα γινομένων (άθροισμα ελαχιστόρων), και γινόμενα αθροισμάτων (γινόμενο μεγιστόρων). Κατά συνέπεια και οι απλοποιημένες μορφές μιας λογικής συνάρτησης θα μπορούν να παρουσιαστούν με δύο διαφορετικούς τρόπους. Ας πάρουμε για παράδειγμα την παρακάτω συνάρτηση:

$$\mathbf{F(A,B,C,D) = A\overline{C} + BCD + \overline{A}D}$$

Το κύκλωμα που υλοποιεί αυτή τη συνάρτηση φαίνεται στο παρακάτω σχήμα και είναι του τύπου AND-OR.

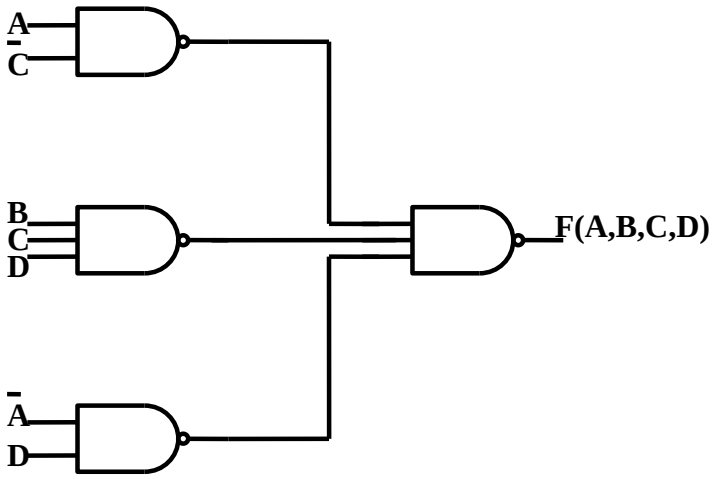


Μπορούμε όμως να γράψουμε τη συνάρτηση F(A,B,C,,D) όπως παρακάτω για να την υλοποιήσουμε με πύλες NAND.

$$\begin{aligned} \mathbf{F(A,B,C,D)} &= \mathbf{A\overline{C} + BCD + \overline{A}D} \\ &= \overline{\overline{\mathbf{A\overline{C}}} + \overline{\mathbf{BCD}} + \overline{\mathbf{\overline{A}D}}} \\ &= \overline{\mathbf{X1 \cdot X2 \cdot X2}} \end{aligned}$$

$$\begin{aligned} \mathbf{X1} &= \overline{\mathbf{A\overline{C}}} \\ \mathbf{X2} &= \overline{\mathbf{BCD}} \\ \mathbf{X3} &= \overline{\mathbf{\overline{A}D}} \end{aligned}$$

Και η υλοποίηση φαίνεται στο παρακάτω σχήμα αποκλειστικά με πύλες NAND με την προϋπόθεση ότι οι μεταβλητές είναι διαθέσιμες αντεστραμμένες στην είσοδο ή ότι αντιστροφείς είναι διαθέσιμοι στην είσοδο του κυκλώματος κατά την υλοποίηση του. Αν δεν είναι διαθέσιμα τα παραπάνω και θέλουμε να υλοποιηθεί το κύκλωμα αποκλειστικά με NAND οι αντιστροφείς μπορούν να υλοποιηθούν από πύλες NAND βραχυκυκλώνοντας απλά τις εισόδους της πύλης.



Υλοποίηση αποκλειστικά με πύλες NAND της προηγούμενης συνάρτησης.

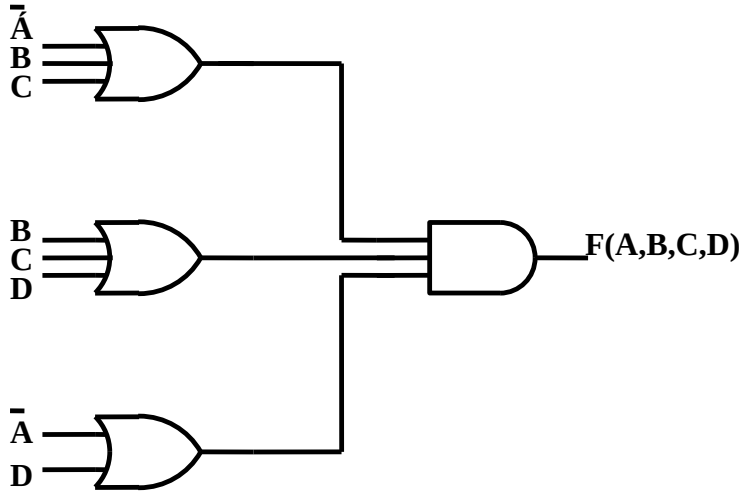
Και τα δύο προηγούμενα κυκλώματα είναι δύο επιπέδων. Αυτό σημαίνει ότι έχουμε δύο σειρές πυλών, ή δύο BLOCK πυλών που υλοποιούν τη λογική συνάρτηση και ότι η καθυστέρηση του κυκλώματος είναι δύο φορές αυτή της πύλης NAND τριών εισόδων, διότι στο παραπάνω κύκλωμα αυτός ο κλάδος είναι ο πιο αργός.

Υλοποίηση τύπου OR-AND και NOR

Εάν μας έχει ζητηθεί να υλοποιήσουμε μία συνάρτηση αποκλειστικά με πύλες AND-OR η NOR τότε η ελαχιστοποιημένη μορφή της λογικής συνάρτησης πρέπει να είναι γινόμενα αθροισμάτων όπως φαίνεται στο παρακάτω παράδειγμα.

$$\mathbf{F(A,B,C,D) = (\overline{A} + B + C)\cdot(B + C + D)\cdot(\overline{A} + D)}$$

Η παρακάτω συνάρτηση υλοποιείται πολύ εύκολα με πύλες OR και μία AND όπως φαίνεται στο παρακάτω σχήμα.

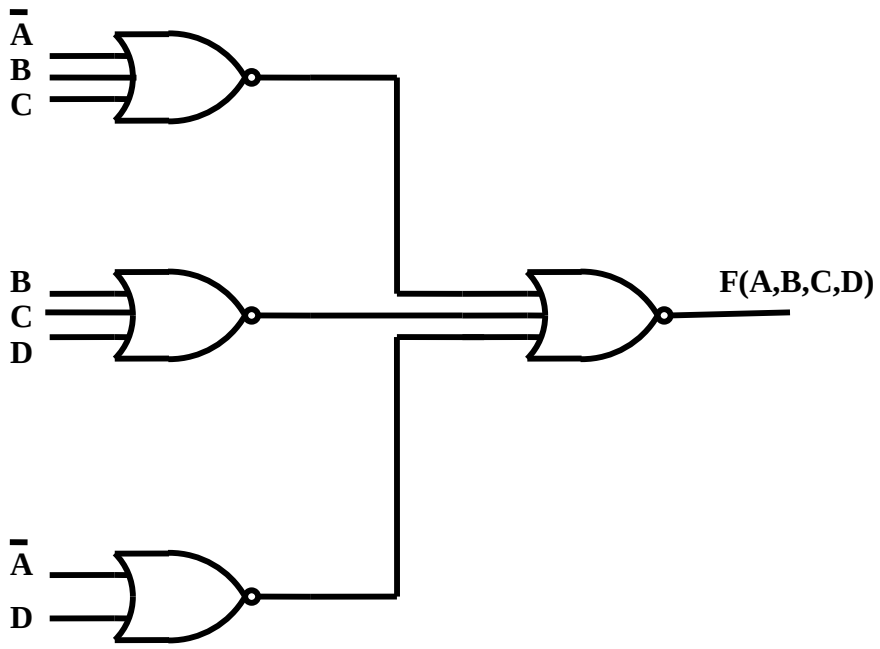


Υλοποίηση OR-AND της παραπάνω λογικής συνάρτησης

Για να υλοποιήσουμε την παραπάνω λογική συνάρτηση αποκλειστικά με πύλες NOR ακολουθούμε την ίδια διαδικασία όπως και προηγουμένως.

$$
\begin{aligned}
\mathbf{F(A,B,C,D)} &= \mathbf{(\overline{A} + B + C)\cdot(B + C + D)\cdot(\overline{A} + D)} \\
&= \overline{\mathbf{(\overline{A} + B + C)\cdot(B + C + D)\cdot(\overline{A} + D)}} \\
&= \overline{\mathbf{Y1\cdot Y2\cdot Y3}} \\
\mathbf{Y1} &= \overline{\mathbf{(\overline{A} + B + C)}} \\
\mathbf{Y2} &= \overline{\mathbf{(B + C + D)}} \\
\mathbf{Y3} &= \overline{\mathbf{(\overline{A} + D)}}
\end{aligned}
$$

Με αυτή τη μορφή η υλοποίηση γίνεται εύκολα αποκλειστικά με πύλες NOR όπως φαίνεται στο παρακάτω σχήμα αν υποθέσουμε όπως και προηγουμένως ότι οι αντιστροφές των μεταβλητών είναι διαθέσιμες στην είσοδο των πυλών. Διαφορετικά και εάν μας έχει ζητηθεί να υλοποιήσουμε μία συγκεκριμένη συνάρτηση αποκλειστικά με πύλες NOR βραχυκυκλώνοντας τις εισόδους μίας πύλης NOR έχουμε ένα αντιστροφέα και μπορούμε να το χρησιμοποιήσουμε όπως έχουμε ήδη δει και με τις πύλες NAND.



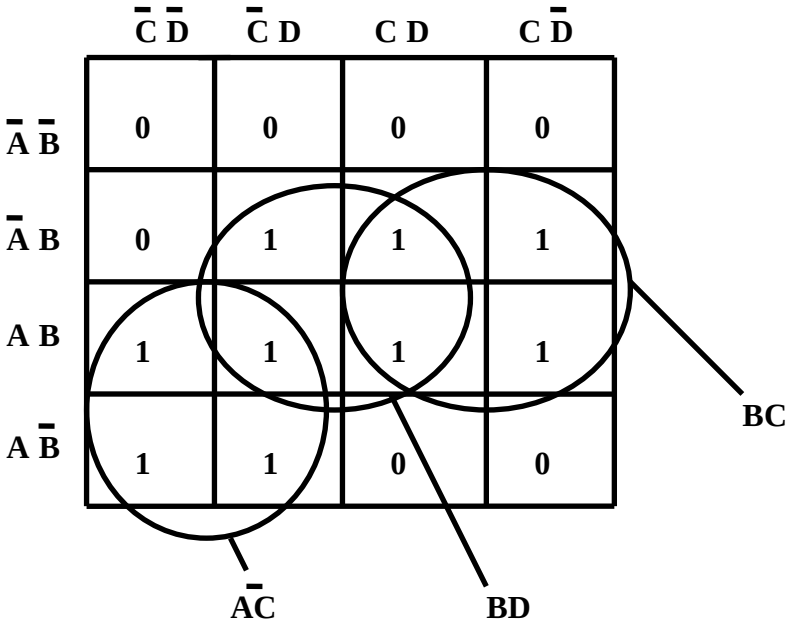
Υλοποίηση αποκλειστικά με πύλες NOR της προηγούμενης λογικής συνάρτησης.

Διεπίπεδες υλοποιήσεις τύπου NAND-NAND και NOR-NOR.

Διεπίπεδες υλοποιήσεις τύπου NAND-NAND και NOR-NOR γίνονται πολύ εύκολα όπως είδαμε και προηγουμένως. Η διαδικασία υλοποίησης συναρτήσεων με αυτό τον τρόπο είναι η παρακάτω.

Απ' ότι είδαμε μέχρι τώρα αν χρησιμοποιήσουμε τον πίνακα **Karnaugh** ή το χάρτη μίας συνάρτησης μπορούμε να τη φέρουμε στην ελάχιστη μορφή σαν άθροισμα γινομένων και να την υλοποιήσουμε αποκλειστικά με πύλες NAND αν μας έχει ζητηθεί. Αν το ζητούμενο είναι να πραγματοποιήσουμε μία διεπίπεδη υλοποίηση τύπου NOR-NOR τότε από τον πίνακα Karnaugh της F συνδυάζω αντί για τους άσσους τα μηδενικά οπότε απλοποιώ σε άθροισμα γινομένων τη συμπληρωματική συνάρτηση $\overline{F}$. Αν συμπληρώσω τη συμπληρωματική της $\overline{\overline{F}} = F$ βρίσκω πάλι στην αρχική συνάρτηση. Με αυτή όμως τη διαδικασία μετατρέπονται τα αθροίσματα γινομένων σε γινόμενα αθροισμάτων και η χρήση αποκλειστικά πυλών NOR είναι δυνατή όπως φαίνεται στο παρακάτω παράδειγμα.

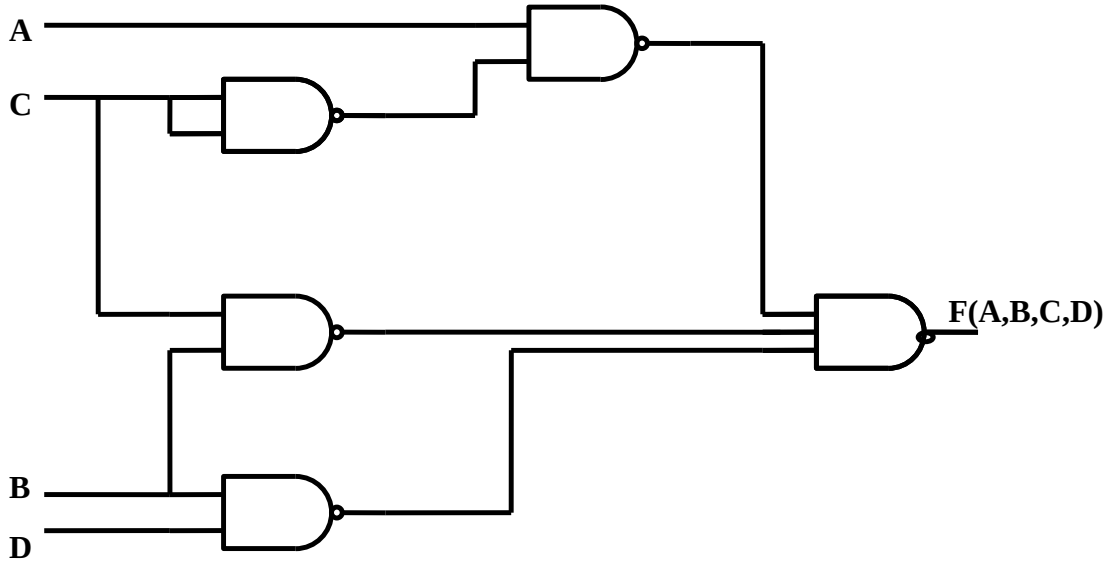
Δίνεται η λογική συνάρτηση F(A,B,C,D)=Σ(5,6,7,8,9,12,13,14,15). Να υλοποιηθεί αποκλειστικά με πύλες NAND και μετά αποκλειστικά με NOR. Υποθέτουμε ότι δεν είναι διαθέσιμες οι αντιστροφές στην είσοδο. Ο πίνακας Karnaugh της παραπάνω συνάρτησης φαίνεται στο παρακάτω σχήμα.



Πίνακας Karnaugh.

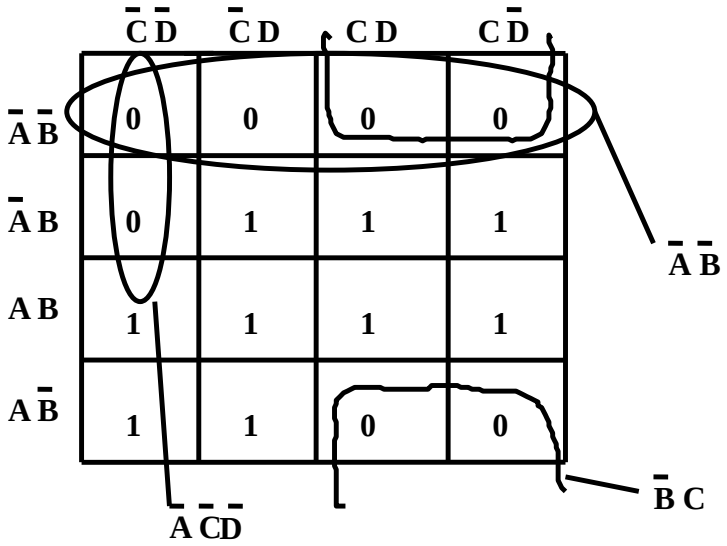
Βλέπουμε ότι ο μέγιστος αριθμός τετραγώνων που μπορούμε να συνδυάσουμε είναι τέσσερα. Σύμφωνα με τους κανόνες που έχουμε δώσει για τη χρήση του πίνακα Karnaugh συνδυάζουμε τους άσσους σε συνδυασμούς 4 τετραγώνων 3 φορές για να μπορέσουμε να συνδυάσουμε όλα τα τετράγωνα τουλάχιστον 1 φορά το καθένα. Στον παραπάνω πίνακα το παραπάνω είναι εφικτό.

Δεν συμβαίνει όμως πάντα. Κατά συνέπεια η απλοποίηση σε άθροισμα γινομένων της συνάρτησης μας δίνει $F(A,B,C,D) = A\bar{C} + BC + BD$ δηλαδή την απλοποιημένη μορφή με τρεις όρους των δύο μεταβλητών. Η υλοποίηση αποκλειστικά με πύλες NAND είναι απλή όπως φαίνεται στο παρακάτω σχήμα. Για την υλοποίηση της συνάρτησης αποκλειστικά με πύλες NOR εφαρμόζουμε την γνωστή διαδικασία που έχει να κάνει με τον συνδυασμό των μηδενικών στον πίνακα Karnaught όπως θα δούμε και παρακάτω.



Υλοποίηση αποκλειστικά με πύλες NAND μίας λογικής συνάρτησης.

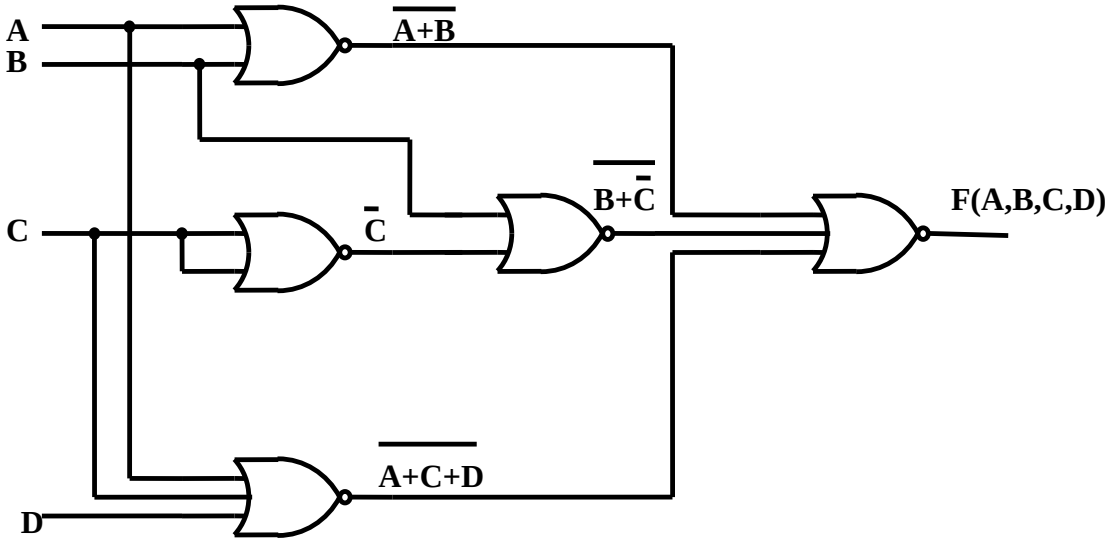
Η απλοποίηση της συμπληρωματικής συνάρτησης φαίνεται στο παρακάτω σχήμα.



Απλοποίηση της συμπληρωματικής συνάρτησης.

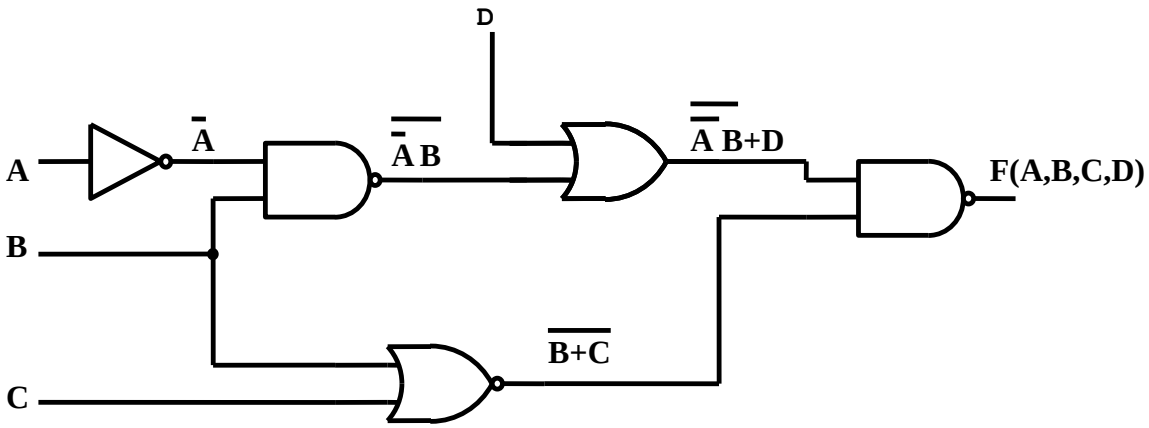
Η απλοποιημένη συμπληρωματική συνάρτηση σαν άθροισμα γινομένων είναι φαίνεται παρακάτω. Βλέπω πόσο εύκολα συμπληρώνοντας τη βρίσκω ξανά μία απλοποιημένη μορφή της κανονικής σαν γινόμενο αθροισμάτων της οποίας η υλοποίηση μπορεί να γίνει αποκλειστικά με πύλες NOR.

$$\bar{F}(A,B,C,D) = \bar{A}\bar{B} + \bar{B}C + \bar{A}C\bar{D} \Rightarrow$$
$$\bar{\bar{F}}(A,B,C,D) = \overline{\bar{A}\bar{B} + \bar{B}C + \bar{A}C\bar{D}} \Rightarrow$$
$$F(A,B,C,D) = (A+B)\cdot(B+\bar{C})\cdot(A+C+D)$$



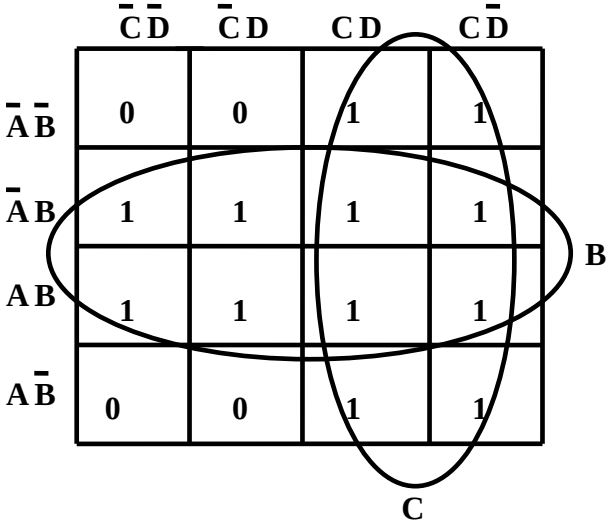
Υλοποίηση της F αποκλειστικά με πύλες NOR.

Δίνεται το κύκλωμα του παρακάτω σχήματος. Βρείτε τη συνάρτηση F(A,B,C,D) στην έξοδο, στην συνέχεια να απλοποιηθεί και να υλοποιηθεί με μία μόνο λογική πύλη δύο εισόδων.

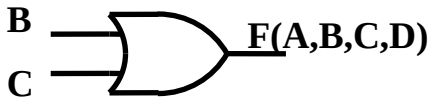


Η συνάρτηση στην έξοδο μετά από απλοποίηση της εξόδου με τις σχέσεις του De Morgan είναι **F(A,B,C,D)=** $\overline{A}B\overline{D} + B + C$. Βλέπω ότι η συνάρτηση είναι σε μία τυχαία μορφή η οποία δεν είναι απαραίτητα η ελάχιστη. Γι αυτόν το λόγο πρέπει να ανατρέξω στον πίνακα Karnaugh ο οποίος δείχνει με σιγουριά αν η παραπάνω μορφή είναι η ελάχιστη. Όπως έχουμε δει για να συμπληρώσουμε το χάρτη μίας συνάρτησης πρέπει να γνωρίζουμε την κανονική της μορφή δηλαδή τους ελαχιστόρους που την ορίζουν. Έχουμε επίσης δει πώς είναι η διαδικασία συμπληρώνοντας τους όρους με τις μεταβλητές που λείπουν μέχρι να αποκτήσουν όλοι οι όροι τον ίδιο αριθμό μεταβλητών.

Εμείς θα αποφύγουμε όλη αυτή τη διαδικασία κάνοντας την αντίστροφη ακριβώς διαδικασία από αυτή της απλοποίησης. Δηλαδή αν έχω ένα όρο με μία μεταβλητή λέω ότι έχει προκύψει από ένα συνδυασμό οκτώ τετραγώνων τα οποία και καθορίζω χωρίς καμία δυσκολία, αν έχω ένα όρο με μία μεταβλητή λέω ότι έχει προκύψει από το συνδυασμό τεσσάρων τετραγώνων κ.λ.π. Με αυτή τη διαδικασία συμπληρώνω το χάρτη της συνάρτησης και βλέπω αν η μορφή που βρήκα είναι η ελάχιστη. Ο όρος B προέκυψε από συνδυασμό των τετραγώνων με τους ελαχιστόρους m4, m5, m6, m7, m12, m13, m14, m15 και ο όρος C από το συνδυασμό των τετραγώνων με τους ελαχιστόρους m2, m3, m6, m7, m10, m11, m14, και m15. Ο όρος που μένει των τριών μεταβλητών έχει προκύψει από το συνδυασμό των τετραγώνων με τους ελαχιστόρους m4 και m6. Έτσι ο χάρτης της συνάρτησης είναι αυτός του παρακάτω σχήματος.

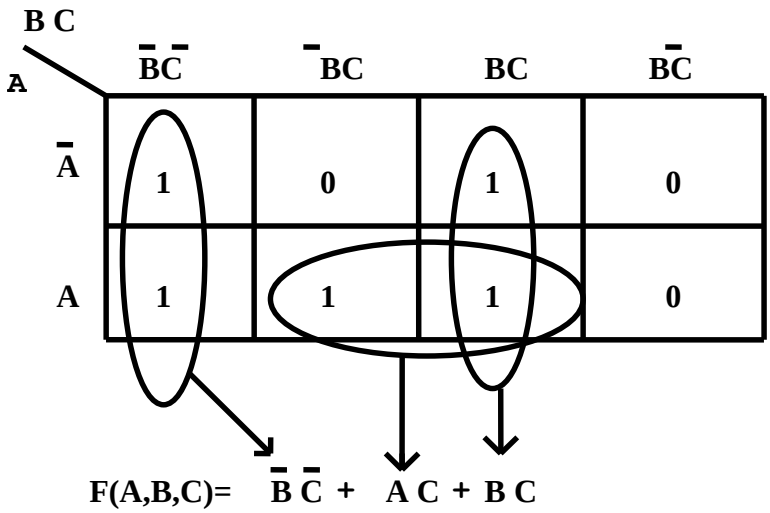


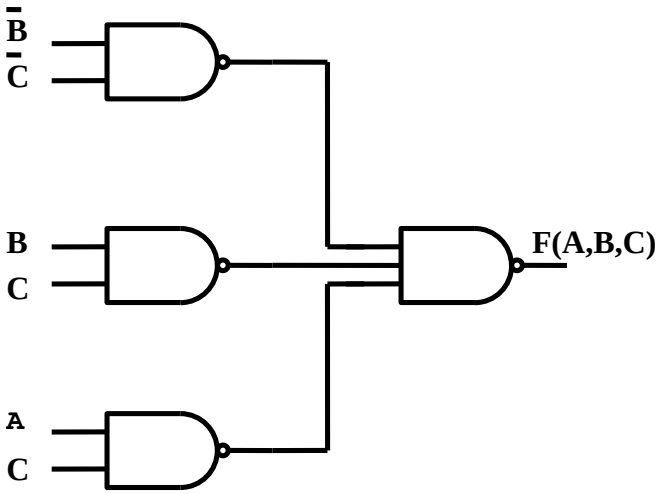
Βλέπουμε ότι αρχική συνάρτηση έγινε F(A,B,C,D)=B+C και υλοποιείται με μία μόνο πύλη OR όπως φαίνεται στο παρακάτω σχήμα. Έτσι βλέπουμε πόσο απλοποιείται το αρχικό κύκλωμα.

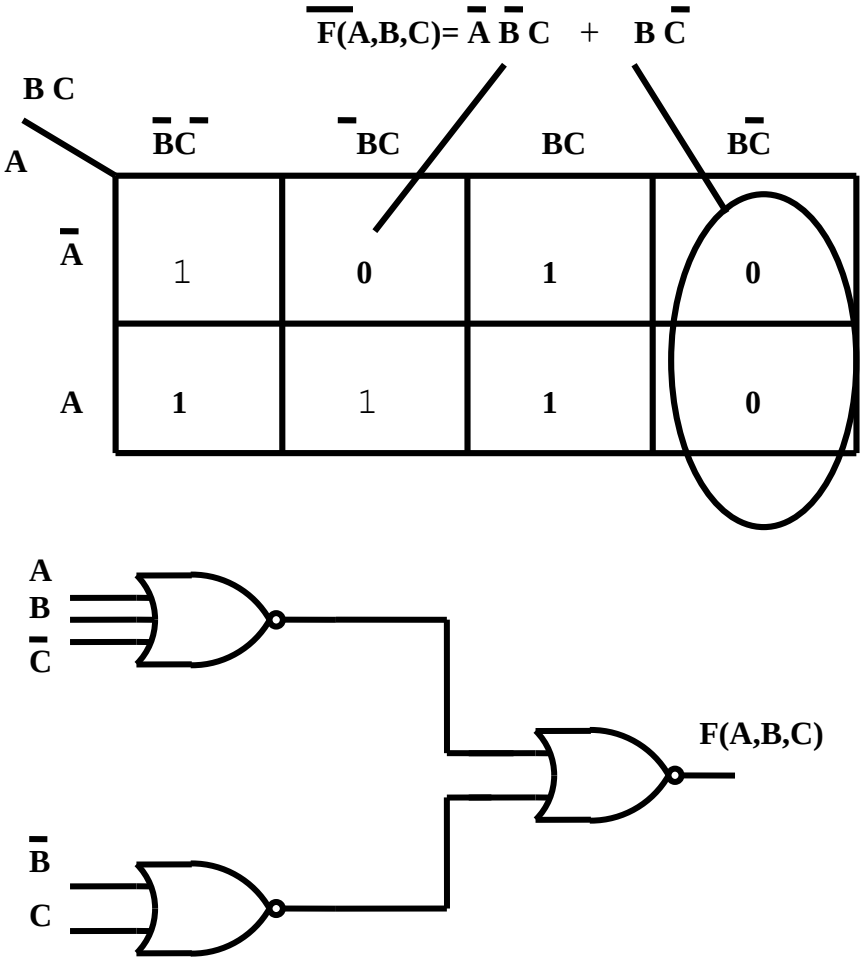


Δίνεται η λογική συνάρτηση F(A,B,C)=Σ(0,3,4,5,7). Να απλοποιηθεί σε άθροισμα γινομένων και σε γινόμενο αθροισμάτων. Αν υποθέσουμε ότι οι πύλες NOR και NAND έχουν την ίδια κατανάλωση υλοποιήστε το πιο οικονομικό κύκλωμα.

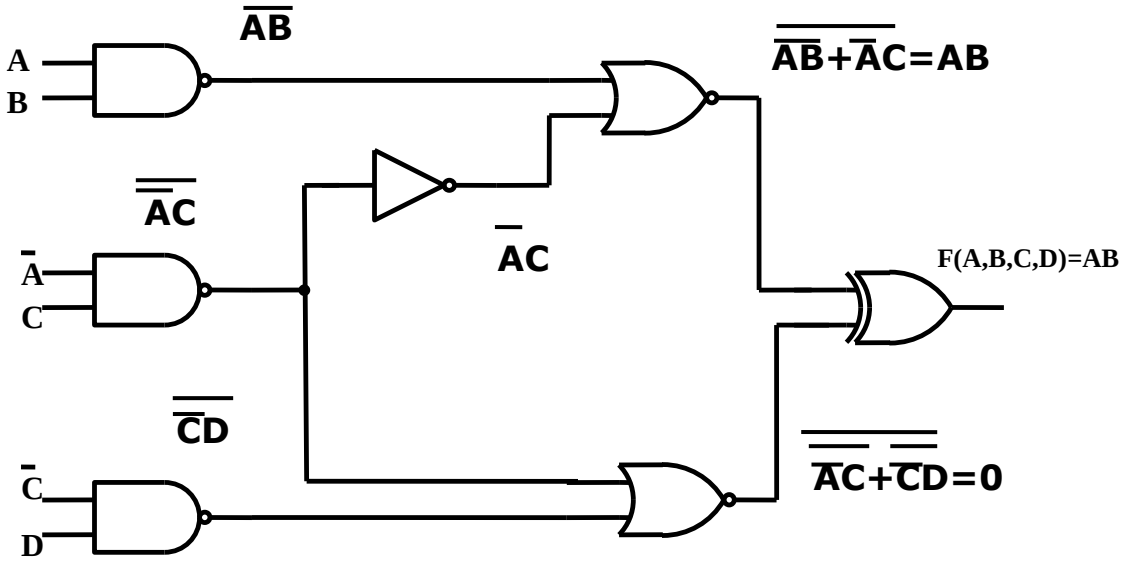
Ο πίνακας Karnaugh της παραπάνω λογικής συνάρτησης φαίνεται παρακάτω στο σχήμα. Η συνάρτηση F(A,B,C) έχει απλοποιηθεί σε τρεις όρους των δύο μεταβλητών και κατά συνέπεια χρειάζονται τρεις πύλες NAND δύο εισόδων και μία τριών για να υλοποιηθεί. Το κύκλωμα φαίνεται στο σχήμα. Αν από τον πίνακα Karnaugh συνδυάσω τα μηδενικά τότε ελαχιστοποιώ τη συμπληρωματική συνάρτηση της F(A,B,C) όπως φαίνεται στο παρακάτω σχήμα. Το δε συμπλήρωμα της μου δίνει την απλοποιημένη συνάρτηση F(A,B,C) σαν γινόμενο αθροισμάτων. Η υλοποίηση σε αυτή τη μορφή γίνεται αποκλειστικά με πύλες NOR. Στην συγκεκριμένη περίπτωση απαιτούνται δύο πύλες δύο εισόδων και μία τριών και κατά συνέπεια η υλοποίηση NOR είναι η πιο οικονομική όπως στο σχήμα.



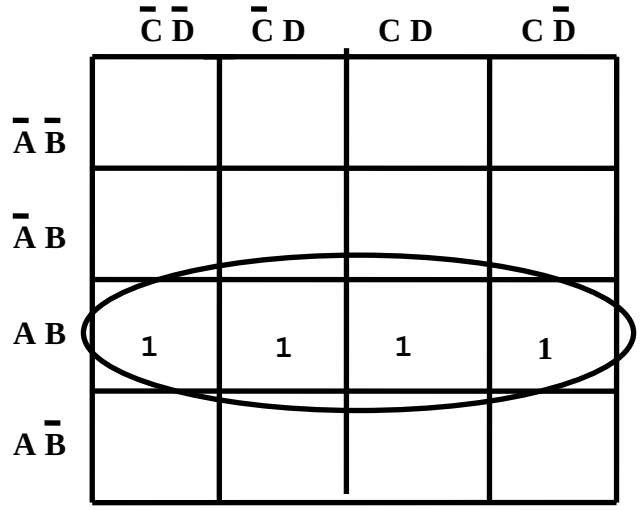


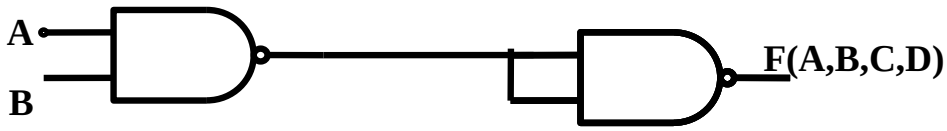


Άσκηση: Δίνεται το παρακάτω λογικό κύκλωμα. Να βρεθεί η συνάρτηση F(A,B,C,D) στην έξοδο, να απλοποιηθεί και να υλοποιηθεί αποκλειστικά με πύλες NAND.

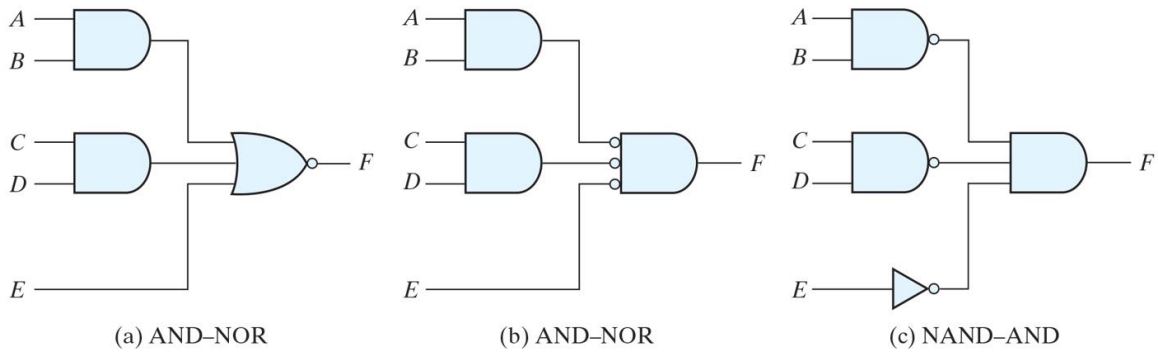


Αν εφαρμόσω τις γνωστές σχέσεις του De Morgan στο παραπάνω κύκλωμα και κάνω τις πράξεις τότε η συνάρτηση στην έξοδο είναι: F(A,B,C,D) =Σ(12,13,14,15). Ο χάρτης αυτής της συνάρτησης φαίνεται παρακάτω. Η συνάρτηση που δίνει το κύκλωμα είναι η F(A,B,C,D)=AB. Βλέπουμε από τον πίνακα Karnaugh της συνάρτησης ότι η ελάχιστη μορφή είναι αυτή που δίνει το κύκλωμα και δεν απλοποιείται παραπάνω. Όλα τα τετράγωνα έχουν συνδυαστεί τουλάχιστον 1 φορά, και ο συνδυασμός είναι ο μέγιστος των 4 τετραγώνων. Η υλοποίηση γίνεται εύκολα με δύο πύλες NAND δύο εισόδων (τσιπ 7400).

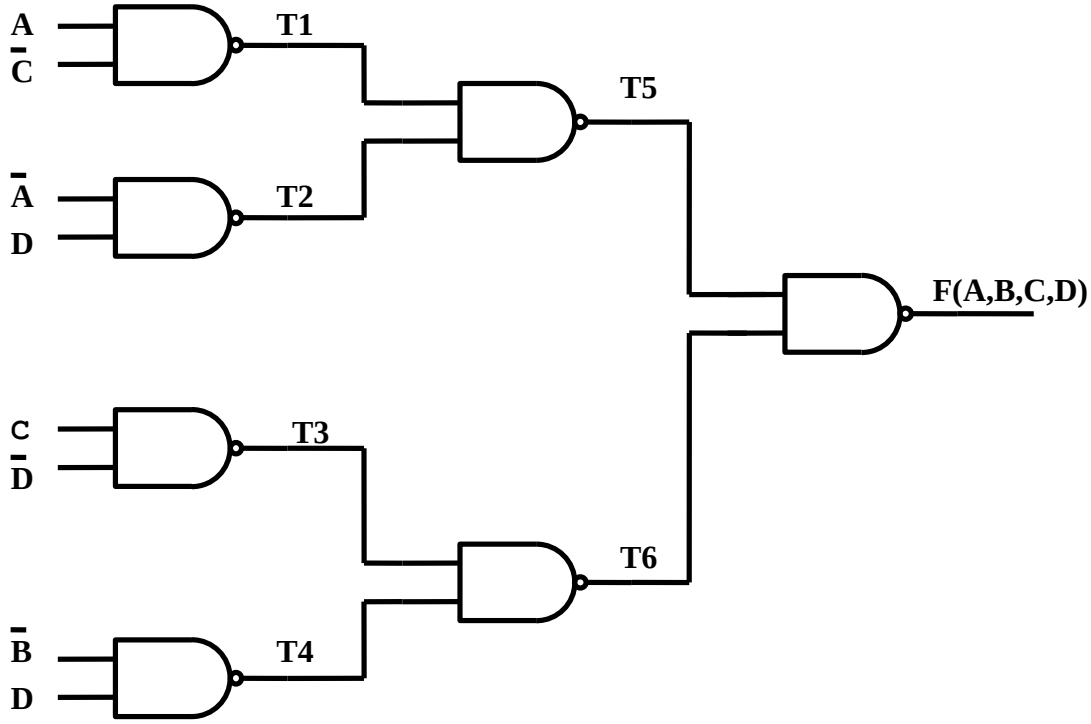




Άσκηση: Να βρεθούν για τα παρακάτω 3 κυκλώματα διεπίπεδης υλοποίησης οι συναρτήσεις που αναπαράγονται στην έξοδο. Ένας μικρός κύκλος πριν την είσοδο μιας πύλης σημαίνει αντιστροφή.



Άσκηση: Δίνεται το παρακάτω λογικό κύκλωμα. Βρείτε τη συνάρτηση F(A,B,C,D) στην έξοδο. Απλοποιήστε τη σαν γινόμενο αθροισμάτων και στην συνέχεια κάνετε μία διεπίπεδη υλοποίηση NOR-NOR.

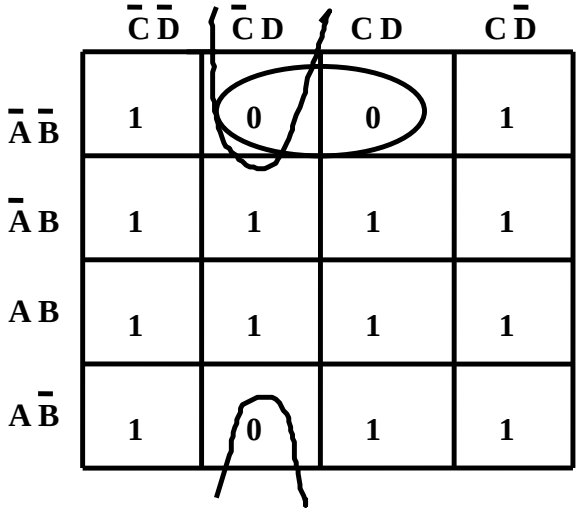


Αν κάνω τις πράξεις στην έξοδο κάθε πύλης θα βρω ότι η συνάρτηση F(A,B,C,D)=Σ(0,2,3,4,5,6,7,8,10,11,12,13,14,15). Όταν έχω να επιλύσω ένα κύκλωμα αρκετά περίπλοκο το καλύτερο που έχω να κάνω είναι να χρησιμοποιήσω βοηθητικές συναρτήσεις στην έξοδο κάθε πύλης σαν ενδιάμεσο εργαλείο. Συνήθως πριν την τελευταία πύλη έχουμε συχνά διαδοχικές φορές συμπληρώματα μίας παράστασης οπότε έχουμε απλοποίηση ήδη πριν ξεκινήσουμε να κάνουμε την ανάλυση της αλγεβρικής παράστασης.

$$T1 = \overline{A\bar{C}}$$
$$T2 = \overline{\bar{A}D}$$
$$T3 = \overline{C\bar{D}}$$
$$T4 = \overline{\bar{B}D}$$

$$\textbf{Οπότε} \quad T_5 = \overline{T_1 \cdot T_2} = \overline{\overline{A\bar{C}} \cdot \overline{\bar{A}D}}$$
$$\textbf{και} \quad T_6 = \overline{T_3 \cdot T_4} = \overline{\overline{C\bar{D}}\,\overline{\bar{B}D}}$$
$$\textbf{Και} \quad F(A,B,C,D) = \overline{T_5 \cdot T_6} = \Sigma(0,2,4,5,6,7,8,10,11,12,13,14,15)$$

Ο πίνακας Karnaugh της παραπάνω συνάρτησης φαίνεται παρακάτω στο σχήμα.

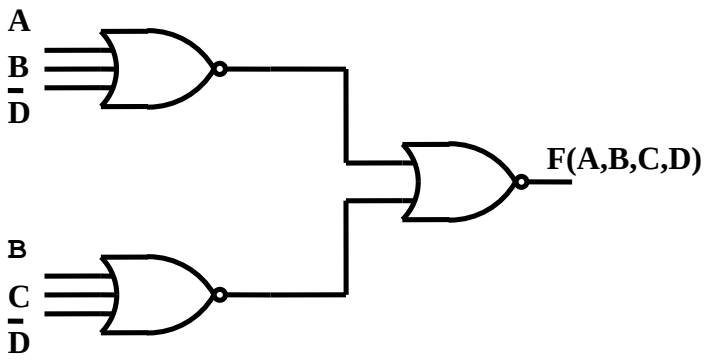


$$\bar{F}(A,B,C,D) = \bar{A}\bar{B}D + \bar{B}\bar{C}D$$

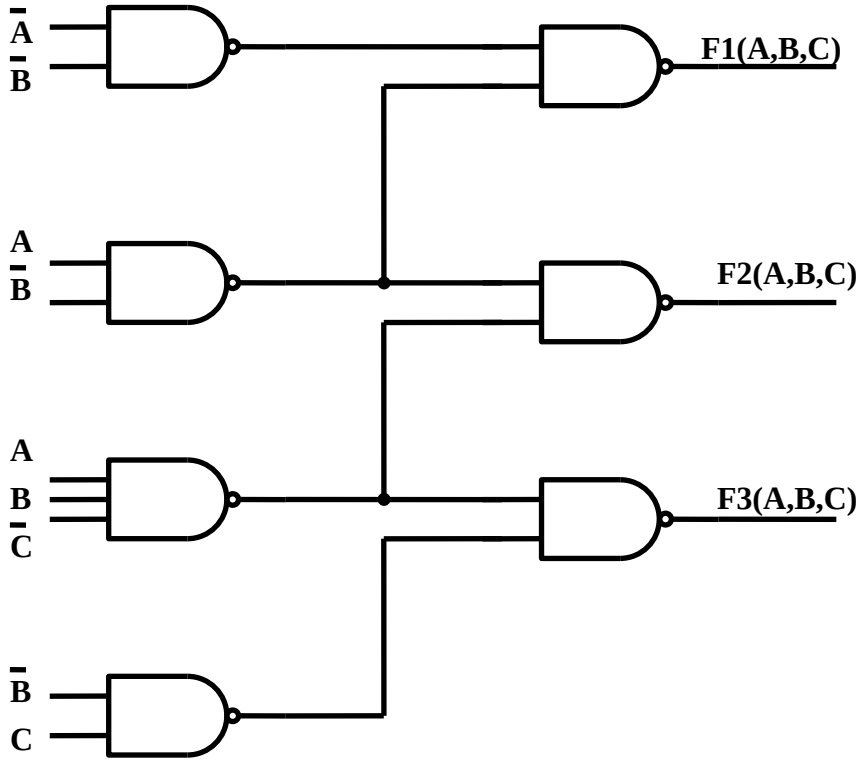
Αυτό που με ενδιαφέρει είναι να υλοποιήσω τη λογική συνάρτηση αποκλειστικά με πύλες NOR γι' αυτό το λόγο απλοποιώ τη συμπληρωματική συνάρτηση. Αν πάρω το συμπλήρωμα της παραπάνω σχέσης θα έχω:

$$\bar{F}(A,B,C,D) = \bar{A}\bar{B}D + \bar{B}\bar{C}D \Leftrightarrow$$
$$\overline{\bar{F}}(A,B,C,D) = \overline{\bar{A}\bar{B}D + \bar{B}\bar{C}D} \Leftrightarrow$$
$$F(A,B,C,D) = (A + B + \bar{D}) \cdot (B + C + \bar{D})$$

Η παραπάνω απλοποιημένη συνάρτηση υλοποιείται με τρεις πύλες NOR όπως φαίνεται στο παρακάτω σχήμα.



Δίνεται το κύκλωμα παραπάνω σχήματος. Υπολογίστε τις κανονικές μορφές των συναρτήσεων στην έξοδο.



Οι όροι που θα πάρω στην έξοδο θα πρέπει να συμπληρωθούν με τις μεταβλητές που λείπουν για να μου δώσουν τους ελαχιστόρους όπως φαίνεται παρακάτω.

$$F1(A,B,C) = \overline{A}\,\overline{B} + A\overline{B} = \overline{A}\,\overline{B}(C + \overline{C}) + A\overline{B}(C + \overline{C}) \Rightarrow$$

$$F1(A,B,C) = \overline{A}\,\overline{B}C + \overline{A}\,\overline{B}\,\overline{C} + A\overline{B}C + A\overline{B}\,\overline{C} \Rightarrow$$

$$F1(A,B,C) = \Sigma(0,1,4,5) = \Pi(2,3,6,7)$$

$$F2(A,B,C) = A\overline{B} + AB\overline{C} = A\overline{B}(C + \overline{C}) + AB\overline{C} \Rightarrow$$

$$F2(A,B,C) = A\overline{B}C + A\overline{B}\,\overline{C} + AB\overline{C} \Rightarrow$$

$$F2(A,B,C) = \Sigma(4,5,6) = \Pi(0,1,2,3,7)$$

$$F3(A,B,C) = AB\overline{C} + \overline{B}C \Rightarrow$$

$$F3(A,B,C) = AB\overline{C} + (A + \overline{A})\overline{B}C = AB\overline{C} + A\overline{B}C + \overline{A}\,\overline{B}C \Rightarrow$$

$$F3(A,B,C) = \Sigma(1,5,6) = \Pi(0,2,3,4,7)$$

Αδιάφοροι όροι

Σε ορισμένες περιπτώσεις η έξοδος ενός ψηφιακού κυκλώματος δεν είναι αναγκαστικά συνάρτηση όλων των δυνατών συνδυασμών των μεταβλητών εισόδου αλλά μέρους αυτών. Οι συνδυασμοί εισόδου που δεν παράγουν κάποια έξοδο ονομάζονται αδιάφοροι όροι. Αυτό σημαίνει ότι ένα ορισμένος αριθμός των συνδυασμών των μεταβλητών εισόδου δεν πρόκειται ποτέ να υπάρξει για πολλούς λόγους που καθορίζονται από την ίδια τη φύση του προβλήματος.

Ας δούμε όμως ένα παράδειγμα. Έστω ότι έχουμε ένα κύκλωμα το οποίο δέχεται στην είσοδο ένα αριθμό στον κώδικα BCD και παράγει στην έξοδο διάφορες συναρτήσεις F1, F2, ... Όλες οι συναρτήσεις θα είναι τεσσάρων μεταβλητών και δε θα ορίζονται για τους συνδυασμούς 1010, 1011, 1100, 1101, 1110 και 1111 λόγω του ότι η είσοδος είναι ένας αριθμός στον κώδικα BCD όπου οι παραπάνω συνδυασμοί δεν υπάρχουν. Οι παραπάνω συνδυασμοί των μεταβλητών εισόδου ονομάζονται αδιάφοροι όροι και είναι επίσης οι ελαχιστόροι m10, m11, m12, m13, m14 και m15. Τους αδιάφορους όρους τους συμβολίζουμε με ένα X στον πίνακα αλήθειας μίας συνάρτησης όπως επίσης στο πίνακα Karnaugh της συνάρτησης. Τους συμβολίζουμε επίσης με d=d(i,j,k,……) όπου i,j,k,…. Είναι οι δείκτες των ελαχιστόρων αντίστοιχα i,j,k,…. Στην περίπτωση του κώδικα BCD θα έχουμε δηλαδή αδιάφορους όρους τους d=d(10,11,12,13,14,15).

Το X σημαίνει αδιάφορος όρος και δηλώνει ότι μπορεί να πάρει την τιμή 1 η 0. Στην ουσία οι συνδυασμοί αυτοί δε συμβαίνουν ποτέ στην είσοδο. Αφού όμως το X μπορεί να πάρει την τιμή 1 η 0 θα μπορέσω να χρησιμοποιώ και τους αδιάφορους όρους στην απλοποίηση μιας λογικής συνάρτησης **μόνο** αν αυτό με εξυπηρετεί, αν δηλαδή οι συνδυασμοί των τετραγώνων που θα κάνω μεγιστοποιούνται αν συμπεριλάβουμε τους αδιάφορους όρους γιατί τότε απλοποιούνται περισσότερες μεταβλητές. Στην αντίθετη περίπτωση οι αδιάφοροι όροι μίας λογικής συνάρτησης δε χρησιμοποιούνται.

Ας δούμε όμως όλα τα παραπάνω σε ένα παράδειγμα. Δίνεται παρακάτω ο πίνακας αλήθειας μίας λογικής συνάρτησης τεσσάρων μεταβλητών. Να

απλοποιηθεί και σαν άθροισμα γινομένων και σαν γινόμενο αθροισμάτων και στην συνέχεια να υλοποιηθεί αποκλειστικά με πύλες NAND και στην συνέχεια με πύλες NOR.

A	B	C	D	F(A,B,C,D)
0	0	0	0	1
0	0	0	1	1
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	X
1	0	0	0	X
1	0	0	1	X
1	0	1	0	X
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	X
1	1	1	1	1

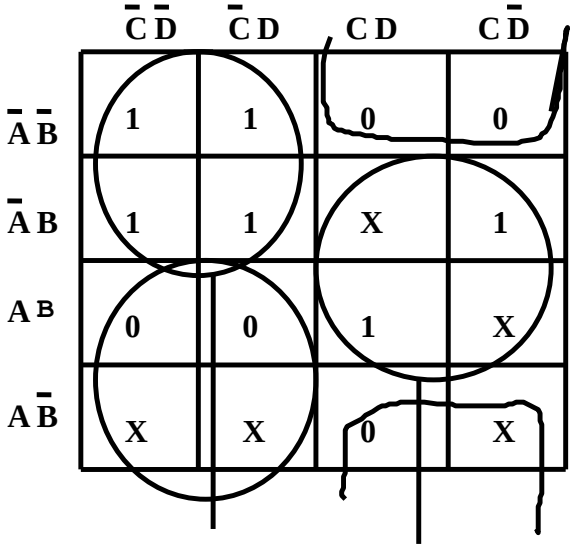
Πίνακας αλήθειας

Παρακάτω φαίνεται ο χάρτης της συνάρτησης. Οι αδιάφοροι όροι φαίνονται μέσα στον πίνακα αλήθειας και είναι οι **d=d(7,8,9,10,14)** Παρατηρώ ότι για να βρω και τις δύο ελάχιστες μορφές κάνω χρήση των αδιάφορων όρων στον παρακάτω πίνακα και παίρνω ότι οι 2 απλοποιημένες εκφράσεις της F σαν άθροισμα γινόμενων και σαν γινόμενο αθροισμάτων είναι οι παρακάτω:

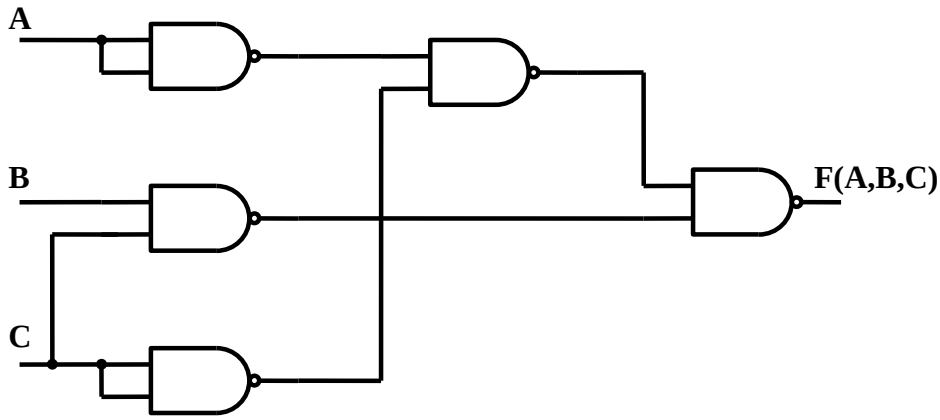
$$\mathbf{F(A,B,C,D) = \overline{A}\,\overline{C} + BC = (\overline{A} + C)\cdot(B + \overline{C})}$$

Στην πρώτη περίπτωση χρειαζόμαστε πέντε πύλες NAND 2 εισόδων για να υλοποιήσουμε το κύκλωμα. Θέλουμε 2 NAND 2 εισόδων για να αντιστρέψουμε τις μεταβλητές A και C, άλλες 2 για να σχηματίσουμε τους

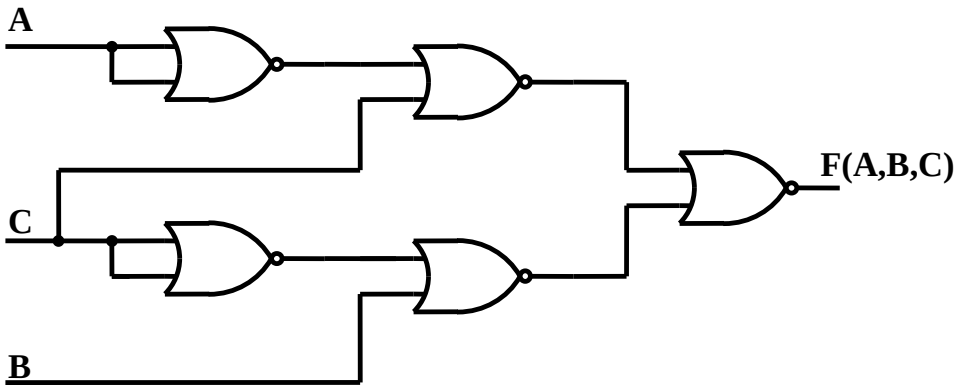
2 όρους και άλλη 1 για να βγει η συνάρτηση. Στην δεύτερη υλοποίηση χρειαζόμαστε επίσης πέντε NOR όπως φαίνεται στα παρακάτω σχήματα. Πρέπει εδώ να τονίσουμε ότι λογικές συναρτήσεις με αδιάφορους όρους συναντούμε πολύ συχνά σε πολλά προβλήματα στην ψηφιακή σχεδίαση και ιδιαίτερα στο δεύτερο μέρος του μαθήματος δηλαδή στα συνδυαστικά κυκλώματα όπως για παράδειγμα στους μετρητές.



$$F(A,B,C,D)= \bar{A}\bar{C} + B\,C$$



Υλοποίηση με NAND

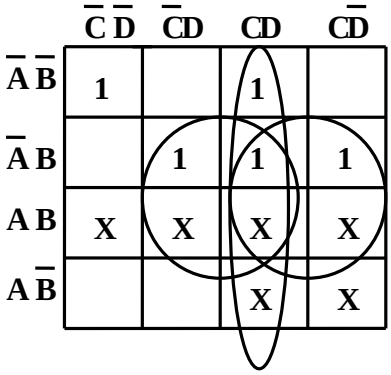


Υλοποίηση με NOR

Δίνεται ο παρακάτω πίνακας αλήθειας. Απλοποιήστε τις τέσσερις συναρτήσεις Y1, Y2, Y3, Y4, και υλοποιήστε το ψηφιακό κύκλωμα αποκλειστικά με πύλες NAND.

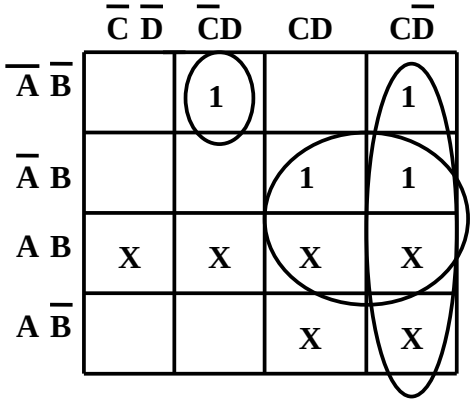
A	B	C	D1	Y1	Y2	Y3	Y4
0	0	0	0	1	0	1	0
0	0	0	1	0	1	0	1
0	0	1	0	0	1	1	1
0	0	1	1	1	0	0	1
0	1	0	0	0	0	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	1	1	0
0	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0
1	0	0	1	0	0	0	1
1	0	1	0	X	X	X	X
1	0	1	1	X	X	X	X
1	1	0	0	X	X	X	X
1	1	0	1	X	X	X	X
1	1	1	0	X	X	X	X
1	1	1	1	X	X	X	X

Παρατηρούμε ότι οι ελαχιστόροι 10, 11, 12, 13, 14, 15 είναι αδιάφοροι όροι d=d(10,11,12,13,14,15) και για τις τέσσερις συναρτήσεις. Οι χάρτες των συναρτήσεων φαίνονται παρακάτω όπως επίσης και οι απλοποιημένες μορφές των 4 συναρτήσεων.



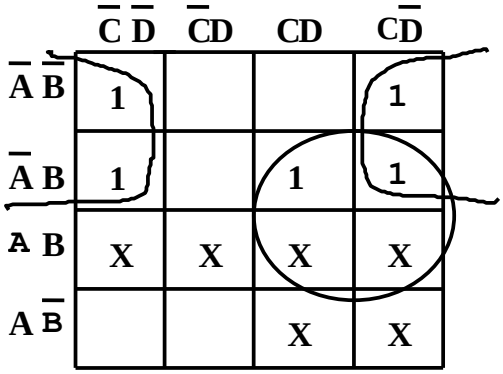
Y1(A,B,C,D)=$\bar{A}\,\bar{B}\,\bar{C}\,\bar{D} + BC + BD + CD$

Παρατηρούμε ότι σε ένα τετράγωνο ένας μοναχικός άσσος, δεν έχει γειτονικό τετράγωνο και δεν συνδυάζεται και παραμένει ένας όρος με τέσσερεις μεταβλητές. Βλέπουμε όμως πώς συνδυάζονται οι αδιάφοροι όροι και μόνο όταν πρόκειται να ελαχιστοποιηθεί καλύτερα η συνάρτηση διαφορετικά δεν τους χρησιμοποιούμε καθόλου. Δεν χρειάζεται να τους συμπεριλάβουμε στους συνδυασμούς μόνο όταν αυτό εξυπηρετεί και στοχεύει στην καλύτερη απλοποίηση. Το ζητούμενο πάντα παραμένει ο συνδυασμός των περισσότερων τετραγώνων, και να συνδυάσουμε όλους τους άσσους τουλάχιστον κατά 1 φορά. Παρατηρούμε ότι έχουμε συνδυάσει 4 τετράγωνα από τρεις φορές που θα μας δώσουν 3 όρους 2 μεταβλητών και ένα όρος ασυνδύαστος των 4 μεταβλητών.



Y2(A,B,C,D)= $\bar{A}\,\bar{B}\,\bar{C}\,D + B\,C + C\,\bar{D}$

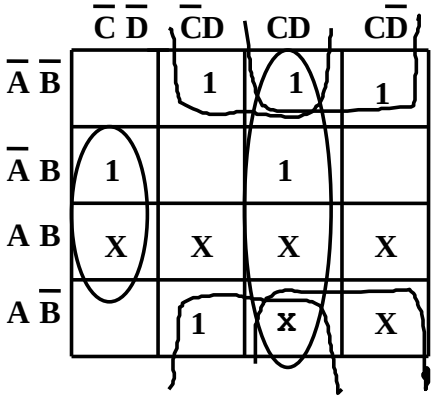
Και εδώ παρατηρούμε επίσης ότι σε ένα τετράγωνο πάλι ένας άσσος που δεν συνδυάζεται και παραμένει ένας όρος με τις τέσσερεις μεταβλητές. Βλέπουμε πώς συνδυάζονται οι αδιάφοροι όροι σε κάθε περίπτωση.



$$Y3(A,B,C,D)= \bar{A}\,\bar{D} + B\,C$$

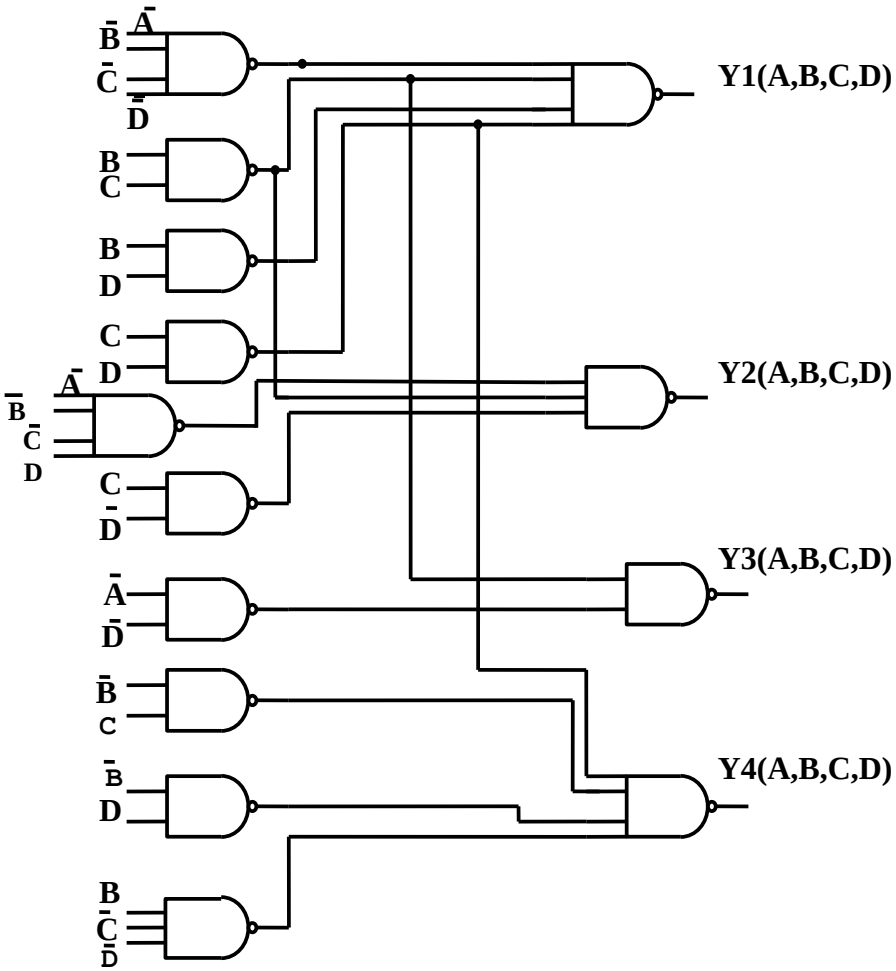
Στην περίπτωση της συνάρτησης Υ3 οι συνδυασμοί είναι απλοί. Το ζητούμενο κάθε φορά είναι να συνδυάσουμε τον μέγιστο αριθμό 1 τουλάχιστον κατά 1 φορά αν αυτό προφανώς είναι εφικτό. Όπως επίσης πρέπει να βρούμε σωστά και τον ελάχιστο αριθμό συνδυασμών, διότι κάθε συνδυασμός μας δίνει και ένα όρο παραπάνω και μεγαλώνει το κύκλωμα. Με τους δύο συνδυασμούς των τεσσάρων τετραγώνων καταφέρνουμε να απλοποιήσουμε την συνάρτηση σε μια απλή μορφή με δύο όρους των δύο μεταβλητών.

Παρακάτω βλέπουμε τον πίνακα αλήθειας της συνάρτησης Υ4.



$$Y4(A,B,C,D)= B\,\bar{C}\,\bar{D} + \bar{B}\,D + C\,D + \bar{B}\,C$$

Στην περίπτωση της συνάρτησης Y4 οι συνδυασμοί είναι πιο μπερδεμένοι και θέλει στις επιλογές των συνδυασμών να τηρούνται πάλι όλοι οι κανόνες απλοποίησης. Για την επιλογή των συνδυασμών υπάρχει κατά κάποιο τρόπο για να γίνει το κύκλωμα ελάχιστο μια μοναδικότητα στην επιλογή των συνδυασμών. Μετά την απλοποίηση η υλοποίηση του κυκλώματος είναι αρκετά περίπλοκη όπως φαίνεται στο παρακάτω σχήμα. Εδώ θεωρούμε δεδομένες τις αντιστροφές στην είσοδο και για την υλοποίηση αποκλειστικά με πύλες NAND χρειαζόμαστε 13 πύλες, αναλυτικότερα 8 πύλες NAND 2 εισόδων, 2 πύλες NAND 3 εισόδων και 3 πύλες NAND 4 εισόδων. Το παρακάτω κύκλωμα είναι σε όλο του το μεγαλείο ένα συνδυαστικό κύκλωμα.



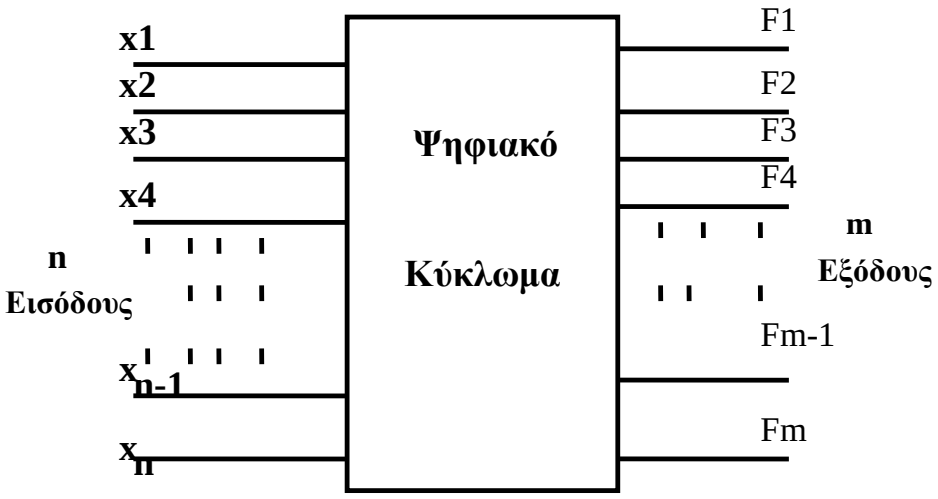
Υλοποίηση συνδυαστικού κυκλώματος με τέσσερις εξόδους αποκλειστικά με πύλες NAND δύο (7400), τριών (74010) και τεσσάρων εισόδων.

Κεφάλαιο 5. Συνδυαστικά και Αριθμητικά Κυκλώματα

Όπως έχουμε ήδη πει τα ψηφιακά κυκλώματα χωρίζονται σε δύο μεγάλες κατηγορίες, που αποτελούν και τις 2 ενότητες του μαθήματος και είναι τα συνδυαστικά κυκλώματα και τα **ακολουθιακά**. Τα συνδυαστικά κυκλώματα με των οποίων τη σχεδίαση θα ασχοληθούμε σε αυτό το κεφάλαιο δε χρησιμοποιούν στοιχεία μνήμης και οι καταστάσεις των εξόδων δεν αλλάζουν (με την προϋπόθεση ότι τροφοδοτούνται συνεχώς) αν δεν αλλάξουν οι είσοδοι. Τα συνδυαστικά κυκλώματα εκτελούν μία συγκεκριμένη διαδικασία η οποία εκφράζει μία λογική συνάρτηση. Αυτό σημαίνει ότι σε ένα συνδυασμό των μεταβλητών εισόδου αντιστοιχεί μία και μόνο έξοδος η συνδυασμός εξόδων. Η έξοδος μπορεί να αλλάζει σε μία μεταβολή των μεταβλητών εισόδου μετά από μία καθυστέρηση η οποία χαρακτηρίζει την κάθε οικογένεια ολοκληρωμένων κυκλωμάτων. Για τη σχεδίαση τους είναι ανάγκη να ακολουθήσουμε μία κάποια διαδικασία η οποία περιγράφεται συνοπτικά παρακάτω.

1. Καθορίζουμε το πρόβλημα. Προσπαθούμε να καταλάβουμε ακριβώς την κατασκευή που μας έχουν ζητήσει να κάνουμε σε όλα της τα μέρη.

2. Καθορίζουμε τον αριθμό εισόδων και εξόδων του κυκλώματος.



3. Θεωρούμε κάθε μεταβλητή εξόδου σαν μία ανεξάρτητη συνάρτηση Boole των μεταβλητών εισόδου. Αυτό σημαίνει ότι θα έχουμε να

απλοποιήσουμε ένα μεγάλο αριθμό λογικών συναρτήσεων ίσων με τον αριθμό εξόδων του κυκλώματος. Θα κάνουμε απλοποιήσεις με τη βοήθεια των πινάκων Karnaugh για κάθε έξοδο και την ελάχιστη μορφή θα την καθορίζουν τα δεδομένα.

4. Υλοποίηση του συνδυαστικού κυκλώματος με τις πύλες η τα ολοκληρωμένα κυκλώματα MSI που θα δούμε παρακάτω και ανάλογα με την ευκολία κατασκευής.

Υλοποίηση του ημιαθροιστή και του πλήρη αθροιστή

Λέμε ημιαθροιστή ένα ψηφιακό κύκλωμα το οποίο είναι ικανό να προσθέσει δύο BIT. Έχουμε ήδη δει ότι οι κανόνες της πρόσθεσης στο δυαδικό σύστημα αρίθμησης είναι ίδιοι με αυτούς που ξέρουμε και χρησιμοποιούμε στο δεκαδικό. Εφαρμόζοντας τους κανόνες σχεδίασης που αναφέραμε πριν, πρέπει αφού έχουμε προσδιορίσει το πρόβλημα να βρούμε τον αριθμό εισόδων και εξόδων.

Έχουμε δύο εισόδους, αυτό σημαίνει ότι για να προσδιορίσω τις εξόδους θα πρέπει να ξέρω ποιος θα είναι ο μέγιστος αριθμός που μπορεί να έχω στην έξοδο του ημιαθροιστή όταν προφανώς προσθέσω τα 2 BIT.

Ο μέγιστος αριθμός θα είναι το δυαδικό δύο 10 και θα το έχω όταν τα σήματα στην είσοδο θα είναι 1 και 1. Δηλαδή θα έχω στο συνδυαστικό μου κύκλωμα δύο εξόδους τις οποίες ονομάζω αντίστοιχα τη μία άθροισμα S και την άλλη κρατούμενο C. Με τα παραπάνω δεδομένα ο πίνακας αλήθειας του συνδυαστικού μας κυκλώματος θα είναι αυτός του παρακάτω σχήματος αν x και y είναι οι είσοδοι και S, C οι έξοδοι με C το MSB.

x	y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Πίνακας αλήθειας

Ο πίνακας αλήθειας περικλείει όλες τις περιπτώσεις που δίνονται από την παρακάτω εξίσωση x+y=CS όταν τα x και τα y πάρουν όλους τους συνδυασμούς με 0 και 1. Το αποτέλεσμα είναι ένας δίμπιτος αριθμός CS.

έχουμε τις περιπτώσεις

x=0 και y=0 τότε x+y=CS=00

x=0 και y=1 τότε x+y=CS=01

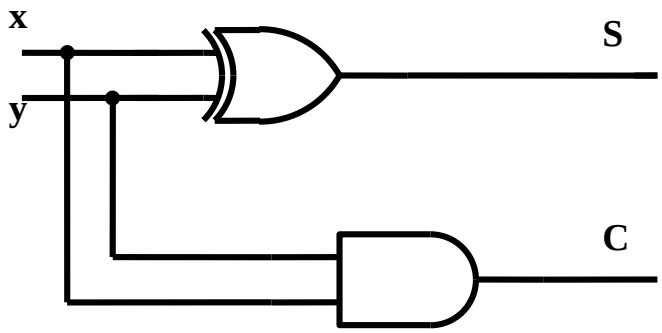
x=1 και y=0 τότε x+y=CS=01

x=1 και y=1 τότε x+y=CS=10

Από τον πίνακα αλήθειας είναι πολύ εύκολο να ορίσω τις συναρτήσεις C και S.

$$S(X,Y) = \overline{X}Y + X\overline{Y}$$
$$C(X,Y) = XY$$

Και η υλοποίηση του κυκλώματος του ημιαθροιστή φαίνεται παρακάτω με μία πύλη XOR και μία AND.



An	An-1	...	A4	A3	A2	A1	A0

Bn	Bn-1	...	B4	B3	B2	B1	B0

Αν τώρα μου δώσουν να προσθέσω δύο αριθμούς των n bits A και B όπως φαίνεται παραπάνω τότε ξεκινώντας από το τελευταίο ψηφίο και προσθέτοντας στο A0 το B0.Η παραπάνω πρόσθεση μου δίνει ένα άθροισμα και ένα κρατούμενο και συνεχίζω την πρόσθεση του A1 με το B1 συν το

κρατούμενο. Αν το κρατούμενο είναι διαφορετικό του μηδενός για να μπορέσω να συνεχίσω την πρόσθεση θα πρέπει το συνδυαστικό μου κύκλωμα να μπορεί να προσθέτει τρία bit.

Αυτό σημαίνει ότι το κύκλωμα του ημιαθροιστή δεν αρκεί και πρέπει να σχεδιάσω το κύκλωμα του πλήρη αθροιστή όπως ονομάζεται, δηλαδή ενός κυκλώματος ικανού να προσθέσει τρία bit. Και σε αυτή την περίπτωση ο αριθμός εξόδων θα είναι πάλι δύο γιατί η ακραία περίπτωση είναι να έχω τρεις άσσους στην είσοδο.

Έχω όμως τότε ότι 1+1+1=11 στο δυαδικό. Κατά συνέπεια ο πίνακας αλήθειας του πλήρη αθροιστή μπορεί εύκολα να βρεθεί αρκεί να προσθέσω τρία Bit A+B+C με όλους τους δυνατούς συνδυασμούς όπως φαίνεται παρακάτω:

A	**B**	**C**	**C**	**S**
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Πίνακας αλήθειας του κυκλώματος πλήρη αθροιστή

Από τον παραπάνω πίνακα αλήθειας παίρνουμε τις συναρτήσεις S(A,B,C)=Σ(1,2,4,7) και C(A,B,C)=Σ(3,5,6,7). Η συνάρτηση S(A,B,C) δεν απλοποιείται όπως φαίνεται στον πίνακα Karnaugh παρακάτω στο σχήμα ενώ η C(A,B,C) απλοποιείται.

$$S(A,B,C) = \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\overline{C} + ABC$$

$$C(A,B,C) = \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC$$

A \ BC	$\bar{B}\bar{C}$	$\bar{B}C$	BC	$B\bar{C}$
$\bar{A}$	0	1	0	1
A	1	0	1	0

Βλέπουμε από τον πίνακα Karnaugh της συνάρτησης S ότι δεν έχουμε κανένα γειτονικό τετράγωνο κατά συνέπεια δεν υπάρχουν συνδυασμοί τετραγώνων γειτονικών και η συνάρτηση δεν απλοποιείται. Ας δούμε όμως αν μπορούμε να την παρουσιάσουμε κάπως διαφορετικά και να υλοποιήσουμε με λιγότερες πύλες.

$$S(A,B,C) = \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + ABC \Rightarrow$$

$$S(A,B,C) = (\bar{A}B + A\bar{B})\bar{C} + (\bar{A}\bar{B} + AB)C \Rightarrow$$

$$S(A,B,C) = (A \oplus B)\bar{C} + (\overline{A \oplus B})C \Rightarrow$$

$$S(A,B,C) = A \oplus B \oplus C$$

Ο πίνακας Karnaugh της συνάρτησης κρατούμενο C(A,B,C) φαίνεται παρακάτω.

A \ BC	$\bar{B}\bar{C}$	$\bar{B}C$	BC	$B\bar{C}$
$\bar{A}$	0	0	1	0
A	0	1	1	1

Βλέπουμε ότι μπορούμε να έχουμε 3 διαφορετικούς συνδυασμούς των δύο τετραγώνων. Με την απλοποίηση παίρνουμε ότι C(A,B,C)=AB+BC+AC. Η αρχική συνάρτηση κρατούμενο από 4 όρους των τριών μεταβλητών που είχε στην κανονικής της μορφή απλοποιήθηκε σε τρεις όρους των δύο μεταβλητών. Αν θέλουμε να χρησιμοποιήσουμε κυκλώματα ημιαθροιστών

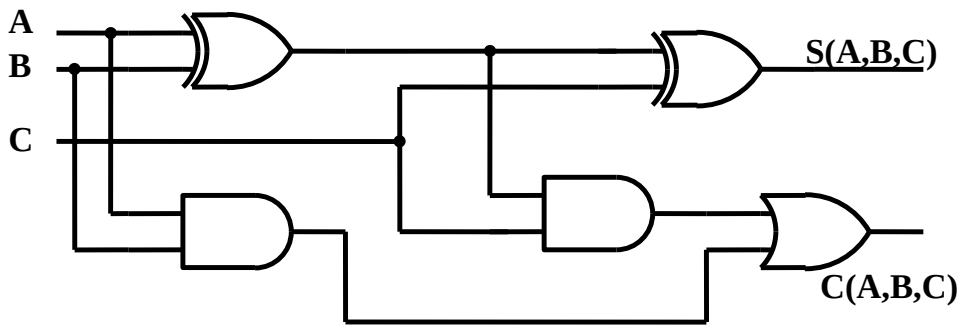
για την υλοποίηση του πλήρη αθροιστή τότε χρειαζόμαστε τη συνάρτηση κρατούμενο C στην παρακάτω μορφή. C(A,B,C)=(A⊕B)·C+AB .

Η απόδειξη της παραπάνω σχέσης όπως επίσης και η υλοποίηση του κυκλώματος φαίνεται παρακάτω.

$$C(A,B,C) = \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC \Rightarrow$$

$$C(A,B,C) = (\overline{A}B + A\overline{B})C + AB(\overline{C} + C) \Rightarrow$$

$$C(A,B,C) = (A \oplus B)C + AB$$



Υλοποίηση του πλήρη αθροιστή με δύο κυκλώματα ημιαθροιστών και μία πύλη OR.

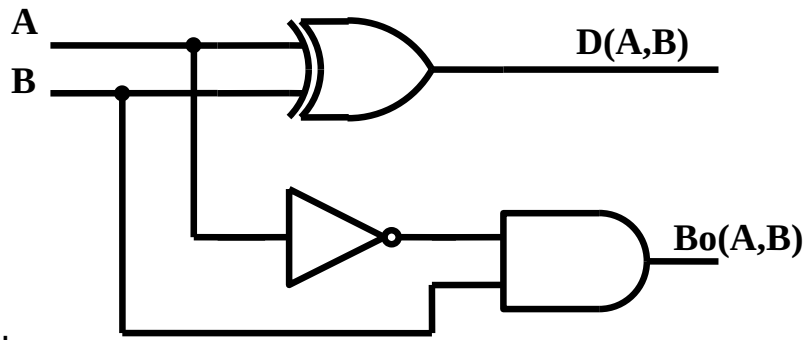
ΗΜΙΑΦΑΙΡΕΤΗΣ και ΑΦΑΙΡΕΤΗΣ

Για τον ίδιο λόγο και στην περίπτωση της αφαίρεσης έχουμε ένα ημιαφαιρέτη και ένα αφαιρέτη. Η πράξη της αφαίρεσης είναι ίδια με αυτή που έχουμε συνηθίσει από το δεκαδικό σύστημα. Ο αριθμός εξόδων του ψηφιακού κυκλώματος του ημιαφαιρέτη θα είναι δύο. Και βγαίνει από την αφαίρεση 0-1 μας δίνει μία διαφορά=1 και ένα κρατούμενο=1. Παρακάτω στο σχήμα φαίνεται ο πίνακας αλήθειας του ημιαφαιρέτη. Bo είναι το κρατούμενο και D η διαφορά.

A	B	Bo(A,B)	D(A,B)
0	0	0	0
0	1	1	1
1	0	0	1

1	1	0	0

Βλέπουμε ότι η συνάρτηση D(A,B,) και η S(A,B) είναι ίδιες και ότι το Bo είναι ίδιο με το C μόνο που το A είναι με το συμπλήρωμα του. Δηλαδή $D(A,B) = A \oplus B$ και $B_0 = \overline{A}B$ Παρακάτω στο σχήμα φαίνεται το κύκλωμα του ημιαφαιρέτη.



.

Όταν υπάρχει κρατούμενο στην αφαίρεση το κύκλωμα του ημιαφαιρέτη δεν μπορεί να συνεχίσει την αφαίρεση γιατί πρέπει να κάνει A-(B+C). Έτσι πρέπει όπως και προηγουμένως στην περίπτωση του πλήρη αθροιστή να σχεδιάσουμε το κύκλωμα του πλήρη αφαιρέτη. Όσον αφορά τον πίνακα αλήθειας των συναρτήσεων Bo και C έχουμε.

A	B	C	B_o(A,B,C)	D(A,B,C)
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

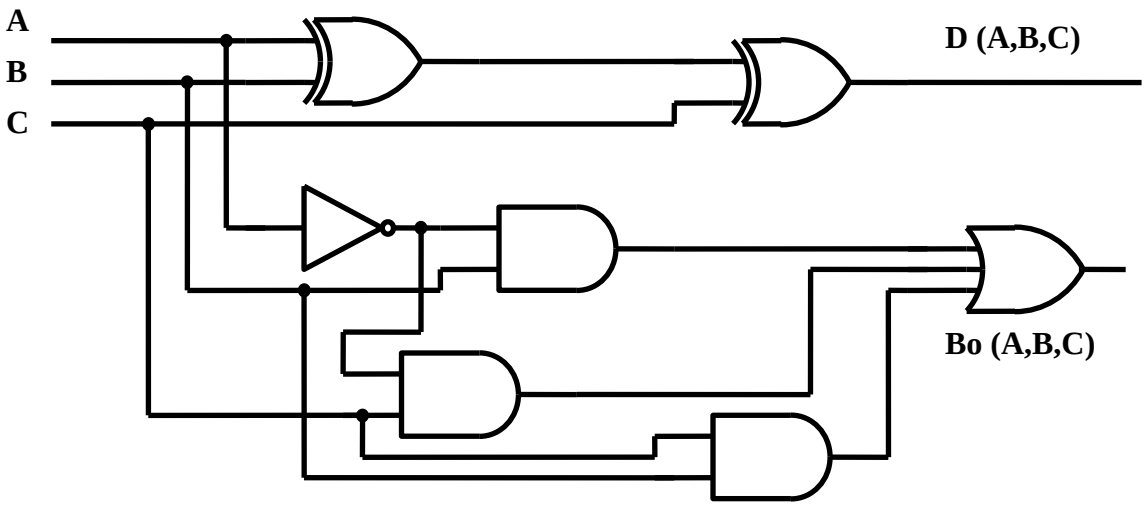
Βλέπουμε ότι η συνάρτηση $D(A,B,C) = \Sigma(1,2,4,7)$ είναι ίδια με τη συνάρτηση S(A,B,C) του πλήρη αθροιστή γιατί έχουν τον ίδιο πίνακα αλήθειας και κατά συνέπεια $D(A,B,C) = A \oplus B \oplus C$.

Η συνάρτηση κρατούμενο $B_0(A,B,C) = \Sigma(1,2,3,7)$ είναι όπως αυτή του πλήρη αθροιστή με τη διαφορά ότι η μεταβλητή Α είναι συμπληρωμένη όπως φαίνεται μετά την απλοποίηση στον πίνακα Karnaugh του παρακάτω σχήματος.

B C / A	$\bar{B}\bar{C}$	$\bar{B}C$	BC	$B\bar{C}$
$\bar{A}$	0	1	1	1
A	0	0	1	0

Απλοποίηση της συναρτήσεως κρατούμενο του πλήρη αφαιρέτη μας δίνει ότι: $$B_0(A,B,C) = \bar{A}B + \bar{A}C + BC$$

Η υλοποίηση του κυκλώματος του πλήρη αφαιρέτη με λογικές πύλες φαίνεται παρακάτω στο σχήμα.



Πλήρης αφαιρέτης.

ΚΥΚΛΩΜΑ ΠΟΛΛΑΠΛΑΣΙΑΣΜΟΥ

Έστω ότι μου δίνουν δύο δίμπιτους αριθμούς A1A0 και B1B0 και μου ζητούν να σχεδιάσω ένα συνδυαστικό κύκλωμα ικανό να τους πολλαπλασιάζει. Αν εκτελέσουμε την διαδικασία σχεδίασης που είδαμε προηγουμένως, κατανοούμε ότι θέλουμε ένα κύκλωμα να εκτελεί πολλαπλασιασμό στο δυαδικό.

Θα αναζητήσουμε τον αριθμό εισόδων και εξόδων του κυκλώματος. Ο αριθμός εισόδων είναι προφανές ότι θα είναι τέσσερεις A1A0B1B0. Ο αριθμός εξόδων καθορίζεται από τους μέγιστους αριθμούς που μπορούν να εμφανιστούν στην είσοδο και αυτό καθορίζει τον αριθμό εξόδων του ψηφιακού κυκλώματος.

Ο αριθμός στην έξοδο θα είναι μέγιστος όταν οι είσοδοι είναι μέγιστοι, δηλαδή 11 και 11 αντίστοιχα.

Τότε 11 x 11= 1001 (3x3=9). Το 9 στο δυαδικό είναι 1001 οπότε για να σχεδιάσω το κύκλωμα χρειάζομαι τέσσερις εξόδους F1F2F3F4. Με τα παραπάνω μπορούμε να προχωρήσουμε στη συμπλήρωση του πίνακα αλήθειας του κυκλώματος πολλαπλασιαστή δίμπιτων δυαδικών αριθμών όπως φαίνεται παρακάτω. Η πλήρωση του πίνακα αλήθειας γίνεται εκτελώντας τους 16 διαφορετικούς συνδυασμούς πολλαπλασιασμού $A1A0xB1B0$ και γράφοντας το αποτέλεσμα στο δυαδικό.

A_1	A_0	B_1	B_0	F_1	F_2	F_3	F_4
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	1	0	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	0	0	0	0	0
0	1	0	1	0	0	0	1
0	1	1	0	0	0	1	0

0	1	1	1	0	0	1	1
1	0	0	0	0	0	0	0
1	0	0	1	0	0	1	0
1	0	1	0	0	1	0	0
1	0	1	1	0	1	1	0
1	1	0	0	0	0	0	0
1	1	0	1	0	0	1	1
1	1	1	0	0	1	1	0
1	1	1	1	1	0	0	1

Πίνακας αλήθειας του δίμπιτου πολλαπλασιαστή

Θεωρώντας κάθε έξοδο σαν μία ανεξάρτητη συνάρτηση Boole των μεταβλητών εισόδου θα έχω τέσσερις συναρτήσεις Boole στην έξοδο. Παρακάτω φαίνονται οι κανονικές μορφές των τεσσάρων συναρτήσεων εξόδου. **Προσοχή!** Στις παρακάτω εκφράσεις των τεσσάρων συναρτήσεων όπως και στους πίνακες Karnaugh έχω αντικαταστήσει το A με το A1, το B με το A0, το C με το B1 και το D με το B0, για τεχνικούς λόγους συγγραφής των σχέσεων.

$F_1(A,B,C,D)=\Sigma(15)=\Pi(0,1,2,3,4,5,6,7,8,9,10,11,12,13,14)$

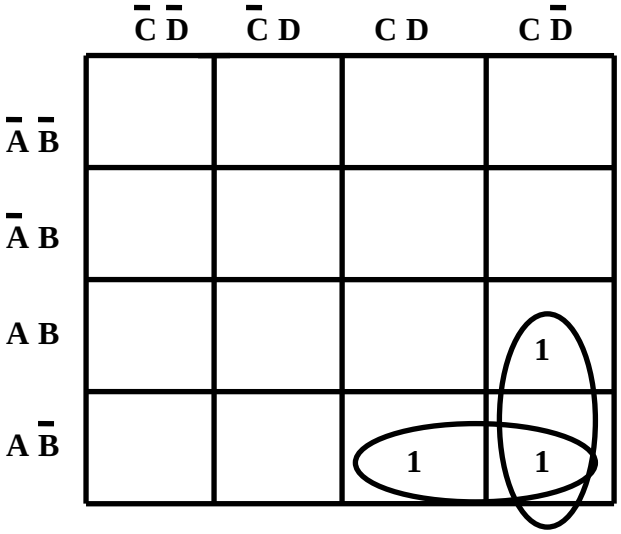
$F_2(A,B,C,D)=\Sigma(10,11,14)=\Pi(0,1,2,3,4,5,6,7,8,9,12,13,15)$

$F_3(A,B,C,D)=\Sigma(6,7,9,11,13,14)=\Pi(0,1,2,3,4,5,8,10,12,15)$

$F_4(A,B,C,D)=\Sigma(5,7,13,15)=\Pi(0,1,2,3,4,6,8,9,10,11,12,14)$

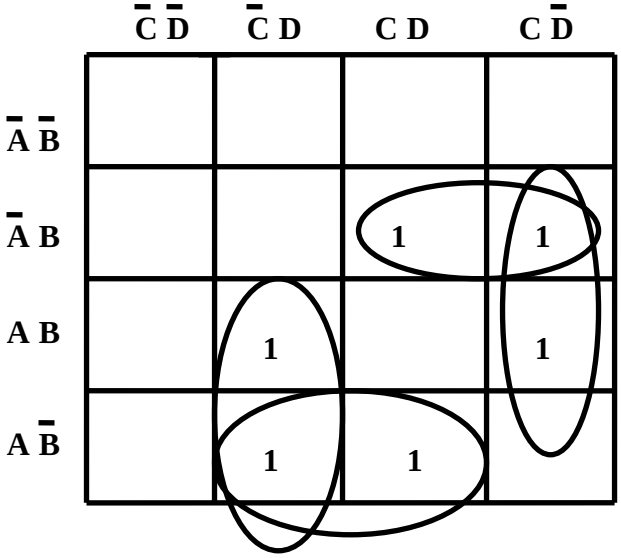
Βλέπουμε ότι η συνάρτηση $F_1(A,B,C,D)=A_1A_0B_1B_0$ έχει μόνο 1 όρο και κατά συνέπεια δεν απλοποιείται.

Η F_2 απλοποιείται πολύ εύκολα με τον παρακάτω πίνακα Karnaugh.



$$F2(A,B,C,D) = AC\bar{D} + A\bar{B}C$$

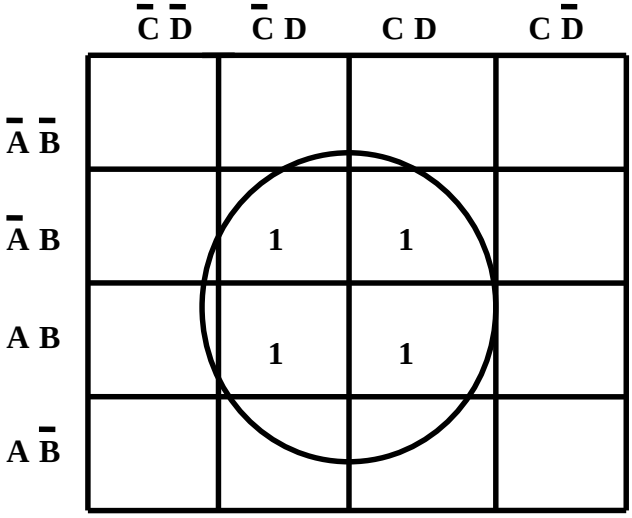
Για την F3 χρειάζομαι τον πίνακα Karnaugh της συνάρτησης για την απλοποίηση. Οι έξη ελαχιστόροι συμπληρώνουν τον πίνακα όπως φαίνεται στο παρακάτω σχήμα.



Απλοποίηση της F3

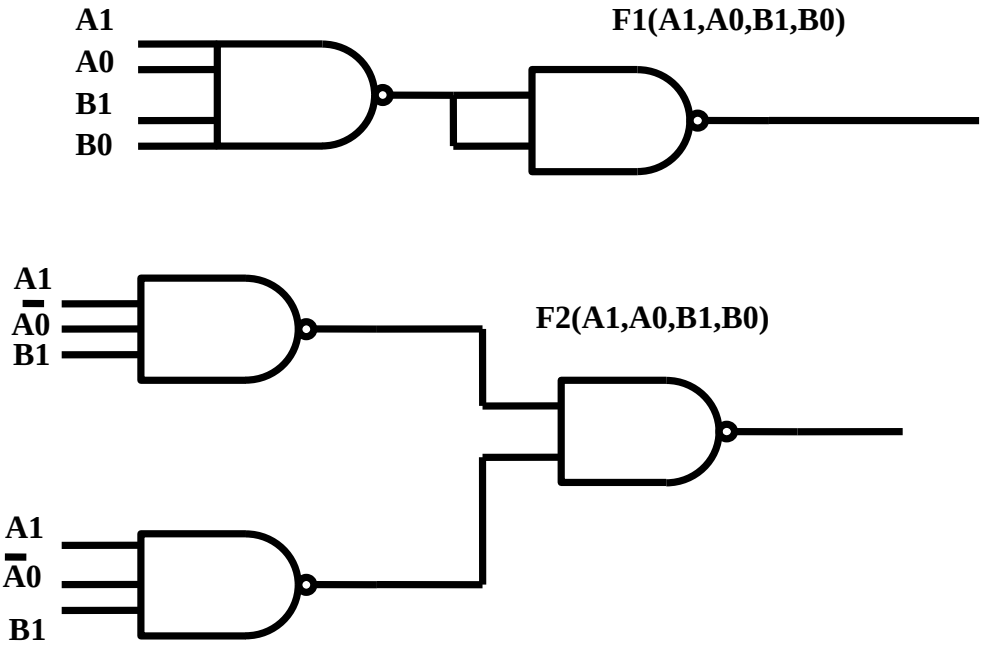
Μετά την απλοποίηση της F3(A,B,C,D) έχω τέσσερεις όρους των τριών μεταβλητών ο καθένας. $F3(A,B,C,D) = \bar{A}BC + BC\bar{D} + A\bar{C}D + A\bar{B}D$ Για την συνάρτηση F4(A,B,C,D) έχω τον πίνακα Karnaugh που φαίνεται στο παρακάτω σχήμα. Όλα τα τετράγωνα είναι γειτονικά, συνδυάζονται κατά

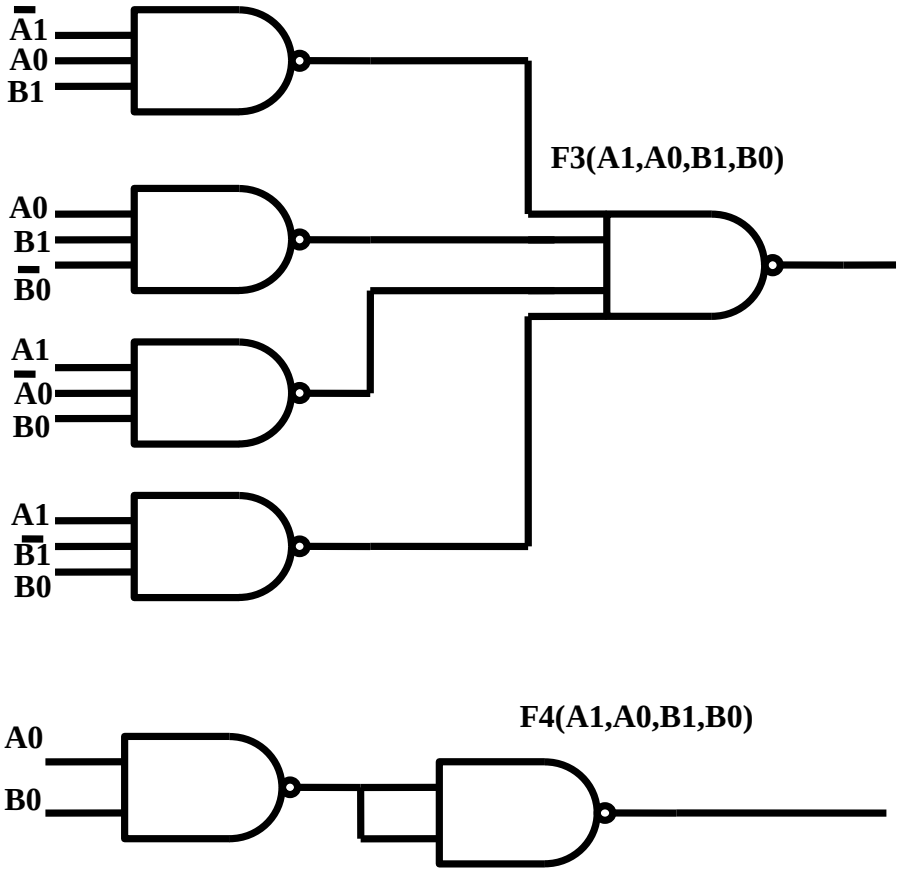
συνέπεια και τα τέσσερα, απλοποιούνται δύο μεταβλητές και μένει ένας μοναδικός όρος γινόμενο μόνο 2 μεταβλητών.



Απλοποίηση της F4

οπότε θα έχω ότι $F4(A,B,C,D) = BD$ Η υλοποίηση του κυκλώματος αποκλειστικά με πύλες NAND του δίμπιτου πολλαπλασιαστή φαίνεται στο παρακάτω σχήμα.





Υλοποίηση του δίμπιτου πολλαπλασιαστή αποκλειστικά με πύλες NAND.

Μετατροπή Κωδίκων

Υπάρχει μία μεγάλη ποικιλία κωδίκων οι οποίοι χρησιμοποιούνται από τα ψηφιακά συστήματα. Πολλές φορές η επίλυση ενός προβλήματος μας οδηγεί στην δημιουργία ενός κώδικα μοναδικού για αυτή τη χρήση. Η δυνατότητα επικοινωνίας μεταξύ των ψηφιακών συστημάτων οδηγεί στην ανάγκη κατασκευής κυκλωμάτων μετατροπείς κωδίκων **interfaces η code converter**.

Με αυτό τον τρόπο δύο συστήματα τα οποία χρησιμοποιούν διαφορετικούς κώδικες γίνονται συμβατά μεταξύ τους.

Έστω ότι έχουμε να κατασκευάσουμε ένα ψηφιακό κύκλωμα το οποίο θα επιτρέπει τη μετατροπή του κώδικα **BCD** σε **Excess-3**. Όπως είναι γνωστό και οι δύο κώδικες είναι τετράμπιτοι.

Άρα το συνδυαστικό κύκλωμα μετατροπέας από **BCD** σε **Excess-3** θα έχει τέσσερις εισόδους και τέσσερις εξόδους. Ο πίνακας αλήθειας του κυκλώματος φαίνεται παρακάτω στον πίνακα.

A	B	C	D	F1	F2	F3	F4
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0

Πίνακας

Οι συναρτήσεις στην έξοδο που είναι ο κώδικας Excess-3 από τον πίνακα αλήθειας θα είναι οι παρακάτω:

$$F1(A,B,C,D) = \bar{A}B\bar{C}D + \bar{A}BC\bar{D} + \bar{A}BCD + A\bar{B}\bar{C}\bar{D} + A\bar{B}\bar{C}D$$
$$F2(A,B,C,D) = \bar{A}\bar{B}\bar{C}D + \bar{A}\bar{B}C\bar{D} + \bar{A}\bar{B}CD + \bar{A}B\bar{C}\bar{D} + A\bar{B}\bar{C}D$$
$$F3(A,B,C,D) = \bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}CD + \bar{A}B\bar{C}\bar{D} + \bar{A}BCD + A\bar{B}\bar{C}\bar{D}$$
$$F4(A,B,C,D) = \bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}C\bar{D} + \bar{A}B\bar{C}\bar{D} + \bar{A}BC\bar{D} + A\bar{B}\bar{C}\bar{D}$$

Και οι κανονικές μορφές θα είναι:

F1(A,B,C,D)=Σ(5,6,7,8,9)=Π(0,1,2,3,4,10,11,12,13,14,15)

F2(A,B,C,D)=Σ(1,2,3,4,9)=Π(0,5,6,7,8,10,11,12,13,14,15)

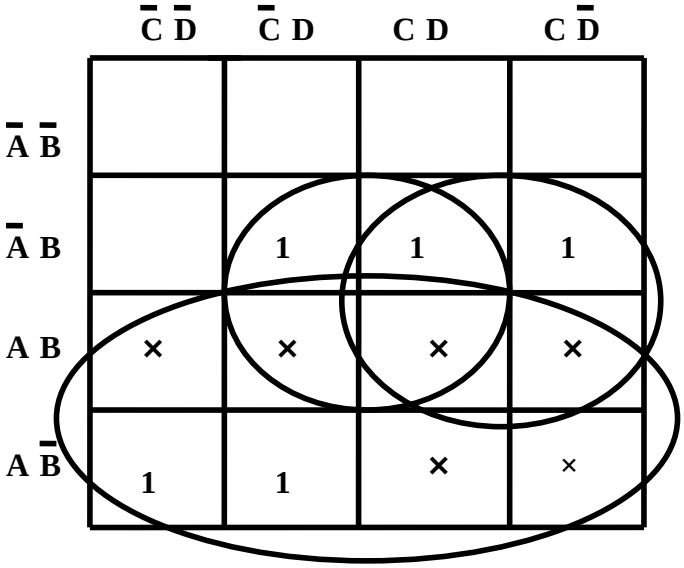
F3(A,B,C,D)=Σ(0,3,4,7,8)=Π(1,2,5,6,9,10,11,12,13,14,15)

F4(A,B,C,D)=Σ(0,2,4,6,8)=Π(1,3,5,7,9,10,11,12,13,14,15)

Όπως είναι γνωστό ο κώδικας BCD κωδικοποιεί τα δέκα δεκαδικά ψηφία του δεκαδικού συστήματος αρίθμησης.

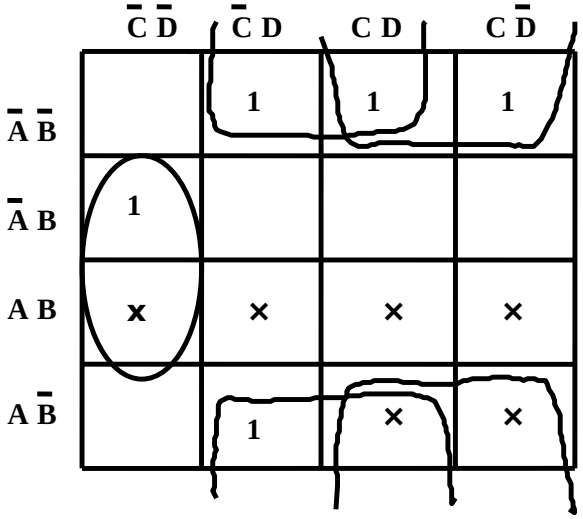
Κατά συνέπεια θα έχουμε αδιάφορους όρους αφού πρόκειται για συναρτήσεις τεσσάρων μεταβλητών. Πρέπει να τους κατατάξουμε σωστά μέσα στους πίνακες και να τους χρησιμοποιήσουμε για την απλοποίηση.

Οι πίνακες Karnaugh των παραπάνω συναρτήσεων F1, F2, F3, F4 είναι αν λάβουμε υπ' όψιν του έξη αδιάφορους όρους οι παρακάτω:



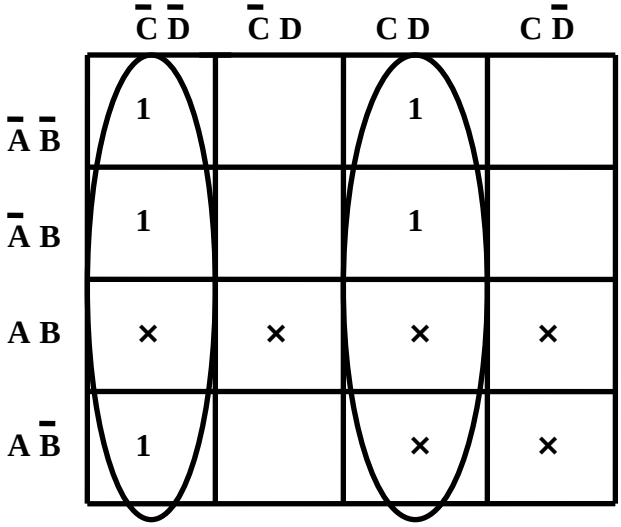
F1(A,B,C,D)=A+BD+BC

Πίνακας Karnaugh και ελάχιστη μορφή της F1(A,B,C,D)



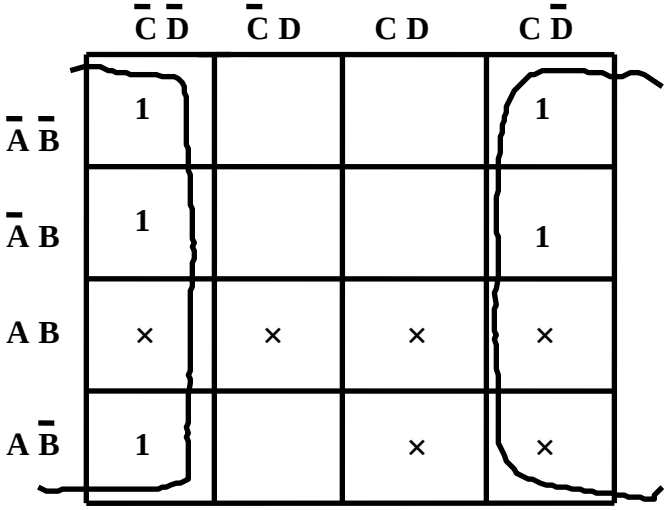
F(A,B,C,D)= $\bar{B}$ D + $\bar{B}$ C + B$\bar{C}\bar{D}$

Πίνακας Karnaugh και ελάχιστη μορφή της F2(A,B,C,D).



$$F3(A,B,C,D) = \bar{C}\,\bar{D} + C\,D$$

Πίνακας Karnaugh και ελάχιστη μορφή της F3(A,B,C,D)

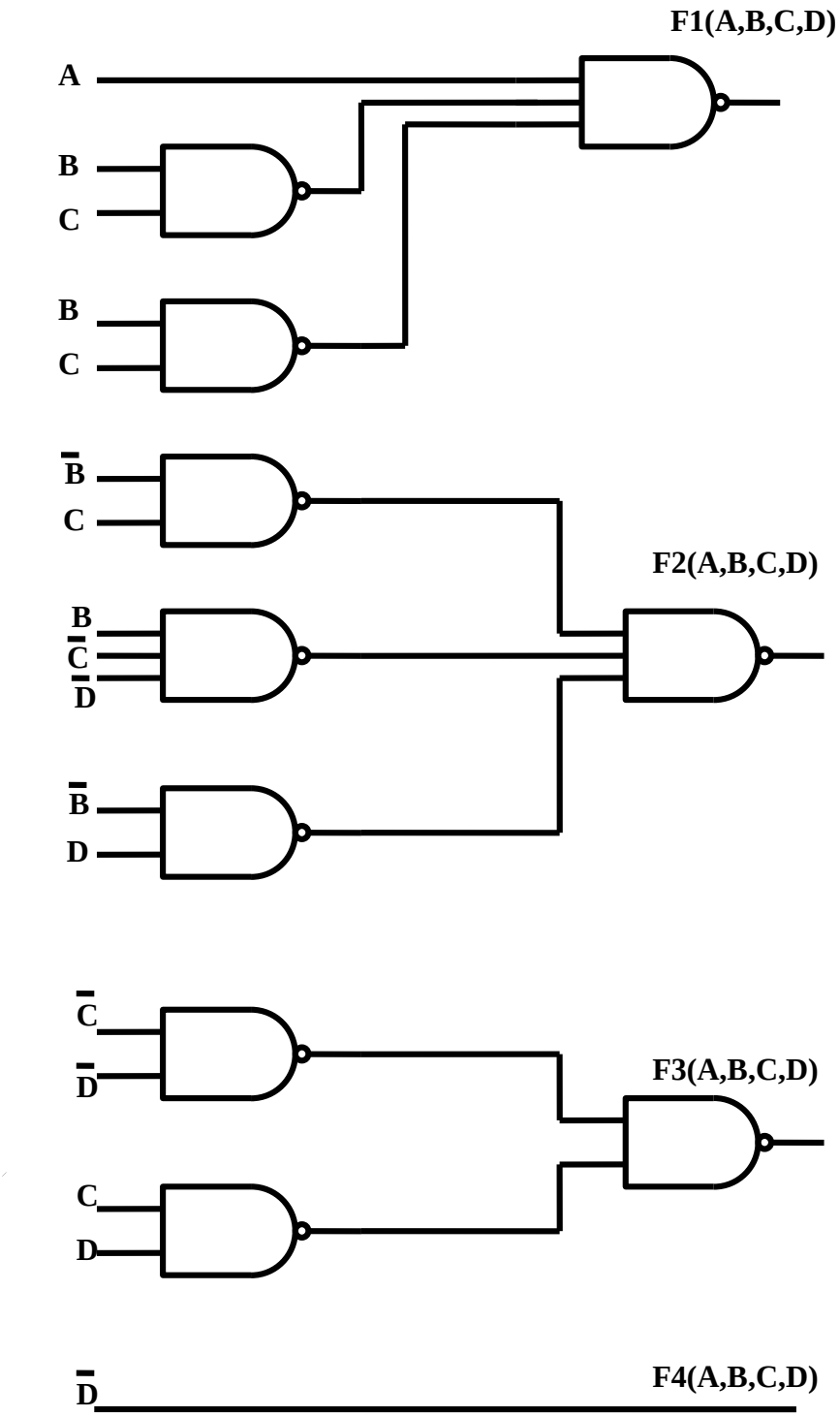


$$F4(A,B,C,D) = \bar{D}$$

Πίνακας Karnaugh και ελάχιστη μορφή της F4(A,B,C,D).

Με την παραπάνω ελαχιστοποίηση που έγινε στις 4 λογικές συναρτήσεις το συνδυαστικό κύκλωμα το οποίο δέχεται στην είσοδο του ένα αριθμό στον κώδικα BCD και το μετατρέπει στην έξοδο σε Excess-3 υλοποιείται με πύλες NAND αποκλειστικά. Αν μας είχε ζητηθεί να το υλοποιήσουμε με NOR θα είχαμε απλοποιήσει τα συμπληρώματα των συναρτήσεων και στη συνέχεια συμπληρώνοντας τις απλοποιημένες συμπληρωματικές συναρτήσεις και κάνοντας χρήση των σχέσεων του De Morgan θα

φθάναμε στις απλοποιημένες συναρτήσεις σε μορφή γινομένου αθροισμάτων. Το παραπάνω κύκλωμα θα μπορούσε να υλοποιηθεί αποκλειστικά με πύλες NOR. Παρακάτω στο σχήμα φαίνεται το κύκλωμα αποκλειστικά με πύλες NAND. Παρατηρούμε ότι η υλοποίηση είναι διεπίπεδη, και ότι η τελευταία συνάρτηση F4 υλοποιείται χωρίς καμία πύλη, απλά με αντεστραμμένη τη μεταβλητή D. Το κύκλωμα υλοποιείται με 7 πύλες NAND 2 εισόδων και τρεις πύλες NAND 3 εισόδων. Η καθυστέρηση του κυκλώματος είναι 2φορές η καθυστέρηση μιας NAND 3 εισόδων διότι αν παρατηρήσουμε το κύκλωμα είναι η πιο αργή διαδρομή.



Διεπίπεδη υλοποίηση αποκλειστικά με πύλες NAND του μετατροπέα κώδικα BCD σε Excess-3.

Αν μου είχε ζητηθεί να σχεδιάσω το μετατροπέα κώδικα από Excess-3 σε BCD τότε οι είσοδοι θα ήταν ο κώδικας Excess-3 και η έξοδος ο κώδικας BCD. Κατά συνέπεια οι αδιάφοροι όροι δε θα ήταν οι ίδιοι με προηγουμένως. Αυτή τη φορά οι συνδυασμοί που δεν υπάρχουν στον

Excess-3 είναι οι παρακάτω 0000(m0), 0001(m1), 0010(m2), 1101(m13), 1110(m14), 1111(m15). Οι μόνιμοι αδιάφοροι όροι στον πίνακα Karnaugh για τις συναρτήσεις εξόδου, δηλαδή του κώδικα BCD φαίνονται στον παρακάτω πίνακα 80φ.

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	×	×		×
$\bar{A}B$				
AB		×	×	×
$A\bar{B}$				

Πίνακας 80φ: Αδιάφοροι όροι για τη μετατροπή Excess-3⇒BCD.

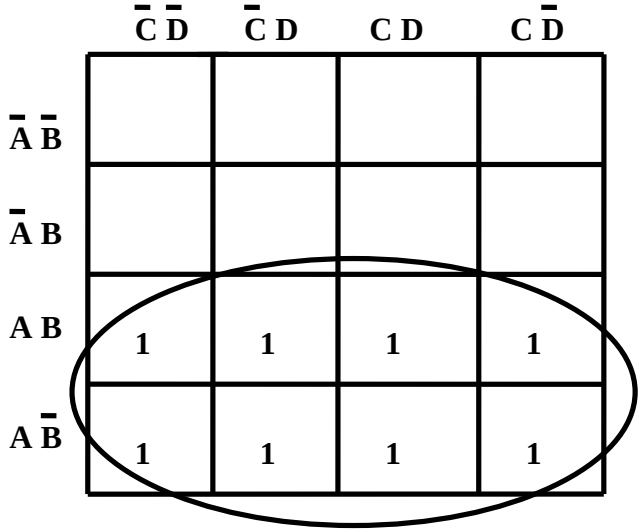
Άσκηση

Σχεδιάστε ένα ψηφιακό κύκλωμα το οποίο μετατρέπει τον κώδικα Gray στο δυαδικό. Ο κώδικας Gray κωδικοποιεί από το 0 μέχρι το 15 και χρησιμοποιείται για έλεγχο. Είναι φτιαγμένος με τέτοιο τρόπο έτσι ώστε πηγαίνοντας στον επόμενο αριθμό ένα μόνο bit αλλάζει. Η αντιστοιχία και κατά συνέπεια ο πίνακας αλήθειας του μετατροπέα φαίνεται παρακάτω στο σχήμα. Η υλοποίηση του κυκλώματος να γίνει αποκλειστικά με ένα τσιπ 7486 το οποίο ως γνωστό περιέχει τέσσερις πύλες XOR δύο εισόδων.

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

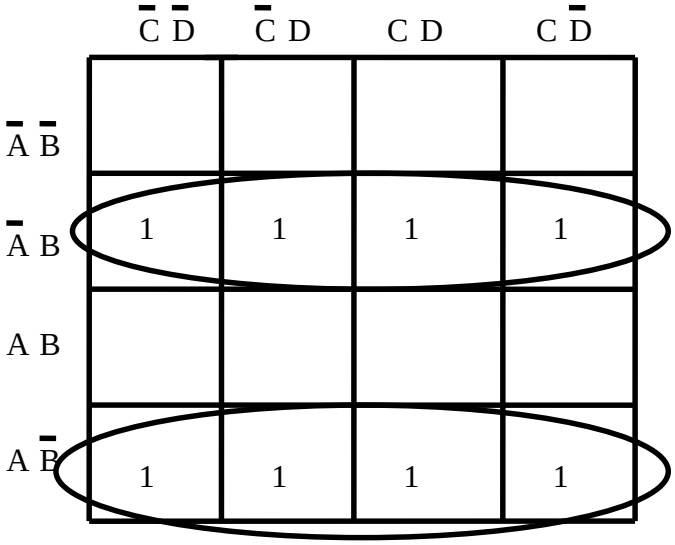
0	1	1	1	0	1	0	1
0	1	0	1	0	1	1	0
0	1	0	0	0	1	1	1
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	1
1	1	1	1	1	0	1	0
1	1	1	0	1	0	1	1
1	0	1	0	1	1	0	0
1	0	1	1	1	1	0	1
1	0	0	1	1	1	1	0
1	0	0	0	1	1	1	1

Πίνακας αλήθειας για τη μετατροπή από τον κώδικα Gray στο δυαδικό σύστημα.



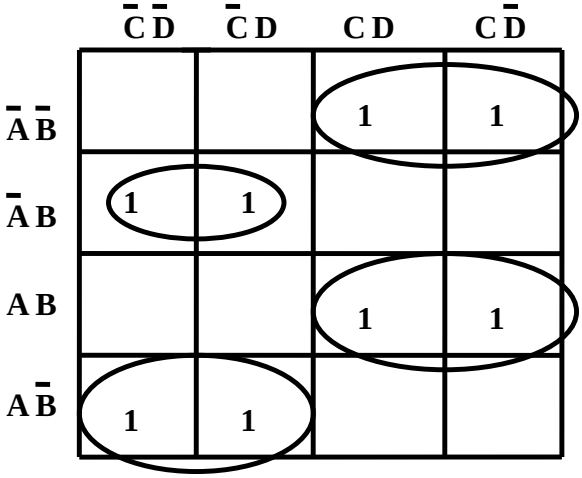
$X(A,B,C,D)=A$

Απλοποίηση της εξόδου Χ.



$F(A,B,C,D)=\bar{A}B+A\bar{B}$

Απλοποιημένη έξοδος $Y(A,B,C,D)=A\oplus B$



Απλοποίηση της Z(A,B,C,D)

$$\begin{aligned} \mathbf{Z(A,B,C,D)} &= \mathbf{\bar{A}\bar{B}C + ABC + \bar{A}B\bar{C} + A\bar{B}\bar{C}} \\ &= \mathbf{(\bar{A}\bar{B} + AB)\cdot C + (\bar{A}B + A\bar{B})\cdot\bar{C}} \\ &= \mathbf{\overline{(A \oplus B)}\cdot C + (A \oplus B)\cdot C} \\ &= \mathbf{A \oplus B \oplus C} \end{aligned}$$

Βλέπουμε ότι και οι τρεις συναρτήσεις της εξόδου έχουν μπει σε τέτοια μορφή και μπορούν να υλοποιηθούν μόνο με πύλες XOR. Απομένει να κοιτάξουμε και την τελευταία W(A,B,C,D).

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$		1		1
$\bar{A}B$	1		1	
AB		1		1
$A\bar{B}$	1		1	

Πίνακας Karnaugh της W(A,B,C,D).

Βλέπουμε ότι η τελευταία συνάρτηση δεν απλοποιείται αν είχαμε θελήσει να την απλοποιήσουμε. Μπορούμε όμως να κάνουμε τις παρακάτω πράξεις:

$$W = \bar{A}\bar{B}\bar{C}D + \bar{A}\bar{B}C\bar{D} + \bar{A}B\bar{C}\bar{D} + \bar{A}BCD + AB\bar{C}D + ABC\bar{D} + A\bar{B}\bar{C}\bar{D} + A\bar{B}CD$$

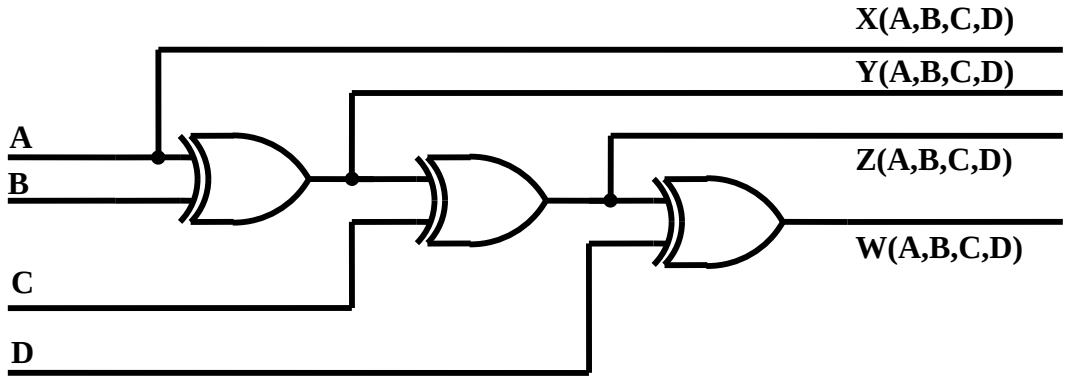
$$= \bar{A}\bar{B}(C \oplus D) + \bar{A}B\overline{(C \oplus D)} + AB(C \oplus D) + A\bar{B}\overline{(C \oplus D)}$$

$$= (\bar{A}\bar{B} + AB)(C \oplus D) + (\bar{A}B + A\bar{B})\overline{(C \oplus D)}$$

$$= \overline{(A \oplus B)}(C \oplus D) + (A \oplus B)\overline{(C \oplus D)}$$

$$= A \oplus B \oplus C \oplus D$$

Και το κύκλωμα φαίνεται στο παρακάτω σχήμα.



Μετατροπέας κώδικα από Gray στο δυαδικό με XOR.

Άσκηση Σχεδιάστε ένα ψηφιακό κύκλωμα το οποίο δέχεται στην είσοδο ένα αριθμό τριών bit και παράγει στην έξοδο το τετράγωνο του. Η υλοποίηση να γίνει αποκλειστικά με πύλες NAND.

Ξεκινώντας το πρόβλημα πρέπει να προσδιορίσουμε τον αριθμό εισόδων και εξόδων του κυκλώματος. Ο αριθμός των εισόδων είναι δεδομένος 3. Ας δούμε όμως πως θα προσδιορίσουμε τον αριθμό των εξόδων. Ο μεγαλύτερος τρίμπιτος αριθμός που μπορεί να παρουσιασθεί δοθεί στην είσοδο του κυκλώματος είναι το 7 (111 στο δυαδικό). Κατά συνέπεια θα πρέπει το κύκλωμα στην έξοδο να μπορεί να παρουσιάσει το 7·7=49=(110001 στο δυαδικό). Το 49 στο δυαδικό για να γραφεί χρειάζονται 6 BIT άρα θα έχουμε 6 εξόδους στο κύκλωμα. Με τα παραπάνω δεδομένα ο πίνακας αλήθειας του κυκλώματος συμπληρώνεται πολύ εύκολα όπως φαίνεται στον παρακάτω πίνακα. Στην έξοδο του κυκλώματος θα εμφανίζεται στο δυαδικό το τετράγωνο του αριθμού εισόδου.

A	B	C	F5	F4	F3	F2	F1	F0
0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	1
0	1	0	0	0	0	1	0	0
0	1	1	0	0	1	0	0	1
1	0	0	0	1	0	0	0	0
1	0	1	0	1	1	0	0	1
1	1	0	1	0	0	1	0	0
1	1	1	1	1	0	0	0	1

Πίνακας

Αν θεωρήσω κάθε μεταβλητή εξόδου σαν μία ανεξάρτητη συνάρτηση Boole των μεταβλητών εισόδου θα έχω:

$$F5(A,B,C)=\Sigma(6,7)=\Pi(0,1,2,3,4,5)$$

$$F4(A,B,C)=\Sigma(4,5,7)=\Pi(0,1,2,3,6)$$

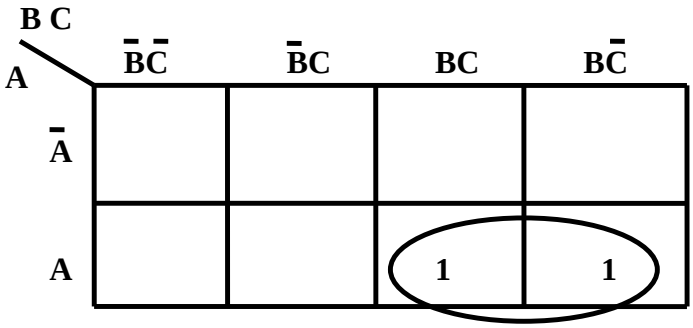
F3(A,B,C)=Σ(3,5)=Π(0,1,2,4,6,7)

F2(A,B,C)=Σ(2,6)=Π(0,1,3,4,5,7)

F1(A,B,C)=0

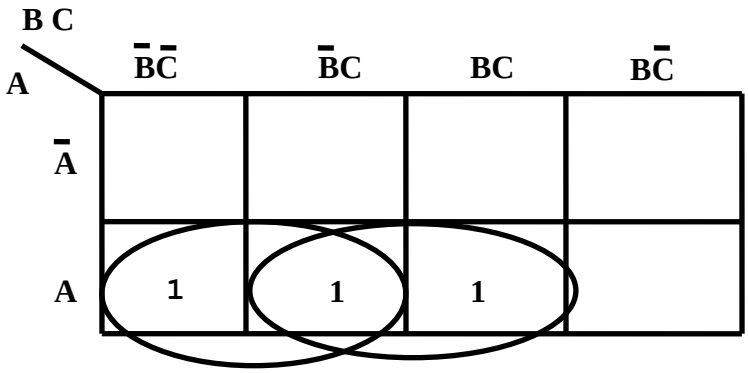
F0(A,B,C)=Σ(1,3,5,7)=Π(0,2,4,6)

Ο πίνακας Karnaugh της συνάρτησης **$F_5(A,B,C)$** φαίνεται παρακάτω στο σχήμα.



Πίνακας της συνάρτησης F5(A,B,C)

Μετά την απλοποίηση παίρνω ότι $F_5(A,B,C)=AB$. Για την F2 ο πίνακας Karnaugh φαίνεται παρακάτω στο σχήμα.



Απλοποίηση της F4(A,B,C).

Με την απλοποίηση θα έχω ότι $F_4(A,B,C) = A\bar{B} + AC$. Για τις υπόλοιπες συναρτήσεις των οποίων οι πίνακες Karnaugh και κατά συνέπεια η απλοποίηση είναι πολύ απλή και θα έχω:

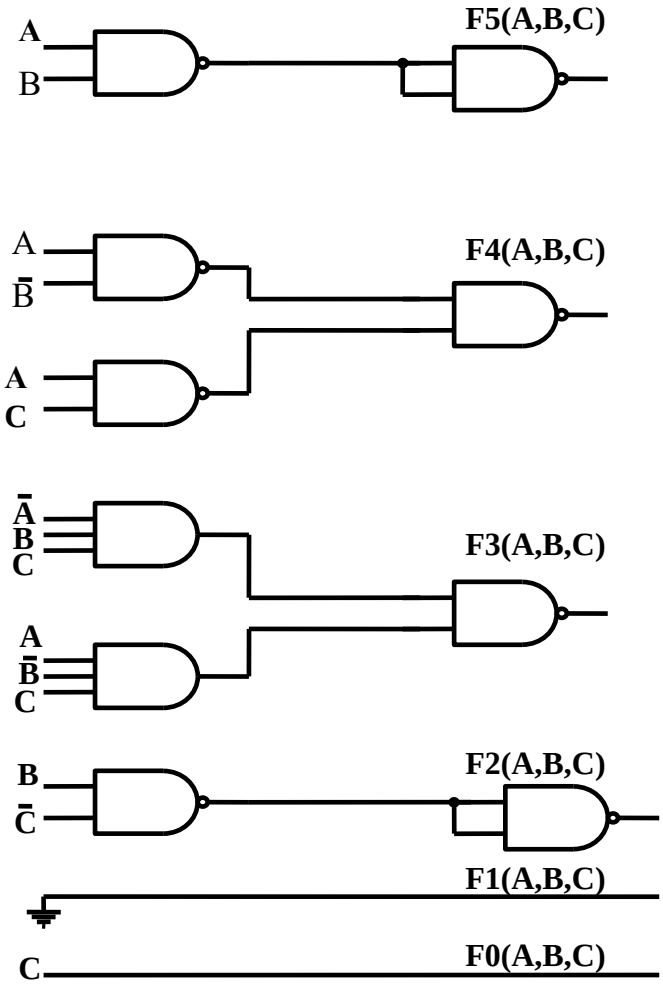
$$F_3(A,B,C) = \bar{A}BC + A\bar{B}C$$

$$F_2(A,B,C) = B\bar{C}$$

$$F_1(A,B,C) = 0$$

$$F_0(A,B,C) = C$$

Η υλοποίηση του κυκλώματος αποκλειστικά με πύλες NAND φαίνεται παρακάτω στο σχήμα.



Κύκλωμα υλοποίησης μετατροπέα

Μετατροπή δυαδικού στον Κώδικα Gray

Ο κώδικας Gray είναι ένας τετράμπιτος κώδικας που κωδικοποιεί τα δυαδικά ψηφία από το 0000 μέχρι το 1111. Η κωδικοποίηση είναι με τέτοιο τρόπο ώστε στη σειρά μέτρησης στον κώδικα Gray να επιτρέπει την αλλαγή σε ένα μόνο ψηφίο. Ο κανόνας είναι ίδιος με αυτόν που καθορίζει τα γειτονικά τετράγωνα στον πίνακα Karnaugh.Ας προσπαθήσουμε να σχεδιάσουμε το κύκλωμα που μετατρέπει ένα δυαδικό στον κώδικα Gray. Ο πίνακας αλήθειας του κυκλώματος φαίνεται παρακάτω. Δεν υπάρχουν

αδιάφοροι όροι στην περίπτωση μετατροπής στον κώδικα Gray. Από τον πίνακα αλήθειας μπορούμε να βρούμε τις κανονικές μορφές των συναρτήσεων στην έξοδο.

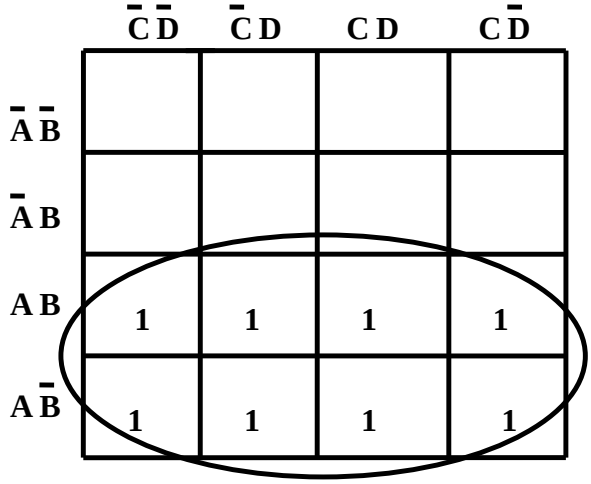
$Y_4 = \Sigma(8,9,10,11,12,13,14,15)$

$Y_3 = \Sigma(4,5,6,7,8,9,10,11)$

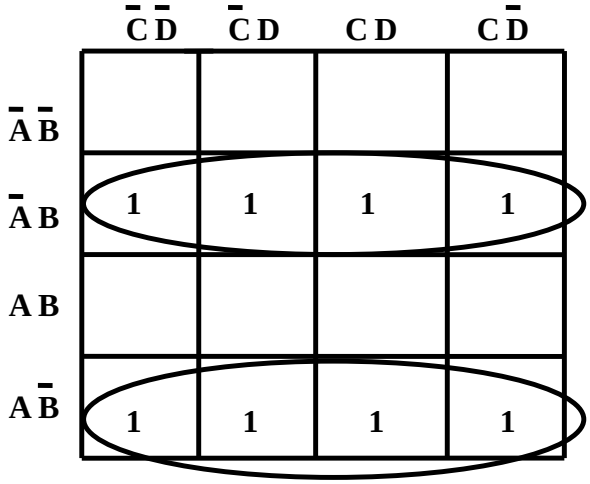
$Y_2 = \Sigma(2,3,4,5,10,11,12,13)$

$Y_1 = \Sigma(1,2,5,6,9,10,13,14)$

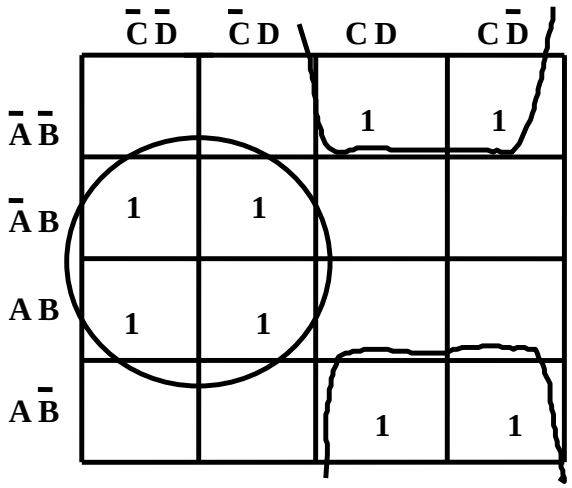
A	**B**	**C**	**D**	**Y_4**	**Y_3**	**Y_2**	**Y_1**
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0



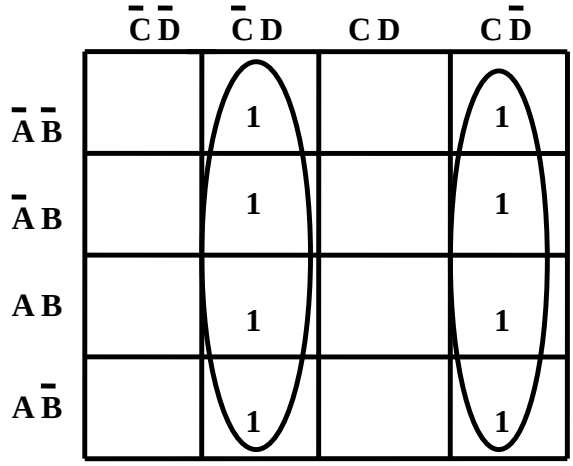
Βλέπουμε μετά την απλοποίηση ότι **$Y_4 = A$**



Η συνάρτηση $Y_3 = \overline{A}B + A\overline{B} = A \oplus B$

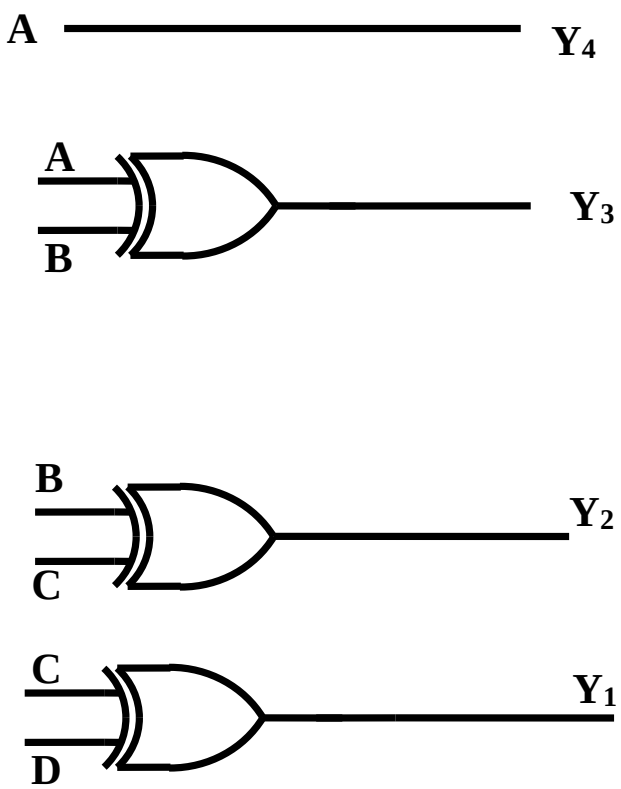


Η συνάρτηση $Y_2 = B\overline{C} + \overline{B}C = B \oplus C$



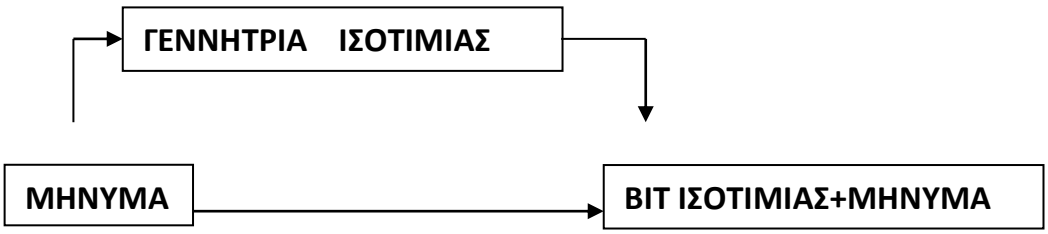
Η συνάρτηση $Y_1 = \overline{C}D + C\overline{D} = C \oplus D$

Βλέπουμε ότι το κύκλωμα του μετατροπέα κώδικα από το δυαδικό στον κώδικα Gray μπορεί να κατασκευαστεί με 3 πύλες XOR των 2 εισόδων.



Γεννήτριες Ισοτιμίας

Οι γεννήτριες ισοτιμίας είναι τα κυκλώματα που παράγουν το BIT ισοτιμίας σε ένα δυαδικό μήνυμα το οποίο μεταδίδεται. Είδαμε ότι μπορούμε να έχουμε 2 διαφορετικούς τύπους ισοτιμίας, την άρτια και την περιττή. Στη περίπτωση της άρτιας ισοτιμίας θα πρέπει ο αριθμός των άσσων του μηνύματος και του BIT ισοτιμίας να είναι άρτιος αριθμός ενώ στην περιττή περιττός. Ας δούμε ένα BLOCK διάγραμμα της διαδικασίας.



Έστω ότι έχουμε ένα μήνυμα 8 BIT A1A2A3A4A5A6A7A8 το παραγόμενο από τη γεννήτρια BIT ισοτιμίας τοποθετείται μπροστά από το μήνυμα και το μήνυμα γίνεται 9 BIT όπως φαίνεται παρακάτω στο σχηματικό διάγραμμα.

A1	**A2**	**A3**	**A4**	**A5**	**A6**	**A7**	**A8**

BIT ΙΣΟΤΙΜΙΑΣ	**A1**	**A2**	**A3**	**A4**	**A5**	**A6**	**A7**	**A8**

Ας προσπαθήσουμε να σχεδιάσουμε το κύκλωμα μιας γεννήτριας άρτιας ισοτιμίας σε ένα τρίμπιτο μήνυμα ABC. Ο πίνακας αλήθειας θα είναι:

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

1	0	1	0
1	1	0	0
1	1	1	1

Βλέπουμε ότι η συνάρτηση BIT άρτιας ισοτιμίας είναι η παρακάτω F(A,B,C)=Σ(1,2,4,7). Στο πίνακα Karnaugh βλέπουμε ότι δεν έχουμε γειτονικά τετράγωνα. Άρα η συνάρτηση δεν απλοποιείται, αλλά πιθανόν να μπορεί να κατασκευαστεί με λιγότερες πύλες όπως θα δούμε παρακάτω:

$$F(A,B,C) = \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\overline{C} + ABC \Rightarrow$$

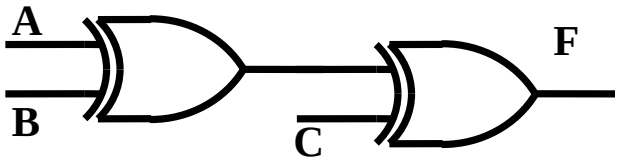
$$F(A,B,C) = \overline{A}(\overline{B}C + B\overline{C}) + A(\overline{B}\overline{C} + BC) \Rightarrow$$

$$F(A,B,C) = \overline{A}(B \oplus C) + A\overline{(B \oplus C)} \Rightarrow$$

$$F(A,B,C) = A \oplus B \oplus C$$

A \ BC	$\overline{B}\overline{C}$	$\overline{B}C$	BC	$B\overline{C}$
$\overline{A}$	0	1	0	1
A	1	0	1	0

Το κύκλωμα υλοποιείται με δύο πύλες XOR όπως φαίνεται παρακάτω

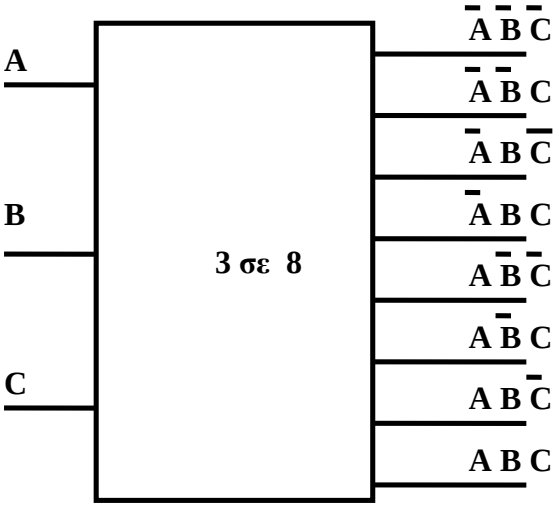


Κεφάλαιο 6. Συνδυαστικά Κυκλώματα με MSI, LSI

Είναι σημαντικό κατά την σχεδίαση και υλοποίηση λογικών κυκλωμάτων να ελέγχουμε σχολαστικά στη βιβλιογραφία αν το κύκλωμα που θέλουμε να υλοποιήσουμε ή μέρος από αυτό υπάρχει ήδη έτοιμο σαν τσιπ. Τα υπάρχοντα κυκλώματα είναι πάρα πολλά και συχνά με κάποιες μικρές παρεμβάσεις μπορούμε να έχουμε το ζητούμενο. Όπως θα δούμε παρακάτω στη Μεσαία Κλίμακα Ολοκλήρωσης (MSI) υπάρχουν τσιπ σχεδιασμένα για να κάνουν τις υλοποιήσεις που είδαμε μέχρι τώρα δαιφορετικές και συνήθως απλούστερες. Χρησιμοποιώντας MSI ολοκληρωμένα κυκλώματα αντί διακριτών πυλών SSI ελαττώνουμε σημαντικά τις καλωδιώσεις και τον αριθμό των τσιπ. Στην συνέχεια του μαθήματος θα παρουσιάσουμε σημαντικά τσιπ MSI που χρησιμοποιούνται στην κατασκευή ψηφιακών κυκλωμάτων. Τα πιο σημαντικά από αυτά είναι τα κυκλώματα αποκωδικοποιητών, τα κυκλώματα πολυπλεκτών και τα κυκλώματα αποπλεκτών.

Αποκωδικοποιητής (Decoder)

Ο αποκωδικοποιητής είναι ένα ολοκληρωμένο κύκλωμα MSI με **n** εισόδους και m εξόδους όπου $m \leq \mathbf{2^n}$ εξόδους. Οι έξοδοι αναπαράγουν όλους τους ελαχιστόρους των μεταβλητών εισόδου. Έτσι έχουμε αποκωδικοποιητές 2 σε 4 ή 2x4, 3 σε 8 ή 3x8, 4 σε 16 ή 4x16 κ.λ.π. Ο πρώτος αριθμός δηλώνει τις εισόδους και ο δεύτερος τον αριθμό εξόδων. Στο παρακάτω σχήμα φαίνεται το κύκλωμα ενός αποκωδικοποιητή 3x8.



Κύκλωμα αποκωδικοποιητή 3x8.

Ο πίνακας αλήθειας του παραπάνω κυκλώματος φαίνεται παρακάτω στο σχήμα και δείχνει ότι κάθε έξοδος του αποκωδικοποιητή (D0, D1, D2, D3, D4, D5, D6, D7) αναπαράγει και από ένα ελαχιστόρο.

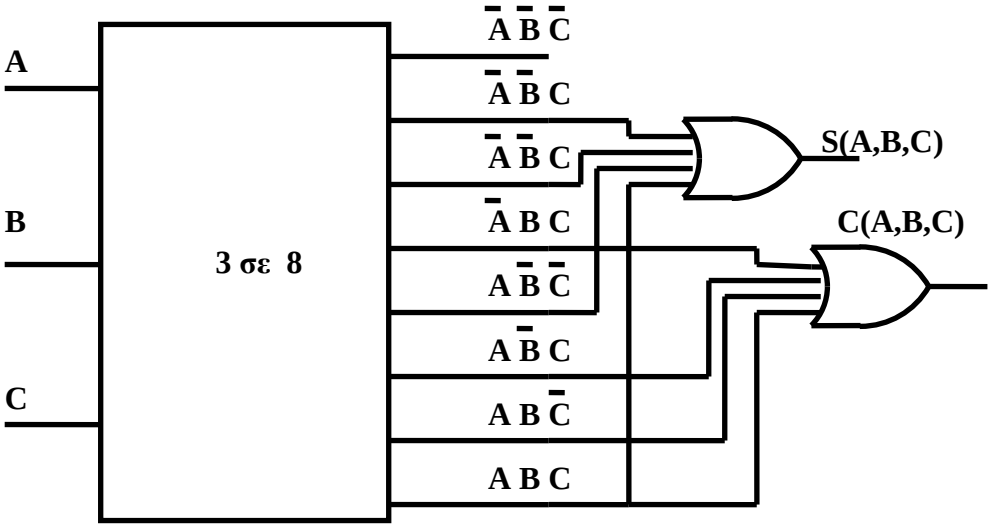
X	**Y**	**Z**	**D_0**	**D_1**	**D_2**	**D_3**	**D_4**	**D_5**	**D_6**	**D_7**
0	0	0	**1**	0	0	0	0	0	0	0
0	0	1	0	**1**	0	0	0	0	0	0
0	1	0	0	0	**1**	0	0	0	0	0
0	1	1	0	0	0	**1**	0	0	0	0
1	0	0	0	0	0	0	**1**	0	0	0
1	0	1	0	0	0	0	0	**1**	0	0
1	1	0	0	0	0	0	0	0	**1**	0
1	1	1	0	0	0	0	0	0	0	**1**

Πίνακας αλήθειας του αποκωδικοποιητή 3x8

Βλέπουμε δηλαδή ότι ο ελαχιστόρος D0 παίρνει την τιμή 1 μόνο όταν A=0, B=0 και C=0, δηλαδή τον συνδυασμό 000, η D1=1 μόνο όταν A=0, B=0 και C=1, δηλαδή για τον συνδυασμό 001 κ.λ.π. Για την υλοποίηση συνδυαστικών κυκλωμάτων με τη χρήση αποκωδικοποιητή χρειαζόμαστε τις κανονικές μορφές των λογικών συναρτήσεων αφού οι έξοδοι των αποκωδικοποιητών περιέχουν τους ελαχιστόρους. Χρειαζόμαστε επίσης πύλες OR. Σε πολλά κυκλώματα τα οποία υλοποιούνται με αποκωδικοποιητή μέρος των εξόδων παραμένει ανενεργό δηλαδή δεν χρησιμοποιείται όπως στην περίπτωση που θα δούμε παρακάτω.

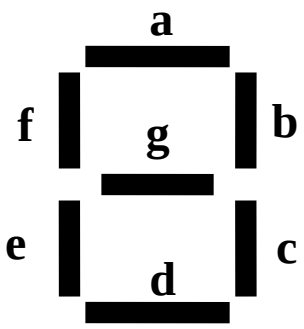
Παράδειγμα: Να υλοποιηθεί το κύκλωμα του πλήρη αθροιστή με τον κατάλληλο αποκωδικοποιητή και με πύλες OR. Στην ανάλυση του κυκλώματος του πλήρη αθροιστή είχαμε βρει ότι οι συναρτήσεις άθροισμα S(A,B,C) και κρατούμενο C(A,B,C) είχαν τις παραπάνω κανονικές μορφές. S(A,B,C)=Σ(1,2,4,7) και C(A,B,C)=Σ(3,5,6,7). Για την υλοποίηση του παραπάνω κυκλώματος χρειαζόμαστε ένα πολυπλέκτη 3x8 και 2 πύλες OR δύο εισόδων. Παρακάτω φαίνεται η υλοποίηση του πλήρη αθροιστή με

αποκωδικοποιητή. Βλέπουμε ότι η έξοδος του αποκωδικοποιητή με τον ελαχιστόρο $\mathbf{m_o}$ δεν χρησιμοποιείται ενώ αυτή του $\mathbf{m_7}$ χρησιμοποιείται 2 φορές οπότε έχουμε ένα κόμβο στο κύκλωμα που οδηγεί τον ελαχιστόρο $\mathbf{m_7}$ στις εισόδους των 2 πυλών OR.



Υλοποίηση του πλήρη αθροιστή με αποκωδικοποιητή και πύλες OR τεσσάρων εισόδων.

Παράδειγμα: Να σχεδιαστεί ένα κύκλωμα το οποίο θα δέχεται στην είσοδο ένα αριθμό στον κώδικα BCD και θα σχηματίζει το δεκαδικό του αντίστοιχο στην έξοδο ενεργοποιώντας τις ανάλογες φωτοδιόδους a,b,c,d,e,f,g που φαίνονται στο παρακάτω σχήμα.



Αναπαράσταση με φωτοδιόδους των δεκαδικών ψηφίων

Όπως φαίνεται στο παρακάτω πίνακα για να παραστήσουμε το μηδέν πρέπει να ενεργοποιηθούν οι φωτοδίοδοι a,b,c,d,e,f, για να παραστήσουμε

το ένα πρέπει να ενεργοποιηθούν οι φωτοδίοδοι b,c, για να παραστήσουμε το δύο οι a,b,g,e,d κ.λ.π. Ανάλογα προχωρούμε και στα υπόλοιπα δεκαδικά ψηφία δηλαδή από το μηδέν μέχρι και το εννέα οπότε ο πίνακας αλήθειας του παραπάνω κυκλώματος συμπληρώνεται εύκολα και είναι αυτός που φαίνεται παρακάτω.

Δεκ/κός	**A**	**B**	**C**	**D**	**a**	**b**	**c**	**d**	**e**	**f**	**g**
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	1	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	1	0	1	1

Πίνακας αλήθειας του ενδείκτη επτά κομματιών

Για να υλοποιήσουμε το παραπάνω κύκλωμα χρειαζόμαστε μόνο τους δέκα πρώτους ελαχιστόρους και πύλες NOR. Είναι γιατί έχουμε να κάνουμε με τον κώδικα BCD ο οποίος κωδικοποιεί τα δέκα ψηφία του δεκαδικού συστήματος αρίθμησης. Οι υπόλοιποι συνδυασμοί είναι αδιάφοροι όροι. Το παραπάνω κύκλωμα ονομάζεται driver του ενδείκτη επτά κομματιών γιατί οδηγεί την τροφοδοσία των επτά leds. Για να το υλοποιήσουμε αρκεί να βρούμε τις συναρτήσεις ελέγχου των LEDs συναρτήσει των μεταβλητών A,B,C,D όπως φαίνεται παρακάτω.

$a(A,B,C,D)=\Sigma(0,2,3,5,6,7,8,9)$, $b(A,B,C,D)=\Sigma(0,1,2,3,4,7,8,9)$

$c(A,B,C,D)=\Sigma(0,1,3,4,5,6,7,8,9)$, $d(A,B,C,D)=\Sigma(0,2,3,5,6,8,9)$

$e(A,B,C,D)=\Sigma(0,2,6,8)$, $f(A,B,C,D)=\Sigma(0,4,5,6,8,9)$

$g(A,B,C,D)=\Sigma(2,3,4,5,6,8,9)$

Το παραπάνω κύκλωμα υλοποιείται εύκολα με ένα αποκωδικοποιητή και 7 πύλες OR.

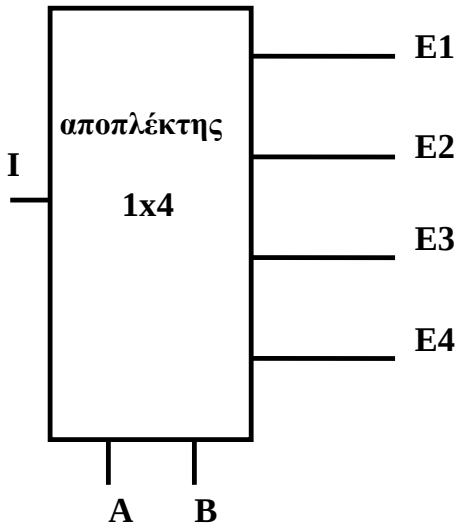
Αποπλέκτης Ο αποπλέκτης είναι ένα σημαντικό ολοκληρωμένο κύκλωμα το οποίο έχει μία είσοδο και πολλές εξόδους m. Ο αριθμός εξόδων είναι συνάρτηση των γραμμών επιλογής οι οποίες καθορίζουν σε ποιά έξοδο θα οδηγηθεί το σήμα της εισόδου. Για να μπορέσουμε να ελέγξουμε τη μεταφορά θα πρέπει αν έχουμε n γραμμές επιλογής να έχουμε το πολύ $m=2^n$ γραμμές εξόδου. Έτσι με 2 γραμμές επιλογής μπορούμε να έχουμε $\mathbf{2^2=4}$ τέσσερις εξόδους, με τρεις γραμμές $\mathbf{2^3=8}$ οκτώ εξόδους με τέσσερις γραμμές **24=16** δεκάξι κ.λ.π. και ονομάζουμε τότε τα αντίστοιχα κυκλώματα αποπλέκτη 1 σε 4 ή 1x4, αποπλέκτη 1 σε 8 ή 1x8, αποπλέκτη 1 σε 16 ή 1x16 κ.λ.π. Βλέπουμε παρακάτω τον πίνακα αλήθειας ενός αποπλέκτη 1x4.

I	**A**	**B**	$\mathbf{E_1}$	$\mathbf{E_2}$	$\mathbf{E_3}$	$\mathbf{E_4}$
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1

Πίνακας αλήθειας του αποπλέκτη 1x4 με είσοδο I

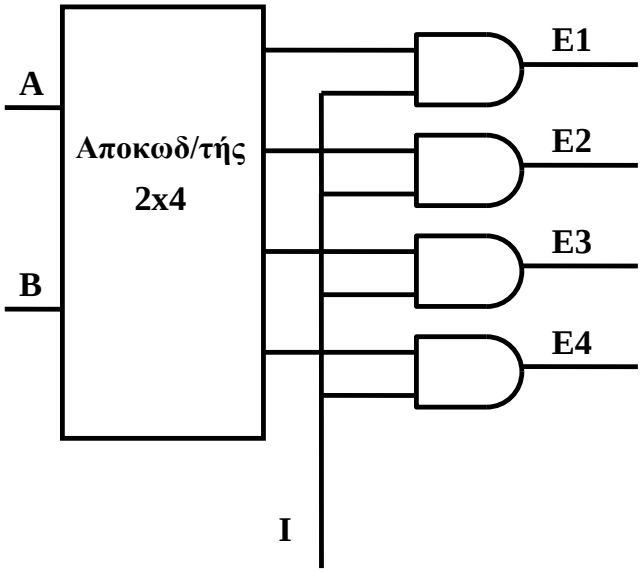
Υποθέτουμε ότι I είναι το σήμα στην είσοδο του πολυπλέκτη 1x4 και A,B είναι οι 2 γραμμές επιλογής. Βλέπουμε ότι για ένα συγκεκριμένο συνδυασμό των A και B μόνο μία έξοδος ενεργοποιείται όπως φαίνεται παρακάτω στο σχήμα.

Βλέπουμε στον παραπάνω αποπλέκτη ότι αν A=0, B=0 τότε και μόνο τότε η έξοδος E1 =1, όταν A=0, B=1 τότε και μόνο τότε η έξοδος E2 =1, αν A=1 και B=0 τότε και μόνο τότε η έξοδος E3 =1, και τέλος αν A=1 και B=1 τότε και μόνο τότε η έξοδος E4 =1.



Σχηματικό διάγραμμα του αποπλέκτη

Στο παρακάτω κύκλωμα βλέπουμε την υλοποίηση ενός αποπλέκτη 1x4 με την βοήθεια ενός αποκωδικοποιητή 2x4 και 4 πύλες AND.

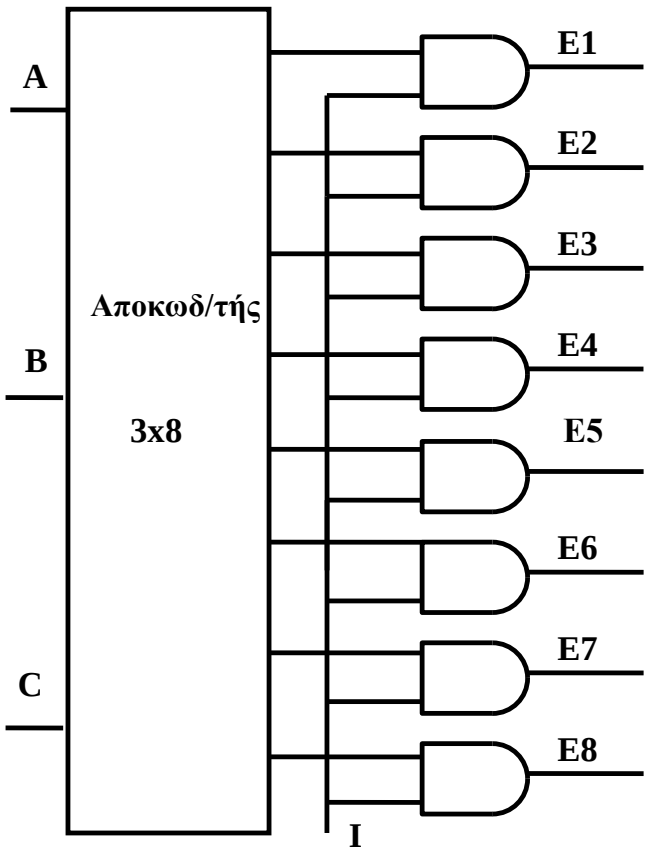


Οι είσοδοι του αποκωδικοποιητή είναι Α και Β ενώ για κάθε μία από τις 4 εξόδους τoy χρειαζόμαστε και από μία πύλη AND δύο εισόδων. Η 1 είσοδος κάθε πύλης AND είναι συνδεδεμένη σε μία έξοδο του αποκωδικοποιητή ενώ η άλλη στην είσοδο I. Οι έξοδοι του αποπλέκτη 1 σε 4 ή 1x4, E1,E2,E3 και E4 βλέπουμε ότι είναι $\mathbf{m_i}$ **AND I**. Δηλαδή αν A=0, B=0 τότε και μόνο τότε η έξοδος E1 =1=I, όταν A=0, B=1 τότε και μόνο τότε η έξοδος E2 =1=I, αν A=1 και B=0 τότε και μόνο τότε η έξοδος E3 =1=I, και τέλος αν A=1 και B=1 τότε και μόνο τότε η έξοδος E4 =1=I.

Αποπλέκτης 1x8

Σύμφωνα με αυτά που είδαμε παραπάνω αν πάρουμε ένα αποπλέκτη 1x8 θα χρειαστούμε 3 γραμμές ελέγχου A, B, C . Ο πίνακας αλήθειας του παραπάνω κυκλώματος θα είναι ο παρακάτω.

I	A	B	C	E1	E2	E3	E4	E5	E6	E7	E8
1	0	0	0	1	0	0	0	0	0	0	0
1	0	0	1	0	1	0	0	0	0	0	0
1	0	1	0	0	0	1	0	0	0	0	0
1	0	1	1	0	0	0	1	0	0	0	0
1	1	0	0	0	0	0	0	1	0	0	0
1	1	0	1	0	0	0	0	0	1	0	0
1	1	1	0	0	0	0	0	0	0	1	0
1	1	1	1	0	0	0	0	0	0	0	1

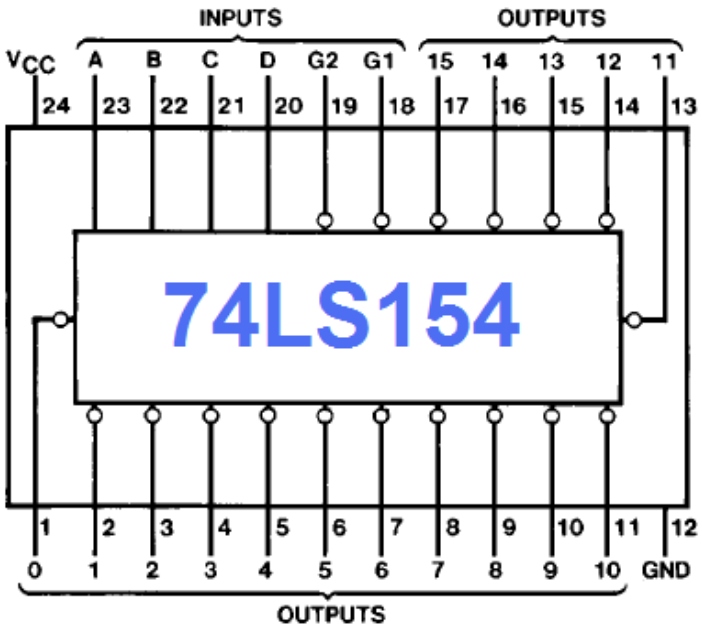


Βλέπουμε επίσης στο προηγούμενο διάγραμμα την υλοποίηση ενός αποπλέκτη 1x8 με ένα αποκωδικοποιητή 3x8 και 8 πύλες AND

Αποπλέκτης 1x16

Ο αποπλέκτης 1x16 είναι ένα κύκλωμα με 1 είσοδο και 16 εξόδους. Μόνο και μόνο 1 έξοδος βλέπει την είσοδο. Για να συμβεί αυτό χρειαζόμαστε 4 γραμμές ελέγχου. Ένα κύκλωμα που κάνει αυτή τη δουλειά είναι το κύκλωμα MSI 74154. Το 74154 έχει 24 PINs από τα οποία το 18 είναι η είσοδος DATA, η είσοδος 19 είναι η γραμμή ελέγχου STROBE η οποία επιτρέπει την λειτουργία της διάταξης. Οι ακροδέκτες 20,21,22,23 είναι οι γραμμές ελέγχου A,B,C,D οι ακροδέκτες από 1 μέχρι 11 και από 13 μέχρι 17 είναι οι 16 έξοδοι του ολοκληρωμένου Y0,Y1,……Y14,Y15, ο ακροδέκτης 24 είναι η τροφοδοσία και ο ακροδέκτης 12 η γείωση.

Το κύκλωμα είναι active low. Για να λειτουργήσει θα πρέπει η είσοδος ελέγχου STROBE να είναι στην κατάσταση low τότε οι γραμμές ελέγχου θα μας δείξουν ποια έξοδος θα είναι επίσης σε κατάσταση με την προϋπόθεση ότι η είσοδος DATA είναι στην κατάσταση Low. Όταν η είσοδος DATA είναι σε κατάσταση High όλες οι γραμμές εξόδου θα είναι σε κατάσταση High , ενώ όταν η είσοδος STROBE είναι σε κατάσταση High τότε πάλι όλες οι γραμμές στην έξοδο θα είναι σε κατάσταση High. Βλέπουμε παρακάτω το BLOCK διάγραμμα του 74154.



Από το διάγραμμα διακρίνουμε ότι η είσοδος G1=DATA και η G2=STROBE ABCD είναι οι γραμμές ελέγχου. Βλέπουμε επίσης τις 16 εξόδους. Από τον πίνακα αλήθειας βλέπουμε ότι αν η είσοδος G2 είναι

Function Table

Inputs						Outputs														
G1	G2	D	C	B	A	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
L	L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	L	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	L	H	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	L	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	H	L	L	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H
L	L	L	H	L	H	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H
L	L	L	H	H	L	H	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H
L	L	L	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H	H	H	H
L	L	H	L	L	L	H	H	H	H	H	H	H	H	L	H	H	H	H	H	H	H
L	L	H	L	L	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H	H
L	L	H	L	H	L	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H
L	L	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H
L	L	H	H	L	L	H	H	H	H	H	H	H	H	H	H	H	H	L	H	H	H
L	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L	H	H
L	L	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L	H
L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	H	X	X	X	X	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
H	L	X	X	X	X	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
H	H	X	X	X	X	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H

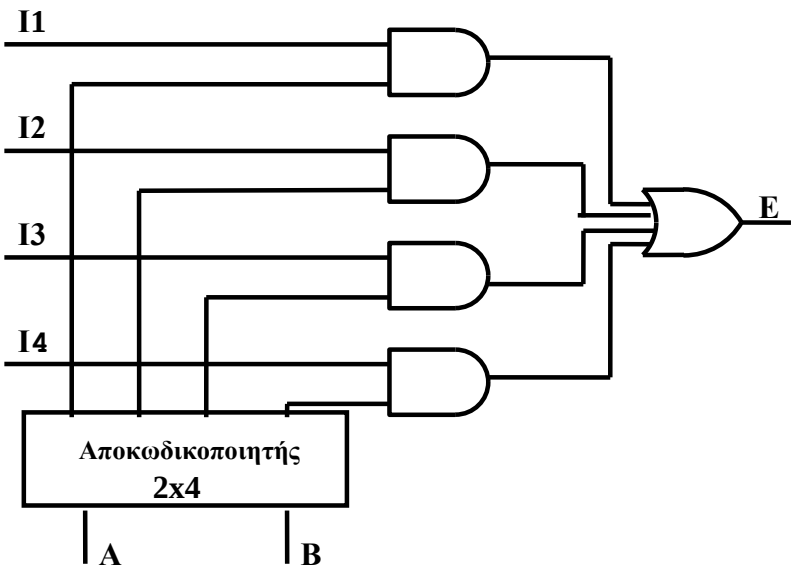
H = High Level, L = Low Level, X = Don't Care

στην κατάσταση High όλες οι έξοδοι είναι στην κατάσταση High , και ισχύει το ίδιο όταν η G1 ή η G2 είναι στην κατάσταση High όπως φαίνεται και στον πίνακα αλήθειας. Το κύκλωμα επαναλαμβάνουμε είναι active low.

Πολυπλέκτης

Είναι ένα λογικό κύκλωμα (επιλογέας δεδομένων) και δουλεύει αντίστροφα από τον αποπλέκτη. Έχει m εισόδους και μία έξοδο. Στην έξοδο μεταφέρεται το περιεχόμενο μίας και μοναδικής μίας εισόδου με τη βοήθεια των n γραμμών επιλογής ή γραμμών ελέγχου της διάταξης έτσι ώστε $m=2^n$. Έτσι αν έχουμε τέσσερις εισόδους I1, I2, I3, I4 η έξοδος E θα είναι ένας συνδυασμός αυτών των εισόδων και των τεσσάρων γραμμών επιλογής όπως φαίνεται παρακάτω.

Το διάγραμμα ενός πολυπλέκτη που φαίνεται στο παρακάτω σχήμα έχει υλοποιηθεί με ένα αποκωδικοποιητή 2x4, 4 πύλες AND 2 εισόδων και μία πύλη OR τεσσάρων εισόδων. Οι γραμμές ελέγχου είναι οι A, B και οι είσοδοι του πολυπλέκτη οι I1, I2, I3, I4.



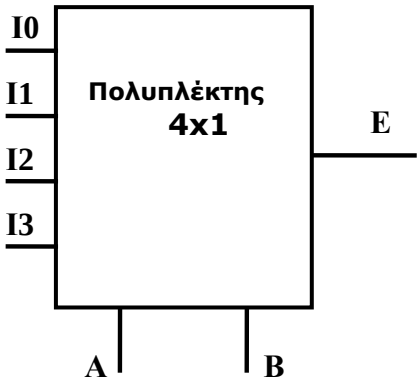
Διάγραμμα ενός πολυπλέκτη τέσσερα σε ένα.

Τα κυκλώματα του πολυπλέκτη και του αποπλέκτη χρησιμοποιούνται πολύ στην σειριακή επικοινωνία όπου ένας μεγάλος αριθμός μηνυμάτων μπορεί με τη βοήθεια του πολυπλέκτη να γίνει ένα μοναδικό μήνυμα και να μεταδοθεί μέσα από μία γραμμή μεταφοράς και στην λήψη να γίνει ακριβώς η αντίθετη διαδικασία. Δηλαδή να γίνει απόπλεξη του μηνύματος. Έτσι είναι δυνατή η επικοινωνία Ν χρηστών μεταξύ τους μέσα από 1 μόνο γραμμή μεταφοράς σημάτων. Τα κυκλώματα πολυπλεκτών και αποπλεκτών έχουν μεγάλη εφαρμογή στις ψηφιακές επικοινωνίες. Με την πολύπλεξη οι συνομιλίες πολλών χρηστών καταλήγουν σε μια γραμμή δηλαδή σε ένα σήμα το οποίο μεταδίδεται μέσω ενός πομπού, το σήμα αυτό λαμβάνεται από το δέκτη, αποπλέκεται και οδηγείται στους αντίστοιχους χρήστες. Επειδή γίνεται σάρωση σε μεγάλη συχνότητα οι αισθήσεις μας δεν μπορούν να αντιληφθούν την διακοπτόμενη επικοινωνία η οποία συμβαίνει στην πραγματικότητα.

Υλοποίηση συναρτήσεων Boole με πολυπλέκτες $2^n \rightarrow 1$ με n γραμμές ελέγχου.

Έστω η συνάρτηση F(A,B,C)=Σ(1,3,5,6). Μου ζητούν να την υλοποιήσω με ένα πολυπλέκτη. Αν πάρω ένα πολυπλέκτη 4x1 ο οποίος έχει 2 γραμμές ελέγχου και συνδέσω εκεί τις 2 από τις τρεις μεταβλητές το ζητούμενο είναι με ποιο τρόπο θα συνδέσω την τρίτη μεταβλητή στις εισόδους του πολυπλέκτη για να αναπαράγεται η συνάρτηση F(A,B,C) στην έξοδο. Ένα

τέτοιο κύκλωμα θα έχει τέσσερις εισόδους μία έξοδο και δύο γραμμές ελέγχου όπως φαίνεται παρακάτω στο σχήμα.



Πολυπλέκτης

Η έξοδος σύμφωνα με τον ορισμό του πολυπλέκτη είναι ίση με:

$$\mathbf{E(I1,I2,I3,I4) = \bar{A}\bar{B}I1 + \bar{A}BI2 + A\bar{B}I3 + ABI4}$$

Μία και μόνο μία είσοδος οδηγείται στην έξοδο και αυτό το καθορίζουν οι γραμμές ελέγχου Α και Β. Έτσι αν Α=0 και Β=0 τότε η έξοδος Ε=Ι1, αν Α=0 και Β=1 τότε η έξοδος Ε=Ι2, αν Α=1 και Β=0 τότε η έξοδος Ε=Ι3 και τέλος αν Α=1 και Β=1 τότε η έξοδος Ε=Ι4. Έχουμε τη δυνατότητα να συνδέσουμε τις εισόδους στο Α στο συμπλήρωμα του Α όπως επίσης και

$$\bar{A} + A = 1$$

δηλαδή στο λογικό 1 στο Vcc ή στο λογικό 0 που είναι το GND. Με αυτά τα δεδομένα βλέπω ότι αν συνδέσω τα Β και C με τις γραμμές ελέγχου το σήμα στην έξοδο θα είναι ίσο με:

$$\mathbf{E(I1,I2,I3,I4) = \bar{B}\bar{C}I1 + \bar{B}CI2 + B\bar{C}I3 + BCI4}$$

Είναι σχετικά εύκολο εμπειρικά να βρούμε με πιο τρόπο θα συνδέσουμε τις εισόδους του πολυπλέκτη με το Α ή με το συμπλήρωμα του ή με το 1 (Vcc) ή με το 0 (GND) για να εμφανίσουμε και την τρίτη μεταβλητή. Η παραπάνω διαδικασία γίνεται ευκολότερη αν χρησιμοποιήσουμε τον πίνακα του παρακάτω σχήματος ο οποίος δεν έχει καμία σχέση με τον πίνακα Karnaugh. Είναι ένας πίνακας διπλής εισόδου με 2 γραμμές και 4 κολώνες. Στις γραμμές έχουμε την 1η μεταβλητή με την συμπληρωματική και την

κανονική της μορφή ενώ στις 4 κολώνες έχουμε τις 4 εισόδους του πολυπλέκτη. Στα 8 τετράγωνα τοποθετούμε τους 8 ελαχιστόρους αριθμώντας από το 0 μέχρι το 7. Στη συνέχεια κυκλώνουμε τους ελαχιστόρους που ορίζουν τη συνάρτηση όπως φαίνεται παρακάτω. Βλέπουμε από το τελικό αποτέλεσμα ότι το I0=0. Το I1 είναι ίσο και με το A αλλά και με το συμπλήρωμα του και το συνδέουμε στο Vcc, το I2=A και το I3=$\bar{A}$.

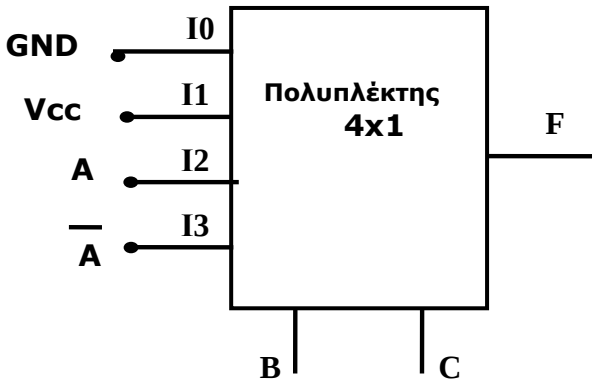
	I_0	I_1	I_2	I_3
$\bar{A}$	0	1	2	3
A	4	5	6	7

Βοηθητικός πίνακας για την υλοποίηση της συνάρτησης με πολυπλέκτη.

Φαίνεται ότι το I1 πρέπει να είναι ίσο με A και με $\bar{A}$ και κατά συνέπεια θα συνδεθεί με το λογικό 1. Το I0=0 θα συνδεθεί με το λογικό μηδέν άρα το συνδέουμε στη γείωση, το I2 με το A και το I3=$\bar{A}$. Αν κάνω υπολογισμούς και σύμφωνα με τη σχέση που έδωσα προηγουμένως θα έχω.

$$E(I_0\ I_1, I_2, I_3) = I_0 \cdot \bar{B} \cdot \bar{C} + I_1 \cdot \bar{B} \cdot C + I_2 \cdot B \cdot \bar{C} + I_3 \cdot B \cdot C$$
$$= 0 \cdot \bar{B} \cdot \bar{C} + (A + \bar{A}) \cdot \bar{B} \cdot C + A \cdot B \cdot \bar{C} + \bar{A} \cdot B \cdot C$$
$$= A \cdot \bar{B} \cdot C + \bar{A} \cdot \bar{B} \cdot C + A \cdot B \cdot \bar{C} + \bar{A} \cdot B \cdot C$$

θα έχω φτιάξει τη συνάρτηση τη ζητούμενη με ένα μόνο κύκλωμα πολυπλέκτη όπως φαίνεται παρακάτω στο σχήμα.



Υλοποίηση λογικής συνάρτησης με πολυπλέκτη 4x1.

Με τον ίδιο τρόπο μπορώ να σχεδιάσω τη συνάρτηση πού φαίνεται παρακάτω **F(A,B,C,D)=Σ(0,1,3,4,8,9,15)** με ένα πολυπλέκτη 8x1. Ο βοηθητικός πίνακας του παρακάτω σχήματος 93 όπως και προηγουμένως θα μου δώσει τη συνδεσμολογία αν υποθέσω ότι έχω βάλει στις γραμμές ελέγχου τις μεταβλητές B,C,D. Αυτός είναι ο γενικός κανόνας που ακολουθούμε κάθε φορά. Δηλαδή η πρώτη μεταβλητή συνδέεται στις εισόδους του πολυπλέκτη και οι άλλες στις γραμμές ελέγχου. Για τον παραπάνω λόγο για να υλοποιήσουμε μια συνάρτηση 3 μεταβλητών χρειαζόμαστε ένα πολυπλέκτη με 2 γραμμές ελέγχου δηλαδή ένα πολυπλέκτη 4x1, για μια συνάρτηση 4 μεταβλητών ένα πολυπλέκτη με 3 γραμμές ελέγχου δηλαδή ένα πολυπλέκτη 8x1. Γενικά για μια συνάρτηση n μεταβλητών χρειαζόμαστε ένα πολυπλέκτη με (n-1) γραμμές ελέγχου (n-1)x1. Για την συνάρτηση F(A,B,C,D) θα έχουμε όπως και προηγουμένως.

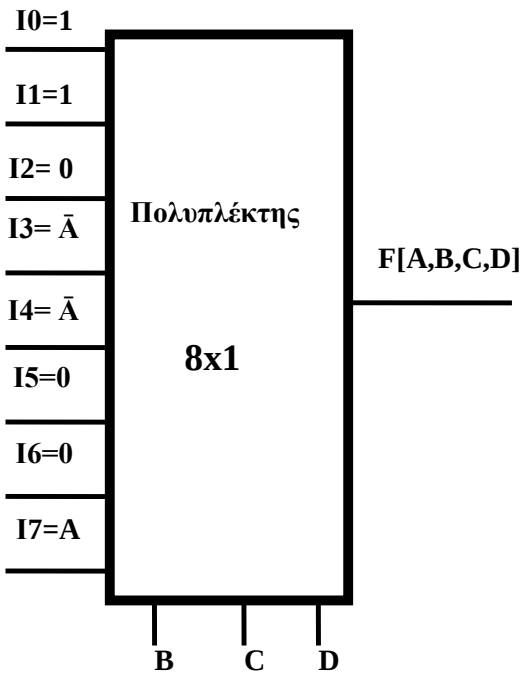
	I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7
$\bar{A}$	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
A	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**

Βοηθητικός πίνακας για την υλοποίηση της προηγούμενης συνάρτησης **F(A,B,C,D)=Σ(0,1,3,4,8,9,15)** με ένα πολυπλέκτη 8x1.

Όπως και προηγουμένως θα έχω να συνδέσω στην είσοδο του πολυπλέκτη το I0= I1=1 με το λογικό 1 δηλαδή το Vcc επειδή A+Ā=1, τις εισόδους I2=I5=I6=0 με το λογικό 0 δηλαδή με το GND, τις εισόδους I3=I4= Ā και τέλος την I7=A. Στην έξοδο του πολυπλέκτη θα έχω:

$$\begin{aligned} E(A,B,C,D) &= (A+\bar{A})\bar{B}\bar{C}\bar{D} + (A+\bar{A})\bar{B}\bar{C}D + \bar{A}\bar{B}CD + \bar{A}B\bar{C}\bar{D} + ABCD \\ &= A\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}\bar{C}\bar{D} + A\bar{B}\bar{C}D + \bar{A}\bar{B}\bar{C}D + \bar{A}\bar{B}CD + \bar{A}B\bar{C}\bar{D} + ABCD \\ &= m8 + m0 + m9 + m1 + m3 + m4 + m15 \\ &= \Sigma(0,1,3,4,8,9,15) \end{aligned}$$

δηλαδή τη ζητούμενη συνάρτηση. Το κύκλωμα υλοποιείται με ένα πολυπλέκτη 8x1 όπως φαίνεται στη παρακάτω σχήμα.



Υλοποίηση συνάρτησης Boole τεσσάρων μεταβλητών με πολυπλέκτη 8x1.

Για να υλοποιήσουμε το κύκλωμα αρκεί να συνδέσουμε το $\mathbf{I_0=I_1=1}$ δηλαδή στο Vcc το $\mathbf{I_3=I_4=\bar{A}}$, το $\mathbf{I_2=I_5=I_6=0}$ δηλαδή στη γείωση και το στο $\mathbf{I_7=A}$.

Κεφάλαιο 7. Λυμένες ασκήσεις συνολικά της 1ης ενότητας

Σε αυτή τη σειρά ασκήσεων θα προσπαθήσουμε να επαναλάβουμε με παραδείγματα τα βασικά σημεία της 1ης ενότητας του μαθήματος που αφορά τη σχεδίαση συνδυαστικών κυκλωμάτων με διακριτές πύλες και με κυκλώματα MSI.

Άσκηση 1: Μετατρέψετε τους ακόλουθους δυαδικούς αριθμούς σε δεκαδικούς.

α)1010

β)1110101

γ)10101111

Απαντήσεις:

α) 1010 $= 1\cdot2^3 + 0\cdot2^2 + 1\cdot2^1 + 0\cdot2^0 = 10$

β) 1110101 $= 1\cdot2^6 + 1\cdot2^5 + 1\cdot2^4 + 0\cdot2^3 + 1\cdot2^2 + 0\cdot2^1 + 1\cdot2^0 = 117$

γ) 10101111 $= 1\cdot2^7 + 0\cdot2^6 + 1\cdot2^5 + 0\cdot2^4 + 1\cdot2^3 + 1\cdot2^2 + 1\cdot2^1 + 1\cdot2^0 = 175$

Άσκηση 2: Να γίνουν οι προσθέσεις

α) 101011+110101 β) 111010 + 101111

Απαντήσεις:

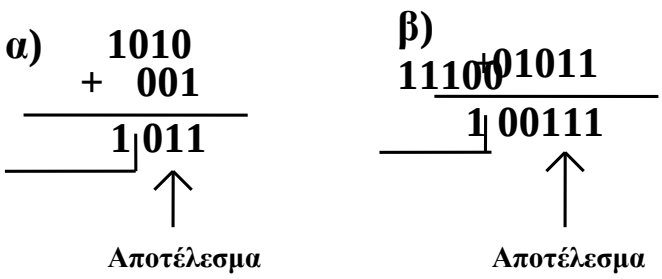
$$\begin{array}{r} 101011 \\ +\ 110101 \\ \hline 1100000 \end{array} \qquad \begin{array}{r} 1110101 \\ +\ 101111 \\ \hline 11010001 \end{array}$$

Άσκηση 3: Να γίνουν οι παρακάτω αφαιρέσεις χρησιμοποιώντας το συμπλήρωμα ως προς δύο.

α. 1010-111

β. 11100-10101

Το συμπλήρωμα του 111 ως προς 2 είναι 001 και το συμπλήρωμα του 10101 ως προς 2 είναι 01011 και κατά συνέπεια οι δύο αφαιρέσεις σύμφωνα με τη θεωρία ισοδυναμούν με δύο προσθέσεις



Άσκηση 4: Να γίνουν οι μετατροπές

α. $(E9)_{16}=(\ ?\ )_{10}$

β. $(\ 17\ )_{8}=(\ ?\ )_{10}$

γ. $11011011=(\ ?\ )_{16}$

δ. $1111111111111=(\ ?\ )_{8}$

Απαντήσεις:

$$(\ E9\ )_{16} = 14.16^1 + 9.16^0 = [233]_{10}$$
$$[17]_8 = 1.8^1 + 7.8^0 = [15]_{10}$$
$$11011011 = 1101\ 1011 = [DB]_{16}$$
$$1111111111\ 111 = 0001\ 1111\ \ 1111\ 1111 = [1FFF]_{16}$$

Άσκηση 5: Να γραφούν στον κώδικα BCD οι παραπάνω δεκαδικοί 159, 1665.

Απαντήσεις:

159 = 000101011001

1665=0001011001100101

Άσκηση 6: Να απλοποιηθούν οι παρακάτω συναρτήσεις Boole.

1. $yx + x\bar{y}$
2. $(x + y)\cdot(x + \bar{y})$
3. $xyz + xy + xyz$
4. $\overline{(A + B)}\cdot\overline{(\bar{A} + \bar{B})}$
5. $xz + \bar{x}yz$

6. $yx + x\bar{y} = xy + x\bar{y} = x \cdot (y + \bar{y}) = x$
7. $(x + y) \cdot (x + \bar{y}) = x + y\bar{y} = x$
8. $xyz + \bar{x}y + xy\bar{z} = xyz + \bar{x}y(z + \bar{z}) + xy\bar{z} = xyz + \bar{x}yz + \bar{x}y\bar{z} + xy\bar{z} =$
$= (x + \bar{x})yz + (x + \bar{x})y\bar{z}$
$= yz + y\bar{z} = y(z + \bar{z}) = y$
9. $\overline{(A + B)} \cdot (\bar{A} + \bar{B}) = \bar{A} \cdot \bar{B} \cdot A \cdot B = 0$

10. $xz + \bar{x}yz = (x + \bar{x}y)z = (x + \bar{x}) \cdot (x + y) \cdot z = (x + y) \cdot z$

Άσκηση 7: Βρείτε το συμπλήρωμα και στην συνέχεια απλοποιείστε τις παρακάτω συναρτήσεις Boole.

$F1 = A \cdot \bar{B} + \bar{C} \cdot \bar{D}$

$F2 = \left[\left[\overline{AB}\right] \cdot A\right] \cdot \left[\left[\overline{AB}\right] \cdot B\right]$

a. $F_1 = A\bar{B} + \bar{C}\bar{D} \Rightarrow \bar{F}_1 = \overline{A\bar{B} + \bar{C}\bar{D}} = \overline{A\bar{B}}\,\overline{\bar{C}\bar{D}} = (\bar{A} + B) \cdot (C + D)$
b. $F_2 = \left[\left(\overline{AB}\right) \cdot A\right] \cdot \left[\left(\overline{AB}\right) \cdot B\right] = [[\bar{A} + \bar{B}] \cdot A][[\bar{A} + \bar{B}] \cdot B] = \bar{B}A \cdot \bar{A}B = 0$
άρα $\bar{F}_2 = 1$

Άσκηση 8: Να βρεθεί ο πίνακας αλήθειας της παρακάτω συνάρτησης

$$F = xy + x\bar{y} + \bar{y}z$$

Για να βρούμε τον πίνακα αλήθειας της παραπάνω συνάρτησης πρέπει να συμπληρωθούν οι όροι με τις μεταβλητές που λείπουν για να βρούμε τους ελαχιστόρους.

$$\begin{aligned} F &= xy + x\bar{y} + \bar{y}z \\ &= xy(z + \bar{z}) + x\bar{y}(z + \bar{z}) + (x + \bar{x}) \cdot \bar{y}z \\ &= xyz + xy\bar{z} + x\bar{y}z + x\bar{y}\bar{z} + x\bar{y}z + \bar{x}\bar{y}z \end{aligned}$$

Γνωρίζοντας τους ελαχιστόρους μπορούμε πολύ εύκολα να βρούμε τον πίνακα αλήθειας όπως φαίνεται στον παρακάτω πίνακα.

x	y	z	F(x,y,z)
0	0	0	0

0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Πίνακας

Άσκηση 9: Δίνεται ο παρακάτω πίνακας αλήθειας για δύο λογικές συναρτήσεις F1 και F2 τριών μεταβλητών.

A	**B**	**C**	**F1(A,B,C)**	**F2(A,B,C)**
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Πίνακας

α. Δώστε τις κανονικές μορφές των συναρτήσεων Boole F1 F2

β. Να απλοποιηθούν σαν αθροίσματα γινομένων.

Η κανονική μορφή μίας συνάρτησης Boole δίνεται σαν άθροισμα ελαχιστόρων η σαν γινόμενο μεγιστόρων. Θα έχουμε λοιπόν

F1(A,B,C)=m1+m2+m4+m7=Σ(1,2,4,7)=Π(0,3,5,6)

F2(A,B,C)=m3+m5+m6+m7=Σ(3,5,6,7)=Π(0,1,2,4)

B C / A	$\bar{B}\bar{C}$	$\bar{B}C$	BC	$B\bar{C}$
$\bar{A}$	0	1	0	1
A	1	0	1	0

Από τον παραπάνω πίνακα Karnaugh βλέπουμε ότι δεν υπάρχουν τετράγωνα γειτονικά και κατά συνέπεια η λογική συνάρτηση **F_1** δεν απλοποιείται. Για την F2 θα έχουμε

B C / A	$\bar{B}\bar{C}$	$\bar{B}C$	BC	$B\bar{C}$
$\bar{A}$	0	0	1	0
A	0	1	1	1

F2(A,B,C)= AB+AC+BC

Άσκηση 10: Δίνεται η συνάρτηση Boole τεσσάρων μεταβλητών

$$F = \bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}CD + \bar{A}B\bar{C}D + A\bar{B}\bar{C}\bar{D} + ABC\bar{D} + ABCD$$

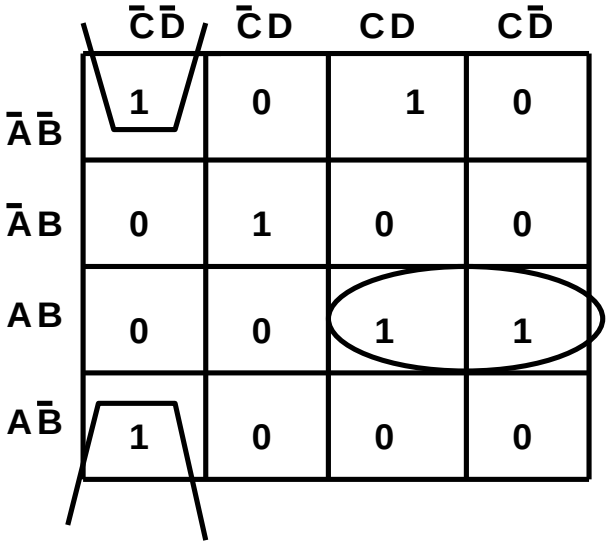
α. Βρείτε τον πίνακα αληθείας

β. Απλοποιείστε την F με τη βοήθεια του χάρτη.

γ. Υλοποιείστε την απλοποιημένη μορφή της F με πύλες AND,OR, και NOT.

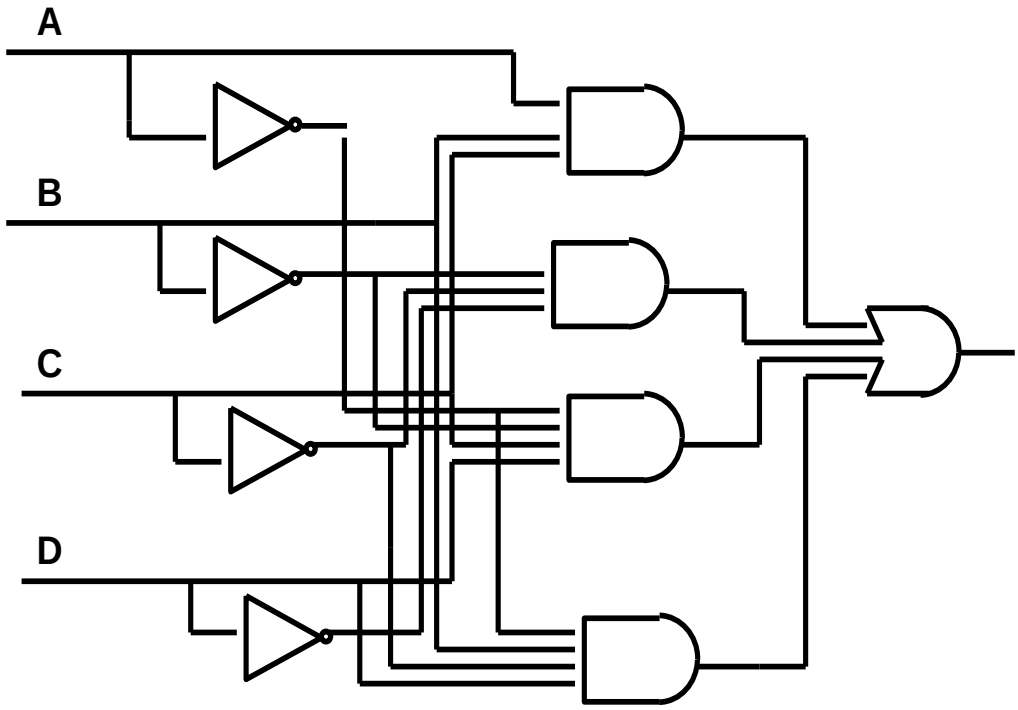
Από την εκφώνηση είναι πολύ εύκολο να βρούμε τον πίνακα αλήθειας όπως φαίνεται παρακάτω.

A	B	C	D	F(A,B,C,D)
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1



Η συνάρτηση απλοποιημένη και το κύκλωμα φαίνονται παρακάτω.

$$F = \overline{B}\overline{C}\overline{D} + \overline{A}\overline{B}CD + \overline{A}B\overline{C}D + ABC$$

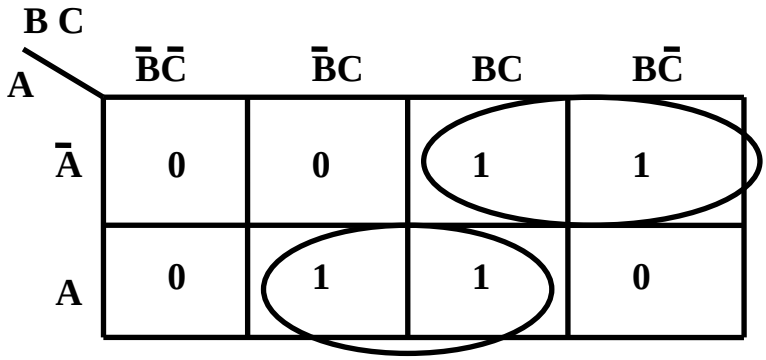


Άσκηση 11: Δίνεται η παρακάτω συνάρτηση F.Να απλοποιηθεί και στην συνέχεια να υλοποιηθεί με πύλες NAND αποκλειστικά υποθέτοντας ότι τα συμπληρώματα δεν είναι διαθέσιμα στην είσοδο.

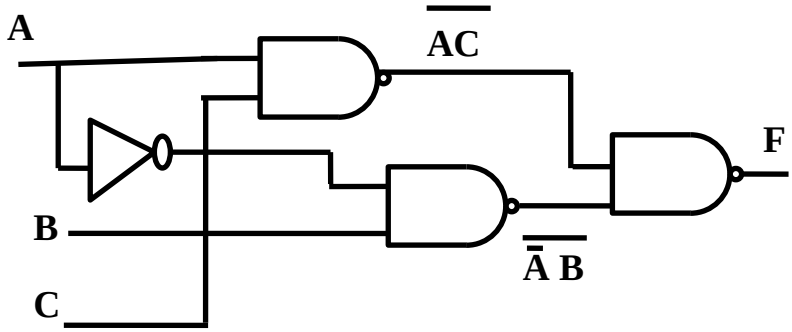
A	B	C	F(A,B,C)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Από τον πίνακα αλήθειας μπορούμε εύκολα να συμπληρώσουμε το χάρτη της συνάρτησης και θα πάρουμε μετά την απλοποίηση

$$F = \overline{A}B + AC$$



Το κύκλωμα είναι αυτό του σχήματος που φαίνεται παρακάτω.



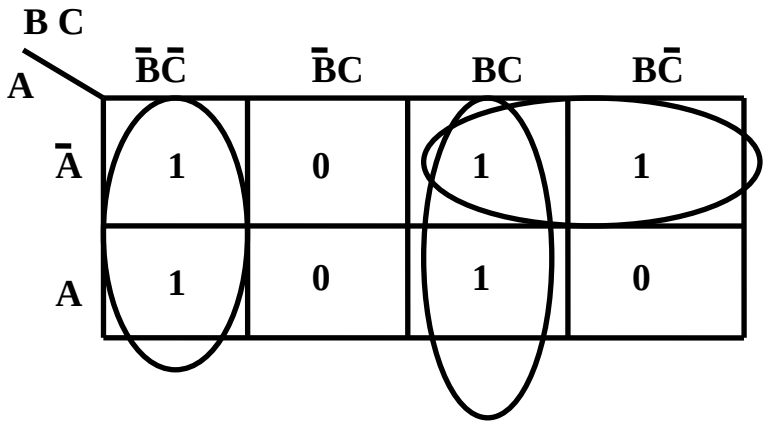
Άσκηση 12: Δίνεται η παρακάτω συνάρτηση Boole.

A	B	C	F(A,B,C)
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

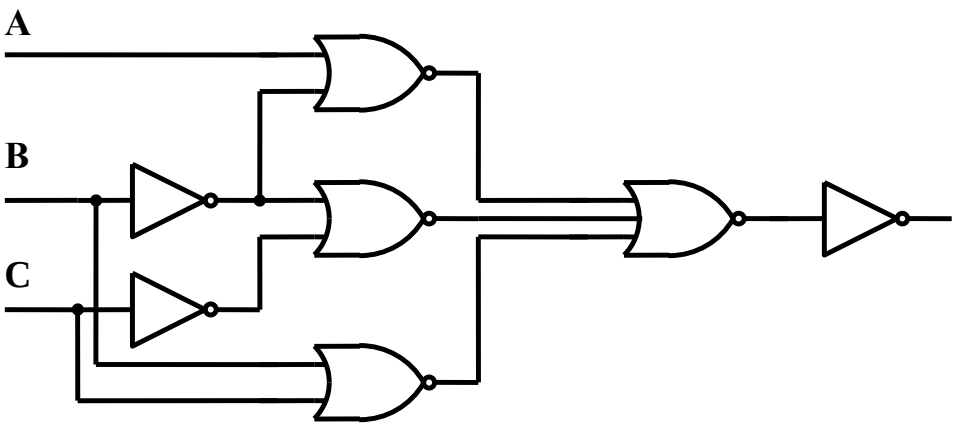
Να απλοποιηθεί με τη μέθοδο του χάρτη και στην συνέχεια να υλοποιηθεί με πύλες NOR και NOT.

Από τον πίνακα αλήθειας μπορούμε εύκολα να βρούμε τη συνάρτηση F. Ο πίνακας θα είναι ο παρακάτω.

$$F = \overline{A}\overline{B}\overline{C} + \overline{A}B\overline{C} + \overline{A}BC + A\overline{B}\overline{C} + ABC$$



Άρα η απλοποιημένη μορφή της F θα είναι $F = \overline{B}\overline{C} + BC + \overline{A}B$ και το κύκλωμα θα υλοποιηθεί με πύλες NOR και NOT όπως φαίνεται στο παρακάτω σχήμα.



Άσκηση 13: Ένα συνδυαστικό κύκλωμα έχει τρεις εισόδους και μία έξοδο. Η έξοδος ισούται με ένα όταν, όλες οι είσοδοι είναι ίσοι με ένα , ή όταν όλες οι είσοδοι είναι ίσοι με μηδέν και όταν μία μόνο είσοδος είναι ίση με ένα.

α. Βρείτε τον πίνακα αλήθειας της συνάρτησης στην έξοδο.

β. Απλοποιήστε την συνάρτηση στην έξοδο σε γινόμενο αθροισμάτων.

γ. Απλοποιήστε την συνάρτηση στην έξοδο σε άθροισμα γινομένων.

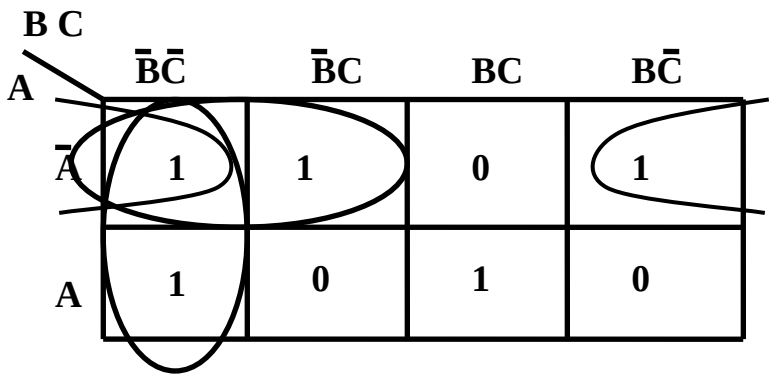
δ. Σχεδιάστε τα παραπάνω λογικά κυκλώματα.

A	B	C	F(A,B,C)
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

$$F = \overline{A}\overline{B}\overline{C} + \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\overline{C} + ABC$$

Ο πίνακας Karnaugh αφήνει ένα όρο ο οποίος δεν μπορεί να συνδυαστεί και η απλοποιημένη μορφή θα είναι

$$F = \overline{A}\overline{B} + \overline{B}\overline{C} + \overline{A}\overline{C} + ABC$$



Από τον πίνακα αλήθειας της F μπορούμε να βρούμε το συμπλήρωμα της

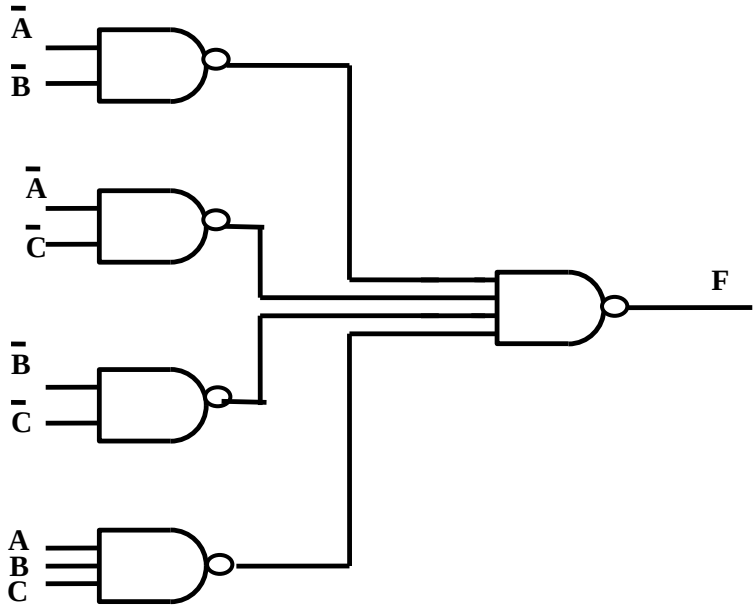
$$\overline{F} = \overline{A}BC + A\overline{B}C + AB\overline{C}$$

Βλέπουμε επίσης ότι το συμπλήρωμα της αρχικής συνάρτησης δεν απλοποιείται επειδή δεν υπάρχουν γειτονικά τετράγωνα. Έτσι η συνάρτηση συμπλήρωμα θα έχει τρεις όρους των τριών μεταβλητών. Αν

συμπληρώσουμε ξανά την προηγούμενη σχέση θα πάρουμε την αρχική μας συνάρτηση σε γινόμενο αθροισμάτων.

$$\overline{\overline{F}} = \overline{\overline{A}BC + A\overline{B}C + AB\overline{C}}$$
$$F = \overline{\overline{A}BC}\cdot\overline{A\overline{B}C}\cdot\overline{AB\overline{C}}$$
$$F = [A + \overline{B} + \overline{C}]\cdot[\overline{A} + B + \overline{C}]\cdot[\overline{A} + \overline{B} + C]$$

Από τα παραπάνω βγαίνει ότι η λογική συνάρτηση F και το συμπλήρωμα της μπορούν πολύ εύκολα να υλοποιηθούν όπως φαίνεται στα παρακάτω σχήματα αντίστοιχα. Βλέπουμε ότι η απλοποιημένη έκφραση της F σε γινόμενο αθροισμάτων υλοποιείται με 5 πύλες NAND ενώ η άλλη έκφραση σε γινόμενο αθροισμάτων υλοποιείται με 4 πύλες NOR.



Άσκηση 14: Σχεδιάστε ένα συνδυαστικό κύκλωμα το οποίο θα δέχεται στην είσοδο του ένα τρίμπιτο αριθμό και θα παράγει στην έξοδο του ένα αριθμό ίσο με το τετράγωνο του αριθμού εισόδου. Για να λυθεί το παραπάνω πρόβλημα θα πρέπει να καθορίσουμε τον αριθμό εξόδων που θα έχει το συνδυαστικό κύκλωμα. Έτσι θα έχουμε ότι ο μέγιστος αριθμός στην είσοδο θα είναι το 7 στο δυαδικό το 111 και κατά συνέπεια στην έξοδο το 49. Το 49 για να γραφεί στο δυαδικό χρειάζονται 6 bit και κατά συνέπεια η

έξοδος θα έχει έξη συναρτήσεις. Ο πίνακας αλήθειας του παραπάνω συνδυαστικού κυκλώματος θα είναι ο παρακάτω:

Δεκαδικός	**A**	**B**	**C**	**B_5**	**B_4**	**B_3**	**B_2**	**B_1**	**B_0**
0	0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	1
2	0	1	0	0	0	0	1	0	0
3	0	1	1	0	0	1	0	0	1
4	1	0	0	0	1	0	0	0	0
5	1	0	1	0	1	1	0	0	1
6	1	1	0	1	0	0	1	0	0
7	1	1	1	1	1	0	0	0	1

Θα έχουμε λοιπόν στην έξοδο.

$$B5(A,B,C) = AB\overline{C} + ABC = AB(C + \overline{C}) = AB$$

$$B4(A,B,C) = A\overline{B}\overline{C} + A\overline{B}C + ABC = A\overline{B} + BC$$

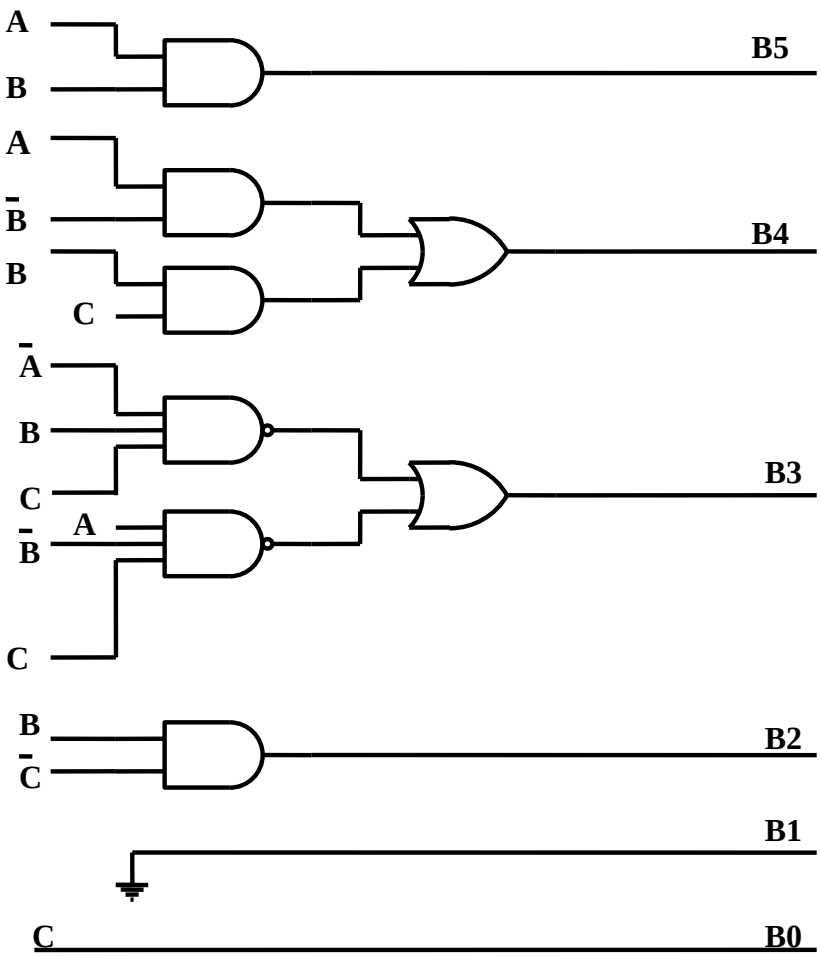
$$B3(A,B,C) = \overline{A}BC + A\overline{B}C$$

$$B2(A,B,C) = \overline{A}B\overline{C} + AB\overline{C} = B\overline{C}$$

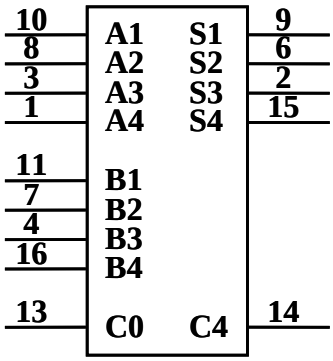
$$B1(A,B,C) = 0$$

$$B0(A,B,C) = \overline{A}\overline{B}C + \overline{A}BC + A\overline{B}C + ABC \Rightarrow$$
$$B0(A,B,C) = \overline{A}C(\overline{B} + B) + AC(\overline{B} + B) \Rightarrow$$
$$B0(A,B,C) = \overline{A}C + AC = (\overline{A} + A)C = C$$

Το συνδυαστικό κύκλωμα που υλοποιεί τις παραπάνω λογικές συναρτήσεις είναι πολύ εύκολο να βγει από τις απλοποιημένες εξισώσεις που δώσαμε και είναι αυτό του παρακάτω σχήματος.



Άσκηση 15: Δίνεται το τσιπ **7483** της TTL το οποίο είναι ένας ολοκληρωμένος τετράμπιτος αθροιστής και ζητείται α) να υλοποιήστε ένα μετατροπέα κώδικα από BCD σε Excess-3 β) και στη συνέχεια ένα ψηφιακό κύκλωμα με δυνατότητα επιλογής μεταξύ αθροιστή και αφαιρέτη. Στο παρακάτω σχήμα βλέπουμε το 74LS83 TTL.



74LS83

Οι αριθμοί για πρόσθεση δίνονται στις εισόδους A1A2A3A4 και B1B2B3B4 C0 είναι το κρατούμενο εξόδου και τέλος το άθροισμα δίνεται από τις εξόδους S1S2S3S4 .

α) Για να υλοποιήσουμε ένα μετατροπέα κώδικα από BCD σε EXCESS 3 με το ολοκληρωμένο κύκλωμα ενός τετράμπιτου πλήρη αθροιστή αρκεί να δίνουμε τον αριθμό στον κώδικα BCD στην είσοδο A1A2A3A4 και το BIB2B3B4 να το κρατούμε σταθερά στο τρία. Έτσι το αποτέλεσμα θα είναι BCD +3 δηλαδή ο κώδικας EXCESS 3 στην έξοδο. Για να συμβεί το παραπάνω θα πρέπει αν το MSB είναι το B1 θα πρέπει η συνδεσμολογία να είναι η ακόλουθη. B1=0, B2=0, B3=1, B4=1 δηλαδή ο αριθμός 0011=3, και το C0=0

β) Το δεύτερο κύκλωμα για να υλοποιηθεί θα κάνουμε χρήση του συμπληρώματος ενός αριθμού. Ξέρουμε ότι με τη βοήθεια του συμπληρώματος αποφεύγουμε την πράξη της αφαίρεσης σε ένα ψηφιακό κύκλωμα και αντί αυτής γίνεται πρόσθεση. Αυτή την ιδιότητα θα χρησιμοποιήσουμε για να μετατρέψουμε τον τετράμπιτο αθροιστή σε τετράμπιτο αφαιρέτη. Το συμπλήρωμα ενός δυαδικού αριθμού ως προς τη βάση 2 για να βρεθεί μετατρέπουμε τα 0 σε 1 και τα 1 σε 0 και προσθέτουμε 1. Αρκεί δηλαδή να αντιστρέψουμε τις εισόδους του αριθμού Β και να κρατήσουμε σταθερά το κρατούμενο εισόδου στο 1. Θέλουμε επίσης να μπορούμε να ελέγχουμε τη λειτουργία του κυκλώματος οπότε θέλουμε να είναι πλήρης αφαιρέτης και οπότε θέλουμε να είναι πλήρης αθροιστής.

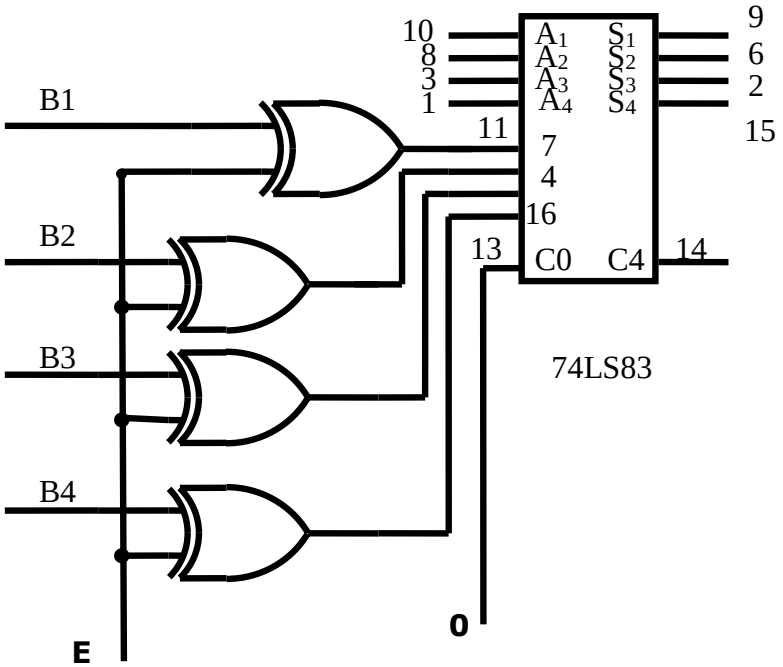
Για να γίνει αυτό θα πρέπει η γραμμή ελέγχου **E** μαζί με κάθε ψηφίο του αριθμού Β να περνάνε από μία πύλη XOR. Έτσι αν η γραμμή ελέγχου είναι ίση με 1 θα παίρνουμε στην έξοδο της πύλης το συμπλήρωμα του κάθε ψηφίου του αριθμού Β διότι:

$$B \otimes 1 = \overline{B}.1 + B.\overline{1} = \overline{B} + 0 = \overline{B}$$

και με το 1 στο κρατούμενο εισόδου θα κάνουμε αφαίρεση. Αντίθετα αν η γραμμή ελέγχου είναι στο 0 τότε

$$B \otimes 0 = \overline{B}.0 + B.\overline{0} = 0 + B1 = B$$

δηλαδή ο αριθμός Β θα μείνει αναλοίωτος και οι έξοδοι των πυλών XOR θα δίνουν τον αριθμό Β κανονικά οπότε θα γίνεται πρόσθεση. Τα παραπάνω φαίνονται στο παρακάτω σχήμα.



Άσκηση 16: Θέλουμε να σχεδιάσουμε ένα ψηφιακό κύκλωμα το οποίο θα ελέγχει τη λειτουργία ενός διανομέα αναψυκτικών. Τα κουμπιά ν, π, κ, ελέγχουν τις ηλεκτροβάννες Ν, Π, Κ και παίρνουμε νερό σκέτο, πορτοκαλάδα η κόκα κόλα αντίστοιχα.

Το νερό είναι δωρεάν. Αν κάποιος χρησιμοποιήσει κέρμα για το νερό θα πρέπει η συσκευή να το γυρίζει πίσω όπως επίσης θα πρέπει να γυρίζει το κέρμα πίσω σε περίπτωση κακού χειρισμού.(δηλαδή αν πατηθούν περισσότερα του ενός κουμπιά συγχρόνως).

α. Γράψτε τις εξισώσεις για τις ηλεκτροβάννες Ν, Π, Κ όπως επίσης και για τη συνάρτηση ΚΕ που επιστρέφει το κέρμα συναρτήσει των μεταβλητών εισόδου που είναι ν, π, κ, ρ (ρ είναι η μεταβλητή για το κέρμα)

β. Απλοποιείστε τις στην απλούστερη μορφή.

γ. Να γίνει υλοποίηση του ψηφιακού συστήματος με πύλες AND, OR και αποκωδικοποιητή.

Ο πίνακας αλήθειας του παραπάνω κυκλώματος θα είναι σύμφωνα με την αρχή λειτουργίας της συσκευής ο παρακάτω.

ν	**π**	**κ**	**ρ**	**Ν**	**Π**	**Κ**	**ΚΕ**
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	0	0
0	0	1	1	0	0	1	0
0	1	0	0	0	0	0	0
0	1	0	1	0	1	0	0
0	1	1	0	0	0	0	0
0	1	1	1	0	0	0	1
1	0	0	0	1	0	0	0
1	0	0	1	1	0	0	1
1	0	1	0	0	0	0	0
1	0	1	1	0	0	0	1
1	1	0	0	0	0	0	0
1	1	0	1	0	0	0	1
1	1	1	0	0	0	0	0
1	1	1	1	0	0	0	1

Η κανονική μορφή των λογικών συναρτήσεων στην έξοδο φαίνεται παρακάτω

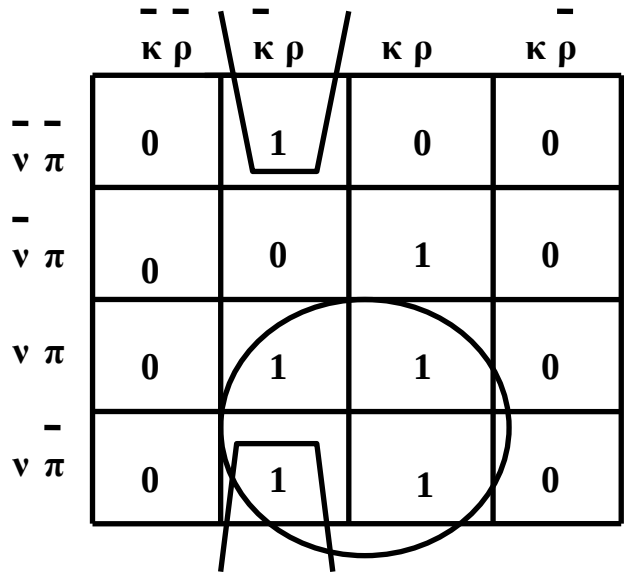
$$N = \nu \cdot \bar{\pi} \cdot \bar{\kappa} \cdot \bar{\rho} + \nu \cdot \bar{\pi} \cdot \bar{\kappa} \cdot \rho$$

$$\Pi = \nu \cdot \bar{\pi} \cdot \bar{\kappa} \cdot \rho$$

$$K = \bar{\nu} \cdot \bar{\pi} \cdot \kappa \cdot \rho$$

$$KE = \bar{\nu} \cdot \bar{\pi} \cdot \bar{\kappa} \cdot \rho + \bar{\nu} \cdot \pi \cdot \kappa \cdot \rho + \nu \cdot \bar{\pi} \cdot \bar{\kappa} \cdot \rho + \nu \cdot \bar{\pi} \cdot \kappa \cdot \rho + \nu \cdot \pi \cdot \bar{\kappa} \cdot \rho + \nu \cdot \pi \cdot \kappa \cdot \rho$$

Η απλοποίηση της συνάρτησης ΚΕ δίνει



Μετά την απλοποίηση θα πάρουμε για τις τέσσερις λογικές συναρτήσεις τα ακόλουθα.

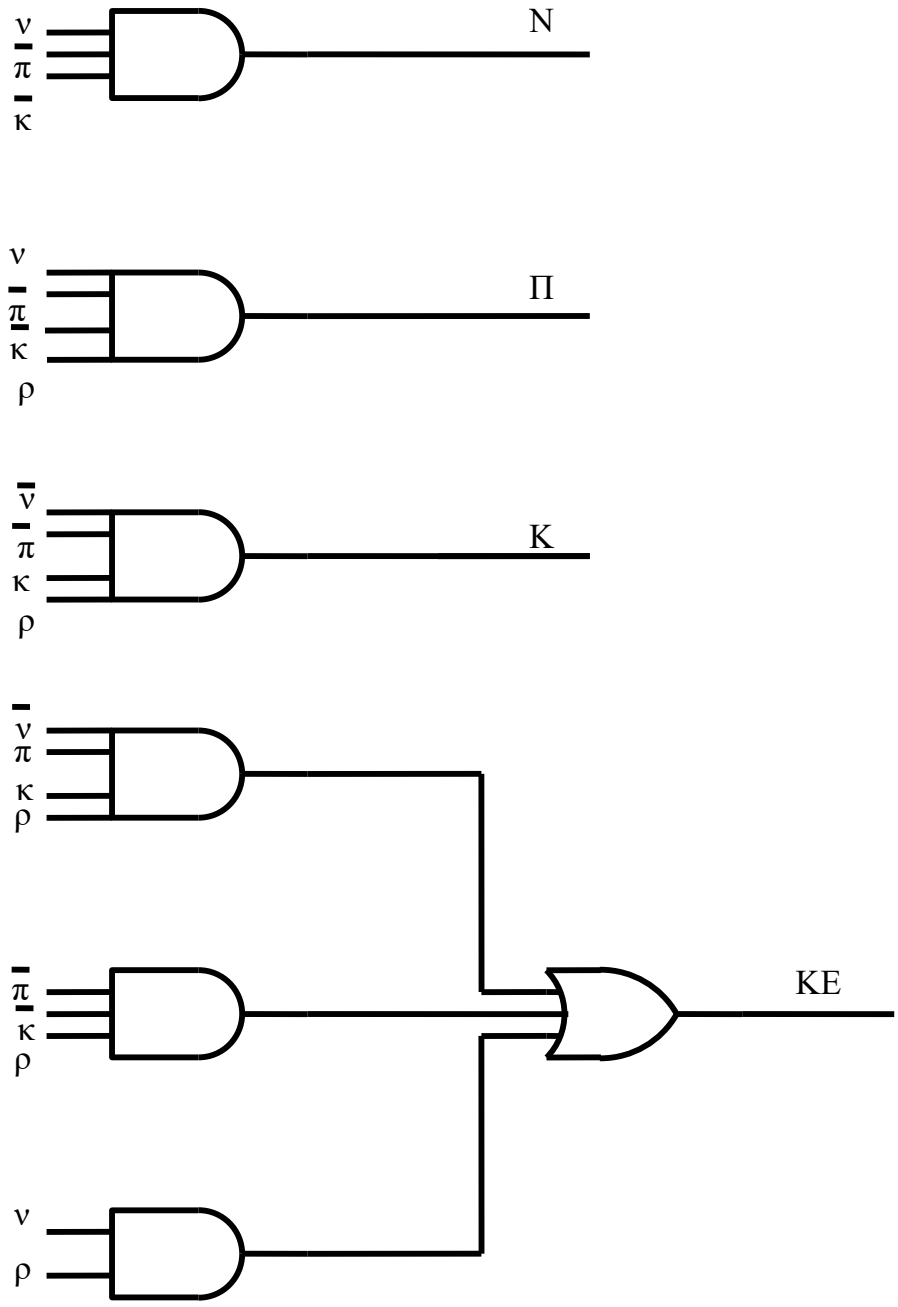
$$N = \nu \cdot \bar{\pi} \cdot \bar{\kappa}$$

$$\Pi = \nu \cdot \bar{\pi} \cdot \bar{\kappa} \cdot \rho$$

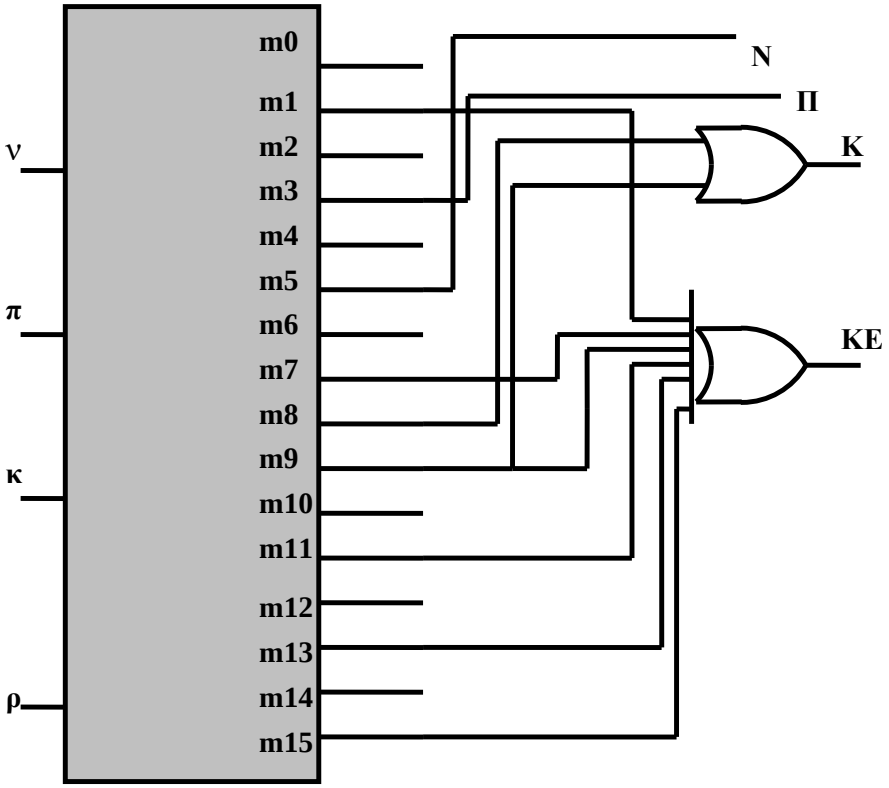
$$K = \bar{\nu} \cdot \bar{\pi} \cdot \kappa \cdot \rho$$

$$KE = \bar{\pi} \cdot \bar{\kappa} \cdot \rho + \bar{\nu} \cdot \pi \cdot \kappa \cdot \rho + \nu \cdot \rho$$

Το κύκλωμα θα είναι του παρακάτω σχήματος.



Για να υλοποιήσουμε το κύκλωμα του διανομές με αποκωδικοποιητή χρειαζόμαστε ένα 4x16 και πύλες OR όπως φαίνεται στο κύκλωμα του παρακάτω σχήματος.


m0
m1
m2
m3
m4
m5
m6
m7
m8
m9
m10
m11
m12
m13
m14
m15
ν
π
κ
ρ
N
Π
Κ
ΚΕ

Αριθμητικές Λογικές Μονάδες ALU(arithmetic Logic Unit)

Οι αριθμητικές λογικές μονάδες (ALU) είναι κυκλώματα τα οποία εκτελούν αριθμητικούς και λογικούς υπολογισμούς. Σε ένα υπολογιστικό σύστημα η αριθμητική λογική μονάδα αποτελεί ένα θεμελιώδες δομικό στοιχείο του επεξεργαστή του υπολογιστή. Στους σύγχρονους υπολογιστές οι επεξεργαστές και οι μονάδες επεξεργασίας γραφικών, διαθέτουν ισχυρές και πολύ πολύπλοκες αριθμητικές λογικές μονάδες.

Η αριθμητική λογική μονάδα εγγυάται ότι ένας υπολογιστής μπορεί να εκτελεί στοιχειώδεις μαθηματικούς υπολογισμούς, όπως πρόσθεση, αφαίρεση, πολλαπλασιασμό και διαίρεση.

Μια ALU, πρέπει να έχει δυνατότητα επεξεργασίας αριθμών χρησιμοποιώντας την ίδια κωδικοποίηση με το υπόλοιπο ψηφιακό κύκλωμα στο οποίο ανήκει. Ένας περίπλοκος επεξεργαστής μπορεί να έχει περισσότερες από μία ALU. Κατά την λειτουργία της φορτώνει δεδομένα από καταχωρητές εισόδου, μια διαφορετική εξωτερική μονάδα συντονίζει την λειτουργία της ALU, και τι πράξεις πρέπει να κάνει στα δεδομένα.

Στο τέλος η ALU αποθηκεύει το αποτέλεσμα σε έναν εξωτερικό καταχωρητή και στη συνέχεια μεταφέρεται στη μνήμη. Ο σχεδιασμός για μία ALU γίνεται για τον προγραμματισμό οιανδήποτε πράξης. Οι μηχανικοί που τις σχεδιάζουν παρέχουν στον επεξεργαστή μια ALU αρκετά δυνατή για να κάνει τον επεξεργαστή γρήγορο, αλλά και όχι τόσο πολύπλοκο για να μην γίνεται το κόστος και το μέγεθος του απαγορευτικό.

Ας δούμε όμως σε ένα παράδειγμα πως σχεδιάζουμε μια αριθμητική λογική μονάδα. Ας θυμηθούμε το κύκλωμα το οποίο δέχεται στην είσοδο του ένα τρίμπιτο δυαδικό αριθμό και αναπαράγει το τετράγωνο του στην έξοδο σε δυαδική μορφή.

Είχαμε δει ότι το κύκλωμα θα έχει 3 εισόδους και 6 εξόδους. Θα πρέπει να αναπαράγει στην έξοδο του το τετράγωνο του 111 που είναι το 49 και για να γραφεί στο δυαδικό θέλουμε 6 ψηφία.

Με τα παραπάνω δεδομένα ο πίνακας αλήθειας του κυκλώματος συμπληρώνεται πολύ εύκολα όπως φαίνεται παρακάτω:

A	B	C	F5	F4	F3	F2	F1	F0
0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	1
0	1	0	0	0	0	1	0	0
0	1	1	0	0	1	0	0	1
1	0	0	0	1	0	0	0	0
1	0	1	0	1	1	0	0	1
1	1	0	1	0	0	1	0	0
1	1	1	1	1	0	0	0	1

Πίνακας

Οι συναρτήσεις πριν την απλοποίηση:

F5(A,B,C)=Σ(6,7)=Π(0,1,2,3,4,5)

F4(A,B,C)=Σ(4,5,7)=Π(0,1,2,3,6)

F3(A,B,C)=Σ(3,5)=Π(0,1,2,4,6,7)

F2(A,B,C)=Σ(2,6)=Π(0,1,3,4,5,7)

F1(A,B,C)=0

F0(A,B,C)=Σ(1,3,5,7)=Π(0,2,4,6)

Και μετά την απλοποίηση:

F_5(A,B,C)=AB.

$$F_4(A,B,C) = A\overline{B} + AC$$

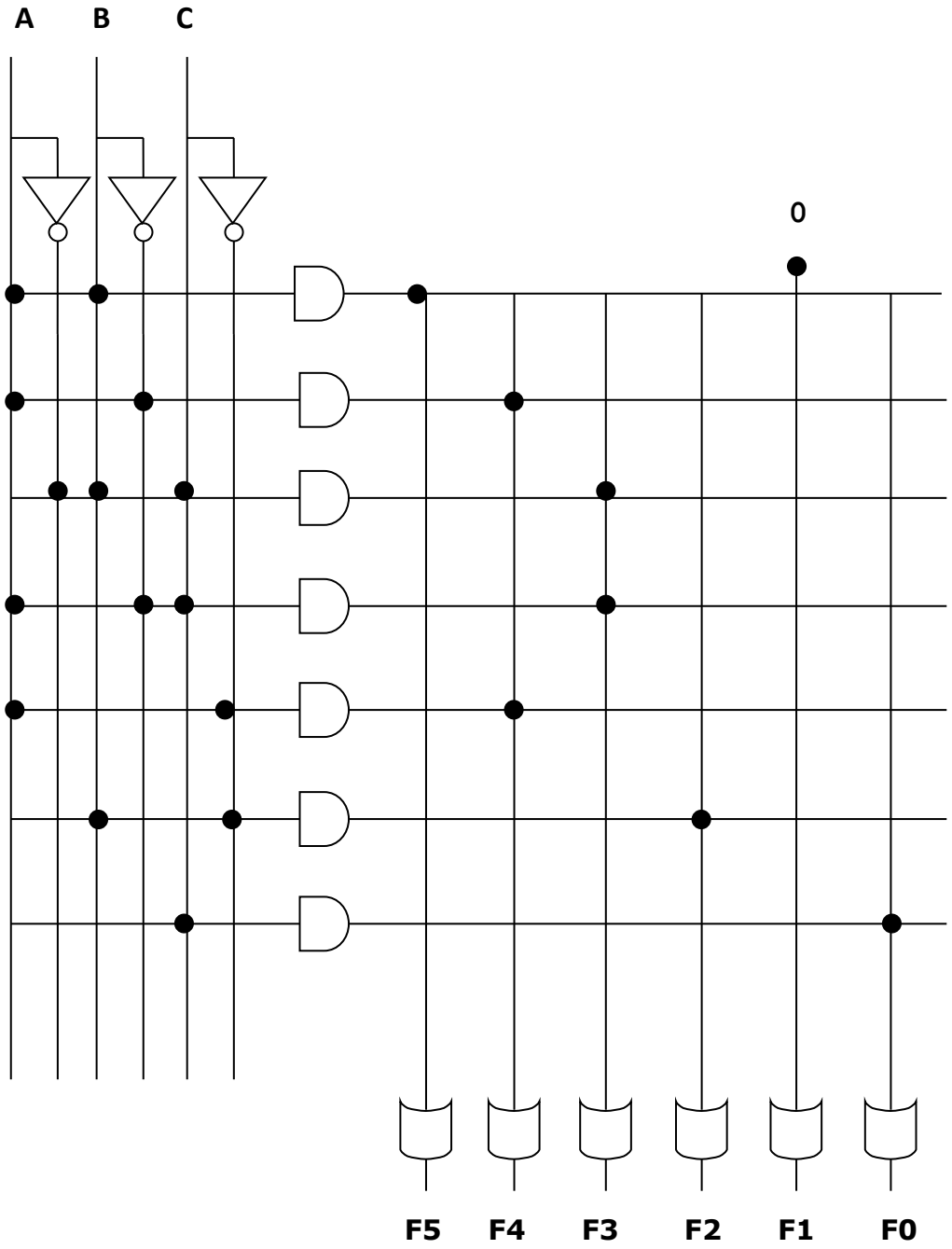
$$F_3(A,B,C) = \overline{A}BC + A\overline{B}C$$

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

$$F_1(A,B,C) = 0$$

$$F_0(A,B,C) = C$$

Το κύκλωμα μιας ALU που θα υλοποιεί τα παραπάνω μπορεί να υλοποιηθεί όπως φαίνεται παρακάτω:


A
B
C
0
F5
F4
F3
F2
F1
F0

Βλέπουμε ότι η αριθμητική λογική μονάδα υλοποιείται εύκολα. Θα πρέπει να απλοποιήσουμε τις λογικές συναρτήσεις και στη συνέχεια θα πρέπει να έχουμε διαθέσιμες πύλες AND και OR. Οι κόμβοι με το έντονο μαύρο χρώμα μας υποδεικνύουν τις συνδέσεις.

Προτεινόμενες Ασκήσεις για Λύση 1ης ενότητας.

1. Μετατρέψετε το δεκαδικό αριθμό 245 στην βάση 3, 4, 7, 8 και 16 και στην συνέχεια στο δυαδικό.

2. Γράψετε το δεκαδικό αριθμό 8620 στον κώδικα BCD στον κώδικα Excess 3 και τέλος στον κώδικα 4221.

3. Ένας δυαδικός κώδικας χρησιμοποιεί 10 bits για να παραστήσει ένα από τα δέκα δεκαδικά ψηφία. Κάθε ψηφίο αναπαριστάνεται με εννέα 0 και ένα 1. Δώστε τον πιο απλό δυαδικό κώδικα.

4. Γράψτε στον κώδικα ASCII τη λέξη Europe

5. Απλοποιήστε τις παρακάτω συναρτήσεις Boole.

a. $F(x, y, z)=\Sigma(2,3,6,7)$

b. $F(A, B, C, D)=\Sigma(7,13,14,15)$

c. $F(x, y, z, w)=\Sigma(2,3,4,5,7,13,15)$

6. Βρείτε τις απλοποιημένες εκφράσεις σε άθροισμα γινομένων των παρακάτω συναρτήσεων.

a. F(x, y, z)=Π(0,1,4,5)

b. F(A, B, C, D)=Π(0,1,2,3,4,10,11)

c. F((x,y,z,w)=Π(1,3,5,7,13,15)

d. F(A,B,C,D)=Σ(4,6,7,15)

7. Σχεδιάστε ένα ψηφιακό σύστημα το οποίο δέχεται ένα τρίμπιτο αριθμό στην είσοδο του και παράγει στην έξοδο ένα δυαδικό αριθμό ίσον με το τετράγωνο του αριθμού εισόδου συν 10.

8. Σχεδιάστε ένα ψηφιακό κύκλωμα το οποίο θα πολλαπλασιάζει επί πέντε ένα δεκαδικό ψηφίο σε αναπαράσταση BCD. Η έξοδος του επίσης θα είναι στον κώδικα BCD.

9. Σχεδιάστε ένα ψηφιακό κύκλωμα το οποίο θα μετατρέπει ένα δυαδικό αριθμό τεσσάρων ψηφίων στον κώδικα BCD.

10. Δείξετε ότι A⊙B⊙C⊙D=Σ(0,3,5,6,9,10,12,15)

11. Υλοποιήστε τις παρακάτω λογικές συναρτήσεις χρησιμοποιώντας αποκλειστικά κυκλώματα ημιαθροιστών.

α. F1(A, B, C)=A⊕B⊕C

β. F2(A, B, C)=Σ(3,5)

γ. F2(A, B, C)=Σ(7)

12. Δίνεται ο παρακάτω πίνακας αλήθειας μίας λογικής συνάρτησης. Να βρείτε τον πίνακα Karnaugh της συνάρτησης , να την απλοποιήστε και στις δύο ελάχιστες μορφές και στην συνέχεια να υλοποιηθεί με το πιο

οικονομικό κύκλωμα αν υποθέσουμε ότι οι πύλες NAND και NOR με ίδιο αριθμό εισόδων έχουν ίδια κατανάλωση.

A	B	C	F(A,B,C)
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

13. Δίνονται παρακάτω οι πίνακες αλήθειας δύο λογικών συναρτήσεων F1, και F2 τεσσάρων μεταβλητών. Να βρείτε το χάρτη κάθε συνάρτησης , να τις απλοποιήστε και στις δύο ελάχιστες μορφές και στην συνέχεια να τις υλοποιήστε με το πιο οικονομικό κύκλωμα αν υποθέσουμε ότι οι πύλες NAND και NOR με ίδιο αριθμό εισόδων έχουν ίδια κατανάλωση.

A	B	C	D	F1(A,B,C,D)
0	0	0	0	1
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0

1	1	0	0	1
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

A	**B**	**C**	**D**	**F2(A,B,C,D)**
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	1
1	1	1	1	0

14. Θέλουμε να σχεδιάσουμε ένα ψηφιακό κύκλωμα το οποίο θα ελέγχει με τον ακόλουθο τρόπο ένα τετράμπιτο μήνυμα. Το ψηφιακό κύκλωμα θα έχει τέσσερις εισόδους a, b, c, d και τέσσερις εξόδους A,B,C,D. Η έξοδος του κυκλώματος A θα δηλώνει πότε το σήμα εισόδου διαιρείται με το 2. Η έξοδος B θα δηλώνει πότε το σήμα εισόδου διαιρείται με το 3. Η έξοδος C θα δηλώνει πότε το σήμα εισόδου διαιρείται με το 4 και τέλος η D δηλώνει πότε το σήμα στην είσοδο θα είναι μικρότερο από το 10.

α. Βρείτε το πίνακα αλήθειας του κυκλώματος

β. Βρείτε τις κανονικές μορφές των λογικών συναρτήσεων στην έξοδο.

γ. Να απλοποιηθούν σε άθροισμα γινομένων.

δ. Να απλοποιηθούν σε γινόμενο αθροισμάτων.

15. Σχεδιάστε με τους κατάλληλους αποκωδικοποιητές τις παρακάτω λογικές συναρτήσεις:

1. F(x,y,z)=Σ(1,2,3,7)
2. G(x,y,z,w)=Σ(2,3,5,6,7,12,13,14,15)

16. Σχεδιάστε με τους κατάλληλους πολυπλέκτες τις παρακάτω λογικές συναρτήσεις:

1. F(x,y,z)=Σ(1,2,3,7)
2. G(x,y,z,w)=Σ(2,3,5,6,7,12,13,14,15)
3. H(A,B,C,D,E)=Σ(0,1,2,3,4,78,12,13,19,22,23,24,25,27,30)

Λύσεις των προτεινόμενων ασκήσεων 1,7,8,11,15

της 1ης Ενότητας του Μαθήματος Λογική Σχεδίαση ΗΜΜΥ

1. Μετατρέψετε το δεκαδικό αριθμό 245 στην βάση 3, 4, 7, 8 και 16 και στην συνέχεια στο δυαδικό.

Λύση: Μετατροπή στο δυαδικό

245	2
122	1
61	0
30	1
15	0
7	1
3	1
1	1
0	1

Κατά συνέπεια θα έχουμε ότι $(245)_{10}=(11110101)_2$

Μετατροπή στη βάση 3 εκτελούμε τη διαίρεση όπως και προηγουμένως.

245	3
81	2
27	0
9	0
3	0
1	0
0	1

Κατά συνέπεια θα έχουμε ότι $(245)_{10}=(11110101)_2=(100002)_3$ Για να γράψουμε τον αρχικό αριθμό στη βάση 4, 8 και 16 μπορούμε να

εκτελέσουμε την ίδια διαδικασία ή να πάρουμε τον δυαδικό και να τον κόψουμε σε δυάδες για τη βάση 4, σε τριάδες για τη βάση 8 και σε τετράδες για τη βάση 16. Όπως:

Δυαδικός	
11 11 01 01	$(3311)_4$
011 110 101	$(365)_8$
1111 0101	$(F5)_{16}$

Τελικά θα έχουμε ότι $(245)_{10}=(11110101)_2= (3311)_4=(365)_8=(F5)_{16}$

7. Σχεδιάστε ένα ψηφιακό σύστημα το οποίο δέχεται ένα τρίμπιτο αριθμό στην είσοδο του και παράγει στην έξοδο ένα δυαδικό αριθμό ίσον με το τετράγωνο του αριθμού εισόδου συν 10.

Λύση: Βλέπουμε ότι η είσοδος του κυκλώματος θα έχει 3 εισόδους. Ο μεγαλύτερος αριθμός θα είναι 111 δηλαδή το 7 του οποίου το τετράγωνο είναι 72=49 θέλουμε όμως συν 10 άρα ο μεγαλύτερος αριθμός που θα αναπαράγει η έξοδος είναι το 59. Για να γράψουμε το 59 χρειαζόμαστε 6 εξόδους τις οποίες ονομάζουμε F6, F5, F4, F3, F2, F1.Θα έχουμε τον παρακάτω πίνακα αλήθειας ο οποίος θα ικανοποιεί την παρακάτω εξίσωση:

x	**y**	**z**	**F6**	**F5**	**F4**	**F3**	**F2**	**F1**
0	0	0	0	0	1	0	1	0
0	0	1	0	0	1	0	1	1
0	1	0	0	0	1	1	1	0
0	1	1	0	1	0	0	1	1
1	0	0	0	1	1	0	1	0
1	0	1	1	0	0	0	1	1
1	1	0	1	0	1	1	1	0
1	1	1	1	1	1	0	1	1

$$\mathbf{(F_6F_5F_4F_3F_2F_1)_2=(xyz)^2+1010}$$

Βλέπουμε ότι:

F6=Σ(5,6,7)

F5=Σ(3,4,7)

F4=Σ(0,1,2,4,6,7)

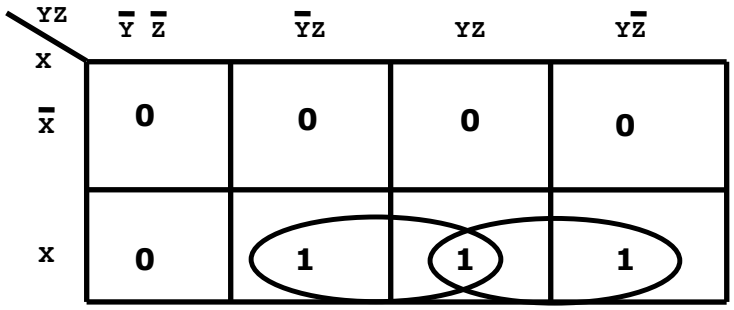
F3=Σ(2,6)

F2=Σ(0,1,2,3,4,5,6,7)=1

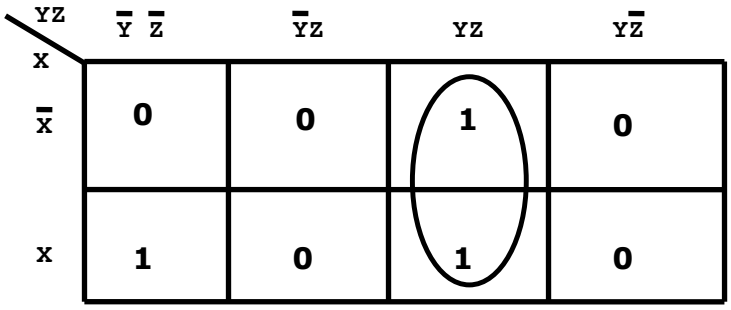
F1=Σ(1,3,5,7)

Η υλοποίηση του παραπάνω κυκλώματος μπορεί να γίνει με πολλούς τρόπους. Αρχικά παρατηρούμε ότι η συνάρτηση F2=1. Οπότε μένουν 5 συναρτήσεις να διαχειριστούμε για να υλοποιήσουμε το κύκλωμα. Ένας τρόπος είναι οι συναρτήσεις να απλοποιηθούν με τη βοήθεια των πινάκων Karnaugh και στη συνέχεια να υλοποιηθεί το κύκλωμα με πύλες NAND ή με πύλες NOR .

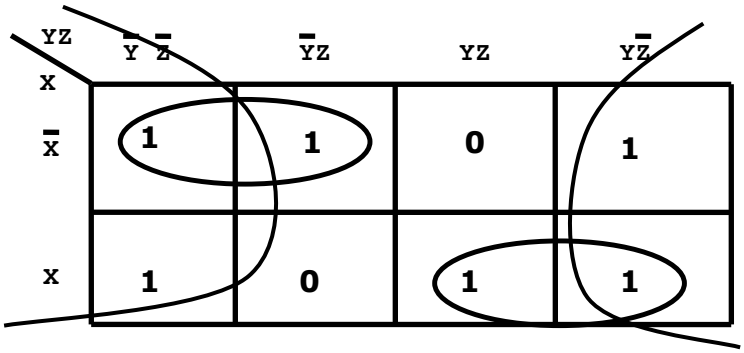
Θα έχουμε για την F6=Σ(5,6,7). Μετά την απλοποίηση $F_6 = xz + xy$



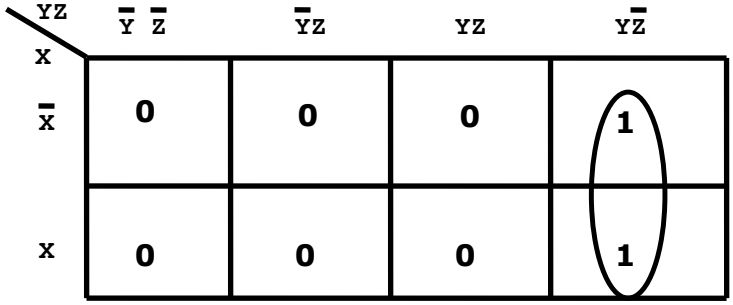
Για την F5=Σ(3,4,7). Μετά την απλοποίηση $F_5 = x\bar{y}\bar{z} + yz$



Για την F4=Σ(0,1,2,4,6,7). Μετά την απλοποίηση $F_4 = \bar{z} + \bar{x}\bar{y} + xy$

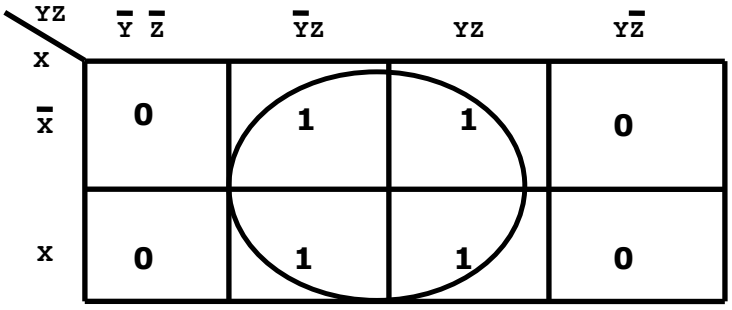


Για την F3=Σ(2,6). Και μετά την απλοποίηση $F_3 = y\bar{z}$



Η F2=1 συνδέεται κατά συνέπεια στο Vcc .

Τέλος η F1=Σ(1,3,5,7). Και μετά την απλοποίηση $F_1 = z$



Το κύκλωμα μετά την συγκεκριμένη απλοποίηση μπορεί να υλοποιηθεί εύκολα αποκλειστικά με πύλες NAND. Χωρίς απλοποίηση μπορεί να υλοποιηθεί το κύκλωμα με ένα αποκωδικοποιητή και 5 πύλες OR . Είναι ο

απλούστερος τρόπος. Μπορεί επίσης να υλοποιηθεί με 5 πολυπλέκτες 4x1 και αντιστροφείς.

8. Σχεδιάστε ένα ψηφιακό κύκλωμα το οποίο θα πολλαπλασιάζει επί πέντε ένα δεκαδικό ψηφίο σε αναπαράσταση BCD. Η έξοδος του επίσης θα είναι στον κώδικα BCD.

Λύση: Έχουμε BCD στην είσοδο άρα θα έχουμε 4 εισόδους. Από τα 10 δεκαδικά ψηφία που κωδικοποιεί το μεγαλύτερο είναι το 9=1001. Αν πολλαπλασιάσουμε το9x5=45 ή 1001x101=0100 0101 στον κώδικα BCD. Θα έχουμε κατά συνέπεια 8 εξόδους στο κύκλωμα. Ο πίνακας αλήθειας θα είναι ο παρακάτω:

A	B	C	D	F8	F7	F6	F5	F4	F3	F2	F1
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	1	0	1
0	0	1	0	0	0	0	1	0	0	0	0
0	0	1	1	0	0	0	1	0	1	0	1
0	1	0	0	0	0	1	0	0	0	0	0
0	1	0	1	0	0	1	0	0	1	0	1
0	1	1	0	0	0	1	1	0	0	0	0
0	1	1	1	0	0	1	1	0	1	0	1
1	0	0	0	0	1	0	0	0	0	0	0
1	0	0	1	0	1	0	0	0	1	0	1

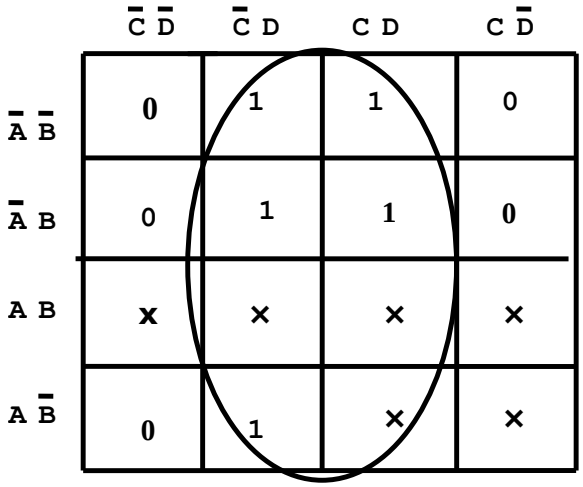
Οι συναρτήσεις θα είναι:

$F_1=\Sigma(1,3,5,7,9)$, $F_2=0$, $F_3=\Sigma(1,3,5,7,9)$, $F_4=0$

$F_5=\Sigma(2,3,6,7)$, $F_6=\Sigma(4,5,6,7)$, $F_7=\Sigma(8,9)$, $F_8=0$

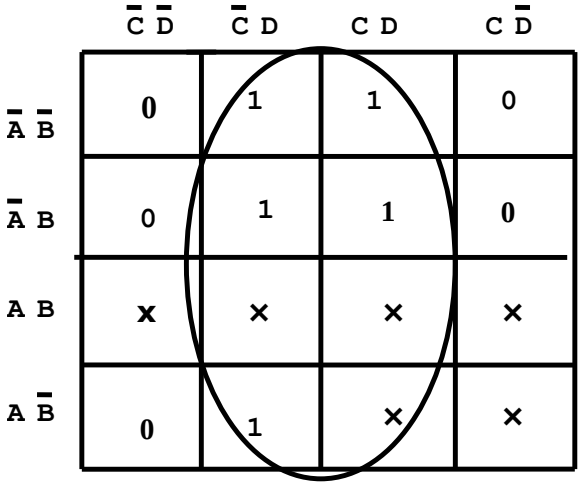
Όπως και προηγουμένως η υλοποίηση του παραπάνω κυκλώματος μπορεί να γίνει με πολλούς τρόπους. Παρατηρούμε ότι από τις 8 συνολικά συναρτήσεις οι 3 είναι μηδενικές. Μένουν κατά συνέπεια 5 συναρτήσεις να διαχειριστούμε για να υλοποιήσουμε το κύκλωμα. Ένας όπως είδαμε και πριν τρόπος είναι οι συναρτήσεις να απλοποιηθούν με τη βοήθεια των πινάκων Karnaugh και στη συνέχεια να υλοποιηθεί το κύκλωμα με πύλες NAND ή με πύλες NOR .

$F_1=\Sigma(1,3,5,7,9)$ και μετά την απλοποίηση F1=D



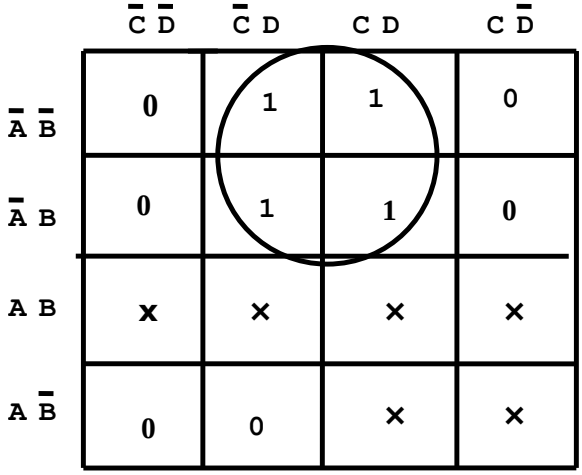
$F_2=0$

$F_3=\Sigma(1,3,5,7,9)$ και μετά την απλοποίηση F3=D

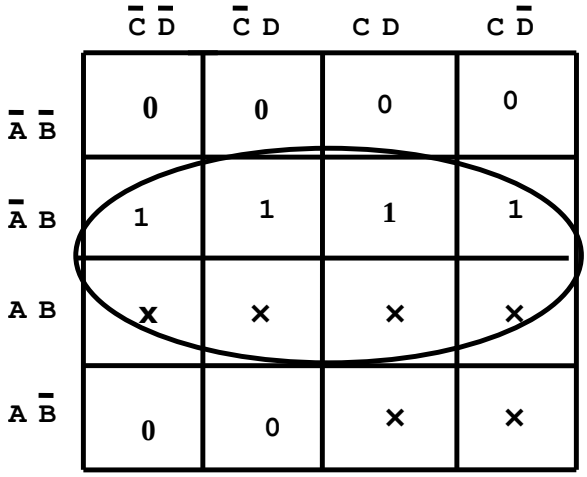


$F_4=0$

F_5=Σ(2,3,6,7) και μετά την απλοποίηση $F_5 = \overline{A}D$

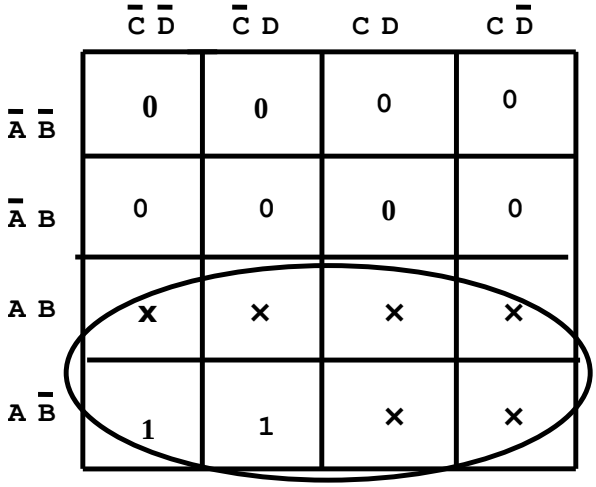


F_6=Σ(4,5,6,7) και μετά την απλοποίηση $F_6 = B$



F_7=Σ(8,9) και μετά την απλοποίηση θα έχουμε F7=A , F_8=0

Βλέπουμε σε όλες τις απλοποιήσεις ότι έχουμε τους γνωστούς αδιάφορους όρους στον πίνακα Karnaugh που είναι οι ελαχιστόροι m10, m11, m12, m13, m14 και ο m15 ή διαφορετικά d=d(10,11,12,13,14,15). Σε πολλές απλοποιήσεις αν τους συμπεριλάβουμε στους συνδυασμούς τα πράγματα βελτιώνονται πολύ και οι συναρτήσεις αποκτούν μια πραγματικά ελάχιστη μορφή όπως είδαμε προηγουμένως. Βασική συνέπεια των παραπάνω είναι η ελαχιστοποίηση των συναρτήσεων και η κυκλωματική τους υλοποίηση με ένα ελάχιστο αριθμό πυλών NAND.



$F_8=0$

Το κύκλωμα υλοποιείται εύκολα με πολλούς τρόπους όπως και προηγουμένως.

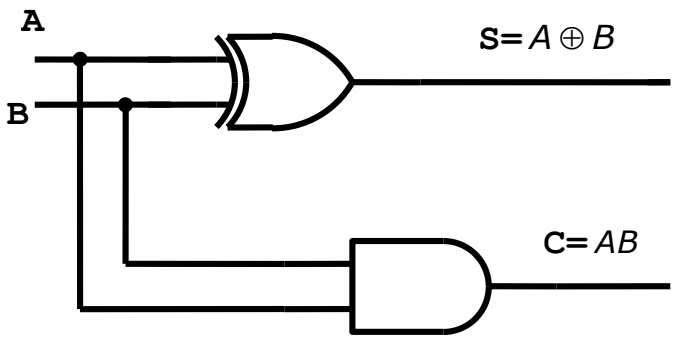
11. Υλοποιήστε τις παρακάτω λογικές συναρτήσεις χρησιμοποιώντας αποκλειστικά κυκλώματα ημιαθροιστών.

α. F1(A, B, C)=A⊕B⊕C

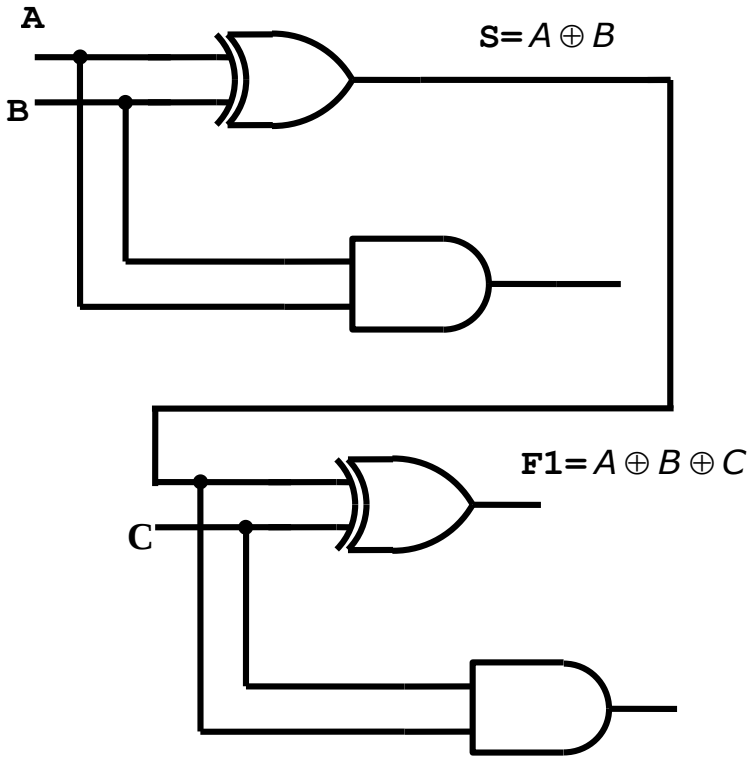
β. F2(A, B, C)=Σ(3,5)

γ. F3(A, B, C)=Σ(7)

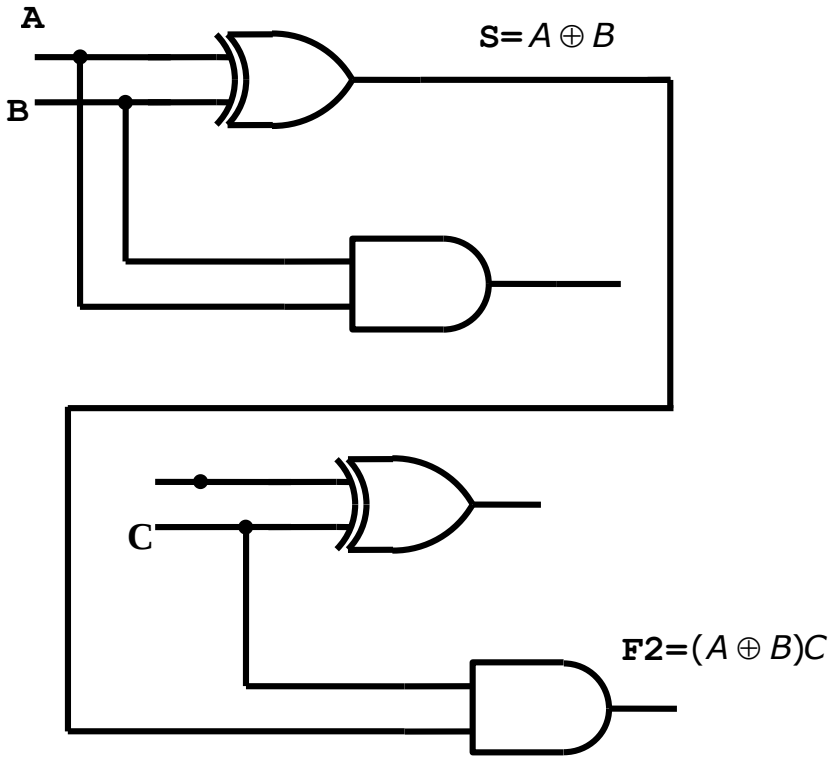
Λύση: Το κύκλωμα ημιαθροιστή είναι το παρακάτω:



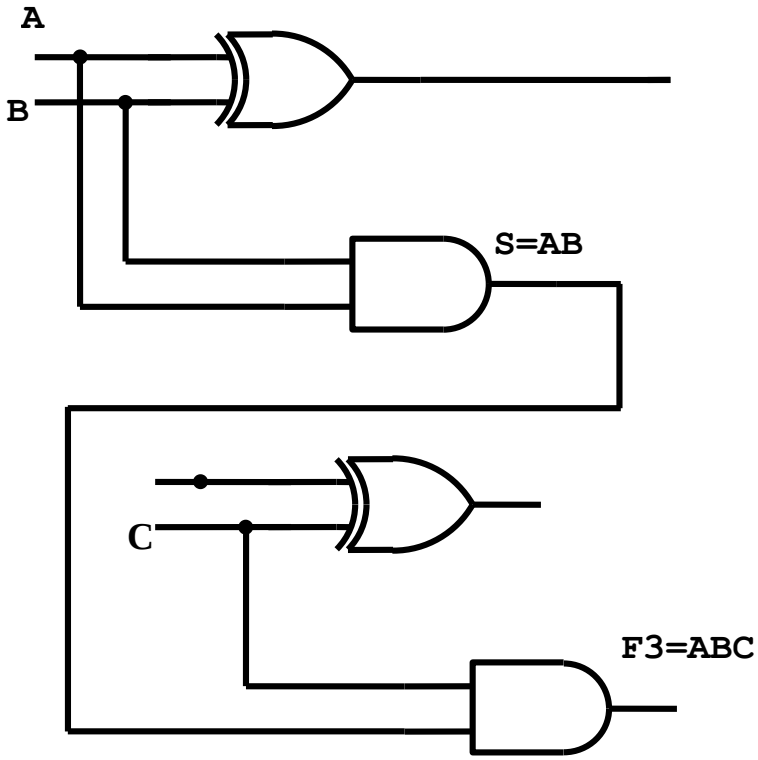
Για να υλοποιήσουμε την συνάρτηση α. χρειαζόμαστε 2 κυκλώματα ημιαθροιστών συνδεδεμένα όπως:



Για την συνάρτηση β F2=Σ(3,5) δηλαδή $F_2 = \bar{A}BC + A\bar{B}C = (A \oplus B)C$



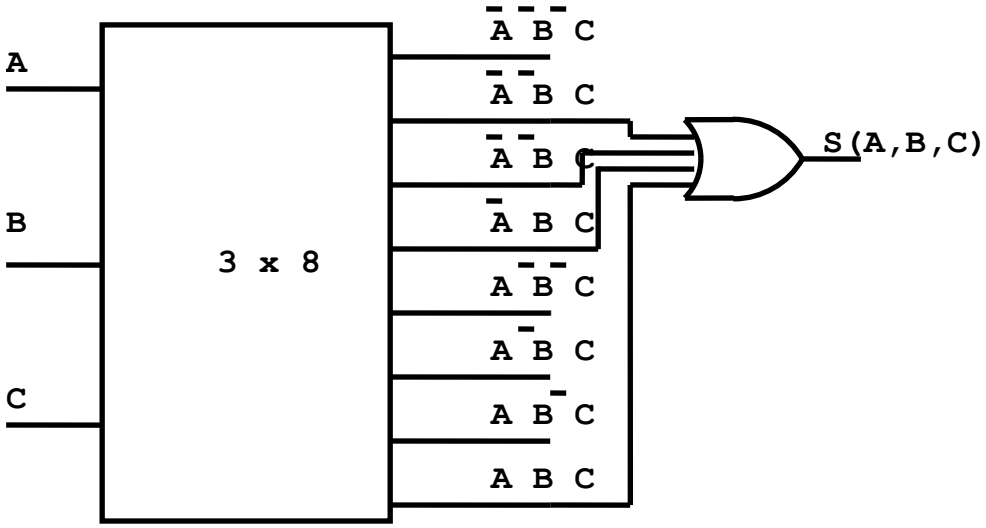
Και τέλος η συνάρτηση F3=Σ(7)=ABC



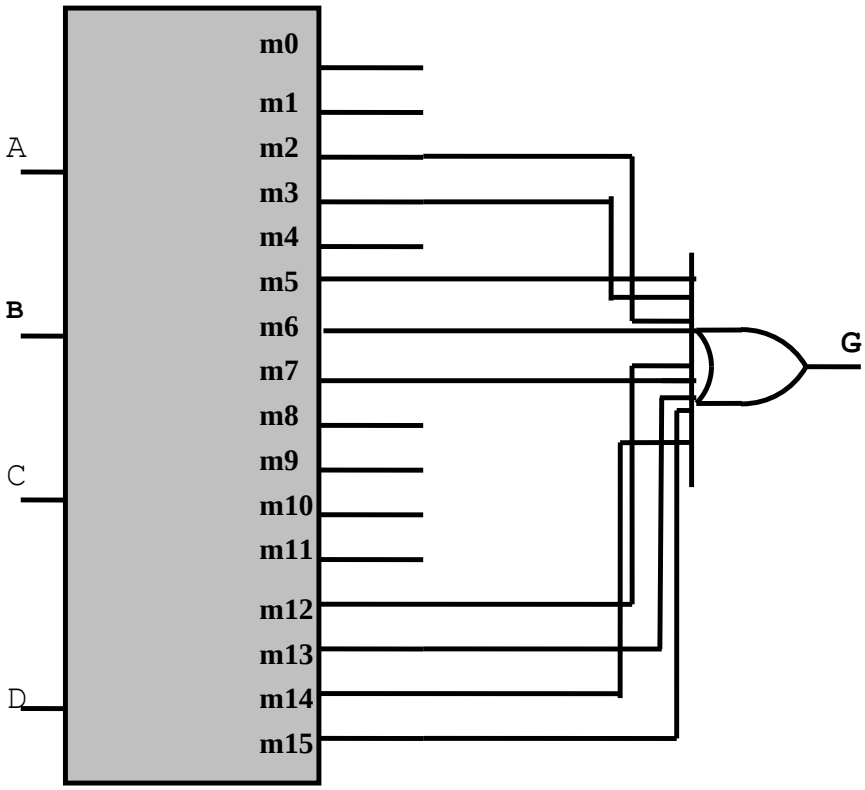
15. Σχεδιάστε με τους κατάλληλους αποκωδικοποιητές τις παρακάτω λογικές συναρτήσεις:

1. F(A,B,C)=Σ(1,2,3,7) και 2. G(A,B,C,D)=Σ(2,3,5,6,7,12,13,14,15)

Λύση: Για τη συνάρτηση F(x,y,z)=Σ(1,2,3,7) χρειαζόμαστε ένα αποκωδικοποιητή 3x8 και 1 πύλη OR 4 εισόδων.



Για τη συνάρτηση G(A,B,C,D)=Σ(2,3,5,6,7,12,13,14,15) χρειαζόμαστε ένα αποκωδικοποιητή 4x16 και 1 πύλη OR 9 εισόδων.



Κεφάλαιο 8. Ακολουθιακά Κυκλώματα

Τα ψηφιακά κυκλώματα που μελετήσαμε μέχρι τώρα ήταν συνδυαστικά έτσι όπως τα είχαμε ορίσει στην αρχή του μαθήματος. Δηλαδή είναι κυκλώματα τα οποία δίνουν σταθερή έξοδο στον χρόνο με την προϋπόθεση ότι δεν έχουμε αλλαγές στις εισόδους.

Τα ακολουθιακά κυκλώματα τα οποία αποτελούν την δεύτερη ενότητα του μαθήματος είναι κυκλώματα τα οποία χρονίζονται με ένα εξωτερικό ρολόι. Επιπλέον χρησιμοποιούν στοιχεία μνήμης τα οποία περιγράφουν την προηγούμενη κατάσταση του κυκλώματος σε προηγούμενους χρόνους οι οποίες όμως επηρεάζουν τις επόμενες αλλαγές αφού αυτές οι αλλαγές συναρτώνται άμεσα από αυτές τις καταστάσεις δηλαδή τις προγενέστερες. Ειδικότερα μπορούμε να πούμε ότι στα ακολουθιακά κυκλώματα, η επόμενη κατάσταση εξόδου δεν είναι αποκλειστικά συνάρτηση των μεταβλητών εισόδου τις οποίες εμείς καθορίζουμε, αλλά και της προηγούμενης κατάστασης, που είχε η έξοδός τους.

Ένα περίπλοκο ψηφιακό κύκλωμα με τη σειρά του και αυτό χρησιμοποιεί συνδυαστικά κυκλώματα όπως επίσης και ακολουθιακά τα οποία έχουν στοιχεία μνήμης και κάνουν το όλο κύκλωμα να ονομάζεται ακολουθιακό. Έχουμε δύο κατηγορίες ακολουθιακών κυκλωμάτων τα α) σύγχρονα και β) ασύγχρονα.

Τα σύγχρονα ακολουθιακά κυκλώματα είναι εκείνα στα οποία η εφαρμογή των παλμών με το ρολόι χρονισμού (clock) γίνεται σε όλα τα FLIP-FLOP και η εκτέλεση των διαφόρων λειτουργιών και η έξοδος του κυκλώματος γίνεται σε τακτά χρονικά διαστήματα που καθορίζονται από τη συχνότητα του ρολογιού.

Τα ασύγχρονα ακολουθιακά κυκλώματα είναι εκείνα στα οποία οι διάφορες λειτουργίες του κυκλώματος δεν εκτελούνται σε τακτά χρονικά διαστήματα από τη στιγμή εφαρμογής των παλμών εισόδου, αλλά η ταχύτητα εκτέλεσης εξαρτάται από αυτό καθαυτό το κύκλωμα. Στην περίπτωση των ασύγχρονων κυκλωμάτων δεν υπάρχει καθολική συνδεσμολογία όλων των FLIP-FLOP στο κεντρικό ρολόι. Πολλά FLIP-FLOP χρονίζονται από την έξοδο ενός άλλου FLIP-FLOP ή από την έξοδο κάποιου άλλου κυκλώματος.

Τα στοιχεία μνήμης έχουν ληφθεί από την έξοδο του ίδιου του κυκλώματος σε μία προηγούμενη χρονική στιγμή και παίζουν καθοριστικό λόγο στην εξέλιξη των καταστάσεων των εξόδων όπως φαίνεται στο παρακάτω σχηματικό BLOCK διάγραμμα. Μια τέτοια συνδεσμολογία ονομάζεται συνδεσμολογία με ανάδραση.

Ένα συνδυαστικό κύκλωμα και τα στοιχεία μνήμης συνθέτουν ένα σύγχρονο ακολουθιακό κύκλωμα

Λέμε ότι έχουμε ανάδραση όταν η έξοδος ενός κυκλώματος μέσω κάποιου άλλου κυκλώματος συνδέεται με την είσοδο. Αυτή η διαδικασία εκτελεί δειγματοληψία ενός ποσοστού του σήματος εξόδου το μεταφέρει στη είσοδο όπου μπορεί να προστεθεί ή να αφαιρεθεί με το αντίστοιχο σήμα της εισόδου εννοείται. Ένα ακολουθιακό κύκλωμα χρησιμοποιεί σήματα τα οποία επηρεάζουν τα στοιχεία μνήμης σε τακτά χρονικά διαστήματα. Τα σήματα αυτά που είναι επίπεδα τάσης δημιουργούνται από μία γεννήτρια " κύριο ρολόι master clock " και διανέμονται παντού στο σύστημα. Η είσοδος του ρολογιού ονομάζεται Cp (Clock pulses) ή απλά CLK (Clock).Με αυτό τον τρόπο συγχρονίζεται το κύκλωμα και οι μεταβολές των στοιχείων μνήμης όπως και των εξόδων του ακολουθιακού κυκλώματος γίνονται με τον ερχομό του παλμού του ρολογιού, ή του επιπέδου τάσης. Τα στοιχεία μνήμης λέγονται **Flip-Flop (FLIP-FLOP)** και μπορούν να αποθηκεύουν ένα bit πληροφορίας. Είναι τα στοιχειώδη κύτταρα αποθήκευσης πληροφορίας. Εκτός από την κατά περίπτωση αριθμό εισόδων, διαθέτουν δύο εξόδους. Η μία δίνει το αποθηκευμένο bit και η άλλη το συμπλήρωμα του. Η κατάσταση ενός FLIP-FLOP μπορεί να

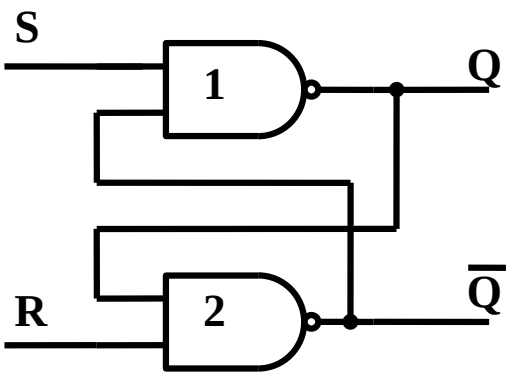
διατηρηθεί επ' αόριστο αρκεί να το τροφοδοτούμε σε ισχύ. Υπάρχουν πολλά είδη FLIP-FLOP ανάλογα με τον τρόπο εγγραφής και ανάγνωσης μίας πληροφορίας όπως θα δούμε παρακάτω. Τα FLIP-FLOP ονομάζονται επίσης και δισταθείς πολυδονητές, επειδή έχουν δύο σταθερές καταστάσεις. Οι δισταθείς πολυδονητές αποτελούν τις δομικές μονάδες και αποτελούν τα στοιχειώδη κύτταρα για πιο πολύπλοκα ψηφιακά κυκλώματα όπως καταχωρητές, μετρητές, κ.λ.π. όπως θα δούμε παρακάτω.

Βασικά Flip Flop

Τα Flip Flop είναι όπως είδαμε και πριν είναι κυκλώματα με δύο σταθερές καταστάσεις. Η εξέλιξη των εξόδων όταν αυτό απαιτηθεί είναι συνάρτηση των εισόδων τη δεδομένη χρονική στιγμή αλλά και από αυτή των εξόδων που υπήρχαν πριν από την διέγερση.

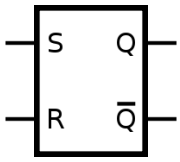
Flip Flop τύπου RS

Ο δισταθής πολυδονητής τύπου RS έχει δύο σύγχρονες εισόδους που ονομάζονται R και S. Είναι το απλούστερο ακολουθιακό κύκλωμα και υλοποιείται με δύο πύλες NAND όπως φαίνεται στο παρακάτω σχήμα. Το όνομα του πηγάζει από το R (RESET) δηλαδή μηδενισμός της κανονικής εξόδου Q=0 και S (SET) που θέτει την έξοδο Q στη κατάσταση Q=1. Η λειτουργία του περιγράφεται παρακάτω. Συμβολίζοντας την επόμενη κατάσταση με Q(t+1) θα δούμε παρακάτω τις μεταβολές της εξόδου και πως επηρεάζονται αυτές.



Βασικό κύκλωμα RS

Βλέπουμε ότι η συνδεσμολογία του Flip-Flop χρησιμοποιεί 2 πύλες NAND 2 εισόδων διασταυρωμένες. Η έξοδος της 1ης είναι είσοδος της 2ης και η έξοδος της 2ης είναι είσοδος της 1ης. Ας υποθέσουμε ότι αρχικά έχουμε την κατάσταση Ι Q=0 και $\overline{Q} = 0$.



Αν φέρουμε απότομα το R στο 0 η έξοδος δε θα αλλάξει και το FLIP-FLOP θα παραμείνει στην αρχική του κατάσταση. Αν φέρουμε το S στο 0 τότε η NAND 1 θα δώσει στην έξοδο Q=1 και κατά συνέπεια (αν R=1) το FLIP-FLOP πηγαίνει στη κατάσταση 1. Εφαρμόζοντας διαδοχικά παλμούς στο S και R πηγαίνουμε από τη μία κατάσταση στην άλλη.

Συνοψίζοντας την λειτουργία του θα έχουμε:

- Όταν S=0 και R=0, τότε η επόμενη κατάσταση (Q(t+1)) είναι ίδια με την προηγούμενη κατάσταση, δηλαδή Q(t).
- Όταν S=0 και R=1, τότε η επόμενη κατάσταση είναι Q(t+1)=0.
- Όταν S=1 και R=0, τότε η επόμενη κατάσταση είναι Q(t+1)=1.
- Όταν S=1 και R=1, τότε η επόμενη κατάσταση είναι απροσδιόριστη. Αυτή είναι μη χρησιμοποιούμενη κατάσταση και συμβολίζεται στον παρακάτω πίνακα αληθείας με Χ.

Πίνακας αληθείας του δισταθούς πολυδονητή τύπου RS

R	**S**	**Q(t+1)**
0	0	Q(t)
0	1	1
1	0	0
1	1	?

Οι ιδιότητες του δισταθούς αυτού βασικού κυκλώματος μπορούν να περιγραφούν και από τον παρακάτω πίνακα αλήθειας. Σε αυτή τη περίπτωση θεωρούμε ότι η έξοδος συναρτάται με την προηγούμενη της τιμή Q(t) με το R, και με το S. Θα έχουμε δηλαδή τη συνάρτηση $\mathbf{Q_{t+1}=Q_{t+1}(Q_t,R,S)}$ η οποία δίνει την έξοδο Q_{t+1} του Q μετά από την εφαρμογή συγκεκριμένων τιμών στις εισόδους S και R του FLIP-FLOP γνωρίζοντας επίσης την τιμή Q_t του Q πριν από την εφαρμογή αυτή των τιμών. Το Q_{t+1} είναι η κατάσταση της εξόδου του FLIP-FLOP την χρονική στιγμή t+1 ή μετά την έλευση του παλμού t+1 ενώ το Q_t είναι είναι η κατάσταση της εξόδου του FLIP-FLOP την χρονική στιγμή t ή μετά την έλευση του παλμού t που είναι βασικά ο προτελευταίος παλμός χρονισμού. Μπορούμε να δούμε παρακάτω τον πίνακα αλήθειας του FLIP-FLOP ο οποίος μας δίνει την έξοδο συναρτήσει της προηγούμενης τιμής και των εισόδων του FLIP-FLOP.

$\mathbf{Q_t}$	**S**	**R**	$\mathbf{Q_{t+1}}$
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	**?**
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	**?**

Αν R=S=0 $Q_{t+1}=Q_t$

Αν R=S=1 Q_{t+1} είναι απροσδιόριστο.

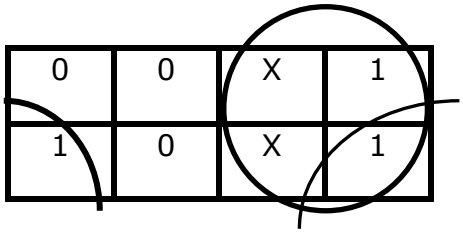
Αν R=0, S=1 το κύκλωμα οδηγείται στην κατάσταση $Q_{t+1}=1$

Αν R=1, S=0 το κύκλωμα οδηγείται στην κατάσταση $Q_{t+1}=0$

Βλέπουμε ότι για τη παραπάνω συνάρτηση $\mathbf{Q_{t+1}= Q_{t+1}(Q(t),S, R)}$ οι αδιάφοροι όροι της συνάρτησης είναι ο m3 και ο m7 ή d=d(3,7) και η κανονική της μορφή σαν άθροισμα γινομένων είναι $\mathbf{Q_{t+1}=\Sigma(2,4,6)}$ Ο πίνακας Karnaugh της συνάρτησης είναι ο παρακάτω.

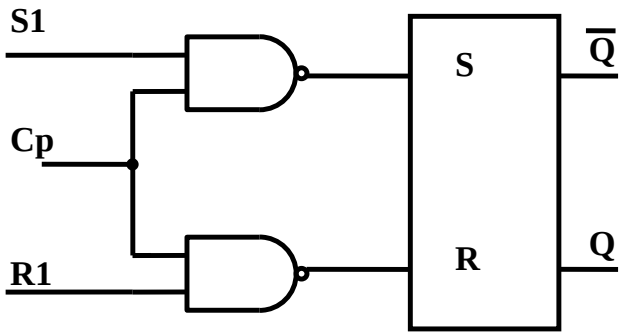
Βλέπουμε ότι τα τρία τετράγωνα με 1 συνδυάζονται 1 τετράδα με τους 2 αδιάφορους όρους και παίρνουμε **S** ενώ οι δύο άσσοι μας δίνουν $\bar{R}.Q(t)$.

Θα έχουμε λοιπόν τη παρακάτω χαρακτηριστική εξίσωση του FLIP-FLOP τύπου RS $Q(t+1) = S + \bar{R}.Q(t)$



Τα γράμματα R και S όπως είδαμε και προηγουμένως προέρχονται από την αγγλική και δηλώνουν διαδοχικά το R (reset) και S (set). Έτσι όταν S=1 η έξοδος μπαίνει στο 1 και όταν R=1 στο 0.

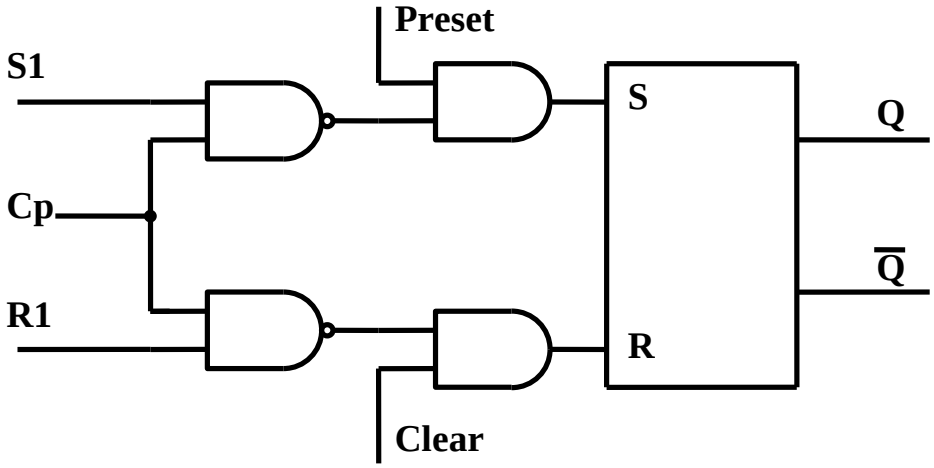
Μπορούμε να φτιάξουμε λίγο το προηγούμενο κύκλωμα αν του προσθέσουμε πύλες NAND όπως φαίνεται στο παρακάτω σχήμα.



Flip Flop RS με ρολόι

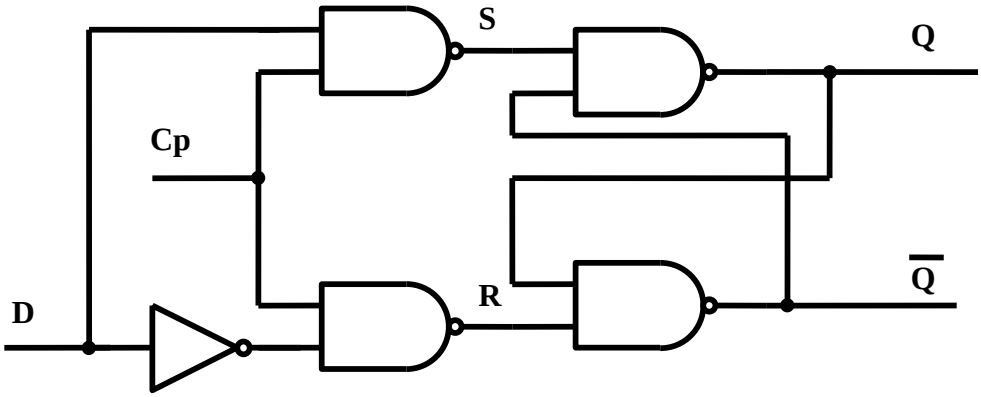
Αν Cp=0 οι έξοδοι R και S των δύο NAND είναι στο 1 και το δισταθές RS μπορεί να βρεθεί σε μία οιανδήποτε κατάσταση. Ας υποθέσουμε τώρα ότι R1=1 και S1=0 όταν το ρολόι περάσει από το 0 στο 1 το R γίνεται 0 και το S γίνεται 1 και κατά συνέπεια η έξοδος Q=0. Έτσι μετά από το πέρασμα του ρολογιού από το 0 στο 1 οι έξοδοι Q και Q' αντιγράφουν τις εισόδους S

και R. Το σήμα που εφαρμόζεται στο Cp είναι το σήμα του κεντρικού ρολογιού. Ονομάζουμε καμιά φορά αυτό τον τύπο Flip-Flop RSH. Συχνά προσθέτουμε στο παραπάνω κύκλωμα δύο συμπληρωματικές εισόδους οι οποίες μας επιτρέπουν να ελέγχουμε την αρχική κατάσταση του Flip-Flop. Τη μία είσοδο την ονομάζουμε "Clear" και την άλλη "Preset". Η πρώτη είσοδος θέτει την έξοδο του FLIP-FLOP στην κατάσταση Q=0 και η δεύτερη Q=1 με τον ερχομό του παλμού όταν το ρολόι είναι στο 0. Όπως διαμορφώνεται το νέο κύκλωμα με τις δύο συμπληρωματικές εισόδους φαίνεται στο παρακάτω σχήμα.



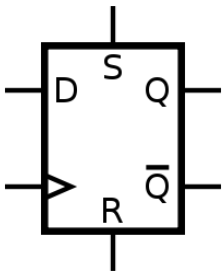
Flip-Flop τύπου RS με συμπληρωματικές εισόδους Clear και Preset.

Το FLIP-FLOP τύπου RS μπορεί να υλοποιηθεί πολύ εύκολα με πύλες NOR. Σε αυτή την περίπτωση η απαγορευμένη κατάσταση του FLIP-FLOP είναι η 11 αντί για την 00. Στο παρακάτω σχήμα βλέπουμε μία άλλη συνδεσμολογία του RS. Είναι το FLIP-FLOP τύπου D. Το παραπάνω FLIP-FLOP υλοποιείται αν βραχυκυκλώσουμε το R με το S και παρεμβάλουμε ένα αντιστροφέα πριν από την είσοδο R όπως φαίνεται στο παρακάτω κύκλωμα. Σε αυτή την περίπτωση έχουμε μόνο 2 δυνατές καταστάσεις. Έτσι αν D=0 τότε S=0 και R=1 οπότε το FLIP-FLOP γίνεται reset $Q(t+1)=0$ και αν D=1 τότε S=1 και R=0 οπότε το FLIP-FLOP γίνεται set δηλαδή με την έλευση του επόμενου παλμού του ρολογιού η μετάβαση της εξόδου θα είναι ίση με 1 $Q(t+1)=1$.



Συνδεσμολογία D του RS.

Αν Cp=0 το δισταθές RS μπορεί να βρεθεί σε μία οιανδήποτε κατάσταση δηλαδή η έξοδος δεν έχει καμία επαφή με την είσοδο. Αν Cp=1 τότε $\mathbf{S = \overline{D}\ και\ R = D}$ και κατά συνέπεια βλέπουμε ότι η έξοδος αντιγράφει την είσοδο D.



Εδώ βλέπουμε ένα Flip-Flop τύπου D με τη θέση σύνδεσης του ρολογιού, όπως επίσης και με τις άμεσες εισόδους Reset ή Clear και Preset ή Set.

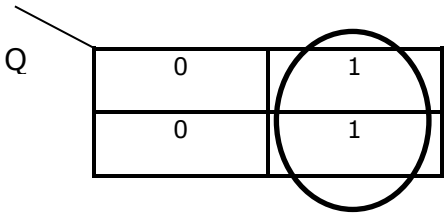
Το παραπάνω κύκλωμα είναι μία βασική κυψελίδα μνήμης γιατί κρατάει το αποτέλεσμα D στην έξοδο. Ο πίνακας αλήθειας του Flip Flop τύπου D " Data " φαίνεται παρακάτω.

Q(t)	D	Q(t+1)
0	0	0
0	1	1
1	0	0
1	1	1

Πίνακας αλήθειας του FLIP-FLOP τύπου D

Από τον παραπάνω πίνακα αλήθειας βλέπουμε ότι η συνάρτηση της εξόδου του FLIP-FLOP είναι $\mathbf{Q_{t+1} = Q_{t+1}(Q(t),D)}$ οι μεταβλητές είναι Q και

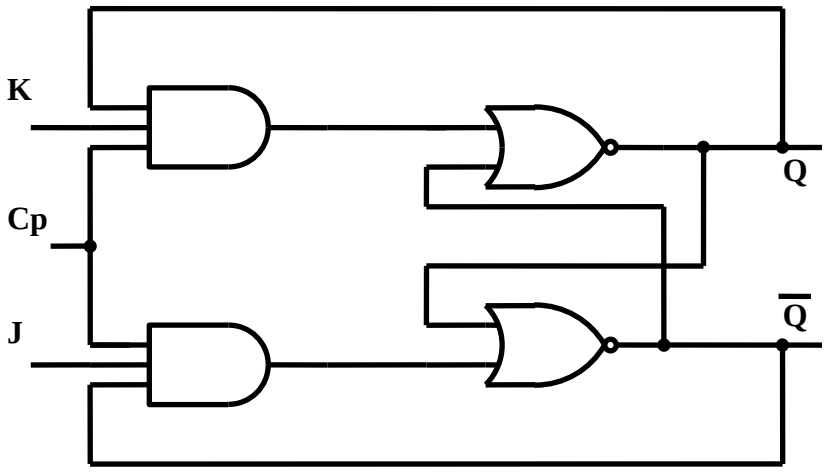
D.η κανονική της μορφή σαν άθροισμα ελαχιστόρων είναι **Q_{t+1}=Σ(1,3).** Ο πίνακας της συνάρτησης έχει τέσσερα τετράγωνα όπως:



Συνδυάζοντας του 2 άσσους προκύπτει η παρακάτω χαρακτηριστική εξίσωση για το FLIP-FLOP τύπου D.

$$Q(t+1) = D$$

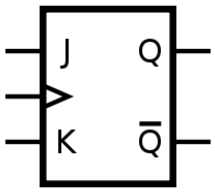
Είδαμε προηγουμένως ότι το FLIP-FLOP τύπου RS είχε μία απροσδιόριστη κατάσταση. Η απροσδιοριστία αυτή δεν υπάρχει πλέον με το Flip-Flop τύπου JK του οποίου το κύκλωμα φαίνεται στο παρακάτω σχήμα. Δηλαδή όταν οι είσοδοι J και K είναι αντίστοιχα J=1 και K=1 δεν υπάρχει απροσδιοριστία.



Flip-Flop τύπου JK με ρολόι

Το Flip-Flop τύπου JK άρει την απροσδιοριστία που υπήρχε για το Flip-Flop τύπου RS. Δηλαδή και για τον συνδυασμό R=S=1 δηλαδή αντίστοιχα J=K=1 το FLIP-FLOP δεν έχει απροσδιόριστη κατάσταση όπως θα δούμε και στον πίνακα αλήθειας που φαίνεται παρακάτω. Σε αυτή την περίπτωση η έξοδος του FLIP-FLOP συμπληρώνεται.

J	***K***	***Q(t+1)***
0	0	Q(t)
0	1	0
1	0	1
1	1	$\overline{Q(t)}$



Βλέπουμε ότι J=K=0 η κατάσταση του FLIP-FLOP παραμένει αμετάβλητη. Όταν J=0 και K=1 το FLIP-FLOP γίνεται Reset, όταν J=1 και K=0 το FLIP-FLOP γίνεται Set, ενώ όταν J=K=1 η έξοδος του FLIP-FLOP συμπληρώνεται δηλαδή αν το Q(t)=0 τότε με τον ερχομό του επόμενου παλμού θα έχουμε Q(t+1)=1, ενώ αν το Q(t)=1 τότε με τον ερχομό του επόμενου παλμού θα έχουμε Q(t+1)=0. Κατά συνέπεια ο χαρακτηριστικός πίνακας αλήθειας του JK και πιο συγκεκριμένα η κατάσταση της εξόδου Q(t+1) θα δίνεται από τον πίνακα αλήθειας που βλέπουμε παρακάτω. Είναι η συνάρτηση:

$$\mathbf{Q_{t+1} = Q_{t+1}(Q(t),J,K)}$$

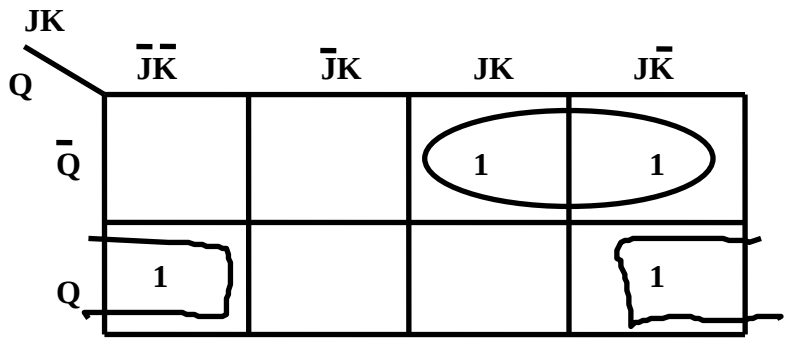
Και ο πίνακας αλήθειας παρακάτω μας δίνει τις τιμές για το Q_{t+1} για όλες τις διακριτές τιμές των J, K και Q_t.

Q(t)	**J**	**K**	**Q(t+1)**
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1

1	0	1	0
1	1	0	1
1	1	1	0

Πίνακας αλήθειας του Flip-Flop JK.

Από τον παραπάνω πίνακα μπορούμε να εξάγουμε την κανονική μορφή της συνάρτησης Q(t+1)=Σ(2,3,4,6) και στην συνέχεια να την απλοποιήσουμε όπως φαίνεται στον παρακάτω πίνακα Karnaugh. Βλέπουμε ότι έχουμε δύο συνδυασμούς των δύο τετραγώνων κατά συνέπεια θα πάρουμε μετά την απλοποίηση δύο όρους των δύο μεταβλητών όπως φαίνεται χαρακτηριστικά παρακάτω.

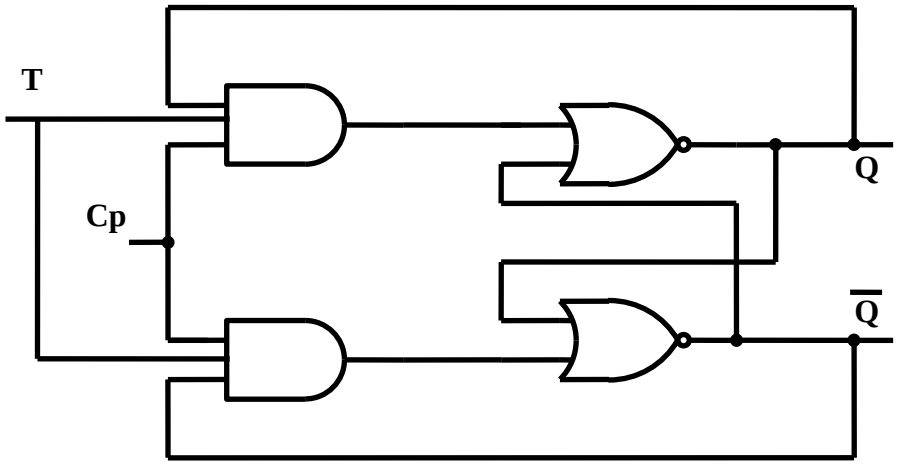


Πίνακας karnaugh του Flip-Flop JK

Από τον παραπάνω πίνακα Karnaugh βγαίνει ότι η χαρακτηριστική του εξίσωση της κανονικής εξόδου του Flip-Flop τύπου JK είναι η παρακάτω.

$$\mathbf{Q(t+1) = \bar{Q}J + Q\bar{K}}$$

Το Flip-Flop τύπου T είναι μία παραλλαγή του Flip-Flop τύπου JK και διαθέτει μία μόνο είσοδο. Μπορούμε εύκολα να το υλοποιήσουμε βραχυκυκλώνοντας το J με το K όπως φαίνεται στο παρακάτω σχήμα. Σε αυτή την περίπτωση έχουμε μόνο δύο επιλογές δηλαδή το T να είναι ίσο με 0 ή το T να είναι ίσο με 1. Αν το T=0 τότε J=K=0 και ή έξοδος $Q(t+1) = Q(t)$ δεν έχουμε αλλαγή ενώ αν T=1 τότε J=K=1 και η έξοδος συμπληρώνεται δηλαδή $Q(t+1) = \overline{Q(t)}$.

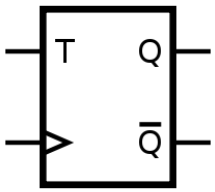


Flip-Flop τύπου T

Ο πίνακας καταστάσεων του T φαίνεται στο παρακάτω σχήμα όπως επίσης και η χαρακτηριστική του εξίσωση την οποία μπορούμε να πάρουμε από αυτή του JK αλλάζοντας τα J και K σε T. Θα πάρουμε τελικά την παρακάτω χαρακτηριστική εξίσωση του FLIP-FLOP τύπου T. Αν υλοποιήσουμε τον πίνακα Karnaugh θα δούμε ότι η συνάρτηση T δεν απλοποιείται όπως φαίνεται στον παρακάτω πίνακα Karnaugh όπου δεν υπάρχουν γειτονικά τετράγωνα και η συνάρτηση παραμένει ως είναι.

0	1
1	0

Δηλαδή θα έχουμε: $\mathbf{Q(t+1) = \overline{Q}T + Q\overline{T}}$

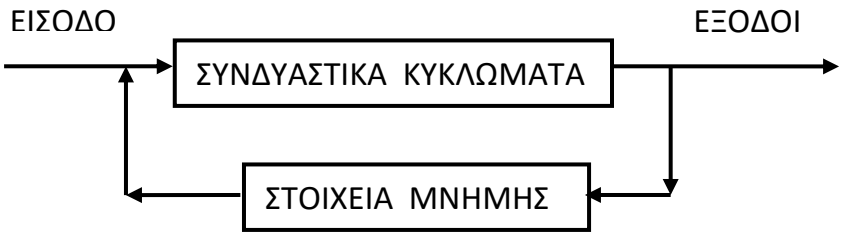


Q(t)	T	Q(t+1)
0	0	0
0	1	1
1	0	1
1	1	0

Πίνακας αλήθειας του Flip-Flop τύπου T

Η αλλαγή κατάστασης ενός Flip-Flop λέγεται πυροδότηση. Τα σύγχρονα Flip-Flops πυροδοτούνται από τους παλμούς του ρολογιού. Ένας τέτοιος παλμός ξεκινάει από την κατάσταση 0, στην συνέχεια πηγαίνει στην 1 όπου και παραμένει ένα ορισμένο χρόνο και στην συνέχεια ξαναγυρίζει στην αρχική κατάσταση 0. Ο παλμός κατά την μετάβαση στην κατάσταση 1 παρουσιάζει ένα ανοδικό μέτωπο ενώ κατά την μετάβαση από το1 στο 0 ένα καθοδικό.

Η κατάσταση λοιπόν ενός Flip-Flop αλλάζει με μια στιγμιαία αλλαγή ενός σήματος εισόδου. Αυτή η στιγμιαία αλλαγή λέγεται «πυροδότηση» ("trigerring") - λέμε ότι πυροδοτεί το Flip-Flop. Τα Flip-Flop με ρολόι πυροδοτούνται από τους παλμούς του ρολογιού. Ας θυμηθούμε το Block διάγραμμα που φαίνεται παρακάτω και χαρακτηρίζει τα ακολουθιακά κυκλώματα. Ο βρόχος ανάδρασης μπορεί να προκαλέσει αστάθεια εάν οι έξοδοι των στοιχείων μνήμης των Flip-Flop, που πηγαίνουν στο συνδυαστικό κύκλωμα, αλλάζουν την ώρα που οι είσοδοι των Flip-Flops, που προέρχονται από το συνδυαστικό κύκλωμα, δειγματοληπτούνται με τον παλμό του ρολογιού.

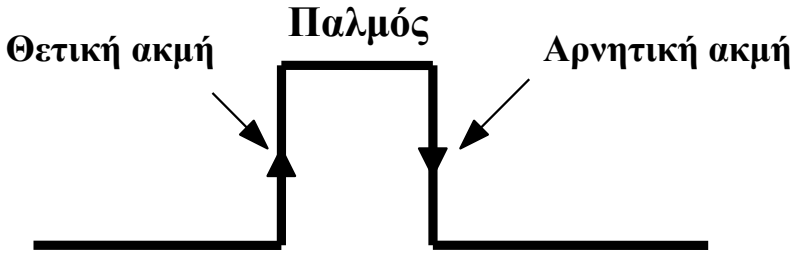


Ένας τρόπος να λύσουμε το πρόβλημα χρονισμού της ανάδρασης είναι να κάνουμε το Flip-Flop ευαίσθητο στις μεταβάσεις των παλμών από το ένα επίπεδο στο άλλο, αντί στη διάρκεια των παλμών. Οι μεταβάσεις των παλμών ρολογιού μπορεί να είναι θετικές ή αρνητικές.

Μπορούμε χρησιμοποιώντας μια χωρητική ζεύξη να ρυθμίσουμε το Flip-Flop έτσι ώστε να αντιδρά μόνο στις ακμές, των παλμών. Αυτό σημαίνει ότι αν τοποθετήσουμε ένα κύκλωμα RC (αντίσταση-πυκνωτής) στην είσοδο του ρολογιού, το κύκλωμα αυτό θα δημιουργεί ένα «σπινθήρα» ("spike") όποτε αλλάζει το σήμα εισόδου, με τη θετική ακμή θα δίνει ένα θετικό σπινθήρα και με την αρνητική ακμή θα δίνει αρνητικό σπινθήρα. Εάν τώρα

σχεδιάσουμε το Flip-Flop έτσι ώστε να αγνοεί τον ένα σπινθήρα και να πυροδοτείται από τον άλλο, τότε θα έχουμε εξασφαλίσει τύχει την «ακμοπυροδότηση» του Flip-Flop.

Βλέπουμε λοιπόν ότι η πυροδότηση εξαλείφει το πρόβλημα της αστάθειας ακολουθώντας δύο διαφορετικές μεθοδολογίες. Κατά πρώτον κατά τη σχεδίαση τοποθετείται ένα δεύτερο αποθηκευτικό στοιχείο το οποίο λειτουργεί συμπληρωματικά με το 1ο και κόβει την ανάδραση που δημιουργεί το κύκλωμα. Κατά δεύτερον επιτρέπει την αποθήκευση να γίνεται στιγμιαία (κατά την άνοδο ή κάθοδο του ρολογιού) ώστε να μην υπάρχει χρόνος ενεργοποίησης της ανάδρασης και πρόκλησης έτσι αστάθειας στο κύκλωμα.



Παλμός του ρολογιού, θετική και αρνητική ακμή

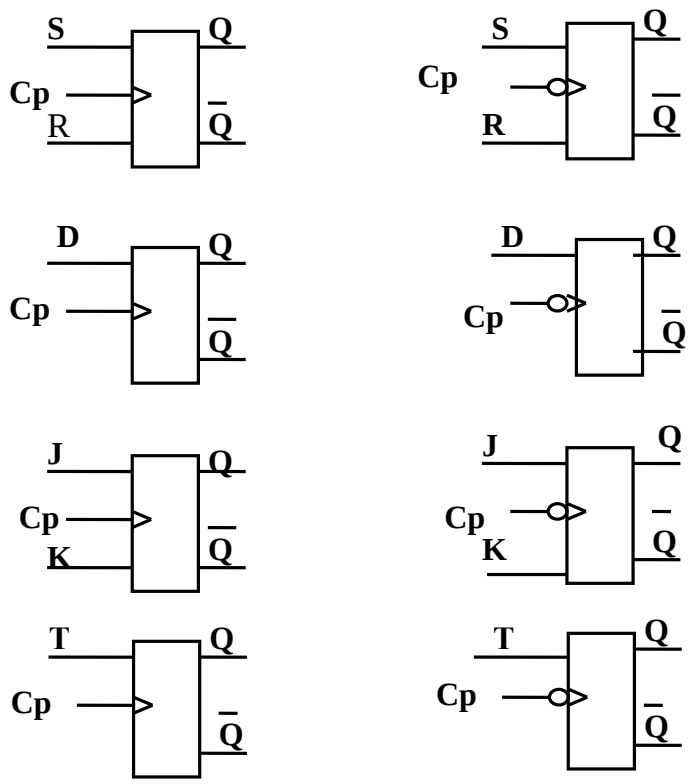
Αν το Flip-Flop αλλάζει κατάσταση όταν ανεβαίνει ο παλμός από το μηδέν στο ένα (0 → 1) τότε θα λέμε ότι το Flip-Flop είναι θετικής ακμής και όταν αλλάζει κατάσταση αντίστροφα στην μετάβαση από το ένα στο μηδέν (1 → 0) θα λέμε ότι το FF είναι αρνητικής ακμής.

Αναφέρεται επίσης σαν **PT** (από positive triggering) θετικός σκανδαλισμός του FF όταν έχουμε FF θετικής ακμής και σαν **NT** (από negative triggering) αρνητικού σκανδαλισμού σαν αρνητικής ακμής. Με αυτόν τον τρόπο σε κάθε παλμό ένα Flip-Flop αλλάζει κατάσταση μόνο μία φορά. Στο παρακάτω σχήμα βλέπουμε πώς συμβολίζουμε τα τέσσερα Flip-Flop που παρουσιάσαμε προηγουμένως.

Φαίνεται επίσης αν το αντίστοιχο FLIP-FLOP είναι θετικής ακμής ή αρνητικής από την αντιστροφή στο σημείο σύνδεσης του Cp ή του CLK.

Τα FF αρνητικής ακμής έχουν ένα μικρό κύκλο στο σημείο που συνδέεται το κεντρικό ρολόι, δηλαδή πριν το μικρό τριγωνάκι όπως μπορούμε να δούμε στο παρακάτω διάγραμμα όπου παρουσιάζονται τα Flip-Flop θετικής και αρνητικής ακμής.

Θετικής ακμής **Αρνητικής ακμής**



Το τριγωνάκι συμβολίζει το μέρος που πρέπει να συνδεθεί το ρολόι και ο κύκλος πριν το τριγωνάκι δείχνει ότι το **Flip-Flop** είναι αρνητικής ακμής.

Άμεσες είσοδοι. Ένα ακολουθιακό κύκλωμα μπορεί να περιέχει ένα μεγάλο αριθμό από Flip-Flop όπως επίσης πύλες η κυκλώματα MSI κλπ. Με τη χρήση των αμέσων εισόδων δίνεται η δυνατότητα στον χρήστη να μηδενίσει τα Flip-Flop και να μεταφέρει το όλο σύστημα σε μία αρχική μηδενική κατάσταση, η σε κάποια άλλη πολύ συγκεκριμένη. Η δυνατότητα των αμέσων εισόδων των Flip-Flop έχει ήδη περιγραφεί προηγουμένως. Οι άμεσες είσοδοι μπορούν να ενεργοποιηθούν ανά πάσα στιγμή και η λειτουργία και η ενεργοποίηση στους δεν σχετίζεται σε καμία περίπτωση με το κεντρικό ρολόι.

Πίνακες διέγερσης και πίνακες μετάβασης των FLIP-FLOP

Για να μπορέσουμε να αναλύσουμε τη λειτουργία ενός τυχαίου ακολουθιακού κυκλώματος πρέπει να γνωρίζουμε με ακρίβεια τα επί μέρους κυκλώματα που το απαρτίζουν όπως επίσης και του πίνακες διέγερσης των Flip-Flop που χρησιμοποιούνται στην συγκεκριμένη κατασκευή. Η ανάλυση ξεκινάει τη χρονική στιγμή t και προχωράει στον

χρόνο. Παρακάτω θα δούμε αναλυτικά τους πίνακες διέγερσης των τεσσάρων Flip-Flop με δύο διαφορετικούς τρόπους. Με δεδομένο το συνδυασμό εισόδων και γνωρίζοντας την έξοδο τη χρονική στιγμή t προσπαθούμε να βρούμε την έξοδο τη χρονική στιγμή t+1. Δηλαδή για δεδομένες τιμές στην εξόδου, και εισόδων των Flip-FlopFlip-Flop πως με τον ερχομό του νέου παλμού αυτά αλλάζουν.

Το αποτέλεσμα φαίνεται στον παρακάτω πίνακα. Ο δεύτερος τρόπος ορισμού του πίνακα διέγερσης είναι με ποιους συνδυασμούς των εισόδων θα έχω ένα επιθυμητό αποτέλεσμα στην έξοδο.

S	R	Q(t+1)
0	0	Q(t)
0	1	0
1	0	1
1	1	?

J	K	Q(t+1)
0	0	Q(t)
0	1	0
1	0	1
1	1	$\overline{Q}(t)$

D	Q(t+1)
0	0
1	1

T	Q(t+1)
0	Q(t)
1	$\overline{Q}(t)$

Πίνακες διέγερσης των τεσσάρων βασικών Flip__Flop

Ο άλλος πίνακας διέγερσης βρίσκεται πολύ εύκολα. Είπαμε ότι σε αυτή την περίπτωση γνωρίζουμε Q(t) και Q(t+1) και προσπαθούμε να βρούμε τις εισόδους των Flip-Flop για να το επιτύχουμε. Ας υποθέσουμε ότι έχουμε ένα Flip-Flop τύπου JK και θέλουμε Q(t)=0 και Q(t+1)=0. Από τον προηγούμενο χαρακτηριστικό πίνακα του JK βλέπω ότι αν J=K=0 τότε θα πάρω στην έξοδο Q(t)=Q(t+1), επίσης αν έχω J=0 και K=1 τότε θα έχω στην έξοδο του Flip-Flop Q(t+1)=0. Από τα παραπάνω βλέπω ότι αν είμαι στην κατάσταση 0 στην έξοδο και θέλω να παραμείνω και στον επόμενο παλμό του ρολογιού θα πρέπει υποχρεωτικά να έχω J=0 και το K=0 η K=1. Δηλαδή το K θα είναι αδιάφορος όρος αφού δεν αλλάζει η μετάβαση του FLIP-FLOP είτε το K=0 είτε το K=1 αρκεί το J=0.

Με την ίδια λογική και χρησιμοποιώντας τους πίνακες που δώσαμε προηγουμένως οι μπορούμε εύκολα να φτιάξουμε νέους πίνακες τους οποίους ονομάζουμε πίνακες μετάβασης των Flip-Flop τους οποίους και θα χρησιμοποιήσω πολύ πιο συχνά όπως θα δούμε παρακάτω. Οι πίνακες μετάβασης μας δείχνουν πως θα πρέπει να συνδέσω τις εισόδους των FLIP-FLOP έτσι ώστε η μετάβαση της εξόδου του FLIP-FLOP να γίνει από την

κατάσταση Q(t) με τον ερχομό του επόμενου παλμού στην επιθυμητή κατάσταση Q(t+1).

Q(t)	Q(t+1)	J	K	S	R	D	T
0	0	0	X	0	X	0	0
0	1	1	X	1	0	1	1
1	0	X	1	0	1	0	1
1	1	X	0	X	0	1	0

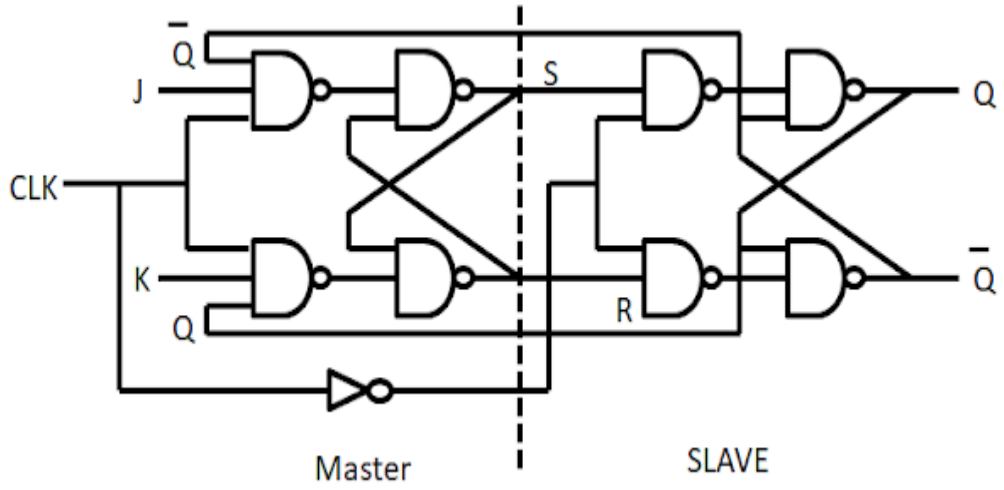
Πίνακες διέγερσης των τεσσάρων βασικών Flip-Flop.

Ο παραπάνω πίνακας μετάβασης όλων των FLIP-FLOP είναι πάρα πολύ σημαντικός και χρησιμοποιείται συνέχεια στη σχεδίαση των κυκλωμάτων που εμπλέκουν FLIP-FLOP.

Flip-Flop αφέντη σκλάβου ή Master slave Flip-Flop

Το FLIP-FLOP αφέντη σκλάβου είναι ένα διπλό FLIP-FLOP, όπου με τον πρώτο παλμό του ρολογιού αποθηκεύεται η πληροφορία στο FLIP-FLOP με ρόλο master, ενώ με το δεύτερο παλμό μεταφέρεται στο slave η πληροφορία που είχε αποθηκευτεί στο master. Σε όλα τα FLIP-FLOP που εξετάσαμε ως τώρα θα πρέπει ο χρόνος ton, που χρειάζεται ο παλμός ρολογιού να μεταβεί σε λογικό 1, να είναι μικρότερος από το συνολικό χρόνο καθυστέρησης διάδοσης του σήματος από την είσοδο ως την έξοδο του κυκλώματος tpd (tpd pulse delay σημαίνει χρόνος καθυστέρησης του παλμού) ώστε οι νέες καταστάσεις των εξόδων, όταν εμφανισθούν, να μην επανατροφοδοτούν στις πύλες εισόδου όσο διαρκεί ο παλμός ρολογιού. Σε αντίθετη περίπτωση το FLIP-FLOP θα περιπέσει σε αστάθεια αλλάζοντας διαρκώς τις καταστάσεις των εξόδων του έως ότου η είσοδος ρολογιού γίνει 0. Επίσης θα πρέπει η περίοδος του παλμού ρολογιού T, να είναι τέτοια ώστε ο επόμενος παλμός ρολογιού να εμφανισθεί αφού σταθεροποιηθεί η νέα κατάσταση των εξόδων. Έτσι, για σωστή λειτουργία, πρέπει να ισχύει: ton < tpd < T όπου T είναι η περίοδος του κεντρικού ρολογιού. Αν δεν ισχύει η παραπάνω σχέση τότε, για παράδειγμα στο J-K FLIP-FLOP, όταν J=1 και K=1 και οι παλμοί του ρολογιού (Cp=1) έχουν μεγάλη χρονική διάρκεια, τότε η κατάσταση του FLIP-FLOP αφού

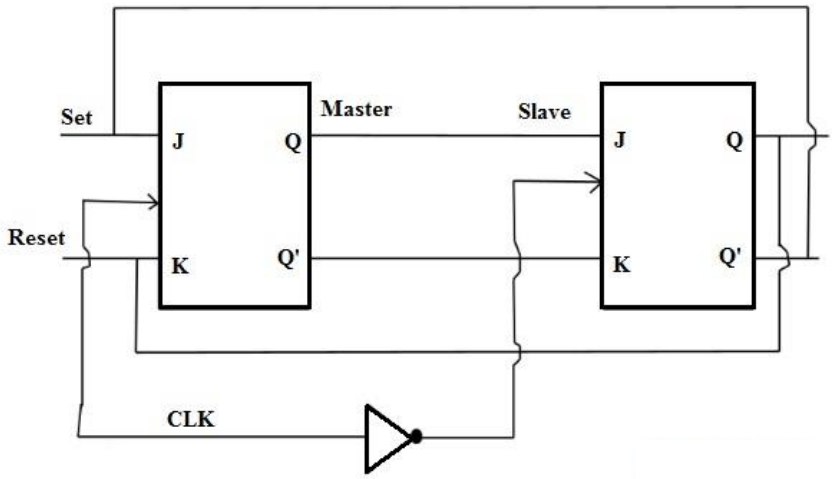
αντιστραφεί μία φορά, αντιστρέφεται συνεχώς σε όλη τη διάρκεια του παλμού του ρολογιού (δηλαδή μέχρι να γίνει Cp=0). Γενικά σε μια τέτοια περίπτωση δεν ακολουθείται ο πίνακας αλήθειας και η έξοδος του FLIP-FLOP είναι απροσδιόριστη.Η συνθήκη ton < tpd δεν είναι πάντα εύκολο να ικανοποιηθεί και αυτό διότι ο χρόνος tpd είναι πολύ μικρός όπως ήδη έχουμε δει. Για να αντιμετωπισθεί το πρόβλημα επινοήθηκαν τα Master-Slave FLIP-FLOP. Στο παρακάτω σχήμα φαίνεται το κύκλωμα ενός Master-Slave J-K FLIP-FLOP.



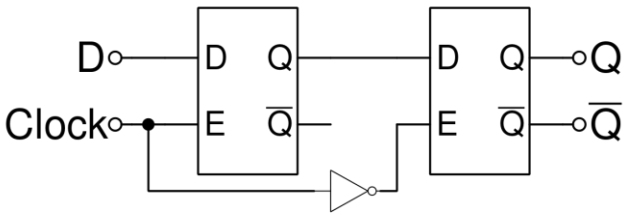
Βλέπουμε στη συνδεσμολογία του master-slave έχουμε τέσσερεις πύλες NAND για το κύκλωμα του αφέντη και τέσσερεις για το κύκλωμα του σκλάβου. Βλέπουμε επίσης τον αντιστροφέα που υπάρχει για το ρολόι του σκλάβου. Με την άνοδο του παλμού του ρολογιού ο αφέντης βλέπει την θετική ακμή ενώ ο σκλάβος την αρνητική και αντίστροφα.Βλέπουμε επίσης ότι η είσοδος J του αφέντη καθορίζεται από την έξοδο $\overline{Q}$ του σκλάβου ενώ η είσοδος S του αφέντη από την κανονική έξοδο Q του σκλάβου.

Όπως φαίνεται στο παρακάτω σχήμα το πρώτο FLIP-FLOP που ονομάζεται master ενεργοποιείται με το παλμό ρολογιού σε λογικό 1 ενώ το δεύτερο FLIP-FLOP που ονομάζεται slave ενεργοποιείται με το παλμό ρολογιού σε λογικό 0. Έτσι η αλλαγή κατάστασης εμφανίζεται στις εξόδους του slave αφού ο παλμός ρολογιού γίνει λογικό 0, και οι νέες καταστάσεις επανατροφοδοτούνται στο master FLIP-FLOP όταν σε αυτό ο παλμός ρολογιού είναι ήδη σε λογικό 0 με αποτέλεσμα να μην έχουμε νέες

μεταβολές στις εξόδους του master, και επομένως και του slave, που είναι και οι έξοδοι όλου του Master-Slave FLIP-FLOP.



Παρακάτω βλέπουμε ένα FLIP-FLOP αφέντη σκλάβου με FLIP-FLOP τύπου D. Βλέπουμε ότι η κανονική έξοδος του αφέντη είναι είσοδος για τον σκλάβο. Βλέπουμε επίσης τον αντιστροφέα ανάμεσα στο ρολόι του σκλάβου σε σχέση με αυτό του αφέντη. Η ενεργοποίηση και η λειτουργία του είναι όπως και προηγουμένως ανάλογη με αυτή του JK.



Κεφάλαιο 9. Μετρητές

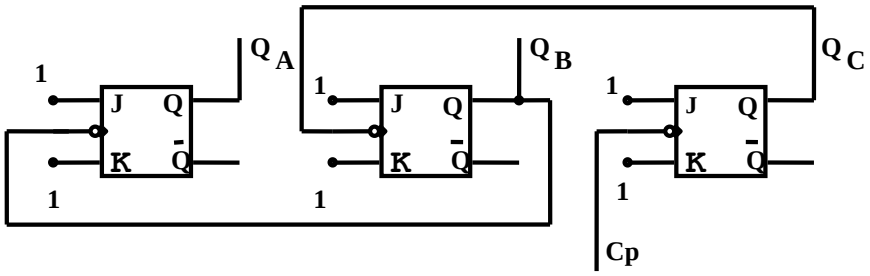
Ονομάζουμε μετρητή ένα ακολουθιακό κύκλωμα το οποίο χρονίζεται από ένα ρολόι και περνάει διαδοχικά από μία συγκεκριμένη ακολουθία καταστάσεων. Αυτοί οι παλμοί λέγονται παλμοί μέτρησης. Η μέτρηση μπορεί να είναι η δυαδική σειρά μέτρησης η οιανδήποτε άλλη σειρά μέτρησης όπως θα δούμε παρακάτω σε συγκεκριμένα παραδείγματα. Έτσι ένας μετρητής των n bit αποτελείται από n Flip-Flop έχει 2^n καταστάσεις και μπορεί να μετράει από το 0 μέχρι το 2^{n-1}. Για να σχεδιάσουμε ένα μετρητή πρέπει να ξέρουμε με ακρίβεια την ακολουθία καταστάσεων του.

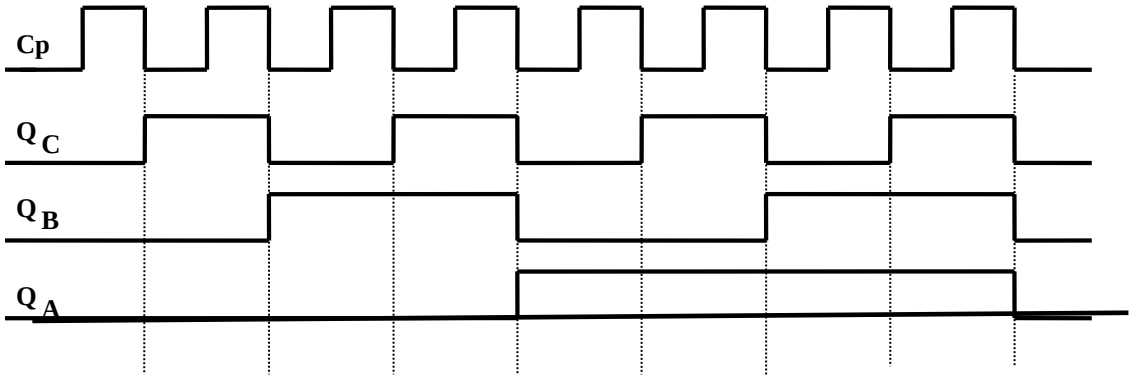
Πρέπει επίσης να ξέρουμε τι είδους Flip-Flop θα χρησιμοποιήσουμε και τέλος αν το ζητούμενο κύκλωμα θα είναι σύγχρονο η ασύγχρονο. Και όπως θα δούμε και παρακάτω σύγχρονο κύκλωμα λέγεται το ακολουθιακό κύκλωμα όπου όλα τα Flip-Flop είναι συνδεδεμένα στο κεντρικό ρολόι ενώ ασύγχρονο λέμε ένα ακολουθιακό κύκλωμα όπου ένα τουλάχιστον ή περισσότερα Flip-Flop είναι συνδεδεμένα στο κεντρικό ρολόι και τα υπόλοιπα χρονίζονται από τις εξόδους των άλλων απευθείας η μέσω κάποιων απλών ή περίπλοκων κυκλωμάτων.

Ασύγχρονοι μετρητές

Οι μετρητές ριπής είναι ασύγχρονα ακολουθιακά κυκλώματα. Αυτό σημαίνει ότι δεν είναι όλα τα Flip-Flop συνδεδεμένα με το κεντρικό ρολόι. Στην συγκεκριμένη περίπτωση ένα μόνο Flip-Flop συνδέεται με το κεντρικό ρολόι και τα ρολόγια των υπόλοιπων Flip-Flop χρονίζονται από εξόδους των άλλων η μετά από παρεμβολή πυλών. Ένα παράδειγμα φαίνεται στο παρακάτω σχήμα. Είναι ένας τρίμπιτος μετρητής ριπής σχηματιζόμενος με τρία Flip-Flop τύπου JK αρνητικής ακμής. Παρατηρούμε ότι και στα τρία FLIP-FLOP οι είσοδοι J και K είναι συνδεδεμένες με το 1, δηλαδή είναι στο $\mathbf{V_{cc}}$. Παρατηρούμε επίσης ότι και τα 3 FLIP-FLOP έχουν ένα μικρό κύκλο στο μικρό τρίγωνο που συνδέεται το ρολόι. Σε αυτή την περίπτωση και τα 3 FLIP-FLOP είναι αρνητικής ακμής δηλαδή με την πτώση του παλμού αλλάζουν κατάσταση δηλαδή η έξοδος τους θα συμπληρώνεται. Βλέπουμε ότι το τελευταίο FLIP-FLOP του οποίου η έξοδος $\mathbf{Q_C}$ αποδίδει το **LSB** του

μετρητή είναι συνδεδεμένο με το κεντρικό ρολόι. Αυτό σημαίνει ότι σε κάθε παλμό ρολογιού το 3ο FLIP-FLOP θα αλλάζει κατάσταση, το 2ο FLIP-FLOP θα αλλάζει κατάσταση κάθε 2 διαδοχικούς παλμούς και το τρίτο FLIP-FLOP θα αλλάζει κατάσταση κάθε 4 παλμούς. Αν υποθέσουμε ότι έχουμε μηδενίσει τις εξόδους των τριών FLIP-FLOP $Q_A=Q_B=Q_C=0$ πριν από την εφαρμογή του ρολογιού, τότε το διάγραμμα καταστάσεων του κυκλώματος των 3 FLIP-FLOP και για 8 παλμούς του ρολογιού φαίνεται παρακάτω.





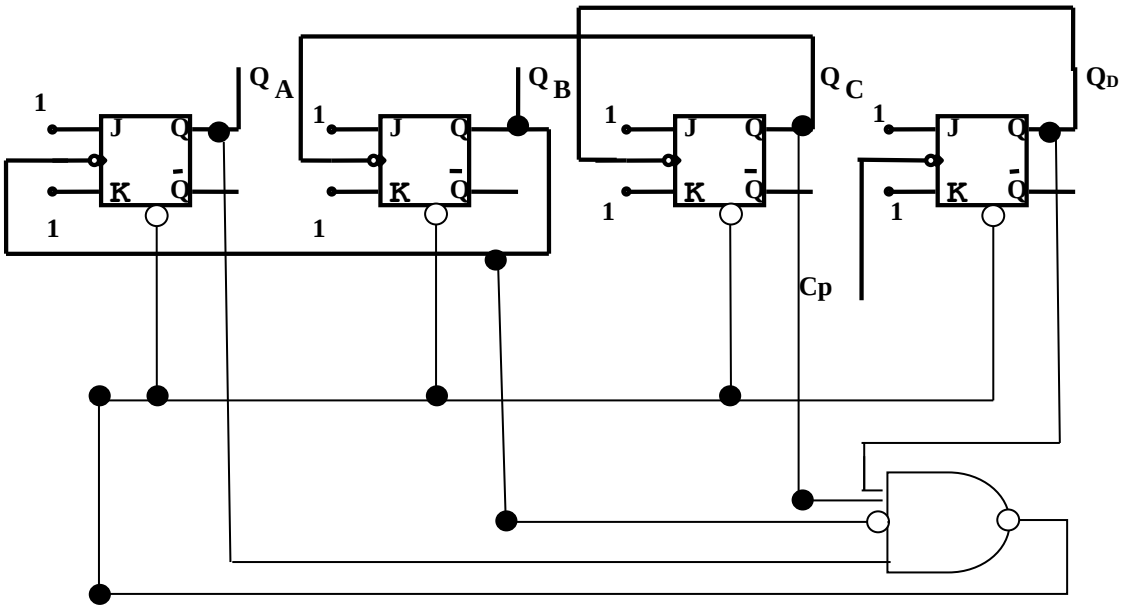
000 t_1 001 t_2 010 t_3 011 t_4 100 t_5 101 t_6 110 t_7 111 t_8 000

Τρίμπιτος μετρητής ριπής

Βλέπουμε ότι οι χαρακτηριστικοί χρόνοι είναι οι t_1 , t_2 , t_3 , t_4 , t_5 , t_6 , t_7 και t_8 με το αρνητικό μέτωπο του παλμού, είναι δηλαδή οι χρόνοι που μπορούν να αλλάξουν κατάσταση αν είναι επιβεβλημένο. Βλέπουμε ότι οι καταστάσεις που διαγράφουν οι έξοδοι Q_AQ_B Q_C διαδοχικά και για τους 8 παλμούς είναι 000, 001, 010, 011, 100, 101, 110, 111 και ξανά 000 δηλαδή από την αρχή. Είναι ένα κύκλωμα που μετράει από το 0 μέχρι το 7 και ξανά από την αρχή. Με το κύκλωμα του μετρητή ριπής όπως παρουσιάσαμε αναλυτικά προηγουμένως με τις μεταβολές των εξόδων των 3 FLIP-FLOP όπως φαίνεται στο διάγραμμα χρονισμού δηλαδή μεταβολής

των εξόδων των τριών Flip-Flop σαν συνάρτηση του χρόνου παρατηρούμε ότι σε κάθε έξοδο η συχνότητα του αρχικού ρολογιού διαιρείται ανάλογα με τη θέση που κατέχει μέσα στο κύκλωμα. Έτσι η συχνότητα του ρολογιού έχει διαιρεθεί διά του δύο στην έξοδο του πρώτου Flip-Flop διά του τέσσερα στην έξοδο του δεύτερου και διά του οκτώ στην έξοδο του τρίτου Flip-Flop. Γενικά αν έχουμε ένα μετρητή ριπής με **n** Flip-Flop μετρητή η έξοδος του τελευταίου θα διαιρεί διά 2^n την αρχική συχνότητα του ρολογιού. Το παραπάνω κύκλωμα έχει οκτώ καταστάσεις και θα λέμε ότι είναι ένας μετρητής **mode 8**. Μετράει από το 0 μέχρι το 2^3-1, και τέλος ότι διαιρεί την αρχική συχνότητα του ρολογιού διά οκτώ.

Στο παρακάτω κύκλωμα φαίνεται ένας τετράμπιτος μετρητής ριπής. Τα 4 FLIP-FLOP είναι συνδεδεμένα όπως και προηγουμένως. Ο τετράμπιτος μετρητής ριπής σχηματιζόμενος με τέσσερα Flip-Flop τύπου JK αρνητικής ακμής με τις εισόδους J και K να είναι συνδεδεμένες με το 1, δηλαδή είναι στο $\mathbf{V_{CC}}$. Αν οδηγήσουμε τις εξόδους $\mathbf{Q_A Q_B Q_C Q_D}$ σε ένα τετράμπιτο καταχωρητή θα παρατηρήσουμε την παρακάτω ακολουθία μέτρησης με την προϋπόθεση ότι έχουμε μηδενίσει τις εξόδους των 4 FLIP-FLOP. Οι καταστάσεις λοιπόν θα είναι 0000, 0001, 0010, 0011, 0100, 0101, 0111, 1000, 1001, 1010, 1011, 1100, 1101, 1110, 1111 και ξανά από την αρχή , δηλαδή 0000.

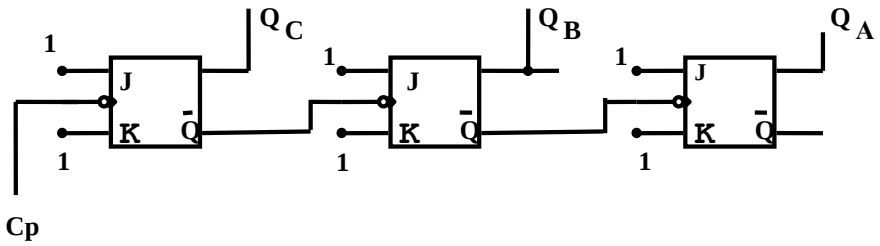


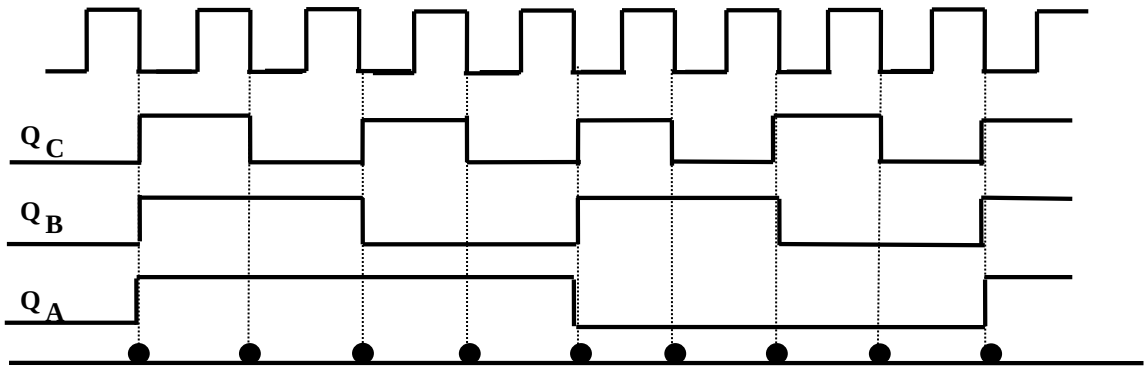
Ας υποθέσουμε ότι θέλουμε ο παραπάνω μετρητής να σταματάει στο 12, δηλαδή να έχει 13 καταστάσεις από το 0000 μέχρι και το 1100. Σε αυτή την περίπτωση ενεργοποιούμε την άμεση είσοδο reset και των 4 FLIP-FLOP την κατάλληλη στιγμή. Είναι σημαντικό να καταλάβουμε πότε γίνεται αυτή η ενέργεια. Ας θυμηθούμε όμως πότε μια πύλη NAND δίνει 1 στην έξοδο της. Για παράδειγμα μια πύλη NAND 4 εισόδων θα μηδενίσει την έξοδο της όταν όλες οι είσοδοι είναι 1. Μια τέτοια πύλη χρειαζόμαστε για το συγκεκριμένο κύκλωμα. Δηλαδή μια πύλη NAND 4 εισόδων όπου έχουμε ένα αντιστροφέα στην 3 είσοδο όπως φαίνεται στο κύκλωμα. Εμείς θέλουμε να μηδενίσουν όλα τα FLIP-FLOP μετά το 12. Αυτό γίνεται μόλις περάσει στο 13, δηλαδή στο 1101, και γίνεται ακαριαία. Βλέπουμε ότι αντιστρέφοντας την3^{η} είσοδο της NAND, η πύλη βλέπει αντί για το 1101 το 1111, οπότε μηδενίζεται η έξοδος της. Αυτό το μηδέν μεταφέρεται και στις 4 άμεσες εισόδους των FLIP-FLOP τα οποία επειδή έχουν ένα μικρό κύκλο στην άμεση είσοδο τους είναι Active Low, δηλαδή μηδενίζονταν όταν οι άμεσες είσοδοι γίνουν 0 πράγμα που συμβαίνει στο κύκλωμα μας και ο μετρητής μετά το 12 ξεκινάει από την αρχή. Ο χρόνος παραμονής στην κατάσταση 13 είναι ελάχιστος και δεν γίνεται αντιληπτός. Αν τα FLIP-FLOP ήταν Active High αντί για πύλη NAND θα έπρεπε να χρησιμοποιήσουμε μια πύλη AND 4 εισόδων, για να βγάλει 1 στην έξοδο και να λειτουργήσει τα reset.

Με ανάλογο τρόπο ενεργοποιώντας την άλλη άμεση είσοδο την preset μπορούμε να οδηγήσουμε τον μετρητή σε μια άλλη αρχική κατάσταση. Αν θέλουμε ο προηγούμενος ασύγχρονος τετράμπιτος μετρητής να ξεκινάει από το 0110 και να τελειώνει στο 1111. Θα πρέπει το κύκλωμα να γίνεται preset μόλις εμφανιστεί η κατάσταση 0000. Αν χρησιμοποιήσουμε NAND θα πρέπει να αντιστρέψουμε και τις 4 εισόδους και η έξοδος να πηγαίνει στο preset του 2^{ου} και 3^{ου} FLIP-FLOP, με την προϋπόθεση βέβαια ότι όλα τα FLIP-FLOP είναι Active Low. Στην περίπτωση που είναι Active High μπορούμε να χρησιμοποιήσουμε μια πύλη NOR 4 εισόδων η οποία στο 0000 θα μας δώσει 1 στην έξοδο της με την οποία ενεργοποιούμε τις άμεσες εισόδους Preset των FLIP-FLOP 2 και 3.

Οι προηγούμενοι ασύγχρονοι μετρητές μετράνε προς τα πάνω και τους ονομάζουμε **UP COUNTERs**. Στα λογικά κυκλώματα έχουμε και το

ακριβώς αντίθετο, δηλαδή τους μετρητές που μετράνε προς τα κάτω ή DOWN COUNTERs. Η σχεδίαση είναι όπως και προηγουμένως, δηλαδή πρέπει να γνωρίζουμε τα BIT μέτρησης, την αρχική κατάσταση και την τελική. Ας δούμε ένα παράδειγμα παρακάτω με ένα ασύγχρονο τρίμπιτο μετρητή mod 8, που μετράει διαδοχικά από το 111 μέχρι το 000.

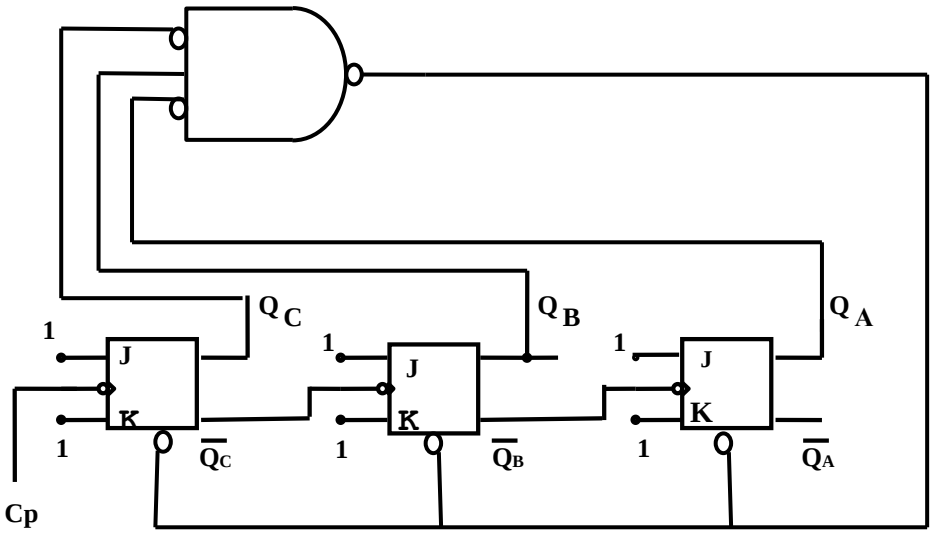




000 t_1 111 t_2 110 t_3 101 t_4 100 t_5 011 t_6 110 t_7 111 t_8 000 t_9

Βλέπουμε στο παραπάνω κύκλωμα ότι σε αντίθεση με τους **UP COUNTERs** οι **DOWN COUNTERs** συνδέονται κάπως διαφορετικά. Η συμπληρωματική έξοδος του πρώτου FLIP-FLOP γίνεται ρολόι για το δεύτερο FLIP-FLOP και η συμπληρωματική έξοδος του δεύτερου FLIP-FLOP γίνεται ρολόι για το τρίτο. Αν υποθέσουμε ότι πριν από την έλευση του 1ου παλμού οι κανονικές έξοδοι και των τριών FLIP-FLOP έχουν μηδενισθεί η ακολουθία μέτρησης του παραπάνω κυκλώματος θα είναι 111-110-101-100-0111-010-001-000-111-κ.λ.π. Ο μετρητής είναι ένας DOWN COUNTER, mod8 διότι έχει 8 καταστάσεις. Στην περίπτωση που θέλουμε να έχουμε μόνο την παρακάτω ακολουθία καταστάσεων, 111-110-101-100-011- και ξανά από την αρχή, δηλαδή ο μετρητής να είναι ένας DOWN COUNTER mod5, θα πρέπει μετά την κατάσταση 011 ο μετρητής να πηγαίνει στο 111. Για να πετύχουμε το παραπάνω λειτουργούμε όπως και προηγουμένως. Δηλαδή

μετά την τελευταία κατάσταση θα πρέπει ο μετρητής να γίνει Preset στο 111. Αν τα FLIP-FLOP είναι active low θα χρησιμοποιήσουμε μια πύλη NAND 3 εισόδων όπως φαίνεται παρακάτω. Το preset θα γίνει αμέσως μετά την τελευταία κατάσταση που είναι η 011, δηλαδή μόλις περάσει στο 010, η έξοδος του 1ου FLIP-FLOP αφού αντιστραφεί, η έξοδος του 2ου FLIP-FLOP ως έχει και η έξοδος του 3ου FLIP-FLOP αφού αντιστραφεί και αυτή οδηγούνται στην είσοδο της NAND η οποία θα δώσει 0 στην έξοδο και συνδεόμενη με τα 3 preset των FLIP-FLOP θα τα κάνει preset, δηλαδή θα τα οδηγήσει στην κατάσταση 111.

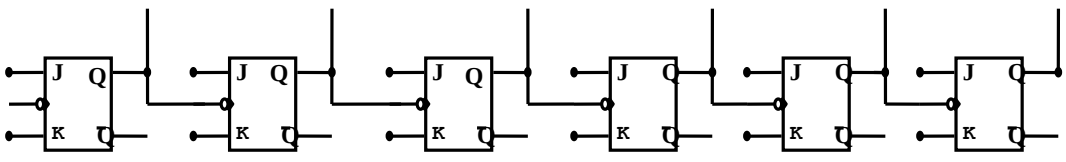


ΆΣΚΗΣΗ: Δίνεται ο τετράμπιτος μετρητής ριπής του παρακάτω σχήματος. Υποθέτουμε ότι και τα 4 FLIP-FLOP είναι αρνητικής ακμής και τα J και τα K τους είναι συνδεδεμένα στο 1, δηλαδή στο $\mathbf{V_{cc}}$.

1. Υπολογίστε το mode

2. Υπολογίστε τη συχνότητα στην έξοδο κάθε Flip-Flop αν η συχνότητα του κεντρικού ρολογιού είναι 1 MHz.

3. Ποιους αριθμούς μετράει αυτός ο μετρητής.



Μετρητής ριπής των έξη bit

Ο μετρητής αυτός έχει 2^6=64 διαφορετικές καταστάσεις είναι mod64 και μπορεί να μετρήσει από το 000000 μέχρι το 111111. Η συχνότητα στην έξοδο κάθε Flip-Flop είναι $F/2^k$ όπου k είναι ο αριθμός του Flip-Flop. Για το τελευταίο θα έχουμε στην έξοδο ότι F_6=1MHz/64=15,625 KHz.

Ασκήσεις

1. Σχεδιάστε ένα τετράμπιτο ασύγχρονο μετρητή ικανό να μετράει στον κώδικα BCD. Για την κατασκευή θα χρησιμοποιήσουμε T FLIP-FLOP αρνητικής ακμής, Active Low.

2. Σχεδιάστε ένα τετράμπιτο ασύγχρονο μετρητή ικανό να μετράει στον κώδικα EXCESS3. Για την κατασκευή θα χρησιμοποιήσουμε JK FLIP-FLOP αρνητικής ακμής, Active Low.

3. Σχεδιάστε ένα πεντάμπιτο ασύγχρονο μετρητής ριπής ικανό να μετράει από το 00000 μέχρι το 11000 και ξανά από την αρχή. Για την κατασκευή θα χρησιμοποιήσουμε JK FLIP-FLOP αρνητικής ακμής, Active Low.

4. Σχεδιάστε ένα πεντάμπιτο ασύγχρονο μετρητής ριπής ικανό να μετράει από το 00011 μέχρι το 11000 και ξανά από την αρχή. Για την κατασκευή θα χρησιμοποιήσουμε JK FLIP-FLOP αρνητικής ακμής, Active Low.

<u>Σύγχρονα κυκλώματα μετρητών</u>

Στον σύγχρονο δυαδικό μετρητή, η εφαρμογή των παλμών γίνεται ταυτόχρονα στις εισόδους ρολογιού όλων των FLIP-FLOP. Έτσι ο ρυθμός αρίθμησης, σε σχέση μ' έναν ασύγχρονο μετρητή, είναι πολύ μικρότερος, αφού όλα τα FLIP-FLOP αλλάζουν κατάσταση ταυτόχρονα, και όχι το ένα μετά το άλλο. Ως εκ τούτου, ο χρόνος αλλαγής της κατάστασης ολοκλήρου

του μετρητή είναι ίσος με το χρόνο λειτουργίας (αλλαγής κατάστασης) ενός μόνο FLIP-FLOP. Φυσικά η βελτίωση του χρόνου απόκρισης δεν πραγματοποιείται χωρίς κόστος. Το κύκλωμα του μετρητή γίνεται πολυπλοκότερο, αφού απαιτεί περισσότερα εξαρτήματα, προκειμένου να λειτουργήσει σωστά, δηλαδή εκτός από τα FLIP-FLOP χρειαζόμαστε και πύλες. Έχουν όμως και ένα σημαντικό πλεονέκτημα σε σχέση με τα ασύγχρονα κυκλώματα μετρητών που αφορά την δυνατότητα οι καταστάσεις μέτρησης να μην είναι αναγκαστικά διαδοχικές. Ας δούμε ένα παράδειγμα παρακάτω με ένα σύγχρονο δίμπιτο μετρητή με FLIP-FLOP τύπου JK αρνητικής ακμής. Για να προχωρήσουμε στην σχεδίαση του κυκλώματος βλέπουμε ότι χρειαζόμαστε 2 FLIP-FLOP. Πως θα συνδεθούν για να μας δώσουν το ζητούμενο αποτέλεσμα. Στην περίπτωση των σύγχρονων μετρητών ενεργούμε κατά κάποιο τρόπο όπως κάναμε στην συνδυαστική λογική. Πρέπει να βρούμε τον πίνακα αλήθειας του κυκλώματος. Ας δούμε πως γίνεται αυτό παρατηρώντας τον πίνακα μετάβασης του FLIP-FLOP τύπου JK. Τι μας λέει αυτός ο πίνακας. Λέει ότι αν το **Q(t)** είναι σε μια συγκεκριμένη κατάσταση και θέλω με τον ερχομό του παλμού να μεταβεί σε μια άλλη κατάσταση **Q(t+1)** θα πρέπει το J και το K του FLIP-FLOP να έχουν συγκεκριμένες τιμές.

Αν δηλαδή **Q(t)=0** και θέλω να γίνει **Q(t+1)=1** στον επόμενο παλμό θα πρέπει το J=1 και το K=X, δηλαδή αδιάφορη η τιμή του K. Από το 0 στο 0 θα πρέπει το J=0 και το K=X, από το 1 στο 0 θα πρέπει το J=X και το K=1, και από το 1 στο 1 θα πρέπει το J=X και το K=0.

Q(t)	Q(t+1)	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Με τα παραπάνω δεδομένα μπορούμε να συμπληρώσουμε τον παρακάτω πίνακα αλήθειας του μετρητή, λαμβάνοντας σαν δεδομένο ότι η αρχική

κατάσταση είναι η 00, η επόμενη είναι η 01, η επόμενη η 10, η επόμενη η 11, η επόμενη η 00 και ξανά από την αρχή. Για να συμβεί η πρώτη μετάβαση, δηλαδή από το 00 στο 01 θα πρέπει για το 1ο FLIP-FLOP το οποίο ονομάζω Α, πριν από την έλευση του παλμού το J_A=0 και το K_A =X, ενώ για το δεύτερο FLIP-FLOP που ονομάζω Β για να γίνει η μετάβαση από το 0 στο 1 θα πρέπει πριν από την έλευση του παλμού το J_B=1 και το K_B =X . Ομοίως για τις επόμενες μεταβάσεις. Αυτό που παρατηρώ είναι οι γραμμές των καταστάσεων που φαίνονται στον πίνακα καταστάσεων του μετρητή που είναι οι έξοδοι των FLIP-FLOP JK του μετρητή. Αν η γραμμή n είναι η παρούσα κατάσταση η επόμενη κατάσταση είναι η γραμμή (n+1). Με αυτά τα δεδομένα βλέπουμε την απαραίτητη αλλαγή που πρέπει να κάνει το κάθε FLIP-FLOP JK για να γίνει η απαιτούμενη μετάβαση. Με αυτά τα δεδομένα ο πίνακας αλήθειας του μετρητή συμπληρώνεται όπως φαίνεται παρακάτω.

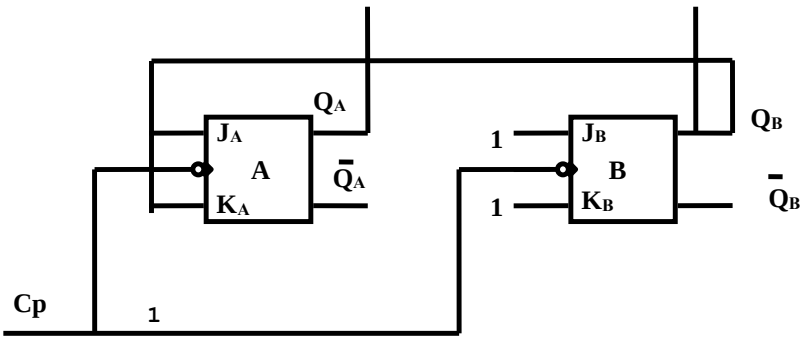
Q_A	Q_B	J_A	K_A	J_B	K_B
0	0	0	X	1	X
0	1	1	X	X	1
1	0	X	O	1	X
1	1	X	1	X	1

Βλέπουμε ότι η συνάρτηση J_A=Σ(1) με d=d(2,3), η K_A=Σ(3) με d=d(0,1), J_B=Σ(0,2) με d=d(1,3), η K_A=Σ(1,3) με d=d(0,2).

Μπορούμε πολύ εύκολα να απλοποιήσουμε τις παραπάνω 4 συναρτήσεις με τη βοήθεια των πινάκων Karnaugh και των αδιάφορων όρων της κάθε μίας. Θα έχουμε λοιπόν:

$$\mathbf{J_A=Q_B \text{ και } K_A= Q_B \quad \text{και} \quad J_A=K_A=1}$$

Το κύκλωμα θα είναι το παρακάτω.



Βλέπουμε στο προηγούμενο παράδειγμα ότι το κύκλωμα του δίμπιτου μετρητή υλοποιείται αποκλειστικά με τα FLIP-FLOP και όχι με πύλες.

Ας δούμε ένα άλλο παράδειγμα με υλοποίηση σύγχρονου μετρητή. Έστω ότι έχουμε να σχεδιάσουμε ένα τρίμπιτο μετρητή με Flip-Flop τύπου Τ. Για να υλοποιήσουμε ένα τέτοιο μετρητή χρειαζόμαστε 3FLIP-FLOP. Οι δυνατές καταστάσεις των εξόδων του μετρητή θα είναι διαδοχικά **000, 001, 010, 011, 100, 101, 110, 111**. Το πρώτο ψηφίο μέτρησης θα το παίρνουμε από την έξοδο του 1ου FLIP-FLOP το δεύτερο από το 2ο FLIP-FLOP και το τρίτο από το τρίτο FLIP-FLOP κ.λ.π. Δηλαδή η αρχική κατάσταση του μετρητή θα είναι η 000 και με τον πρώτο παλμό του ρολογιού θα πάει στην 001. Αυτό σημαίνει ότι το πρώτο FLIP-FLOP από την κατάσταση 0 θα πάει στην 0 το δεύτερο από την 0 θα πάει στην 0 και τέλος το, τρίτο από την κατάσταση 0 θα πάει στην 1. Έτσι θα προχωρήσουμε μέχρι το 111 που σημαίνει ότι σε ορισμένες περιπτώσεις θα έχουμε αλλαγή της εξόδου και σε άλλες όχι. Αυτό που μας ενδιαφέρει είναι με πιο τρόπο θα πρέπει να συνδέσουμε τις εξόδους των Flip-Flop για να πετύχουμε με ακρίβεια αυτές τις συγκεκριμένες μεταβάσεις και κατά συνέπεια την προκαθορισμένη ακολουθία μετρήσεων. Στον παρακάτω πίνακα φαίνεται πώς θα διεγείρουμε τα Flip-Flop τύπου Τ για να ακολουθήσουν τη σειρά μέτρησης **000, 001, 010, 011, 100, 101, 110, 111**.

Q_A	Q_B	Q_C	T_A	T_B	T_C
0	0	0	0	0	1
0	0	1	0	1	1
0	1	0	0	0	1

0	1	1	1	1	1
1	0	0	0	0	1
1	0	1	0	1	1
1	1	0	0	0	1
1	1	1	1	1	1

Πίνακας αλήθειας του τρίμπιτου μετρητή

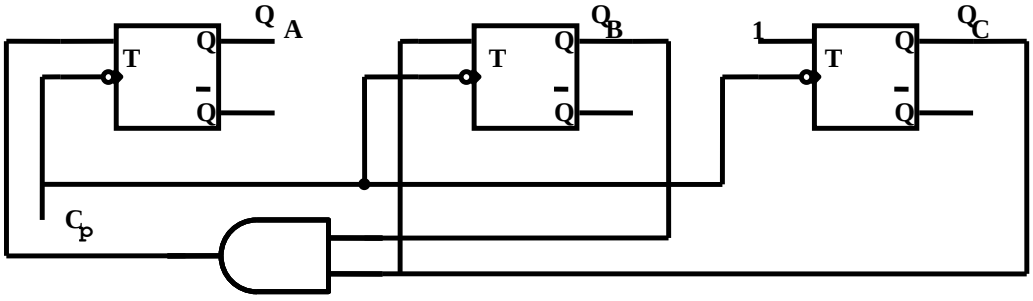
Βλέπουμε ότι το τρίτο FLIP-FLOP πρέπει να αλλάζει κατάσταση σε κάθε παλμό του ρολογιού. Επειδή είναι FLIP-FLOP τύπου Τ η αλλαγή αυτή θα γίνεται όταν το Τ είναι μόνιμα στο 1. Από τον παραπάνω πίνακα αλήθειας μπορούμε να γράψουμε ότι:

- $T_A(Q_A,Q_B,Q_C)=\Sigma(3,7)$
- $T_B(Q_A,Q_B,Q_C)=\Sigma(1,3,5,7)$
- $T_C(Q_A,Q_B,Q_C)=\Sigma(0,1,2,3,4,5,6,7)=1$

Οι παραπάνω συναρτήσεις μπορούν να απλοποιηθούν με τον πίνακα Karnaugh και θα έχουμε ότι:

- $T_A(Q_A,Q_B,Q_C)=Q_BQ_C$
- $T_B(Q_A,Q_B,Q_C)=Q_C$
- $T_C(Q_A,Q_B,Q_C)=1$

Και η υλοποίηση του μετρητή φαίνεται στο παρακάτω σχήμα.



Τρίμπιτος σύγχρονος δυαδικός μετρητής με Flip-Flop τύπου Τ και μία πύλη AND.

Σχεδίαση σύγχρονου μετρητή με αχρησιμοποίητες καταστάσεις

Έστω ότι μας ζητούν να σχεδιάσουμε ένα μετρητή τριών bit με Flip-Flop τύπου JK ο οποίος θα έχει την παρακάτω ακολουθία καταστάσεων 000, 001, 010, 100, 111. Για να υλοποιήσουμε ένα τέτοιο μετρητή χρειαζόμαστε τρία Flip-Flop. Θα χρησιμοποιήσουμε πέντε από τις οκτώ καταστάσεις που μπορούμε να πάρουμε στην έξοδο των Flip-Flop. Οι υπόλοιποι όροι θα είναι αδιάφοροι και μπορώ να τους χρησιμοποιήσω όπως και στα συνδυαστικά κυκλώματα για τη απλοποίηση των λογικών συναρτήσεων. Στην περίπτωση των σύγχρονων μετρητών με αχρησιμοποίητες καταστάσεις έχουμε δύο διαφορετικούς αδιάφορους όρους.

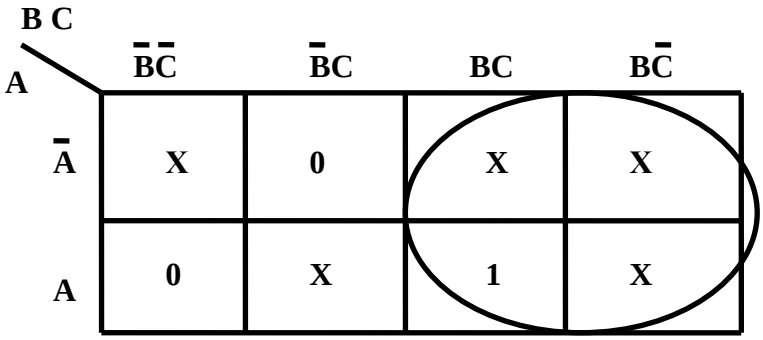
- Από τις αχρησιμοποίητες καταστάσεις
- Από τους πίνακες διέγερσης των FLIP-FLOP

Από τον πίνακα διέγερσης του Flip-Flop τύπου JK σχηματίζω εύκολα τον πίνακα αλήθειας του σύγχρονου μετρητή όπως φαίνεται παρακάτω.

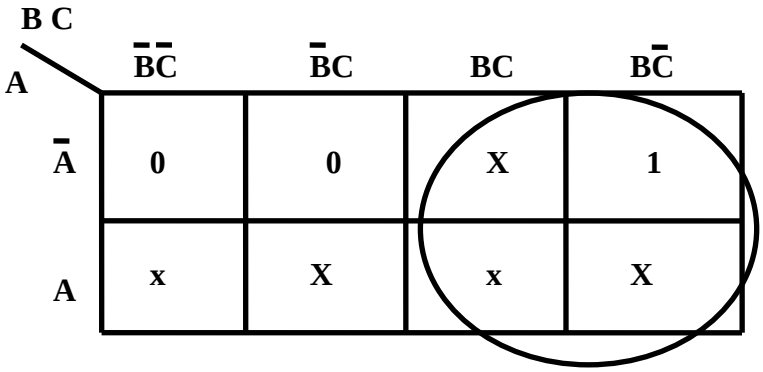
Q_c	Q_b	Q_a	J_c	K_c	J_b	K_b	J_a	K_a
0	0	0	0	x	0	x	1	x
0	0	1	0	x	1	x	x	1
0	1	0	1	x	x	1	0	x
1	0	0	x	0	1	x	1	x
1	1	1	x	1	x	1	x	1

Πίνακας αληθειας του μετρητή

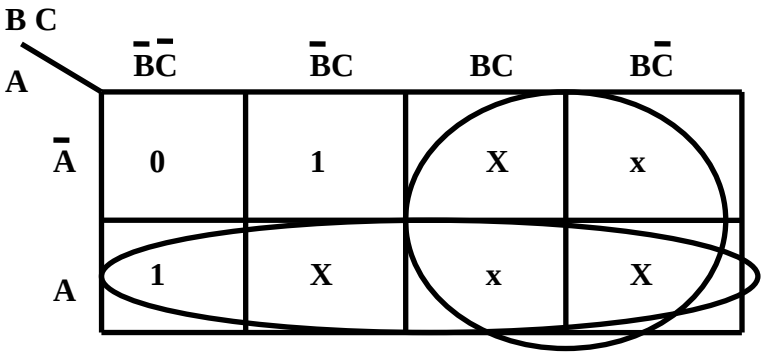
Βλέπουμε ότι το Ka=Kb=1 διότι τους αδιάφορους όρους μπορώ να τους πάρω ίσους με ένα και με αυτό τον τρόπο να έχω μόνο άσσους στον πίνακα αλήθειας των παραπάνω συναρτήσεων. Μένουν οι υπόλοιπες τέσσερεις για απλοποίηση με τον κλασσικό τρόπο δηλαδή με τον πίνακα Karnaugh. Δε θα πρέπει να ξεχάσω ότι θα έχω δύο ειδών αδιάφορους όρους αυτούς που προκύπτουν από τις αχρησιμοποίητες καταστάσεις του μετρητή. Για να μπορούμε να ελέγξουμε το κύκλωμα και επίσης τη διαδικασία μπορούμε να συμβολίζουμε με ένα μεγάλο Χ τις αχρησιμοποίητες καταστάσεις και αυτούς που προκύπτουν από τον πίνακα διέγερσης του JK με ένα μικρό x. Θα έχω λοιπόν ότι:



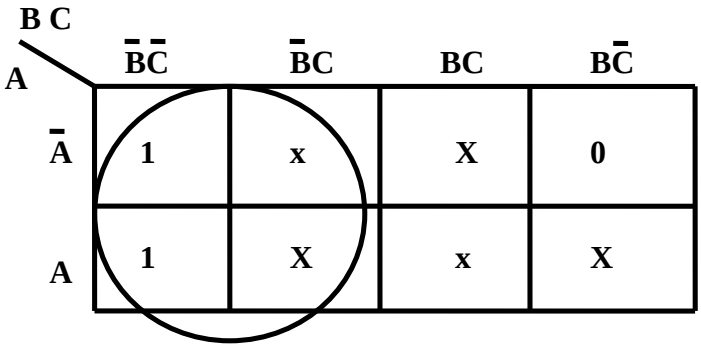
$Kc=Q_b$



$Jc=Qb$



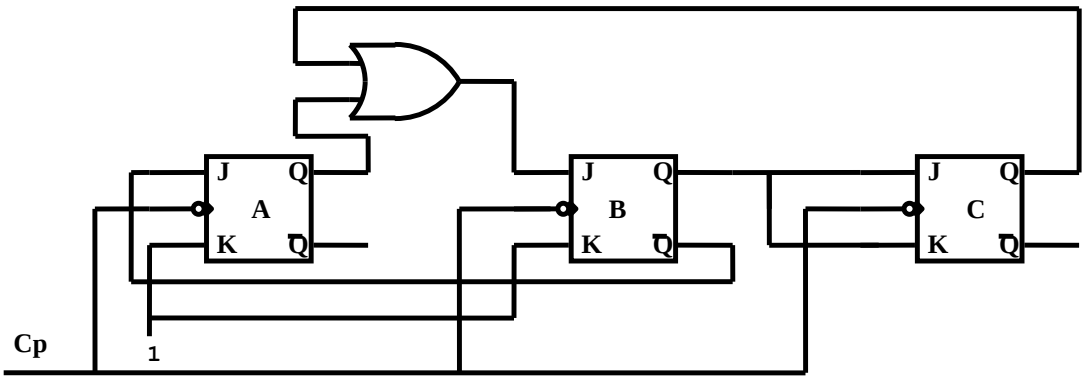
$Jb=Qc+Qa$



$Ja=\overline{Qb}$

Απλοποίηση με πίνακα Karnaugh των λογικών συναρτήσεων που καθορίζουν τις εισόδους των τριών Flip-Flop.

Η υλοποίηση του παραπάνω μετρητή είναι πολύ εύκολη αφού χρειάζονται μόνο τρία Flip-Flop τύπου JK και μία πύλη OR όπως δείχνει το παρακάτω σχήμα.



Κύκλωμα μετρητή όπου χρειάστηκε να χρησιμοποιήσουμε δύο ειδών αδιάφορους όρους.

Σχεδίαση μετρητών με εξισώσεις καταστάσεων

Συχνά σε ένα πρόβλημα είναι δυνατόν να γνωρίζουμε τις εξισώσεις καταστάσεων δηλαδή πως θα εξελίσονται οι έξοδοι των FLIP-FLOP στον χρόνο την χρονική στιγμή (t+1) γνωρίζοντας τις καταστάσεις την χρονική στιγμή t. Παρακάτω έχουμε ένα τέτοιο παράδειγμα όπου μας ζητούν να σχεδιάσουμε ένα ακολουθιακό κύκλωμα με Flip-Flop τύπου JK ικανό να ικανοποιεί τις παρακάτω εξισώσεις καταστάσεων :

$$A(t+1) = \overline{A}\cdot\overline{B}\cdot C\cdot D + \overline{A}\cdot\overline{B}\cdot C + A\cdot C\cdot D + A\cdot\overline{C}\cdot\overline{D}$$
$$B(t+1) = \overline{A}\cdot C + C\cdot\overline{D} + \overline{A}\cdot B\cdot\overline{C}$$
$$C(t+1) = B$$
$$D(t+1) = \overline{D}$$

Οι τιμές των μεταβλητών A,B,C, και D εννοείτε ότι είναι τη χρονική στιγμή t και μας ζητούν για τη χρονική στιγμή t+1 ότι πρέπει να πληρούν τις παραπάνω εξισώσεις. Μας ζητούν επιπλέον να χρησιμοποιήσουμε Flip-Flop τύπου JK. Αποδείξαμε προηγουμένως ότι η χαρακτηριστική εξίσωση του JK είναι:

$$Q(t+1) = J\cdot\overline{Q}(t) + \overline{K}\cdot Q(t)$$

Θα προσπαθήσουμε να εμφανίσουμε αυτή τη σχέση στις εξισώσεις του προβλήματος μας και θα έχουμε :

$$A(t+1) = \overline{A}\cdot\overline{B}\cdot C\cdot D + \overline{A}\cdot\overline{B}\cdot C + A\cdot C\cdot D + A\cdot\overline{C}\cdot\overline{D}$$
$$B(t+1) = \overline{A}\cdot(\overline{B}\cdot C\cdot D + \overline{B}\cdot C) + A\cdot(C\cdot D + \overline{C}\cdot\overline{D})$$
$$B(t+1) = J_A\cdot\overline{A} + \overline{K}_A\cdot A$$

$$J_A = \overline{B}\cdot C\cdot D + \overline{B}\cdot C = \overline{B}\cdot C\cdot(D+1) = \overline{B}\cdot C$$

$$\overline{K_A} = C\cdot D + \overline{C}\cdot\overline{D} \quad K_A = \overline{C}\cdot D + \overline{D}\cdot C$$

Όπου J_A , K_A είναι οι είσοδοι του Flip-Flop και **A** είναι η κανονική του έξοδος.

Με το ίδιο τρόπο μπορούμε να δούμε και για το δεύτερο Flip-Flop. Θα έχουμε λοιπόν:

$$B(t+1) = \overline{A}\cdot C + C\cdot\overline{D} + \overline{A}\cdot B\cdot\overline{C}$$
$$B(t+1) = \overline{A}\cdot C\cdot(B+\overline{B}) + C\cdot\overline{D}\cdot(B+\overline{B}) + \overline{A}\cdot B\cdot\overline{C}$$
$$B(t+1) = \overline{A}\cdot C\cdot B + \overline{A}\cdot C\cdot\overline{B} + C\cdot\overline{D}\cdot B + C\cdot\overline{D}\cdot\overline{B} + \overline{A}\cdot B\cdot\overline{C}$$
$$B(t+1) = (\overline{A}\cdot C + C\cdot\overline{D})\cdot B + (\overline{A}\cdot B + C\cdot\overline{D} + \overline{A}\cdot\overline{C})\cdot B$$
$$B(t+1) = J_B\cdot B + \overline{K}_B\cdot B$$

$$J_B = \overline{A}\cdot C + C\cdot\overline{D} \quad \text{και} \quad K_B = A\cdot\overline{C} + A\cdot D$$

βλέπουμε τη συνδεσμολογία των εισόδων του δεύτερου Flip-Flop το οποίο ονομάζουμε Β.

για την τρίτη εξίσωση τα πράγματα είναι πολύ πιο εύκολα όπως φαίνεται παρακάτω και θα έχουμε

$$C(t+1) = B = B\cdot(C+\overline{C}) = B\cdot C + B\cdot\overline{C}$$

και βγαίνει αν συγκρίνουμε την παραπάνω εξίσωση με τη χαρακτηριστική του **JK** ότι $J_C = B$ και $K_C = \overline{B}$ και τέλος για την τελευταία εξίσωση $D(t+1) = \overline{D}$

Με τις παραπάνω εξισώσεις είναι πολύ εύκολο να υλοποιήσουμε το κύκλωμα του ακολουθιακού κυκλώματος το οποίο ακολουθεί τις εξισώσεις καταστάσεων που είδαμε στην αρχή. Χρησιμοποιούμε Flip-Flop τύπου JK και τον απαραίτητο αριθμό λογικών πυλών.

Μετρητής BCD

Σχεδίαση μετρητή BCD με FLIP-FLOP τύπου Τ. Ο κώδικας BCD είναι ένας τετράμπιτος κώδικας, χρειαζόμαστε κατά συνέπεια 4 FLIP-FLOP τύπου Τ. Ο πίνακας διέγερσης του FLIP-FLOP τύπου Τ είναι ο παρακάτω.

T	Q(t+1)
0	Q(t)
1	$\overline{Q}(t)$

Αν το Τα είναι στο 0 το FLIP-FLOP παραμένει στην ίδια κατάσταση ενώ αν είναι στο 1 η έξοδος συμπληρώνεται. Ο πίνακας αλήθειας του μετρητή θα είναι ο παρακάτω:

A	B	C	D	T_A	T_B	T_C	T_D
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	1
0	0	1	0	0	0	0	1
0	0	1	1	0	1	1	1
0	1	0	0	0	0	0	1
0	1	0	1	0	0	1	1
0	1	1	0	0	0	0	1
0	1	1	1	1	1	1	1
1	0	0	0	0	0	0	1
1	0	0	1	1	0	0	1

Πίνακας

Βλέπουμε ότι οι συναρτήσεις T_A,T_B, T_C και T_D θα είναι οι παρακάτω.

T_A=Σ (7, 9)

T_B= Σ (3, 7)

T_C = Σ (1, 3, 5,7)

T_D =1

Επειδή έχουμε τον κώδικα BCD οι αδιάφοροι όροι θα είναι οι παρακάτω όπως φαίνεται στον πίνακα Karnaugh. d=d(10,11,12,13,14,15). Στην περίπτωση που έχουμε FLIP-FLOP τύπου Τ δεν έχουμε αδιάφορους όρους από τον πίνακα διέγερσης του FLIP-FLOP αλλά μόνο από τις αχρησιμοποίητες καταστάσεις όπως εδώ από τον κώδικα BCD. Οι αδιάφοροι όροι είναι 6 αυτοί που έχουμε ήδη συναντήσει σε παλαιότερα προβλήματα. Ο πίνακας Karnaugh που θα χρησιμοποιηθεί θα είναι τυποποιημένος και για τις τέσσερεις συναρτήσεις που αποτελούν τις εξόδους των 4 FLIP-FLOP τύπου Τ.

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$				
$\bar{A}B$				
AB	X	X	X	X
$A\bar{B}$			X	X

Ας απλοποιήσουμε πρώτα τη συνάρτηση **T_A=Σ (7, 9).** Ο πίνακας αλήθειας είναι ο παρακάτω.

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$				
$\bar{A}B$			1	
AB	X	X	X	X
$A\bar{B}$		1	X	X

Μετά από τη απλοποίηση θα πάρουμε τελικά $T_A = AD + BCD$

Πάμε στη συνέχεια να απλοποιήσουμε την επόμενη συνάρτηση που είναι η **T_B= Σ (3, 7).** Βλέπουμε ότι έχουμε μόνο ένα συνδυασμό με τέσσερα γειτονικά τετράγωνα όπου απλοποιούνται δύο από τις τέσσερεις μεταβλητές και η απλοποιημένη συνάρτηση θα είναι. $T_B = CD$.

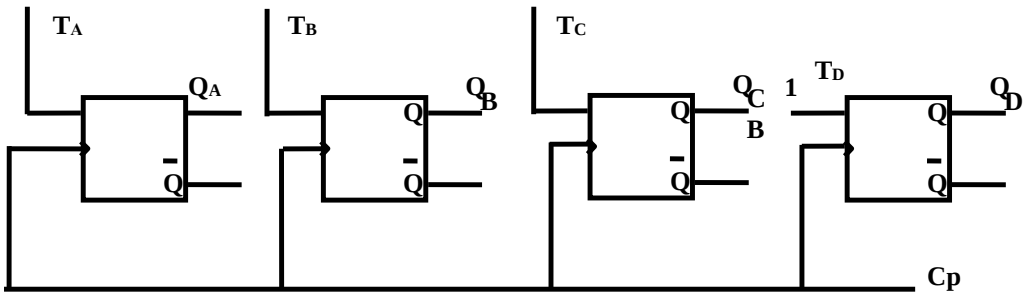
	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$			1	
$\bar{A}B$			1	
AB	X	X	X	X
$A\bar{B}$			X	X

Ας προχωρήσουμε στην επόμενη συνάρτηση **$T_C = \Sigma$ (1, 3, 5, 7)**. Ο πίνακας Karnaugh φαίνεται παρακάτω.

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$		1	1	
$\bar{A}B$		1	1	
AB	X	X	X	X
$A\bar{B}$			X	X

Τελικά μετά την απλοποίηση η συνάρτηση $T_C = \bar{A}D$. Βρήκαμε ότι το **T_D =1** και κατά συνέπεια το κύκλωμα θα είναι το παρακάτω.

$T_A = AD + BCD$ $T_B = CD$ $T_C = \bar{A}D$ $T_D = 1$

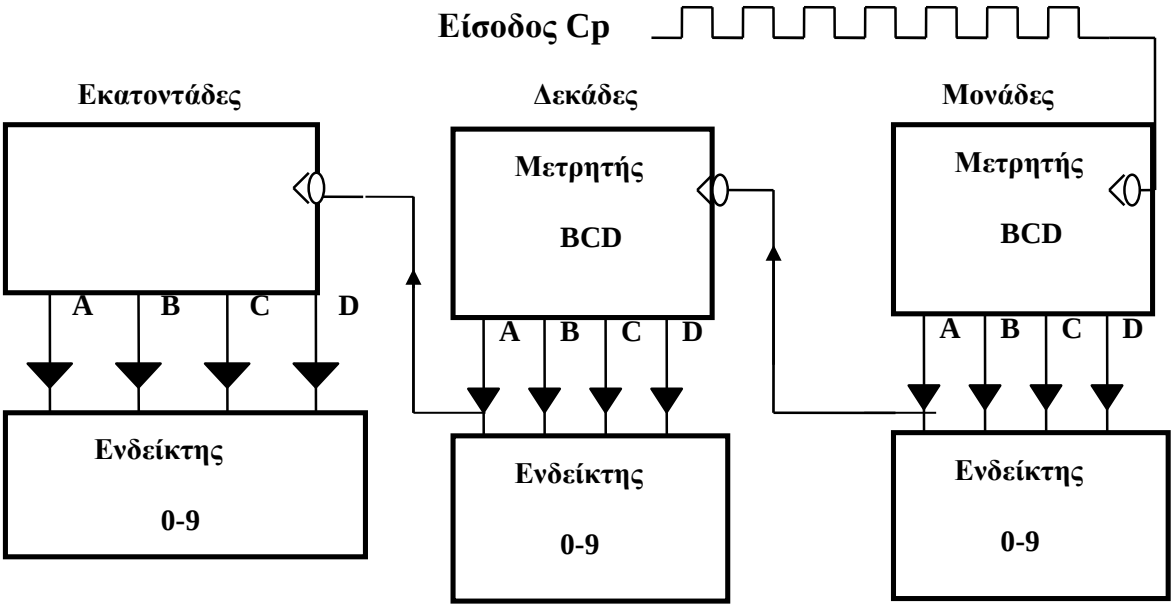


Δηλαδή η είσοδος **T_A** του 1ου Flip-Flop θα πρέπει να βλέπει τη παρακάτω συνάρτηση $T_A = AD + BCD$ η οποία υλοποιείται με δύο πύλες AND μία 2 εισόδων και 1 τριών εισόδων όπως επίσης και μία πύλη OR 2 εισόδων. Η συνάρτηση $T_B = CD$ υλοποιείται με μία πύλη AND 2 εισόδων. Η συνάρτηση

$T_C = \overline{A}D$ υλοποιείται με μία πύλη AND 2 εισόδων και τέλος η είσοδος του 4ου Flip-Flop συνδέεται με τη τροφοδοσία, δηλαδή στο λογικό 1.

Μετρητές BCD σε συνδεσμολογία

Όταν χρειάζεται να μετρήσουμε στο δεκαδικό χρησιμοποιούμε μετρητές BCD. Ένας μετρητής BCD μπορεί να μετράει από το 0 μέχρι το 9 όπως έχουμε ήδη δει και στην συνέχεια ξαναγυρίζουν από το 0 δηλαδή μηδενίζουν. Όταν χρειάζεται να μετρήσουμε παραπάνω από το 9 χρησιμοποιούμε τέτοιους στοιχειώδεις μετρητές BCD συνδεδεμένοι όπως φαίνεται στο παρακάτω σχήμα. Βλέπουμε ότι MSB του πρώτου μετρητή συνδέεται στο Clock του δεύτερου, το MSB του δεύτερου γίνεται Clock για το τρίτο κ.λ.π. Ξεκινώντας τη μέτρηση έχουν μηδενίσει όλοι οι μετρητές. Ξεκινάει λοιπόν ο 1ος από 0000, μετά πηγαίνει στο 0001, συνεχίζει και όταν φθάσει στο 1001 πρέπει στη συνέχεια να μηδενίσει δηλαδή να πάει από το 1001 στο 0000. Το MSB κάθε μετρητή είναι ρολόι για τον επόμενο, τα FLIP-FLOP είναι αρνητικής ακμής και αλλάζουν κατάσταση στην μετάπτωση του MSB από 1 σε 0. Η έξοδος κάθε μετρητή είναι συνδεδεμένη με ένα ενδείκτη 7 κομματιών οπότε εμείς θα βλέπουμε την μέτρηση στο δεκαδικό.+

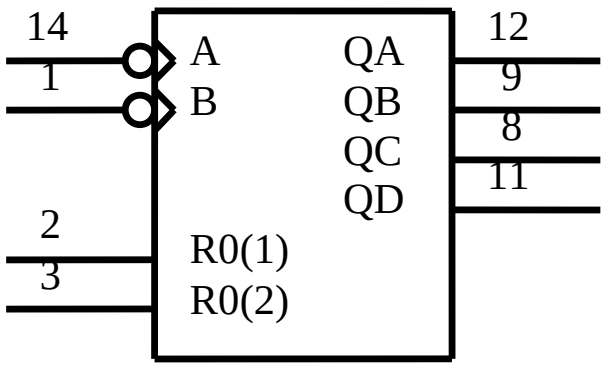


Η λειτουργία του παραπάνω κυκλώματος είναι η ακόλουθη. Αρχικά όλοι οι μετρητές οι οποίοι εννοείτε ότι μπορεί να είναι παραπάνω από τρεις

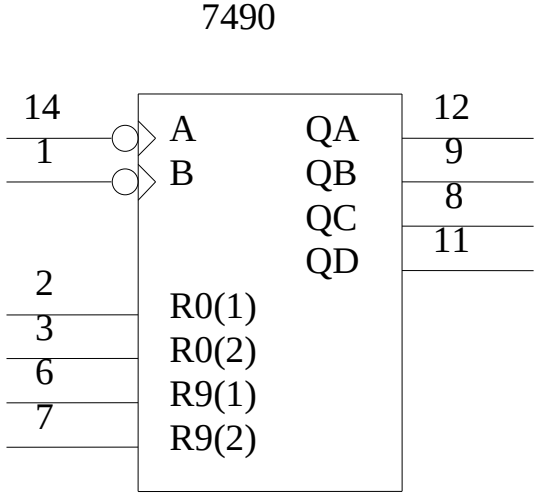
βρίσκονται αρχικά στο 000. Όπως φθάνουν οι παλμοί του ρολογιού οι μονάδες ανεβαίνουν ενώ οι δεκάδες και οι εκατοντάδες παραμένουν στο 0. Μετά από 9 παλμούς η ένδειξη του μετρητή θα είναι 009. Με τον ερχομό του δέκατου παλμού ο πρώτος μετρητής ξαναγυρίζει στο 0 ενώ ο δεύτερος ο οποίος βλέπει ένα παλμό ρολογιού στην είσοδο του κάνει την πρώτη του μέτρηση. Έτσι μετά από δέκα παλμούς του ρολογιού η ένδειξη του συστήματος θα είναι 010δηλαδή το 10. Μετά από 99 παλμούς του ρολογιού ο μετρητής των δεκάδων είναι στο 9 και ο μετρητής των μονάδων επίσης στο 9. Στον εκατοστό παλμό του ρολογιού ο μετρητής των μονάδων μηδενίζει, και με τη σειρά του ο μετρητής των δεκάδων κάνοντας έτσι το ασύγχρονο κύκλωμα να δίνει ένα παλμό στο μετρητή των εκατοντάδων και να αλλάζει με αυτό τον τρόπο κατάσταση. Έτσι μετά από 100 παλμούς του ρολογιού η ένδειξη του μετρητή θα είναι 100. Η διαδικασία συνεχίζεται μέχρι το 999. Στον χιλιοστό παλμό και οι τρεις μετρητές μηδενίζουν και έτσι η ένδειξη του μετρητή γίνεται ξανά 000 όπως στην αρχή.

Πρέπει να πούμε ότι αν θέλουμε να μετρήσουμε παραπάνω αρκεί να συνδέσουμε ένα μεγαλύτερο αριθμό στοιχειωδών μετρητών BCD με τον ίδιο τρόπο όπως φαίνεται στο προηγούμενο κύκλωμα. Έτσι για να μετρήσουμε στο δεκαδικό μέχρι το 999.999 θα χρειαστεί να συνδέσουμε έξη μετρητές BCD όπως κάναμε και προηγουμένως. Μπορούμε να υλοποιήσουμε του μετρητές MOD-10 με το ολοκληρωμένο 7493 που φαίνεται στο παρακάτω σχήμα.

7493



ή χρησιμοποιώντας τον ολοκληρωμένο δεκαδικό μετρητή TTL 7490 που φαίνεται στο παρακάτω σχήμα ή τον ολοκληρωμένο μετρητή πάνω κάτω (up-down) 74192 .

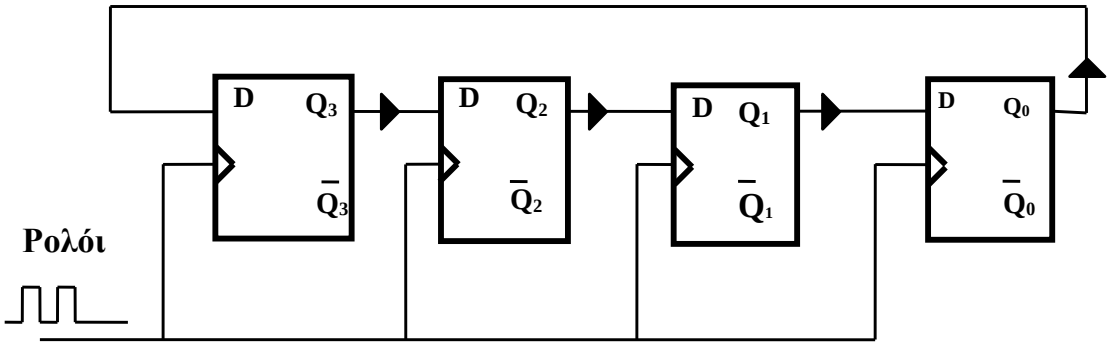


Κυκλικοί Μετρητές

Είναι δυνατόν να χρησιμοποιήσουμε καταχωρητές ολίσθησης για να κατασκευάσουμε διάφορους τύπους μετρητών. Όλοι οι κυκλικοί μετρητές χρησιμοποιούν την ανάδραση όπως θα δούμε και παρακάτω, δηλαδή η έξοδος του τελευταίου Flip-Flop είναι είσοδος για το πρώτο. Οι μετρητές ολίσθησης είναι όλοι μετρητές δακτυλίου η μετρητές Johnson.

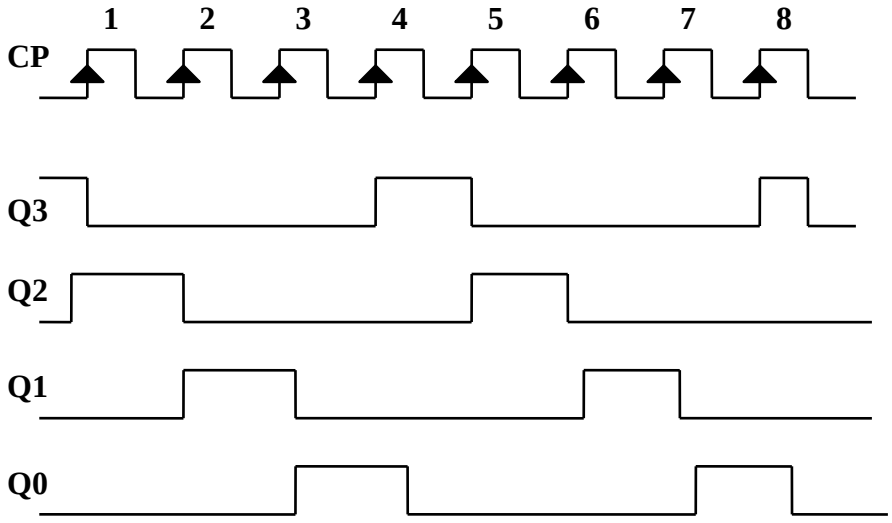
Μετρητής δακτυλίου

Ο μετρητής δακτυλίου ο πιο στοιχειώδης μετρητής που περιγράψαμε και πριν. Μπορούμε να δούμε ένα τέτοιο μετρητή στο παρακάτω σχήμα. Είναι κατασκευασμένος με Flip-Flop τύπου D αλλά θα μπορούσε να είχε επίσης κατασκευαστεί και με JK.



Μετρητής δακτυλίου

Τα Flip-Flop έχουν συνδεθεί με τέτοιο τρόπο ώστε η πληροφορία να μετακινείται από το αριστερό Flip-Flop στο δεξιό και μετά από το Q_0 στο Q_3. Γενικά υπάρχει ένας άσος στο κύκλωμα και κυκλοφορεί με τους παλμούς του ρολογιού και για το λόγο αυτό λέγεται μετρητής δακτυλίου. Οι κυματομορφές των εξόδων φαίνονται στο παρακάτω σχήμα παράλληλα με τους παλμούς του ρολογιού. Επειδή τα FLIP-FLOP είναι τύπου D για να δώσουν 1 στην κανονική έξοδο Q θα πρέπει το D=1. Έτσι κατά την έναρξη λειτουργίας του κυκλώματος 1 FLIP-FLOP γίνεται preset στο 1 και τα άλλα FLIP-FLOP reset στο 0. Ας πούμε ότι τη χρονική στιγμή t=0 έχουμε Q_3=1, Q_2=0, Q_1=0, Q_0=0 έχουμε δηλαδή εγγράψει τον αριθμό 1000. Μετά από 1 παλμό θα έχουμε 0100, μετά από 2 παλμούς 0010, μετά από 3 παλμούς 0001 και τέλος μετά από 4 θα έχουμε την αρχική κατάσταση 1000.



Ανάλυση λειτουργίας ενός μετρητή δακτυλίου

Η αρχική κατάσταση των Flip-Flop είναι η ακόλουθη Q_3=0, Q_2=0, Q_1=0, Q_0=0 [1000]. Μετά από τον πρώτο παλμό του ρολογιού η μονάδα πηγαίνει από το πρώτο Flip-Flop στο δεύτερο και η κατάσταση γίνεται [0100]. Με το δεύτερο παλμό έχουμε μετατόπιση του 1, δηλαδή [0010] με τον τρίτο [0001] και τέλος με τον τέταρτο ξαναγυρίζουμε στην αρχική κατάσταση [1000]. Αν συνεχίσουμε να εφαρμόζουμε παλμούς του ρολογιού θα έχουμε την ίδια ακριβώς εξέλιξη των καταστάσεων για τον

ίδιο ακριβώς αριθμό παλμών, δηλαδή όπως και προηγουμένως. Η χρονική μεταβολή των εξόδων φαίνεται στον παρακάτω πίνακα.

Q_3	Q_2	Q_1	Q_0	**Παλμοί Ρολογιού**
1	0	0	0	0
0	1	0	0	1
0	0	1	0	2
0	0	0	1	3
1	0	0	0	4
0	1	0	0	5
0	0	1	0	6
0	0	0	1	7
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.

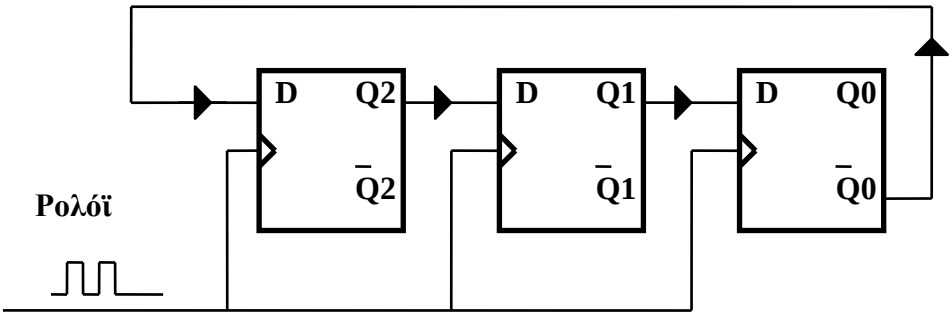
Ο παραπάνω μετρητής είναι ένας μετρητής MOD-4 γιατί έχει τέσσερις καταστάσεις. Για να φτιάξουμε ένα μετρητή δακτυλίου MOD-N θα πρέπει να συνδέσουμε N Flip-Flop. Βλέπουμε έτσι ότι για να φτιάξουμε ένα μετρητή δακτυλίου MOD-8 θα χρειαστούμε 8 Flip-Flop ενώ ένας απλός μετρητής MOD-8 χρειάζεται μόνο 3 Flip-Flop. Ένας μετρητής δακτυλίου για να ξεκινήσει θα πρέπει ένα Flip-Flop να είναι στο 1 και τα άλλα στο 0. Αυτό το επιτυγχάνουμε με ένα παλμό set σε ένα Flip-Flop και βάζοντας στο reset τα υπόλοιπα.

Μετρητής Jonson

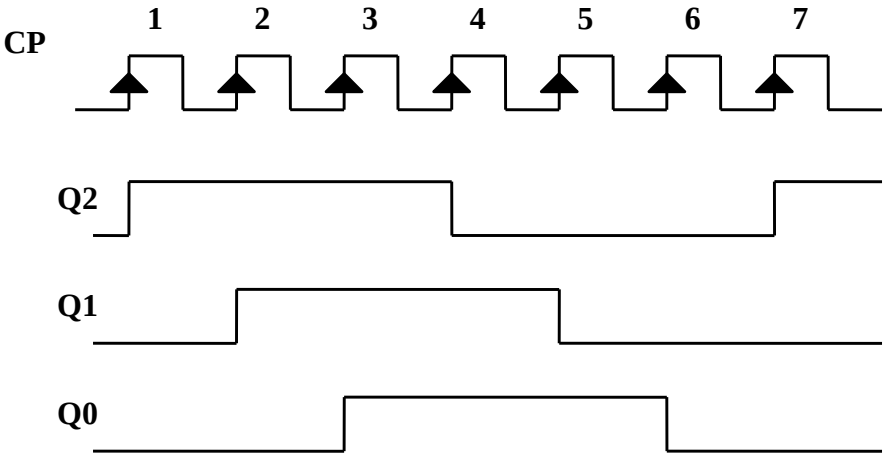
Αν τροποποιήσουμε ένα μετρητή δακτυλίου είναι εύκολα να φτιάξουμε ένα άλλο κυκλικό μετρητή όπως είναι ο μετρητής Johnson. Ένας μετρητής τέτοιου τύπου φαίνεται στο παρακάτω σχήμα.

Είναι ένας μετρητής Johnson 3 Bit. Είναι κατασκευασμένος ακριβώς όπως και ο μετρητής δακτυλίου που είδαμε προηγουμένως εκτός από το ότι η συμπληρωματική έξοδος του τελευταίου Flip-Flop είναι συνδεδεμένη με

την είσοδο του πρώτου. Ο παραπάνω μετρητής παίρνει έξη διαφορετικές καταστάσεις 000,100,110,111,011, και 001 όπως δείχνει και το διάγραμμα καταστάσεων των εξόδων των Flip-Flop στο παρακάτω σχήμα και είναι πολύ εύκολο να το επιβεβαιώσουμε από το κύκλωμα που είδαμε προηγουμένως .



Ο μετρητής Johnson με τρία Flip-Flop είναι ένας μετρητής MOD-6.



Το MOD ενός μετρητή Johnson είναι ίσο με δύο φορές των αριθμό των Flip-Flop που είναι φτιαγμένο. Αν για παράδειγμα συνδέσουμε πέντε Flip-Flop όπως στο προηγούμενο σχήμα θα πάρουμε ένα μετρητή Johnson MOD-10 του οποίου η έξοδος του κάθε Flip-Flop θα διαιρεί τη συχνότητα του ρολογιού με δέκα. Για να κατασκευάσουμε λοιπόν ένα μετρητή Johnson MOD-N όπου το N είναι άρτιος αριθμός θα πρέπει να χρησιμοποιήσουμε $\frac{N}{2}$ Flip-Flop.

Μετρητές πάνω κάτω [up-down counters]

Ένας μετρητής πάνω κάτω είναι ένας μετρητής ο οποίος μετράει πάνω η κάτω ανάλογα με την εντολή που του έχουμε δώσει. Θα σχεδιάσουμε παρακάτω ένα τέτοιο κύκλωμα για να δούμε τη διαδικασία που πρέπει να ακολουθήσουμε. Το ακολουθιακό κύκλωμα το οποίο θα αποτελείται από ένα δίμπιτο μετρητή πάνω κάτω και από δύο έξτρα εισόδους, την Ε και την Χ. Το Ε είναι η είσοδος επίτρεψης. Όταν η είσοδος όταν είναι ίση με μηδέν Ε=0 ο μετρητής δε μετράει και όταν είναι ίσο με ένα Ε=1 ο μετρητής μετράει. Η άλλη είσοδος Χ ενεργοποιεί τον τρόπο μέτρησης, δηλαδή όταν το Χ=1 ο μετρητής μετράει πάνω και όταν Χ=0 ο μετρητής μετράει κάτω. Για την κατασκευή θα χρησιμοποιήσουμε το τσιπ 7476 της TTL και τον μικρότερο αριθμό αριθμό πυλών. Ο πίνακας αλήθειας του μετρητή θα είναι ο παρακάτω:

A(t)	**B(t)**	**X**	**E**	**A[t+1]**	**B[t+1]**	**J_A**	**K_A**	**J_B**	**K_B**
0	0	0	0	0	0	0	X	0	X
0	0	0	1	1	1	1	X	1	X
0	0	1	0	0	0	0	X	0	X
0	0	1	1	0	1	0	X	1	X
0	1	0	0	0	1	0	X	X	0
0	1	0	1	0	0	0	X	X	1
0	1	1	0	0	1	0	X	X	0
0	1	1	1	1	0	1	X	X	1
1	0	0	0	1	0	X	0	0	X
1	0	0	1	0	1	X	1	1	X
1	0	1	0	1	0	X	0	0	X
1	0	1	1	1	1	X	0	1	X
1	1	0	0	1	1	X	0	X	0
1	1	0	1	1	0	X	0	X	1
1	1	1	0	1	1	X	0	X	0
1	1	1	1	0	0	X	1	X	1

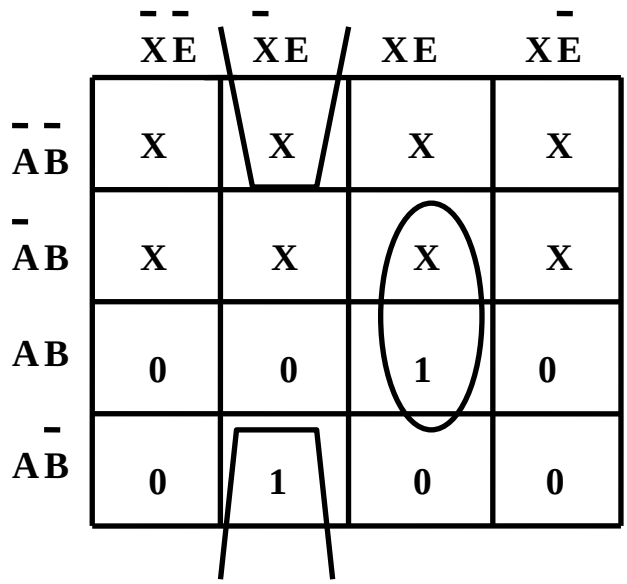
Ο παραπάνω πίνακας περιέχει τις εξόδους Α και Β των δύο Flip Flop τις χρονικές στιγμές (t) και (t+1), τις τιμές των εισόδων Ε και Χ και επίσης τις τιμές των εισόδων J και K των Flip-Flop για να είναι δυνατές οι παραπάνω μεταβάσεις. Για να συμπληρωθεί αυτός ο πίνακας αλήθειας χρειαστήκαμε τον πίνακα διέγερσης του Flip-Flop τύπου **JK** και τίποτα άλλο εκτός από τα δεδομένα του προβλήματος. Η επίλυση του προβλήματος τώρα είναι πολύ απλή και αρκεί να βρούμε τις τιμές των J και K των δύο Flip-Flop. Για το λόγο αυτό αρκεί να τις θεωρήσουμε σαν ανεξάρτητες συναρτήσεις Boole των μεταβλητών εισόδου και να τις απλοποιήσουμε. Θα έχουμε για τις τέσσερεις συναρτήσεις τις παρακάτω κανονικές μορφές τους παρακάτω αδιάφορους όρους και επίσης τους αντίστοιχους πίνακες Karnaugh που θα χρησιμοποιήσουμε για την απλοποίηση:

Για τη συνάρτηση $\mathbf{J_A=\Sigma(1,7)\ \ d_{jA}=d_{jA}(8,9,10,11,12,13,14,15)}$

$A B$ \ $X E$	$\bar{X}\bar{E}$	$\bar{X}E$	XE	$X\bar{E}$
$\bar{A}\bar{B}$	0	1	0	0
$\bar{A}B$	0	0	1	0
AB	X	X	X	X
$A\bar{B}$	X	X	X	X

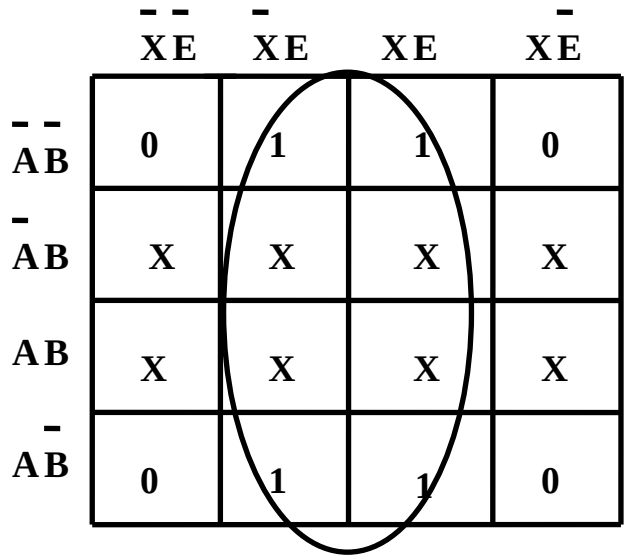
Βλέπουμε ότι η απλοποίηση δίνει $J_A = \bar{B}\bar{X}E + BXE$

Για την συνάρτηση $\mathbf{K_A=\Sigma(9,15)\ \ d_{KA}=(0,1,2,3,4,5,6,7)}$ ο πίνακας Karnaugh της συνάρτησης φαίνεται παρακάτω. Βλέπουμε ότι ενώ έχει διαφορετικούς ελαχιστόρους από την συνάρτηση $\mathbf{J_A}$ οι συνδυασμοί που προκύπτουν στον πίνακα Karnaugh της $\mathbf{K_A}$ είναι ίδιοι και κατά συνέπεια η απλοποιημένη μορφή της $\mathbf{K_A}$ θα είναι ίδια με την $\mathbf{J_A}$ όπως φαίνεται παρακάτω.



Μετά την απλοποίηση θα έχουμε $K_A = \bar{B}\bar{X}E + BXE$

Για το δεύτερο Flip-Flop η κανονική μορφή της εισόδου J είναι **J_B=Σ(1,3,9,11)** και οι αδιάφοροι όροι **d_{jB}=d_{jB}(4,5,6,7,12,13,14,15).** Ο πίνακας Karnaugh είναι ο παρακάτω.



Μετά την απλοποίηση θα πάρουμε ότι $J_B = E$

Τέλος η συνάρτηση της δεύτερης εισόδου θα είναι

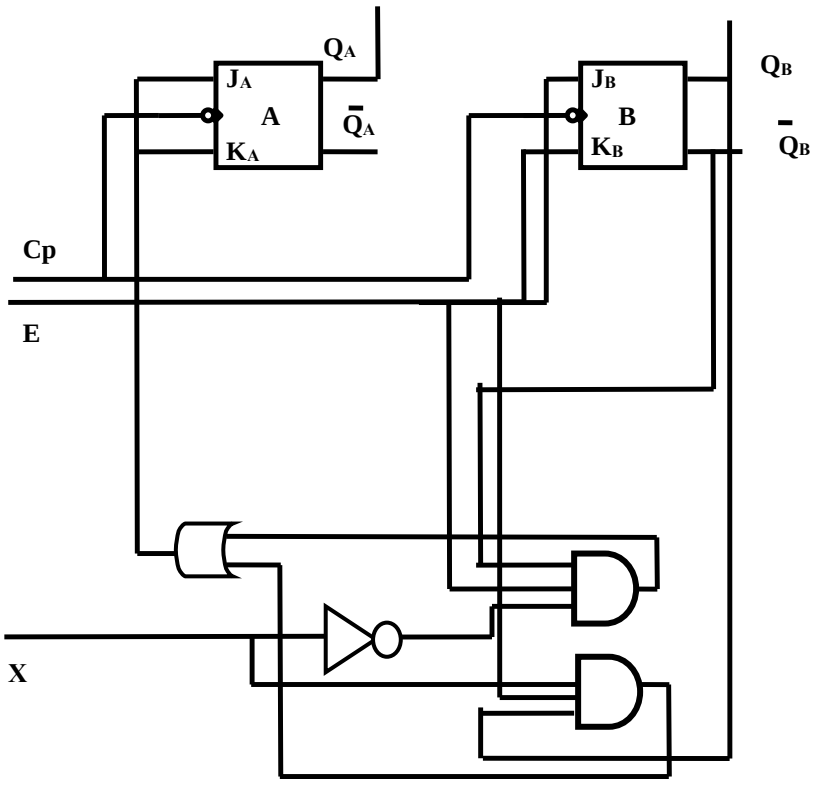
K_B=Σ(5,7,13,15) και αδιάφορους όρους **d_{KB}=d_{KB}(0,1,2,3,8,9,10,11)**

AB \ XE	$\bar{X}\bar{E}$	$\bar{X}E$	XE	$X\bar{E}$
$\bar{A}\bar{B}$	X	X	X	X
$A\bar{B}$	0	1	1	0
AB	0	1	1	0
$\bar{A}B$	X	X	X	X

Βλέπουμε ότι μετά την απλοποίηση θα πάρουμε **K_B=E.** Η κατασκευή του παραπάνω κυκλώματος είναι απλή. Ας συνοψίσουμε τις απλοποιημένες σχέσεις που βρήκαμε.

$J_A = \bar{B}\bar{X}E + BXE \qquad K_A = \bar{B}\bar{X}E + BXE$ ή $J_B = E \qquad K_B = E$

Για την κατασκευή του κυκλώματος θα χρειαστούμε πύλες όπως επίσης και 2 FLIP-FLOP τύπου JK.



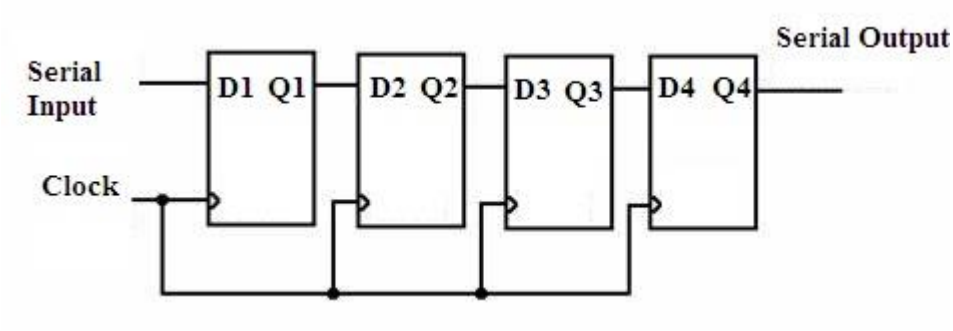
Κεφάλαιο 10. Καταχωρητές

Ένα σύνολο από δυαδικά κύτταρα κατάλληλο να αποθηκεύει δυαδικές πληροφορίες λέγεται καταχωρητής. Οι καταχωρητές (Registers) είναι ακολουθιακά ψηφιακά κυκλώματα στα οποία αποθηκεύονται δυαδικές πληροφορίες, με μήκος συνήθως, ίσο μ' αυτό της μιας λέξης. Αποτελούνται από έναν αριθμό ανεξαρτήτων FLIP-FLOP, ίσο με τον αριθμό των δυαδικών ψηφίων (κυττάρων) που αποτελούν την κάθε λέξη. Κάθε FLIP-FLOP καταγράφει ένα, και μόνο ένα, δυαδικό ψηφίο. Η πληροφορία η οποία καταγράφεται σ' έναν καταχωρητή, είναι πολύ εύκολο να ανακληθεί από αυτόν ανά πάσα στιγμή, θεωρώντας σαν δεδομένο ότι αμέσως μετά την εγγραφή της, εμφανίζεται στις εξόδους Q των FLIP-FLOP που τον αποτελούν. Οι καταχωρητές διατίθενται στην αγορά σε μορφή ολοκληρωμένου κυκλώματος και η κύρια χρήση που έχουν είναι από κυκλώματα που κάνουν υπολογισμούς, τα οποία τους χρησιμοποιούν για την αποθήκευση των αποτελεσμάτων των πράξεων που υλοποιούν. Κατηγοριούνται ανάλογα με τον τρόπο εγγραφής και ανάκλησης της πληροφορίας. Το μήκος ή τα BIT των καταχωρητών, μπορεί να είναι ίσο με 8, 16, 32, 64 ή 128.

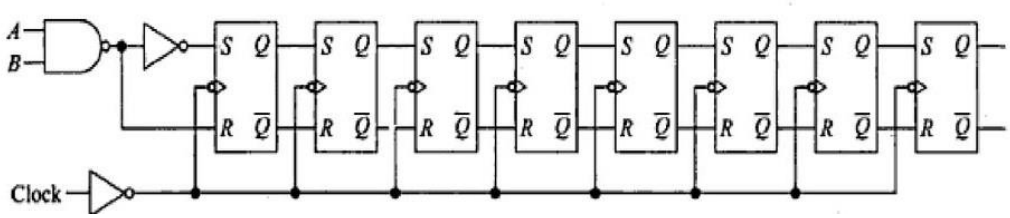
Ένας καταχωρητής αποτελείται από FLIP-FLOP και πύλες οι οποίες ελέγχουν πότε και πώς θα μεταφερθούν οι πληροφορίες από η προς τον καταχωρητή. Υπάρχουν πολλά είδη καταχωρητών ανάλογα με το πως γίνεται η εγγραφή της πληροφορίας και πως γίνεται η ανάκτηση ή η ανάγνωση. Οι περισσότερο γνωστοί είναι οι παράλληλοι και οι σειριακοί. Στους μεν πρώτους όλα τα bits φορτώνονται με ένα μοναδικό παλμό ενώ στην περίπτωση του σειριακού " η ολίσθησης " χρειάζονται τόσοι παλμοί όσος είναι ο αριθμός των BITs του καταχωρητή. Ο δεύτερος διαχωρισμός γίνεται με τον τρόπο που γίνεται η ανάκτηση της πληροφορίας παράλληλα η σειριακά και έτσι έχουμε τεσσάρων ειδών καταχωρητές.

1. **Serial in serial out** Και η πλήρωση του καταχωρητή και η ανάκτηση του περιεχομένου του γίνεται σειριακά. (serial-in , serial-out SISO) Ένα χαρακτηριστικό κύκλωμα τετράμπιτου καταχωρητή με Flip-Flop τύπου D φαίνεται στο παρακάτω κύκλωμα. Βλέπουμε τον τρόπο σύνδεσης των Flip-

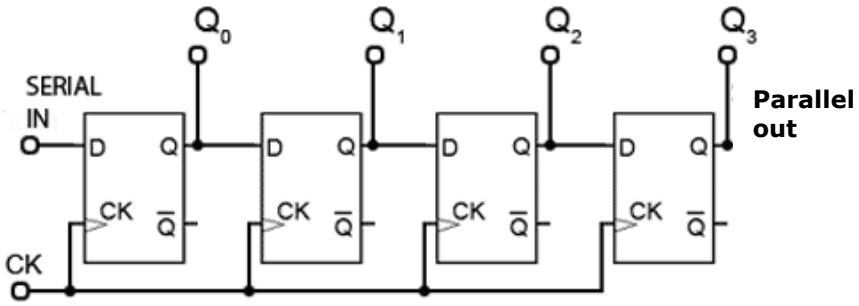
Flop όπως επίσης τη σύνδεση με το κεντρικό ρολόι. Όλα τα Flip-Flop αλλάζουν ταυτόχρονα κατάσταση.



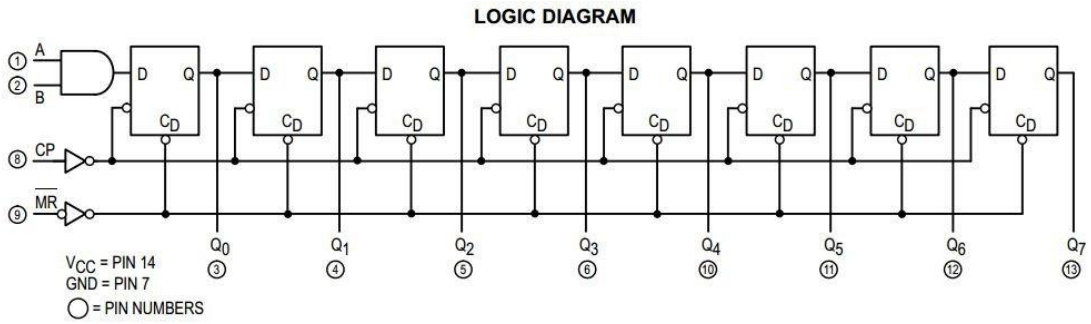
Βλέπουμε παρακάτω τη δομή του ολοκληρωμένου 74LS91 που περιέχει ένα καταχωρητή SISO των 8 bit.



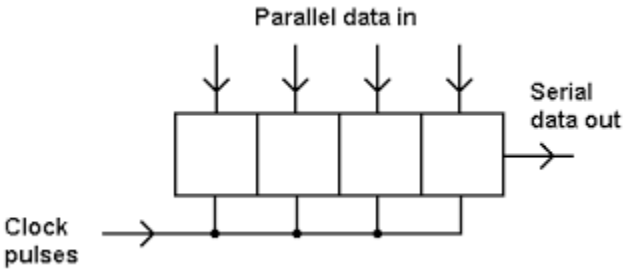
2. Serial in parallel out Η πλήρωση του καταχωρητή γίνεται σε σειρά και η ανάκτηση του περιεχομένου παράλληλα. Σειριακής εισόδου–παράλληλης εξόδου (serial-in , parallel-out SIPO). Βλέπουμε ότι ενώ η πλήρωση ή η εγγραφή για να γίνει στο κύκλωμα SIPO χρειάζονται 4 παλμοί όσα είναι τα Flip-Flop η ανάγνωση και των τεσσάρων bits γίνεται ταυτόχρονα.



Βλέπουμε παρακάτω τη δομή του ολοκληρωμένου 74LS164 που περιέχει ένα καταχωρητή SIPO των 8 bit.

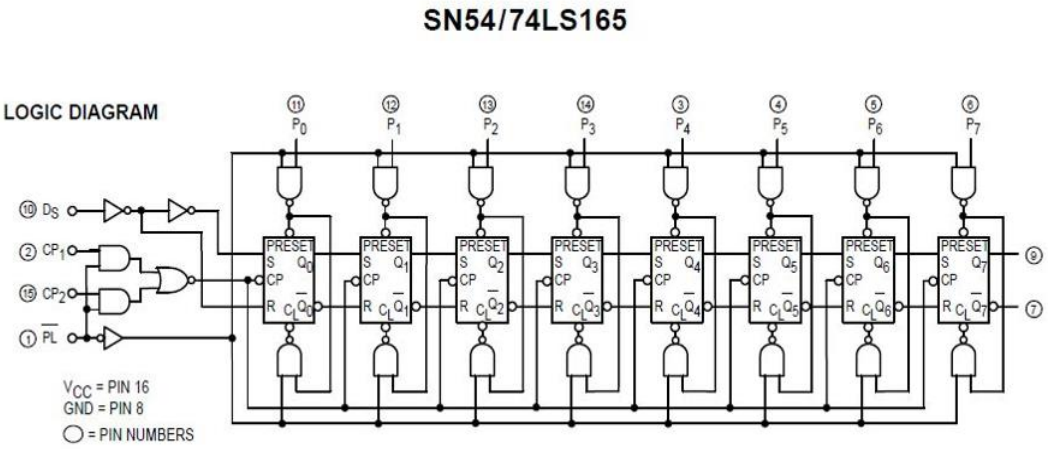


3. Parallel in serial out Η πλήρωση του καταχωρητή γίνεται παράλληλα και η ανάκτηση του περιεχομένου σε σειρά. Παράλληλης εισόδου – σειριακής εξόδου (parallel- in , serial-out PISO)

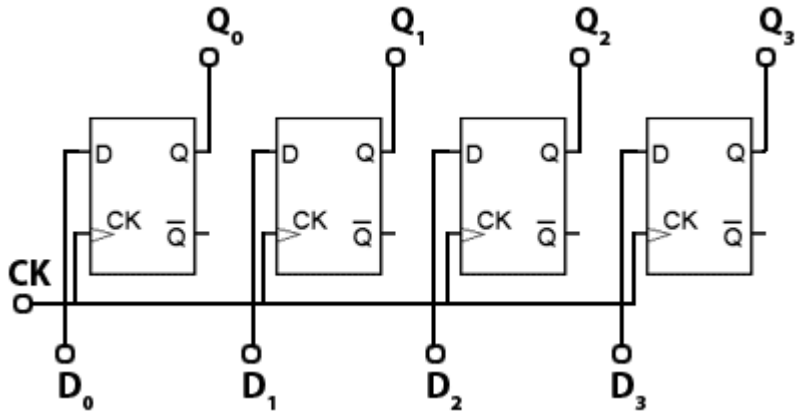


Βλέπουμε για τον παραπάνω καταχωρητή PISO για να γίνει η εγγραφή και στα τέσσερα κύτταρα μνήμης χρειάζεται αποκλειστικά ένας παλμός ενώ η ανάκτηση του περιεχομένου του απαιτεί τέσσερεις παλμούς.

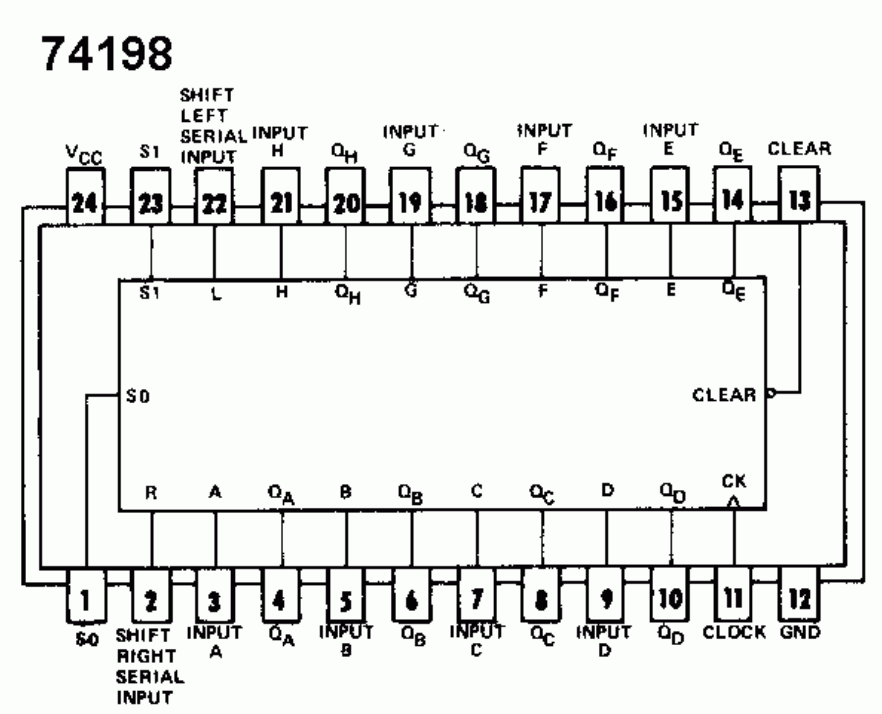
Βλέπουμε παρακάτω τη δομή του ολοκληρωμένου 74LS165 που περιέχει ένα καταχωρητή PISO των 8 bit.



4. Parallel in parallel out Η πλήρωση του καταχωρητή γίνεται παράλληλα σειρά και η ανάκληση του περιεχομένου του παράλληλα. Παράλληλης εισόδου – παράλληλης εξόδου (parallel-in , parallel-out PIPO)



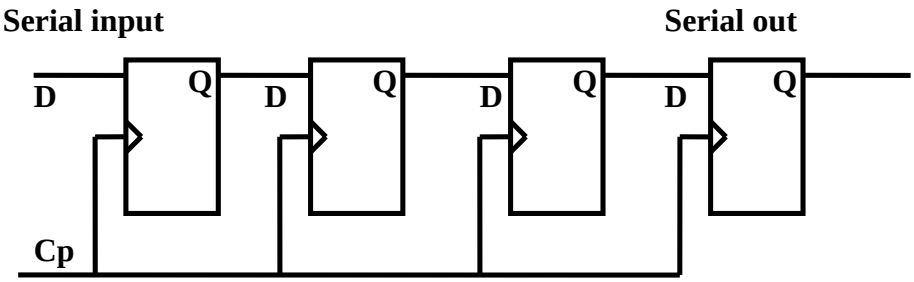
Βλέπουμε παρακάτω τη δομή του ολοκληρωμένου 74LS198 που περιέχει ένα καταχωρητή PIPO των 8 bit.



Στον καταχωρητή PIPO χρειάζεται ένας παλμός για την εγγραφή και κανένας για την ανάκτηση. Για τις τέσσερεις τύπους καταχωρητών οι παλμοί εγγραφής και ανάγνωσης φαίνονται παρακάτω.

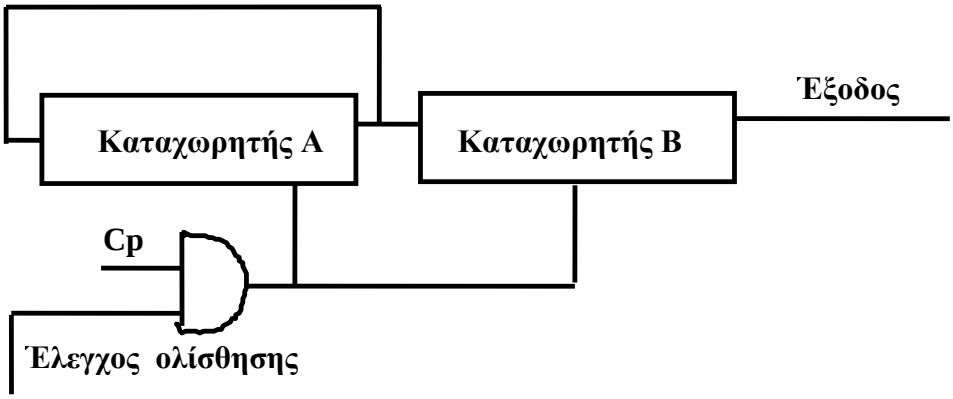
Τύπος καταχωρητή	**Παλμοί για εγγραφή n bits**	**Παλμοί για ανάκτηση n bits**
SISO	n	n-1
SIPO	n	0
PISO	1	n-1
PIPO	1	0

Ας δούμε ξανά τον σειριακό καταχωρητή SISO που είδαμε και προηγουμένως κατασκευασμένος με τέσσερα Flip-Flop τύπου D. Ας παρατηρήσουμε την συνδεσμολογία του κυκλώματος και τον τρόπο που έχουν συνδεθεί τα τέσσερα Flip-Flop.



Σειριακός καταχωρητής

Βλέπουμε ότι η έξοδος κάθε Flip-Flop τύπου D συνδέεται με την είσοδο του επομένου. Με αυτό τον τρόπο συνδεσμολογίας το περιεχόμενο του καταχωρητή σε κάθε παλμό ολισθαίνει και μία θέση δεξιά. Ένα ψηφιακό σύστημα λέγεται σειριακό όταν λειτουργεί με αυτό τον τρόπο. Ένα παράδειγμα μεταφοράς πληροφορίας από ένα καταχωρητή A σε ένα άλλο B φαίνεται στο παρακάτω σχήμα. Μία συμπληρωματική είσοδος η οποία καθορίζει τον έλεγχο ολίσθησης συνδέεται με το ρολόι σε μία πύλη AND δύο εισόδων. Με αυτό τον τρόπο μπορούμε να ελέγξουμε τη μεταφορά.

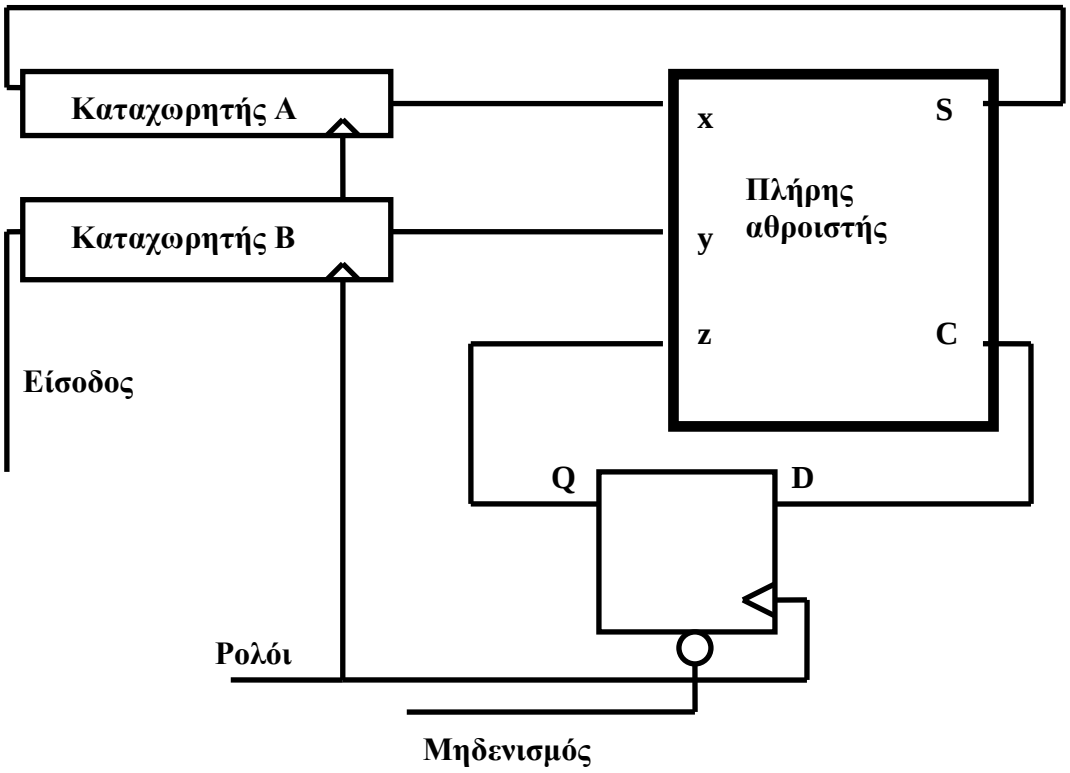


Συνδεσμολογία καταχωρητών για μεταφορά του περιεχομένου του καταχωρητή Α στον καταχωρητή Β.

Παρατηρούμε ότι κατά τη μεταφορά του περιεχομένου του καταχωρητή Α στον καταχωρητή Β το περιεχόμενο του καταχωρητή Α επανεγράφεται στον Α. Με τον τρόπο αυτό ακόμα και αν γίνει κάποιο λάθος μεταφοράς το αρχικό περιεχόμενο του Α παραμένει αναλλοίωτο. Αν για παράδειγμα έχουμε δύο τετράμπιτους καταχωρείται Α και Β με αρχικά περιεχόμενα **A=[1010]** και **B=[0000]**. Μετά από τέσσερις παλμούς που χρειάζονται για την ολίσθηση θα έχουμε τις παρακάτω μεταβολές στα περιεχόμενα των καταχωρητών.

παλμός	**Περιεχόμενο του Α**	**Περιεχόμενο του Β**
1	0101	0000
2	1010	1000
3	0101	0100
4	1010	1010

Ένα άλλο παράδειγμα για να καταλάβουμε τη λειτουργία του σειριακού καταχωρητή είναι η σειριακή πρόσθεση που φαίνεται παρακάτω.



Κύκλωμα σειριακού αθροιστή

Το παραπάνω κύκλωμα είναι το κύκλωμα ενός σειριακού αθροιστή και έχει τη δυνατότητα να προσθέσει δύο αριθμούς n bits όση είναι η χωρητικότητα των δύο καταχωρητών. Οι προσθετέοι βρίσκονται αποθηκευμένοι σε δύο καταχωρητές ολίσθησης Α, Β. Με τη χρήση ενός μόνο πλήρη αθροιστή τα bits των καταχωρητών προσθέτονται ζευγάρι ζευγάρι. Το άθροισμα S όπως ο καταχωρητής Α ολισθαίνει δεξιά παίρνει τη θέση του αρχικού περιεχομένου του καταχωρητή Α. Το κρατούμενο από την πρόσθεση το κρατάει το Flip-Flop τύπου D. Κατά τη διαδικασία της πρόσθεσης όπως ο αριθμός Β ολισθαίνει δεξιά μπορεί το περιεχόμενο του καταχωρητή να γεμίζει με ένα νέο αριθμό C ο οποίος εισάγεται στον καταχωρητή από τη σειριακή είσοδο. Ας δούμε όμως τη λειτουργία του κυκλώματος με ένα συγκεκριμένο παράδειγμα. Ας υποθέσουμε ότι έχουμε δύο αριθμούς Α και Β που θέλουμε να προσθέσουμε και οι οποίοι αριθμοί έχουν εγγραφεί σε δύο αντίστοιχα τετράμπιτους καταχωρητές. Έστω 0101 το περιεχόμενο του καταχωρητή Α και 0111 του καταχωρητή Β. Το περιεχόμενο του καταχωρητή Α όπως επίσης και η τιμή της εξόδου Q του Flip-Flop τύπου D θα είναι μετά από τους διαδοχικούς παλμούς.

1.	**A=0010**	**S=0**	**Q=1**
2.	**A=0001**	**S=0**	**Q=1**
3.	**A=1000**	**S=1**	**Q=1**
4.	**A=1100**	**S=0**	**Q=0**

Καταχωρητής Serial In - Parallel Out

Σε ένα τέτοιο καταχωρητή η εισαγωγή των δεδομένων γίνεται σειριακά και η εξαγωγή τους παράλληλα. Παράδειγμα ενός τέτοιου καταχωρητή φαίνεται στο παρακάτω σχήμα. Τα δεδομένα εισέρχονται σε μία από τις δύο εισόδους της πύλης NAND και η άλλη είσοδος χρησιμοποιείται σαν γραμμή ελέγχου. Ας υποθέσουμε την είσοδο A για τα δεδομένα και την B για τον έλεγχο της φόρτωσης και της μεταφοράς. Αν B=1 η πύλη NAND επιτρέπει τη διέλευση του A αντεστραμμένου. Τα δεδομένα ολισθαίνουν σειριακά στον καταχωρητή με το ρολόι. Αν B=0 η έξοδος της πύλης NAND είναι 1 και με τον 1^{Ο} παλμό το πρώτο RS θα έχει 0. Μετά από 4 παλμούς όλα τα RS θα έχουν ένα 0 στην έξοδο.

Στο προηγούμενο σχήμα είναι ένας καταχωρητής σειριακής πλήρωσης και παράλληλης εξαγωγής. (Serial in-Parallel out).

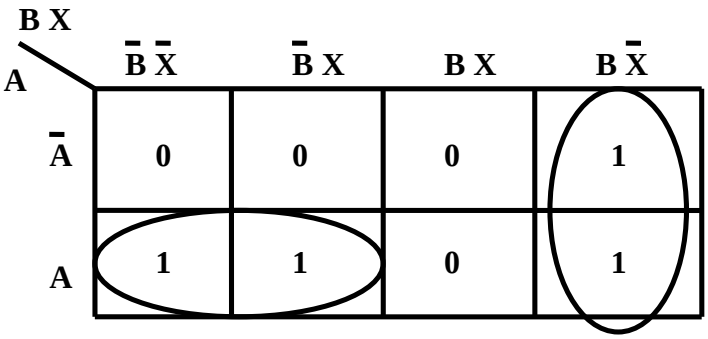
Παράδειγμα

Σχεδιάστε το ακολουθιακό κύκλωμα του οποίου ο πίνακας καταστάσεων δίνεται παρακάτω χρησιμοποιώντας ένα καταχωρητή των δύο bits και λογικές πύλες.

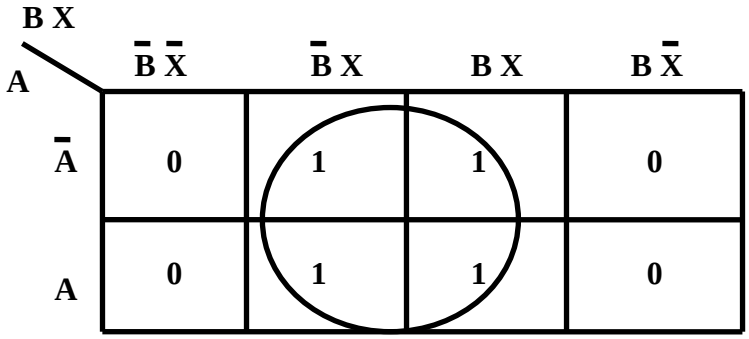
Παρούσα Κατάσταση **A**	Παρούσα Κατάσταση **B**	Είσοδος Εξωτερική **X**	Επόμενη Κατάσταση **A**	Επόμενη Κατάσταση **B**
0	0	0	0	0
0	0	1	0	1
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	1	1
1	1	0	1	0

1	1	1	0	1

Έχουμε λοιπόν τις παρακάτω κανονικές μορφές των συναρτήσεων A(t+1) και B(t+1). A(t+1)=Σ(2,4,5,6) και B(t+1)=Σ(1,3,5,7). Μπορούμε πολύ εύκολα να τις απλοποιήσουμε με τη βοήθεια του πίνακα Karnaugh.



$$A(t+1) = B\bar{X} + A\bar{B}$$



$$B(t+1) = X$$

Και το κύκλωμα υλοποιείται πολύ εύκολα.

Καταχωρητές ολίσθησης

Στους καταχωρητές ολίσθησης(shift registers) τα FLIP-FLOP που τους συνθέτουν συνδέονται μεταξύ τους με τέτοιον τρόπο, ώστε το συνολικό περιεχόμενο του καταχωρητή να μπορεί να ολισθαίνει είτε προς τα δεξιά είτε προς τα αριστερά, ανάλογα με τη τον σχεδιασμό και την κατασκευή του καταχωρητή.

Η μετακίνηση του περιεχομένου, που έχει καταχωρηθεί στα FLIP-FLOP μπορεί να γίνει τόσο αριστερά όσο και δεξιά, ανάλογα με την εφαρμογή που χρησιμοποιείται η διάταξη. Στην περίπτωση που η μετακίνηση είναι εφικτή και προς τις δύο διευθύνσεις, έχουμε τον λεγόμενο αμφίδρομο ολισθητή, του οποίου η φορά ολίσθησης καθορίζεται κάθε φορά με τη χρήση εξωτερικών κυκλωμάτων. Στην περίπτωση ενός καταχωρητή δεξιάς ολίσθησης, κάθε φορά που πραγματοποιείται μια ολίσθηση, το περισσότερο σημαντικό ψηφίο μπορεί να «γεμίζει» με λογικό «0» ή με λογικό «1» ανάλογα με τη λογική κατάσταση που επιβάλλεται στην είσοδο του αριστερότερου FLIP-FLOP. Αν η λογική κατάσταση, που επιβάλλεται, είναι πάντοτε το λογικό «0», τότε η ολίσθηση αυτή ονομάζεται «λογική ολίσθηση». Αν στη θέση του περισσότερου σημαντικού ψηφίου επανεισάγεται (δηλαδή παραμένει) η ίδια κατάσταση, τότε έχουμε «αριθμητική ολίσθηση». Έτσι η έκφραση «αριθμητική ολίσθηση» δηλώνει πως η ολίσθηση προς τα δεξιά ισοδυναμεί με την αριθμητική πράξη της διαίρεσης δια 2, ενός οποιουδήποτε προσημασμένου ως προς τη βάση 2 αριθμού. Αν, για παράδειγμα, έχουμε τον θετικό αριθμό $(00001000)_2 = (8)_{10}$, τότε μετά από μια ολίσθηση προς τα δεξιά θα έχουμε τον αριθμό $(00000100)_2 = (4)_{10}$. Αν έχουμε τον προσημασμένο αρνητικό δυαδικό αριθμό $(11000000)_2 = -(64)_{10}$, και πραγματοποιήσουμε μια ολίσθηση προς τα δεξιά, τότε θα λάβουμε τον $(10100000)_2 = -32(10)$ Με όμοιο τρόπο, γίνεται εύκολα αντιληπτό πως ολίσθηση προς τα αριστερά, με ταυτόχρονη εισαγωγή λογικού «0» στη θέση του λιγότερο σημαντικού ψηφίου, έχει σαν αποτέλεσμα τον πολλαπλασιασμό επί 2 του αριθμού που διατηρούσε αρχικά στις εξόδους του ο καταχωρητής ολίσθησης. Έτσι, αν για παράδειγμα, αρχικά ήταν καταχωρημένος ο αριθμός $(00001000)_2 = (8)_{10}$, τότε μετά από μια ολίσθηση αριστερά θα λάβουμε τον $(00010000)_2 = (16)_{10}$.

Ένας άλλος τύπος ολισθητή είναι ο κυκλικός καταχωρητής ολίσθησης, στον οποίο το LSB (λιγότερο σημαντικό ψηφίο) του καταχωρητή κατά τη δεξιά ολίσθηση δε χάνεται, αλλά μετακινείται πάλι στην είσοδό του καταχωρητή στο αριστερότερο FLIP-FLOP και αποκτά δηλαδή τη μέγιστη αξία ή αντίστοιχα το MSB (αριστερά ολίσθηση) επιβάλλεται στο δεξιότερο FLIP-FLOP του καταχωρητή και γίνεται δηλαδή το λιγότερο σημαντικό.

Στον παρακάτω πίνακας φαίνονται διάφοροι τρόποι ολίσθησης στους καταχωρητές. Παρατηρούμε ότι στα δύο πρώτα παραδείγματα με την κατακράτηση των ψηφίων ότι το τελικό περιεχόμενο περιέχει ακριβώς τα ψηφία που ολίσθησαν.

Θέσεις ολίσθησης	**3**	**2**	**1**	**4**
Φορά ολίσθησης	Αριστερά	Δεξιά	Δεξιά	Αριστερά
Κατακράτηση ψηφίων	ΝΑΙ	ΝΑΙ	ΟΧΙ	ΟΧΙ
Αρχικό περιεχόμενο καταχωρητή	**01001010**	**11010111**	**01011110**	**10111110**
Τελικό περιεχόμενο καταχωρητή	**01010010**	**11110101**	**00101111**	**11100000**

Με λίγο έντονο χρώμα γκρι φαίνονται οι μεταβολές σε κάθε περίπτωση.

Αμφίδρομοι καταχωρητές ολίσθησης

Ο αμφίδρομος καταχωρητής ολίσθησης είναι όπως λέει και το όνομα του ένας καταχωρητής ο οποίος μπορεί να ολισθαίνει και αριστερά και δεξιά. Η δε πλήρωση του μπορεί να γίνει είτε σειριακά είτε παράλληλα. Ας δούμε ορισμένα χαρακτηριστικά που διαθέτει ένας καταχωρητής αμφίδρομης ολίσθησης με παράλληλη φόρτωση.

- Μια είσοδο μηδενισμού ή είσοδο reset του καταχωρητή
- Μία είσοδο χρονισμού Cp για τους παλμούς ρολογιού
- Μία είσοδο ελέγχου της δεξιάς ολίσθησης του καταχωρητή
- Μία είσοδο ελέγχου της αριστερής ολίσθησης του καταχωρητή
- Μία είσοδο ελέγχου παράλληλης φόρτωσης του καταχωρητή
- Μία γραμμή σειριακής εισόδου του καταχωρητή
- Μία γραμμή σειριακής εξόδου του καταχωρητή
- n γραμμές παράλληλης εισόδου όσα και τα bits του καταχωρητή

- n γραμμές παράλληλης εξόδου όσα και τα bits του καταχωρητή
- Μία κατάσταση ελέγχου η οποία αφήνει αναλλοίωτες τις εγγραφές του καταχωρητή.

Εκτός από τους αμφίδρομους καταχωρητές έχουμε και τους μονόδρομους καταχωρητές ολίσθησης: Ένας καταχωρητής που ολισθαίνει τα περιεχόμενά του μόνο προς τη μία κατεύθυνση.

Αν υποθέσουμε ότι S1 και S2 είναι 2 άμεσες είσοδοι του καταχωρητή που επιτρέπουν τον έλεγχο λειτουργίας του τότε θα έχουμε:

S2	S1	Κατάσταση του καταχωρητή
0	0	Αναλλοίωτος
0	1	Δεξιά ολίσθηση
1	0	Αριστερά ολίσθηση
1	1	Παράλληλη πλήρωση

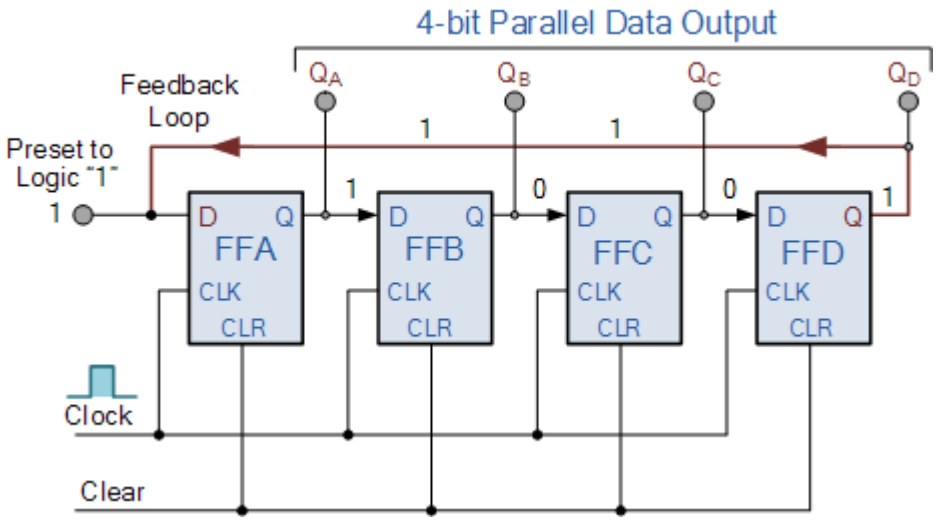
Καταχωρητής Δακτυλίου Johnson

Σε προηγούμενη αναφορά των καταχωρητών ολίσθησης Shift Register είδαμε ότι εάν εφαρμόσουμε ένα σειριακό σήμα δεδομένων στην είσοδο ενός Serial-in to Serial-out (SISO) καταχωρητή ολίσθησης Shift Register, θα έχουμε στην έξοδο την ίδια ακολουθία δεδομένων η οποία θα ανακτηθεί από το τελευταίο Flip-Flop. Αυτή η σειριακή μετακίνηση δεδομένων πραγματοποιείται μετά από έναν προκαθορισμένο αριθμό κύκλων ρολογιού επιτρέποντας έτσι στον καταχωρητή SISO να ενεργεί ως ένα είδος κυκλώματος καθυστέρησης χρόνου σε σχέση με το αρχικό σήμα δεδομένων που δέχεται στην είσοδο.

Τι γίνεται όμως αν συνδέσουμε την έξοδο αυτού του καταχωρητή μετατόπισης με την είσοδό του έτσι ώστε η έξοδος από το τελευταίο Flip-Flop, $\mathbf{Q_D}$ να γίνει η είσοδος του πρώτου Flip-Flop, $\mathbf{Q_A}$. Θα έχουμε ένα κύκλωμα κλειστού βρόγχου μέσα στον οποίο μετακινείται το ίδιο κομμάτι

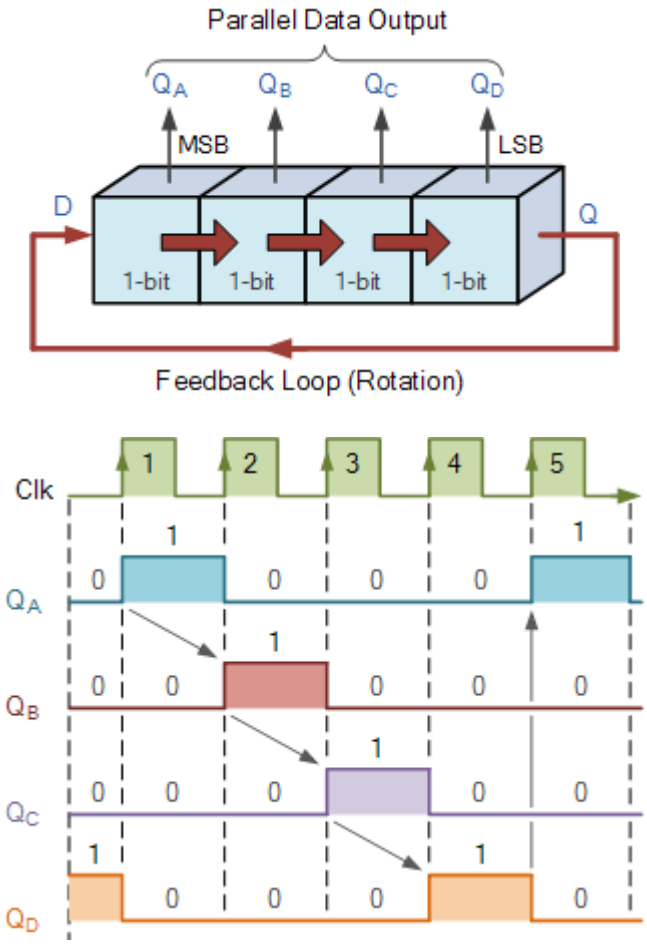
δεδομένων DATA. Αυτή είναι η κύρια λειτουργία ενός καταχωρητή δακτυλίου, όπως ονομάζεται.

Μπορούμε εύκολα, ενώνοντας την έξοδο του κυκλώματος με την είσοδο, (ανάδραση - feedback) να μετατρέψουμε ένα τυπικό κύκλωμα καταχωρητή ολίσθησης σε έναν καταχωρητή δακτυλίου. Ας το δούμε στο παρακάτω κύκλωμα.



Στον παραπάνω σύγχρονο καταχωρητή δακτυλίου, είναι καθορισμένο εξ' αρχής ότι ένα bit δεδομένων αποκλειστικά στον καταχωρητή να έχει λογική τιμή "1" και όλα τα υπόλοιπα bit να είναι στο "0". Για να επιτευχθεί αυτό, ένα σήμα "CLEAR" εφαρμόζεται πρώτα σε όλα τα Flip-Flops μαζί για να γίνουν "RESET" όλες οι έξοδοι και να πάνε σε λογικό επίπεδο "0" και στη συνέχεια εφαρμόζεται ένας παλμός "PRESET" στην είσοδο του πρώτου Flip -Flop πριν εφαρμοστούν οι παλμοί του ρολογιού. Αυτό άμεσα το τοποθετεί στη λογική τιμή 1. Έτσι, σε κάθε διαδοχικό παλμό ρολογιού, ο καταχωρητής κυκλοφορεί το ίδιο bit δεδομένων μεταξύ των τεσσάρων Flip-Flop ξανά και ξανά κάθε τέταρτο κύκλο ρολογιού. Αυτός ο τύπος κίνησης δεδομένων ονομάζεται "περιστροφή" και, όπως και ο προηγούμενος καταχωρητής μετατόπισης, το αποτέλεσμα της κίνησης του bit δεδομένων από αριστερά προς τα δεξιά μέσω ενός καταχωρητή δακτυλίου μπορεί να παρουσιαστεί γραφικά ως εξής μαζί με το διάγραμμα χρονισμού:

Περιστροφική κίνηση δεδομένων για ένα καταχωρητή δακτυλίου



Με το προηγούμενο παράδειγμα του καταχωρητή δακτυλίου βλέπουμε ότι έχει τέσσερις ξεχωριστές καταστάσεις, και όπως είναι γνωστό τον ονομάζουμε καταχωρητή «modulo-4» ή «mod-4» με την έξοδο κάθε Flip-Flop να παρουσιάζει μία συχνότητα ίση με το ένα τέταρτο ή το τέταρτο (1 / 4) της συχνότητας του κύριου ρολογιού όπως φαίνεται παραπάνω.

Το "MODULO" ή "MOD" ενός καταχωρητή είναι ο αριθμός των διαφορετικών καταστάσεων που μετράει ο καταχωρητής ονομάζονται επίσης ακολουθίες καταστάσεων. Μπορούμε σύμφωνα με την προηγούμενη συνδεσμολογία να σχεδιάσουμε ένα καταχωρητή δακτυλίου οποιουδήποτε αριθμού modulo. Έτσι ένας καταχωρητής δακτυλίου «mod-n» θα απαιτήσει για το σχεδιασμό του n Flip-Flops που συνδέονται μεταξύ τους για να κυκλοφορήσουν ένα μόνο bit δεδομένων παρέχοντας «n» διαφορετικές καταστάσεις στην έξοδο του.

Για παράδειγμα, ένας καταχωρητής δακτυλίου mod-8 απαιτεί οκτώ Flip-Flops και ένας καταχωρητής δακτυλίου mod-16 θα απαιτούσε δεκαέξι Flip-Flops . Ωστόσο, όπως στο παραπάνω παράδειγμά μας, χρησιμοποιούνται

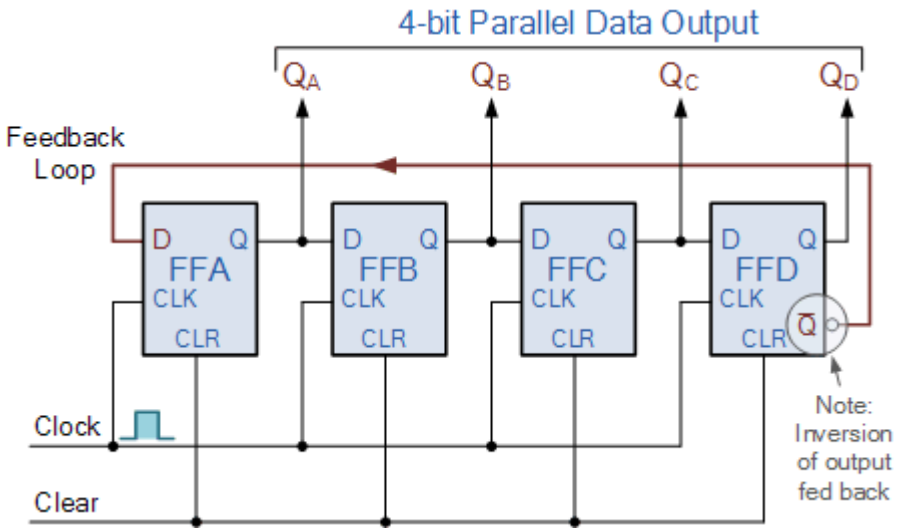
μόνο τέσσερις από τις πιθανές δεκαέξι καταστάσεις, καθιστώντας τους μετρητές δακτυλίων πολύ αναποτελεσματικούς όσον αφορά τη χρήση τους.

Καταχωρητής δακτυλίου Johnson

Ο καταχωρητής δακτυλίου Johnson είναι ένας άλλος καταχωρητής με ανατροφοδότηση όπως τον καταχωρητή δακτυλίου που είδαμε παραπάνω, εκτός του ότι αυτή τη φορά η ανεστραμμένη έξοδος Q του τελευταίου Flip-Flop είναι πλέον συνδεδεμένη με την είσοδο D του του πρώτου Flip-Flop όπως φαίνεται παρακάτω.

Το κύριο πλεονέκτημα αυτού του τύπου καταχωρητή δακτυλίου είναι ότι χρειάζεται μόνο το ήμισυ του αριθμού των Flip-Flop σε σύγκριση με τον κλασσικό καταχωρητή δακτυλίου. Έτσι, ένας καταχωρητής Johnson "n-Flip-Flop " θα κυκλοφορήσει ένα μόνο bit δεδομένων δίνοντας ακολουθία 2n διαφορετικών καταστάσεων και ως εκ τούτου μπορεί να θεωρηθεί ως "καταχωρητής mod-2n".

Τετράμπιτος καταχωρητής δακτυλίου Johnson



Η χρήση της συμπληρωματικής εξόδου Q’ του τελευταίου Flip-Flop που τροφοδοτεί την είσοδο D του 1ου Flip-Flop προκαλεί τον καταχωρητή να "μετρήσει" με διαφορετικό τρόπο. Ο καταχωρητής Johnson μετράει επάνω και στη συνέχεια προς τα κάτω καθώς η αρχική λογική "1" περνάει προς τα δεξιά, αντικαθιστώντας την προηγούμενη λογική τιμή "0". Ένας τετράμπιτος καταχωρητής δακτυλίου Johnson αναπαράγει 8 διαφορετικές

καταστάσεις. Καθώς η αντεστραμμένη έξοδος Q συνδέεται με την είσοδο D, αυτές οι 8 καταστάσεις των εξόδων των τεσσάρων Flip_Flop επαναλαμβάνονται συνεχώς και είναι: "1000", "1100", "1110", "1111", "0111", "0011", "0001", "0000" όπως φαίνονται στον παρακάτω πίνακα αλήθειας.

Πίνακας αλήθειας ενός τετράμπιτου καταχωρητή δακτυλίου Johnson

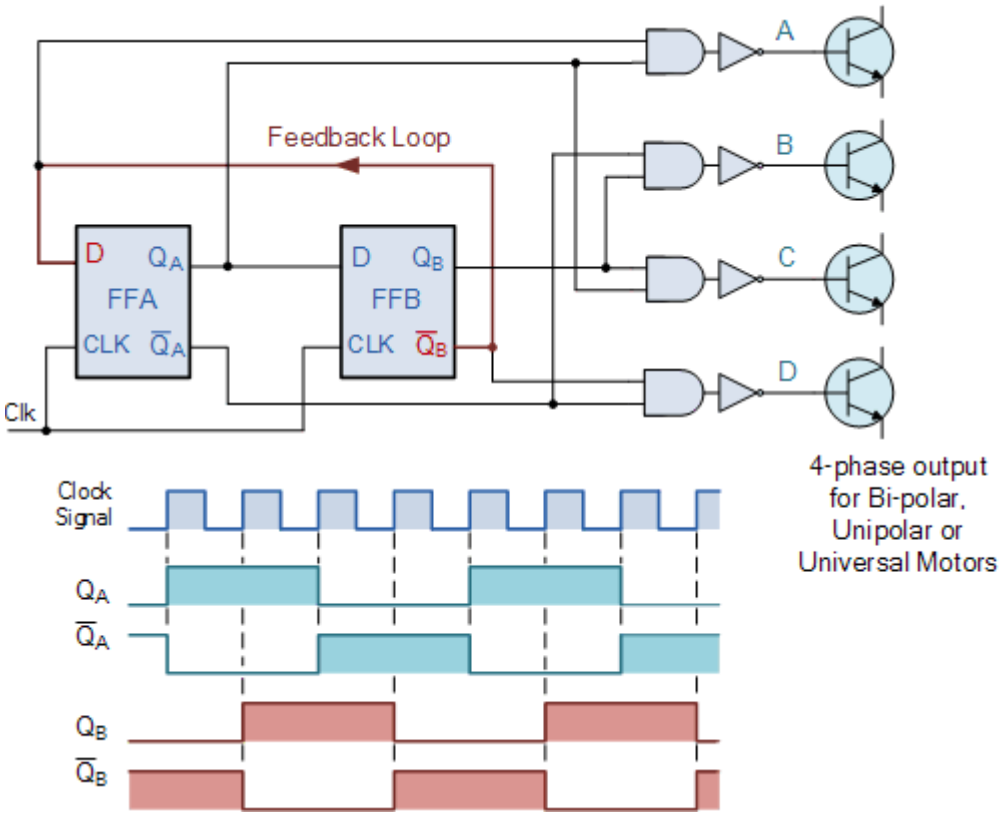
Αριθμός παλμού	**FF_A**	**FF_B**	**FF_C**	**FF_D**
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0
4	1	1	1	1
5	0	1	1	1
6	0	0	1	1
7	0	0	0	1

Εκτός από τις καταστάσεις μέτρησης ή την περιστροφή των δεδομένων μέσα σε έναν συνεχή κλειστό βρόχο, οι μετρητές δακτυλίου μπορούν επίσης να χρησιμοποιηθούν για την ανίχνευση ή την αναγνώριση διαφόρων συνδυασμών bits ή αριθμητικών τιμών σε ένα σύνολο δεδομένων. Συνδέοντας απλές λογικές πύλες όπως οι πύλες AND ή OR στις εξόδους των Flip-Flops, το κύκλωμα μπορεί να ανιχνεύσει έναν καθορισμένο αριθμό ή μια καθορισμένη τιμή.

Οι τυπικοί μετρητές δακτυλίου Johnson 2, 3 ή 4 Flip-Flops μπορούν επίσης να χρησιμοποιηθούν για να διαιρέσουν τη συχνότητα του κεντρικού ρολογιού. Για παράδειγμα, ένας καταχωρητής δακτυλίου Johnson 3 καταστάσεων θα μπορούσε να χρησιμοποιηθεί ως γεννήτρια κύματος 3-φάσεων, διαφοροποιημένες κατά 120 μοίρες μεταξύ τους.

Ο τυπικός καταχωρητής Johnson 5 σταδίων, όπως το κοινώς διαθέσιμο CD4017 χρησιμοποιείται γενικά ως σύγχρονο κύκλωμα καταχωρητή.

Άλλοι συνδυασμοί όπως το μικρότερο κύκλωμα 2 Flip-Flop που ονομάζεται επίσης τετραπλός ταλαντωτής ή γεννήτρια μπορεί να χρησιμοποιηθεί για την παραγωγή τεσσάρων μεμονωμένων εξόδων που έχουν διαφορά φάσης 90 μοίρες όπως φαίνεται παρακάτω.



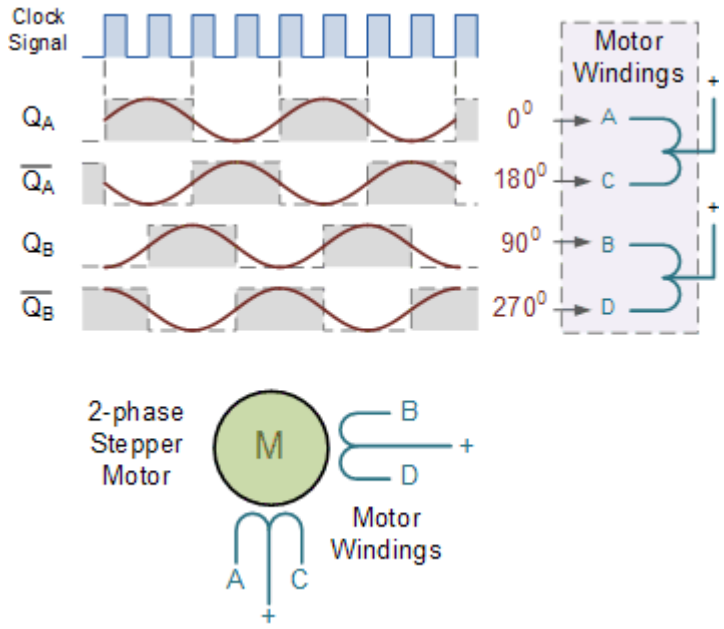
Οι 4 δυνατοί συνδυασμοί των κανονικών και συμπληρωματικών εξόδων των 2 Flip_Flop του καταχωρητή δίνουν στα σημεία A, B, C, D τις συναρτήσεις που φαίνονται παρακάτω:

ΕΞΟΔΟΣ	**A**	**B**	**C**	**D**
$\overline{Q}_A + Q_B$	1	0	0	0

$Q_A + \overline{Q}_B$	0	1	0	0
$\overline{Q}_A + \overline{Q}_B$	0	0	1	0
$Q_A + Q_B$	0	0	0	1

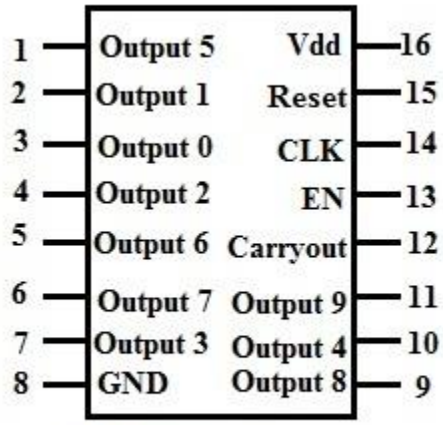
Δεδομένου ότι οι τέσσερις έξοδοι, από το Α στο Δ μετατοπίζονται με διαφορά φάσης κατά 90 μοίρες η μία με την άλλη, μπορούν να χρησιμοποιηθούν σε κυκλώματα οδήγησης βηματικού κινητήρα για έλεγχο θέσης όπως φαίνεται παρακάτω.

Έλεγχος βηματικού κινητήρα



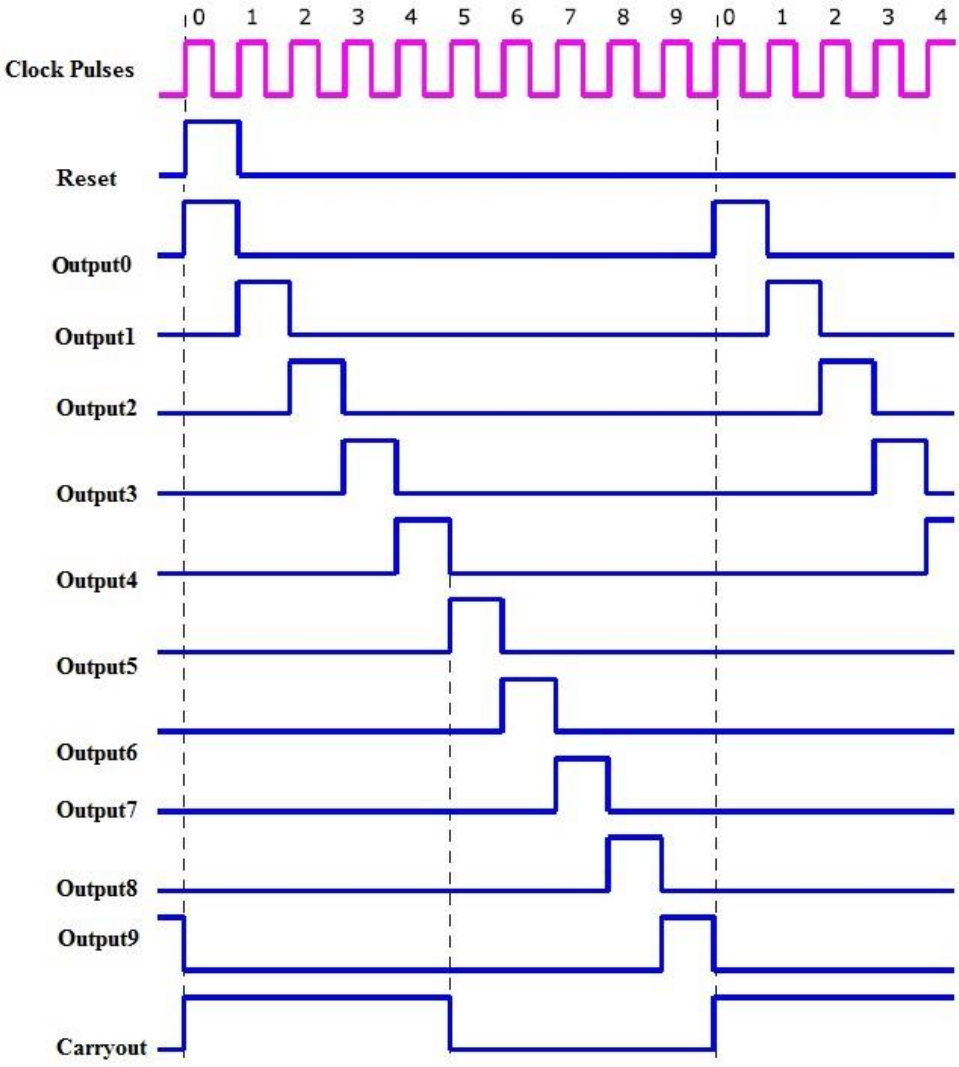
Κύκλωμα οδήγησης βηματικού κινητήρα. Η ταχύτητα περιστροφής του βηματικού κινητήρα εξαρτάται κυρίως από τη συχνότητα του ρολογιού και τα πρόσθετα κυκλώματα για την οδήγηση των παραπάνω κινητήρων θα τα δείτε στο αντίστοιχο μάθημα των κινητήρων όπου θα σας εξηγηθούν με λεπτομέρεια οι απαιτήσεις κίνησης των βηματικών κινητήρων.

Οι μετρητές δακτυλίου Johnson διατίθενται σε τυπική μορφή IC TTL ή CMOS, όπως το CD4017, ή το CD4022.

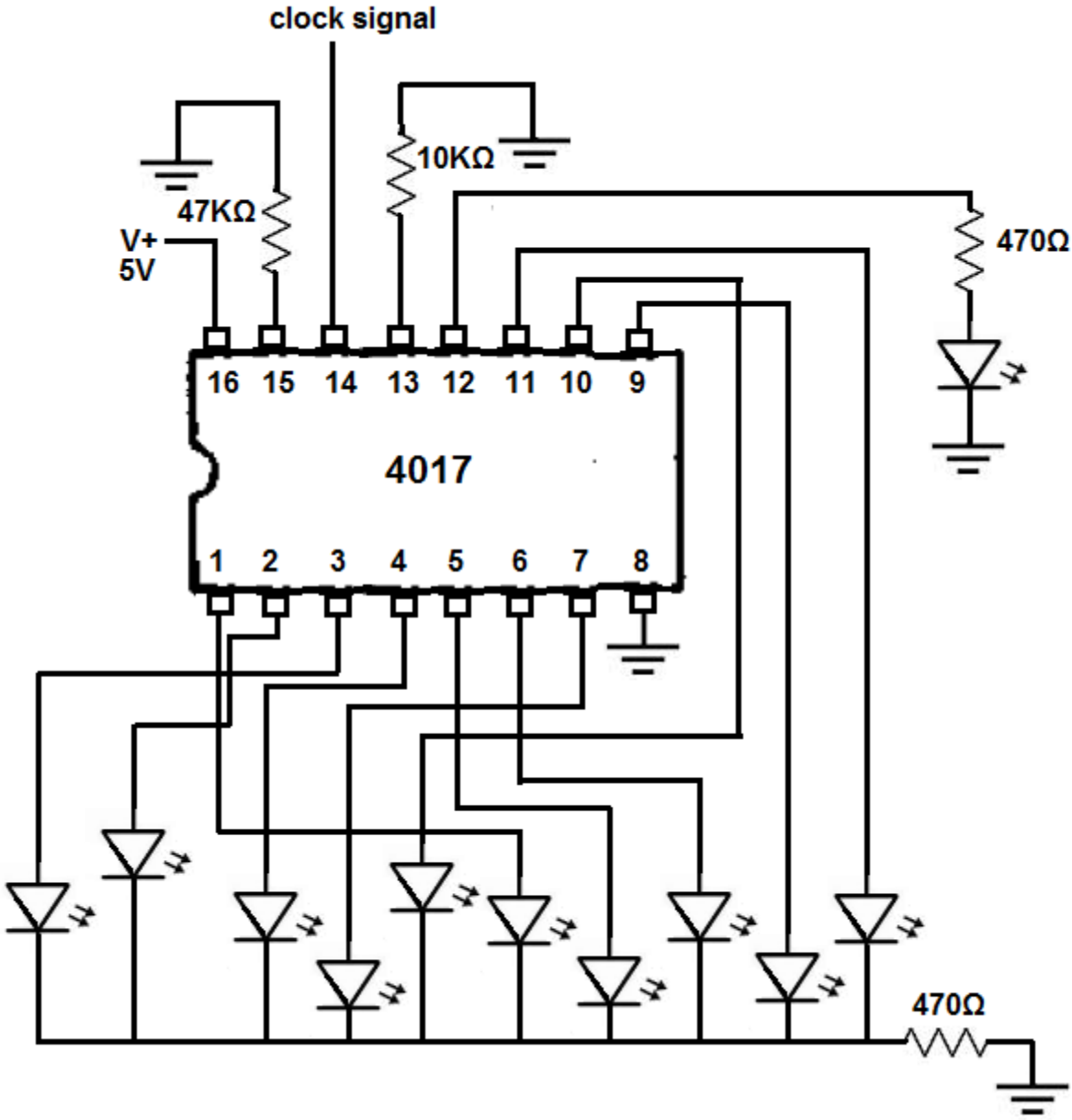


4017 -Decade Counter

Το παραπάνω κύκλωμα έχει 10 διαφορετικές καταστάσεις όπως φαίνεται και στο παρακάτω διάγραμμα χρονισμού.



Βλέπουμε τον μοναδικό λογικό 1 το οποίο μετακινήται για 10 παλμούς του ρολογιού από την έξοδο 0 (Output 0) στην έξοδο 9 (Output 9) και ξανά από την αρχή. Συνδέοντας κατάλληλα το παραπάνω ολοκληρωμένο κύκλωμα μπορούμε να έχουμε κυκλικό μετρητή δακτυλίου ή ένα μετρητή Johnson με mod της αρεσκείας μας. Στην παρακάτω συνδεσμολογία του ολοκληρωμένου CD 4017 φαίνεται η λειτουργία του μετρητή 10 καταστάσεων. Η λειτουργία απεικονίζεται με την σταδιακή ενεργοποίηση των LEDs. Βλέπουμε την τροφοδοσία των 5 Volts DC όπως επίσης και τη γείωση. Βλέπουμε ότι τα LED ενδείκτες των 10 καταστάσεων συνδέονται σε ένα φορτίο 470 Ω όπως επίσης και το PIN κρατουμένου εξόδου.



Ανάλυση Ακολουθιακών Κυκλωμάτων

Σε γενικές γραμμές η συμπεριφορά ενός ακολουθιακού κυκλώματος όπως έχουμε ήδη δει εξαρτάται από τις εισόδους του, τις εξόδους του και τις καταστάσεις των FLIP-FLOP που περιέχει. Η επόμενη κατάσταση και η εξέλιξη στο χρόνο ενός τέτοιου κυκλώματος και των εξόδων του είναι συναρτήσεις της παρούσας κατάστασης και των εισόδων. Εάν ένα λογικό διάγραμμα περιέχει FLIP-FLOP, τότε ξέρουμε ότι παριστάνει ένα ακολουθιακό κύκλωμα αναγκαστικά. Σε αυτή τη περίπτωση και γενικότερα FLIP-FLOP μπορούν να είναι οιουδήποτε τύπου και το κύκλωμα μπορεί να περιέχει ή να μην περιέχει συνδυαστικές πύλες ή άλλα κυκλώματα.

Η ανάλυση των ακολουθιακών κυκλωμάτων συνίσταται στον ακριβή εντοπισμό ή στην εύρεση ενός πίνακα ή διαγράμματος για τη χρονική ακολουθία των εισόδων, εξόδων και εσωτερικών καταστάσεων. Επίσης μπορούν να διατυπωθούν διάφορες συναρτήσεις Boole για την περιγραφή των ακολουθιακών κυκλωμάτων. Στις εκφράσεις αυτές όμως θα πρέπει να υπάρχει ως παράμετρος ο χρόνος είναι σημαντικό. Οι μέθοδοι περιγραφής ενός ακολουθιακού κυκλώματος με ρολόι, είναι με:

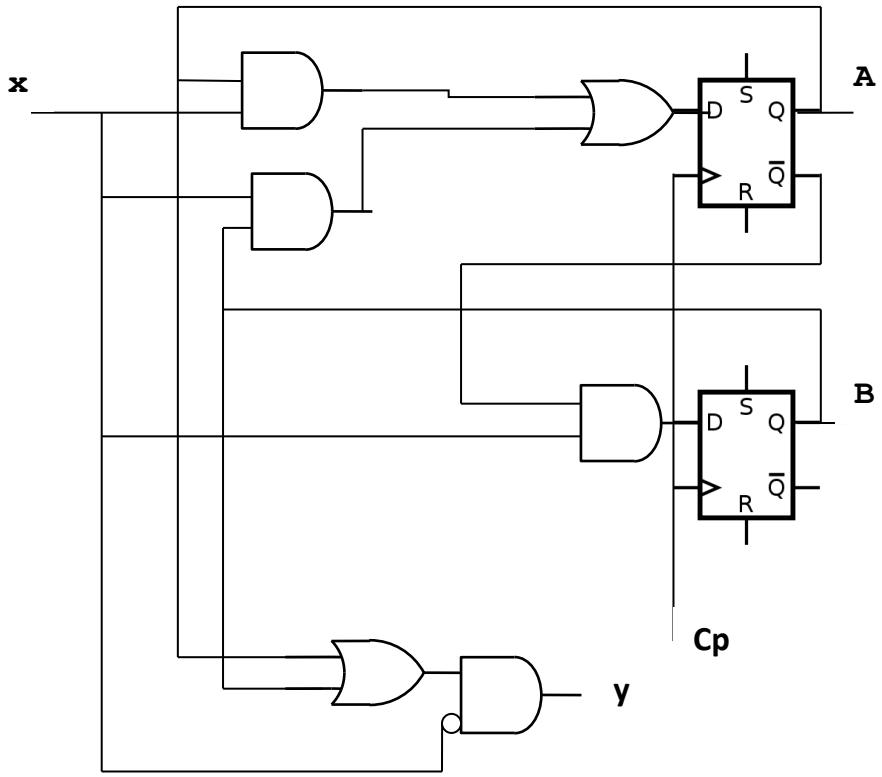
- Εξισώσεις κατάστασης ή αλγεβρική αναπαράσταση
- Πίνακας καταστάσεων
- Διάγραμμα καταστάσεων

Θα παρουσιάσουμε ένα συγκεκριμένο παράδειγμα παρακάτω με ένα ακολουθιακό κύκλωμα με ρολόι και μετά θα προσπαθήσουμε να προσδιορίσουμε τις διάφορες μεθόδους περιγραφής της συμπεριφοράς των ακολουθιακών κυκλωμάτων.

Το κύκλωμα που θα αναλύσουμε φαίνεται παρακάτω. Το κύκλωμα περιέχει:

- Δύο FLIP-FLOP τύπου D το A και τοB
- Μία είσοδο x
- Μία έξοδο y
- Και ένα συνδυαστικό κομμάτι με πύλες

Η εξίσωση κατάστασης είναι μια αναλυτική σχέση που καθορίζει τις συνθήκες μεταβολής της κατάστασης ενός FLIP-FLOP. Το πρώτο μέρος της σχέσης ή η αριστερή πλευρά της εξίσωσης είναι η επόμενη κατάσταση του Flip-Flop και η δεξιά πλευρά δίνει αυτή την επόμενη κατάσταση σαν συνάρτηση της παρούσας κατάστασης και των εισόδων του κυκλώματος. Μια παρόμοια εξίσωση καταστάσεων μοιάζει με τη χαρακτηριστική εξίσωση ενός Flip-Flop που έχουμε ήδη αναλύσει , με μόνη διαφορά που μας δίνει την επόμενη κατάσταση του Flip-Flop σαν συνάρτηση των εξωτερικών μεταβλητών εισόδου και των τιμών των άλλων Flip-Flops.



Η είσοδος του Flip-Flop D είναι αυτή που καθορίζει την επόμενη κατάσταση της εξόδου του οπότε θα έχουμε τις εξισώσεις της επόμενης κατάστασης όπως:

- $A(t+1) = A(t)x(t) + B(t)x(t)$
- $B(t+1) = \overline{A(t)}x(t)$
- $y(t) = (A(t) + B(t))\overline{x(t)}$

Επειδή όλες οι μεταβλητές στις εξισώσεις κατάστασης, είναι συνάρτηση της παρούσας κατάστασης, μπορούμε να παραλείψουμε την ένδειξη (t) η οποία αποκλειστικά αυτό δηλώνει. Έτσι οι εξισώσεις κατάστασης του κυκλώματος μετασχηματίζονται απλοποιούνται και γράφονται:

- $A(t+1) = Ax + Bx$
- $B(t+1) = \overline{A}x$
- $y = (A+B)\overline{x}$

Οι παραπάνω χρονικές ακολουθίες εισόδων, και μεταβάσεις εξόδων και καταστάσεων των Flip-Flop, καταγράφονται σε ένα πίνακα καταστάσεων. Ο πίνακας αυτός αποτελείται από τέσσερα διαφορετικά τμήματα τα οποία με τη σειρά τους δηλώνουν: παρούσα κατάσταση, είσοδος, επόμενη κατάσταση, έξοδος. Το κομμάτι παρούσα κατάσταση, δείχνει τις καταστάσεις των Flip-Flopς A και B, σε κάθε χρονική στιγμή t. Η είσοδος δείχνει την τιμή του x, για κάθε δυνατή παρούσα κατάσταση. Η επόμενη κατάσταση δείχνει τις καταστάσεις των Flip-Flop A και B, στη χρονική στιγμή t+1. Η έξοδος δείχνει την τιμή του y για κάθε παρούσα κατάσταση.

Παρούσα κατάσταση		**Είσοδος**	**Επόμενη κατάσταση**		**Έξοδος**
A	**B**	**x**	**A**	**B**	**y**
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	0	0
1	1	0	0	0	1
1	1	1	1	0	0

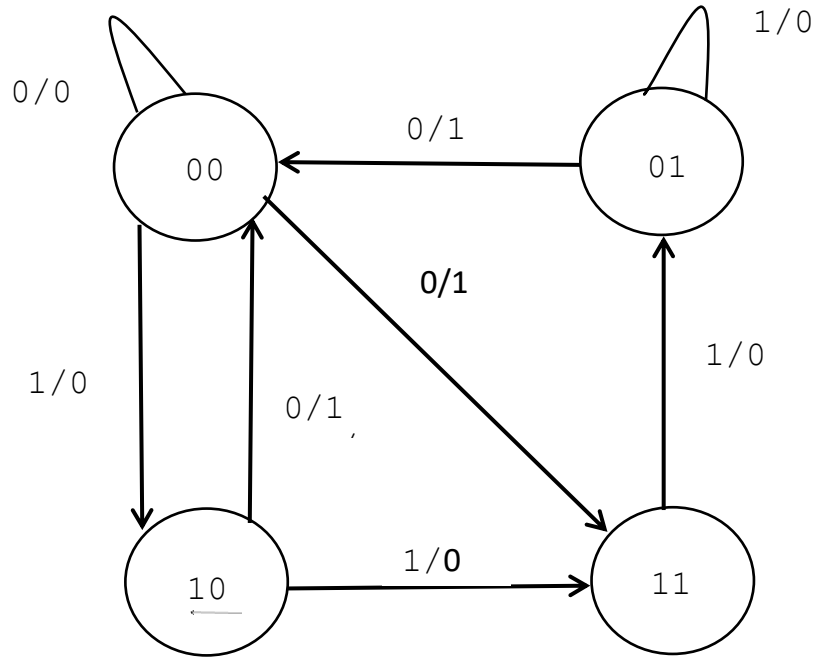
Για να σχηματίσουμε τον παραπάνω πίνακα πρέπει να απαριθμήσουμε όλους τους δυνατούς συνδυασμούς παρούσας κατάστασης και εισόδων. Οι

επόμενες καταστάσεις των εξόδων των Flip-Flops πρέπει να ικανοποιούν τις εξισώσεις που βρήκαμε προηγουμένως όπως επίσης και η έξοδος y.

- $A(t+1) = Ax + Bx$
- $B(t+1) = \overline{A}x$
- $y = (A+B)\overline{x}$

Ένα ακολουθιακό κύκλωμα με n Flip-Flops τύπου D και k εισόδους, χρειάζεται 2^{n+k} γραμμές στον πίνακα καταστάσεων.

Το διάγραμμα καταστάσεων (state diagram) αποτελεί μια γενική σχηματική αναπαράσταση της λειτουργίας του κυκλώματος και είναι εύκολο στην κατανόηση της λειτουργίας του κυκλώματος. Ο πίνακας μπορεί να αντληθεί από το λογικό διάγραμμα και το διάγραμμα καταστάσεων βγαίνει επίσης άμεσα από τον πίνακα. Το διάγραμμα καταστάσεων χρησιμοποιείται πολύ συχνά σαν την αρχική προδιαγραφή που δίνεται για τη σχεδίαση ενός ακολουθιακού κυκλώματος.



Οι καταστάσεις αναπαριστάνονται με κύκλους και οι μεταβάσεις από τη μία κατάσταση στην άλλη με βέλη που συνδέουν τους κύκλους. Τα βέλη έχουν επάνω τους δύο δυαδικούς αριθμούς που χωρίζονται μεταξύ τους με μία κάθετο /. Οι δύο αυτοί αριθμοί δηλώνουν είσοδο και έξοδο ειδικότερα ο

πρώτος αριθμός είναι η τιμή των εισόδων που προκαλεί αυτή τη μετάβαση καταστάσεων και ο δεύτερος αριθμός δίνει την τιμή των εξόδων κατά τη διάρκεια της παρούσας κατάστασης και για την τιμή των εισόδων που αντιστοιχεί στο βέλος αυτό.

ΑΣΚΗΣΕΙΣ

1. Σχεδιάστε ένα μετρητή με Flip-Flop τύπου JK ικανό να μετράει 000, 001, 011, 101, 111 και ξανά από την αρχή.

2. Σχεδιάστε ένα τετράμπιτο μετρητή με Flip-Flop τύπου T.

3. Ένα Flip-Flop τύπου **JK'** είναι ένα JK όπου έχω παρεμβάλει ένα αντιστροφέα μεταξύ της εξωτερικής εισόδου K' και της εσωτερικής K.

α. Βρείτε το χαρακτηριστικό πίνακα του Flip-Flop.

β. Βρείτε τη χαρακτηριστική του εξίσωση.

γ. Δείξετε πως αν ενώσουμε τις δύο εισόδους του θα σχηματίσουμε ένα Flip-Flop τύπου D.

Κεφάλαιο 11. Μνήμες Ημιαγωγών

Η μνήμη ενός ψηφιακού συστήματος είναι ο φυσικός χώρος αποθήκευσης στοιχείων και διαδικασιών όπου δηλαδή αποθηκεύονται δεδομένα και προγράμματα, ότι δηλαδή χρειάζεται το ψηφιακό σύστημα για να λειτουργήσει. Αποτελείται από ένα σύνολο καταχωρητών έκαστος των οποίων αποθηκεύει μία δυαδική πληροφορία. Οι μνήμες των ψηφιακών συστημάτων χωρίζονται σε διάφορες κατηγορίες όπως θα δούμε παρακάτω με βασικά κριτήρια διαχωρισμού τον τρόπο εγγραφής, τον τρόπο ανάγνωσης και τέλος τη βασική ηλεκτρονική διάταξη πάνω στην οποία στηρίζεται η κατασκευή της μνήμης.

Σημαντικό πλεονέκτημα των ψηφιακών συστημάτων-κυκλωμάτων σε σύγκριση με τα αναλογικά είναι η πολύ μεγάλη ικανότητα που έχουν τα πρώτα να αποθηκεύουν μεγάλες ποσότητες ψηφιακών δεδομένων. Το στοιχειώδες κύτταρο αποθήκευσης δυαδικής πληροφορίας όπως έχουμε ήδη δει είναι ο δισταθής πολυδονητής, κύκλωμα δύο σταθερών καταστάσεων ή Flip-Flop. Το FLIP-FLOP μπορεί να αποθηκεύσει δύο διαφορετικά BIT πληροφορίας το 0 και το 1. Συνδυάζοντας FLIP-FLOP σχηματίζουμε ένα καταχωρητή ο οποίος μπορεί να αποθηκεύσει μία ψηφιακή λέξη, το μήκος της οποίας εξαρτάται από τον αριθμό των FLIP-FLOP που αποτελούν τον καταχωρητή αφού κάθε FLIP-FLOP αποτελεί ένα και μόνο ένα κύτταρο μνήμης.

Με την εξέλιξη της Μικροηλεκτρονικής και την σύνθεση όλων και ποιο περίπλοκων ολοκληρωμένων κυκλωμάτων έγινε εφικτή μέσα από τη τεχνολογία LSI, VLSI και UVLSI η κατασκευή ολοκληρωμένων που περιέχουν μεγάλο αριθμό καταχωρητών και επομένως διαθέτουν μεγάλη χωρητικότητα αποθήκευσης ψηφιακής πληροφορίας. Οι μνήμες ημιαγωγών είναι σήμερα οι ταχύτερες και φθηνότερες μορφές μνήμης, δηλαδή τα απαραίτητα εργαλεία καταγραφής και διατήρησης ψηφιακών δεδομένων. Ένα σύγχρονο υπολογιστικό σύστημα για να λειτουργήσει χρειάζεται αποθηκευμένα προγράμματα και δεδομένα. Αυτά αποθηκεύονται στη μνήμη του ηλεκτρονικού υπολογιστή η οποία αποτελείται από δύο διακριτά κομμάτια που είναι η μνήμη ROM (Read Only Memory) και η μνήμη RAM (Random Access Memory).

ΟΡΟΛΟΓΙΑ ΓΙΑ ΤΙΣ ΜΝΗΜΕΣ

Στοιχειώδες κύτταρο

Το στοιχειώδες κύτταρο είναι ένα ηλεκτρικό κύκλωμα η μία διάταξη με δυνατότητα αποθήκευσης ενός bit 0 η 1.

Λέξη μνήμης

Η λέξη μνήμης είναι ένα σύνολο από στοιχειώδη κύτταρα αποθήκευσης. Για παράδειγμα μπορούμε να θεωρήσουμε ένα καταχωρητή 8 bits ο οποίος μπορεί να αποθηκεύσει λέξεις των 8 bits. Το μήκος των λέξεων μέσα στα διάφορα ψηφιακά συστήματα εγγραφής δεδομένων κυμαίνεται από 4 μέχρι 128 bits.

Χωρητικότητα

Η μνήμη χρησιμεύει στην αποθήκευση δυαδικών ψηφίων οργανωμένων σε ομάδες (συνήθως οκτάδες) οι οποίες λέγονται κύτταρα και συνήθως σε κάθε κύτταρο αποθηκεύεται ένα byte. Η χωρητικότητα της μνήμης είναι αποφασιστικός παράγοντας για την ταχύτητα εκτέλεσης υπολογισμών από ένα ηλεκτρονικό υπολογιστικό σύστημα. Η χωρητικότητα της μνήμης, καθορίζει επίσης και το λογισμικό που μπορεί να χρησιμοποιηθεί από ένα υπολογιστικό σύστημα. Τα υπολογιστικά συστήματα ξεκίνησαν με πολύ μικρές χωρητικότητες (από 12 bytes (!) έως 3 Kbytes). Από γενιά σε γενιά ηλεκτρονικών υπολογιστικών συστημάτων και καθώς η τεχνολογία και η μικροηλεκτρονική έκαναν γιγαντιαία άλματα, η μνήμη αυξανόταν διαρκώς σε μέγεθος. Σήμερα στη δεκαετία του 2020 η μνήμες σε μεγάλα υπολογιστικά συστήματα ή υπερυπολογιστές διαθέτουν χωρητικότητες αρκετών TeraBytes, ενώ στα μικρότερα υπολογιστικά συστήματα ανέρχεται σε κάποια Gbytes (από 1 έως 48 Gbytes, ανάλογα με το σύστημα). Καθοριστικό, επίσης, είναι το μέγεθος της χωρητικότητας της μνήμης για τη σταθερότητα και την ομαλή λειτουργία ενός υπολογιστικού συστήματος. Ας υποθέσουμε ότι έχουμε για παράδειγμα μία μνήμη με 4096 λέξεων των 20 bits. Η συνολική χωρητικότητα της μνήμης είναι 81920 bits. Ο αριθμός των bits σε μία μνήμη είναι συνήθως ένα πολλαπλάσιο του 1024 το οποίο ονομάζουμε 1 "K". Η χωρητικότητα της μνήμης RAM μετράται σε Kbytes,

Mbytes και GBytes. Η χωρητικότητα γενικά μιας μνήμης είναι το σύνολο των bits που μπορεί να αποθηκεύσει.

Διεύθυνση

Η διεύθυνση είναι ένας αριθμός ετικέτα ο οποίος μας επιτρέπει να εντοπίσουμε τη θέση στην οποία έχει γίνει καταχώρηση ή εγγραφή μίας δυαδικής πληροφορίας στην μνήμη και όπως καταλαβαίνουμε είναι μοναδική. Την πληροφορία αυτή τη χρειαζόμαστε επίσης για την ανάκτηση της εγγραφής.

Χρόνος προσπέλασης

Είναι ο αναγκαίος χρόνος για να διαβάσουμε μία πληροφορία από τη μνήμη από τη χρονική στιγμή που δέχεται το σήμα ελέγχου η μνήμη για να ξεκινήσει αυτή η διαδικασία. Αναλυτικότερα είναι ο χρόνος ανάμεσα στη στιγμή εκκίνησης ενός αιτήματος για ανάκτηση ενός byte ή λέξη από τη μνήμη, μέχρι αυτό να προσκομιστεί πραγματικά στον επεξεργαστή και να αποθηκευθεί σε κάποιον καταχωρητή του. Το χρονικό αυτό διάστημα ονομάζεται υστέρηση ή χρόνος προσπέλασης. Ο χρόνος αυτός μπορεί να δοθεί ως το διάστημα από τη στιγμή που ζητείται μια διεύθυνση στη μνήμη μέχρι τη στιγμή που τα αντίστοιχα δεδομένα θα είναι διαθέσιμα για χρήση. Αποτελεί θεμελιώδες μέτρο ταχύτητας της μνήμης: όσο μικρότερη η υστέρηση τόσο μεγαλύτερη η ταχύτητα της μνήμης. Η υστέρηση δεν θα πρέπει να συγχέεται με το εύρος μνήμης, το μέγεθος του διαύλου (bus) της RAM σε bit, με το οποίο μετράται η διαμεταγωγή της μνήμης ή η μεταφορά της συγκεκριμένης εγγραφής. Είναι πιθανό μια προηγμένη τεχνολογία μνήμης να έχει αυξημένο εύρος μνήμης αλλά, παράλληλα, και αυξημένο χρόνο προσπέλασης. Για παράδειγμα η μνήμη DDR, η εμφάνιση της οποίας στην αγορά προηγείται χρονικά της μνήμης DDR2, έχει μικρότερη υστέρηση αν και πρόκειται για παλαιότερη τεχνολογία.

Τα διπολικά τρανζίστορ είναι πιο γρήγορες διατάξεις από τα MOSFETs και κατά συνέπεια οι διπολικές μνήμες είναι πιο γρήγορες από τις MOS. Έτσι το τσιπ 7636 το οποίο είναι μία διπολική PROM έχει access time 80ns και το 2716 MOS EPROM 450 ns αντίστοιχα. Η ταχύτητα μίας μνήμης πληρώνετε και κατά συνέπεια οι διπολικές μνήμες έχουν υψηλότερο κόστος από τις MOS. Όλες οι μνήμες που είδαμε παραπάνω (ROMs, PROMs, EPROM’s, και

EEPROMs) είναι non-volatile memories, δηλαδή ακόμα και χωρίς τροφοδοσία διατηρούν το περιεχόμενο τους.

Κόστος

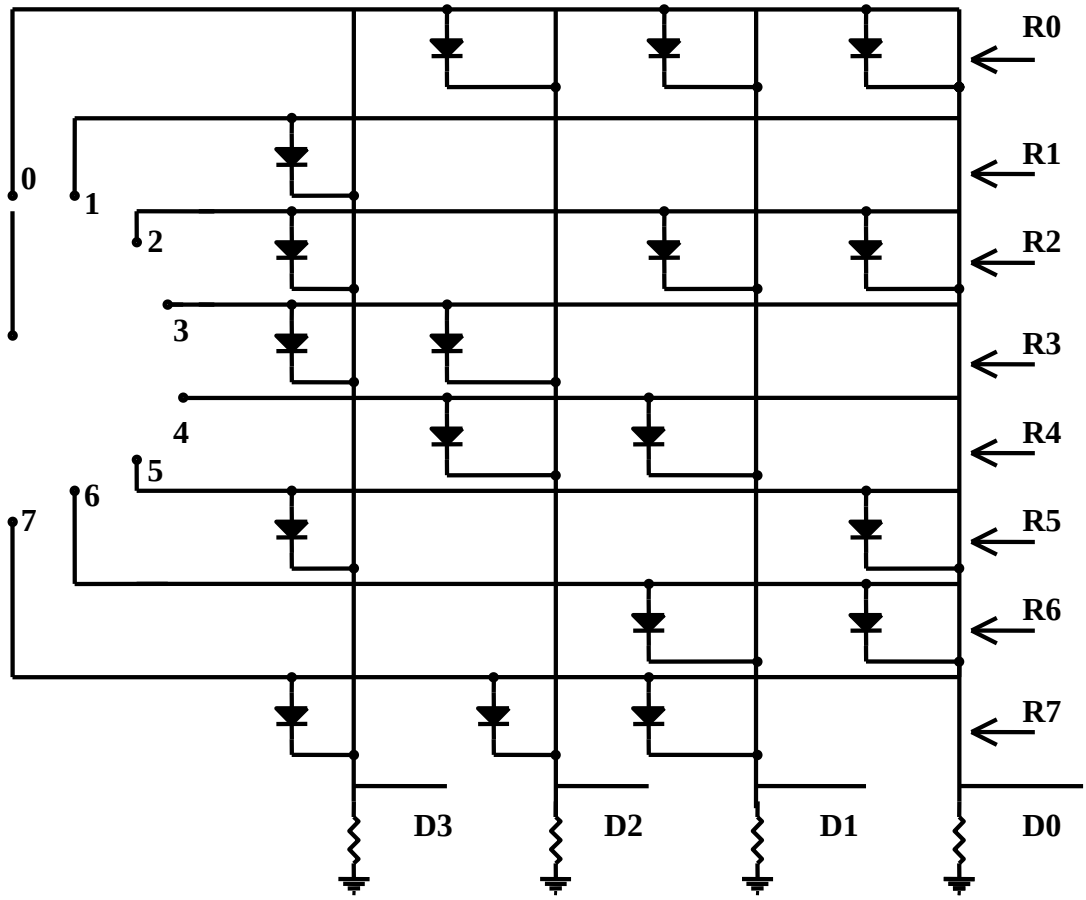
Το κόστος της μνήμης RAM είναι ευμετάβλητο και εξαρτάται από την αγορά μητρικών πλακετών. Μνήμες οι οποίες πριν μερικούς μήνες είχαν πολύ χαμηλές τιμές σήμερα μπορεί να στοιχίζουν πανάκριβα λόγω στάσης της παραγωγής τους και μνήμες οι οποίες στοίχιζαν ακριβά μπορεί σήμερα να στοιχίζουν πολύ φθηνά επειδή μπορεί χρησιμοποιούνται ευρέως και υπάρχει μεγάλη παραγωγή. Πολλές φορές η διαφοροποίηση της τιμής συναρτάται με τις διαστάσεις ενός μεμονωμένου αρθρώματος. Πολλές φορές αρθρώματα διπλάσιας μνήμης στοιχίζουν λιγότερο από 2 ίδια με τη μισή μνήμη και σε άλλες περιπτώσεις στοιχίζουν περισσότερο.

Μνήμη ROM

Η μνήμη ROM (Read Only Memory) είναι η πιο απλή κατηγορία μνήμης. Αποτελείται από ένα σύνολο καταχωρητών. Κάθε ένας έχει μία χαρακτηριστική διεύθυνση και περιέχει μία λέξη. Αν κάνουμε μια ιστορική αναδρομή θα δούμε ότι οι περισσότεροι προσωπικοί υπολογιστές στη δεκαετία του '80 χρησιμοποιούσαν τη μνήμη ROM, για μόνιμη αποθήκευση του λειτουργικού τους συστήματος. Αργότερα το μέγεθος του λειτουργικού συστήματος μεγάλωσε σημαντικά και έτσι προτιμήθηκε από τους κατασκευαστές υπολογιστών, λόγω υψηλού κόστους, το λειτουργικό να φορτώνεται από ένα μαγνητικό μέσο, με τη βοήθεια μιας στοιχειώδους αρχικοποίησης από ένα λογισμικό που εγγράφεται στο BIOS. Παλαιότερα το BIOS αποθηκευόταν σε μια μνήμη EPROM ενώ τώρα αποθηκεύεται σε μνήμη τύπου Flash ROM. Υπάρχουν παρόμοιες μνήμες με τη ROM όσον αφορά κυρίως στη μη πτητικότητα τους όπως οι :

- PROM (Programmable Read Only Memory)
- EPROM (Erasable Programmable Read Only Memory)
- EEPROM (Electrically Erasable Programmable Read-Only Memory)
- Flash ROM

Μνήμες ROM διόδων. Το σχήμα παρακάτω δείχνει μία ROM με διόδους. Κάθε οριζόντια γραμμή είναι ένας καταχωρητής.



Μνήμη ROM με διόδους

Κάθε οριζόντια γραμμή είναι ένας καταχωρητής. Έτσι ο **R_0 π**εριέχει τρεις διόδους ο **R_1** μία κλπ. Η έξοδος της ROM είναι η λέξη **$D_3D_2D_1D_0$**. Όταν ο διακόπτης είναι στην θέση 0 όλες οι δίοδοι της **R_0** βλέπουν 5V και οδηγούν ενώ όλες οι άλλες είναι μπλοκαρισμένες. Έτσι αν ο διακόπτης είναι στην θέση 0 τότε η έξοδος είναι στο 0111. Αν γυρίσει ο διακόπτης στην θέση 1 τότε η έξοδος θα βλέπει το δυαδικό αριθμό 1000. Ανάλογα με τη θέση του διακόπτη στον παρακάτω πίνακα βλέπουμε το περιεχόμενο των καταχωρητών της παραπάνω ROM.

Καταχωρητής	Διεύθυνση	Περιεχόμενο
R0	0	0111
R1	1	1000

R2	2	1011
R3	3	1100
R4	4	0110
R5	5	1001
R6	6	0011
R7	7	1110

Διευθύνσεις και περιεχόμενα καταχωρητών

Η διεύθυνση και το περιεχόμενο ενός καταχώρηση είναι δύο εντελώς διαφορετικά πράγματα μεταξύ τους όπως είδαμε στο προηγούμενο παράδειγμα. Έτσι ο αριθμός των καταχωρητών μίας ROM είναι ο ίδιος με τον αριθμό των λέξεων που περιέχει. Κατασκευαστικά η επιλογή μιας λέξης μίας συγκεκριμένης διεύθυνσης γίνεται πάνω στο ίδιο τσιπ με αποκωδικοποιητή. Έτσι με ένα αριθμό εισόδων n του αποκωδικοποιητή μπορώ να έχω 2^n λέξεις στην μνήμη.

Γενικά η δομή μιας μνήμης ROM έχει n γραμμές εισόδου οι οποίες ορίζουν την διεύθυνση όπου είναι καταχωρημένη η πληροφορία και m γραμμές στην έξοδο για την μεταφορά αυτής της πληροφορίας. Στις n γραμμές εισόδου αντιστοιχούν 2^n διαφορετικοί συνδυασμοί , άρα μπορούμε να έχουμε 2^n διαφορετικές διευθύνσεις, δηλαδή 2^n λέξεις των m bit.

Αν για παράδειγμα για μια μνήμη ROM έχουμε το n=5 και το m=8 η εν λόγω μνήμη θα περιέχει $2^5 = 32$ λέξεις τω 8 bit. Με 5 bit στον καταχωρητή διεύθυνσης αριθμούμε από το 00000 μέχρι το 11111. Έτσι η διεύθυνση της $1^{ης}$ εγγραφής θα είναι η 00000, της $16^{ης}$ θα είναι 1111 και της $32^{ης}$ 11111.

Μια μνήμη με χωρητικότητα $2^n.m$ έχει 15 γραμμές εισόδου και 8 γραμμές εξόδου $2^{15}x8 = 32768x8 = 262144$ bit ή περιέχει 32768 λέξεις τω 8 bit.

PROMs και EPROMs

Ο κατασκευαστής μίας μνήμης ROM χρειάζεται για την κατασκευή της όλα τα στοιχεία που θα αποθηκεύσει. Με αυτό τον τρόπο ετοιμάζει όλη τη σχεδιαστική και φωτολιθογραφική διαδικασία και στην συνέχεια προχωρεί στην υλοποίηση της μνήμης ROM. Μπορεί να χρησιμοποιεί διπολικά τρανζίστορ η MOSFETs. Η βασική λειτουργία μίας μνήμης δεν αλλάζει και οι παραπάνω ηλεκτρονικές διατάξεις λειτουργούν σαν δίοδοι. Αντίθετα με μία ROM η οποία προγραμματίζεται στην κατασκευή της και δεν μπορεί ποτέ να αλλάξει το περιεχόμενο της μόνο αν καταστραφεί μία PROM (Programmable Read Only Memory) μπορεί να προγραμματιστεί με ένα ειδικό όργανο το οποίο ονομάζουμε προγραμματιστή PROM. Όταν προγραμματιστεί μία PROM είναι σαν μία ROM δεν μπορούμε δηλαδή πλέον να επέμβουμε και να αλλάξουμε το περιεχόμενο της με κανένα τρόπο.

EPROM

Μία **EPROM** (**Erasable Programmable Read Only Memory**) μπορεί να χρησιμοποιήσει τον προγραμματιστή PROM χωρίς ποτέ το περιεχόμενο της να γίνει μόνιμο για πάντα. Είναι δυνατόν με κατάλληλο φωτισμό με υπεριώδη ακτινοβολία από ένα παράθυρο που αφήνει ο κατασκευαστής να σβήσει και να ξαναπρογραμματισθεί πάλι από την αρχή.

Οι EEPROMs μπορούν να σβήσουν το περιεχόμενο τους με ηλεκτρικό τρόπο και να ξαναπρογραμματισθούν. Οι μνήμες αυτού του τύπου είναι πολύ χρήσιμες όταν πρόκειται να προγραμματίσουμε ένα ψηφιακό σύστημα γιατί μας δίνει τη δυνατότητα για άπειρες πρακτικά δοκιμές μέχρι την τελική εξέλιξη του συστήματος μας.

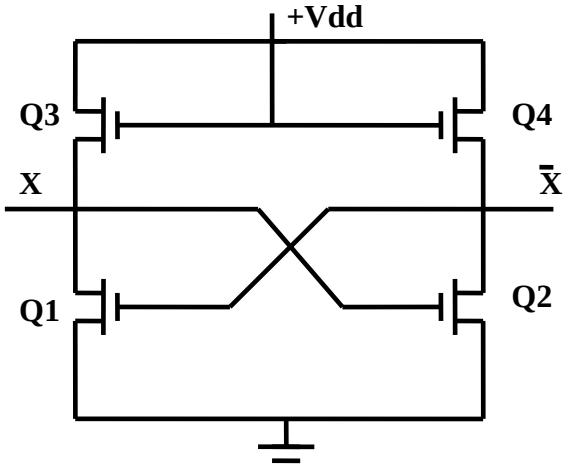
Τα διπολικά τρανζίστορ είναι πιο γρήγορες διατάξεις από τα MOSFETs και κατά συνέπεια οι διπολικές μνήμες είναι πιο γρήγορες από τις MOS. Έτσι το τσιπ 7636 το οποίο είναι μία διπολική PROM έχει access time 80ns και το 2716 MOS EPROM 450 ns αντίστοιχα. Η ταχύτητα μίας μνήμης πληρώνετε και κατά συνέπεια οι διπολικές μνήμες έχουν υψηλότερο κόστος από τις MOS. Όλες οι μνήμες που είδαμε παραπάνω (ROMs, PROMs, EPROM’s, και EEPROMs) είναι non-volatile memories, δηλαδή ακόμα και χωρίς τροφοδοσία διατηρούν το περιεχόμενο τους.

ΜΝΗΜΕΣ RAM

Η μνήμη τυχαίας προσπέλασης RAM (Random Access memory) είναι ο όρος που χρησιμοποιούμε για ηλεκτρονικές διατάξεις προσωρινής αποθήκευσης ψηφιακών δεδομένων, οι οποίες επιτρέπουν πρόσβαση στα αποθηκευμένα δεδομένα στον ίδιο χρόνο οπουδήποτε και αν βρίσκονται αυτά, δηλαδή με «τυχαία πρόσβαση». Σε αντιδιαστολή βρίσκονται συσκευές αποθήκευσης δεδομένων, όπως οι μαγνητικές ταινίες, οι μαγνητικοί δίσκοι («σκληροί» ή «εύκαμπτοι»), στα οποία η πρόσβαση και η ανάκτηση δεδομένων μπορεί να γίνει μόνο με κάποιον προκαθορισμένο τρόπο, συνήθως σειριακά, λόγω του τρόπου κατασκευής τους. Υπάρχουν δύο βασικοί τύποι RAM: η δυναμική RAM (DRAM) και η στατική RAM (SRAM). Η DRAM είναι η πιο κοινή μορφή αλλά πρέπει να «ανανεώνεται» (refresh) χιλιάδες φορές ανά δευτερόλεπτο, ενώ η SRAM δεν χρειάζεται κάτι τέτοιο. Η SRAM, ως διάταξη, είναι πιο δαπανηρή στην κατασκευή της, και επομένως και στην αγορά της σε σχέση με την DRA

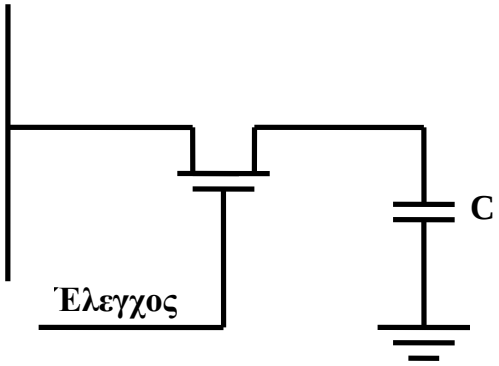
Στην πληροφορική με τον όρο RAM αναφερόμαστε στην κύρια ή κεντρική μνήμη ενός υπολογιστικού συστήματος, δηλαδή τη μνήμη στην οποία αποθηκεύονται προγράμματα και δεδομένα, προκειμένου είτε να εκτελεστούν είτε να υποστούν επεξεργασία αντίστοιχα. Τμήμα, επίσης, της κεντρικής μνήμης είναι και η μνήμη μόνο ανάγνωσης (ROM), η οποία επίσης επιτρέπει την τυχαία προσπέλαση. Η βασική διαφορά των δύο τύπων μνήμης είναι ότι η μεν RAM διατηρεί τα περιεχόμενά της μόνο όσο της επιτρέπει ο χρήστης ή το λογισμικό που εκτελείται και μόνο εφόσον το υπολογιστικό σύστημα τροφοδοτείται με ηλεκτρική ενέργεια. Σε αντίθετη περίπτωση, τα περιεχόμενά της είτε αντικαθίστανται από άλλα είτε χάνονται ολοσχερώς. Αν οι όροι που έχουν επικρατήσει ήταν απόλυτα ακριβείς, η μνήμη RAM έπρεπε να ονομάζεται ως «μνήμη τυχαίας προσπέλασης εγγραφής / ανάγνωσης», ενώ η μνήμη ROM ως «μνήμη τυχαίας προσπέλασης μόνο ανάγνωσης». Η μνήμη ROM έχει προεγγεγραμμένο περιεχόμενο, πάντα από τον κατασκευαστή του συστήματος, και χρησιμεύει, συνήθως, για την εκκίνηση λειτουργίας του συστήματος (BIOS), μόλις αυτό αρχίσει να τροφοδοτείται με ρεύμα, οπότε και η μνήμη RAM είναι κενή από περιεχόμενο.

Οι στατικές RAM χρησιμοποιούν διπολικά η MOS Flip-Flop (σχήμα 302) και κρατούν την πληροφορία όσο τροφοδοτούνται.



Flip-Flop για στατική RAM.

Οι δυναμικές RAM χρησιμοποιούν MOSFETs και χωρητικότητες για την εγγραφή μίας πληροφορίας όπως φαίνεται στο παρακάτω σχήμα.



Κύτταρο δυναμικής RAM

Επειδή υπάρχουν απώλειες λόγω των ρευμάτων διαρροής στις πύλες των τρανζίστορ πεδίου η πληροφορία ελέγχεται κάθε μερικά milliseconds και η διαδικασία αυτή ονομάζεται "refresh" της μνήμης. Η οργάνωση μίας RAM ICs γίνεται με bit και με λέξεις. Όταν έχουμε οργάνωση κατά bit σε κάθε προσπέλαση της μνήμης έχουμε το δικαίωμα εγγραφής η ανάγνωσης ενός μόνο bit. Όταν έχουμε οργάνωση κατά λέξεις σε κάθε προσπέλαση έχουμε δικαίωμα εγγραφής η ανάγνωσης μίας μόνο λέξης m bits. Και στις δύο περιπτώσεις η προσπέλαση είναι δυνατή διαμέσου η κωδικοποιημένων

γραμμών διεύθυνσης για μία ολική χωρητικότητα της μνήμης $mx2^n$ =bits σύνολο (όταν έχουμε οργάνωση κατά bit m=1).

Διάφοροι τύποι Μνήμης RAM

- SRAM (Static Random Access Memory. Αυτές οι μνήμες ημιαγωγών χρησιμοποιούν πολλά τρανζίστορ, τυπικά από 4 έως 6, για κάθε κύτταρο μνήμης αλλά δεν χρησιμοποιεί πυκνωτή. Χρησιμοποιείται συνήθως για τη μνήμη cache, που είναι γνωστή και ως λανθάνουσα μνήμη.
- DRAM (Dynamic Random Access Memory). Περιέχει κύτταρα μνήμης με ζευγάρια τρανζίστορς και πυκνωτές τα οποία χρειάζονται συνεχείς ανανεώσεις (Refresh).
- FPM DRAM (Fast Page Mode Dynamic Random Access Memory). Ήταν η αρχική μορφή της DRAM. Περιμένει για ολόκληρη τη διαδικασία εντοπισμού ενός bit δεδομένων με στήλη (column) και σειρά (row) και μετά διαβάζει αυτό το bit πριν πάει στο επόμενο bit. Η μέγιστη ταχύτητα μεταφοράς (transfer rate) στη μνήμη L2 cache είναι περίπου 176 MBps.
- EDO DRAM (Extended Data-Out Dynamic Random Access Memory). Δεν περιμένει για ολόκληρη τη διαδικασία επεξεργασίας του πρώτου bit πριν συνεχίσει με το επόμενο. Μόλις γίνει εντοπισμός της διεύθυνσης του πρώτου bit, η EDO DRAM αρχίζει να ψάχνει για το επόμενο bit. Η ταχύτητα της είναι περίπου 5% μεγαλύτερη από αυτή της FPM. Η μέγιστη ταχύτητα μεταφοράς δεδομένων (transfer rate) στη μνήμη L2 cache είναι περίπου 264 MBps.
- SDRAM (Synchronous Dynamic Random Access Memory). Η SDRAM είναι περίπου 5% ταχύτερη από την EDO RAM και είναι η πιο κοινή μορφή μνήμης στους σημερινούς προσωπικούς υπολογιστές. Η μέγιστη ταχύτητα μεταφοράς δεδομένων (transfer rate) στη μνήμη L2 cache είναι περίπου 528 MBps.
- DDR SDRAM (Double Data Rate Synchronous Dynamic RAM). Είναι σαν την SDRAM με τη διαφορά ότι έχει μεγαλύτερο εύρος ζώνης (bandwidth), που σημαίνει μεγαλύτερη ταχύτητα. Η μέγιστη ταχύτητα

μεταφοράς (transfer rate) στη μνήμη L2 cache είναι περίπου 1.064 MBps (για DDR SDRAM στα 133 MHZ).

- RDRAM (Rambus Dynamic Random Access Memory). Αποτελεί ένα παράρτημα της προηγούμενης αρχιτεκτονικής της DRAM. Σχεδιασμένη από τον Rambus, η RDRAM χρησιμοποιεί ένα άρθρωμα RIMM (Rambus in-line memory module), το οποίο είναι παρόμοιο σε μέγεθος και διάταξη pin μ' ένα κλασικό DIMM. Αυτό που κάνει την RDRAM να ξεχωρίζει είναι ότι χρησιμοποιεί ένα ειδικό κανάλι δεδομένων (data bus) υψηλής ταχύτητας που αποκαλείται το κανάλι (channel) Rambus. Τα τσιπς της μνήμης RDRAM εργάζονται σε παράλληλη διάταξη για να επιτύχουν ρυθμό μετάδοσης δεδομένων στα 800 MHz ίσο με 1.600 MBps. Εφόσον λειτουργούν σε τόσο μεγάλες ταχύτητες, παράγεται πολύ περισσότερη θερμότητα (heat) από άλλα είδη τσιπς. Για να απορροφηθεί αυτή η επιπλέον θερμότητα, τα τσιπς Rambus είναι εξοπλισμένα μ' ένα ειδικό ψήκτη. Όπως υπάρχουν μικρότερες παραλλαγές των DIMMs, υπάρχουν επίσης και τα SO-RIMMs, που είναι σχεδιασμένα για φορητούς υπολογιστές (notebook computers).
- Credit Card Memory. Είναι ένας ειδικός τύπος μνήμης DRAM που συνδέεται σε μια ειδική θύρα (slot) για χρήση σε φορητούς υπολογιστές.
- PCMCIA Memory Card. Είναι ένας άλλος τύπος μνήμης DRAM για φορητούς υπολογιστές.
- CMOS RAM. Είναι ένας όρος για τη μικρή ποσότητα μνήμης που χρησιμοποιείται από τον υπολογιστή μας και από κάποιες άλλες συσκευές για να θυμάται πράγματα όπως είναι οι ρυθμίσεις του σκληρού δίσκου. Αυτή η μνήμη χρησιμοποιεί μια μικρή μπαταρία ώστε να έχει την απαραίτητη ισχύ για να μπορεί να κρατήσει τα περιεχόμενα της μνήμης.
- VRAM (VideoRAM). Είναι γνωστή και ως multiport dynamic random access memory (MPDRAM), είναι ένας τύπος RAM που χρησιμοποιείται ειδικά για video adapters ή επιταχυντές (accelerators) 3-D. Ο όρος "multiport" προέρχεται από το γεγονός ότι η VRAM κανονικά έχει δύο ανεξάρτητες θύρες πρόσβασης (access ports) αντί για μια, κάτι που

επιτρέπει στην CPU και στον επεξεργαστή γραφικών (graphics processor) να έχουν πρόσβαση στην RAM ταυτόχρονα. Η VRAM βρίσκεται στην κάρτα γραφικών (graphics card) και έχει πολλά χαρακτηριστικά. Η ποσότητα της VRAM είναι ένας καθοριστικός παράγοντας για την ανάλυση (resolution) και το βάθος χρώματος (color depth) της εμφάνισης. Η VRAM χρησιμοποιείται επίσης για να διατηρήσει πληροφορίες σχετικά με τα γραφικά, όπως είναι η γεωμετρία των δεδομένων 3-D και οι χάρτες υφής (texture maps). Επειδή οι αληθινές multiport VRAM είναι ακριβές, σήμερα πολλές κάρτες γραφικών χρησιμοποιούν την SGRAM (synchronous graphics RAM). Ενώ η απόδοση είναι στην ουσία η ίδια, η SGRAM είναι φθηνότερη.

Μνήμες μαγνητικών φυσαλίδων

Στην φύση υπάρχουν σπάνια σύνθετα υλικά με απίθανες ιδιότητες. Μια από αυτές τις κατηγορίες υλικών ονομάζονται γαρνήτες (garnets στα αγγλικά). Μέρος αυτών των υλικών παρουσιάζει ένα φαινόμενο το οποίο ονομάζεται σιδηριμαγνητισμός. Τέτοια υλικά διαθέτουν άτομα στο κρυσταλλικό τους πλέγμα με ροπές μαγνητικές αντίθετες σε αντίθεση με τα σιδηρομαγνητικά. Υλικά τέτοιου τύπου χρησιμοποιούνται για να κατασκευαστούν μνήμες. Το λογικό 1 η το λογικό 0 αντιπροσωπεύεται από τη διεύθυνση του μαγνητικού πεδίου που έχουν πολύ μικρές περιοχές μαγνητικές περιοχές με το όνομα περιοχές Weiss μέσα στο υλικό ανάλογα με τη διαμόρφωση που έχουν υποστεί λόγω του εξωτερικού πεδίου. Τα παραπάνω υλικά βάσης έχουν πολύ μεγάλη σταθερότητα. Κατά συνέπεια οι μνήμες μαγνητικών φυσαλίδων είναι σχετικά αργές έχουν όμως πολύ μεγάλη αξιοπιστία και μπορούν να χρησιμοποιηθούν χωρίς κανένα πρόβλημα κάτω από δύσκολες γενικά συνθήκες. Δεν αλλοιώνεται το περιεχόμενο τους ακόμα και αν δεχθούν πολύ μεγάλες δόσεις ραδιενέργειας και γ’ αυτό τον λόγο η χρήση τους ήταν μυστική και πού περιορισμένη αποκλειστικά σε στρατιωτικές εφαρμογές. Έχουν μεγάλο κόστος γι' αυτό το λόγο δεν είναι γνωστές στο ευρύ κοινό.

Ο Έλεγχος Λαθών (Error Checking)

Οι περισσότερες μνήμες που διατίθενται σήμερα στην αγορά είναι πάρα πολύ αξιόπιστες και τα περισσότερα συστήματα υπολογιστών κάνουν έλεγχο μέσω του ελεγκτή μνήμης (memory controller) για εντοπισμό λαθών κατά την εκκίνηση που θεωρείται ενέργεια αρκετή. Τα τσιπ μνήμης τα οποία διαθέτουν ενσωματωμένο έλεγχο λάθους χρησιμοποιούν μια μέθοδο που είναι γνωστή ως μέθοδος ελέγχου ισοτιμίας (parity) για να κάνουν έλεγχο λαθών. Τα τσιπς ισοτιμίας (parity chips) διαθέτουν ένα επιπλέον bit για κάθε 8 bits δεδομένων. Ο τρόπος που δουλεύει η ισοτιμία είναι απλός και τον έχουμε ήδη δει στην αρχή του μαθήματος. Υπενθυμίζουμε ότι στην άρτια ισοτιμία (even parity) όταν τα 8 bits ενός byte λάβουν δεδομένα, το τσιπ μετράει τον συνολικό αριθμό των 1 και αν αυτός ο αριθμός είναι περιττός (odd), το bit ισοτιμίας (parity bit) γίνεται ίσο με 1, ενώ αν αυτός ο αριθμός είναι άρτιος (even), το bit ισοτιμίας (parity bit) γίνεται ίσο με 0. Όταν αυτά τα δεδομένα διαβασθούν, υπολογίζεται ξανά ο συνολικός αριθμός των bits και συγκρίνεται με το bit ισοτιμίας. Αν το σύνολο των bits είναι περιττός αριθμός και το parity bit είναι ίσο με 1, τότε τα δεδομένα θεωρούνται ότι είναι σωστά (έγκυρα) και στέλνονται στην CPU για περαιτέρω επεξεργασία. Αλλά αν ο συνολικός αριθμός των bits είναι περιττός και το parity bit είναι ίσο με 0, το τσιπ γνωρίζει ότι υπάρχει λάθος (error) σε κάποιο από τα 8 bits και απορρίπτει τα δεδομένα. Η περιττή ισοτιμία (odd parity) εργάζεται με παρόμοιο τρόπο, με τη διαφορά ότι το parity bit γίνεται ίσο με 1 όταν ο συνολικός αριθμός των 1 που υπάρχουν byte είναι άρτιος (even). Το πρόβλημα με την ισοτιμία είναι ότι βρίσκει τα λάθη αλλά δεν κάνει τίποτα για να τα διορθώσει. Αν ένα byte δεδομένων δεν ταιριάξει με το bit ισοτιμίας του, τότε τα δεδομένα απορρίπτονται και το σύστημα κάνει άλλη μια προσπάθεια για να ξαναστείλει.

Όμως, οι υπολογιστές που βρίσκονται σε σημαντικές και καίριες θέσεις και η σωστή λειτουργία τους είναι απολύτως αναγκαία χρειάζονται ένα υψηλότερο επίπεδο ανοχής σε λάθη. Οι high-end servers διαθέτουν ένα είδος ελέγχου λαθών (error-checking) που είναι γνωστό ως κώδικας διόρθωσης λαθών (ECC, error-correction code), που όπως η ισοτιμία χρησιμοποιεί επιπλέον bits για να μπορεί να παρακολουθεί και να ελέγχει

τα δεδομένα που υπάρχουν σε κάθε byte. Η διαφορά είναι ότι ο ECC χρησιμοποιεί αρκετά bits για τον έλεγχο λαθών (error checking), όπου το πόσα εξαρτάται από το εύρος ζώνης του καναλιού (bus), αντί για ένα. Η μνήμη του ECC χρησιμοποιεί έναν ειδικό αλγόριθμο όχι μόνο για να μπορέσει να εντοπίσει μεμονωμένα λάθη bit, αλλά και για να τα διορθώσει. Η μνήμη του ECC μπορεί επίσης να εντοπίσει περιπτώσεις όπου υπάρχουν λάθη σε περισσότερα από ένα bits δεδομένων σ' ένα byte. Τέτοιες περιπτώσεις είναι πολύ σπάνιες και δεν μπορούν να διορθωθούν ούτε με τον ECC. Η πλειοψηφία των σημερινών υπολογιστών χρησιμοποιούν nonparity τσιπς μνήμης, τα οποία δεν παρέχουν κάποιο είδος ενσωματωμένου ελέγχου λαθών (built-in error checking), αλλά αντίθετα βασίζονται στον ελεγκτή μνήμης (memory controller) για τον εντοπισμό λαθών.

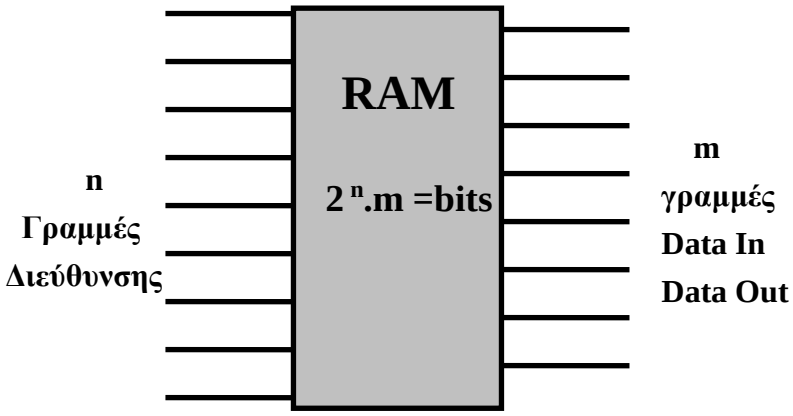
Άσκηση: Βρείτε τον αριθμό γραμμών μεταφοράς (Data lines) και τον αριθμό γραμμών διεύθυνσης για μία μνήμη χωρητικότητας 4Kbits εάν.

α. έχει οργάνωση κατά bit.

β. έχει οργάνωση κατά λέξεις.

α. στην οργάνωση κατά bit έχουμε μία γραμμή μεταφοράς η το πολύ να έχουμε μία γραμμή αποκλειστικά για το Data In και μία άλλη για το Data Out. Έτσι $2^{12}=4096 =4K$ και έχουμε κατά συνέπεια 12 γραμμές διεύθυνσης.

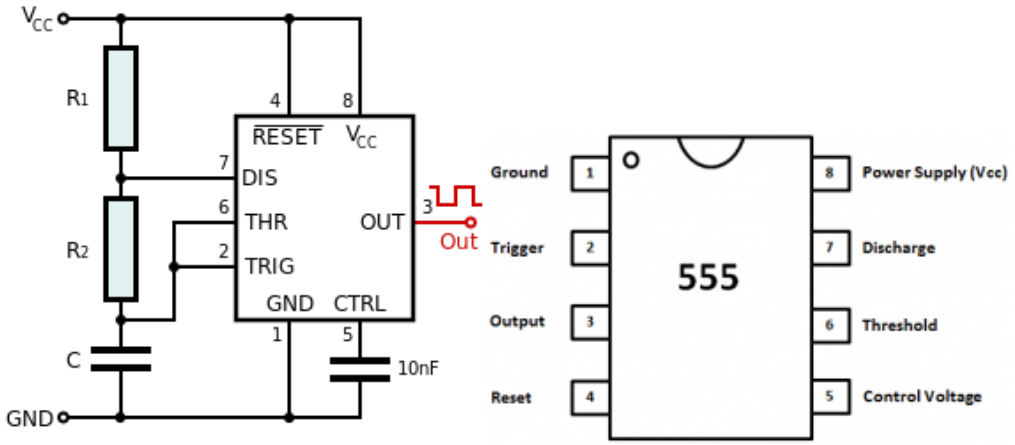
β. στην οργάνωση κατά λέξεις έχουμε 8 γραμμές μεταφοράς δεδομένων η 8 γραμμές για το Data In και άλλες 8 για το Data Out. Έχουμε $m\cdot 2^n=4096 \Rightarrow 8\cdot 2^n=4096 \Rightarrow 2^n=512 \Rightarrow n=9$ γραμμές διεύθυνσης όπως φαίνεται στο παρακάτω σχήμα.



Οργάνωση μίας μνήμης RAM.

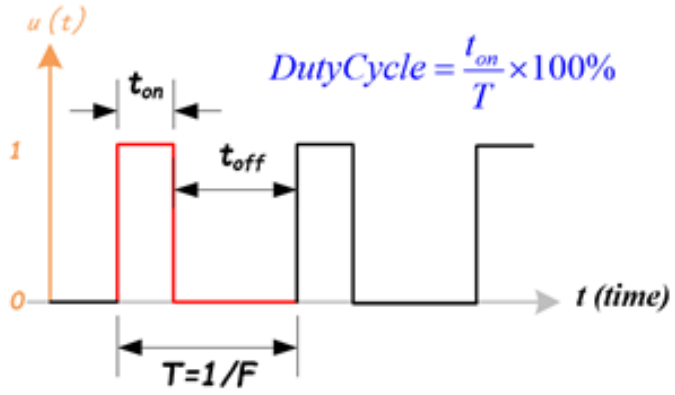
Κυκλώματα χρονισμού

Τα κυκλώματα χρονισμού είναι αυτά που χρησιμοποιούμε για να δημιουργήσουμε τους παλμούς χρονισμού, δηλαδή τους παλμούς του κεντρικού ρολογιού (master clock). Ένα από αυτά είναι το IC 555 το οποίο βλέπουμε παρακάτω συνδεδεμένο με δύο αντιστάσεις R1, R2 και ένα πυκνωτή C όπως βλέπουμε στο παρακάτω κύκλωμα.



Όταν είναι συνδεδεμένα τα εξωτερικά εξαρτήματα R1, R2, C μπορούμε να ελέγξουμε την συχνότητα ταλάντωσης της εξόδου του 555 μεταξύ της κατάστασης 1 και 0. Το 555 γι' αυτό το λόγο ονομάζεται ελεύθερος πολυδονητής (free- running multivibrator) ή ασταθής πολυδονητής (astable multivibrator). Στο παραπάνω κύκλωμα η συχνότητα ταλάντωσης καθώς και ο κύκλος απόδοσης ελέγχονται με ακρίβεια από τις δύο ανατιστάσεις και τον πυκνωτή. Ο πυκνωτής ο οποίος ονομάζεται πυκνωτής

χρονισμού φορτίζεται μέσω των αντιστάσεων R1, R2 από την εξωτερική τάση τροφοδοσίας Vcc.



Στο παραπάνω διάγραμμα βλέπουμε την έξοδο του 555. Με κόκκινο έχει σχεδιαστεί μία περίοδος Τ. Βλέπουμε ότι ο χρόνος που η έξοδος παραμένει στην κατάσταση 1 συμβολίζεται με $\mathbf{t_{on}}$ και ο χρόνος που η έξοδος παραμένει στην κατάσταση ο συμβολίζεται με $\mathbf{t_{off}}$. Οι δύο αυτοί χρόνοι όταν προστεθούν μας δίνουν την περίοδο του παλμικού σήματος στην έξοδο του 555 ίσον $\mathbf{T=t_{on}+t_{off}}$. Βλέπουμε επίσης τον ορισμό του κύκλου απόδοσης ή Duty Cycle όπως:

$$Duty-cycle = \frac{t_{on}}{t_{on}+t_{off}} = \frac{t_{on}}{T} x100\%$$

Ο χρόνος $\mathbf{t_{on}}$ είναι ίσος με:

$$t_{on} = 0,693(R_1 + R_2)C$$

Ο χρόνος $\mathbf{t_{off}}$ είναι ίσος με:

$$t_{off} = 0,693R_2C$$

Η συχνότητα λειτουργίας του παραπάνω κυκλώματος είναι:

$$F = \frac{1}{T} = \frac{1,44}{(R_1 + 2R_2)C}$$

Και ο κύκλος απόδοσης:

$$κύκλος - απόδοσης = \frac{t_{on}}{t_{on}+t_{off}} = \frac{R_2}{R_1 + 2R_2}$$

Παράδειγμα 1: Να βρεθεί η σχυχνότητα χρονισμού και ο κύκλος απόδοσης για το 555 όπως φαίνεται συνδεδεμένο στο προηγούμενο κύκλωμα αν R1=R2=1KΩ και C=1000pF.

Θα έχουμε για τη συχνότητα ότι:

$$\boldsymbol{F} = \frac{1}{\boldsymbol{T}} = \frac{1,44}{(\boldsymbol{R}_1 + 2\boldsymbol{R}_2)\boldsymbol{C}} = \frac{1,44}{(1000 + 2\boldsymbol{x}1000)10^{-9}} = 480\boldsymbol{KHz}$$

Επίσης για τους χρόνους:

$$\boldsymbol{t}_{\boldsymbol{on}} = 0,693(1000 + 1000)10^{-9} = 1,386\mu\text{sec} \quad \text{και}$$

$$\boldsymbol{t}_{\boldsymbol{off}} = 0,693.1000.10^{-9} = 0,693\mu\text{sec}$$

$$\boldsymbol{\kappa\acute{\upsilon}\kappa\lambda o\varsigma - \alpha\pi\acute{o}\delta o\sigma\eta\varsigma} = \frac{\boldsymbol{t}_{\boldsymbol{on}}}{\boldsymbol{t}_{\boldsymbol{on}} + \boldsymbol{t}_{\boldsymbol{off}}} = \frac{0,693}{2,08} = 33,3\%$$

Το παραπάνω κύκλωμα θα έχει έξοδο 1 στο 33,3% της περιόδου και έξοδο 0 στο 66,7% της περιόδου.

Παράδειγμα 2: Αν για την παραπάνω συνδεσμολογία του 555 έχουμε ότι η αντίσταση R2=500Ω ποια πρέπει να είναι η τιμή της αντίστασης R1 και του πυκνωτή C έτσι ώστε να έχουμε συχνότητα ταλάντωσης F=1MHz και κύκλο απόδοσης 25%.

$$\boldsymbol{\alpha\pi\acute{o}\delta o\sigma\eta} = \frac{\boldsymbol{R}_2}{\boldsymbol{R}_1 + 2.\boldsymbol{R}_2} \Rightarrow \boldsymbol{R}_1 = \frac{\boldsymbol{R}_2}{\text{Α}\boldsymbol{\pi\acute{o}\delta o\sigma\eta}} - 2.\boldsymbol{R}_2 \Rightarrow$$

$$\boldsymbol{R}_1 = \frac{500}{0,25} - 2.500 \Rightarrow \boldsymbol{R}_1 = 1000\boldsymbol{\Omega}$$

Για τη συχνότητα θα έχουμε:

$$\boldsymbol{F} = \frac{1,44}{(\boldsymbol{R}_1 + 2\boldsymbol{R}_2)\boldsymbol{C}} \Rightarrow \boldsymbol{C} = \frac{1,44}{(\boldsymbol{R}_1 + 2\boldsymbol{R}_2)\boldsymbol{F}} = \frac{1,44}{(1000 + 2.500)10^6} = 720\,\boldsymbol{pF}$$

Κεφάλαιο 12. Μετατροπές A/D και D/A

Όπως έχουμε δει σε όλη τη διάρκεια του μαθήματος μελετήθηκαν διάφορα τεχνικά θέματα τα οποία μας επιτρέπουν να κατανοήσουμε την λειτουργία των ψηφιακών κυκλωμάτων και να είμαστε σε θέση και να τα σχεδιάσουμε. Στις περισσότερες εφαρμογές τα ηλεκτρικά σήματα που λαμβάνουμε είναι αναλογικά επειδή στο σύνολο τους προέρχονται από αισθητήρες και θα πρέπει να τα μετατρέψουμε σε ψηφιακά για να είναι δυνατή η επεξεργασία τους, και ανάποδα ένα επεξεργασμένο σήμα το οποίο παράγεται από ένα περίπλοκο ψηφιακό κύκλωμα πρέπει να μετατραπεί σε αναλογικό για να φέρει το επιθυμητό αποτέλεσμα στην έξοδο του. Για παράδειγμα ένα ψηφιακός ακουστικός ενισχυτής θα πρέπει να διαβάζει την έξοδο ενός μικροφώνου σε ψηφιακή μορφή οπότε χρειαζόμαστε ένα **A to D** μετατροπέα, ενώ στην έξοδο ο ενισχυτής για να αποδόσει όλη την ονομαστική του ισχύ σε ένα ηχείο χρειάζεται ένα μετατροπέα **D to A.** Ο μετατροπέας **A to D** (**A**nalogue to **D**igital converter) είναι ένα κύκλωμα το οποίο μετατρέπει ένα αναλογικό σήμα σε ψηφιακό ενώ ένας **D to A** (Digital to Analogue converter) είναι ένα κύκλωμα το οποίο μετατρέπει ένα ψηφιακό σήμα σε αναλογικό.

1. Μετατροπέας D/A

Μία ψηφιακή λέξη (αριθμός) παρίσταται με το δυαδικό σύστημα αρίθμησης η το BCD. Ας υποθέσουμε ότι έχουμε ένα ψηφιακό σήμα τεσσάρων bits και θέλουμε να το μετατρέψουμε ανάλογα σε ένα αναλογικό σήμα από 0 μέχρι 3 Volts. Θα έχουμε την αναλογία που φαίνεται παρακάτω στον πίνακα μεταξύ του ψηφιακού και του αναλογικού σήματος.

D	C	B	A	Έξοδος Volts
0	0	0	0	0
0	0	0	1	0.2
0	0	1	0	0.4
0	0	1	1	0.6
0	1	0	0	0.8
0	1	0	1	1.0

0	1	1	0	1.2
0	1	1	1	1.4
1	0	0	0	1.6
1	0	0	1	1.8
1	0	1	0	2.0
1	0	1	1	2.2
1	1	0	0	2.4
1	1	0	1	2.6
1	1	1	0	2.8
1	1	1	1	3.0

Αναλογία μεταξύ ενός ψηφιακού και ενός αναλογικού σήματος

Το σχηματικό διάγραμμα ενός μετατροπέα κώδικα από ψηφιακό σήμα σε αναλογικό δηλαδή D/A φαίνεται παρακάτω στο σχήμα.

Σχηματική μετατροπή D/A

Η έξοδος Vo ενός μετατροπέα D/A των N bit θα δίνεται από την παρακάτω σχέση:

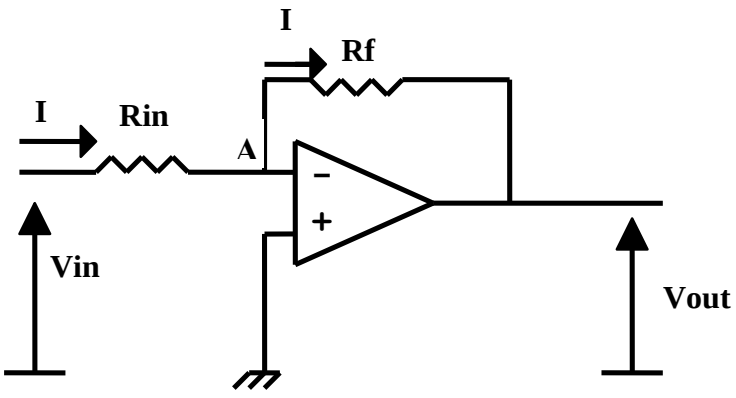
$$V_0 = V_R.(a_{n-1}.2^{n-1} + a_{n-2}.2^{n-2} + ... + a_0.2^{-n})$$

όπου V_R είναι μία τάση αναφοράς του κυκλώματος και a_n είναι 1 η 0 ανάλογα με την τιμή του αντίστοιχου bit. Το περισσότερο σημαντικό ψηφίο MSB αντιστοιχεί στο a_{n-1} με συντελεστή βαρύτητας $V_R/2$ και το λιγότερο

σημαντικό LSB αντιστοιχεί στο a_o με συντελεστή βαρύτητας $V_R/2^N$. Ας θεωρήσουμε μία λέξη των τεσσάρων bits. Τότε n=4. Άρα θα έχουμε ότι:

$$
\begin{aligned}
V &= (a3 \cdot 2^{-1} + a2 \cdot 2^{-2} + a1 \cdot 2^{-3} + a0 \cdot 2^{-4}) \cdot V_R \\
&= \left(\frac{a3}{2} + \frac{a2}{4} + \frac{a1}{8} + \frac{a0}{16}\right) V_R \\
&= (8a3 + 4a2 + 2a1 + a0). \frac{V_R}{16} \\
&= (8a3 + 4a2 + 2a1 + a0)
\end{aligned}
$$

Αν η τάση αναφοράς $\mathbf{V_R}$ **ε**ίναι ίση με 16 Volts. Αν βάλουμε στην είσοδο του μετατροπέα τον αριθμό 0001 δηλαδή a3=a2=a1=0 και ao=1 θα πάρουμε στην έξοδο του μετατροπέα 1 Volt. Αν βάλουμε το σήμα 0111 θα πάρουμε στην έξοδο 7 Volts. Άρα η αναλογική τάση στη έξοδο του μετατροπέα είναι ανάλογη της τιμής του ψηφιακού σήματος. Το κύκλωμα το οποίο κάνει μία τέτοια μετατροπή βασίζεται πάνω σε ορισμένες ιδιότητες του διαφορικού ολοκληρωμένου ενισχυτή η τελεστικού ενισχυτή που παρουσιάζουμε παρακάτω. Ο τελεστικός ενισχυτής είναι ένας ενισχυτής με πολύ μεγάλη σύνθετη αντίσταση εισόδου και πολύ μικρή αντίσταση εξόδου σχεδόν μηδενική. Η ενίσχυση του είναι πολύ μεγάλη και αρκεί ένα πολύ μικρό σήμα ηλεκτρικό σήμα στην είσοδο για να τον οδηγήσει στον κόρο. Για το παραπάνω λόγο ένας τελεστικός ενισχυτής χρησιμοποιείται αποκλειστικά σε συνδεσμολογία ανάδρασης για να μπορέσουμε να ελέγξουμε το κέρδος τάσης ή ρεύματος σε πεπερασμένες τιμές όπως φαίνεται παρακάτω σε μια απλή συνδεσμολογία.

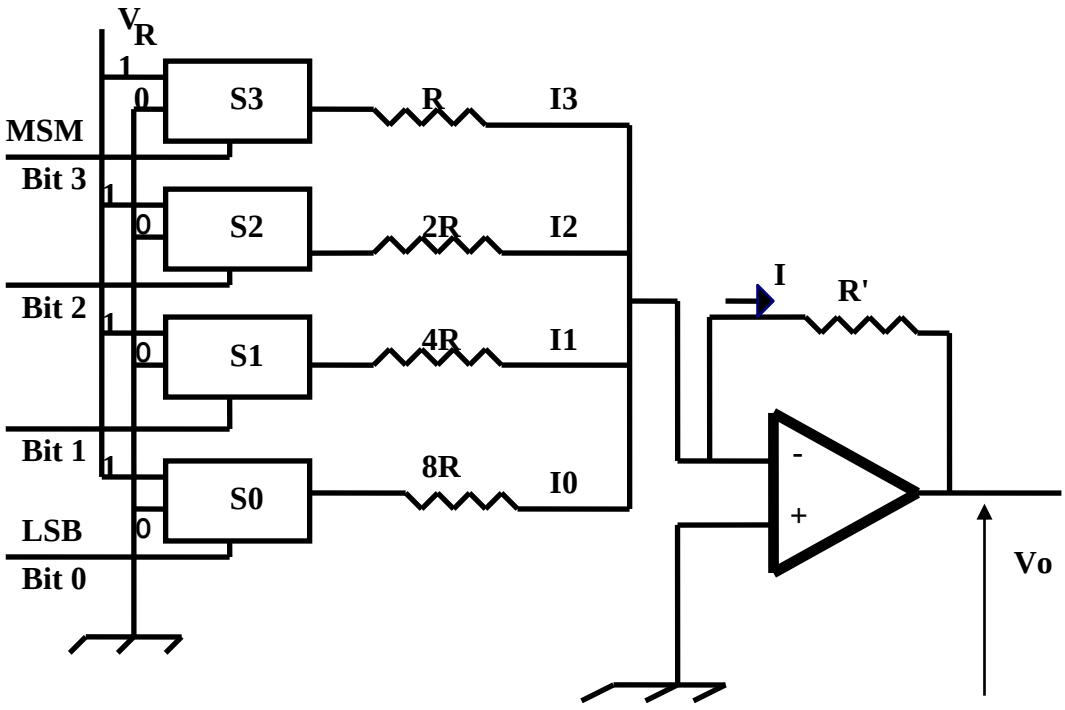


Τελεστικός ενισχυτής και συνδεσμολογία με ανάδραση

Λόγω της μεγάλης αντίστασης στην είσοδο του τελεστικού δεν εισέρχεται καθόλου ρεύμα. Κατά συνέπεια το δυναμικό του σημείου Α είναι 0 (Virtual Zero). Μπορούμε να γράψουμε ότι $\mathbf{V_{in}=R_{in}\cdot I}$ και $\mathbf{V_{out}=R_f.I}$. Αν διαιρέσουμε αυτές τις δύο σχέσεις θα πάρουμε ότι $\mathbf{V_{out}=Av\cdot V_{in}}$ και όπου $\mathbf{Av=R_f/R_{in}}$. Αν μεταβάλλουμε τις αντιστάσεις θα μεταβάλλουμε την ενίσχυση του παραπάνω κυκλώματος. Η παραπάνω συνδεσμολογία του τελεστικού ενισχυτή λέγεται συνδεσμολογία με ανάδραση. Παρατηρούμε ότι η έξοδος του κυκλώματος συνδέεται με την είσοδο μέσω μιας αντίστασης. Αν δεν συμβεί αυτό επειδή η ενίσχυση του τελεστικού ενισχυτή είναι στην ουσία άπειρη το κύκλωμα οδηγείται στιν κόρο και δεν λειτουργεί.

Οι τελεστικοί ενισχυτές είναι ένα μέρος του κυκλώματος που επιτρέπει τη μετατροπή ενός σήματος ψηφιακού σε αναλογικού. Χρησιμοποιούνται επίσης συχνά για μετατρέψουμε το ρεύμα σε τάση.

Χρησιμοποιούνται σαν αθροιστές τάσεων και η συνδεσμολογία φαίνεται στο παρακάτω σχήμα.



Μετατροπέας D/A ενός ψηφιακού σήματος στο ανάλογο του αναλογικό με το παραπάνω κύκλωμα τελεστικού ενισχυτή.

S_3, S_2, S_1, S_0 είναι ψηφιακοί διακόπτες. Αν το bit στο οποίο αντιστοιχούν είναι μηδέν τότε γειώνουν το φορτίο, ενώ διαφορετικά αν δηλαδή είναι ένα τότε το συνδέουν με την πηγή $\mathbf{V_R}$. Αν γράψουμε ότι το ρεύμα I=I3+I2+I1+I0 θα πάρουμε ότι:

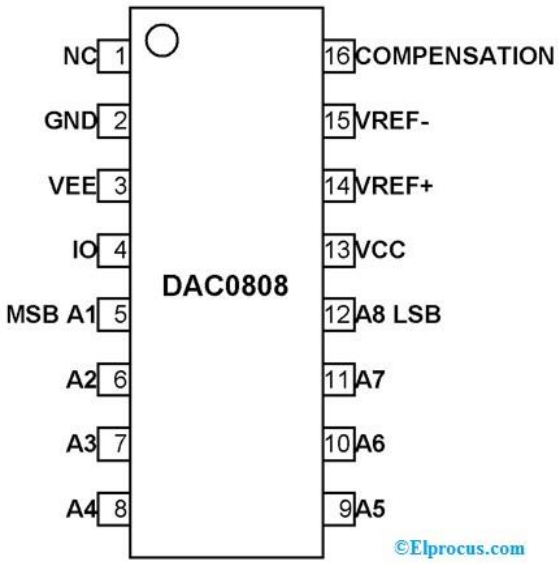
$$V_0 = \frac{R'}{R}.(a_3.8 + a_2.4 + a_1.2 + a_o)\frac{V_R}{8}$$

Αν διαλέξουμε R'=R/2 και την τάση V_R=16Volts τότε θα έχουμε ότι:

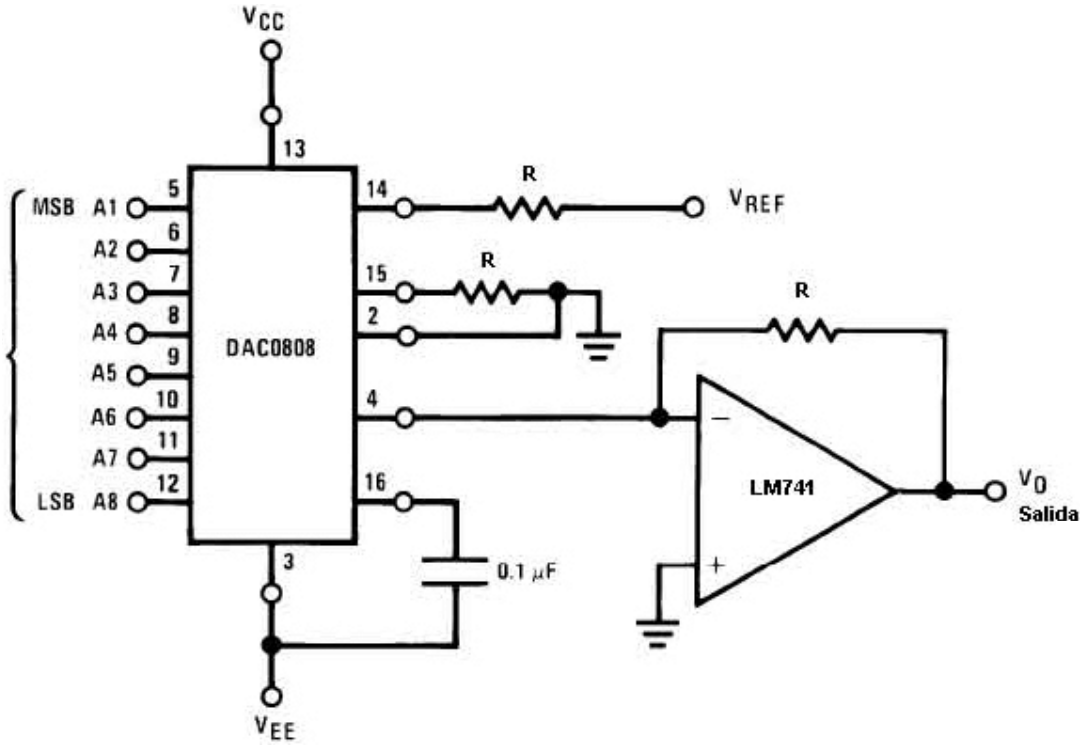
$$V_0 = (a_3.8 + a_2.4 + a_1.2 + a_o)Volts$$

Το Ολοκληρωμένο Κύκλωμα **DAC0808**

Οι μετατροπείς που μετατρέπουν το αναλογικό σήμα σε σε ψηφιακό έχουν μεγάλη χρήση σε διάφορες εφαρμογές ελέγχου και αυτοματισμών. Ένα χαρακτηριστικό παράδειγμα είναι το ολοκληρωμένο κύκλωμα DAC0808. Το παραπάνω είναι ένα μονολιθικό ολοκληρωμένο κύκλωμα με δεκαέξι συνολικά ακροδέκτες, το οποίο μπορεί να δεχτεί στην είσοδο έναν ψηφιακό σήμα οκτώ bits στους ακροδέκτες από 5 έως 12. Μία σειρά από τρανζίστορ σε διακοπτική λειτουργία (current switches) μπορούν να ενεργοποιηθούν από το λογικό 1 στις εισόδους και να τροφοδοτήσουν ένα κύκλωμα κλίμακας R-2R. Η έξοδος του κυκλώματος DAC0808 (ακροδέκτης 4) είναι σχεδιασμένη ώστε να παράγει ένα ρεύμα $\mathbf{I_0}$ σε μία αντίσταση φορτίου R.



Η τυπική υλοποίηση του μετατροπέα D/A απαιτεί τη σύνδεση ενός τελεστικού ενισχυτή αντιστροφής, όπως φαίνεται στο παρακάτω σχήμα, για τη μετατροπή του ρεύματος σε τάση. Χρησιμοποιούμε το τελεστικό ενισχυτή LM741 όπως φαίνεται στην παρακάτω συνδεσμολογία. Φαίνεται καθαρά η τάση αναφοράς Vref. Βλέπουμε στον ακροδέκτη 1 γραμμένο το NC που σημαίνει στα αγγλικά Non Connected δηλαδή δεν είναι συνδεδεμένο εσωτερικά με κάτι. Το περισσότερο σημαντικό ψηφίο MSB του οκτάμπιτου αριθμού στην είσοδο είναι το pin 5 και το LSB το pin 12. Η έξοδος είναι το pin 4.



Εάν πρόκειται για ψηφιακό σήμα μήκους οκτώ bits, τότε πρέπει να τοποθετήσουμε οκτώ εισόδους στο δικτύωμα R-2R και θα έχουμε:

$$I_0 = K(\frac{b_8}{2} + \frac{b_7}{4} + \frac{b_6}{8} + \frac{b_5}{16} + \frac{b_4}{32} + \frac{b_3}{64} + \frac{b_2}{128} + \frac{b_1}{256})$$

Η παραπάνω σχέση οδηγεί μετά από λίγες πράξεις στη εξίσωση

$$I_0 = K\frac{[\varepsilon\acute{\iota}\sigma o\delta o\varsigma]_{10}}{256}$$

όπου στον αριθμητή μέσα στις αγκύλες βρίσκεται η δεκαδική τιμή της εισόδου. Με τη βοήθεια του τελεστικού ενισχυτή, το παραπάνω ρεύμα μετατρέπεται σε τάση με βάση τη σχέση:

$$V_0 = V_{ref}(\frac{b_8}{2} + \frac{b_7}{4} + \frac{b_6}{8} + \frac{b_5}{16} + \frac{b_4}{32} + \frac{b_3}{64} + \frac{b_2}{128} + \frac{b_1}{256})$$

$$V_0 = -V_{ref}\frac{[\varepsilon\acute{\iota}\sigma o\delta o\varsigma]_{10}}{256}$$

Η παραπάνω σχέση οδηγεί στο συμπέρασμα ότι η μέγιστη τάση που μπορεί να παράγει ο μετατροπέας D/A, όταν όλες οι είσοδοι βρίσκονται σε λογικό 1, είναι κατ' απόλυτη τιμή το (255/256) της τάσης αναφοράς στην είσοδο. Οι μετατροπείς ψηφιακού σήματος σε αναλογικό αναφέρονται συχνά και ως «πολλαπλασιαστικοί μετατροπείς» (multiplying DACs), διότι η τάση εξόδου (ή το ρεύμα εξόδου) προκύπτουν από μια τιμή αναφοράς πολλαπλασιασμένη με το κλάσμα της παραπάνω σχέσης. Σύμφωνα με την παραπάνω σχέση, εάν η τάση αναφοράς Vref = +5V, η τάση εξόδου θα μεταβάλλεται από 0 έως 255/256 × Vref = 4.98 V. Ας σημειωθεί ότι το θετικό πρόσημο της τάσης εξόδου προκύπτει από τη φορά του ρεύματος του DAC, προς τον ακροδέκτη εξόδου. Τελειώνοντας την εφαρμογή με τον μετατροπέα DAC0808 μπορούμε να αναφέρουμε ότι ο χρόνος αποκατάστασης του ρεύματος εξόδου για μια αλλαγή του σήματος εισόδου είναι κατά μέγιστο 150 ns. Ο χρόνος αυτός είναι αρκετά μικρός, ώστε ο μετατροπέας αυτός να είναι κατάλληλος για γρήγορες εφαρμογές.

2. Μετατροπή ενός αναλογικού σήματος σε ψηφιακό A/D

Η μετατροπή αναλογικού σήματος σε ψηφιακό (A/D conversion) είναι μια πολύ σημαντική διεργασία διότι επιτρέπει σε ένα περίπλοκο ψηφιακό σύστημα που παράγει ψηφιακά σήματα να τα μετατρέψει σε αναλογικά και να διασυνδεθεί με τον πραγματικό αναλογικό κόσμο. Τα διάφορα φυσικά μεγέθη (πίεση, θερμοκρασία, τάση, απόσταση κλπ) μεταβάλλονται με αναλογικό τρόπο. Είναι τα ηλεκτρικά σήματα που δίνουν αντίστοιχοι αισθητήρες. Κατά συνέπεια, η μετατροπή τους σε τάση μέσω κάποιου κυκλώματος δημιουργεί ένα αναλογικό σήμα τάσης, το οποίο για να εισαχθεί σε κάποιο ψηφιακό σύστημα επεξεργασίας πρέπει να

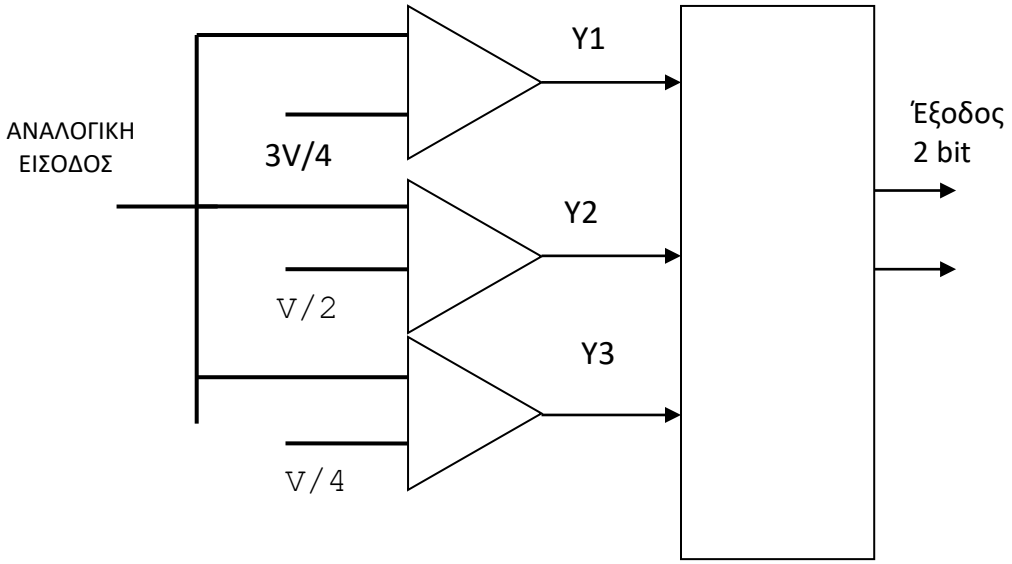
κωδικοποιηθεί ή να ψηφιοποιηθεί κατάλληλα. Το κύκλωμα που χρησιμοποιούμε γι' αυτήν την κωδικοποίηση το αποκαλούμε μετατροπέα A/D (A/D converter). Ένας μετατροπέας A/D έχει κατ' αρχήν μία απλή ή διαφορική είσοδο, στην οποία τοποθετείται η αναλογική τάση που θέλουμε να μετατρέψουμε σε ψηφιακή, και έναν αριθμό από ακροδέκτες εξόδου που θα οδηγηθούν σε κατάλληλες λογικές καταστάσεις 0 ή 1 μετά τη μετατροπή. Σε πρακτικά κυκλώματα ο ελάχιστος αριθμός των δυαδικών ψηφίων από τα οποία αποτελείται η ψηφιακή λέξη μετά την μετατροπή είναι οκτώ, οπότε λέμε ότι το σύστημα είναι οκτάμπιτο και μπορεί να διακρίνει συνολικά 28=256 διαφορετικές στάθμες. Αντίστοιχα, μπορεί να έχουμε μετατροπείς A/D 10 bits (οπότε οι στάθμες είναι 1024), 12 bits (4096 στάθμες), των 16 bits, (65536 στάθμες) κ.ο.κ. Οι τιμές αυτές είναι ενδεικτικές, διότι δεν αποκλείεται να συναντήσουμε μετατροπείς με διάφορες αναλύσεις, αν και πρακτικά τα 24 bits θεωρούνται αρκετά για να μας δώσουν εξαιρετικά καλή διακριτική ικανότητα, σχεδόν στις περισσότερες εφαρμογές. Στα συστήματα μετρήσεων σπάνια συντρέχει κάποιος πρακτικός λόγος για ανάλυση καλύτερη από 16 bits. Σε απλά συστήματα τα οκτώ bits είναι αρκετά, και για το λόγο αυτό θα αρκεστούμε σε τέτοια ολοκληρωμένα κυκλώματα στο πλαίσιο του μαθήματος. Ένας απλός και γρήγορος Μετατροπέας A/D και από τους απλούστερους και ταχύτερους μετατροπείς A/D είναι οι λεγόμενοι μετατροπείς A/D αστραπής (flash converters). Οι μετατροπείς αυτοί διαθέτουν έναν αριθμό από συγκριτές.

Ένα κύκλωμα συγκριτή λειτουργεί με τη βοήθεια τελεστικού ενισχυτή, ο οποίος λαμβάνει ένα δυναμικό αναφοράς (Vref)το οποίο παίρνει συνήθως από μια δίοδο Ζένερ και το συγκρίνει ανά πάσα στιγμή με την τάση που δέχεται στην είσοδο. Εάν η τάση αναφοράς είναι μεγαλύτερη από την τάση εισόδου, η έξοδος του συγκριτή οδηγείται σε λογικό ένα, ενώ εάν είναι μικρότερη από την τάση εισόδου οδηγείται σε λογικό μηδέν. Οι τάσεις αναφοράς των συγκριτών διαιρούν τη συνολική δυναμική περιοχή τάσεων (0 έως V Volts) που διαβάζει ο μετατροπέας A/D σε τέσσερις περιοχές.

Το παρακάτω σχήμα παρουσιάζει έναν μετατροπέα με τρεις συγκριτές με τη βοήθεια 3 τελεστικών ενισχυτών. Οι τάσεις αναφοράς των συγκριτών διαιρούν τη συνολική περιοχή τάσεων (0 έως V Volts) που διαβάζει ο

μετατροπέας A/D σε τέσσερις περιοχές. Ανάλογα με την περιοχή τάσης όπου βρίσκεται η αναλογική τάση εισόδου, οι έξοδοι Y1, Y2 και Y3 μπορούν να βρεθούν σε μία από τις τέσσερις λογικές καταστάσεις του παρακάτω επίσης πίνακα.

Τάση εισόδου	Y3	Y2	Y1
0 έως V/4	0	0	0
V/4 έως V/2	1	0	0
V/2 έως 3V/4	1	1	0
3V/4 έως V	1	1	1



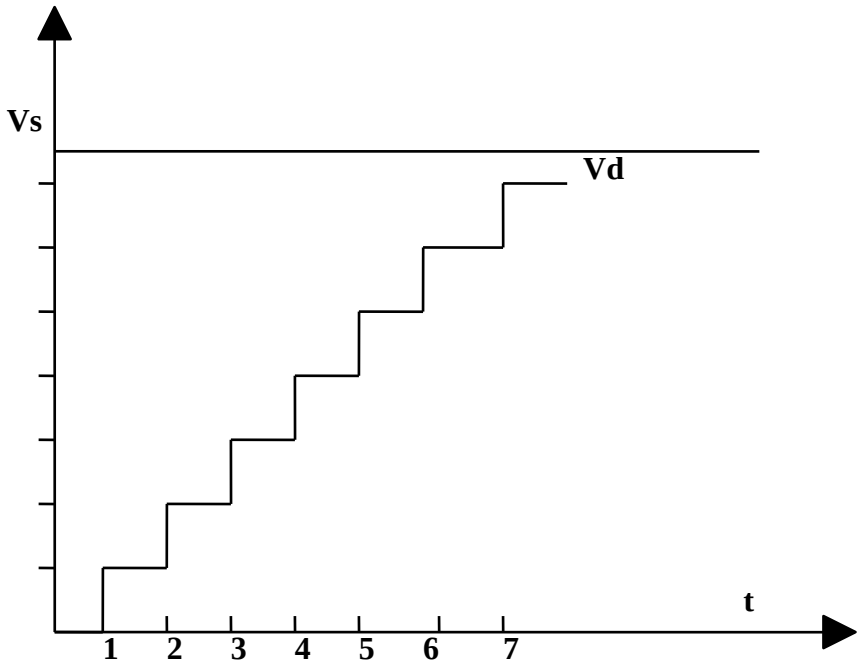
Ένας αποκωδικοποιητής τέσσερα προς δύο μεταφέρει την πληροφορία στην έξοδο. Ένας τέτοιος μετατροπέας είναι εξαιρετικά γρήγορος, αλλά για να έχει ανάλυση καλύτερη από αυτήν που περιγράψαμε παραπάνω θα πρέπει να αυξηθούν εκθετικά τα κυκλώματα εισόδου και ο αριθμός των τάσεων αναφοράς. Σαν αποτέλεσμα, ένας μετατροπέας A/D τέτοιου τύπου υλοποιείται δύσκολα και έχει σημαντικό κόστος.

Στις άλλες τεχνικές μετατροπής αναλογικού σήματος σε ψηφιακό, κεντρική θέση κατέχει πάλι ο συγκριτής. Αντί για πολλά κυκλώματα συγκριτών στη είσοδο, έχουμε έναν μόνο συγκριτή και αντί για πολλές

τάσεις αναφοράς δημιουργούμε μια μεταβλητή τάση αναφοράς, που καλύπτει όλη τη δυναμική περιοχή που μας ενδιαφέρει. Οι διάφορες τεχνικές μετατροπής A/D που χρησιμοποιούμε στα πρακτικά κυκλώματα διαφέρουν στον τρόπο που δημιουργούμε τη μεταβλητή τάση αναφοράς, η οποία συγκρίνεται με τη τάση εισόδου.

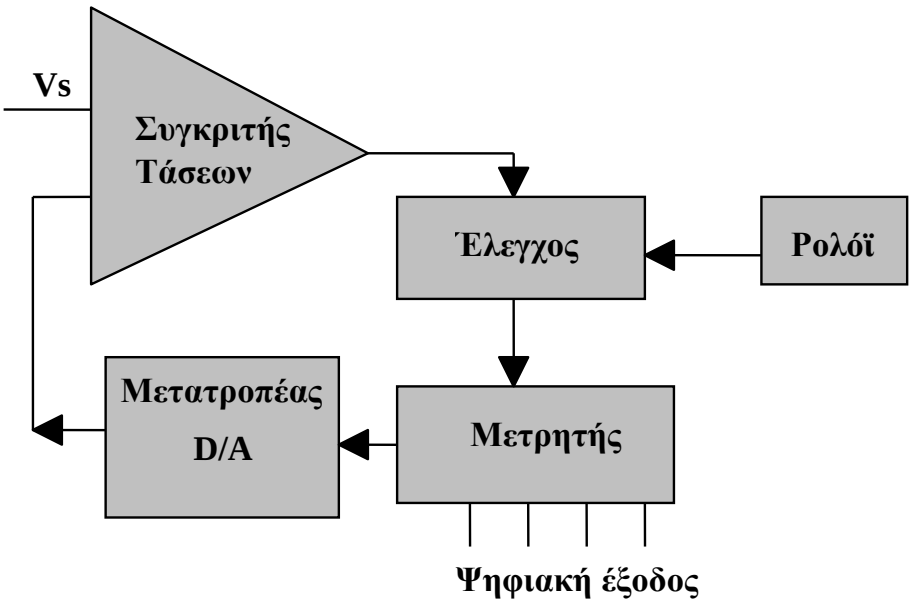
Κλιμακωτή Τάση Αναφοράς

Η τεχνική της διαδοχικής προσέγγισης είναι η περισσότερο πρόσφορη και ευρέως χρησιμοποιούμενη μέθοδος για την παραγωγή μίας μεταβλητής τάσης αναφοράς για σύγκριση με την αναλογική τάση εισόδου με τη χρήση ενός μετρητή σε συνδυασμό με έναν μετατροπέα D/A. Ένα αναλογικό σήμα Vs για να μετατραπεί σε ψηφιακό διαβιβάζεται σε ένα συγκριτή τάσεως μαζί με μία ακολουθία χρονικά ισαπεχόντων παλμών. Οι χρονικά ισαπέχοντες παλμοί όταν ξεκινούν θέτουν σε λειτουργία ένα μετρητή. Όταν η τάση Vs είναι ίση με την τάση αναφοράς μπλοκάρετε το περιεχόμενο του απαριθμητή και το περιεχόμενο του είναι το ψηφιακό ανάλογο του αναλογικού σήματος στην είσοδο. Ένα σχηματικό διάγραμμα της μετατροπής φαίνεται στο παρακάτω σχήμα.



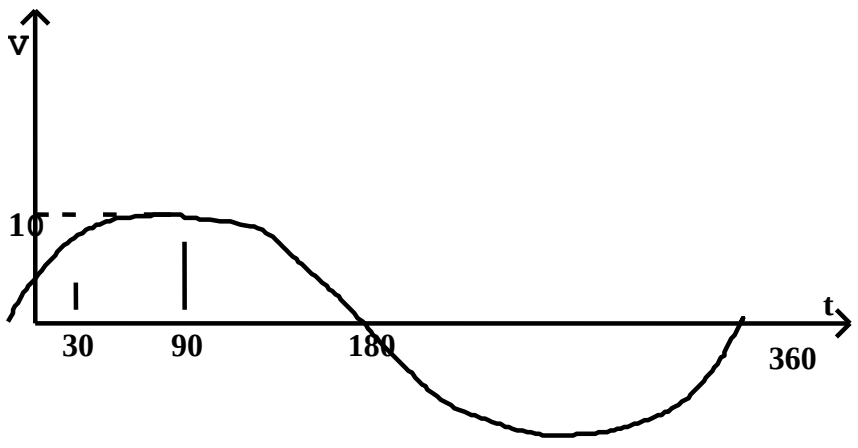
Σχηματικό BLOCK διάγραμμα μετατροπέα A/D. Στην συγκεκριμένη περίπτωση το περιεχόμενο του απαριθμητή θα είναι 7.

Το ηλεκτρονικό κύκλωμα που εκτελεί την όλη διαδικασία φαίνεται παρακάτω στο σχήμα. Ένα τέτοιο κύκλωμα επιτρέπει τη μετατροπή ενός σήματος το οποίο δε μεταβάλλεται χρονικά.



Κύκλωμα ενός μετατροπέα σήματος αναλογικού σε ψηφιακό

Όταν το αναλογικό σήμα μεταβάλλεται με το χρόνο τη θέση του απλού μετρητή κατέχει ένας μετρητής UP DOWN. Η δε ακρίβεια του μετατρεπόμενου σήματος εξαρτάται αποκλειστικά με την ταχύτητα δειγματοληψίας του αναλογικού σήματος. Όσο πιο κοντά στον χρόνο είναι τα σημεία τόσο μεγαλύτερη η ακρίβεια της μετατροπής. Έστω ότι έχουμε ένα σήμα ημιτονοειδούς μορφής όπως φαίνεται στο παρακάτω σχήμα. Το σήμα έχει εύρος 10Volts και συχνότητα 50 Hz.

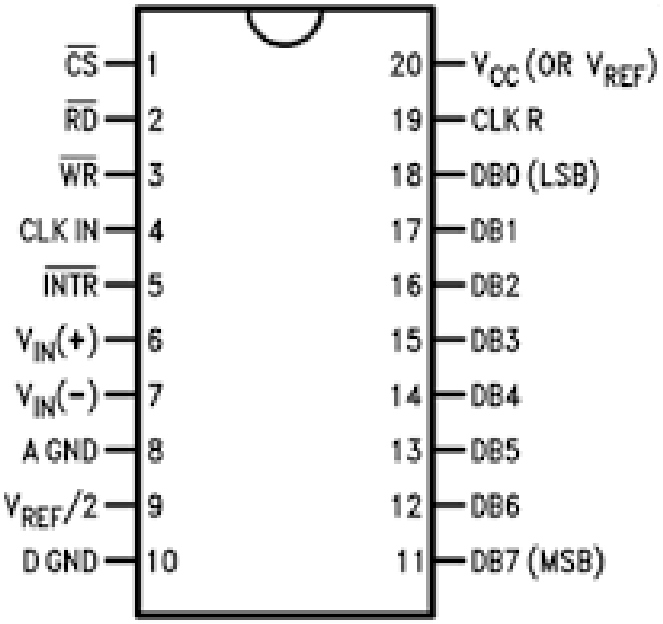


Δειγματοληψία κάθε 30 μοίρες δηλαδή κάθε 30/360.50=1/600 του δευτερολέπτου.

Το Ολοκληρωμένο Κύκλωμα **ADC0804**

Το ολοκληρωμένο κύκλωμα ADC0804 της εταιρείας National Semiconductor είναι ένας μετατροπέας αναλογικού σήματος σε ψηφιακό (A/D) μήκους οκτώ bits, που στηρίζεται στην τεχνική της διαδοχικής προσέγγισης. Είναι σχεδιασμένος να δέχεται και να στέλνει σήματα ελέγχου, ώστε η λειτουργία του μπορεί να ελέγχεται από μικροεπεξεργαστές ή μικροελεγκτές. Μπορεί, βέβαια, να λειτουργήσει και αυτόνομα. Πρόκειται για φθηνό και αξιόπιστο ολοκληρωμένο κύκλωμα κατάλληλο για απλές εφαρμογές ελέγχου και μετρήσεων.

Το διάγραμμα ακροδεκτών του ολοκληρωμένου κυκλώματος φαίνεται παρακάτω.



Ο μετατροπέας αυτός δέχεται μια διαφορά δυναμικού ανάμεσα στους ακροδέκτες Vin(+) και Vin(−) (ακροδέκτες 6 και 7, αντίστοιχα) και παράγει μια ψηφιακή έξοδο στους ακροδέκτες 11 έως 18. Το λιγότερο σημαντικό bit (LSB) αντιστοιχεί στον ακροδέκτη 18. Ο χρονισμός επιτυγχάνεται με ένα δικτύωμα RC, που τοποθετείται ανάμεσα στους ακροδέκτες 4 και 19. Οι ακροδέκτες 1, 2, 3 και 5 προορίζονται για τον έλεγχο του μετατροπέα από

εξωτερικές συσκευές. Μπορούν να συνδεθούν σε κατάλληλες λογικές στάθμες, ώστε το κύκλωμα να λειτουργεί χωρίς εξωτερικό έλεγχο . Η τάση τροφοδοσίας που δέχεται το κύκλωμα είναι Vcc = 5V στον ακροδέκτη 20. Ο τυπικός χρόνος μετατροπής είναι 100 μs.

Κάθε μετατροπέας αναλογικού σήματος σε ψηφιακό δέχεται ένα ορισμένο εύρος αναλογικών τάσεων εισόδου. Εάν δεν ορίσουμε κάτι διαφορετικό, ο μετατροπέας ADC0804 είναι σχεδιασμένος ώστε να δέχεται στη διαφορική είσοδο διαφορές δυναμικού από 0 έως 5 Volts. Ο ακροδέκτης Vin(−) πρέπει απαραίτητα να βρίσκεται σε πιο αρνητικό δυναμικό από τον ακροδέκτη Vin(+). Ο μετατροπέας μπορεί να χειριστεί αρνητικές τάσεις με τη βοήθεια κατάλληλου δικτυώματος στο κύκλωμα εισόδου. Εάν επιθυμούμε να μικρύνουμε το εύρος της περιοχής τάσεων εισόδου, θα πρέπει να εφαρμόσουμε το κατάλληλο δυναμικό στον ακροδέκτη 9. Για μέγιστη περιοχή μετατροπής 5 Volts, η τάση στον ακροδέκτη 9 πρέπει να είναι 4.5V, τάση που παρέχεται εσωτερικά εάν ο ακροδέκτης 9 παραμείνει εξωτερικά ασύνδετος. Εάν θέλουμε η δυναμική περιοχή τάσεων εισόδου να καλύπτει Vref Volts, θα πρέπει να εφαρμόσουμε μια τάση Vref /2 στον ακροδέκτη 9. Έστω, για παράδειγμα, ότι θέλουμε να καλύψουμε την περιοχή αναλογικών τάσεων εισόδου από 0.5 έως 3.5 Volts. Αυτό αντιπροσωπεύει μια δυναμική περιοχή 3V. Άρα, στον ακροδέκτη 9 πρέπει να τοποθετήσουμε δυναμικό 1.5V και στον ακροδέκτη Vin(−) θα εφαρμόσουμε σταθερά μια στάθμη 0.5V σε σχέση με τη γη. Έτσι, όταν στον ακροδέκτη Vin(+) εφαρμοστεί τάση 0.5V, η έξοδος στους ακροδέκτες 18 έως 11 (που αντιστοιχούν στις ψηφιακές εξόδους DB0 έως DB7) θα είναι 00000000. Όταν η τάση στον ακροδέκτη Vin(+) είναι 3.5V τότε στην έξοδο θα προκύπτει ο δυαδικός αριθμός 11111111 (δηλαδή ο δεκαδικός 255).

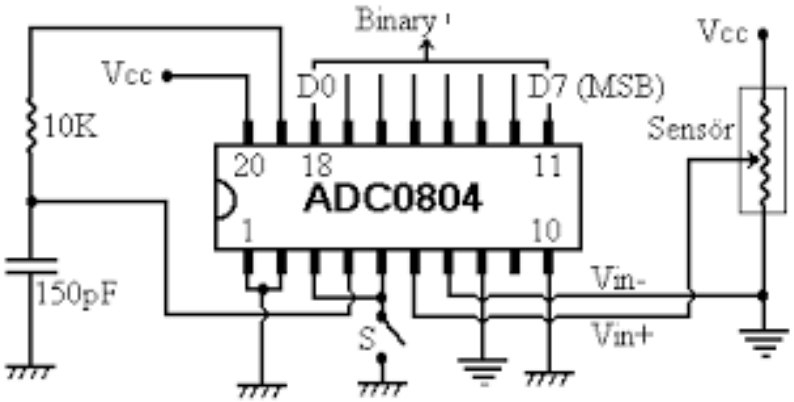
Εφαρμογές με τον Μετατροπέα ADC0804

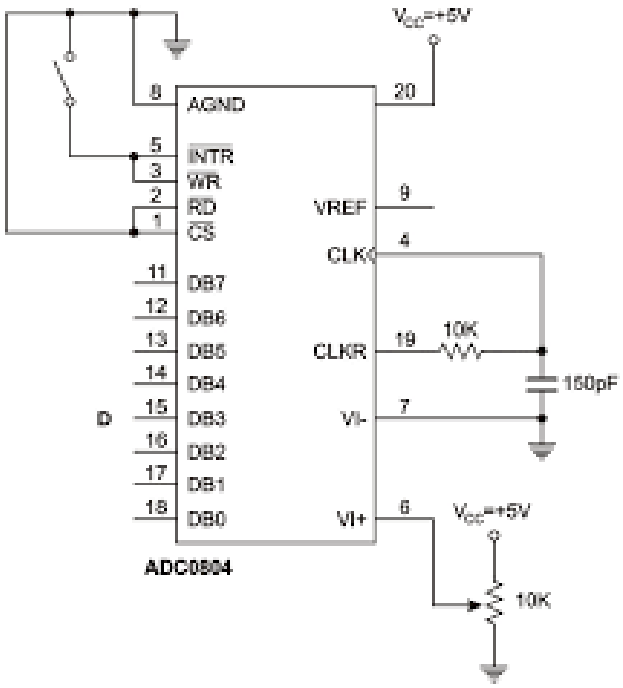
Απλή εφαρμογή συνεχούς λειτουργίας Στην απλούστερη περίπτωση, ο ADC0804 μπορεί να λειτουργεί χωρίς τη βοήθεια εξωτερικού ελέγχου. Για τον σκοπό αυτό συνδέεται στις εισόδους CLK ένα δικτύωμα Rc, που εξασφαλίζει τον αυτοχρονισμό του κυκλώματος (self-clocking). Τυπικές τιμές που δίνει ο κατασκευαστής είναι R = 10 kΩ και C = 150pF. Με βάση

αυτές τις τιμές υπολογίζουμε τη συχνότητα λειτουργίας του εξωτερικού ρολογιού από τον τύπο:

$$F = \frac{1}{1,1RC} = 607KHz$$

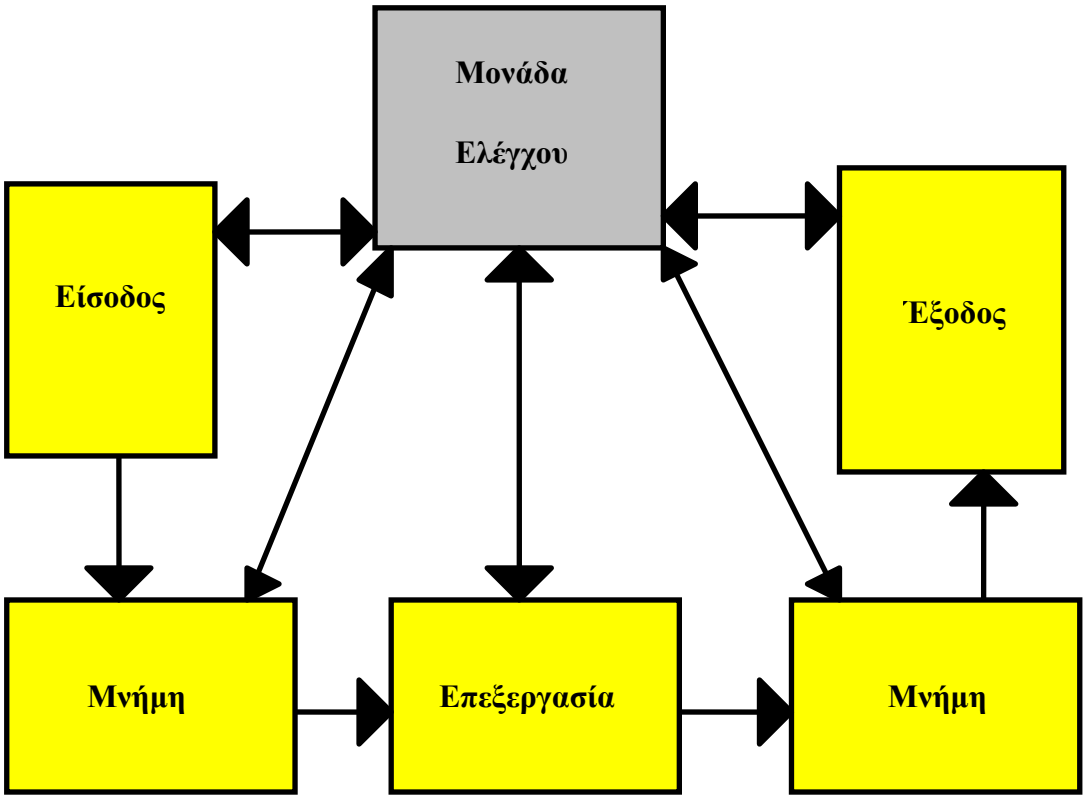
Κάθε μετατροπή απαιτεί περίπου 65 κύκλους του εξωτερικού ρολογιού δηλαδή ο χρόνος της κάθε μετατροπής είναι περίπου 100 μs. Πρακτικά το σύστημα αυτό μπορεί να πετύχει περίπου 9000 μετατροπές το δευτερόλεπτο σε ελεύθερη λειτουργία. Τα όρια αυτά καθορίζουν και τους περιορισμούς στη χρήση του συγκεκριμένου κυκλώματος.





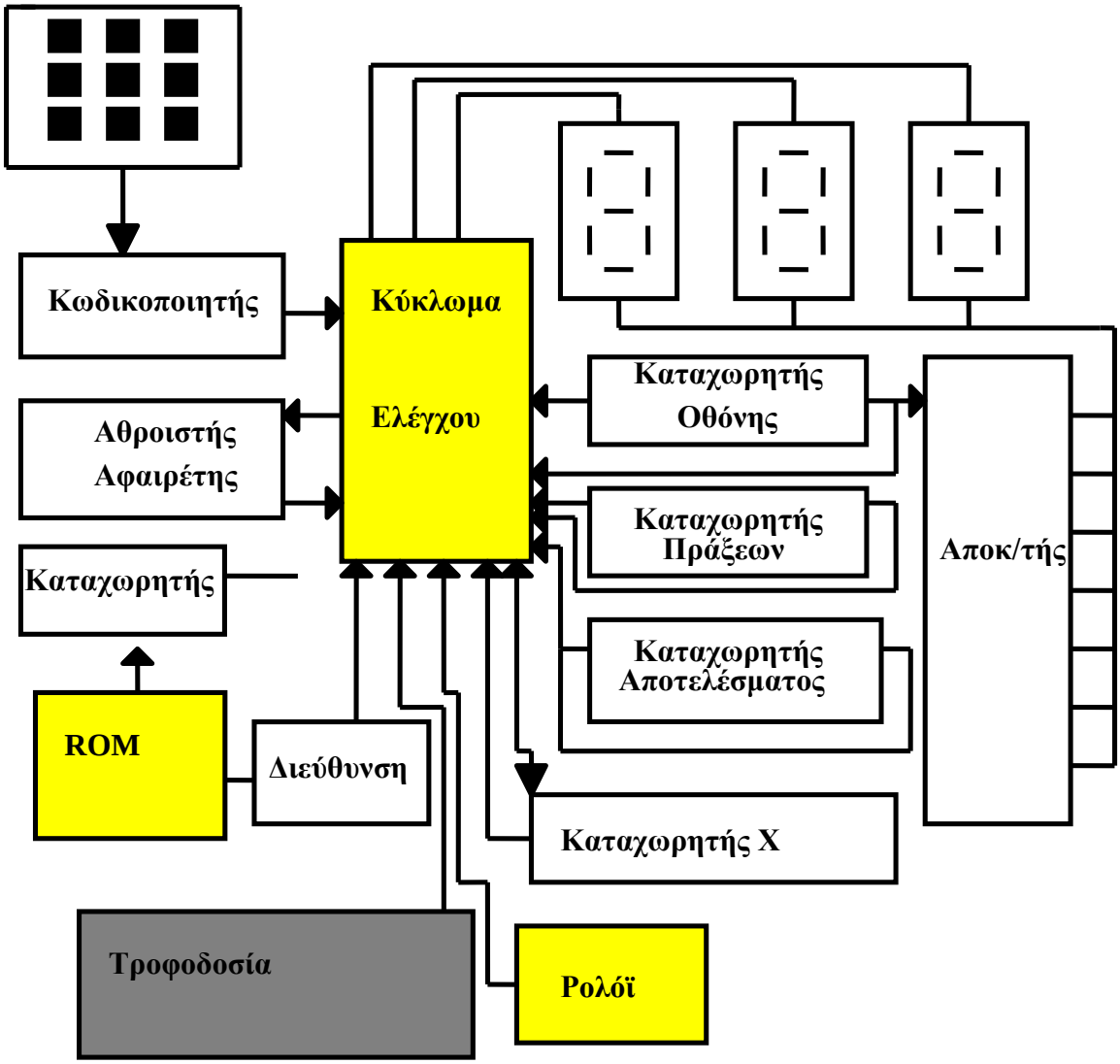
Κεφάλαιο 13. Ψηφιακά Συστήματα

Τα ψηφιακά συστήματα είναι περίπλοκα ηλεκτρονικά συστήματα τα οποία διαχειρίζονται και επεξεργάζονται μία πληροφορία και στην συνέχεια δίνουν ένα αποτέλεσμα. Χονδρικά μπορούμε να πούμε ότι ένα τέτοιο ψηφιακό κύκλωμα στην απλούστερη του μορφή απαρτίζεται από τρία διακριτά μέρη που είναι το κύκλωμα εισόδου, το κύκλωμα εξόδου και το κύκλωμα επεξεργασίας. Αυτή είναι η πιο απλή σύνθεση ενός ψηφιακού συστήματος γιατί τα περισσότερο σύνθετα περιέχουν και στοιχεία μνήμης όπου γίνεται αποθήκευση της πληροφορίας. Μία κεντρική μονάδα ελέγχου επιτρέπει την επικοινωνία των μονάδων που σχηματίζουν ένα ψηφιακό σύστημα μεταξύ τους όπως επίσης και την καλή λειτουργία τους. Στο παρακάτω περιγραφικό Block διάγραμμα του σχήματος δίνουμε τα στοιχεία από τα οποία αποτελείται ένα ψηφιακό σύστημα .



Υπολογιστές τσέπης

Ένας υπολογιστής τσέπης από αυτούς που χρησιμοποιούμε τώρα όλοι στην δουλειά, στο εργαστήριο κ.λ.π. είναι ένα πολύ σύνθετο ψηφιακό σύστημα. Αποτελείται από μία μικρή οθόνη που είναι η έξοδος ένα πληκτρολόγιο που είναι η είσοδος των δεδομένων την τροφοδοσία , και ένα ολοκληρωμένο κύκλωμα που είναι η μονάδα ελέγχου και επεξεργασίας των δεδομένων. Ας δούμε όμως τι συμβαίνει στο εσωτερικό ενός υπολογιστή τσέπης όταν πληκτρολογήσουμε δύο αριθμούς και θέλουμε να τους προσθέσουμε στην συνέχεια . Το παρακάτω κύκλωμα του σχήματος θα μας βοηθήσει για να έχουμε μία εικόνα και να καταλάβουμε την όλη διαδικασία λειτουργίας.

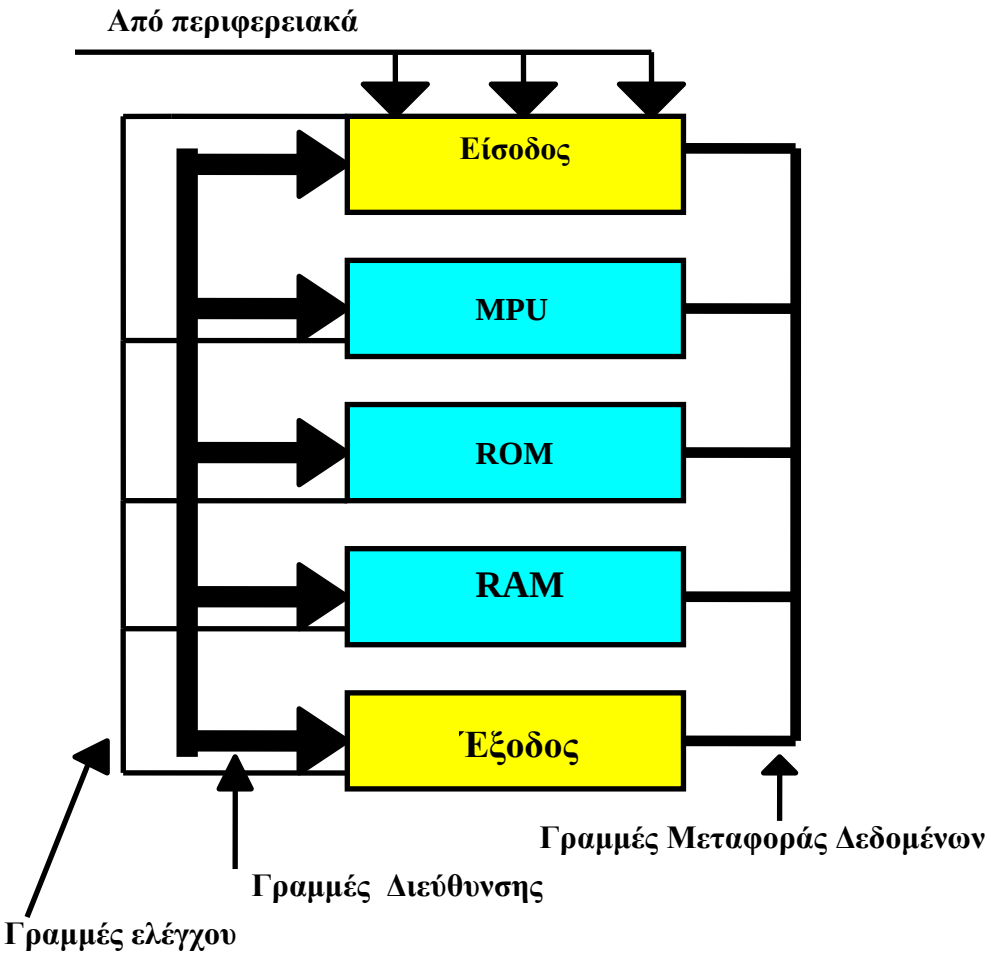


Το IC (chip) του υπολογιστή χωρίζεται σε διάφορα κομμάτια διαφορετικής λειτουργικότητας το καθένα. Αυτός ο τρόπος οργάνωσης είναι ένας από τους πολλούς δυνατούς τρόπους. Η καρδιά του τσιπ είναι το κομμάτι του αθροιστή αφαιρέτη που λειτουργεί όπως τον τετράμπιτο αθροιστή που έχουμε ήδη μελετήσει. Το ρολόι συνδέεται με όλα τα κομμάτια του τσιπ και εκπέμπει παλμούς σταθερής συχνότητας μόλις ανοίξουμε την τροφοδοσία. Ας υποθέσουμε τώρα ότι θέλουμε να κάνουμε την πρόσθεση 2+3. Μόλις πατήσουμε το 2 στο πληκτρολόγιο ο κωδικοποιητής μεταφέρει το 2 στο BCD 0010. Το 0010 κατευθύνεται στον καταχωρητή οθόνης διά μέσου του κυκλώματος ελέγχου και φυλάγεται. Αυτή η πληροφορία μεταφέρεται επίσης στον ενδείκτη επτά κομματιών και με την κατάλληλη ενεργοποίηση των φωτοδιόδων το 2 φαίνεται στην οθόνη. Στην συνέχεια πατάμε το + στο πληκτρολόγιο.

Αυτή η πληροφορία μεταφέρεται επίσης στον ενδείκτη επτά κομματιών και με την κατάλληλη ενεργοποίηση των φωτοδιόδων το 2 φαίνεται στην οθόνη. Στην συνέχεια πατάμε το + στο πληκτρολόγιο. Η πράξη αυτή μεταφέρεται με κωδικοποιημένη μορφή και φυλάγεται στον καταχωρητή Χ. Τώρα πατάμε το 3 και ο αποκωδικοποιητής μεταφέρει το 3 στον κώδικα BCD 0011. Το 0011 μεταφέρεται στον καταχωρητή οθόνης από το κύκλωμα ελέγχου όπως επίσης και στη οθόνη. Κατά τη διάρκεια αυτής της διαδικασίας το κύκλωμα ελέγχου μετέφερε το 2 στον καταχωρητή πράξεων. Μόλις πατήσουμε = το κύκλωμα ελέγχου ελέγχει το περιεχόμενο του καταχωρητή Χ για να δει τι πρέπει να κάνει. Ο καταχωρητής Χ λέει ότι πρέπει να γίνει η πρόσθεση δύο αριθμών στο BCD που βρίσκονται στους καταχωρητές Μικροϋπολογιστές πράξης και οθόνης αντίστοιχα. Έτσι το κύκλωμα ελέγχου υποβάλλει το περιεχόμενο των δύο καταχωρητών στην είσοδο του αθροιστή. Το αποτέλεσμα της πράξης είναι ο αριθμός 0101 στον κώδικα BCD και συλλέγεται στον καταχωρητή αποτελέσματος. Το κύκλωμα ελέγχου οδηγεί επίσης το αποτέλεσμα στην οθόνη. Αυτό το μικρό παράδειγμα ψηφιακού συστήματος φανερώνει τη λειτουργία των επί μέρους κυκλωμάτων που το αποτελούν, και τα οποία έχουν μελετηθεί ένα ένα χωριστά και μας δείχνει επίσης πως ο συνδυασμός τους σχηματίζει ένα σχετικά περίπλοκο σύστημα που είναι ο υπολογιστής τσέπης.

Μικροϋπολογιστές

Οι ηλεκτρονικοί υπολογιστές έγιναν ευρείας χρήσεως στο κοινό από τη δεκαετία του 50. Η πρόοδος που σημειώθηκε στον κατασκευαστικό τομέα της μικροηλεκτρονικής είχε σαν συνέπεια τη μείωση των διαστάσεων και του κόστους των ολοκληρωμένων κυκλωμάτων. Έτσι οδηγηθήκαμε στην σύλληψη κατασκευή και χρήση μίας νέας ηλεκτρονικής διάταξης που ονομάζεται**"microprocessor"**. Ο microprocessor (MPU "micro processing unit") είναι ένα ολοκληρωμένο σύστημα (IC) που περιέχει τις περισσότερες δυνατότητες επεξεργασίας που έχει ένας πολύ μεγάλος υπολογιστής. Ο MPU είναι μία μικρή αλλά πολύ πολύπλοκη **VLSI** ηλεκτρονική διάταξη ικανή να προγραμματιστεί. Ο MPU αποτελεί την καρδιά των μικροϋπολογιστών που βρίσκουμε στο σπίτι το γραφείο κλπ. Ένα τέτοιου είδους μικροϋπολογιστής αποτελείται από ένα πληκτρολόγιο, μία οθόνη και ένα εκτυπωτή για την έξοδο των αποτελεσμάτων. Η οργάνωση ενός μικροϋπολογιστή φαίνεται στο παρακάτω σχήμα.



Αποτελείται από πέντε βασικά κομμάτια που είναι η μονάδα εισόδου, οι μονάδες ελέγχου και η αριθμητική μονάδα που περιέχονται στον MPU, οι μονάδες μνήμης και εξόδου. Ο MPU ελέγχει όλες τις μονάδες του προηγουμένου σχήματος χρησιμοποιώντας τις γραμμές ελέγχου που φαίνονται αριστερά του προηγουμένου σχήματος. Οι γραμμές διεύθυνσης καθορίζουν μία ορισμένη θέση στην μνήμη, στην έξοδο η στην είσοδο. Η μνήμη ROM περιέχει ένα πρόγραμμα από κωδικοποιημένες εντολές οι οποίες καθορίζουν με ακρίβεια τι ακριβώς πρέπει να κάνει ο MPU.

ΑΣΚΗΣΕΙΣ

1. Φτιάξετε το λογικό διάγραμμα ενός Flip-Flop τύπου RS με ρολόι χρησιμοποιώντας τέσσερις πύλες NAND.

2. Έστω ένα Flip Flop τύπου JK' δηλαδή ένα JK αλλά με ένα αντιστροφέα μεταξύ του K' και του K.

α. Βρείτε τον πίνακα αληθείας του Flip Flop.

β. Βρείτε τη χαρακτηριστική του εξίσωση.

γ. Δείξετε ότι αν ενώσουμε τις εισόδους μεταξύ τους θα πάρουμε τελικά ένα Flip Flop τύπου D.

3. Σχεδιάστε ένα μετρητή χρησιμοποιώντας Flip Flop τύπου T ικανό να μετράει την παρακάτω ακολουθία μετρήσεων και ξανά από την αρχή.

4. Βρείτε τον αριθμό Data Lines (γραμμών) επικοινωνίας και τον αριθμό γραμμών διεύθυνσης (address lines) μίας RAM IC χωρητικότητας 4Kbits εάν:

α. Έχει οργάνωση ανά bit.

b. Έχει οργάνωση κατά λέξη με μήκος m=8 bits.

5. Σχεδιάστε ένα σύγχρονο καταχωρητή ολίσθησης με Flip Flop τύπου D.

6. Το περιεχόμενο ενός τετράμπιτου καταχωρητή ολίσθησης είναι αρχικά ίσο με 0111. Το ολισθαίνουμε πέντε φορές δεξιά δίνοντας του συγχρόνως στην είσοδο τον αριθμό 1111. Ποιό είναι το περιεχόμενο του καταχωρητή μετά από κάθε ολίσθηση.

7. Σχεδιάστε ένα τετράμπιτο δυαδικό μετρητή ικανό να μετράει ανάποδα. δηλαδή να ξεκινάει από το 1111 και να κατεβαίνει μέχρι το 0000 και ξανά από την αρχή.

8. Σχεδιάστε ένα κύκλωμα σύγχρονου μετρητή BCD με Flip Flop τύπου JK.

9. Σχεδιάστε ένα κύκλωμα σειριακού αθροιστή με δύο καταχωρείται και ένα Flip Flop τύπου D. Τι αλλαγές χρειάζονται στο κύκλωμα για να γίνει σειριακός αφαιρέτης.

10. Σύμφωνα με το μετατροπέα σήματος από ψηφιακό σε αναλογικό που είδαμε προηγουμένως δώστε την έξοδο του αναλογικού σήματος σε Volts αν η τάση αναφοράς είναι V=16V και στην είσοδο δίνω τα ακόλουθα ψηφιακά σήματα.

1. 1000
2. 1010
3. 1100

Ένας μετατροπέας D/A τι θα έδινε στην έξοδο αν V=64V και τα σήματα εισόδου είναι:

α. 1001000

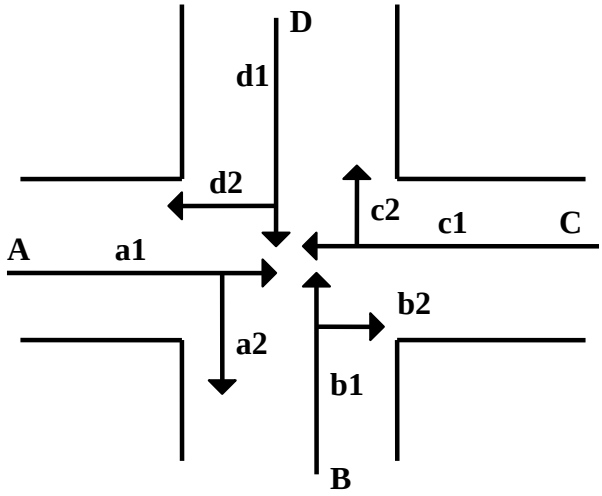
β. 100111

Αναπτύξετε το αποτέλεσμα.

15. Ασκήσεις από επιλεγμένα θέματα εξετάσεων

1. Σχεδιάστε ένα συνδυαστικό κύκλωμα με τέσσερεις εισόδους και τρεις εξόδους το οποίο θα συγκρίνει δύο δίμπιτους αριθμούς A1A0 και B1B0 αντίστοιχα. Οι τρείς έξοδοι F1, F2 και F3 θα δείχνουν το αποτέλεσμα της σύγκρισης. Έτσι αν F1=1 ο A1A0 είναι μεγαλύτερος από τον B1B0, αν F2=1 ο A1A0 είναι ίσος με τον B1B0 και τέλος αν F3=1 ο A1A0 είναι μικρότερος από τον B1B0. Βρείτε τις κανονικές μορφές των λογικών συναρτήσεων F1, F2, F3 και κατόπιν τις ελάχιστες μορφές τους. Υλοποιήστε το κύκλωμα μόνο με πολυπλέκτες 8x1.

2. Θέλουμε να κατασκευάσουμε το ψηφιακό κύκλωμα το οποίο θα ελέγχει την λειτουργία των φανών κυκλοφορίας σε ένα σταυροδρόμι όπως φαίνεται στο παρακάτω σχήμα.



Οι τέσσερεις δρόμοι A,B,C,D οδηγούν και προς τις δύο κατευθύνσεις. Τα αυτοκίνητα που κινούνται πάνω στους δρόμους A,B,C,D μπορούν να προχωρήσουν ευθεία η να πάνε δεξιά ανάλογα με την κατάσταση των φαναριών a1, a2, b1, b2, c1, c2, d1, και d2. Τα a1, b1, , c1, και d1 αν είναι πράσινα (λογικό 1) επιτρέπουν την κίνηση ευθεία και αν είναι κόκκινα όχι (λογικό 0). Αντίστοιχα τα a2, b2, c2, d2 επιτρέπουν την κίνηση δεξιά αν είναι πράσινα η την απαγορεύουν αν είναι κόκκινα. Έτσι αν το a1 είναι πράσινο το αυτοκίνητο A μπορεί

να πάει ευθεία και αν είναι συγχρόνως αναμμένο πράσινο το a2 μπορεί να προχωρήσει κατ' επιλογή δεξιά. Το ίδιο ισχύει για τα υπόλοιπα αυτοκίνητα και τα αντίστοιχα φανάρια τους. Η κυκλοφορία στο σταυροδρόμι υπόκειται σε ορισμένους περιορισμούς. 1] Ένα μόνο αυτοκίνητο μπορεί να κινηθεί ανά διεύθυνση. Δεν γίνεται δηλαδή ο Α να πηγαίνει ευθεία και ο Β να στρίβει δεξιά. Η προτεραιότητα κίνησης των αυτοκινήτων είναι μέγιστη για αυτά που κινούνται στην κατεύθυνση Α μικρότερη για τα C ακόμα μικρότερη για τα Β και τέλος ελάχιστη για τα D. Βρείτε τις εξισώσεις του κυκλώματος. Απλοποιήστε τις. Πόσες πύλες NAND χρειάζονται γιά να υλοποιηθεί το κύκλωμα. Υλοποιήστε το με αποκωδικοποιητή και πύλες της αρεσκείας σας.

3. Σχεδιάστε ένα σύγχρονο μετρητή με Flip-Flop τύπου JK ικανό να ακολουθεί την παρακάτω ακολουθία καταστάσεων 011, 111, 000, 110, 100, 010, και ξανά από την αρχή. Χρησιμοποιείστε τον ελάχιστο αριθμό βοηθητικών πυλών αν χρειάζονται.

4. Σχεδιάστε ένα μετρητή με Flip-Flop τύπου JK ικανό να ακολουθεί την παρακάτω ακολουθία καταστάσεων 000, 111, 101, 110, 100, 011, και ξανά από την αρχή. Χρησιμοποιείστε τον ελάχιστο αριθμό βοηθητικών πυλών αν χρειάζονται.

5. Σχεδιάστε ένα μετρητή με Flip-Flop τύπου JK ικανό να ακολουθεί την παρακάτω ακολουθία καταστάσεων 001, 111, 000, 110, 100, 011, και ξανά από την αρχή. Χρησιμοποιείστε τον ελάχιστο αριθμό βοηθητικών πυλών αν χρειάζονται.

6. Σχεδιάστε ένα σύγχρονο μετρητή ικανό να μετράει στον κώδικα Excess-3 με Flip-Flop τύπου JK και τον ελάχιστο αριθμό πυλών.

7. Δίνονται οι παρακάτω λογικές συναρτήσεις:

a. • F1(A,B,C,D)= Σ(0,2,4,8,9,10,14,15)

b. • F2(X,Y,Z,W)= Π(0,1,2,3,7,8,9,10)

α] να βρεθούν οι δύο ελάχιστες μορφές κάθε συνάρτησης

β] να υλοποιηθούν με τον ελάχιστο αριθμό πυλών NOR η F1 και τον ελάχιστο αριθμό NAND η F2.

γ] αν μία πύλη δύο εισόδων έχει 20mW κατανάλωση και 20ns καθυστέρηση , μία τριών εισόδων 30mW κατανάλωση και 30ns καθυστέρηση αντίστοιχα, όμοια αναλογία και γιά πύλες με μεγαλύτερο αριθμό εισόδων να βρεθεί η ολική κατανάλωση και καθυστέρηση των δύο παραπάνω κυκλωμάτων.

8. Σχεδιάστε ένα μετρητή σύγχρονο με Flip-Flop τύπου Τ ικανό να μετράει στον κώδικα ΔΕΛΤΑ. (*ο κώδικας ΔΕΛΤΑ είναι ο κώδικας BCD+4*) Η υλοποίηση θα γίνει με τον αναγκαίο αριθμό Flip-Flop τύπου Τ και τον ελάχιστο αριθμό πυλών NAND μόνο. Συνδέστε στην συνέχεια τέτοια κυκλώματα έτσι ώστε να σχηματίσετε ένα μετρητή ικανό να μετράει μέχρι το 9999 στον ίδιο κώδικα.

9. Να βρεθεί ο πίνακας αληθείας της παρακάτω συνάρτησης: $F(A,B,C) = (A \oplus B).C + \overline{A}(B \oplus C)$. Στην συνέχεια να την υλοποιήσετε με ένα πολυπλέκτη 4x1 και αντιστροφείς.

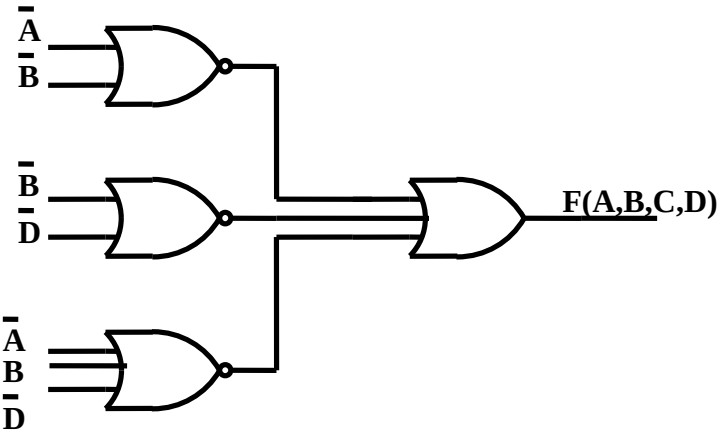
10. Δίνεται η παρακάτω λογική συνάρτηση F(A,B,C,D). Να βρεθεί η ελάχιστη διεπίπεδη υλοποίηση NOR-NOR .

F(A,B,C,D)=Σ(2,3,4,5,10,11,12,14)

11. Σχεδιάστε απλοποιείστε και υλοποιείστε ένα λογικό κύκλωμα το οποίο δέχεται στην είσοδο ένα αριθμό στον κώδικα **BCD** και παράγει στην έξοδο το λογικό **1** μόνο όταν ο αριθμός εισόδου διαιρείται με το 3 η το **4**. Χρησιμοποιείστε πίνακα Karnaugh για την απλοποίηση. Όλες οι πύλες είναι διαθέσιμες για την υλοποίηση.

12. Σχεδιάστε ένα σύγχρονο μετρητή με flip-flop τύπου **T** ικανό να μετράει στον κώδικα **Excess 3.** Για την υλοποίηση όλες οι πύλες είναι διαθέσιμες.

13. Δίνεται το παρακάτω κύκλωμα:



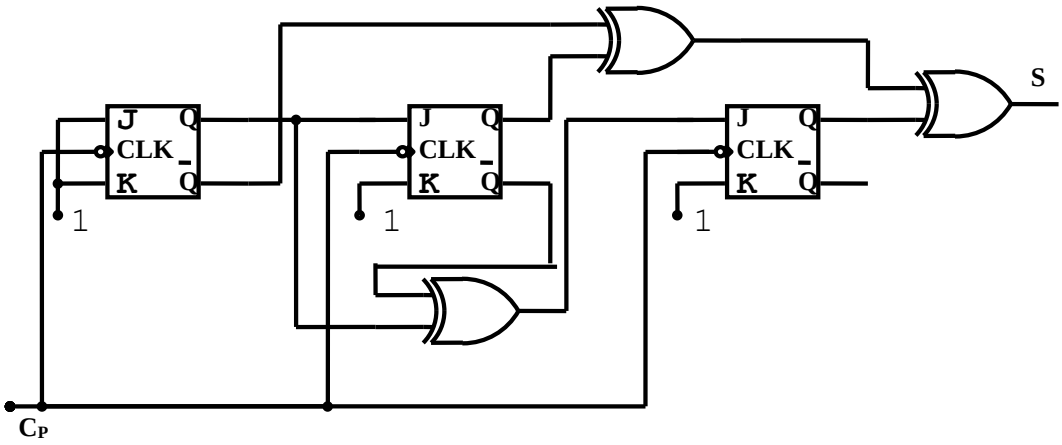
α) Βρείτε την συνάρτηση **F (A,B,C,D)** στην έξοδο.

β) Υλοποιήστε ξανά την **F** με τον ελάχιστο αριθμό πυλών **NOR**. Τα συμπληρώματα είναι διαθέσιμα στην είσοδο.

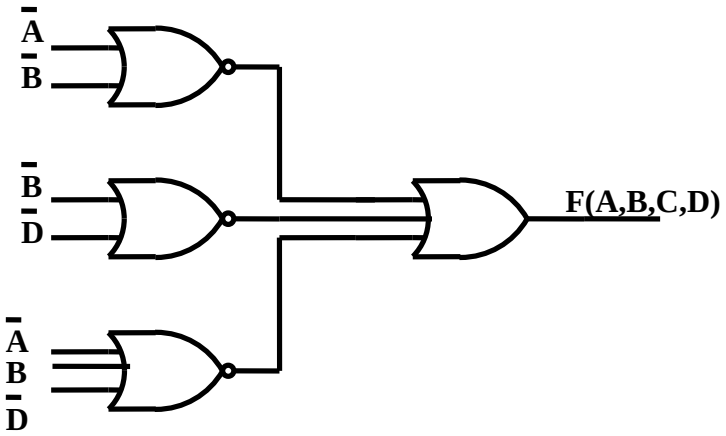
γ) Υποθέτουμε ότι η κατανάλωση μια πύλης δύο εισόδων είναι **a** και μίας **N** εισόδων **(a.N)/2.** Υποθέτουμε επίσης ότι η κατανάλωση εξαρτάται μόνο από τον αριθμό εισόδων μια πύλης (δεν εξαρτάται από το είδος της πύλης). Υποθέτουμε επίσης ότι η καθυστέρηση είναι **b** για όλες τις πύλες. Αν ισχύουν όλες αυτές οι υποθέσεις ποιο κύκλωμα είναι περισσότερο γρήγορο και ποιο περισσότερο οικονομικό, το αρχικό η αυτό της ερώτησης **β**.

δ) Υλοποιήστε την **F** με ένα μόνο πολυπλέκτη 8x1 αφού πρώτα εξηγήσετε την λειτουργία του.

14. Σχεδιάστε τις χρονικές μεταβολές της εξόδου S για δέκα παλμούς του ρολογιού. Υποθέτω ότι έχω αρχικά **(t=0)** $\mathbf{Q_A=Q_B=Q_C=0}$**.**



15. Δίνεται το παρακάτω κύκλωμα:



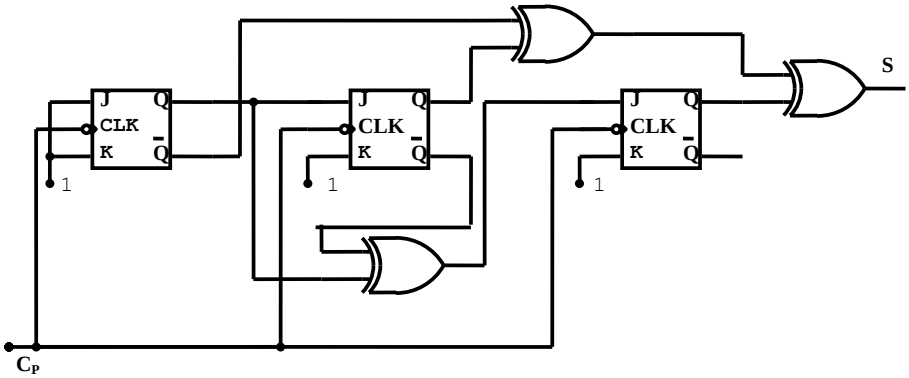
α) Βρείτε την συνάρτηση **F (A,B,C,D)** στην έξοδο.

β) Υλοποιήστε ξανά την **F** με τον ελάχιστο αριθμό πυλών **NAND**. Τα συμπληρώματα είναι διαθέσιμα στην είσοδο.

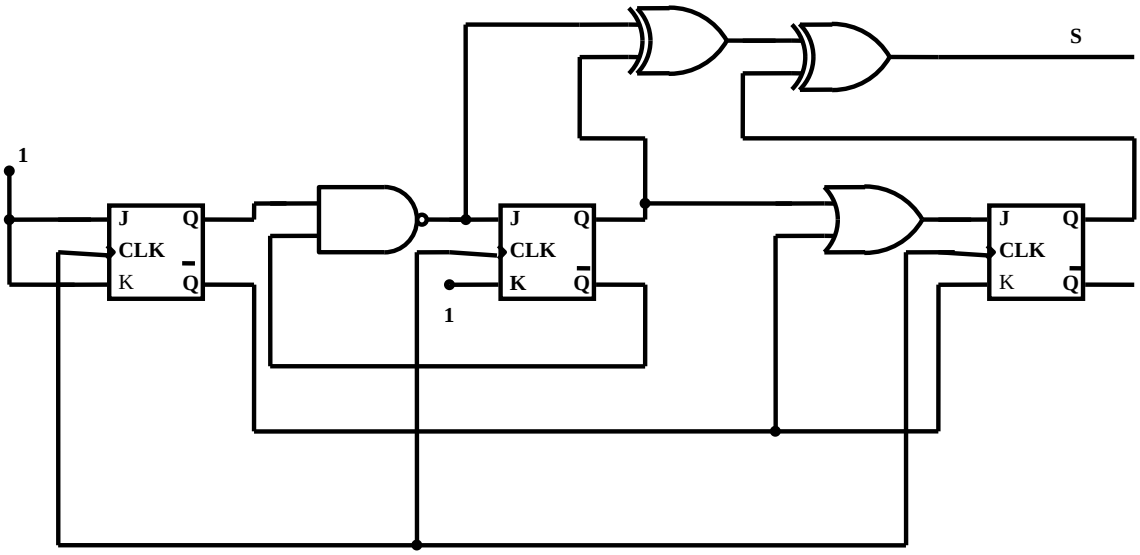
γ) Υποθέτουμε ότι η κατανάλωση μια πύλης δύο εισόδων είναι **a** και μίας **N** εισόδων 2**(a.N)/3.** Υποθέτουμε επίσης ότι η κατανάλωση εξαρτάται μόνο από τον αριθμό εισόδων μια πύλης (δεν εξαρτάται από το είδος της πύλης). Υποθέτουμε επίσης ότι η καθυστέρηση είναι **b** για όλες τις πύλες. Αν ισχύουν όλες αυτές οι υποθέσεις ποιο κύκλωμα είναι περισσότερο γρήγορο και ποιο περισσότερο οικονομικό, το αρχικό η αυτό της ερώτησης **β**.

δ) Υλοποιήστε την **F** με ένα μόνο πολυπλέκτη 8x1 αφού πρώτα εξηγήσετε την λειτουργία του.

16. Σχεδιάστε ακρίβεια τις χρονικές μεταβολές της εξόδου S για δέκα παλμούς του ρολογιού. Υποθέτω ότι έχω αρχικά **(t=0)** $\mathbf{Q_A=Q_B=Q_C=0}$.

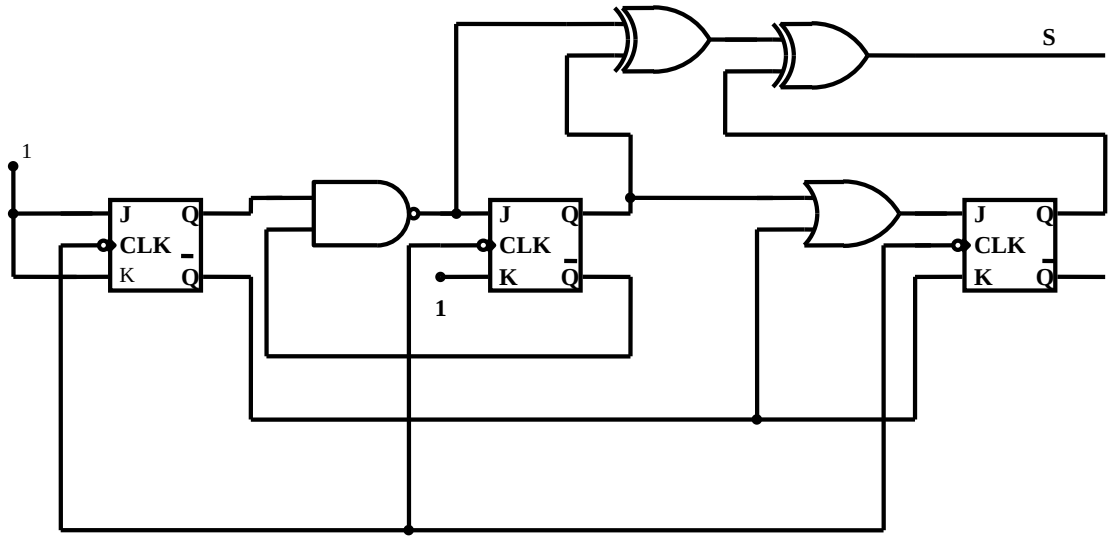


17. Δίνεται το παρακάτω σύγχρονο ακολουθιακό κύκλωμα. Σχεδιάστε τις μεταβολές της εξόδου S αν εφαρμόσουμε στο κύκλωμα δέκα παλμούς του ρολογιού. Υποθέτουμε ότι την χρονική στιγμή t=0 όλα τα Q είναι μηδέν. Τα JK είναι θετικής ακμής.

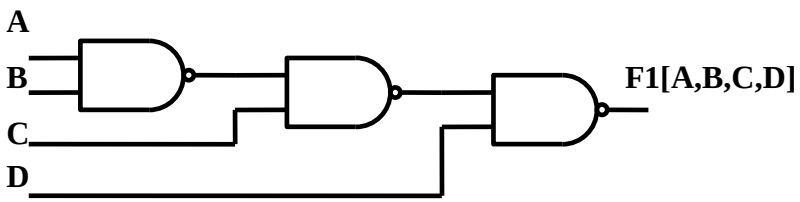


18. Δίνεται το παρακάτω σύγχρονο ακολουθιακό κύκλωμα. Σχεδιάστε τις μεταβολές της εξόδου S αν εφαρμόσουμε στο κύκλωμα δέκα

παλμούς του ρολογιού. Υποθέτουμε ότι την χρονική στιγμή t=0 όλα τα Q είναι μηδέν. Τα JK είναι αρνητικής ακμής.



19. Δίνεται το παρακάτω κύκλωμα



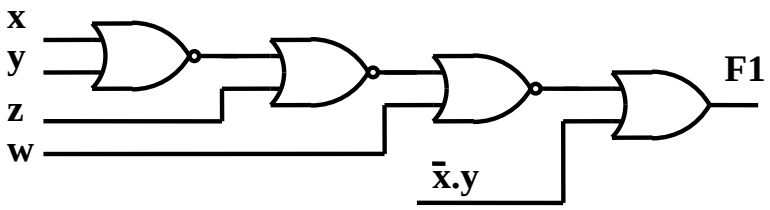
α] Να βρεθεί η συνάρτηση F1 στην έξοδο.

β] Να βρεθούν οι δύο κανονικές της μορφές.

γ] Να βρεθούν οι δύο ελάχιστες μορφές. Αν οι πύλες NOR και NAND με ίδιο αριθμό εισόδων έχουν ίδια κατανάλωση ποια θα ήταν η ποιο οικονομική υλοποίηση της F1.

δ] Υλοποιήστε την συνάρτηση F1 με ένα πολυπλέκτη 8x1 και πύλες NOT αποκλειστικά.

20. Δίνεται το παρακάτω κύκλωμα



α] Να βρεθεί η συνάρτηση F1 στην έξοδο.

β] Να βρεθούν οι δύο κανονικές της μορφές.

γ] Να βρεθούν οι δύο ελάχιστες μορφές. Αν οι πύλες NOR και NAND με ίδιο αριθμό εισόδων έχουν ίδια κατανάλωση ποια θα ήταν η ποιο οικονομική υλοποίηση της F1.

δ] Υλοποιήστε την συνάρτηση F1 με ένα πολυπλέκτη 8x1 και πύλες NOT αποκλειστικά.

21. Δίνονται οι δύο παρακάτω τετράμπιτοι κώδικες Κ1 και Κ2 οι οποίοι κωδικοποιούν τα δέκα ψηφία του δεκαδικού συστήματος αρίθμησης. Σχεδιάστε ένα ψηφιακό κύκλωμα το οποίο θα έχει είσοδο τον κώδικα Κ2 και έξοδο τον Κ1 αποκλειστικά με τον ελάχιστο αριθμό πυλών NAND.

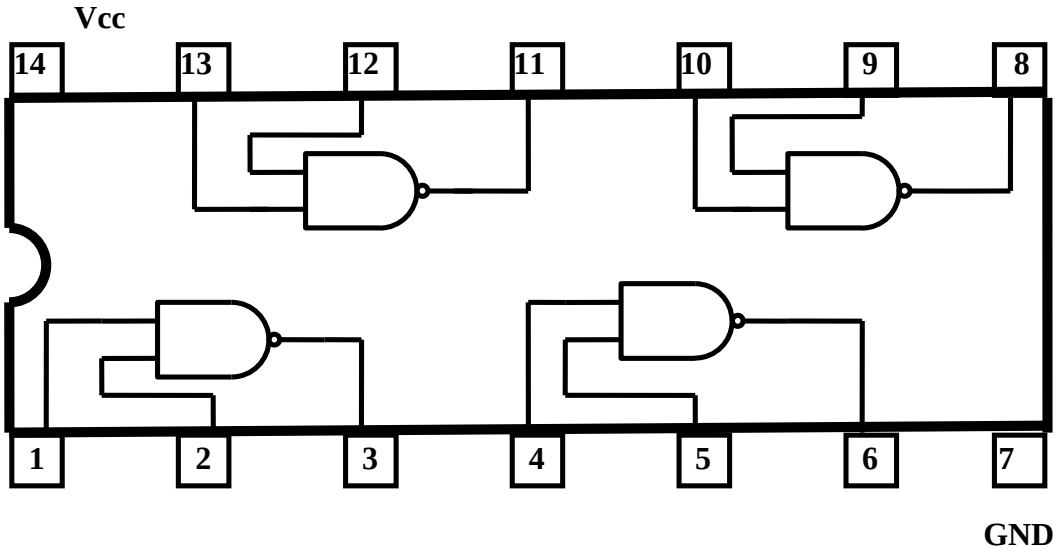
22. Σχεδιάστε ένα ψηφιακό κύκλωμα ικανό να μετατρέπει ένα τετράμπιτο δυαδικό αριθμό στο BCD. Απλοποιήστε στην ελάχιστη μορφή όλες τις συναρτήσεις στην έξοδο και δώστε τον ελάχιστο αριθμό πυλών NAND που χρειάζεται για να υλοποιηθεί το κύκλωμα. Στην συνέχεια αφού πρώτα εξηγήστε την λειτουργία του αποκωδικοποιητή χρησιμοποιήστε ένα (αποκωδικοποιητή) 4 σε 16 και πύλες OR για την υλοποίηση του κυκλώματος. Υπενθυμίζουμε ότι ο κώδικας BCD κωδικοποιεί τα 10 ψηφία του δεκαδικού συστήματος αρίθμησης (0,1,.....,9).

23. Σχεδιάστε ένα μετρητή ικανό να ακολουθεί την δυαδική σειρά 0, 1, 3, 6, 7, 5 και ξανά από την αρχή. Χρησιμοποιήστε Flip-Flop τύπου Τ και πύλες NAND αποκλειστικά .

Κεφάλαιο 15. Βασικά Ο.Κ. TTL

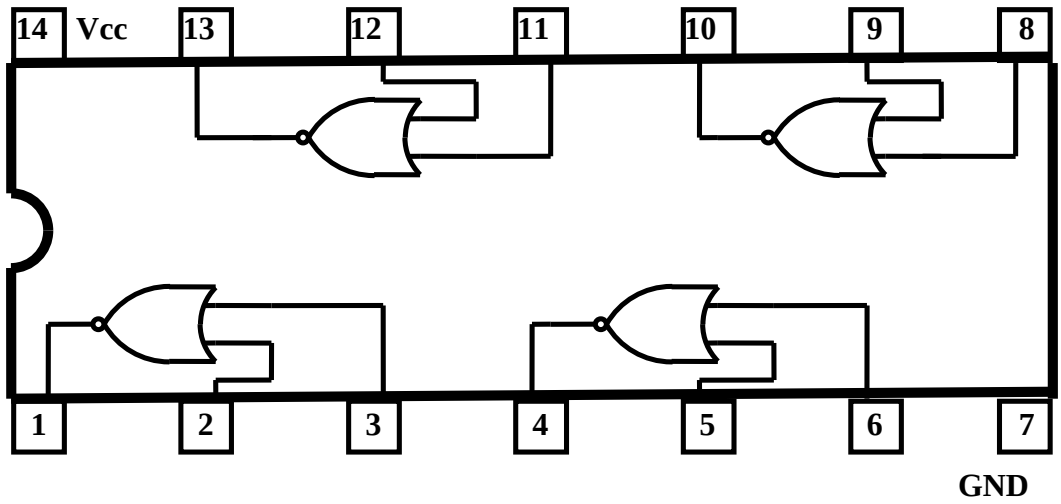
7400 TTL

Ένα από τα περισσότερο γνωστά τσιπ TTL το 7400. Περιέχει τέσσερις πύλες των δύο εισόδων η κάθε μία οι οποίες υλοποιούν τη λογική πράξη NAND. Το διάγραμμα εισόδων και εξόδων των τεσσάρων πυλών φαίνεται παρακάτω όπως επίσης η τροφοδοσία και η γείωση. Το όλο διάγραμμα ονομάζεται "pin out "



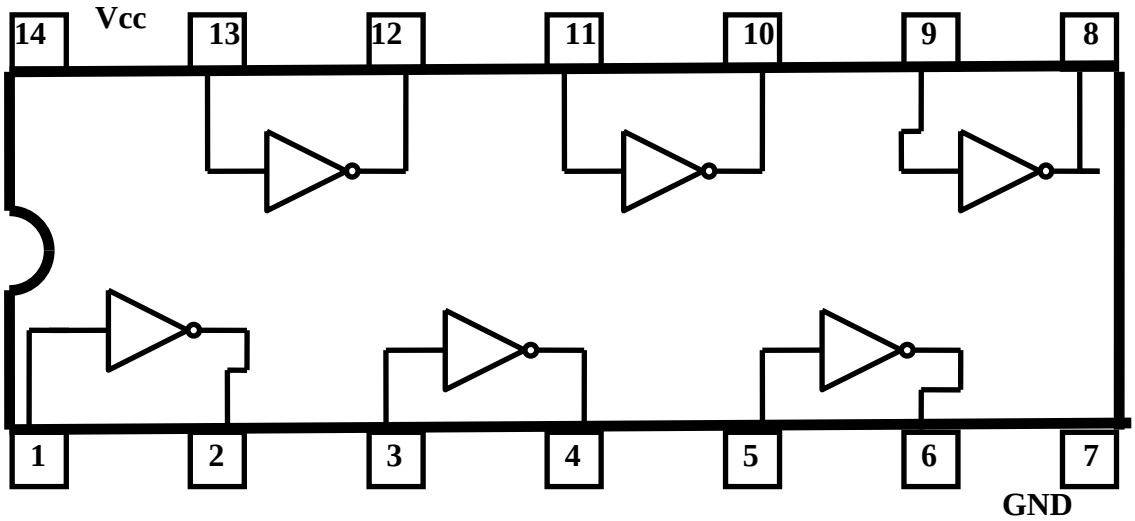
7402 TTL

Ένα τσιπ TTL το 7402. Περιέχει τέσσερις πύλες των δύο εισόδων η κάθε μία οι οποίες υλοποιούν τη λογική πράξη NOR. Το διάγραμμα εισόδων και εξόδων των τεσσάρων πυλών φαίνεται παρακάτω όπως επίσης η τροφοδοσία και η γείωση.



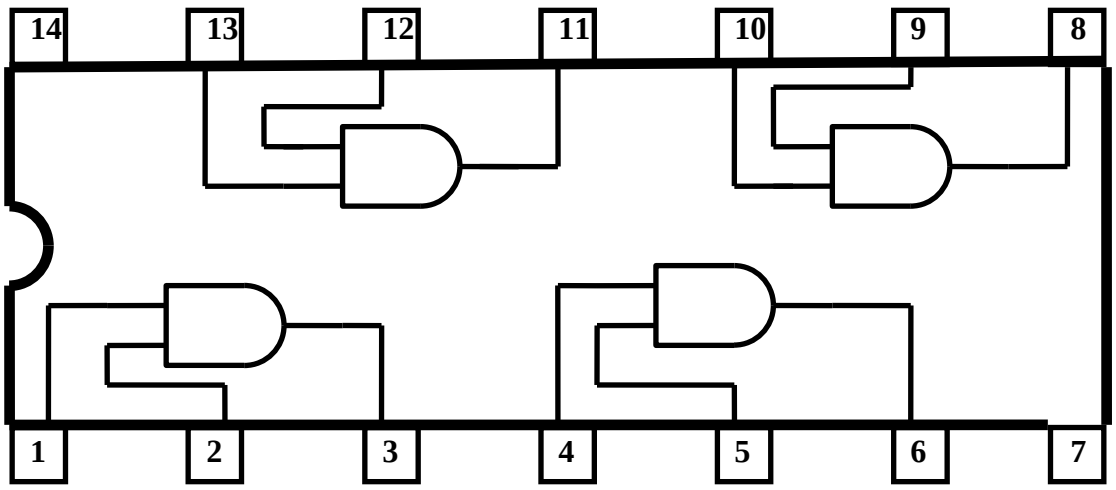
7404 TTL

Τσιπ TTL το 7404. Περιέχει έξη αντιστροφείς οι οποίοι υλοποιούν τη λογική πράξη NOT. Το διάγραμμα εισόδων και εξόδων των έξη λογικών πυλών φαίνεται παρακάτω όπως επίσης η τροφοδοσία και η γείωση



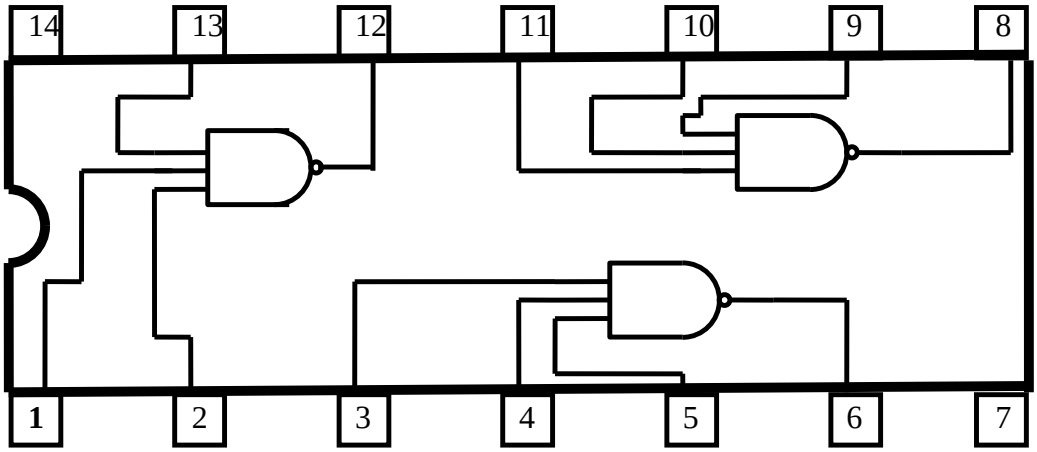
7408 TTL

Ένα τσιπ TTL το 7408. Περιέχει τέσσερις πύλες των δύο εισόδων η κάθε μία οι οποίες υλοποιούν τη λογική πράξη AND. Το διάγραμμα εισόδων και εξόδων των τεσσάρων πυλών φαίνεται παρακάτω όπως επίσης η τροφοδοσία και η γείωση.



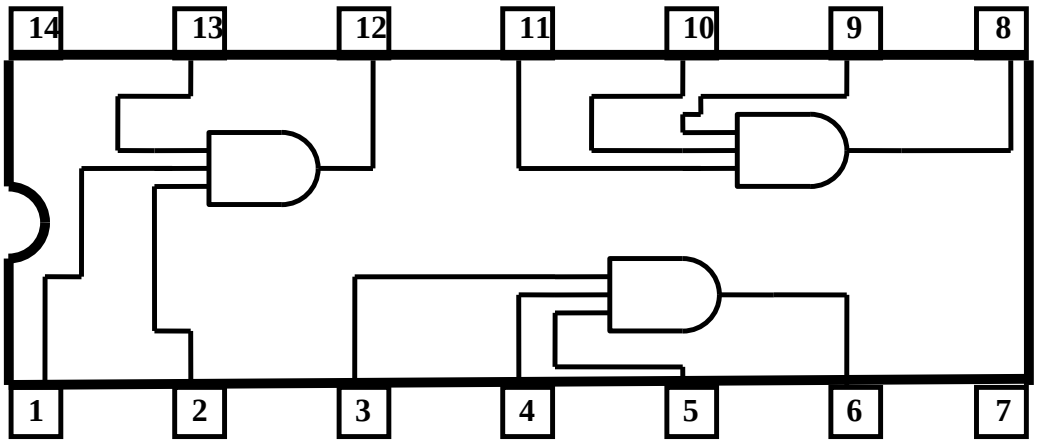
7410 TTL

Ένα από τα περισσότερο γνωστά τσιπ TTL το 7410. Περιέχει τρεις πύλες των τριών εισόδων η κάθε μία οι οποίες υλοποιούν τη λογική πράξη NAND. Το διάγραμμα εισόδων και εξόδων των τριών πυλών φαίνεται παρακάτω όπως επίσης η τροφοδοσία και η γείωση.



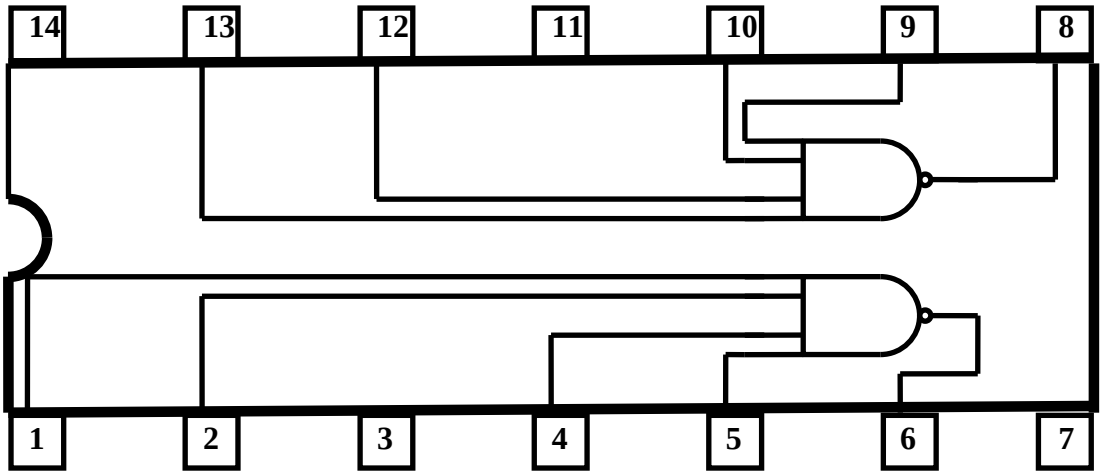
7411 TTL

Ένα τσιπ TTL το 7411. Περιέχει τρεις πύλες των τριών εισόδων η κάθε μία οι οποίες υλοποιούν τη λογική πράξη AND. Το διάγραμμα εισόδων και εξόδων των τριών πυλών φαίνεται παρακάτω όπως επίσης η τροφοδοσία και η γείωση.



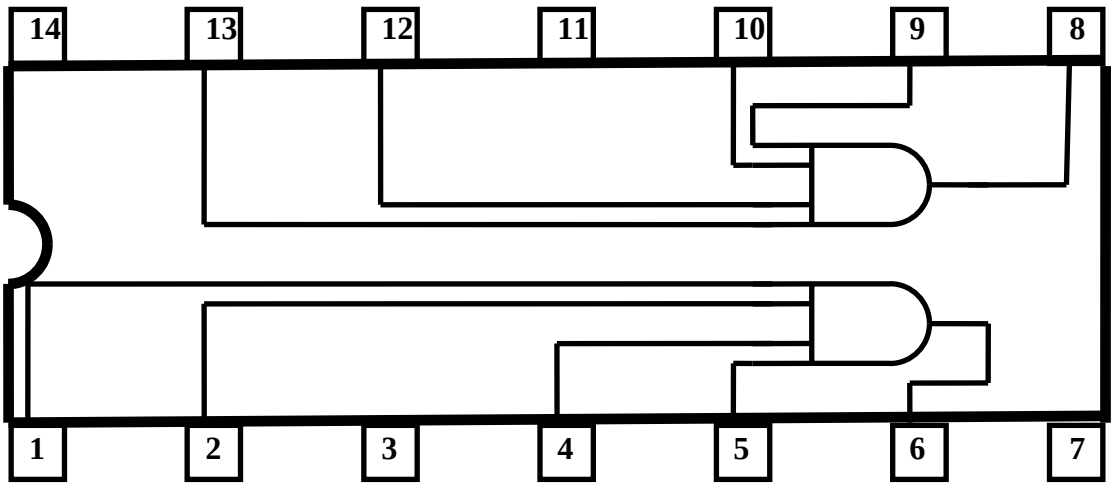
7420 TTL

Ένα γνωστό τσιπ TTL το 7420. Περιέχει δύο πύλες των τεσσάρων εισόδων η κάθε μία οι οποίες υλοποιούν τη λογική πράξη NAND. Το διάγραμμα εισόδων και εξόδων των τριών πυλών φαίνεται παρακάτω όπως επίσης η τροφοδοσία και η γείωση.



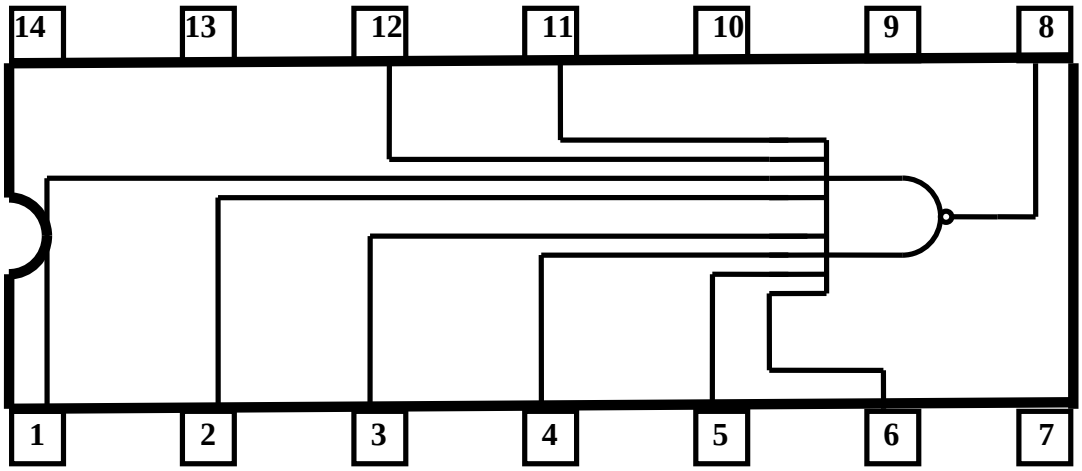
7421 TTL

Ένα τσιπ TTL το 7421. Περιέχει δύο πύλες των τεσσάρων εισόδων η κάθε μία οι οποίες υλοποιούν τη λογική πράξη AND. Το διάγραμμα εισόδων και εξόδων των τριών πυλών φαίνεται παρακάτω όπως επίσης η τροφοδοσία και η γείωση.



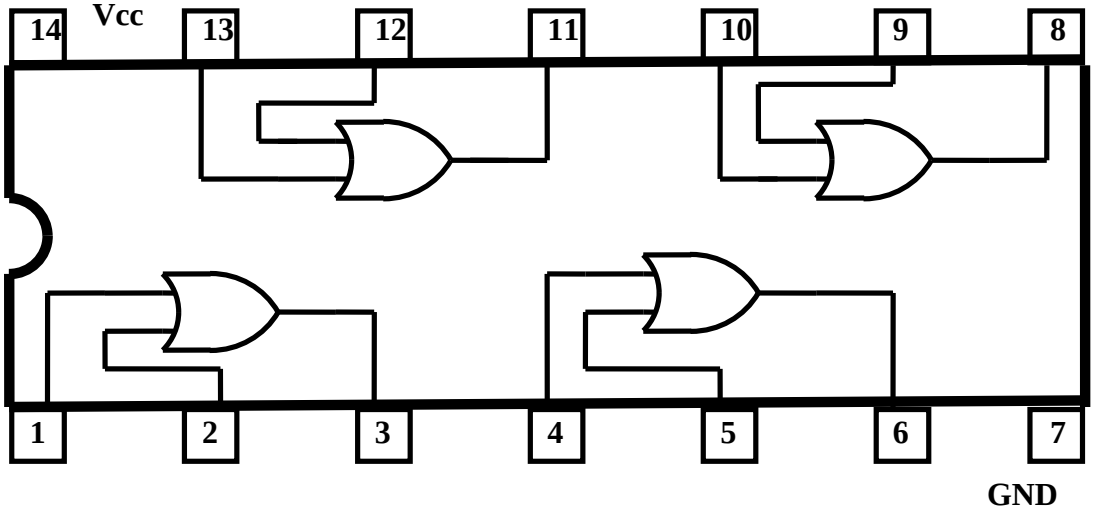
7430 TTL

Ένα γνωστό τσιπ TTL το 7430. Περιέχει μία πύλη οκτώ εισόδων η οποία υλοποιεί τη λογική πράξη NAND. Το διάγραμμα εισόδων και εξόδων της πύλης φαίνεται παρακάτω όπως επίσης η τροφοδοσία και η γείωση.



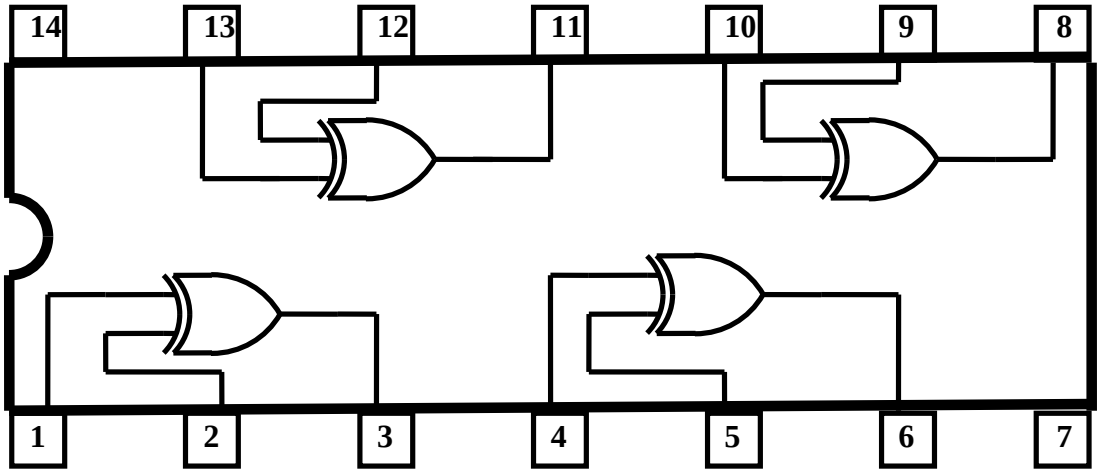
7432 TTL

Τσιπ TTL το 7432. Περιέχει τέσσερις πύλες των δύο εισόδων η κάθε μία οι οποίες υλοποιούν τη λογική πράξη OR. Το διάγραμμα εισόδων και εξόδων των τεσσάρων πυλών φαίνεται παρακάτω όπως επίσης η τροφοδοσία και η γείωση.



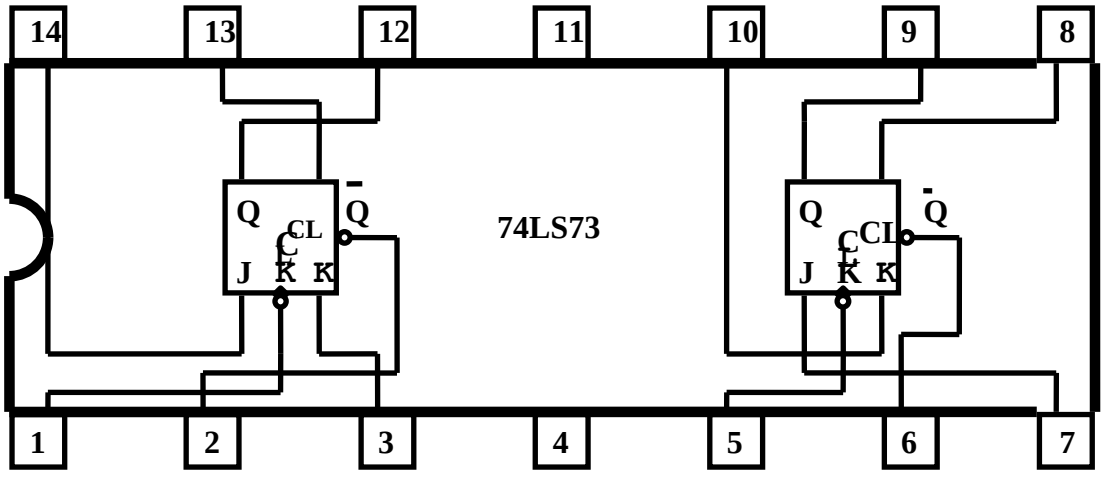
7486 TTL

Ένα τσιπ TTL το 7486. Περιέχει τέσσερις πύλες των δύο εισόδων η κάθε μία οι οποίες υλοποιούν τη λογική πράξη XOR. Το διάγραμμα εισόδων και εξόδων των τεσσάρων πυλών φαίνεται παρακάτω όπως επίσης η τροφοδοσία και η γείωση.



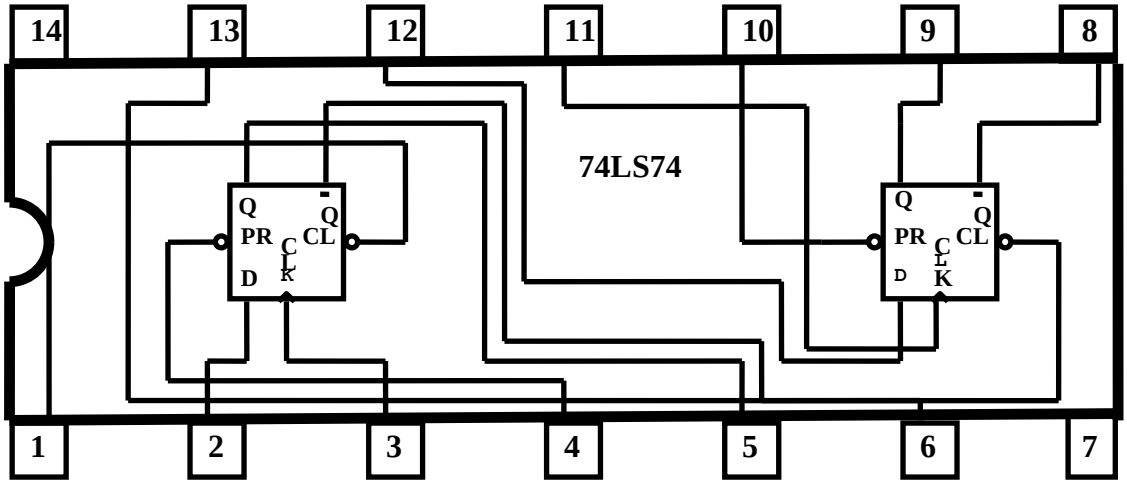
7473 TTL

Το 7473 της TTL περιέχει δύο Flip-Flop τύπου JK. Το διάγραμμα εισόδων και εξόδων των Flip-Flop φαίνεται παρακάτω όπως επίσης η τροφοδοσία και η γείωση.



7474 TTL

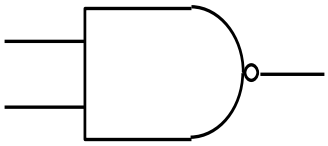
Το 7474 της TTL περιέχει δύο Flip-Flop τύπου D. Το διάγραμμα εισόδων και εξόδων των Flip-Flop φαίνεται παρακάτω όπως επίσης η τροφοδοσία και η γείωση.



ΒΙΒΛΙΟΓΡΑΦΙΑ

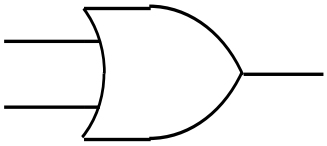
1. ΨΗΦΙΑΚΗ ΣΧΕΔΙΑΣΗ M.MORRIS MANO Εκδόσεις Τ.Ε.Ε.
2. ΨΗΦΙΑΚΗ ΣΧΕΔΙΑΣΗ M. MANO M.Cilleti Εκδόσεις Παπασωτηρίου
3. ΨΗΦΙΑΚΑ ΗΛΕΚΤΡΟΝΙΚΑ W.KLEITZ Εκδόσεις Τζιόλα
4. ΣΧΕΔΙΑΣΗ ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ S.BROWN, Z.VRANESIC Εκδόσεις Τζιόλα
5. ΛΟΓΙΚΗ ΣΧΕΔΙΑΣΗ ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ Κ. Παπαοδυσσεύς, Μ. Έξαρχος, Δ.Αραμπατζής, Φ.Γιαννόπουλος Εκδόσεις Τζιόλα
6. PRINCIPLES OF ELECTRONIC INSTRUMENTATION A.JAMES DIEFENDERFER HOLT AND SAUNDERS International Editions
7. INTEGRATED CIRCUITS IN DIGITAL ELECTRONICS ARPAD BARNA AND DANI.PORAT John Wiley Publications
8. ΟΛΟΚΛΗΡΩΜΕΝΗ ΗΛΕΚΤΡΟΝΙΚΗ MILLMAN ΧΑΛΚΙΑ Εκδόσεις Τ.Ε.Ε.
9. DIGITAL SYSTEM DESIGN BARRY WILKINSON Prentice Hall, 1992
10. ANALYSIS AND DESIGN OF SEQUENTIAL DIGITAL SYSTEMS LIND AND NELSON The MacMillan Press
11. PHYSICS OF SEMICONDUCTORS DEVICES S.M.SZE Εκδόσεις Wiley Easten 1987
12. DIGITAL,ELECTRONICS WILLIAM GOTHMANN Εκδόσεις Prentice Hall
13. ΨΗΦΙΑΚΗ ΣΧΕΔΙΑΣΗ W. Dally, C. Harting Πανεπιστημιακές Εκδόσεις Κρήτης
14. ΨΗΦΙΑΚΗ ΣΧΕΔΙΑΣΗ Κ.Ευσταθίου, Εκδόσεις νέων Τεχνολογιών
15. ΨΗΦΙΑΚΑ ΗΛΕΚΤΡΟΝΙΚΑ ΘΕΩΡΙΑ ΚΑΙ ΕΦΑΡΜΟΓΕΣ, LEACH-MALVINO Εκδόσεις Τζιόλα
16. ΨΗΦΙΑΚΑ ΗΛΕΚΤΡΟΝΙΚΑ R.TOKEHEIM Εκδόσεις Τζιόλα
17. ΨΗΦΙΑΚΗ ΣΧΕΔΙΑΣΗ J. WAKERLY Εκδόσεις Κλειδάριθμος
18. ΨΗΦΙΑΚΗ ΣΧΕΔΙΑΣΗ,Ρουμελιώτης Σουραβλάς Εκδόσεις Τζιόλα

NAND



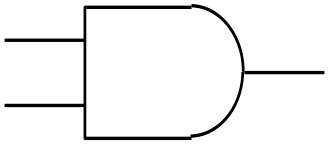
74LS00

OR



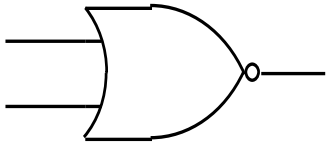
74LS32

AND



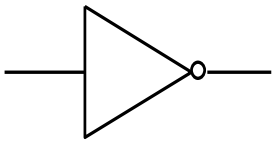
74LS08

NOR



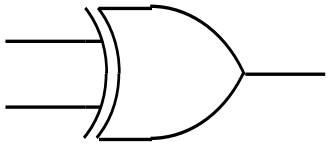
74LS02

NOT



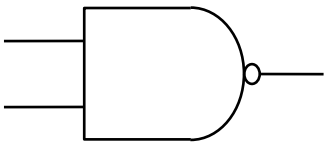
74LS04

XOR



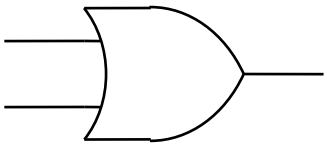
74LS86

NAND



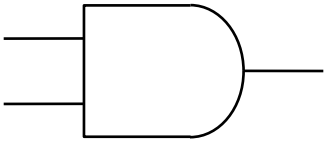
74LS00

OR



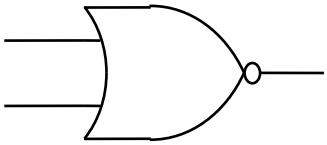
74LS32

AND



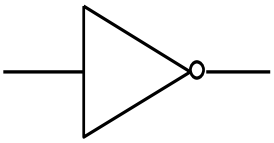
74LS08

NOR



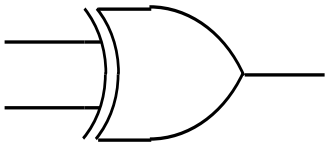
74LS02

NOT



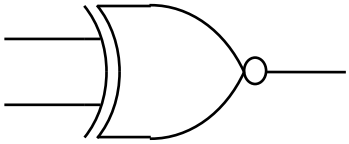
74LS04

XOR

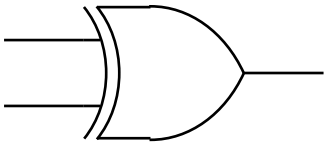


74LS86

XNOR



XOR



74LS86