# 2. ΚΩΔΙΚΕΣ

## 2.1 Εισαγωγή

Κώδικας είναι ένας συστηματικός τρόπος παράστασης πληροφοριών με μεγάλη εφαρμογή στα συστήματα των υπολογιστών και την μεταβίβαση πληροφοριών. Ο υπολογιστής χρησιμοποιεί το δυαδικό σύστημα με στοιχεία τα μηδέν-0 και ένα-1, ενώ ο άνθρωπος χρησιμοποιεί το δεκαδικό σύστημα και τα γράμματα του αλφάβητου. Αυτά γίνονται κατανοητά από τον υπολογιστή αφού κωδικοποιηθούν σε δυαδικούς αριθμούς.

Η κωδικοποίηση γίνεται με διάφορους κώδικες ανάλογα με την εφαρμογή. Οι πιο γνωστοί κώδικες είναι ο **BCD**-Binary Coded Decimal, ο EBCDIC-Extended Binary Coded Decimal Interchange Code, ο **ASCII**-American Standard Code for Information Interchange, ο **UNICODE** Standard που είναι τμήμα του ISO (93).

Μια άλλη χρήση των κωδίκων είναι ο έλεγχος και η διόρθωση σφαλμάτων σε συστήματα ψηφιακής μεταβίβασης (Digital Transmision) ή σε πληροφορίες που μετακινούνται ανάμεσα στα διάφορα τμήματα του υπολογιστή (μεταξύ **Κ.Μ.Ε** και τερματικών μονάδων).

Ο έλεγχος ή και η διόρθωση σφαλμάτων γίνεται με χρήση ψηφίων πλεονασμού στην λέξη του υπολογιστή. Οι πιο γνωστοί και απλοί κώδικες ελέγχου σφάλματος είναι οι κώδικες **άρτιας** & **περιττής** ισοτιμίας. Ακόμη οι κώδικες χρησιμοποιούνται στην μετατροπή του δεκαδικού στο δυαδικό σύστημα και αντίθετα, π.χ. οι αριθμοί εισάγονται από το πληκτρολόγιο στο δεκαδικό σύστημα. Ο υπολογιστής έχει τα ειδικά κυκλώματα κωδικοποίησης του δεκαδικού σε δυαδικό (BCD). Επίσης υπάρχουν και κώδικες που χρησιμοποιούνται για αριθμητικές πράξεις, έχουν φτιαχτεί ειδικά, ώστε να απλουστεύουν τα πράγματα στον τομέα αυτό. Π.χ. ο BCD-2421, ο κώδικας με υπερβολή κατά τρία BCD-X3.... κ.λ.π.

## 2.2 Κώδικες (BCD-Binary Coded Decimal) Κωδικοποίησης του δεκαδικού σε δυαδικό

Η κωδικοποίηση των δεκαδικών ψηφίων 0**..**9 στο δυαδικό σύστημα απαιτεί τουλάχιστον τέσσερα (4) δυαδικά ψηφία για να ισχύει η σχέση $10 \leq 2^n$ η οποία αληθεύει για **η**$\geq 4$**.** Ανάλογα με την αντιστοιχία που θα χρησιμοποιήσουμε για την κωδικοποίηση προκύπτει και ένας BCD κώδικας. Από τους $2^4=16$ διαφορετικούς συνδυασμούς από **0** & **1**, μόνο οι δέκα πρώτοι χρησιμοποιούνται για την κωδικοποίηση ενώ οι υπόλοιποι 6 είναι κενοί.

### 2.2.1 Κώδικας κωδικοποίησης BCD-8421

Είναι ένας απλός κώδικας BCD ο οποίος χρησιμοποιεί τους 10 πρώτους δυαδικούς αριθμούς για την κωδικοποίηση των δεκαδικών ψηφίων (λέγεται και NBCD-Natural BCD). Ο BCD-8421 είναι κώδικας με "βάρη" (Weights).

Τα βάρη είναι δυνάμεις του δύο-2 και καθορίζονται ανάλογα με την θέση του δυαδικού ψηφίου στον κώδικα (τα βάρη γράφονται στο πάνω μέρος της στήλης). Τα βάρη εδώ είναι **8, 4, 2, 1** και έχουν την ιδιότητα ότι αν τα αθροίσουμε, για τα αντίστοιχα ψηφία που είναι ένα-1, τότε προκύπτει ο αντίστοιχος δεκαδικός αριθμός σύμφωνα με την 1.1 σχέση. (Ο κώδικας φαίνεται στο πίνακα που ακολουθεί σε επόμενη σελίδα).

Π.χ. 8 4 2 1
1 0 0 1 $\Rightarrow$ 8.1+1.1=8+1=$9_{(10)}$

Για την κωδικοποίηση αριθμού με πολλά ψηφία αντικαθιστούμε κάθε ψηφίο, του αριθμού, με το αντίστοιχο ψηφίο στον BCD-8421 κώδικα.

Π.χ. $N_{(10)}$=412 $\Rightarrow$ $N_{(2)}$=0100 0001 001 .

Με αυτό το σύστημα κωδικοποίησης αναγνωρίζουμε εύκολα τους δεκαδικούς αριθμούς, όμως είναι δύσκολος ο υπολογισμός του συμπληρώματος ως προς 9*, οπότε είναι δύσκολη και η πράξη της αφαίρεσης. Υπάρχουν άλλοι κώδικες οι οποίοι χρησιμοποιούνται περισσότερο. Για την κατασκευή ενός οποιουδήποτε κώδικα πρέπει να ισχύει ότι το "άθροισμα των βαρών" του να **μην** είναι μεγαλύτερο του 15 και μικρότερο του 9, δηλαδή $\mathbf{9 \leq \Sigma \leq 15}$.

**ΣΗΜΕΙΩΣΗ: Το συμπλήρωμα ως προς εννέα ενός αριθμού βρίσκεται αν αφαιρέσουμε όλα τα ψηφία του, κάθε ένα ξεχωριστά, από το 9. Π.χ. Το συμπλήρωμα του 642 ως προς 9 είναι το 357**

### 2.2.2 Κώδικας κωδικοποίησης BCD-2421

Είναι κώδικας με βάρη 2-4-2-1 (ή άθροισμα 9) και έχει την ιδιότητα να δίδει το συμπλήρωμα ενός δεκαδικού ψηφίου με αντιστροφή των αντίστοιχων δυαδικών ψηφίων του κώδικα. Για τον λόγο αυτό ο κώδικας λέγονται και "αυτοσυμπληρωμένος" (Self Complementing Code). Με την ιδιότητά του αυτή διευκολύνει αρκετά την αφαίρεση των δεκαδικών αριθμών στο σύστημα συμπληρώματος ως προς εννέα-9.

Ο κώδικας BCD-2421 χρησιμοποιείται στις ηλεκτρονικές αριθμομηχανές (τα γνωστά κομπιουτεράκια) Electronic Calculators.

### 2.2.3 Κώδικας με βάρη BCD-5211

Είναι κώδικας με βάρη **5-2-1-1** (ή άθροισμα 9), όπως ο BCD-2421 και έχει ίδιες τις ιδιότητες με αυτόν, δηλαδή δίδει το συμπλήρωμα ενός δεκαδικού ψηφίου με αντιστροφή των αντίστοιχων δυαδικών ψηφίων του κώδικα.

Και οι δύο κώδικες έχουν την ιδιότητα να είναι συμμετρικοί ως προς το κέντρο, αυτό φαίνεται στον πίνακα, οπότε βρίσκεται εύκολα το συμπλήρωμα ως προς 9, με την αντιστροφή του ισοδύναμου δυαδικού αριθμού.

Π.χ. Στον κώδικα BCD-5211 το **4**=0111 με συμπλήρωμα προς 9, **είναι** το 9-4= 5=1000 (αντιστροφή του 4=0111).

Οι παραπάνω κώδικες, αλλά και όσοι άλλοι, έχουν αυτή την ιδιότητα, να είναι αυτοσυμληρούμενοι, είναι κατάλληλοι για αριθμητικές πράξεις.

Αυτοσυμπληρούμενο κώδικα έχουμε αν σε κάθε ψηφίο του BCD-8421 προσθέσουμε το 3. Ο κώδικας που προκύπτει δεν έχει βάρη και καλείται "Κώδικας Υπερβολής κατά τρία "(BCD- eXcess Three Code).

### 2.2.4 Κώδικας Υπερβολής κατά τρία (BCD-eXcess Three Code-χωρίς βάρη)

Για την κωδικοποίηση ενός δεκαδικού ψηφίου, προσθέτουμε 3 στο ψηφίο αυτό και θεωρούμε τον BCD-8421, σαν κώδικα του αθροίσματος π.χ. 0+3=3 κωδικοποίηση του 3, (ο κώδικας φαίνεται στον προηγούμενο πίνακα). Έχει την ιδιότητα του BCD-2421 για το συμπλήρωμα προς 9, δηλαδή την αντιστροφή όλων των δυαδικών ψηφίων του κώδικα.

Παρατηρούμε ότι για την κωδικοποίηση του μηδενός (0) παίρνουμε τον BCD-8421 κώδικα του 3, το **0 0 1 1** και όλα τα άλλα ψηφία κωδικοποιούνται με την συνέχιση της αρίθμησης στο δυαδικό σύστημα από 0 0 1 1 και πάνω.

**<u>Εφαρμογή</u>.** Αφαίρεση στο σύστημα συμπληρώματος ως προς 9 με τη χρήση δεκαδικών κωδικοποιημένων στον κώδικα BCD-**X3**. Και εδώ ισχύει ακριβώς ότι και στο σύστημα συμπληρώματος ως προς ένα, δηλαδή αντί για αφαίρεση γίνεται πρόσθεση και το κρατούμενο της πρόσθεσης, αν υπάρχει, προστίθεται στο άθροισμα.

| | | | |
|---|---|---|---|
| Π.χ αντί | 8 6 4 | κάνουμε | 8 6 4 |
| | - 2 7 8 | | + 7 2 1 |
| | 5 8 6 | | 1 5 8 5 |
| | | | ↓→+ 1 |
| | | | 5 8 6 |

Με τον κώδικα BCD-X3 βρίσκουμε εύκολα το συμπλήρωμα ως προς 9 του αφαιρέτη αν αντιστρέψουμε όλα τα ψηφία του κώδικα

Π.χ 1 7 5= 0 1 0 0 1 0 1 0 1 0 0 0
8 2 4=1 0 1 1 0 1 0 1 0 1 1 1

Πίνακας μερικών βασικών κωδίκων

| ΔΕΚΑΔΙΚΟ ΨΗΦΙΟ | ΚΩΔΙΚΑΣ BCD-8421 8 4 2 1 | ΚΩΔΙΚΑΣ BCD-2421 2 4 2 1 | ΚΩΔΙΚΑΣ BCD-5211 5 2 1 1 | ΚΩΔΙΚΑΣ BCD-7421 7 4 2 1 | ΚΩΔΙΚΑΣ GRAY | ΚΩΔΙΚΑΣ BCD-eXcess 3 Υπερβολής 3 |
|---|---|---|---|---|---|---|
| 0 | 0 0 0 0 | 0 0 0 0 | 0 0 0 0 | 0 0 0 0 | 0 0 0 0 | 0 0 1 1 |
| 1 | 0 0 0 1 | 0 0 0 1 | 0 0 0 1 | 0 0 0 1 | 0 0 0 1 | 0 1 0 0 |
| 2 | 0 0 1 0 | 0 0 1 0 | 0 1 0 0 | 0 0 1 0 | 0 0 1 1 | 0 1 0 1 |
| 3 | 0 0 1 1 | 0 0 1 1 | 0 1 0 1 | 0 0 1 1 | 0 0 1 0 | 0 1 1 0 |
| 4 | 0 1 0 0 | 0 1 0 0 | 0 1 1 1 | 0 1 0 0 | 0 1 1 0 | 0 1 1 1 |
| 5 | 0 1 0 1 | 1 0 1 1 | 1 0 0 0 | 0 1 0 1 | 0 1 1 1 | 1 0 0 0 |
| 6 | 0 1 1 0 | 1 1 0 0 | 1 0 1 0 | 0 1 1 0 | 0 1 0 1 | 1 0 0 1 |
| 7 | 0 1 1 1 | 1 1 0 1 | 1 0 1 1 | 1 0 0 0 | 0 1 0 0 | 1 0 1 0 |
| 8 | 1 0 0 0 | 1 1 1 0 | 1 1 1 0 | 1 0 0 1 | 1 1 0 0 | 1 0 1 1 |
| 9 | 1 0 0 1 | 1 1 1 1 | 1 1 1 1 | 1 0 1 0 | 1 1 0 1 | 1 1 0 0 |
| 10 | | | | | 1 1 1 1 | |
| 11 | | | | | 1 1 1 0 | |
| 12 | | | | | 1 0 1 0 | |
| 13 | | | | | 1 0 1 1 | |
| 14 | | | | | 1 0 0 1 | |
| 15 | | | | | 1 0 0 0 | |

**<u>ΣΗΜ:</u> Ο κώδικας των δεκαδικών ψηφίων 5-9 βρίσκεται αντιστρέφοντας τα δυαδικά ψηφία των συμπληρωμάτων τους. Π.χ Ο αριθμός 4=0100 έχει συμπλήρωμα, προς 9, το 9-4=5=1011, το οποίο βρίσκεται αμέσως με την αντιστροφή των ψηφίων του 4=0100**

**5=1011**

## 2.3. Κώδικας G R A Y (κωδικοποίησης του δεκαδικού σε δυαδικό)

Είναι κώδικας χωρίς βάρη και ανήκει στους κώδικες <u>*μοναδιαίας απόστασης*</u> (Unit Distance Code). Έχει την πολύ βασική ιδιότητα ότι μόνο ένα ψηφίο (bit) αλλάζει μεταξύ δύο διαδοχικών αριθμών (οι κώδικες δύο γειτονικών δεκαδικών ψηφίων διαφέρουν σε ένα δυαδικό ψηφίο μόνο). Χρησιμοποιείται σε κώδικες πληροφοριών.

Ο **Gray** λέγεται και <u>***ανακλώμενος***</u> δυαδικός κώδικας (Reflected Binary Code), γιατί τα τρία (3) δεξιότερα ψηφία του κώδικα, είναι συμμετρικά πάνω και κάτω από την διαχωριστική γραμμή. Η πιθανότητα λάθους είναι μικρή αφού σε κάθε μεταβολή αλλάζει ένα μόνο ψηφίο, σε αντίθεση με τον BCD-8421 όπου π.χ. από το 7 στο 8 αλλάζουν 4 ψηφία (4 -bits) $\Rightarrow$0111$\Rightarrow$1000 .

Η μετατροπή του δυαδικού σε Gray γίνεται με την πρόσθεση ανά δύο των ψηφίων και αγνόηση του κρατούμενου π.χ 1 1 0 0

$\downarrow\ \downarrow\ \downarrow\ \downarrow$

$\Rightarrow$ 1 0 1 0 ενώ το πρώτο ψηφίο μένει όπως είναι,

ή μένει ίδιο το Π.Σ.Ψ. του δυαδικού και τα επόμενα προκύπτουν από την σχέση $G_i = B_{i+1} \oplus B_i$ δηλαδή $G_3 = B_3$ και $G_2 = B_3 \oplus B_2$, $G_1 = B_2 \oplus B_1$, $G_0 = B_1 \oplus B_0$

Η αντίστροφη μετατροπή Gray σε δυαδικό γίνεται με την πρόσθεση ανά δύο των ψηφίων διαγώνια με αγνόηση του κρατούμενου, ενώ το πρώτο ψηφίο μένει όπως είναι Π.χ. 1 1 0 1 0 0 1 1 0

1 0 0 1 1 1 0 1 1

ή μένει το ίδιο το Π.Σ.Ψ. του αριθμού GRAY και τα υπόλοιπα προκύπτουν από την σχέση $B_i = B_{i+1} + G_i$, δηλαδή $B_3 = G_3$ και $B_2 = B_3 + G_2$, $B_1 = B_2 + G_1$, $B_0 = B_1 + G_0$

**2.4. Εφαρμογές.** Στην κωδικοποίηση η μετατροπή ενός δυαδικού αριθμού σε κώδικα BCD γίνεται με την μετατροπή κάθε ψηφίου του στον αντίστοιχο κώδικα.

Π.χ Το $369_{(10)}$ να μετατραπεί στον BCD-8421. Γράφουμε 3 6 9⇒0011 0110 1001.

Π.χ Το $369_{(10)}$ να μετατραπεί στον BCD-X3. Γράφουμε 3 6 9⇒0110 1001 1100.

Π.χ Το $369_{(10)}$ να μετατραπεί στον BCD-2421. Γράφουμε 3 6 9⇒ 0011 1100 1111.

Από τα παραπάνω φαίνεται ότι η παράσταση αριθμών σε BCD κώδικες απαιτεί 12 ψηφία ενώ χρειάζεται μόνο 9 ψηφία για την παράσταση του αριθμού στο δυαδικό σύστημα. Επίσης απαιτεί περισσότερα κυκλώματα.

## 2.5 Κώδικες Ανίχνευσης και διόρθωσης λαθών (Error Correcting Codes)

Για να βρούμε τυχόν λάθη, υπό την μορφή θορύβου, που μπαίνουν κατά την μεταφορά των πληροφοριών είτε εντός του ψηφιακού συστήματος είτε από μονάδα σε μονάδα είτε από ένα Ψ.Σ. σε άλλο, έχουμε διάφορους κώδικες.

Οι κώδικες BCD με περισσότερα από τέσσερα δυαδικά ψηφία λέγονται *κώδικες με πλεονασμό*. Τα πρόσθετα ψηφία χρησιμοποιούνται για την ανίχνευση ή και για τη διόρθωση σφαλμάτων. Υπάρχουν κώδικες με πλεονασμό ενός ψηφίου (1 bit) για την ανίχνευση ενός σφάλματος, κώδικες με πλεονασμό δύο ψηφίων (2 bits) για την ανίχνευση δύο σφαλμάτων ή την διόρθωση ενός σφάλματος κ.λ.π.

### 2.5.1 Κώδικας BCD απλής ισοτιμίας

Εδώ χρησιμοποιείται ένα επιπλέον ψηφίο (4+1), το οποίο λέγεται ψηφίο ισοτιμίας (Parity bit), το οποίο καθορίζεται ανάλογα με το αν η ισοτιμία είναι *άρτια* ή *περιττή*.

**Άρτια** ισοτιμία (Even Parity) όπου το 5ο ψηφίο έχει τιμή τέτοια όπου το σύνολο των άσσων (1) σε κάθε ψηφίο να είναι άρτιος αριθμός ή Σ('1')=άρτιος.

**Περιττή** ισοτιμία (Odd Parity) όπου το 5ο ψηφίο έχει τιμή τέτοια όπου το σύνολο των άσσων (1) σε κάθε ψηφίο να είναι περιττός αριθμός ή Σ('1')=περιττός.

Με το ένα ψηφίο ισοτιμίας μπορούμε να ανιχνεύσουμε την ύπαρξη περιττού πλήθους σφαλμάτων, χωρίς όμως να μπορούμε να καθορίσουμε τη θέση του ψηφίου όπου έγινε το λάθος.

Κώδικες με ψηφίο ισοτιμίας χρησιμοποιούνται στους αναγνώστες καρτών (Card Readers), χαρτοταινιών (Paper tape readers) και στην αποθήκευση πληροφορίας σε μαγνητικές ταινίες. Οι δύο κώδικες απλής ισοτιμίας φαίνονται στον πίνακα.

| ΔΕΚΑΔΙΚΟ ΨΗΦΙΟ | ΚΩΔΙΚΑΣ BCD-8421 Με άρτια ισοτιμία (Even Parity) | | | | | ΚΩΔΙΚΑΣ BCD-8421 Με περιττή ισοτιμία (Odd Parity) | | | | | ΚΩΔΙΚΑΣ Δύο από πέντε (2 out of 5) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 8 | 4 | 2 | 1 | P.B | 8 | 4 | 2 | 1 | P.B | | | | | |
| 0 | 0 | 0 | 0 | 0 | **0** | 0 | 0 | 0 | 0 | **1** | 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 | **1** | 0 | 0 | 0 | 1 | **0** | 0 | 0 | 1 | 1 | 0 |
| 2 | 0 | 0 | 1 | 0 | **1** | 0 | 0 | 1 | 0 | **0** | 0 | 1 | 1 | 0 | 0 |
| 3 | 0 | 0 | 1 | 1 | **0** | 0 | 0 | 1 | 1 | **1** | 1 | 1 | 0 | 0 | 0 |
| 4 | 0 | 1 | 0 | 0 | **1** | 0 | 1 | 0 | 0 | **0** | 0 | 0 | 1 | 0 | 1 |
| 5 | 0 | 1 | 0 | 1 | **0** | 0 | 1 | 0 | 1 | **1** | 0 | 1 | 0 | 1 | 0 |
| 6 | 0 | 1 | 1 | 0 | **0** | 0 | 1 | 1 | 0 | **1** | 1 | 0 | 1 | 0 | 0 |
| 7 | 0 | 1 | 1 | 1 | **1** | 0 | 1 | 1 | 1 | **0** | 0 | 1 | 0 | 0 | 1 |
| 8 | 1 | 0 | 0 | 0 | **1** | 1 | 0 | 0 | 0 | **0** | 1 | 0 | 0 | 1 | 0 |
| 9 | 1 | 0 | 0 | 1 | **0** | 1 | 0 | 0 | 1 | **1** | 1 | 0 | 0 | 0 | 1 |

### 2.5.2 Κώδικας N από M (N out of M) ή (N-μονάδες από M-συνολικά ψηφία)

Εδώ κάθε δεκαδικό ψηφίο κωδικοποιείται με **M**-δυαδικά ψηφία, από τα οποία τα **N**-ψηφία είναι άσσοι-**1** και τα άλλα **M** είναι μηδενικά-**0**. Το σύνολο των συνδυασμών που σχηματίζονται με την μέθοδο αυτή βρίσκεται από τη σχέση:

**Π=(M!)/(N!*(M-N))!** Όπου: **!** = παραγοντικό.

Ο κώδικας 2 από 5 φαίνεται στον παραπάνω πίνακα. Με αυτόν ανιχνεύουμε την ύπαρξη λαθών κατά την μεταβίβαση της πληροφορίας, σε ένα σύστημα επικοινωνίας, αν στην λήψη το πλήθος των μονάδων **N** **<u>δεν είναι</u>** π.χ. δύο. Το πλήθος των συνδυασμών είναι Π=5!/(2!*(5!-2!))=5*4*3!/(2*1*(3!))= 20/2=10.

## 2.6 Κώδικες διόρθωσης λαθών.

Στους κώδικες διόρθωσης λαθών μπαίνουν δύο, τουλάχιστον, ψηφία πλεονασμού. Αύξηση του πλήθους των ψηφίων πλεονασμού, σημαίνει μεγαλύτερη δυνατότητα διόρθωσης περισσότερων λαθών. Αυτό ταυτόχρονα σημαίνει "**λιγότερη**" πληροφορία στην λέξη. Ας δούμε δυο τέτοιους κώδικες, τους πλέον απλούς.

### 2.6.1 Ορθογώνιοι πίνακες (Rectangular Codes)

Είναι μια κατηγορία απλών κωδίκων διόρθωσης λαθών στους οποίους η πληροφορία ταξινομείται σε ένα **ορθογώνιο** πίνακα (ARRAY) **mxn** θέσεων, και στη συνέχεια μπαίνει ψηφίο ισοτιμίας <u>τόσο</u> στις γραμμές <u>όσο</u> και στις στήλες του πίνακα, οπότε ο πίνακας γίνεται (**m+1)x(n+1)** διαστάσεων. Με τον τρόπο αυτό χρησιμοποιούμε ψηφίο ισοτιμίας για κάθε λέξη πληροφορίας και μετά από κάθε **m** λέξεις πληροφορίας εισάγουμε και <u>μία λέξη ελέγχου</u>. Στους πίνακες βλέπουμε ακριβώς τον τρόπο σε ένα ορθογώνιο πίνακα περιττής ισοτιμίας και πώς εντοπίζεται το λάθος. Στο παράδειγμά μας αν εξετάσουμε στην λήψη την περιττή ισοτιμία, για όλες τις γραμμές και τις στήλες του πίνακα, βρίσκουμε το λάθος στην $4^{η}$ γραμμή και $3^{η}$ στήλη.

| | ΕΚΠΟΜΠΗ | | | | | | P.B |
|---|---|---|---|---|---|---|---|
| Π.χ | 1 | 0 | 0 | 1 | 1 | 1 | **1** |
| | 0 | 1 | 0 | 0 | 1 | 0 | **1** |
| | 0 | 1 | 1 | 1 | 1 | 0 | **1** |
| | 1 | 0 | 1 | 0 | 1 | 1 | **1** |
| | 0 | 0 | 1 | 1 | 0 | 1 | **0** |
| P.B | **1** | **1** | **0** | **0** | **1** | **0** | |
| | | | | | | | |

| | ΛΗΨΗ | | | | | | P.B | |
|---|---|---|---|---|---|---|---|---|
| | 1 | 0 | 0 | 1 | 1 | 1 | **1** | |
| | 0 | 1 | 0 | 0 | 1 | 0 | **1** | |
| | 0 | 1 | 1 | 1 | 1 | 0 | **1** | |
| | 1 | 0 | 0 | 0 | 1 | 1 | **1** | **0** |
| | 0 | 0 | 1 | 1 | 0 | 1 | **0** | |
| P.B | **1** | **1** | **0** | **0** | **1** | **0** | | |
| | | | 1 | | | | | |

Το ψηφίο της $4^{ης}$ γραμμής και $3^{ης}$ στήλης είναι λάθος γιατί το PB οριζόντια έπρεπε να είναι ‘**0’** και είναι ‘**1’ και** κατακόρυφα έρεπε να είναι ‘**1**’ και είναι ‘**0’**.

### 2.6.2 Κώδικες Hamming

Οι κώδικες ανιχνεύουν και εντοπίζουν την θέση ενός λάθους κατά την μετάδοση μιας δυαδικής πληροφορίας μήκους **m** ψηφίων, με την χρήση **k** ψηφίων ισοτιμίας κατάλληλα τοποθετημένων μέσα στην πληροφορία.

Για τον αριθμό **k** ισχύει η σχέση $2^k \geq m+k+1$. Από αυτή προκύπτει ο πίνακας ο οποίος δείχνει το πλήθος των ψηφίων **k** (το μικρότερο δυνατό) για τη λέξη μήκους **m** ψηφίων. Η νέα λέξη θα έχει **m**+**k=η** θέσεις (από το ένα έως το **m**+**k**), αρχίζοντας από το ΛΣΨ. Η λέξη συμβολίζεται (**n,m**) π.χ. η λέξη (7,4) .

| m | k min |
|---|---|
| 1 | 2 |
| 2 - 4 | 3 |
| 5 - 11 | 4 |
| 12 - 26 | 5 |
| 27 - 57 | 6 |

Αν το μήκος της λέξης **n** είναι γνωστό τότε το πλήθος των ψηφίων πληροφορίας δίδεται από την σχέση $2^m \leq 2^n/(n+1)$.
Π.χ. για **n**=4 από τη προηγούμενη σχέση έχουμε $m_{max}=1$ και για **n**=7 έχουμε $m_{max}=4$.

Ας δούμε αναλυτικά τον κώδικα για τη λέξη (7,4).

Στην λέξη θα περιέχονται 3 ψηφία ελέγχου (αφού **m**+**k=n** και $m_{max}=4$). Τα ψηφία αυτά θα ελέγχουν με <u>άρτια</u> ισοτιμία ορισμένα ψηφία πληροφορίας.

Για να βρεθούν οι **θέσεις** των ψηφίων ελέγχου, καθώς και ποια ψηφία πληροφορίας ελέγχονται από το κάθε ψηφίο ελέγχου, γράφουμε τον πίνακα του οποίου οι στήλες προκύπτουν από τη κωδικοποίηση των αριθμών 1÷**n** (ή 1÷7) στο δυαδικό σύστημα. Οι στήλες του πίνακα που περιέχουν **μία μόνο** μονάδα, δίδουν τις θέσεις των ψηφίων ελέγχου. Δηλαδή τα ψηφία ελέγχου εδώ θα είναι το **1°** το **2°** και το **4°** ψηφίο.

| ΕΛ | ΕΛ | ΠΛ | ΕΛ | ΠΛ | ΠΛ | ΠΛ | ΕΛ = Έλεγχος ΠΛ = Πληροφορία |
|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
| 0 | 0 | 0 | **1** | 1 | 1 | 1 | 3° ψηφίο ελέγχου |
| 0 | **1** | 1 | 0 | 0 | 1 | 1 | 2° ψηφίο ελέγχου |
| **1** | 0 | 1 | 0 | 1 | 0 | 1 | 1° ψηφίο ελέγχου |

Επίσης η κάθε γραμμή του πίνακα δίδει τα ψηφία της λέξης τα οποία ελέγχονται, από το αντίστοιχο ψηφίο ελέγχου. Η αντιστοιχία σημειώνεται στα δεξιά του πίνακα.

Δηλαδή : Το 1ο ψηφίο ελέγχου ελέγχει τις θέσεις 1, 3, 5, 7
Το 2ο ψηφίο ελέγχου ελέγχει τις θέσεις 2, 3, 6, 7
Το 3ο ψηφίο ελέγχου ελέγχει τις θέσεις 4, 5, 6, 7

Για έλεγχο με άρτια ισοτιμία, μπορούμε να καθορίσουμε τα ψηφία ελέγχου όταν δοθούν τα ψηφία της πληροφορίας.

Π.χ. Έστω η πληροφορία 1101. Σύμφωνα με τον πίνακα κωδικοποίησης των θέσεων η λέξη των 7 ψηφίων θα είναι: 1 2 3 4 5 6 7

ΕΛ1 ΕΛ2 **1** ΕΛ3 **1 0 1**

Όπου: Το ψηφίο ΕΛ1 ελέγχει με άρτια ισοτιμία τις θέσεις 1, 3, 5, 7 οπότε ΕΛ1=1

Το ψηφίο ΕΛ2 ελέγχει με άρτια ισοτιμία τις θέσεις 2, 3, 6, 7 οπότε ΕΛ2=0

Το ψηφίο ΕΛ3 ελέγχει με άρτια ισοτιμία τις θέσεις 4, 5, 6, 7 οπότε ΕΛ3=0

Τελικά η λέξη η οποία θα μεταβιβαστεί είναι η 1 0 **1** 0 **1 0 1**.

Έστω τώρα ότι πήραμε την λέξη 1 0 **1** 0 **0 0 1**, με λάθος στην μεταβίβαση σε ένα ψηφίο της πληροφορίας. 1η 2η 3η 4η 5η 6η 7η

. Από τις θέσεις οι οποίες ελέγχονται από κάθε ψηφίο ελέγχου, μπορούμε να καθορίσουμε, με άρτια ισοτιμία, τρία ψηφία ελέγχου.

Δηλαδή: Στις θέσεις 1, 3, 5, 7 το ψηφίο ισοτιμίας είναι ένα (**1**)

Στις θέσεις 2, 3, 6, 7 το ψηφίο ισοτιμίας είναι μηδέν (**0**)

Στις θέσεις 4, 5, 6, 7 το ψηφίο ισοτιμίας είναι ένα (**1**) ←ΠΣΨ

Θεωρούμε τα τρία ψηφία άρτιας ισοτιμίας, τα οποία βρήκαμε παραπάνω, με την <u>αντίστροφη σειρά</u>, σχηματίζεται μια λέξη **k** ψηφίων η οποία λέγεται "σύνδρομο". Ο δεκαδικός ο οποίος <u>αντιστοιχεί στο σύνδρομο</u> μας δείχνει <u>και την θέση του λάθους</u> στην λέξη.

Στην περίπτωσή μας το σύνδρομο είναι $101=5_{(10)}$, επομένως το ψηφίο με το λάθος είναι το 5ο και η σωστή λέξη θα είναι η 10**10101**, όπου το λάθος διορθώθηκε με απλή αντιστροφή του ψηφίου της 5ης θέσης.

## 2.7. Αλφαριθμητικοί κώδικες (Alphanumeric codes)

Στις Υψηλού επιπέδου (High Language) γλώσσες προγραμματισμού χρησιμοποιούνται, στην σύνταξη των εντολών, αλφαριθμητικοί χαρακτήρες. Για την μετατροπή τους σε ομάδες δυαδικών ψηφίων, κατανοητών από τον υπολογιστή, χρησιμοποιούνται οι λεγόμενοι αλφαριθμητικοί κώδικες. Οι γνωστότεροί τους είναι ο κώδικας ASCII και ο εκτεταμένος κώδικας EBCDIC.

### 2.7.1. Κώδικας ASCII

Ο ASCII (American Standard Code for Information Interchange) είναι ο πλέον γνωστός αλφαριθμητικός κώδικας με μεγάλη χρήση στους μίνι και μίκρο υπολογιστές. Οι βασικοί αλφαριθμητικοί χαρακτήρες είναι 84 (10 αριθμοί 0,1,2,3,4,5,6,7,8,9 τα 26 κεφαλαία γράμματα A,B, **..**,Z και 26 μικρά a,b, **..**,z και περίπου 22 ειδικοί χαρακτήρες). Οι 84 αυτοί χαρακτήρες μπορούν να κωδικοποιηθούν με 7 ψηφία, αφού ισχύει $2^6<84<2^7$ και έχουμε πλεόνασμα 44 επιπλέον θέσεις για κωδικοποίηση άλλων χαρακτήρων.

Ο ASCII χρησιμοποιεί 7 ψηφία για την κωδικοποίηση και μερικές φορές το 8ο ψηφίο σαν ψηφίο ισοτιμίας στην ανίχνευση σφαλμάτων. Στον κώδικα περιλαμβάνονται και ειδικοί χαρακτήρες ή λειτουργίες οι οποίες χρησιμοποιούνται στις επικοινωνίες. Σε μικρούς υπολογιστές δεν χρησιμοποιείται ισοτιμία και το 8ο ψηφίο προστίθεται στην δυνατότητα κωδικοποίησης αλφαριθμητικών χαρακτήρων, $2^8$=256(0-255), πολλοί από τους οποίους είναι γραφικοί χαρακτήρες, το Ελληνικό αλφάβητο κ.λ.π .

| **Character** | ΚΩΔΙΚΑΣ ASCII | ΚΩΔΙΚΑΣ EBCDIC | **Character** | ΚΩΔΙΚΑΣ ASCII | ΚΩΔΙΚΑΣ EBCDIC |
|---|---|---|---|---|---|
| **Space** | 010 0000 | 0100 0000 | **A** | 100 0001 | 1100 0001 |
| **!** | 010 0001 | 0101 1010 | **B** | 100 0010 | 1100 0010 |
| **"** | 010 0010 | 0111 1111 | **C** | 100 0011 | 1100 0011 |
| **#** | 010 0011 | 0111 1011 | **D** | 100 0100 | 1100 0100 |
| **$** | 010 0100 | 0101 1011 | **E** | 100 0101 | 1100 0101 |
| **%** | 010 0101 | 0110 1100 | **F** | 100 0110 | 1100 0110 |
| **&** | 010 0110 | 0101 0000 | **G** | 100 1111 | 1100 0111 |
| **'** | 010 0111 | 0111 1101 | **H** | 100 1000 | 1100 1000 |
| **(** | 010 1000 | 0100 1101 | **I** | 100 1001 | 1100 1001 |
| **)** | 010 1001 | 0101 1101 | **J** | 100 1010 | 1101 0001 |
| ***** | 010 1010 | 0101 1100 | **K** | 100 1011 | 1101 0010 |
| **+** | 010 1011 | 0100 1110 | **L** | 100 1100 | 1101 0011 |
| **,** | 010 1100 | 0110 1011 | **M** | 100 1101 | 1101 0100 |
| **-** | 010 1101 | 0100 0000 | **N** | 100 1110 | 1101 0101 |
| **.** | 010 1110 | 0110 1011 | **O** | 100 1111 | 1101 0110 |
| **/** | 010 1111 | 0110 0001 | **P** | 101 0000 | 1101 0111 |
| **0** | 011 0000 | 1111 0000 | **Q** | 101 0001 | 1101 1000 |
| **1** | 011 0001 | 1111 0001 | **R** | 101 0010 | 1101 1001 |
| **2** | 011 0010 | 1111 0010 | **S** | 101 0011 | 1110 0010 |
| **3** | 011 0011 | 1111 0011 | **T** | 101 0100 | 1110 0011 |
| **4** | 011 0100 | 1111 0100 | **U** | 101 0101 | 1110 0100 |
| **5** | 011 0101 | 1111 0101 | **V** | 101 0110 | 1110 0101 |
| **6** | 011 0110 | 1111 0110 | **W** | 101 0111 | 1110 0110 |
| **7** | 011 0111 | 1111 0111 | **X** | 101 1000 | 1110 0111 |
| **8** | 011 1000 | 1111 1000 | **Y** | 101 1001 | 1110 1000 |
| **9** | 011 1001 | 1111 1001 | **Z** | 101 1010 | 1110 1001 |

### 2.7.2. Κώδικας EBCDIC

Ο EBCDIC (Extended Binary Coded Decimal Interchange Code) κώδικας χρησιμοποιείται στους IBM υπολογιστές. Χρησιμοποιεί 8-ψηφία άρα κωδικοποιεί τους διπλάσιους χαρακτήρες από την πρώτη μορφή του ASCII. Στον παραπάνω πίνακα φαίνονται οι δύο κώδικες.

Από τον πίνακα παρατηρούμε ότι οι αριθμοί κωδικοποιούνται με τον γνωστό κώδικα BCD-8421 αλλά επιπλέον στον ASCII μπαίνει μπροστά το 011 ενώ στον EBCDIC το 111. Επίσης στα γράμματα υπάρχει μια ανάλογη διαδικασία κωδικοποίησης. Δηλαδή στον ASCII ξεκινάμε με το 0001 για το **Ά**λφα με πρόθεμα

100 και αριθμούμε κανονικά μέχρι το 100 1111, που είναι το **Ό**μικρον. Μετά πάμε στο 101 0000 που είναι το **Ρ** κ.ο.κ. (Περισσότερα για τους κώδικες αυτούς θα βρείτε σε ειδικά βιβλία και βιβλία Η/Υ).

### 2.7.3 Κώδικας UNICODE

Ο κώδικας Unicode Standard είναι ένα παγκόσμιο πρότυπο που χρησιμοποιείται στην κωδικοποίηση κειμένου. Ο σχεδιασμός του βασίζεται στον κώδικα ASCII αλλά παρέχει την δυνατότητα κωδικοποίησης όλων των χαρακτήρων που χρησιμοποιούνται από ένα μεγάλο αριθμό γλωσσών.

Για την **κωδικοποίηση** των χιλιάδων διαφορετικών χαρακτήρων των γλωσσών χρησιμοποιεί ένα κώδικα από **16** χαρακτήρες Bits που παρέχει την δυνατότητα κωδικοποίησης 65.536 ($2^{16}$) διαφορετικών συνδυασμών άρα και χαρακτήρων. Έτσι ο κώδικας Unicode κωδικοποιεί όλους τους χαρακτήρες που χρησιμοποιούνται στις πιο σημαντικές ‘ζωντανές’ σημερινές γραπτές γλώσσες και περιλαμβάνει μεταξύ άλλων Ελληνικούς, Λατινικούς, Κυριλλικούς, Αραβικούς κ.λ.π. χαρακτήρες και συλλογές από ειδικούς χαρακτήρες.

## 2.8 Παράσταση πληροφοριών

Σε όλα τα ψηφιακά συστήματα, στους υπολογιστές οι πληροφορίες παρίστανται με ψηφιακό σήμα το οποίο έχει δυο τιμές, την τιμή μηδέν-‘0’ και την τιμή ένα-‘1’. Χρησιμοποιείται δηλαδή το δυαδικό σύστημα για την παράσταση των πληροφοριών αλλά και την εκτέλεση των διαφόρων πράξεων. Τα βασικά ψηφία που χρησιμοποιούνται είναι το μηδέν-‘0’ και το ένα-‘1’, τα οποία παριστάνουν και την λειτουργία ηλεκτρονικών στοιχείων όπως το Tr (όπου ‘1’=ON ‘0’=OFF). Η παράσταση του μηδέν-‘**0**’ & ένα-‘**1**’ γίνεται με δυο τάσεις (+V) ή (-V) όπου το ένα-‘**1**’ αντιστοιχεί στην (+V ή -V ) και το μηδέν-‘**0**’ σε 0V. (Αν το ένα-‘1’ αντιστοιχεί σε θετική τάση +V τότε έχουμε θετική λογική, αυτή χρησιμοποιείται κυρίως)

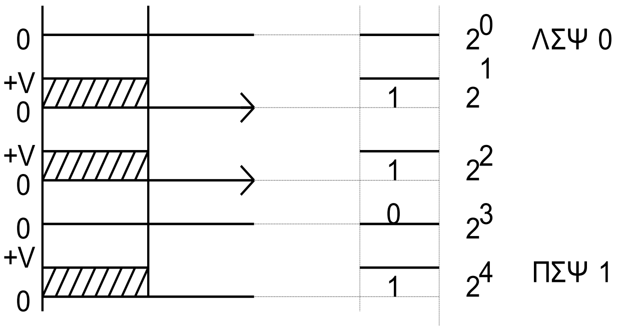
Τα δυαδικό ψηφία μηδέν-‘0’ και ένα-‘1’ στην τεχνολογία TTL, παριστάνονται με ένα τετραγωνικό παλμό, όπου το μηδέν-‘0’ αντιστοιχεί σε 0V και το ‘ένα-‘1’ σε +5V. Η πρώτη αντιστοιχεί στην Χαμηλή Λογική κατάσταση - Low Logic Level και η δεύτερη στην Υψηλή Λογική Κατάσταση - High Logic Level.

Οι παραπάνω τιμές είναι οι θεωρητικές ενώ στην πραγματικότητα οι δυο λογικές στάθμες εμφανίζονται αν οι τιμές της τάσεως, στην τεχνολογία TTL, είναι μεταξύ 0-1,0V=’**0**’ και από 3,5-5V=’**1**’. Οι ενδιάμεσες τιμές τάσης από 1,0-3,5V είναι ανεπιθύμητες. Για την τεχνολογία CMOS τα αντίστοιχα μεγέθη είναι 0-3,0V=’**0**’ και 5-15V=’**1**’.

Η παράσταση δυαδικών αριθμών, με περισσότερα του ενός ψηφία, και η μετάδοσή τους μέσα στις ηλεκτρονικές ψηφιακές μηχανές γίνεται με δυο τρόπους.

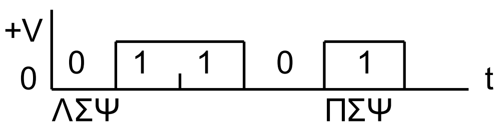
**α) Παράλληλα**

Στον τρόπο αυτό κάθε ψηφίο του αριθμού έχει ένα δικό του αγωγό, μέσα στη μηχανή, ο οποίος έχει τάση ή παλμό και έχει το κατάλληλο "βάρος", όλα δε τα ψηφία του εμφανίζονται ταυτόχρονα. Π.χ Η παράλληλη παράσταση του αριθμού $10110_{(2)}$




**β) Εν Σειρά**

Στον τρόπο αυτό όλα τα ψηφία του αριθμού εμφανίζονται στον ίδιο αγωγό, είτε με την μορφή τάσης είτε με την μορφή παλμών, αλλά χρονικά το ένα ακολουθεί το άλλο κατά ορισμένα χρονικά διαστήματα (Bits times). Συνήθως εμφανίζεται το ΛΣΨ πρώτο. Π.χ Η εν σειρά παράσταση του αριθμού $10110_{(2)}$

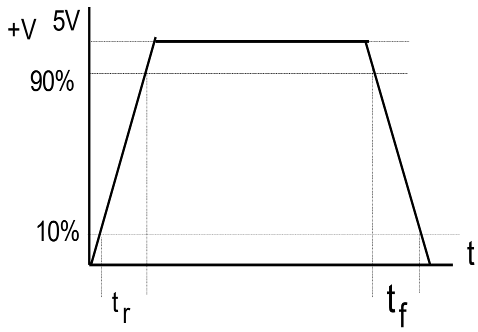



Από τα παραπάνω συνεπάγεται ότι μια λέξη **k**-ψηφίων στον παράλληλο τρόπο εμφανίζεται ταυτοχρόνως σε χρόνο ενός ψηφίου (1 bit times) αλλά σε **k** ιδιαίτερους αγωγούς, ενώ στον εν σειρά τρόπο εμφανίζεται σε <u>ένα αγωγό</u> αλλά απαιτείται χρόνος **k** ψηφίων (**k** bit times). Ο παράλληλος τρόπος απαιτεί περισσότερα κυκλώματα (**k** φορές περισσότερα), αλλά είναι και ταχύτερος κατά **k** φορές του εν σειρά τρόπου.

## 2.9. Ψηφιακό σήμα

Αν τώρα παρατηρήσουμε ένα ψηφιακό σήμα, θεωρητικά, μπορούμε να πούμε ότι η μετάβαση από την μια κατάσταση στην άλλη (από Low σε High ή το αντίστροφο) είναι στιγμιαία. Όμως αυτό δεν είναι πρακτικά δυνατό αφού όλα τα ηλεκτρονικά στοιχεία χρειάζονται κάποιο χρόνο για να λειτουργήσουν, δηλαδή χρειάζονται ένα χρόνο για να αλλάξουν κατάσταση που λέγεται χρόνος μετάβασης (transition time $t_r$).

Ο χρόνος που χρειάζεται το σήμα να ανέλθει από το 10% της τιμής του (λογικό 0) στο 90% της τιμής του (λογικό 1) λέγεται χρόνος ανύψωσης (rise time), ενώ ο χρόνος που χρειάζεται το σήμα να πέσει από το 90% της τιμής του (λογικό ένα-'1') στο 10% της τιμής του (λογικό μηδέν-'0') λέγεται χρόνος πτώσης (fall time).




Σε ένα καλό ψηφιακό σήμα οι δυο παραπάνω χρόνοι πρέπει να είναι οι μικρότεροι δυνατοί ώστε το σήμα να έχει κατά, το δυνατό, κατακόρυφη άνοδο και πτώση

**Μερικές** απαιτήσεις μιας πηγής ψηφιακού σήματος είναι οι παρακάτω:

**1.** Η τάση για το λογικό ένα-'**1'** να είναι πάνω από την ελάχιστη τιμή αλλά όχι μεγαλύτερη από την μέγιστη τιμή. Υψηλότερη τιμή θα προκαλούσε την καταστροφή της εισόδου.

**2.** Η τάση για το λογικό μηδέν-'**0'** να είναι κάτω από την μέγιστη τιμή αλλά όχι μεγαλύτερη από την ελάχιστη τιμή. Χαμηλότερη τιμή θα προκαλούσε την καταστροφή της εισόδου.

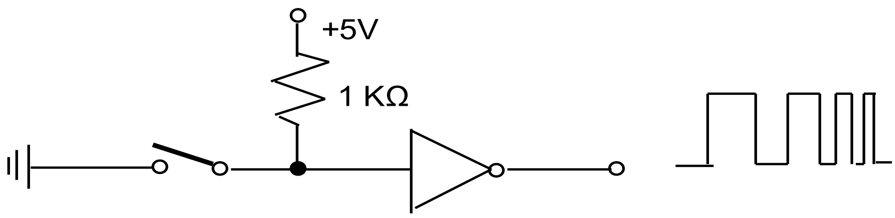
**3.** Η έξοδος της πηγής ψηφιακού σήματος πρέπει να μπορεί να δώσει σε λειτουργία κενού φορτίο αλλά και κάτω από συνθήκες πλήρους φορτίου τις κατάλληλες τιμές τάσης ώστε αυτές να ανταποκρίνονται στις επιτρεπτές τάσεις

Η σταθερότητα του σήματος μέσα στα επιτρεπτά όρια είναι καθοριστικός παράγων καλής λειτουργίας

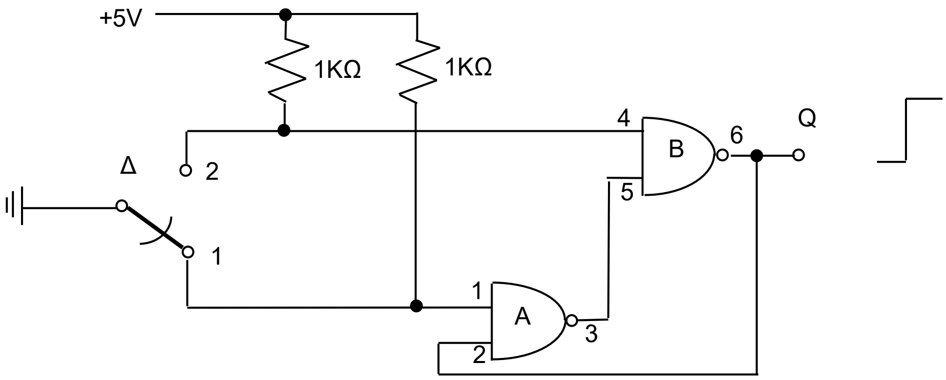
**4.** Αν η πηγή ψηφιακού σήματος είναι διακόπτης πρέπει να χρησιμοποιηθεί σαν είσοδος μια μετάβαση από την **μία** στην **άλλη** κατάσταση ('0' σε '1' ή '1' σε '0') και όχι ολόκληρος παλμός.

**5.** Αν η πηγή ψηφιακού σήματος είναι χρονοπαλμός, ο χρόνος ανύψωσης των παλμών πρέπει να είναι μικρός ώστε η έξοδος της πύλης που οδηγείται από το σήμα αυτό να μην ταλαντώνεται..

Όταν σαν πηγή σήματος χρησιμοποιείται διακόπτης το αναπήδημα το επαφών του διακόπτη διαρκεί μερικά msec. Αυτό δημιουργεί ένα αριθμό παλμών εξόδου αντί της απλής μετάβασης από το '0' στο '1'. Η είσοδος του TTL γίνεται υψηλή μεταξύ δυο διαδοχικών αναπηδήσεων.



Το πρόβλημα διορθώνεται με ένα διαφορετικό κύκλωμα, όπως το επόμενο, όπου




η μεσαία λήψη του διακόπτη **συνδέεται** στην γη. Όταν ο διακόπτης βρίσκεται στη θέση **1** τότε η είσοδος $A_1$ έχει λογικό μηδέν-**'0'**, η έξοδος Q είναι μηδέν-'0' άρα η έξοδος $Q_A$ είναι ένα-'1' επομένως και είσοδος $B_5$ θα είναι ένα-'1'.

Επειδή και η δεύτερη είσοδός της είναι ένα-'1', μέσω της αντίστασης συνδέεται σε λογικό ένα-'**1'**, η έξοδος Q θα είναι μηδέν-'**0'**.

Αν αλλάξει θέση ο διακόπτης και πάει στη θέση 2, τότε $A_4$='1', $B_5$='0' άρα Q='1' οπότε $A_2$='1' και $A_1$='1' άρα η έξοδος $Q_A$='0' και κρατά την έξοδο Q πάντα σε λογικό ένα-'1' ανεξάρτητα από την άλλη της είσοδο.

Άλλα κυκλώματα σχετικά με παλμούς και οδήγηση ψηφιακών κυκλωμάτων θα δούμε στο επόμενο μάθημα των Ψηφιακών Κυκλωμάτων.

## 2.10. Ασκήσεις

**1.** Να μετατραπούν οι δυαδικοί αριθμοί α) 11101011001 β) 10011011 σε ισοδύναμους αριθμούς στον BCD-8421, BCD-2421, BCD-X3 και GRAY

**2.** Να μετατραπεί ο $N_{(10)}$ = 1984 στον BCD-2421 και στον BCD-X3

**3.** Να γίνει η πράξη 43-32 με τον BCD-X3

**4.** Να γράψετε τον BCD του $257_{(10)}$

**5.** Να μετατρέψετε τον δυαδικό 1101101101 στον GRAY

**6.** Να μετατρέψετε τον GRAY 110110101 σε δυαδικό

**7** Να σχεδιαστεί κύκλωμα ελέγχου άρτιας ισοτιμίας για 4 ψηφία. Η έξοδος του κυκλώματος είναι ένα όταν τα 4 ψηφία δεν έχουν άρτια ισοτιμίας.

**8.** Να εκφραστεί ο αριθμός $135_{10}$ στον BCD-8421 άρτιας ισοτιμίας στον κώδικα ASCII και στον EBCDIC.

**9.** Να κωδικοποιηθεί η πληροφορία 1 0 1 0 1 με τον κώδικα Hamming.

**10.** Να γράψετε τον α) $357_{(10)}$ στον BCD άρτιας ισοτιμίας β) $259_{(10)}$ στον BCD-X3 περιττής ισοτιμίας γ) 27 στον ASCII