ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΛΟΠΟΝΝΗΣΟΥ

# Γραμμικοί Κώδικες Μπλοκ

## Θεωρία Πληροφορίας και Κωδίκων

*Εαρινό Εξάμηνο*

Τμήμα Πληροφορικής και Τηλεπικοινωνίων

Νικόλαος Χ. Σαγιάς
*Καθηγητής*
Webpage: https://eclass.uop.gr/courses/DIT221/
e-mail: nsagias@uop.gr

18/5/2020 1:31:44 πμ

# Γραμμικοί Κώδικες Μπλοκ


- Στους κώδικες μπλοκ (*block code*) για κάθε $k$ bit μιας λέξης δεδομένων που εισάγεται στον κωδικοποιητή, εξάγεται μία κωδικολέξη μήκους $n$ bit

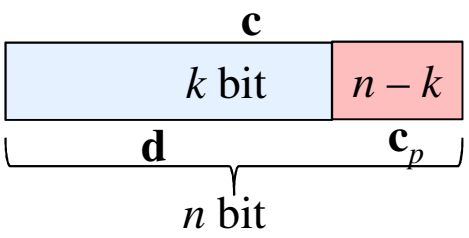
- Οι κώδικες μπλοκ, των οποίων οι λέξεις δεδομένων τοποθετούνται στην αρχή (ή στο τέλος) των κωδικολέξεων αναλλοίωτες, ονομάζονται συστηματικοί (*systematic*)

- Ένας κώδικας μπλοκ ονομάζεται γραμμικός (*linear*) όταν το αποτέλεσμα της πρόσθεσης, με αριθμητική modulo-2, οποιονδήποτε δύο κωδικολέξεων είναι μια άλλη κωδικολέξη του κώδικα

# Γραμμικοί Κώδικες Μπλοκ: Κωδικοποίηση

- Έστω $\mathbf{d} = [d_1, d_2, \ldots, d_k]$ παριστάνει το διάνυσμα της λέξης δεδομένων και $\mathbf{c} = [c_1, c_2, \ldots, c_n]$ παριστάνει το διάνυσμα της αντίστοιχης κωδικολέξης

- Ένας συστηματικός γραμμικός κώδικας μπλοκ γράφεται




$$
\begin{aligned}
c_1 &= d_1 \\
c_2 &= d_2 \\
&\vdots \\
c_k &= d_k
\end{aligned}
$$

$k$ Bit
Λέξη Δεδομένων

$$
\begin{aligned}
c_{k+1} &= p_{11}\, d_1 \oplus p_{12}\, d_2 \oplus \cdots \oplus p_{1k}\, d_k \\
c_{k+2} &= p_{21}\, d_1 \oplus p_{22}\, d_2 \oplus \cdots \oplus p_{2k}\, d_k \\
&\vdots \\
c_n &= p_{m1}\, d_1 \oplus p_{m2}\, d_2 \oplus \cdots \oplus p_{mk}\, d_k
\end{aligned}
$$

$m = n - k$
Bit Ελέγχου

# Γραμμικοί Κώδικες Μπλοκ: Κωδικοποίηση

- Κάθε κωδικολέξη ενός συστηματικού γραμμικού κώδικα μπλοκ σχηματίζεται βάσει του γεννήτορα πίνακα (*generator matrix*) **G** ως εξής

$$\mathbf{c} = \mathbf{d}\,\mathbf{G}$$


- Ο γεννήτορας πίνακας είναι διάστασης $k \times n$ έχει την εξής δομή

$$\mathbf{G} = \underbrace{\begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}}_{\mathbf{I}_k \quad (k \times k)} \underbrace{\begin{bmatrix} p_{11} & p_{21} & \cdots & p_{m1} \\ p_{12} & p_{22} & \cdots & p_{m2} \\ \vdots & \vdots & \cdots & \vdots \\ p_{1k} & p_{2k} & \cdots & p_{mk} \end{bmatrix}}_{\mathbf{P} \quad (k \times m)} = [\mathbf{I}_k \; \mathbf{P}]$$

- Ο **P** είναι ο πίνακας συντελεστών με τιμές {0, 1}

# Γραμμικοί Κώδικες Μπλοκ: Κωδικοποίηση

- Δηλαδή, κάθε το διάνυσμα κάθε κωδικολέξης γράφεται ως συνδυασμός του διανύσματος της λέξης δεδομένων και του διανύσματος των bit ελέγχου

$$\mathbf{c} = \mathbf{d}\,\mathbf{G} \Leftrightarrow \mathbf{c} = \mathbf{d}\,[\mathbf{I}_k \;\; \mathbf{P}] \Leftrightarrow \mathbf{c} = [\mathbf{d} \;\; \mathbf{d}\,\mathbf{P}] \Leftrightarrow \mathbf{c} = [\mathbf{d} \;\; \mathbf{c}_p]$$

- Δεδομένου της γραμμικότητας του κώδικα, από την πρόσθεση, με αριθμητική modulo-2, οποιονδήποτε δύο κωδικολέξεων προκύπτει μια έγκυρη κωδικολέξη του κώδικα

- Η ελάχιστη απόσταση Hamming ενός γραμμικού κώδικα μπλοκ προκύπτει αφού υπολογιστούν οι αποστάσεις Hamming μεταξύ όλων των κωδικολέξεων του κώδικα

- Συνεπώς, η ελάχιστη απόσταση Hamming ενός γραμμικού κώδικα μπλοκ προκύπτει από την κωδικολέξη με το ελάχιστο βάρος Hamming (πλην της κωδικολέξης του μηδενικού διανύσματος)

# Γραμμικοί Κώδικες Μπλοκ: Κωδικοποίηση

- Παράδειγμα: Έστω ο γεννήτορας πίνακας ενός κώδικα μπλοκ

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

- Η διάσταση του πίνακα **G** είναι $k \times n = 3 \times 6$, δηλαδή ο κώδικάς μπλοκ είναι ο $(n, k) = (6, 3)$

- Ο κώδικας ικανοποιεί το όριο Hamming αφού

$$n \leq 2^m - 1 \Leftrightarrow 6 \leq 2^3 - 1 = 7$$

- Με πολλαπλασιασμό της κάθε λέξης δεδομένων **d** με τον **G** προκύπτουν οι αντίστοιχες κωδικολέξεις **c**

- Παρατηρούμε ότι ο κώδικας μπλοκ είναι συστηματικός, αφού τα 3 πρώτα bit της κάθε κωδικολέξης είναι η λέξη δεδομένων

| Λέξη Δεδομένων | Κωδικολέξη |
|---|---|
| 000 | 000000 |
| 001 | 001110 |
| 010 | 010011 |
| 011 | 011101 |
| 100 | 100101 |
| 101 | 101011 |
| 110 | 110110 |
| 111 | 111000 |

# Γραμμικοί Κώδικες Μπλοκ: Κωδικοποίηση

- Παράδειγμα (*συνέχεια*): Επιπλέον, παρατηρούμε ότι ο κώδικας μπλοκ είναι γραμμικός, αφού κάθε αποτέλεσμα πρόσθεσης 2 οποιονδήποτε κωδικολέξεων παράγει μία από τις κωδικολέξεις του κώδικα, πχ `001110` ⊕ `010011` = `011101`, `001110` ⊕ `101011` = `100101`

- Εφόσον ο κώδικας μπλοκ είναι γραμμικός, για να βρούμε την ελάχιστη απόσταση Hamming, αρκεί να βρούμε την κωδικολέξη με το ελάχιστο βάρος και αυτό είναι $d_{\min} = 3$

- Άρα, ο κώδικας (6, 3) μπορεί να διορθώσει έως 1 λάθος

$$t = \left\lfloor \frac{d_{\min} - 1}{2} \right\rfloor = 1$$

- Επίσης, ο συγκεκριμένος (6, 3) μπορεί να ανιχνεύσει έως 2 λάθη

$$t' = d_{\min} - 1 = 2$$

| Λέξη Δεδομένων | Κωδικολέξη |
|---|---|
| 000 | 000000 |
| 001 | 001110 |
| 010 | 010011 |
| 011 | 011101 |
| 100 | 100101 |
| 101 | 101011 |
| 110 | 110110 |
| 111 | 111000 |

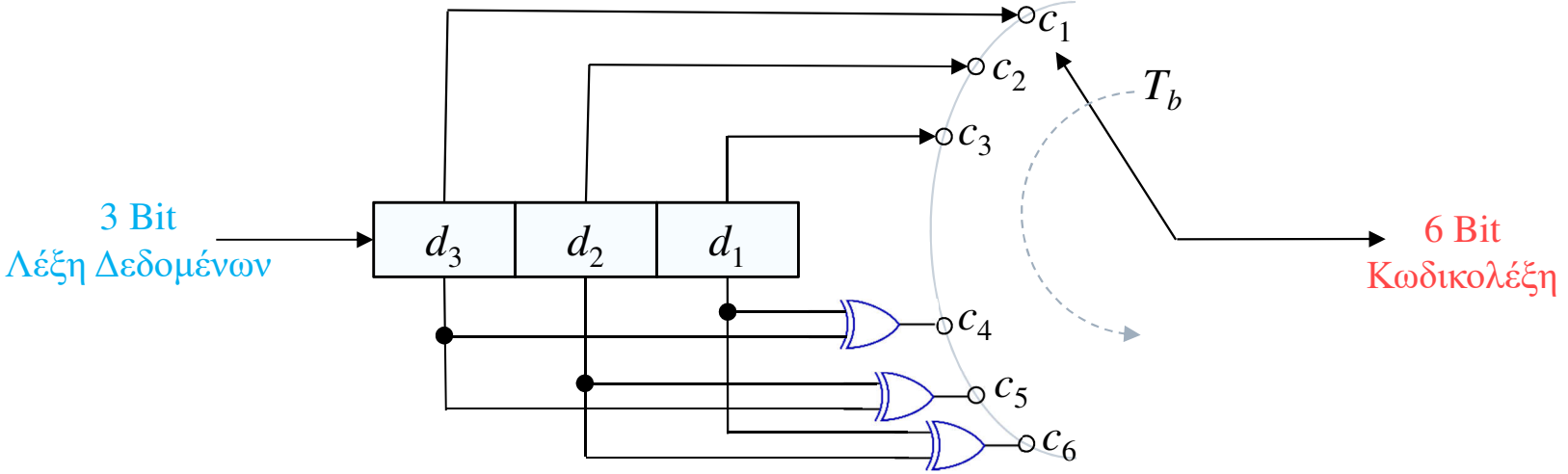
# Γραμμικοί Κώδικες Μπλοκ: Κωδικοποίηση

- Παράδειγμα (*συνέχεια*): Με βάση το γεννήτορα πίνακα του κώδικα μπλοκ μπορούμε να υλοποιήσουμε τον αντίστοιχο κωδικοποιητή

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

$\mathbf{P}$

- Ποιο συγκεκριμένα το κύκλωμα του κωδικοποιητή υλοποιείται με βάση τις στήλες του πίνακα **P** χρησιμοποιώντας καταχωρητές ολίσθησης και πύλες XOR



# Γραμμικοί Κώδικες Μπλοκ: Αποκωδικοποίηση I

- Ας ορίσουμε τον πίνακα

$$\mathbf{H} = \underbrace{\begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1k} \\ p_{21} & p_{22} & \cdots & p_{2k} \\ \vdots & \vdots & \cdots & \vdots \\ p_{m1} & p_{m2} & \cdots & p_{mk} \end{bmatrix}}_{\mathbf{P}^{\mathsf{T}}\ (m \times k)} \underbrace{\begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}}_{\mathbf{I}_m\ (m \times m)} = [\mathbf{P}^{\mathsf{T}}\ \mathbf{I}_m]$$

- Υπολογίζοντας το γινόμενο $\mathbf{H}\,\mathbf{G}^{\mathsf{T}}$ προκύπτει η παρακάτω ιδιότητα

$$\mathbf{H}\,\mathbf{G}^{\mathsf{T}} = [\mathbf{P}^{\mathsf{T}}\ \mathbf{I}_m]\,[\mathbf{I}_k\ \mathbf{P}]^{\mathsf{T}} = [\mathbf{P}^{\mathsf{T}}\ \mathbf{I}_m] \begin{bmatrix} \mathbf{I}_k \\ \mathbf{P}^{\mathsf{T}} \end{bmatrix} = \mathbf{P}^{\mathsf{T}}\,\mathbf{I}_k \oplus \mathbf{I}_m\,\mathbf{P}^{\mathsf{T}} = \mathbf{P}^{\mathsf{T}} + \mathbf{P}^{\mathsf{T}} = \mathbf{0}$$

- Ισοδύναμα, εφαρμόζοντας και στα δύο μέλη τον ανάστροφο πίνακα επιπλέον προκύπτει

$$\mathbf{H}\,\mathbf{G}^{\mathsf{T}} = \mathbf{0} \Leftrightarrow (\mathbf{H}\,\mathbf{G}^{\mathsf{T}})^{\mathsf{T}} = \mathbf{0}^{\mathsf{T}} \Leftrightarrow (\mathbf{G}^{\mathsf{T}})^{\mathsf{T}}\,\mathbf{H}^{\mathsf{T}} = \mathbf{0} \Leftrightarrow \mathbf{G}\,\mathbf{H}^{\mathsf{T}} = \mathbf{0}$$

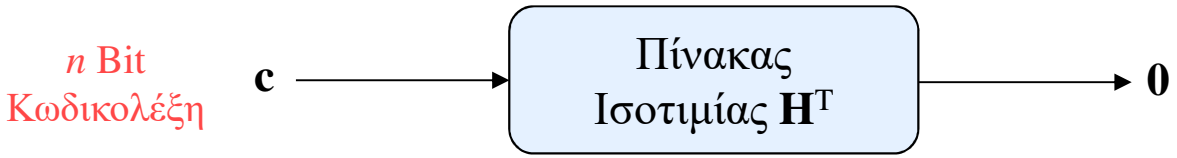
# Γραμμικοί Κώδικες Μπλοκ: Αποκωδικοποίηση I

- Αν στην εξίσωση $\mathbf{G}\,\mathbf{H}^T = \mathbf{0}$ πολλαπλασιάσουμε και τα δύο μέλη με $\mathbf{d}$ προκύπτει

$$\mathbf{G}\,\mathbf{H}^T = \mathbf{0} \Leftrightarrow \underbrace{\mathbf{d}\,\mathbf{G}}_{\mathbf{c}}\,\mathbf{H}^T = \mathbf{d}\,\mathbf{0} \Leftrightarrow \mathbf{c}\,\mathbf{H}^T = \mathbf{0}$$

- Ο πίνακας $\mathbf{H}$ (διάστασης $m \times n$) ονομάζεται πίνακας ελέγχου ισοτιμίας (*parity check matrix*)

- Η παραπάνω εξίσωση ισχύει για κάθε έγκυρη κωδικολέξη του κώδικα



# Γραμμικοί Κώδικες Μπλοκ: Αποκωδικοποίηση Ι

- Αν **r** είναι η κωδικολέξη που λαμβάνει ο δέκτης, τότε αυτή γράφεται ως αποτέλεσμα άθροισης μεταξύ μιας έγκυρης κωδικολέξης **c** και ενός διανύσματος σφάλματος $\mathbf{e} = [e_1, e_2, \ldots, e_n]$, δηλαδή

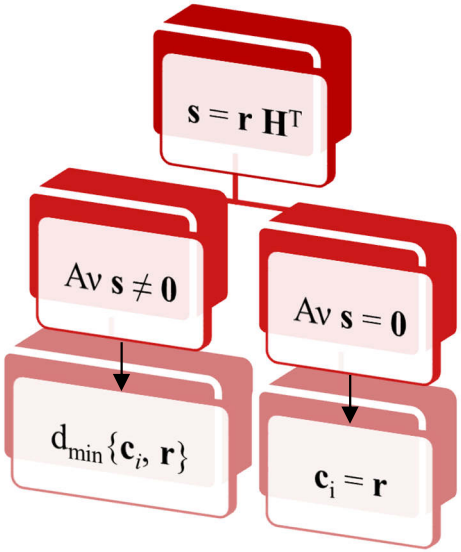
$$\mathbf{r} = \mathbf{c} \oplus \mathbf{e}$$

- Πολλαπλασιάζοντας το **r** με τον $\mathbf{H}^T$ έχουμε το διάνυσμα **s**, το οποίο ονομάζεται σύνδρομο (*syndrome*) και σχετίζεται με το διάνυσμα σφάλματος ως εξής

$$\mathbf{s} = \mathbf{r}\,\mathbf{H}^T = [\mathbf{c} \oplus \mathbf{e}]\,\mathbf{H}^T = \underbrace{\mathbf{c}\,\mathbf{H}^T}_{\mathbf{0}} \oplus \mathbf{e}\,\mathbf{H}^T = \mathbf{e}\,\mathbf{H}^T$$

- Προφανώς, αν δεν έχει συμβεί σφάλμα **e = 0**, το σύνδρομο είναι το μηδενικό διάνυσμα **s = 0**

# Γραμμικοί Κώδικες Μπλοκ: Αποκωδικοποίηση I




- Η διαδικασία αποκωδικοποίησης μπορεί να γίνει ως εξής:
  - Υπολογισμός συνδρόμου $\mathbf{s} = \mathbf{r}\,\mathbf{H}^T$
  - Αν $\mathbf{s} = \mathbf{0}$, τότε δεν έχει συμβεί σφάλμα, δηλαδή $\mathbf{c}_i = \mathbf{r}$
  - Αν $\mathbf{s} \neq \mathbf{0}$, τότε έχει συμβεί κάποιο σφάλμα και η έγκυρη κωδικολέξη βρίσκεται βάσει της ελάχιστης απόσταση Hamming $d_{\min}\{\mathbf{r}, \mathbf{c}_i\}$
- Ο παραπάνω τρόπος αποκωδικοποίησης απαιτεί να υπάρχουν αποθηκευμένες σε μνήμη όλες οι έγκυρες κωδικολέξεις $\mathbf{c}_i$, ώστε να συγκριθούν με το $\mathbf{r}$
- Η απαίτηση σε μνήμη $n\,2^k$ bit είναι υπερβολικά μεγάλη, ενώ απαιτούνται $n$ πύλες XOR 2 εισόδων, όπου για κάθε $\mathbf{r}$ που λαμβάνεται θα πρέπει να εκτελέσει $2^k$ λογικές πράξεις
- Για παράδειγμα, για κώδικα μπλοκ (63, 57), η απαιτούμενη μνήμη είναι

$$n\,2^k = 63 \times 2^{57} \approx 1.1 \text{ Exabyte!!!} \qquad (1 \text{ Exa} = 10^9 \text{ Giga})$$

# Γραμμικοί Κώδικες Μπλοκ: Αποκωδικοποίηση Ι

- Παράδειγμα (*συνέχεια*): Με βάση τον γεννήτορα πίνακα, ο πίνακας ισοτιμίας είναι ο $\mathbf{H} = [\mathbf{P}^T\ \mathbf{I}_3]$, δηλαδή

$$\mathbf{G} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{I}_3} \quad \underbrace{\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}}_{\mathbf{P}}$$

$$\mathbf{H} = [\mathbf{P}^T\ \mathbf{I}_3] = \begin{bmatrix} \underbrace{1 & 0 & 1}_{\mathbf{P}^T = \mathbf{P}} & \underbrace{1 & 0 & 0}_{\mathbf{I}_3} \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

- Ο ανάστροφος πίνακας ισοτιμίας είναι

$$\mathbf{H}^T = \begin{bmatrix} \mathbf{P} \\ \mathbf{I}_3 \end{bmatrix} = \begin{matrix} \left.\begin{matrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{matrix}\right\} \mathbf{P} \\ \left.\begin{matrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{matrix}\right\} \mathbf{I}_3 \end{matrix}$$

| Λέξη Δεδομένων | Κωδικολέξη |
|---|---|
| 000 | 000000 |
| 001 | 001110 |
| 010 | 010011 |
| 011 | 011101 |
| 100 | 100101 |
| 101 | 101011 |
| 110 | 110110 |
| 111 | 111000 |

# Γραμμικοί Κώδικες Μπλοκ: Αποκωδικοποίηση I

- Παράδειγμα (*συνέχεια*): Για ένα διάνυσμα λήψης **r** = [111000], δηλαδή για μια έγκυρη κωδικολέξη, το σύνδρομο υπολογίζεται

$$\mathbf{s} = \mathbf{r}\,\mathbf{H}^{\mathsf{T}} = [1\ 1\ 1\ 0\ 0\ 0] \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = [0\ 0\ 0]$$

- Για ένα διάνυσμα λήψης **r** = [011000], δηλαδή για μια έγκυρη κωδικολέξη, το σύνδρομο υπολογίζεται

$$\mathbf{s} = \mathbf{r}\,\mathbf{H}^{\mathsf{T}} = [0\ 1\ 1\ 0\ 0\ 0] \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = [1\ 0\ 1]$$

| Λέξη Δεδομένων | Κωδικολέξη |
|---|---|
| 000 | 000000 |
| 001 | 001110 |
| 010 | 010011 |
| 011 | 011101 |
| 100 | 100101 |
| 101 | 101011 |
| 110 | 110110 |
| 111 | 111000 |

# Γραμμικοί Κώδικες Μπλοκ: Αποκωδικοποίηση I

- Παράδειγμα (*συνέχεια*): Για το διάνυσμα λήψης **r** = [011000] προέκυψε μη μηδενικό σύνδρομο

- Κατά τον υπολογισμό της απόστασης Hamming μεταξύ του **r** και όλων των έγκυρων κωδικολέξεων **c** προκύπτει ο πίνακας

- Βάσει της ελάχιστης απόστασης Hamming, η έγκυρη κωδικολέξη είναι η

**c** = 111000

| d{r, c} | Κωδικολέξη |
|---|---|
| 2 | 000000 |
| 3 | 001110 |
| 3 | 010011 |
| 2 | 011101 |
| 5 | 100101 |
| 4 | 101011 |
| 4 | 110110 |
| 1 | 111000 |

# Γραμμικοί Κώδικες Μπλοκ: Αποκωδικοποίηση ΙΙ

- Ποιο αποδοτική είναι η αποκωδικοποίηση με χρήση του πίνακα τυπικής διάταξης (*standard array*)
- Αναλυτικότερα, η εξίσωση για το σύνδρομο $\mathbf{s} = \mathbf{e}\,\mathbf{H}^{\mathrm{T}} = \mathbf{e}\,[\mathbf{P}^{\mathrm{T}}\ \mathbf{I}_m]^{\mathrm{T}} = \mathbf{e}\begin{bmatrix}(\mathbf{P}^{\mathrm{T}})^{\mathrm{T}}\\ \mathbf{I}_m\end{bmatrix} = \mathbf{e}\begin{bmatrix}\mathbf{P}\\ \mathbf{I}_m\end{bmatrix}$ γράφεται

$$[s_1,\ s_2, \ldots, s_m] = [e_1,\ e_2, \ldots, e_n] \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1m} \\ p_{21} & p_{22} & \cdots & p_{2m} \\ \vdots & \vdots & \cdots & \vdots \\ p_{k1} & p_{k2} & \cdots & p_{km} \\ 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \begin{matrix} \\ \\ \\ \\ \mathbf{P} \\ \\ \\ \\ \mathbf{I}_m \end{matrix}$$

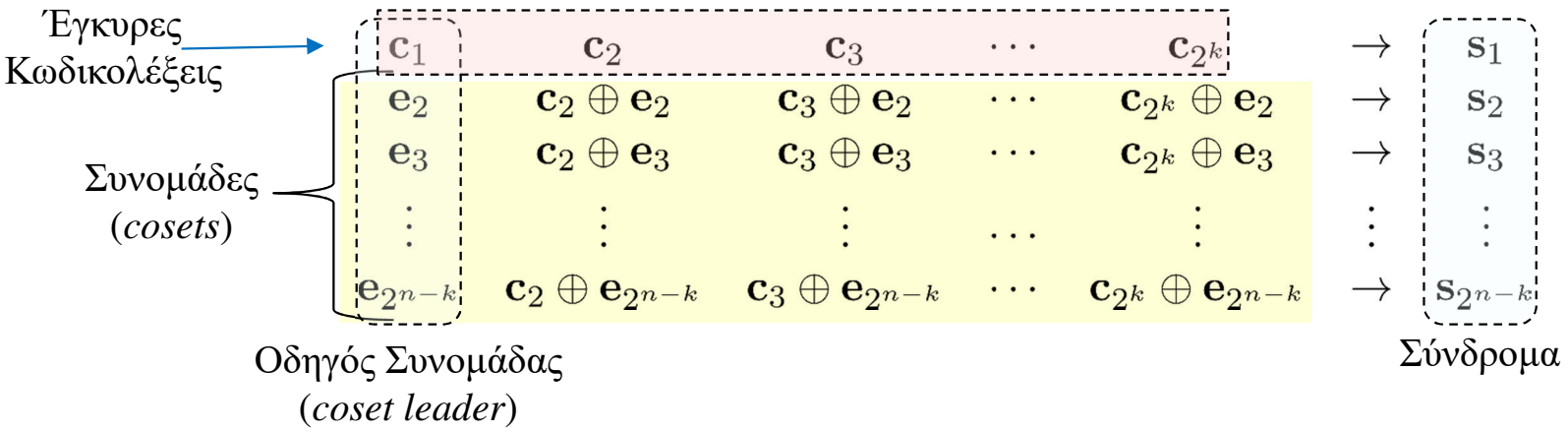
- Πρόκειται για σύστημα $m$ εξισώσεων με $n$ αγνώστους και άρα έχει πολλές λύσεις ως προς **e**
- Ειδικότερα, υπάρχουν $2^{n-k}$ δυνατές διαφορετικές λύσεις

# Γραμμικοί Κώδικες Μπλοκ: Αποκωδικοποίηση ΙΙ

- Ο πίνακας τυπικής διάταξης ενός κώδικα μπλοκ συνοψίζει σε στήλες ανά έγκυρη κωδικολέξη το σύνολο των μη έγκυρων κωδικολέξεων που ο κώδικας μπορεί να διορθώσει

- Η κάθε γραμμή περιλαμβάνει κωδικολέξεις με το ίδιο σφάλμα και αντιστοιχεί στο ίδιο σύνδρομο, ενώ διαφορετικές γραμμές έχουν διαφορετικά σύνδρομα

- Στην 1η στήλη βρίσκεται η κωδικολέξη $\mathbf{c}_1 = \mathbf{0}$ και οπωσδήποτε $\mathbf{s}_1 = \mathbf{0}$



# Γραμμικοί Κώδικες Μπλοκ: Αποκωδικοποίηση ΙΙ

- Παράδειγμα (*συνέχεια*): Η στήλη με τα σύνδρομα υπολογίζεται εύκολα από την σχέση $\mathbf{s} = \mathbf{e}\,\mathbf{H}^T$

- Η πίνακας τυπικής διάταξης του κώδικα μπλοκ (6, 3) είναι παρακάτω

| **c** Έγκυρες Κωδικολέξεις / Συνομάδες | Κωδικολέξεις **r** | | | | | | | Σύνδρομο **s** |
|---|---|---|---|---|---|---|---|---|
| | 000000 | 001110 | 010011 | 011101 | 100101 | 101011 | 110110 | 111000 | 000 |
| | 100000 | 101110 | 110011 | 111101 | 000101 | 001011 | 010110 | 011000 | 101 |
| | 010000 | 011110 | 000011 | 001101 | 110101 | 111011 | 100110 | 101000 | 011 |
| | 001000 | 000110 | 010011 | 010101 | 101101 | 100011 | 111110 | 110000 | 110 |
| | 000100 | 001010 | 010111 | 011001 | 100001 | 101111 | 110010 | 111100 | 100 |
| | 000010 | 001100 | 010001 | 011111 | 100111 | 101001 | 110100 | 111010 | 010 |
| | 000001 | 001111 | 010010 | 011100 | 100100 | 101010 | 110111 | 111001 | 001 |
| **e** Οδηγός Συνομάδας | 100010 | 101100 | 110001 | 111111 | 000111 | 001001 | 010100 | 011010 | 111 |

# Γραμμικοί Κώδικες Μπλοκ: Αποκωδικοποίηση ΙΙ

- Η αποκωδικοποίηση βάσει του πίνακα τυπικής διάταξης συνοψίζεται στα παρακάτω βήματα


- Υπολογισμός συνδρόμου $\mathbf{s} = \mathbf{r}\,\mathbf{H}^T$
  - Αν $\mathbf{s} = \mathbf{0}$, τότε δεν έχει συμβεί σφάλμα, δηλαδή $\mathbf{c}_i = \mathbf{r}$
  - Αν $\mathbf{s} \neq \mathbf{0}$, τότε το σφάλμα προκύπτει από τη γραμμή που αντιστοιχίζει το σύνδρομο με το σφάλμα $\mathbf{e}$
    - Η κωδική λέξη υπολογίζεται

$$\mathbf{c} = \mathbf{e} \oplus \mathbf{r}$$

- Οι απαιτήσεις σε μνήμη της παραπάνω μεθόδου είναι:
  - $2^m$ σύνδρομα των $m$ bit
  - $2^m$ διανύσματα λάθους των $n$ bit

- Δηλαδή, οι απαιτήσεις σε μνήμη είναι $(2\,n - k)\,2^m$ bit (υψηλή)

# Γραμμικοί Κώδικες Μπλοκ: Αποκωδικοποίηση ΙΙ

- Παράδειγμα (*συνέχεια*): Από την σχέση $\mathbf{s} = \mathbf{e}\,\mathbf{H}^T$ σχηματίζουμε την αντιστοιχία μεταξύ σφάλματος και συνδρόμου

- Για το διάνυσμα λήψης **r** = [011000], το αποτέλεσμα υπολογισμού του συνδρόμου είναι

$$\mathbf{s} = \mathbf{r}\,\mathbf{H}^T = [0\ 1\ 1\ 0\ 0\ 0] \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = [1\ 0\ 1]$$

- Το σύνδρομο **s** = [101] αντιστοιχεί στο σφάλμα **e** = [100000]

- Άρα, η έγκυρη κωδικολέξη είναι η

$$\mathbf{c} = \mathbf{r} \oplus \mathbf{e} = 011000 \oplus 100000 = 111000$$

| Σφάλμα **e** | Σύνδρομο **s** |
|---|---|
| 000000 | 000 |
| 100000 | 101 |
| 010000 | 011 |
| 001000 | 110 |
| 000100 | 100 |
| 000010 | 010 |
| 000001 | 001 |
| 100010 | 111 |

# Γραμμικοί Κώδικες Μπλοκ: Αποκωδικοποίηση ΙΙ

- Παράδειγμα (*συνέχεια*): Ο κώδικας μπλοκ (6, 3) πέραν των 6 απλών σφαλμάτων έχει επιπλέον ικανότητα διόρθωσης 1 διπλού σφάλματος και συγκεκριμένα του **e** = [100010]

- Η επιπλέον αυτή ικανότητα προκύπτει από το γεγονός ότι ο συγκεκριμένος κώδικας δεν είναι τέλειος, δηλαδή ικανοποιεί το όριο Hamming «καλύτερα» από την ισότητα $n \leq 2^{n-k} - 1$

- Ένας τέλειος κώδικας μπλοκ διορθώνει μόνο απλά σφάλματα

# Γραμμικοί Κώδικες Μπλοκ: Αποκωδικοποίηση ΙΙ

- Παράδειγμα (*συνέχεια*): Το διάνυσμα λήψης $\mathbf{r}$ = [011000], ωστόσο, μπορεί να έχει προκύψει από την κωδική λέξη $\mathbf{c}_1$ = [000000], έχοντας συμβεί 2 σφάλματα $\mathbf{e}$ = [011000]

- Η απόφαση υπέρ της κωδικολέξης που εκπέμφθηκε γίνεται βάσει μέγιστης πιθανοφάνειας

- Η πιθανότητα να συμβούν 2 σφάλματα είναι χαμηλότερη από να συμβεί 1

- Συνεπώς, η αποκωδικοποίηση γίνεται βάσει του σφάλματος με το ελάχιστο βάρος

$$\mathbf{c} = \mathbf{r} \oplus \mathbf{e}_{\min}$$

# Γραμμικοί Κώδικες Μπλοκ: Σχεδίαση

- Για τη σχεδίαση ενός γραμμικού κώδικα μπλοκ δεν υπάρχει κάποια αυστηρή μέθοδος
- Η σχεδίαση κωδίκων μπλοκ γίνεται έχοντας ως δεδομένα τα *n* και *k*
- Βάσει των *n* και *k* σχηματίζουμε τον ανάστροφο πίνακα ισοτιμίας γεμίζοντας:
  - Τις πρώτες *k* γραμμές με διανύσματα των *m* bit όλα διαφορετικά μεταξύ τους (με εξαίρεση τα διανύσματα με βάρη 0 και 1)
  - Τις υπόλοιπες *m* γραμμές έτσι ώστε να σχηματίζεται ο μοναδιαίος πίνακας $\mathbf{I}_m$

$$\mathbf{H}^{\intercal} = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1m} \\ p_{21} & p_{22} & \cdots & p_{2m} \\ \vdots & \vdots & \cdots & \vdots \\ p_{k1} & p_{k2} & \cdots & p_{km} \\ 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{P} \\ \mathbf{I}_m \end{bmatrix}$$

# Γραμμικοί Κώδικες Μπλοκ: Σχεδίαση

- Ο γεννήτορας πίνακας προκύπτει από τον **P** ως εξής

$$\mathbf{G} = [\mathbf{I}_k \ \mathbf{P}] = \begin{bmatrix} 1 & 0 & \cdots & 0 & p_{11} & p_{21} & \cdots & p_{m1} \\ 0 & 1 & \cdots & 0 & p_{12} & p_{22} & \cdots & p_{m2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & 1 & p_{1k} & p_{2k} & \cdots & p_{mk} \end{bmatrix}$$

- Με τον τρόπο αυτό μπορούν να κατασκευαστούν οι δημοφιλείς κώδικες Hamming

| **Κώδικες Hamming** | |
|---|---|
| Πλήθος bit ισοτιμίας: | $m \geq 3$ |
| Μήκος κωδικολέξεων: | $n = 2^m - 1$ |
| Πλήθος bit πληροφορίας: | $k = 2^m - m - 1$ |
| Ελάχιστη απόσταση Hamming: | $d_{\text{min}} = 3$ |
| Ικανότητα διόρθωσης λάθους: | 1 bit |
| Ικανότητα ανίχνευσης λάθους: | 2 bit |

# Γραμμικοί Κώδικες Μπλοκ: Σχεδίαση

- Παράδειγμα: Να σχεδιαστεί κώδικας Hamming με μήκος λέξης δεδομένων 4 bit

- Κάνοντας δοκιμές στην $n \leq 2^m - 1$ για μήκη κωδικολέξεων $n > k = 4$ bit, ο κώδικας είναι ο (7, 4)

| $m = n - k$ | $n$ | $2^m - 1$ |
|---|---|---|
| 1 | 5 | 1 |
| 2 | 6 | 3 |
| 3 | 7 | 7 |

- Ο ανάστροφος πίνακας ισοτιμίας συμπληρώνεται από $k = 4$ διανύσματα των $m = 3$ bit

$$\mathbf{H}^{\mathsf{T}} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{P} \\ \mathbf{I}_3 \end{bmatrix}$$

# Γραμμικοί Κώδικες Μπλοκ: Σχεδίαση

- Παράδειγμα (*συνέχεια*): Ο γεννήτορας πίνακας σχηματίζεται ως εξής

$$\mathbf{G} = [\mathbf{I}_4 \ \mathbf{P}] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

- Κάθε κωδικολέξη του γραμμικού κώδικα μπλοκ σχηματίζεται ως εξής

$$\mathbf{c} = \mathbf{d}\,\mathbf{G}$$

| d | c |
|---|---|
| 0000 | 0000000 |
| 0001 | 0001111 |
| 0010 | 0010101 |
| 0011 | 0011010 |
| 0100 | 0100110 |
| 0101 | 0101001 |
| 0110 | 0110011 |
| 0111 | 0111100 |
| 1000 | 1000011 |
| 1001 | 1001100 |
| 1010 | 1010110 |
| 1011 | 1011001 |
| 1100 | 1100101 |
| 1101 | 1101010 |
| 1110 | 1110000 |
| 1111 | 1111111 |

# Γραμμικοί Κώδικες Μπλοκ: Σχεδίαση

- Παράδειγμα (*συνέχεια*): Από την σχέση $\mathbf{s} = \mathbf{e}\,\mathbf{H}^T$ σχηματίζουμε την αντιστοιχία μεταξύ σφάλματος και συνδρόμου

- Δεδομένου ότι οι κώδικες Hamming είναι πλήρης διορθώνουν μόνο απλά σφάλματα

- Η αντιστοιχία μεταξύ σφάλματος και συνδρόμου γίνεται εύκολα από τον $\mathbf{H}^T$

- Η θέση του σφάλματος προκύπτει από τη γραμμή του $\mathbf{H}^T$

- Το σύνδρομο δεδομένης γραμμής του $\mathbf{H}^T$ προκύπτει από τα περιεχόμενα της γραμμής αυτής

$$\mathbf{H}^T = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

| Σφάλμα **e** | Σύνδρομο **s** |
|---|---|
| 0000000 | 000 |
| 1000000 | 011 |
| 0100000 | 110 |
| 0010000 | 101 |
| 0001000 | 111 |
| 0000100 | 100 |
| 0000010 | 010 |
| 0000001 | 011 |