ΕΘΝΙΚΌ ΜΕΤΣΌΒΙΟ ΠΟΛΥΤΕΧΝΕΊΟ
ΣΧΟΛΉ ΗΛΕΚΤΡΟΛΌΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
Εξάμηνο 4ο

Κεφάλαιο 3

Το επίπεδο ζεύξης δεδομένων

Διδάσκουσα: Επικ. Καθ. Ιωάννα Ρουσσάκη
E-mail: ioanna.roussaki@cn.ntua.gr

Γενικός ρόλος και αναγκαιότητα του επιπέδου ζεύξης δεδομένων

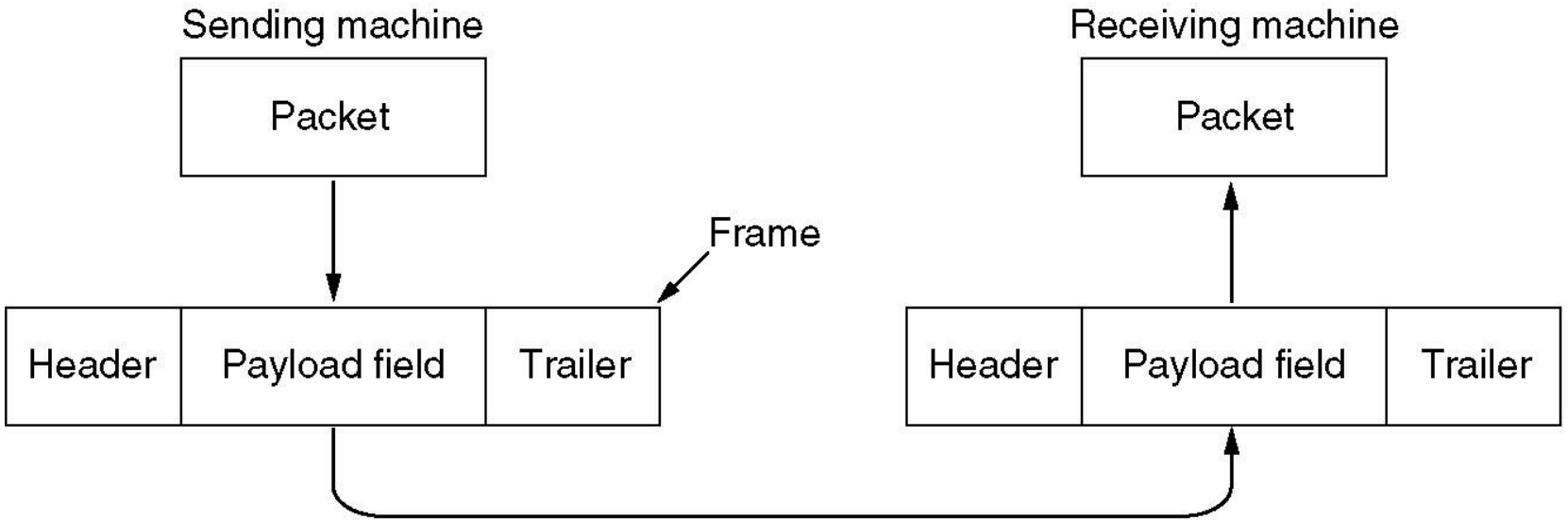
- Κύριος ρόλος του επιπέδου ζεύξης δεδομένων είναι να χρησιμοποιεί τις υπηρεσίες του φυσικού επιπέδου, ήτοι την (ανασφαλή) μεταφορά δεδομένων, και να παρέχει ασφαλή μεταφορά, ανιχνεύοντας ή/και διορθώνοντας τα σφάλματα που παρουσιάζονται.
- Βασικά θέματα εδώ είναι οι αλγόριθμοι και την επίτευξη αξιόπιστης, αποδοτικής επικοινωνίας μεταξύ δύο *γειτονικών* μηχανών στο επίπεδο ζεύξης δεδομένων.
- Με τον όρο γειτονικές, εννοούμε τις δύο μηχανές που είναι φυσικά συνδεδεμένες μέσω ενός καναλιού επικοινωνίας (π.χ., ομοαξονικό καλώδιο ή οπτική ίνα) το οποίο λειτουργεί στην ουσία ως σύρμα.
- Η θεμελιώδης ιδιότητα που κάνει το κανάλι να μοιάζει με σύρμα είναι ότι τα bits παραδίδονται με την ίδια ακριβώς σειρά με την οποία στέλνονται.
- Γιατί χρειαζόμαστε το επίπεδο ζεύξης δεδομένων και το λογισμικό του?
  1. Τα κανάλια επικοινωνίας κάποιες φορές παρουσιάζουν σφάλματα,
  2. έχουν πεπερασμένο ρυθμό μετάδοσης δεδομένων,
  3. παρουσιάζουν μη μηδενική καθυστέρηση διάδοσης και
  4. οι γειτονικές μηχανές έχουν πεπερασμένες και πιθανότατα διαφορετικές ταχύτητες επεξεργασίας.

Βασικές λειτουργίες του επιπέδου ζεύξης δεδομένων

Το επίπεδο ζεύξης δεδομένων υλοποιεί διάφορες λειτουργίες, στις οποίες περιλαμβάνονται οι ακόλουθες:

1. Παροχή μιας καλά ορισμένης διασύνδεσης υπηρεσίας στο επίπεδο δικτύου.
2. Αντιμετώπιση των σφαλμάτων μετάδοσης.
3. Ρύθμιση της ροής των δεδομένων, ώστε οι αργοί παραλήπτες να μην κατακλύζονται με δεδομένα από τους γρήγορους αποστολείς.

Σχέση ανάμεσα σε πακέτα και πλαίσια



- Για να επιτύχει τους στόχους του, το επίπεδο ζεύξης δεδομένων παίρνει τα πακέτα που δέχεται από το επίπεδο δικτύου και τα ενθυλακώνει σε **πλαίσια** (**frames**) προς μετάδοση.
- Κάθε πλαίσιο περιέχει μια **κεφαλίδα** (**header**) πλαισίου, ένα **πεδίο ωφέλιμου φορτίου** (**payload field**) -μέσα στο οποίο περιέχεται το πακέτο- και ένα **επίμετρο** (**trailer**) πλαισίου.
- Η διαχείριση των πλαισίων είναι ο πυρήνας του επιπέδου ζεύξης δεδομένων.

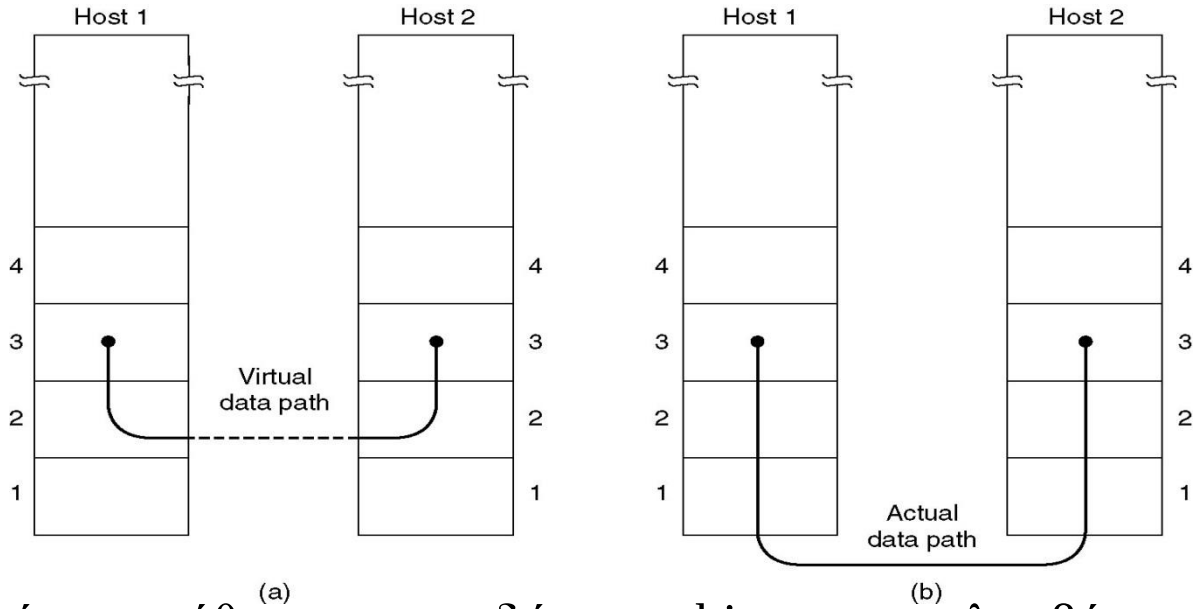
Παρεχόμενες υπηρεσίες προς το επίπεδο δικτύου (I)

- Η θεμελιώδης υπηρεσία που προσφέρει το επίπεδο ζεύξης δεδομένων στο επίπεδο δικτύου είναι η αξιόπιστη μεταφορά δεδομένων από το επίπεδο δικτύου της μηχανής προέλευσης στο επίπεδο δικτύου της μηχανής προορισμού.
- Οι υπηρεσίες που προσφέρει βασίζονται στους ακόλουθους μηχανισμούς που περιλαμβάνει:
  1. πλαισίωση (framing)
  2. συγχρονισμός (synchronization)
  3. έλεγχος ροής (flow control)
  4. ανίχνευση και έλεγχος σφαλμάτων (error detection & control)
- Ανάλογα με την υλοποίησή της, η λογική σύνδεση παρέχει στο επίπεδο δικτύου διαφορετικά είδη υπηρεσίας:
  1. Ασυνδεσμική υπηρεσία χωρίς επιβεβαιώσεις.
  2. Ασυνδεσμική υπηρεσία με επιβεβαιώσεις.
  3. Συνδεσμοστρεφής υπηρεσία με επιβεβαιώσεις.

Παρεχόμενες υπηρεσίες προς το επίπεδο δικτύου (II)

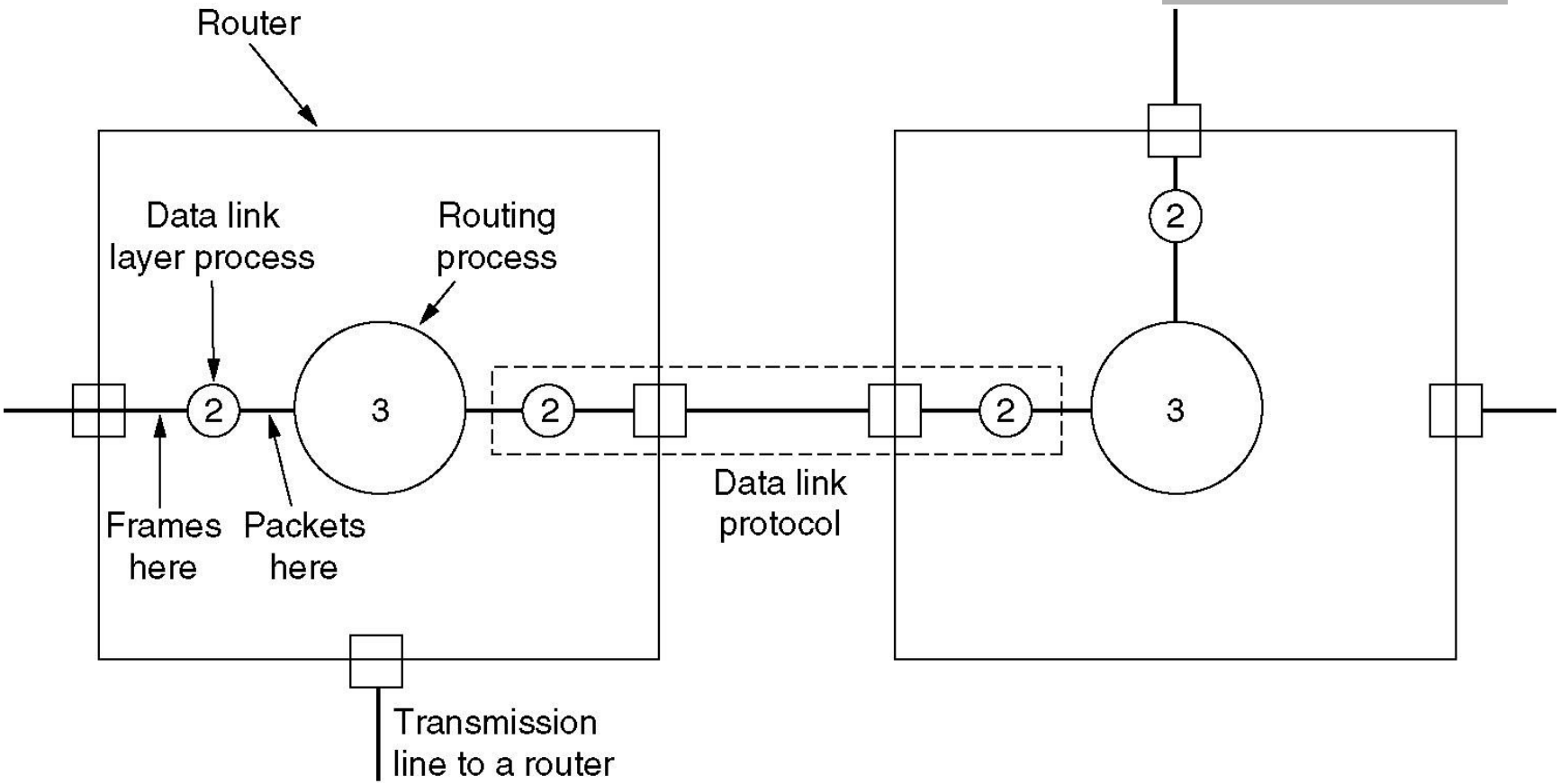
(a) Εικονική επικοινωνία.

(b) Πραγματική επικοινωνία.



Το επίπεδο ζεύξης δεδομένων είναι υπεύθυνο να μεταδώσει τα bit που παραλαμβάνει από τη διεργασία του επιπέδου δικτύου του αποστολέα στην ομότιμη διεργασία του παραλήπτη, όπως φαίνεται στο σχήμα (a). Η πραγματική μετάδοση ακολουθεί τη διαδρομή του σχήματος (b), είναι όμως απλούστερο να τη βλέπουμε ως δύο διεργασίες στο επίπεδο ζεύξης δεδομένων, οι οποίες επικοινωνούν χρησιμοποιώντας ένα πρωτόκολλο ζεύξης δεδομένων. Έτσι, θα θεωρήσουμε ακολούθως το μοντέλο (a).

Τοποθέτηση του πρωτοκόλλου ζεύξης δεδομένων (I)



Παράδειγμα: ένα υποδίκτυο που αποτελείται από δρομολογητές συνδεδεμένους με γραμμές από σημείο-σε-σημείο.

Τοποθέτηση του πρωτοκόλλου ζεύξης δεδομένων (ΙΙ)

Παράδειγμα: ένα υποδίκτυο που αποτελείται από δρομολογητές συνδεδεμένους με γραμμές από σημείο-σε-σημείο.

1. Όταν φτάνει ένα πλαίσιο σε κάποιο δρομολογητή, το hardware το ελέγχει για τυχόν σφάλματα και μετά παραδίδει το πλαίσιο στο λογισμικό του επιπέδου ζεύξης δεδομένων.
2. Το λογισμικό του επιπέδου ζεύξης δεδομένων, το οποίο μπορεί να είναι ενσωματωμένο σε ένα ολοκληρωμένο κύκλωμα στην κάρτα διασύνδεσης με το δίκτυο, ελέγχει αν αυτό είναι το πλαίσιο που αναμένεται και, εφόσον είναι, παραδίδει στο λογισμικό δρομολόγησης (στο επίπεδο δικτύου) το πακέτο που περιέχεται στο πεδίο ωφέλιμου φορτίου (payload field) του πλαισίου.
3. Το λογισμικό δρομολόγησης επιλέγει την κατάλληλη εξερχόμενη γραμμή και παραδίδει ξανά το πακέτο στο λογισμικό του επιπέδου ζεύξης δεδομένων.
4. Το λογισμικό του επιπέδου ζεύξης δεδομένων πλαισιώνει το πακέτο και μεταδίδει το πλαίσιο στην επιλεγμένη εξερχόμενη γραμμή.

Πολλές από τις αρχές του επιπέδου ζεύξης δεδομένων (π.χ., έλεγχος σφάλματος και ροής), απαντώνται στο επίπεδο μεταφοράς αλλά και σε άλλα επίπεδα και πρωτόκολλα.

Πλαισίωση

Τι είναι η πλαισίωση?

Η διαδικασία διάσπασης του αρχικού ρεύματος από bits (ανεπεξέργαστη ροή), που παραδίδεται μέσω του φυσικού επιπέδου, σε κομμάτια, τα οποία τοποθετούνται σε διακριτά πλαίσια (frames).

Γιατί πλαισίωση?

a) Τα πλαίσια εξυπηρετούν και τον σκοπό του ελέγχου λαθών με εισαγωγή bits ελέγχου.

b) Για να είναι δυνατός ο υπολογισμός του αθροίσματος ελέγχου (checksum) για κάθε πλαίσιο.

c) Για την αναγνώριση των περιπτώσεων απωλεσθέντων πλαισίων.

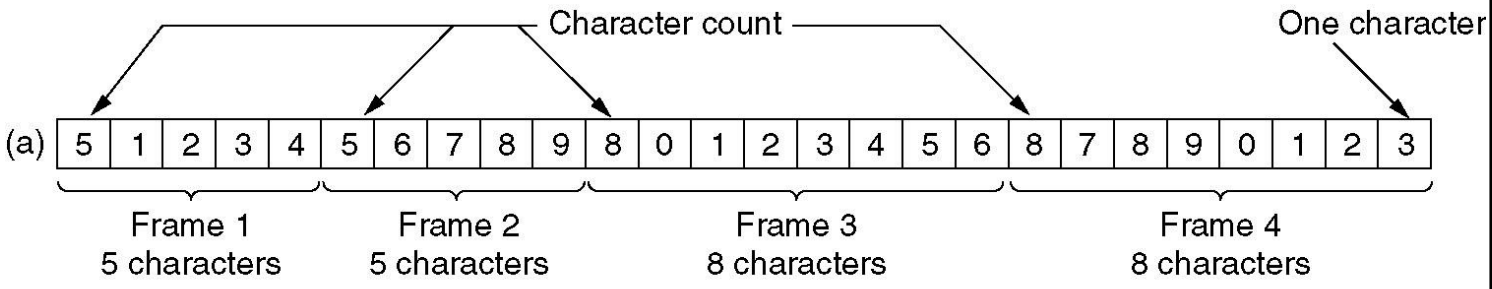
Εντοπισμός των ορίων των πλαισίων

Επειδή είναι παρακινδυνευμένο να βασιζόμαστε στον ακριβή χρονισμό των δικτύων για να σημειώνουμε την αρχή και το τέλος κάθε πλαισίου, έχουν επινοηθεί διάφορες μέθοδοι για τη δουλειά αυτή, οι βασικότερες από τις οποίες είναι:

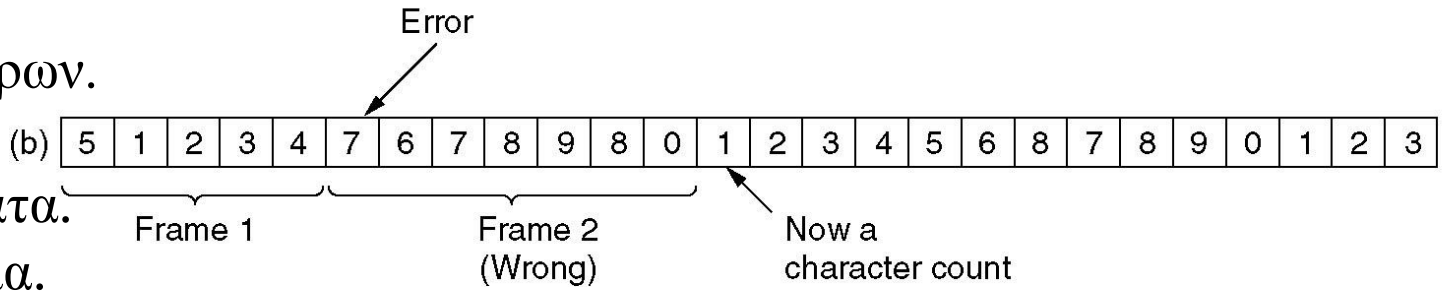
1. Μετρητές χαρακτήρων.
2. Byte σημαίας, με συμπλήρωση byte.
3. Σημαίες αρχής και τέλους, με συμπλήρωση bit.
4. Παραβιάσεις της κωδικοποίησης του φυσικού επιπέδου.

Πλαισίωση με εισαγωγή μετρητών χαρακτήρων

Η μέτρηση χαρακτήρων βασίζεται στην ύπαρξη ενός δείκτη στην αρχή ενός πλαισίου, που δείχνει πόσοι χαρακτήρες ακολουθούν, οπότε αναγνωρίζεται εύκολα το τέλος του. Η μέθοδος αυτή δεν έχει πρακτική εφαρμογή, γιατί αν καταστραφεί ο μετρητής χάνεται ο συγχρονισμός, π.χ.:



Μια ροή χαρακτήρων.



(a) Χωρίς σφάλματα.
(b) Με ένα σφάλμα.

Πλαισίωση με εισαγωγή byte σημαίας και συμπλήρωση byte

Βασίζεται στην ύπαρξη ενός ειδικού χαρακτήρα που οριοθετεί την αρχή και το τέλος κάθε πλαισίου και ονομάζεται σημαία (flag). Ακολουθίες από bits δεδομένων που κατά τύχη μοιάζουν με σημαίες επισημαίνονται με την προσθήκη ενός ESC πριν από αυτές.

(a) Πλαίσιο που οριοθετείται από byte σημαίας.

(b) Τέσσερα παραδείγματα ακολουθιών byte, πριν και μετά τη συμπλήρωση με byte (byte stuffing).

Μειονέκτημα: Η μέθοδος αυτή πλαισίωσης είναι στενά συνδεδεμένη με τη χρήση χαρακτήρων των 8-bit και τον κώδικα χαρακτήρων ASCII, που όμως δεν χρησιμοποιείται από όλους.

Πλαισίωση με εισαγωγή byte σημαίας και συμπλήρωση byte

Βασίζεται στην ύπαρξη ενός ειδικού χαρακτήρα που οριοθετεί την αρχή και το τέλος κάθε πλαισίου και ονομάζεται σημαία (flag). Ακολουθίες από bits δεδομένων που κατά τύχη μοιάζουν με σημαίες επισημαίνονται με την προσθήκη ενός ESC πριν από αυτές.

(a) Πλαίσιο που οριοθετείται από byte σημαίας.

(b) Τέσσερα παραδείγματα ακολουθιών byte, πριν και μετά τη συμπλήρωση με byte (byte stuffing).

(a)

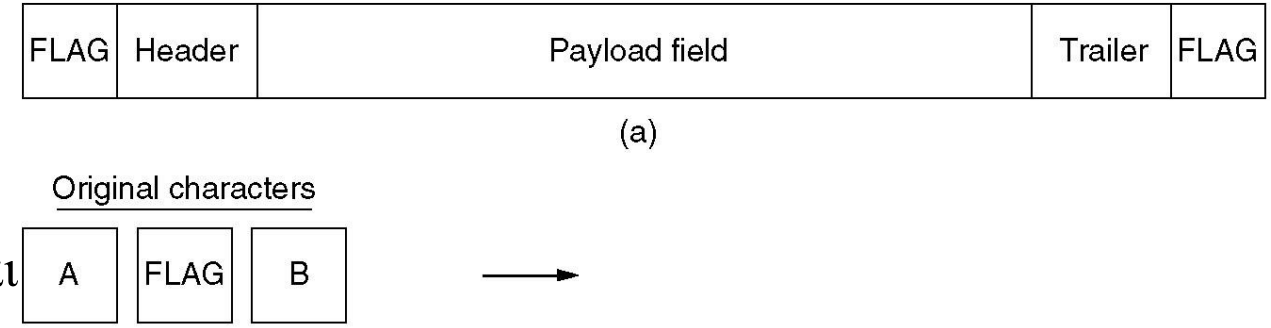
Μειονέκτημα: Η μέθοδος αυτή πλαισίωσης είναι στενά συνδεδεμένη με τη χρήση χαρακτήρων των 8-bit και τον κώδικα χαρακτήρων ASCII, που όμως δεν χρησιμοποιείται από όλους.

Πλαισίωση με εισαγωγή byte σημαίας και συμπλήρωση byte

Βασίζεται στην ύπαρξη ενός ειδικού χαρακτήρα που οριοθετεί την αρχή και το τέλος κάθε πλαισίου και ονομάζεται σημαία (flag). Ακολουθίες από bits δεδομένων που κατά τύχη μοιάζουν με σημαίες επισημαίνονται με την προσθήκη ενός ESC πριν από αυτές.

(a) Πλαίσιο που οριοθετείται από byte σημαίας.

(b) Τέσσερα παραδείγματα ακολουθιών byte, πριν και μετά τη συμπλήρωση με byte (byte stuffing).



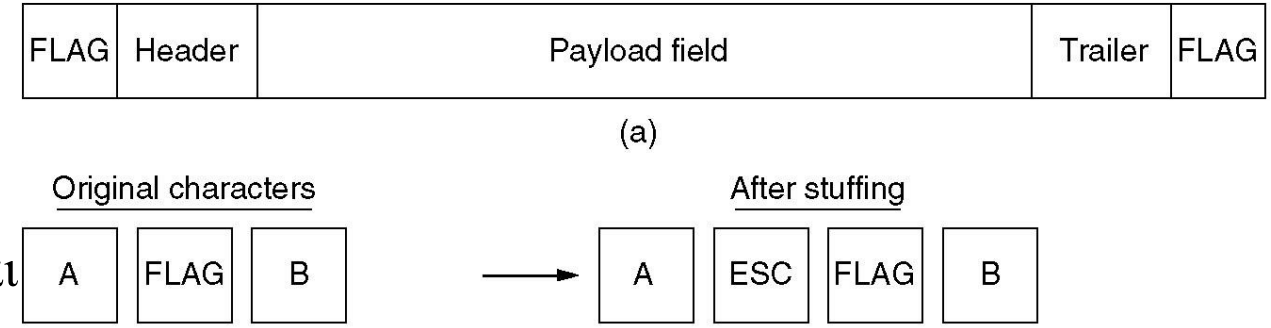
Μειονέκτημα: Η μέθοδος αυτή πλαισίωσης είναι στενά συνδεδεμένη με τη χρήση χαρακτήρων των 8-bit και τον κώδικα χαρακτήρων ASCII, που όμως δεν χρησιμοποιείται από όλους.

Πλαισίωση με εισαγωγή byte σημαίας και συμπλήρωση byte

Βασίζεται στην ύπαρξη ενός ειδικού χαρακτήρα που οριοθετεί την αρχή και το τέλος κάθε πλαισίου και ονομάζεται σημαία (flag). Ακολουθίες από bits δεδομένων που κατά τύχη μοιάζουν με σημαίες επισημαίνονται με την προσθήκη ενός ESC πριν από αυτές.

(a) Πλαίσιο που οριοθετείται από byte σημαίας.

(b) Τέσσερα παραδείγματα ακολουθιών byte, πριν και μετά τη συμπλήρωση με byte (byte stuffing).



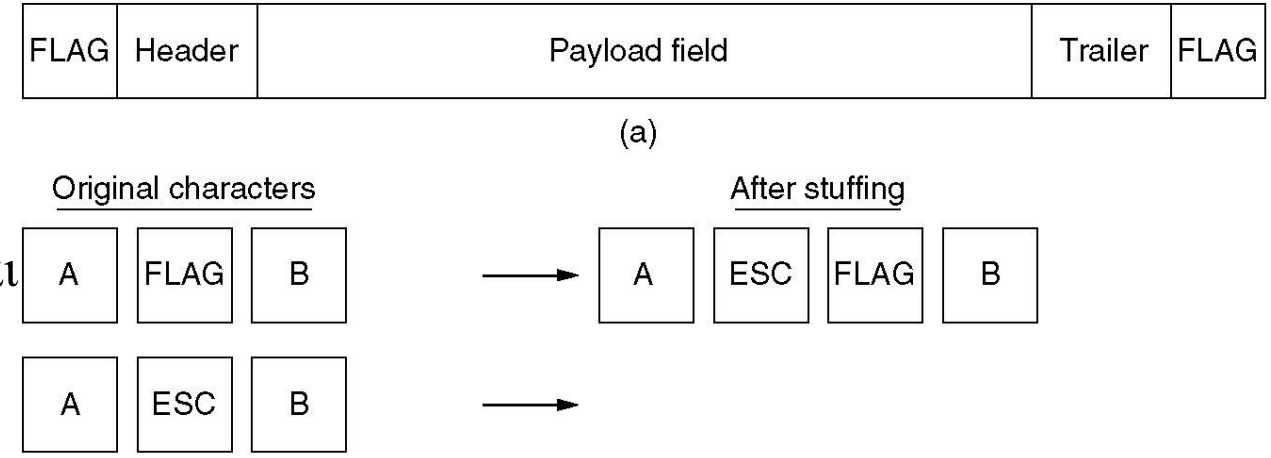
Μειονέκτημα: Η μέθοδος αυτή πλαισίωσης είναι στενά συνδεδεμένη με τη χρήση χαρακτήρων των 8-bit και τον κώδικα χαρακτήρων ASCII, που όμως δεν χρησιμοποιείται από όλους.

Πλαισίωση με εισαγωγή byte σημαίας και συμπλήρωση byte

Βασίζεται στην ύπαρξη ενός ειδικού χαρακτήρα που οριοθετεί την αρχή και το τέλος κάθε πλαισίου και ονομάζεται σημαία (flag). Ακολουθίες από bits δεδομένων που κατά τύχη μοιάζουν με σημαίες επισημαίνονται με την προσθήκη ενός ESC πριν από αυτές.

(a) Πλαίσιο που οριοθετείται από byte σημαίας.

(b) Τέσσερα παραδείγματα ακολουθιών byte, πριν και μετά τη συμπλήρωση με byte (byte stuffing).



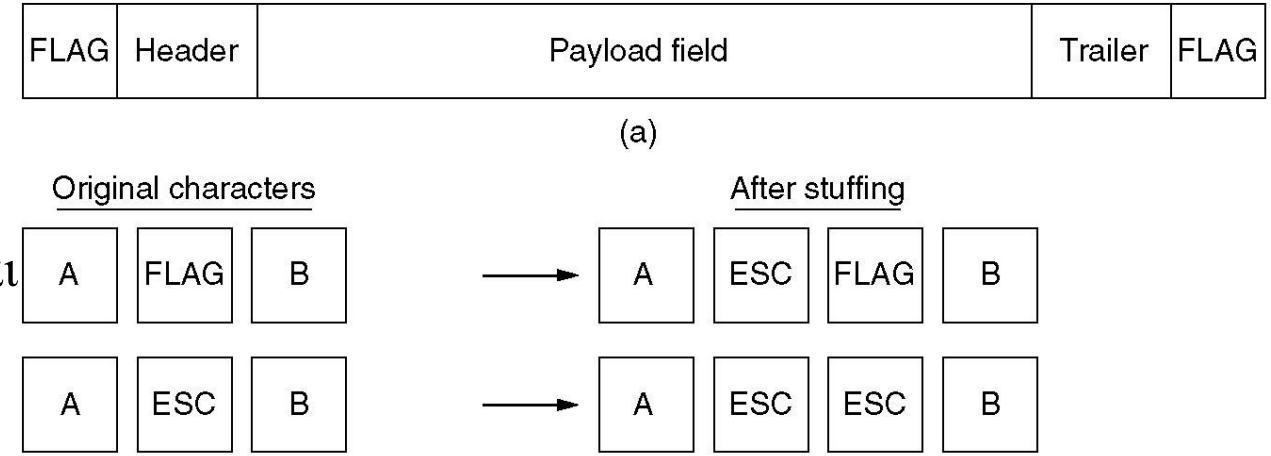
Μειονέκτημα: Η μέθοδος αυτή πλαισίωσης είναι στενά συνδεδεμένη με τη χρήση χαρακτήρων των 8-bit και τον κώδικα χαρακτήρων ASCII, που όμως δεν χρησιμοποιείται από όλους.

Πλαισίωση με εισαγωγή byte σημαίας και συμπλήρωση byte

Βασίζεται στην ύπαρξη ενός ειδικού χαρακτήρα που οριοθετεί την αρχή και το τέλος κάθε πλαισίου και ονομάζεται σημαία (flag). Ακολουθίες από bits δεδομένων που κατά τύχη μοιάζουν με σημαίες επισημαίνονται με την προσθήκη ενός ESC πριν από αυτές.

(a) Πλαίσιο που οριοθετείται από byte σημαίας.

(b) Τέσσερα παραδείγματα ακολουθιών byte, πριν και μετά τη συμπλήρωση με byte (byte stuffing).



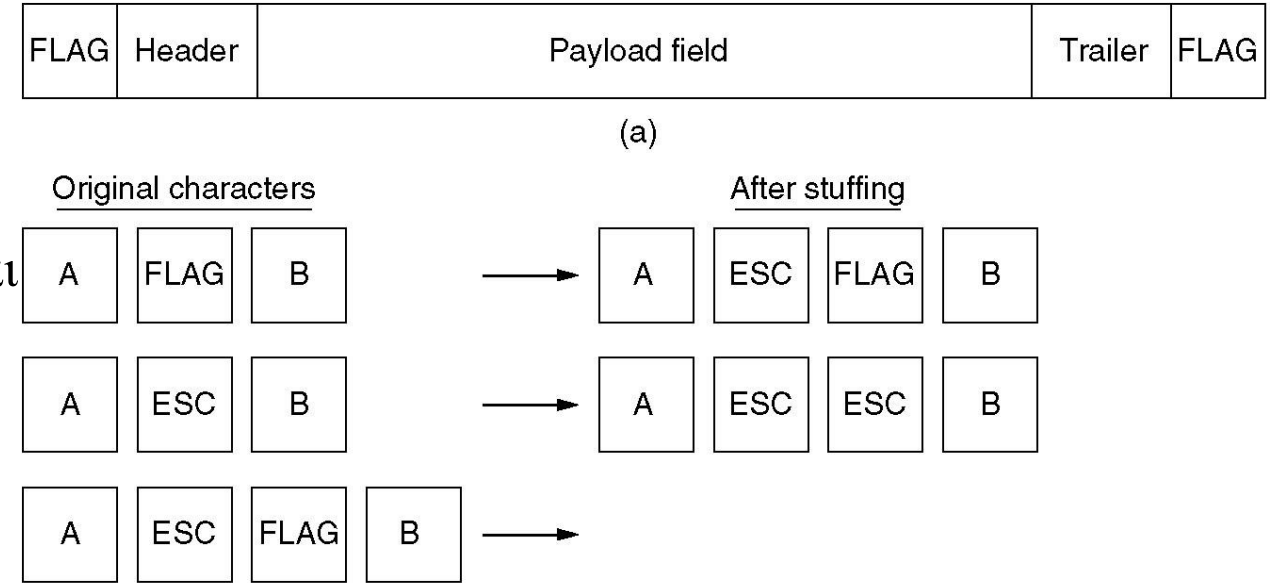
Μειονέκτημα: Η μέθοδος αυτή πλαισίωσης είναι στενά συνδεδεμένη με τη χρήση χαρακτήρων των 8-bit και τον κώδικα χαρακτήρων ASCII, που όμως δεν χρησιμοποιείται από όλους.

Πλαισίωση με εισαγωγή byte σημαίας και συμπλήρωση byte

Βασίζεται στην ύπαρξη ενός ειδικού χαρακτήρα που οριοθετεί την αρχή και το τέλος κάθε πλαισίου και ονομάζεται σημαία (flag). Ακολουθίες από bits δεδομένων που κατά τύχη μοιάζουν με σημαίες επισημαίνονται με την προσθήκη ενός ESC πριν από αυτές.

(a) Πλαίσιο που οριοθετείται από byte σημαίας.

(b) Τέσσερα παραδείγματα ακολουθιών byte, πριν και μετά τη συμπλήρωση με byte (byte stuffing).



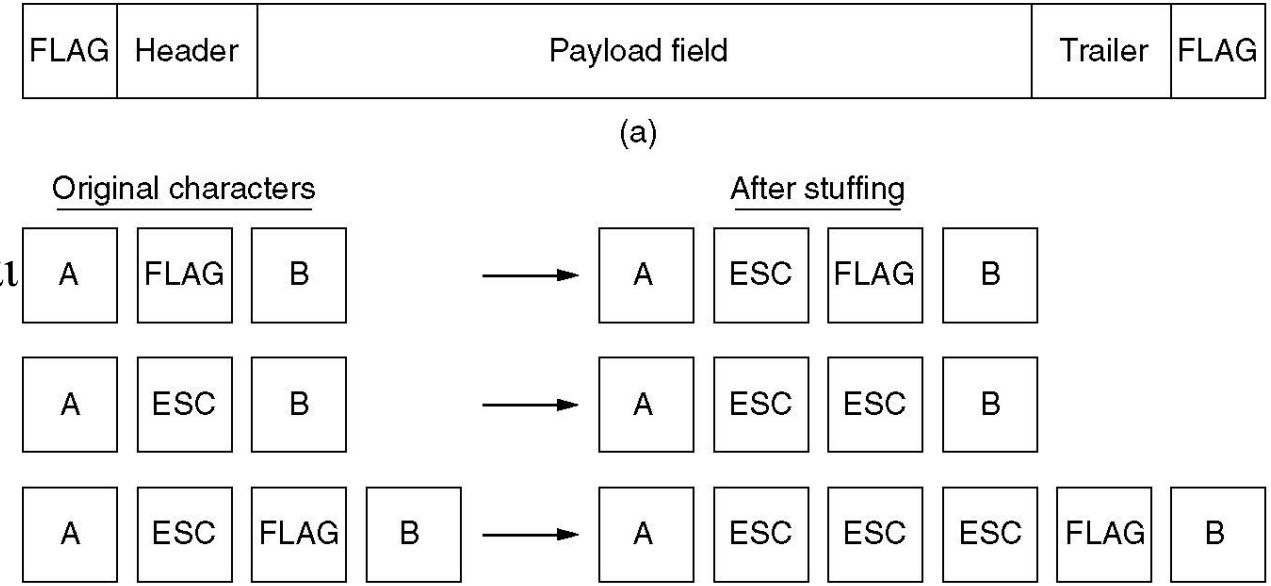
Μειονέκτημα: Η μέθοδος αυτή πλαισίωσης είναι στενά συνδεδεμένη με τη χρήση χαρακτήρων των 8-bit και τον κώδικα χαρακτήρων ASCII, που όμως δεν χρησιμοποιείται από όλους.

Πλαισίωση με εισαγωγή byte σημαίας και συμπλήρωση byte

Βασίζεται στην ύπαρξη ενός ειδικού χαρακτήρα που οριοθετεί την αρχή και το τέλος κάθε πλαισίου και ονομάζεται σημαία (flag). Ακολουθίες από bits δεδομένων που κατά τύχη μοιάζουν με σημαίες επισημαίνονται με την προσθήκη ενός ESC πριν από αυτές.

(a) Πλαίσιο που οριοθετείται από byte σημαίας.

(b) Τέσσερα παραδείγματα ακολουθιών byte, πριν και μετά τη συμπλήρωση με byte (byte stuffing).



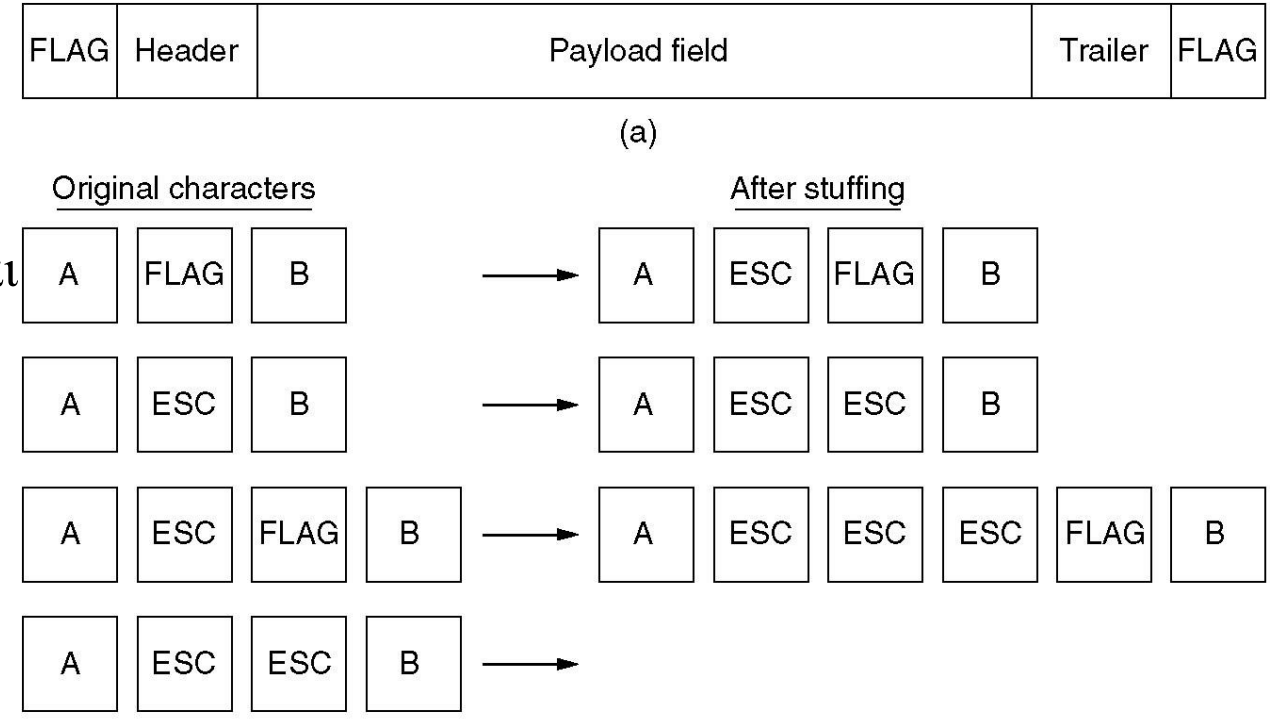
Μειονέκτημα: Η μέθοδος αυτή πλαισίωσης είναι στενά συνδεδεμένη με τη χρήση χαρακτήρων των 8-bit και τον κώδικα χαρακτήρων ASCII, που όμως δεν χρησιμοποιείται από όλους.

Πλαισίωση με εισαγωγή byte σημαίας και συμπλήρωση byte

Βασίζεται στην ύπαρξη ενός ειδικού χαρακτήρα που οριοθετεί την αρχή και το τέλος κάθε πλαισίου και ονομάζεται σημαία (flag). Ακολουθίες από bits δεδομένων που κατά τύχη μοιάζουν με σημαίες επισημαίνονται με την προσθήκη ενός ESC πριν από αυτές.

(a) Πλαίσιο που οριοθετείται από byte σημαίας.

(b) Τέσσερα παραδείγματα ακολουθιών byte, πριν και μετά τη συμπλήρωση με byte (byte stuffing).



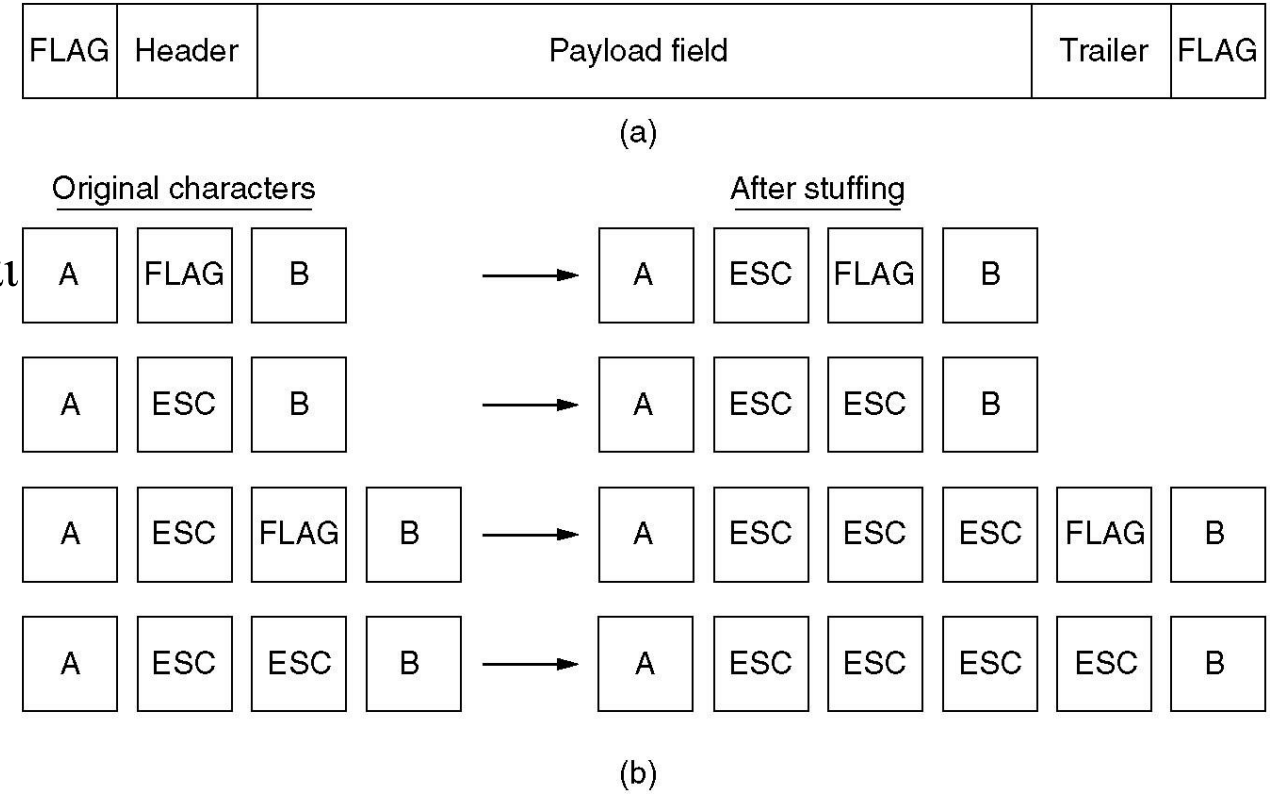
Μειονέκτημα: Η μέθοδος αυτή πλαισίωσης είναι στενά συνδεδεμένη με τη χρήση χαρακτήρων των 8-bit και τον κώδικα χαρακτήρων ASCII, που όμως δεν χρησιμοποιείται από όλους.

Πλαισίωση με εισαγωγή byte σημαίας και συμπλήρωση byte

Βασίζεται στην ύπαρξη ενός ειδικού χαρακτήρα που οριοθετεί την αρχή και το τέλος κάθε πλαισίου και ονομάζεται σημαία (flag). Ακολουθίες από bits δεδομένων που κατά τύχη μοιάζουν με σημαίες επισημαίνονται με την προσθήκη ενός ESC πριν από αυτές.

(a) Πλαίσιο που οριοθετείται από byte σημαίας.

(b) Τέσσερα παραδείγματα ακολουθιών byte, πριν και μετά τη συμπλήρωση με byte (byte stuffing).

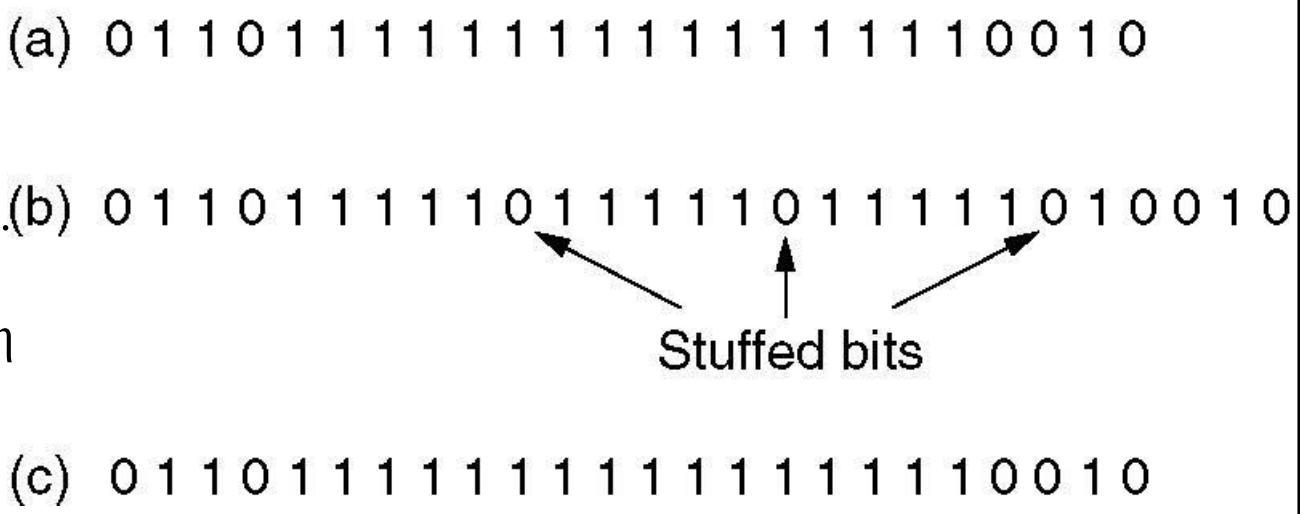


Μειονέκτημα: Η μέθοδος αυτή πλαισίωσης είναι στενά συνδεδεμένη με τη χρήση χαρακτήρων των 8-bit και τον κώδικα χαρακτήρων ASCII, που όμως δεν χρησιμοποιείται από όλους.

Πλαισίωση με χρήση σημαίας και συμπλήρωση bit

Βασίζεται στην προσθήκη μιας ειδικής ακολουθίας από bits (01111110) στην αρχή και στο τέλος κάθε πλαισίου. Σε ακολουθίες από bits δεδομένων που κατά τύχη περιέχουν πέντε ‘1’ στη σειρά προστίθεται κατόπιν ένα ‘0’ από τον αποστολέα (bit stuffing). Αν λάβει ακολούθως ο παραλήπτης πέντε ‘1’ και μετά ‘0’, βγάζει το ‘0’ και συνεχίζει, ενώ αν βρει έξι ‘1’ (ακολουθία σημαίας), τότε σημαίνει ότι τελείωσε το πλαίσιο. Κάποιες φορές χάνονται όμως πακέτα, διότι, αν για παράδειγμα γίνει λάθος στη σημαία, τότε διαβάζει μαζί και το επόμενο πακέτο (ενώ αν έχει δείκτες για το πόσα να περιμένει, το διαπιστώνει).

Συμπλήρωση με bit.
(a) Τα αρχικά δεδομένα.
(b) Τα δεδομένα όπως εμφανίζονται στη γραμμή.
(c) Τα δεδομένα όπως αποθηκεύονται στη μνήμη του παραλήπτη μετά την αντιστροφή της συμπλήρωσης.



Πλαισίωση με παραβιάσεις της κωδικοποίησης του φυσικού επιπέδου

Η μέθοδος αυτή πλαισίωσης εφαρμόζεται μόνο στα δίκτυα όπου η κωδικοποίηση στο φυσικό μέσον παρέχει κάποιον πλεονασμό.

Για παράδειγμα, το πρότυπο για LAN της σειράς IEEE 802 χρησιμοποιεί το ακόλουθο σχήμα κωδικοποίησης (Differential Manchester): το bit ‘1’ κωδικοποιείται ως HL (ζεύγος υψηλής-χαμηλής τιμής) και το bit ‘0’ ως LH (ζεύγος χαμηλής-υψηλής τιμής), χρησιμοποιώντας σε κάθε περίπτωση δύο φυσικά bit για την κωδικοποίηση ενός bit δεδομένων.

Στην περίπτωση αυτή, οι συνδυασμοί HH και LL που δεν χρησιμοποιούνται για τα δεδομένα, μπορούν να χρησιμοποιηθούν για την πλαισίωση.

Σημείωση: Πολλά πρωτόκολλα του επιπέδου ζεύξης δεδομένων χρησιμοποιούν συνδυασμό του μετρητή χαρακτήρων με κάποια από τις άλλες μεθόδους, για πρόσθετη ασφάλεια.

Έλεγχος σφαλμάτων και έλεγχος ροής

Έλεγχος σφαλμάτων: Στόχος μιας αξιόπιστης συνδεσμοστρεφούς υπηρεσίας είναι η εξασφάλιση ότι κάθε πλαίσιο μεταβιβάζεται τελικά στο επίπεδο δικτύου του προορισμού με τη σωστή σειρά, χωρίς σφάλματα και ακριβώς μια φορά. Έτσι χρησιμοποιούνται:

1. Επιβεβαιώσεις (acknowledgements) από τον παραλήπτη στον αποστολέα.
2. Χρονόμετρα (timers) για να εμποδίσουν τους αποστολείς να κολλήσουν αν υπάρξει απώλεια πλαισίου.
3. Αριθμούς ακολουθίας (sequence numbers) για την ανίχνευση της απώλειας πλαισίων και την αντιμετώπιση των διπλών (duplicated) πλαισίων.

Έλεγχος ροής: Αντιμετωπίζει το πρόβλημα που δημιουργείται όταν ο αποστολέας προσπαθεί συστηματικά να μεταδώσει πλαίσια με ρυθμό ταχύτερο από αυτόν που μπορεί να υποστηρίξει ο παραλήπτης. Για τον έλεγχο ροής ακολουθούνται συνήθως δύο προσεγγίσεις:

1. Έλεγχος ροής με ανάδραση (feedback-based flow control): ο παραλήπτης επιστρέφει περιοδικά πληροφορίες στον αποστολέα επιτρέποντάς του να στείλει περισσότερα δεδομένα ή ενημερώνοντάς τον για την κατάσταση του παραλήπτη.
2. Έλεγχος ροής βάσει του ρυθμού (rate-based flow control): το πρωτόκολλο έχει ενσωματωμένο μηχανισμό που περιορίζει το ρυθμό με τον οποίο επιτρέπεται στους αποστολείς να μεταδίδουν δεδομένα, χωρίς να χρειάζεται ανάδραση από τον παραλήπτη.

Ανίχνευση και διόρθωση σφαλμάτων

Τα σφάλματα μετάδοσης είναι και θα παραμείνουν πολύ συχνό φαινόμενο.

Πηγές σφαλμάτων σε τοπικούς βρόχους και ασύρματες ζεύξεις:

1. Θερμικός θόρυβος και ηλεκτρομαγνητικές παρεμβολές.
2. Η εξάρτηση από τη συχνότητα του πλάτους, της ταχύτητας διάδοσης και της παραμόρφωσης της φάσης των σημάτων.
3. Ανακλάσεις του σήματος, πουλιά, κ.λπ.

Τα σφάλματα τείνουν να εμφανίζονται σε ριπές (bursts), παρά μεμονωμένα.

Στρατηγικές για την αντιμετώπιση των σφαλμάτων:

1. **Κωδικοί διόρθωσης σφαλμάτων** (μετάδοση πλεονάζουσας πληροφορίας μαζί με τα δεδομένα, ώστε ο παραλήπτης να μπορεί να συμπεράνει τα αρχικά δεδομένα που μεταδόθηκαν).
2. **Κωδικοί ανίχνευσης σφαλμάτων** (περιλαμβάνει μετάδοση αρκετής πλεονάζουσας πληροφορίας ώστε ο παραλήπτης να καταλαβαίνει ότι συνέβη κάποιο σφάλμα, όχι όμως και ποιο σφάλμα, για να ζητήσει αναμετάδοση των δεδομένων).

Η χρήση κωδικών διόρθωσης σφαλμάτων λέγεται συχνά: **Ευθεία Διόρθωση Σφαλμάτων** (**FEC** – Forward Error Correction).

Κωδικοί διόρθωσης & ανίχνευσης σφαλμάτων (I)

- Κάθε πλαίσιο αποτελείται από m bit δεδομένων και r bit πλεονασμού ή ελέγχου, ενώ έχει συνολικό μέγεθος n ($n = m + r$). Η ομάδα αυτή των n bit (συμπεριλαμβανομένων των bit δεδομένων και των bit ελέγχου) ονομάζεται **κωδικολέξη** (**codeword**) των n bit.
- Το πλήθος των bit που διαφέρουν δύο κωδικολέξεις ονομάζεται **απόσταση Hamming** (**Hamming distance**).
- Για δύο κωδικολέξεις που έχουν απόσταση d απαιτούνται d σφάλματα του ενός bit ώστε να μετατραπεί η μία στην άλλη.
- Όλα τα 2^m πιθανά μηνύματα δεδομένων είναι έγκυρα, αλλά, λόγω του τρόπου που υπολογίζονται τα bit ελέγχου, δεν χρησιμοποιούνται και οι 2^n κωδικολέξεις.

Κωδικοί διόρθωσης & ανίχνευσης σφαλμάτων (II)

- Στη λίστα κωδικολέξεων που προκύπτει βάσει του αλγορίθμου υπολογισμού των bit ελέγχου, η **ελάχιστη απόσταση Hamming δύο κωδικολέξεων, αποτελεί την απόσταση όλου του κωδικού**.

Παράδειγμα κώδικα με 4 μόνον έγκυρες κωδικολέξεις των 10 bits:

- 0000000000
- 0000011111
- 1111100000
- 1111111111

Η απόσταση Hamming του κώδικα αυτού είναι 5.

Όρια απόστασης για διόρθωση & ανίχνευση σφαλμάτων (I)

- Για την **ανίχνευση *d* σφαλμάτων**, απαιτείται απόσταση ***d* + *1*** διότι με τέτοιον κωδικό δεν είναι δυνατόν *d* σφάλματα του ενός bit να μετατρέψουν μια έγκυρη κωδικολέξη σε άλλη έγκυρη κωδικολέξη (π.χ., bit ισοτιμίας ή parity-bit => απόσταση 2, οπότε ανιχνεύει σφάλματα ενός bit).

- Για τη **διόρθωση *d* σφαλμάτων**, απαιτείται απόσταση ***2d* + *1*** διότι έτσι οι έγκυρες κωδικολέξεις είναι τόσο απομακρυσμένες μεταξύ τους, ώστε ακόμη και με *d* αλλαγές, η αρχική κωδικολέξη εξακολουθεί να είναι πιο κοντά από κάθε άλλη κωδικολέξη στο σωστό αποτέλεσμα, το οποίο μπορεί να οριστεί μονοσήμαντα (π.χ., κώδικος με απόσταση 5, διορθώνει σφάλματα των 2 bit).

Όρια απόστασης για διόρθωσης & ανίχνευσης σφαλμάτων (II)

- Στον κώδικα της προ-προηγούμενης διαφάνειας, όπου η απόσταση Hamming είναι 5, μπορεί να γίνει:
  - Ανίχνευση μέχρι 4 σφαλμάτων και
  - Διόρθωση μέχρι 2 σφαλμάτων.

- Κανόνας: Για τη διόρθωση σφαλμάτων ενός bit επιλέγεται το πλήθος *r* των bit ελέγχου, έτσι ώστε να ισχύει $\mathbf{(m + r + 1) \leq 2^r}$ ή $\mathbf{(n + 1)2^m \leq 2^n}$.

Κωδικός Hamming για την διόρθωση σφαλμάτων (I)

Στην πλευρά του αποστολέα:

1. Τα bits της κωδικολέξης αριθμούνται διαδοχικά, αρχίζοντας με το bit 1 στο αριστερότερο άκρο, το bit 2 ακριβώς δεξιά του, κ.ο.κ.
2. Τα bits που είναι δυνάμεις του 2 (1, 2, 4, 8, 16, κ.λπ.) είναι bits ελέγχου. Τα υπόλοιπα (3, 5, 6, 7, κ.λπ.) συμπληρώνονται από τα m bits δεδομένων.
3. Κάθε bit ελέγχου στη θέση 2^i εξαναγκάζει την ισοτιμία των εξής bits να είναι άρτια (ή περιττή):
   - του ιδίου του bit ελέγχου και
   - όλων των bits δεδομένων ο αριθμός της θέσης των οποίων περιέχει τον όρο 2^i, όταν αναλυθεί σε άθροισμα δυνάμεων του 2. Π.χ., το bit ελέγχου στη θέση 4 (2^2), ελέγχει, εκτός από τον εαυτό του, τα bits: 5 = 1 + 4, 6 = 2 + 4, 7 = 1 + 2 + 4, 12 = 4 + 8, 13 = 1 + 4 +8, 14 = 2 + 4 + 8, ...

Κάθε bit δεδομένων ελέγχεται από αυτά και μόνον αυτά τα bits ελέγχου τα οποία εμφανίζονται στην ανάλυσή του σε δυνάμεις του 2.

Κωδικός Hamming για την διόρθωση σφαλμάτων (II)

Στην πλευρά του παραλήπτη:

1. Αρχικοποιείται ένας μετρητής με μηδέν.
2. Κάθε bit ελέγχου στη θέση $k = 2^i$ (= 1, 2, 4, …) και τα bits δεδομένων τα οποία ελέγχει, εξετάζονται για να επιβεβαιωθεί εάν το bit ισοτιμίας είναι σωστό.
3. Εάν όχι, προστίθεται στο μετρητή ο αριθμός k.
4. Εάν ο μετρητής είναι ίσος με μηδέν στο τέλος του ελέγχου, η κωδικολέξη γίνεται αποδεκτή ως έγκυρη. Ειδάλλως, περιέχει τον αριθμό θέσης του λανθασμένου bit.

Παράδειγμα. Εάν τα bits ισοτιμίας που ευρίσκονταν εσφαλμένα ήταν τα bits ελέγχου 1, 2, 4, τότε η τιμή του μετρητή θα ήταν ίση με: 1 + 2 + 4 = 7 και θα μπορούσε να συμπεράνει ο παραλήπτης ότι κατά τη μετάδοση αντεστράφη το bit δεδομένων 7, καθώς το bit 7 είναι το μόνο το οποίο ελέγχεται από τα bits 1, 2, και 4.

Παράδειγμα εφαρμογής κώδικα Hamming διόρθωσης σφαλμάτων

			1,2		1,4	2,4	1,2,4		1,8	2,8	1,2,8	4,8	← Ελέγχετε από αυτά τα bits ελέγχου
Word	**bit_1**	**bit_2**	**bit_3**	**bit_4**	**bit_5**	**bit_6**	**bit_7**	**bit_8**	**bit_9**	**bit_10**	**bit_11**	**bit_12**	
11000100			1		1	0	0		0	1	0	0	

controlled by bits	3	3	5	9
	5	6	6	10
	7	7	7	11
	9	10	12	12
	11	11		

Παράδειγμα εφαρμογής κώδικα Hamming διόρθωσης σφαλμάτων

			1,2		1,4	2,4	1,2,4		1,8	2,8	1,2,8	4,8	← Ελέγχετε από αυτά τα bits ελέγχου
Word	**bit_1**	**bit_2**	**bit_3**	**bit_4**	**bit_5**	**bit_6**	**bit_7**	**bit_8**	**bit_9**	**bit_10**	**bit_11**	**bit_12**	**Codeword**
11000100			1		1	0	0		0	1	0	0	001110010100

controlled by bits				
	3	3	5	9
	5	6	6	10
	7	7	7	11
	9	10	12	12
	11	11		

Παράδειγμα εφαρμογής κώδικα Hamming διόρθωσης σφαλμάτων

			1,2		1,4	2,4	1,2,4		1,8	2,8	1,2,8	4,8	← Ελέγχετε από αυτά τα bits ελέγχου
Word	**bit_1**	**bit_2**	**bit_3**	**bit_4**	**bit_5**	**bit_6**	**bit_7**	**bit_8**	**bit_9**	**bit_10**	**bit_11**	**bit_12**	**Codeword**
11000100			1		1	0	0		0	1	0	0	001110010100
11000101			1		1	0	0		0	1	0	1	

controlled by bits	3	3	5	9
	5	6	6	10
	7	7	7	11
	9	10	12	12
	11	11		

Παράδειγμα εφαρμογής κώδικα Hamming διόρθωσης σφαλμάτων

			1,2		1,4	2,4	1,2,4		1,8	2,8	1,2,8	4,8	← Ελέγχετε από αυτά τα bits ελέγχου
Word	**bit_1**	**bit_2**	**bit_3**	**bit_4**	**bit_5**	**bit_6**	**bit_7**	**bit_8**	**bit_9**	**bit_10**	**bit_11**	**bit_12**	**Codeword**
11000100			1		1	0	0		0	1	0	0	001110010100
11000101			1		1	0	0		0	1	0	1	001010000101

controlled by bits				
	3	3	5	9
	5	6	6	10
	7	7	7	11
	9	10	12	12
	11	11		

Παράδειγμα εφαρμογής κώδικα Hamming διόρθωσης σφαλμάτων

			1,2		1,4	2,4	1,2,4		1,8	2,8	1,2,8	4,8	← Ελέγχετε από αυτά τα bits ελέγχου
Word	**bit_1**	**bit_2**	**bit_3**	**bit_4**	**bit_5**	**bit_6**	**bit_7**	**bit_8**	**bit_9**	**bit_10**	**bit_11**	**bit_12**	**Codeword**
11000100			1		1	0	0		0	1	0	0	001110010100
11000101			1		1	0	0		0	1	0	1	001010000101
			0		1	0	0		0	1	0	0	000110010100

controlled by bits				
	3	3	5	9
	5	6	6	10
	7	7	7	11
	9	10	12	12
	11	11		

Παράδειγμα εφαρμογής κώδικα Hamming διόρθωσης σφαλμάτων

			1,2		1,4	2,4	1,2,4		1,8	2,8	1,2,8	4,8	← Ελέγχετε από αυτά τα bits ελέγχου
Word	**bit_1**	**bit_2**	**bit_3**	**bit_4**	**bit_5**	**bit_6**	**bit_7**	**bit_8**	**bit_9**	**bit_10**	**bit_11**	**bit_12**	**Codeword**
11000100			1		1	0	0		0	1	0	0	001110010100
11000101			1		1	0	0		0	1	0	1	001010000101
11000100			0		1	0	0		0	1	0	0	0010110010100

controlled by bits				
	3	3	5	9
	5	6	6	10
	7	7	7	11
	9	10	12	12
	11	11		

Παράδειγμα εφαρμογής κώδικα Hamming διόρθωσης σφαλμάτων

			1,2		1,4	2,4	1,2,4		1,8	2,8	1,2,8	4,8	← Ελέγχετε από αυτά τα bits ελέγχου
Word	**bit_1**	**bit_2**	**bit_3**	**bit_4**	**bit_5**	**bit_6**	**bit_7**	**bit_8**	**bit_9**	**bit_10**	**bit_11**	**bit_12**	**Codeword**
11000100			1		1	0	0		0	1	0	0	001110010100
11000101			1		1	0	0		0	1	0	1	001010000101
11000100			0		1	0	0		0	1	0	0	00<u>0</u>110010100
			1		1	0	0		0	1	0	0	00111000<u>0</u>0100

controlled by bits				
	3	3	5	9
	5	6	6	10
	7	7	7	11
	9	10	12	12
	11	11		

Παράδειγμα εφαρμογής κώδικα Hamming διόρθωσης σφαλμάτων

			1,2		1,4	2,4	1,2,4		1,8	2,8	1,2,8	4,8	← Ελέγχετε από αυτά τα bits ελέγχου
Word	**bit_1**	**bit_2**	**bit_3**	**bit_4**	**bit_5**	**bit_6**	**bit_7**	**bit_8**	**bit_9**	**bit_10**	**bit_11**	**bit_12**	**Codeword**
11000100			1		1	0	0		0	1	0	0	001110010100
11000101			1		1	0	0		0	1	0	1	001010000101
11000100			0		1	0	0		0	1	0	0	00**0**110010100
11000100			1		1	0	0		0	1	0	0	00111000**0**100

controlled by bits				
	3	3	5	9
	5	6	6	10
	7	7	7	11
	9	10	12	12
	11	11		

Παράδειγμα εφαρμογής κώδικα Hamming διόρθωσης σφαλμάτων

			1,2		1,4	2,4	1,2,4		1,8	2,8	1,2,8	4,8	← Ελέγχετε από αυτά τα bits ελέγχου
Word	**bit_1**	**bit_2**	**bit_3**	**bit_4**	**bit_5**	**bit_6**	**bit_7**	**bit_8**	**bit_9**	**bit_10**	**bit_11**	**bit_12**	**Codeword**
11000100			1		1	0	0		0	1	0	0	001110010100
11000101			1		1	0	0		0	1	0	1	001010000101
11000100			0		1	0	0		0	1	0	0	00<u>0</u>110010100
11000100			1		1	0	0		0	1	0	0	00111000<u>0</u>0100
10101000			1		0	1	0		1	0	0	0	

controlled by bits				
	3	3	5	9
	5	6	6	10
	7	7	7	11
	9	10	12	12
	11	11		

Παράδειγμα εφαρμογής κώδικα Hamming διόρθωσης σφαλμάτων

			1,2		1,4	2,4	1,2,4		1,8	2,8	1,2,8	4,8	← Ελέγχετε από αυτά τα bits ελέγχου
Word	**bit_1**	**bit_2**	**bit_3**	**bit_4**	**bit_5**	**bit_6**	**bit_7**	**bit_8**	**bit_9**	**bit_10**	**bit_11**	**bit_12**	**Codeword**
11000100			1		1	0	0		0	1	0	0	001110010100
11000101			1		1	0	0		0	1	0	1	001010000101
11000100			0		1	0	0		0	1	0	0	00<u>0</u>110010100
11000100			1		1	0	0		0	1	0	0	00111000<u>0</u>0100
10101000			1		0	1	0		1	0	0	0	001101011000

controlled by bits				
	3	3	5	9
	5	6	6	10
	7	7	7	11
	9	10	12	12
	11	11		

Παράδειγμα εφαρμογής κώδικα Hamming διόρθωσης σφαλμάτων

			1,2		1,4	2,4	1,2,4		1,8	2,8	1,2,8	4,8	← Ελέγχετε από αυτά τα bits ελέγχου
Word	**bit_1**	**bit_2**	**bit_3**	**bit_4**	**bit_5**	**bit_6**	**bit_7**	**bit_8**	**bit_9**	**bit_10**	**bit_11**	**bit_12**	**Codeword**
11000100			1		1	0	0		0	1	0	0	001110010100
11000101			1		1	0	0		0	1	0	1	001010000101
11000100			0		1	0	0		0	1	0	0	00<u>0</u>110010100
11000100			1		1	0	0		0	1	0	0	00111000<u>0</u>0100
10101000			1		0	1	0		1	0	0	0	001101011000
10101001			1		0	1	0		1	0	0	1	

controlled by bits	3	3		5				9					
	5	6		6				10					
	7	7		7				11					
	9	10		12				12					
	11	11											

Παράδειγμα εφαρμογής κώδικα Hamming διόρθωσης σφαλμάτων

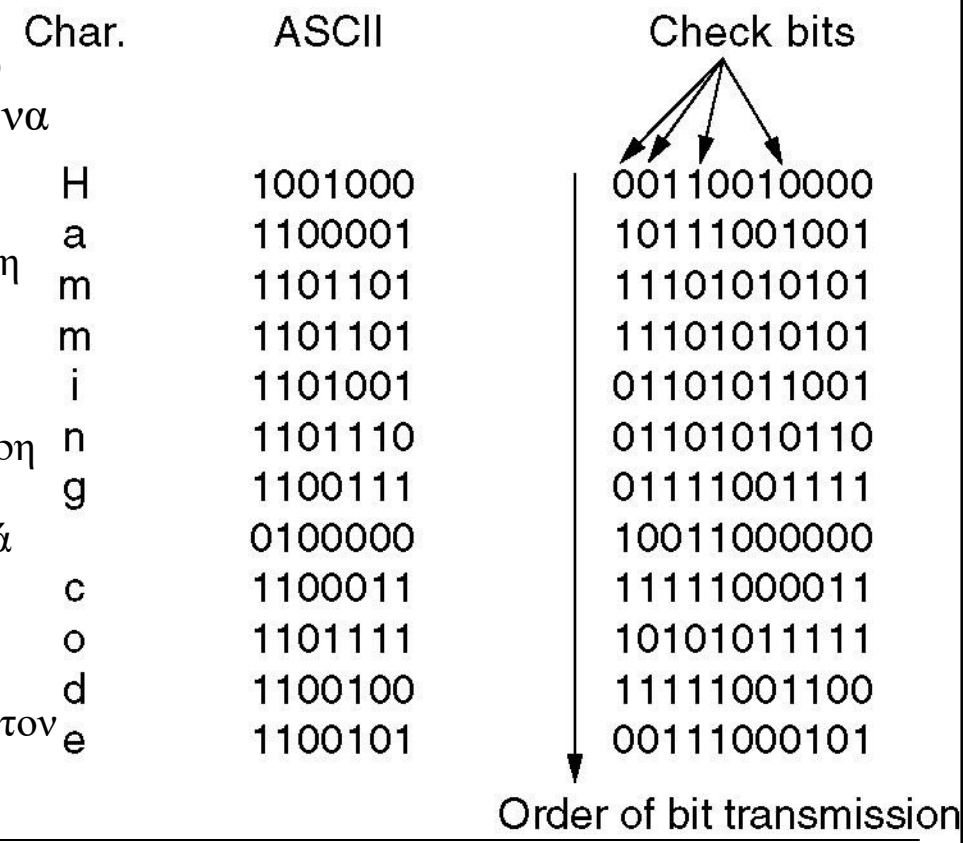
			1,2		1,4	2,4	1,2,4		1,8	2,8	1,2,8	4,8	← Ελέγχετε από αυτά τα bits ελέγχου
Word	**bit_1**	**bit_2**	**bit_3**	**bit_4**	**bit_5**	**bit_6**	**bit_7**	**bit_8**	**bit_9**	**bit_10**	**bit_11**	**bit_12**	**Codeword**
11000100			1		1	0	0		0	1	0	0	001110010100
11000101			1		1	0	0		0	1	0	1	001010000101
11000100			0		1	0	0		0	1	0	0	00<u>0</u>110010100
11000100			1		1	0	0		0	1	0	0	00111000<u>0</u>0100
10101000			1		0	1	0		1	0	0	0	001101011000
10101001			1		0	1	0		1	0	0	1	001001001001
controlled by bits	3	3		5				9					
	5	6		6				10					
	7	7		7				11					
	9	10		12				12					
	11	11											

Χρήση του κωδικού Hamming για διόρθωση ριπών σφαλμάτων

Στο ακόλουθο σχήμα φαίνονται κάποιοι 7μπιτοι χαρακτήρες ASCII κωδικοποιημένοι ως 11μπιτες κωδικολέξεις μέσω ενός κωδικού Hamming. Τα bits δεδομένων βρίσκονται στις θέσεις 3, 5, 6, 7, 9, 10 και 11.

Οι κωδικοί Hamming μπορούν να διορθώνουν μόνον σφάλματα του ενός bit. Με την παρακάτω όμως τεχνική, μπορούν να χρησιμοποιηθούν για να διορθώσουν και ριπές σφαλμάτων:

1. Μια ακολουθία *k* διαδοχικών κωδικολέξεων τοποθετείται σε μορφή πίνακα, με μία κωδικολέξη ανά γραμμή.
2. Τα δεδομένα δεν μεταδίδονται ανά κωδικολέξη, αλλά ανά στήλη, ξεκινώντας από αριστερά.
3. Όταν μεταδοθούν και τα *k* bits στέλνεται η δεύτερη στήλη, κ.ο.κ.
4. Τέλος ο πίνακας ανακατασκευάζεται στην πλευρά του παραλήπτη.
5. Αν παρουσιαστεί ριπή σφαλμάτων μήκους *k*, θα επηρεάσει το πολύ 1 bit σε κάθε μια από τις *k* κωδικολέξεις, το οποίο μπορεί να διορθωθεί από τον κωδικό Hamming. Έτσι μπορεί να ανακτηθεί ολόκληρη η ομάδα των δεδομένων.



Κωδικοί ανίχνευσης σφαλμάτων (Ι)

- Θεωρούμε ένα κανάλι με ρυθμό σφαλμάτων 10^{-6} ανά bit και ομάδες δεδομένων με μέγεθος 1000 bits.
  - Για να παρέχεται διόρθωση μεμονωμένων σφαλμάτων (π.χ., με κωδικό Hamming), απαιτούνται 10 bits ελέγχου ανά ομάδα. Για τη μετάδοση 1 Mbit δεδομένων (ήτοι, 1000 ομάδων) απαιτούνται λοιπόν 10.000 bits ελέγχου.
  - Για να ανιχνευτεί όμως απλώς μια ομάδα με ένα σφάλμα του 1 bit, αρκεί ένα bit ισοτιμίας ανά ομάδα. Κάθε 1000 ομάδες, πρέπει λοιπόν να αναμεταδίδεται μια πρόσθετη ομάδα των 1001 bit, με συνολική επιβάρυνση στην αναμετάδοση του 1 Mbit δεδομένων μόνο 2001 (= 1000 + 1001) bits για τον έλεγχο και την αναμετάδοση της εσφαλμένης ομάδας.
- **Στην περίπτωση αυτή συμφέρει η ανίχνευση σφαλμάτων.**

Κωδικοί ανίχνευσης σφαλμάτων (ΙΙ)

- Οι κωδικοί **διόρθωσης σφαλμάτων** χρησιμοποιούνται ευρέως στις ασύρματες γραμμές, που παρουσιάζουν πολύ θόρυβο και ευαισθησία στα σφάλματα, συγκριτικά με τα χάλκινα καλώδια και τις οπτικές ίνες.
- Η **ανίχνευση σφαλμάτων** και οι αναμεταδόσεις αποτελούν αποδοτικότερη λύση, όταν ο ρυθμός των σφαλμάτων είναι χαμηλός (π.χ., χάλκινα καλώδια, οπτικές ίνες, ...).
- Απλή περίπτωση κώδικα ανίχνευσης ενός σφάλματος αποτελεί η προσθήκη ενός bit (bit ισοτιμίας), τέτοιο ώστε ο συνολικός αριθμός των bits ίσων με 1 στο πλαίσιο να είναι περιττός ή άρτιος αριθμός (**περιττή** ή **άρτια ισοτιμία** – odd or even parity).
- Στην πράξη, για την ανίχνευση σφαλμάτων χρησιμοποιείται ο **Κυκλικός Έλεγχος Πλεονασμού** ή **CRC** (**Cyclic Redundancy Check**).

Κυκλικός Έλεγχος Πλεονασμού ή CRC (Cyclic Redundancy Check) (I)

Ο **Κυκλικός Έλεγχος Πλεονασμού** ή **CRC** αποτελεί πολυωνυμικό κωδικό που βασίζεται στη θεώρηση των ακολουθιών bit ως αναπαραστάσεις πολυωνύμων με συντελεστές μόνο 0 και 1.

- Κάθε πλαίσιο με *k* bit αντιμετωπίζεται ως λίστα συντελεστών πολυωνύμου βαθμού *k-1* με *k* όρους, από x^{k-1} έως x^0, με το πιο σημαντικό (αριστερότερο) bit να είναι ο συντελεστής του x^{k-1} (π.χ., το 110001 αντιστοιχεί στο πολυώνυμο $x^5 + x^4 + x^0$).
- Οι αριθμητικές πράξεις αυτών των πολυωνύμων εκτελούνται με βάση το υπόλοιπο ως προς 2 (modulo 2). Πρόσθεση και αφαίρεση είναι αμφότερες ταυτόσημες με την αποκλειστική διάζευξη ή XOR (EXCLUSIVE OR). Για παράδειγμα:

```
  10011011          01010101
+ 11001010        - 10101111
----------        ----------
  01010001          11111010
```

Κυκλικός Έλεγχος Πλεονασμού ή CRC (Cyclic Redundancy Check) (II)

- Βασική ιδέα της μεθόδου ανίχνευσης σφαλμάτων CRC:
  1. Ο αποστολέας προσαρτά ένα **άθροισμα ελέγχου** (**checksum**) στο τέλος του πλαισίου, έτσι ώστε το πολυώνυμο που παριστάνεται από το πλαίσιο μαζί με το άθροισμα ελέγχου να διαιρείται ακριβώς από το προσυμφωνημένο **παράγον πολυώνυμο** (**generator polynomial**) *G(x)*.
  2. Όταν ο παραλήπτης λάβει το πλαίσιο με το άθροισμα ελέγχου, δοκιμάζει να το διαιρέσει με το *G(x)*. Εάν υπάρξει υπόλοιπο, έχει συμβεί κάποιο σφάλμα μετάδοσης.

Αλγόριθμος υπολογισμού του αθροίσματος ελέγχου στη μέθοδο CRC

Ο αλγόριθμος υπολογισμού του αθροίσματος ελέγχου στη μέθοδο CRC είναι ο ακόλουθος:

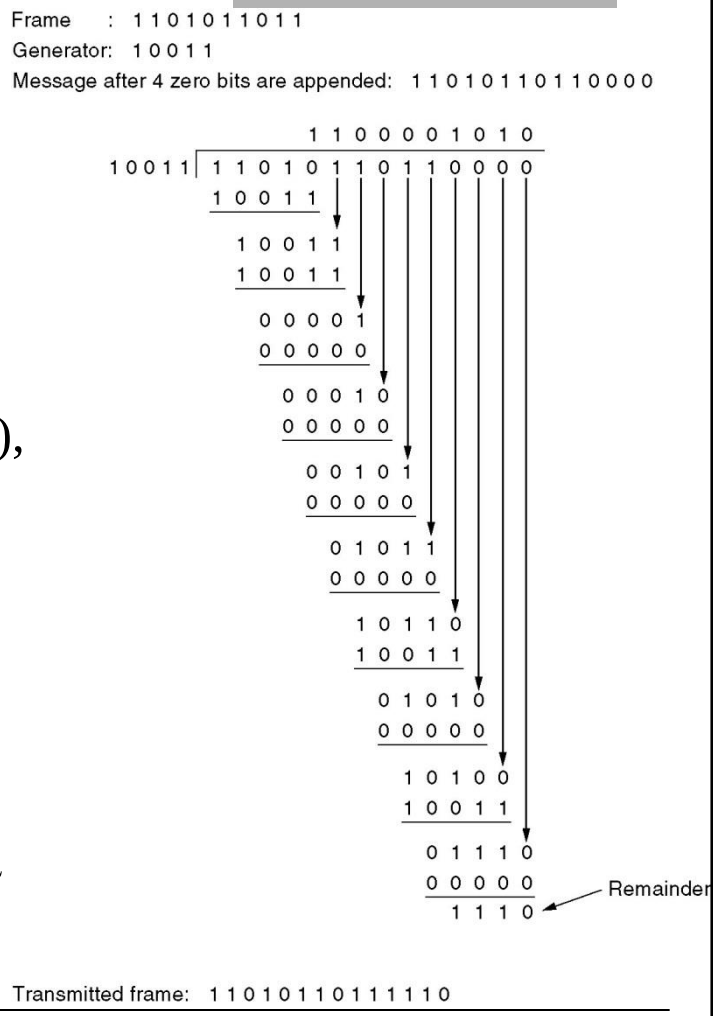
1. Έστω r ο βαθμός του $G(x)$. Προσαρτούμε r μηδενικά bit στο λιγότερο σημαντικό άκρο του πλαισίου, έτσι ώστε να περιέχει πια $m + r$ bit και να αντιστοιχεί στο πολυώνυμο $x^r M(x)$.
2. Διαιρούμε την ακολουθία bit που αντιστοιχεί στο $x^r M(x)$ με την ακολουθία bit που αντιστοιχεί στο $G(x)$, χρησιμοποιώντας διαίρεση με υπόλοιπο ως προς 2.
3. Αφαιρούμε το υπόλοιπο (το οποίο έχει πάντα r ή λιγότερα bit) από την ακολουθία bit που αντιστοιχεί στο $x^r M(x)$, χρησιμοποιώντας αφαίρεση με υπόλοιπο ως προς 2. Το αποτέλεσμα είναι το προς μετάδοση πλαίσιο μαζί με το άθροισμα ελέγχου.

Αλγόριθμος υπολογισμού του αθροίσματος ελέγχου στη μέθοδο CRC

Ο αλγόριθμος υπολογισμού του αθροίσματος ελέγχου στη μέθοδο CRC είναι ο ακόλουθος:

1. Έστω r ο βαθμός του $G(x)$. Προσαρτούμε r μηδενικά bit στο λιγότερο σημαντικό άκρο του πλαισίου, έτσι ώστε να περιέχει πια $m + r$ bit και να αντιστοιχεί στο πολυώνυμο $x^r M(x)$.
2. Διαιρούμε την ακολουθία bit που αντιστοιχεί στο $x^r M(x)$ με την ακολουθία bit που αντιστοιχεί στο $G(x)$, χρησιμοποιώντας διαίρεση με υπόλοιπο ως προς 2.
3. Αφαιρούμε το υπόλοιπο (το οποίο έχει πάντα r ή λιγότερα bit) από την ακολουθία bit που αντιστοιχεί στο $x^r M(x)$, χρησιμοποιώντας αφαίρεση με υπόλοιπο ως προς 2. Το αποτέλεσμα είναι το προς μετάδοση πλαίσιο μαζί με το άθροισμα ελέγχου.

Στο διπλανό σχήμα φαίνεται ο παραπάνω υπολογισμός για το πλαίσιο 1101011011, με παράγον πολυώνυμο το $G(x) = x^4 + x + 1$.



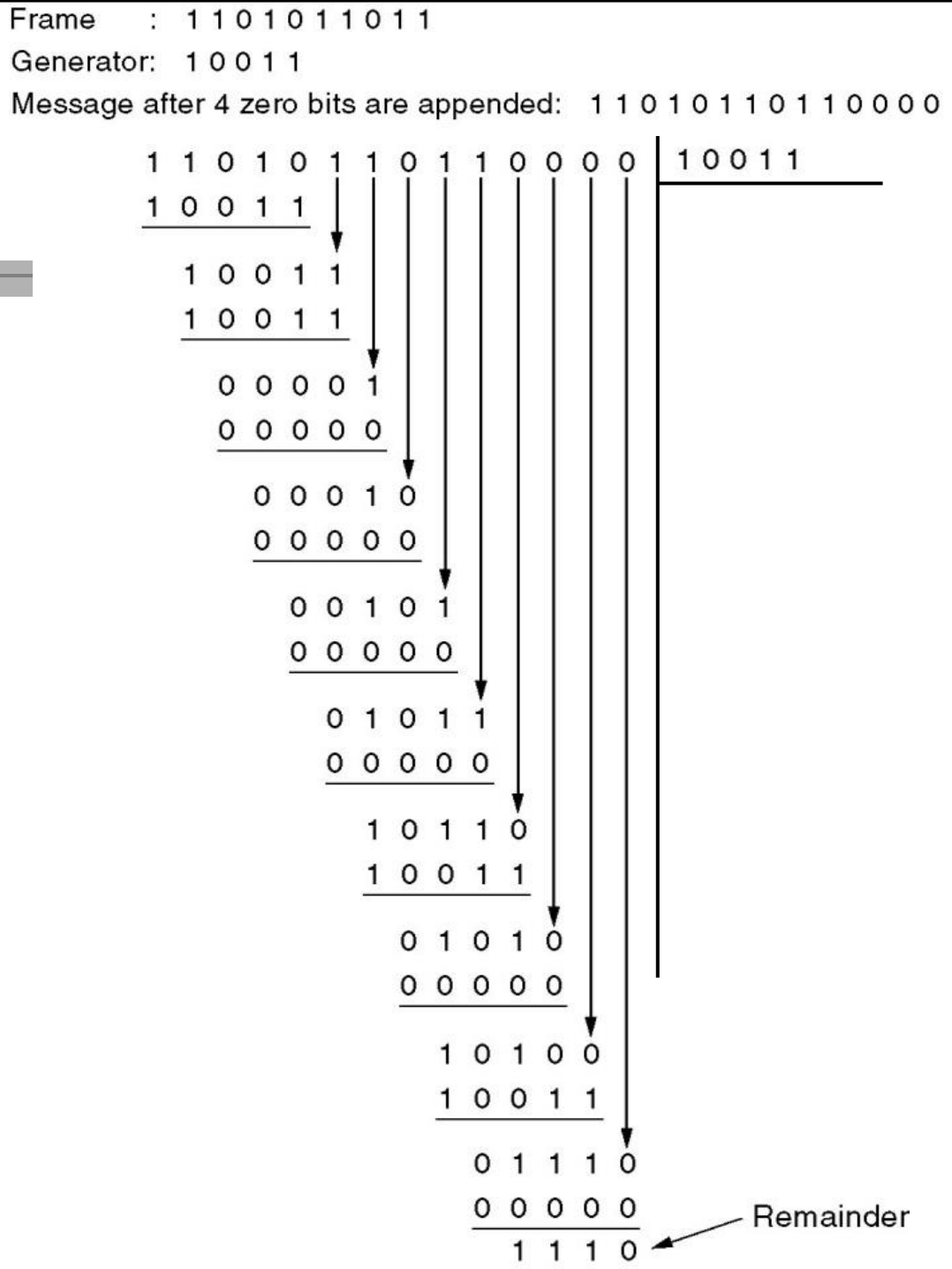
Παράδειγμα εφαρμογής CRC

Αρχικό πλαίσιο (χωρίς έλεγχο σφαλμάτων):
1 1 0 1 0 1 1 0 1 1

Παράγον πολυώνυμο:
$G(x) = x^4 + x + 1 \Leftrightarrow$
1 0 0 1 1

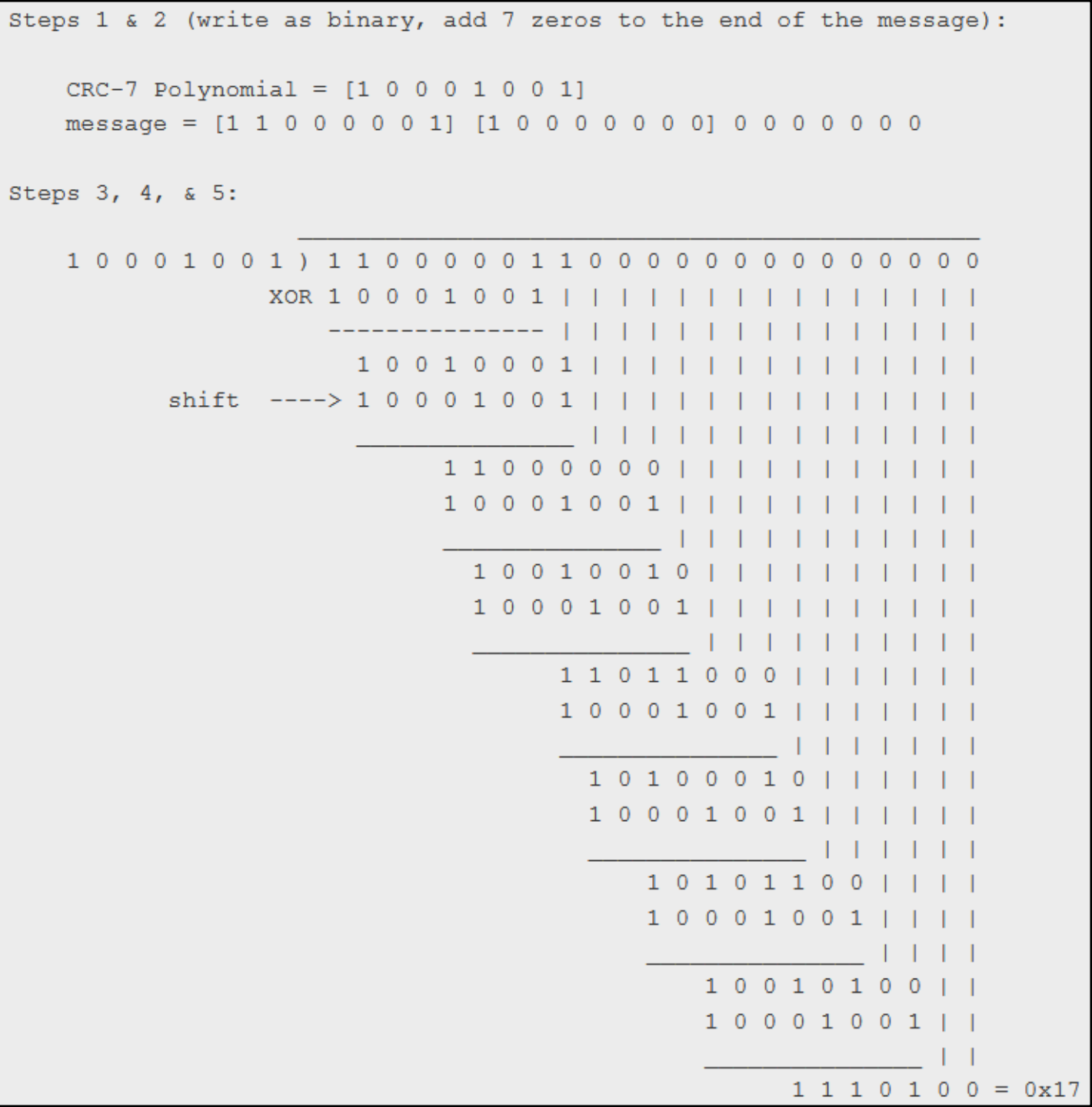
Υπόλοιπο διαίρεσης:
1 1 1 0

Πλαίσιο προς αποστολή:
1 1 0 1 0 1 1 0 1 1 1 1 1 0



Προσοχή:

Κάνω shift μέχρι να βρω μονάδα (όταν στο υπόλοιπο προκύψει μηδενικό στο/α πρώτο/α ψηφίο/α)



Σφάλματα ανιχνεύσιμα με την CRC

Τα σφάλματα που είναι δυνατόν να ανιχνευτούν με τη μέθοδο CRC έχουν ως εξής:

1. Όλα τα μεμονωμένα σφάλματα είναι ανιχνεύσιμα εφόσον το παράγον πολυώνυμο $G(x)$ περιέχει δύο ή παραπάνω όρους.
2. Όλα τα διπλά σφάλματα είναι ανιχνεύσιμα εάν το $G(x)$ δε διαιρεί το $x^k + 1$ για κάθε k μέχρι το μέγιστο μέγεθος του πλαισίου.
3. Κάθε πλαίσιο με περιττό αριθμό εσφαλμένων bit μπορεί να ανιχνευτεί εάν το $G(x)$ περιέχει τον όρο $x + 1$ (αφού κανένα πολυώνυμο με περιττό πλήθος όρων δε διαιρείται με το $x + 1$).
4. Κάθε CRC με r bit ελέγχου (ο βαθμός του $G(x)$) μπορεί να ανιχνεύσει όλες τις ριπές σφαλμάτων με μήκος $\leq r$.
5. Η πιθανότητα ριπή σφαλμάτων με μήκος length $r + 1$ να γίνει αποδεκτή ως έγκυρη είναι $(½)^{r-1}$.
6. Η πιθανότητα να περάσει απαρατήρητο λανθασμένο πλαίσιο, μετά από ριπή σφαλμάτων μεγαλύτερη από $r + 1$ ή μετά από πολλαπλές μικρότερες ριπές, είναι ίση με $(½)^r$.

Μερικά πολυώνυμα έχουν γίνει διεθνή πρότυπα, όπως το ακόλουθο πολυώνυμο της IEEE 802: $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x^1 + 1$. Τώρα, ένα απλό κύκλωμα καταχωρητή ολίσθησης χρησιμοποιείται για τον υπολογισμό και την επαλήθευση των αθροισμάτων ελέγχου.

Απαιτήσεις από τα πρωτόκολλα αναμετάδοσης

Οι απαιτήσεις από τα πρωτόκολλα διακρίνονται σε απαιτήσεις επίδοσης και λειτουργικές απαιτήσεις.

Οι **απαιτήσεις επίδοσης** είναι δυνατόν να μετρηθούν βάσει των ακολούθων δεικτών επίδοσης:

- **Διαπερατότητα**: Ποσοστό του χρόνου που αξιοποιείται το κανάλι.
- **Καθυστέρηση**: Μέσος συνολικός χρόνος παράδοσης ενός πακέτου.
- **Ουρά**: Αριθμός πακέτων που συσσωρεύονται προς μετάδοση.
- **Τελικός ρυθμός σφαλμάτων**: Τα σφάλματα που δεν διορθώνονται.

Οι απαιτήσεις επίδοσης αξιολογούνται με εργαλεία όπως η προσομοίωση και η θεωρία αναμονής.

Οι **λειτουργικές απαιτήσεις** των πρωτοκόλλων περιλαμβάνουν τα ακόλουθα:

- **Ορθότητα**
- **Αντοχή**

Οι λειτουργικές απαιτήσεις επαληθεύονται με εργαλεία όπως τα αυτόματα, οι λογισμοί διεργασιών, οι αλγόριθμοι & γραφοθεωρία, κ.λπ.

Επαλήθευση πρωτοκόλλων

Τα ρεαλιστικά πρωτόκολλα καθώς και τα προγράμματα που τα υλοποιούν είναι συχνά εξαιρετικά πολύπλοκα. Έτσι έχει γίνει πολλή έρευνα για την ανεύρεση μαθηματικών τεχνικών για τον καθορισμό και την επαλήθευση των πρωτοκόλλων. Οι μέθοδοι που θα εξεταστούν ακολούθως, δεν ισχύουν μόνο για το επίπεδο ζεύξης δεδομένων, αλλά εφαρμόζονται και στα υπόλοιπα επίπεδα.

Στα πλαίσια αυτά θα εξεταστούν τα ακόλουθα:

- **Μοντέλα μηχανών πεπερασμένης κατάστασης** (Finite State Machine Models)
- **Μοντέλα δικτύων Petri** (Petri Net Models)

Ένα μονόδρομο πρωτόκολλο για θορυβώδη κανάλια (Ι)

Το πρωτόκολλο **Θετικής Επιβεβαίωσης με Αναμετάδοση** ή **PAR (Positive Acknowledgement with Retransmission)** που φαίνεται ακολούθως είναι κατάλληλο για θορυβώδη κανάλια.

Διαφέρει από το ***Stop and Wait*** ως προς το ότι τόσο ο αποστολέας όσο και ο παραλήπτης έχουν μια μεταβλητή (sequence number), η τιμή της οποίας δε χάνεται όσο το επίπεδο ζεύξης δεδομένων βρίσκεται σε κατάσταση αναμονής. Έτσι ο αποστολέας κρατάει τον αριθμό ακολουθίας του επόμενου πλαισίου που θα στείλει, ενώ ο παραλήπτης κρατά τον αριθμό ακολουθίας του επόμενου πλαισίου που αναμένεται. Με τον τρόπο αυτό, ανιχνεύονται τα διπλά (duplicated) και τα απωλεσθέντα πλαίσια.

Δομή πλαισίου

```
typedef struct {
  frame_kind kind;
  seq_nr seq;
  seq_nr ack;
  packet info;
} frame;
```

```
/* frames are transported in this layer */
/* what kind of a frame is it? */
/* sequence number */
/* acknowledgement number */
/* the network layer packet */
```

Ένα μονό-δρομο πρωτόκολλο για θορυβώδη κανάλια (II)

```
/* Protocol 3 (par) allows unidirectional data flow over an unreliable channel. */

#define MAX_SEQ 1                                   /* must be 1 for protocol 3 */
typedef enum  {frame_arrival, cksum_err, timeout} event_type;
#include "protocol.h"

void sender3(void)
{
  seq_nr next_frame_to_send;                        /* seq number of next outgoing frame */
  frame s;                                          /* scratch variable */
  packet buffer;                                    /* buffer for an outbound packet */
  event_type event;

  next_frame_to_send = 0;                           /* initialize outbound sequence numbers */
  from_network_layer(&buffer);                      /* fetch first packet */
  while (true) {
      s.info = buffer;                              /* construct a frame for transmission */
      s.seq = next_frame_to_send;                   /* insert sequence number in frame */
      to_physical_layer(&s);                        /* send it on its way */
      start_timer(s.seq);                           /* if answer takes too long, time out */
      wait_for_event(&event);                       /* frame_arrival, cksum_err, timeout */
      if (event == frame_arrival) {
          from_physical_layer(&s);                  /* get the acknowledgement */
          if (s.ack == next_frame_to_send) {
              stop_timer(s.ack);                    /* turn the timer off */
              from_network_layer(&buffer);          /* get the next one to send */
              inc(next_frame_to_send);              /* invert next_frame_to_send */
          }
      }
  }
}
```

Συνεχίζεται →

Ένα μονόδρομο πρωτόκολλο για θορυβώδη κανάλια (III)

```
void receiver3(void)
{
  seq_nr frame_expected;
  frame r, s;
  event_type event;

  frame_expected = 0;
  while (true) {
       wait_for_event(&event);                  /* possibilities: frame_arrival, cksum_err */
       if (event == frame_arrival) {            /* a valid frame has arrived. */
            from_physical_layer(&r);            /* go get the newly arrived frame */
            if (r.seq == frame  expected) {     /* this is what we have been waiting for. */
                 to_network_layer(&r.info);     /* pass the data to the network layer */
                 inc(frame_expected);           /* next time expect the other sequence nr */
            }
            s.ack = 1 – frame_expected;         /* tell which frame is being acked */
            to_physical_layer(&s);              /* send acknowledgement */
       }
  }
}
```

Πρωτόκολλο θετικής επιβεβαίωσης με αναμετάδοση.

Μοντέλα μηχανών πεπερασμένης κατάστασης (I)

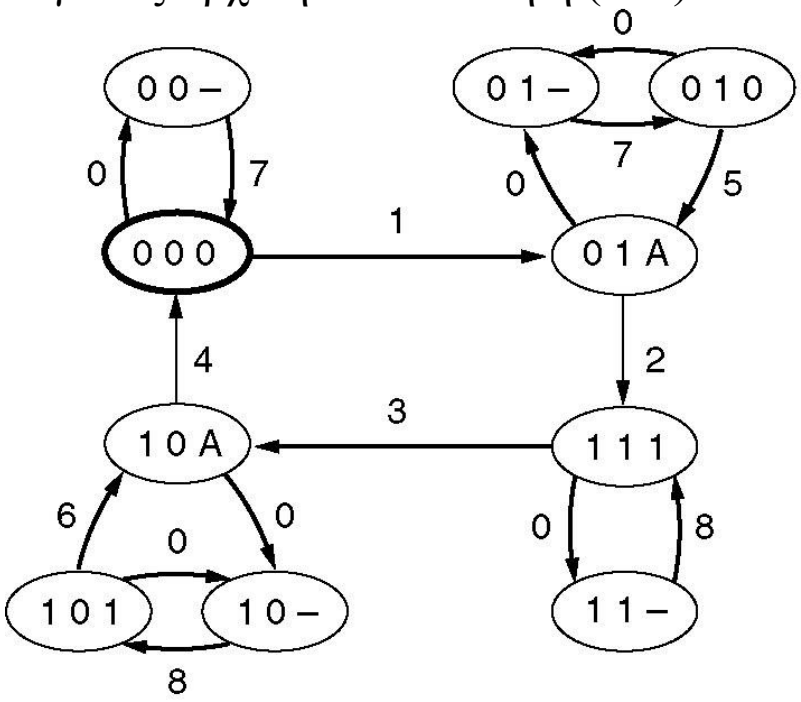
Η **μηχανή πεπερασμένης κατάστασης** αποτελεί τεχνική για την επαλήθευση και τη μαθηματική περιγραφή πρωτοκόλλων. Εδώ, κάθε **μηχανή πρωτοκόλλου** (ήτοι, ο αποστολέας ή ο παραλήπτης) βρίσκεται σε μια συγκεκριμένη **κατάσταση** κάθε χρονική στιγμή. Για παράδειγμα, στο πρωτόκολλο θετικής επιβεβαίωσης με αναμετάδοση (2 προηγούμενες διαφάνειες): ο αποστολέας μπορεί να προσπαθεί να στείλει ένα πλαίσιο 0 ή 1 (κατάσταση 0 ή 1 του αποστολέα), ο παραλήπτης μπορεί να αναμένει ένα πλαίσιο 0 ή 1 (κατάσταση 0 ή 1 του παραλήπτη), ενώ το κανάλι έχει τέσσερεις πιθανές καταστάσεις, ήτοι, να μεταφέρει ένα πλαίσιο 0 ή ένα πλαίσιο 1 από τον αποστολέα, να μεταφέρει πλαίσιο επιβεβαίωσης από τον παραλήπτη, ή να είναι κενό (καταστάσεις 0 ή 1 ή Α ή -). Άρα το σύστημα έχει συνολικά 16 πιθανές καταστάσεις. Από κάθε κατάσταση υπάρχουν μηδέν ή περισσότερες **μεταβάσεις** σε άλλες καταστάσεις. Μια συγκεκριμένη κατάσταση ορίζεται ως **αρχική κατάσταση** (το σημείο από όπου ξεκινά η λειτουργία του συστήματος).

Σε μαθηματικούς όρους, το μοντέλο μηχανής πεπερασμένης κατάστασης ενός πρωτοκόλλου μπορεί να θεωρηθεί ως μια τετράδα (*S*, *M*, *I*, *T*):

- ***S***: είναι το σύνολο των καταστάσεων στις οποίες μπορεί να βρίσκονται οι διεργασίες και το κανάλι.
- ***M***: είναι το σύνολο των πλαισίων που μπορούν να ανταλλαγούν μέσω του καναλιού.
- ***I***: είναι το σύνολο των αρχικών καταστάσεων των διεργασιών.
- ***T***: είναι το σύνολο των μεταβάσεων μεταξύ των καταστάσεων.

Μοντέλα μηχανών πεπερασμένης κατάστασης (II)

Κάθε κατάσταση χαρακτηρίζεται από τρεις χαρακτήρες, *SRC*, όπου το S (0 ή 1) αντιστοιχεί στο πλαίσιο που προσπαθεί να στείλει ο αποστολέας, το R (0 ή 1) αντιστοιχεί στο πλαίσιο που αναμένει ο παραλήπτης και το C (0 ή 1 ή A ή -) αντιστοιχεί στην κατάσταση του καναλιού. Εδώ, έχει επιλεγεί ως αρχική κατάσταση η (000).



(a)

Transition	Who runs?	Frame accepted	Frame emitted	To network layer
0	–	(frame lost)		–
1	R	0	A	Yes
2	S	A	1	–
3	R	1	A	Yes
4	S	A	0	–
5	R	0	A	No
6	R	1	A	No
7	S	(timeout)	0	–
8	S	(timeout)	1	–

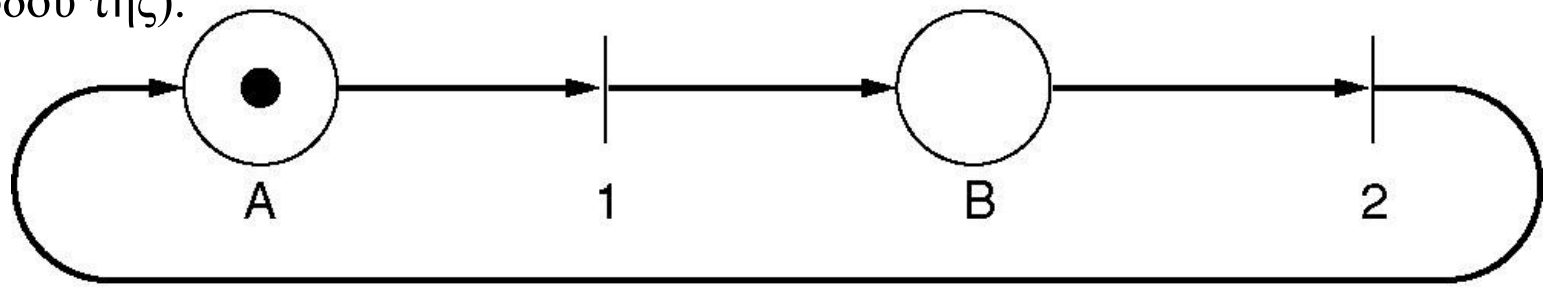
(b)

(a) Διάγραμμα καταστάσεων για το πρωτόκολλο 3. (b) Μεταβάσεις.

Μοντέλα δικτύων Petri (I)

Τα **δίκτυα Petri** είναι άλλη μια τεχνική μαθηματικής περιγραφής πρωτοκόλλων. Αποτελούνται από τα ακόλουθα τέσσερα βασικά συστατικά:

- **Θέση** (**place**): αντιπροσωπεύει μια κατάσταση στην οποία μπορεί να βρίσκεται (εν μέρει) το σύστημα (συμβολίζεται με έναν μεγάλο κύκλο).
- **Διακριτικό** (**token**): σημειώνει την τρέχουσα κατάσταση του συστήματος (συμβολίζεται με μια έντονη κουκίδα).
- **Μετάβαση** (**transition**): αντιπροσωπεύει το πέρασμα μεταξύ καταστάσεων, παριστάνεται με μια οριζόντια γραμμή ή κατακόρυφη ράβδο και έχει μηδέν ή περισσότερα
- **Τόξα εισόδου** (**input arcs**) (που φτάνουν σε αυτή από τις θέσεις εισόδου της) και μηδέν ή περισσότερα **τόξα εξόδου** (**output arcs**) (που κατευθύνονται προς τις θέσεις εξόδου της).



Δίκτυο Petri με δύο θέσεις και δύο μεταβάσεις.

Μοντέλα δικτύων Petri (II)

Μτβ.1: Κανονική μετάδοση πλαισίου 0 από αποστολέα.

Μτβ.2: Λήξη χρόνου αναμονής για τη μετάδοση του πλαισίου 0.

Μτβ.3: Κανονική μετάδοση πλαισίου 1 από αποστολέα.

Μτβ.4: Λήξη χρόνου αναμονής για τη μετάδοση του πλαισίου 0.

Μτβ.5: Απώλεια πλαισίου 0.

Μτβ.6: Απώλεια επιβεβαίωσης.

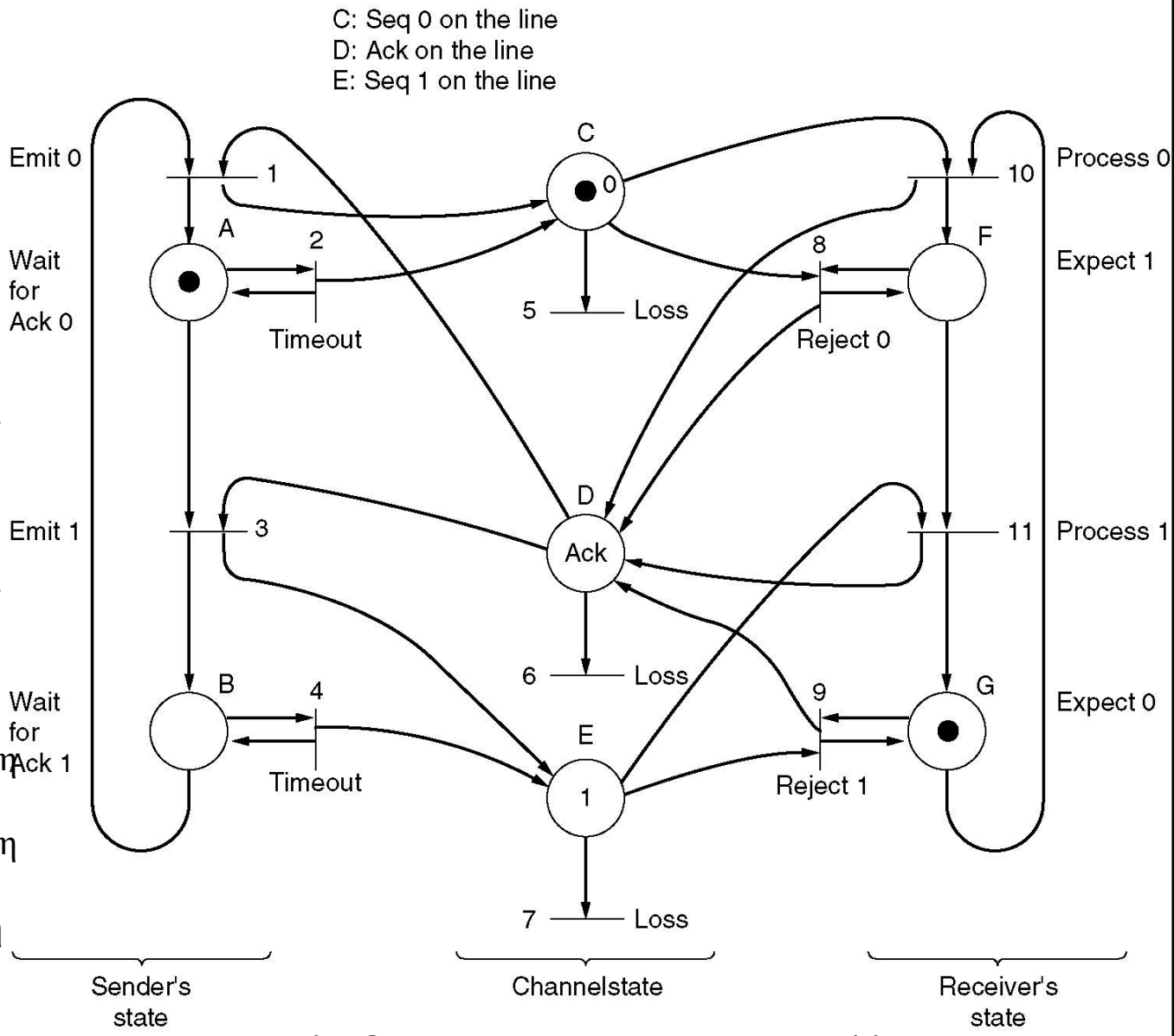
Μτβ.7: Απώλεια πλαισίου 1.

Μτβ.8: Παράδοση σε παραλήπτη κατεστραμμένου πλαισίου 0.

Μτβ.9: Παράδοση σε παραλήπτη κατεστραμμένου πλαισίου 1.

Μτβ.10: Άφιξη στον παραλήπτη του πλαισίου 0.

Μτβ.11: Άφιξη στον παραλήπτη του πλαισίου 1.



Ένα μοντέλο δικτύου Petri για το πρωτόκολλο PAR.

Μοντέλα δικτύων Petri (III)

Μτβ.1: BD → AC

Μτβ.2: A → AC

Μτβ.3: AD → BE

Μτβ.4: B → BE

Μτβ.5: C →

Μτβ.6: D →

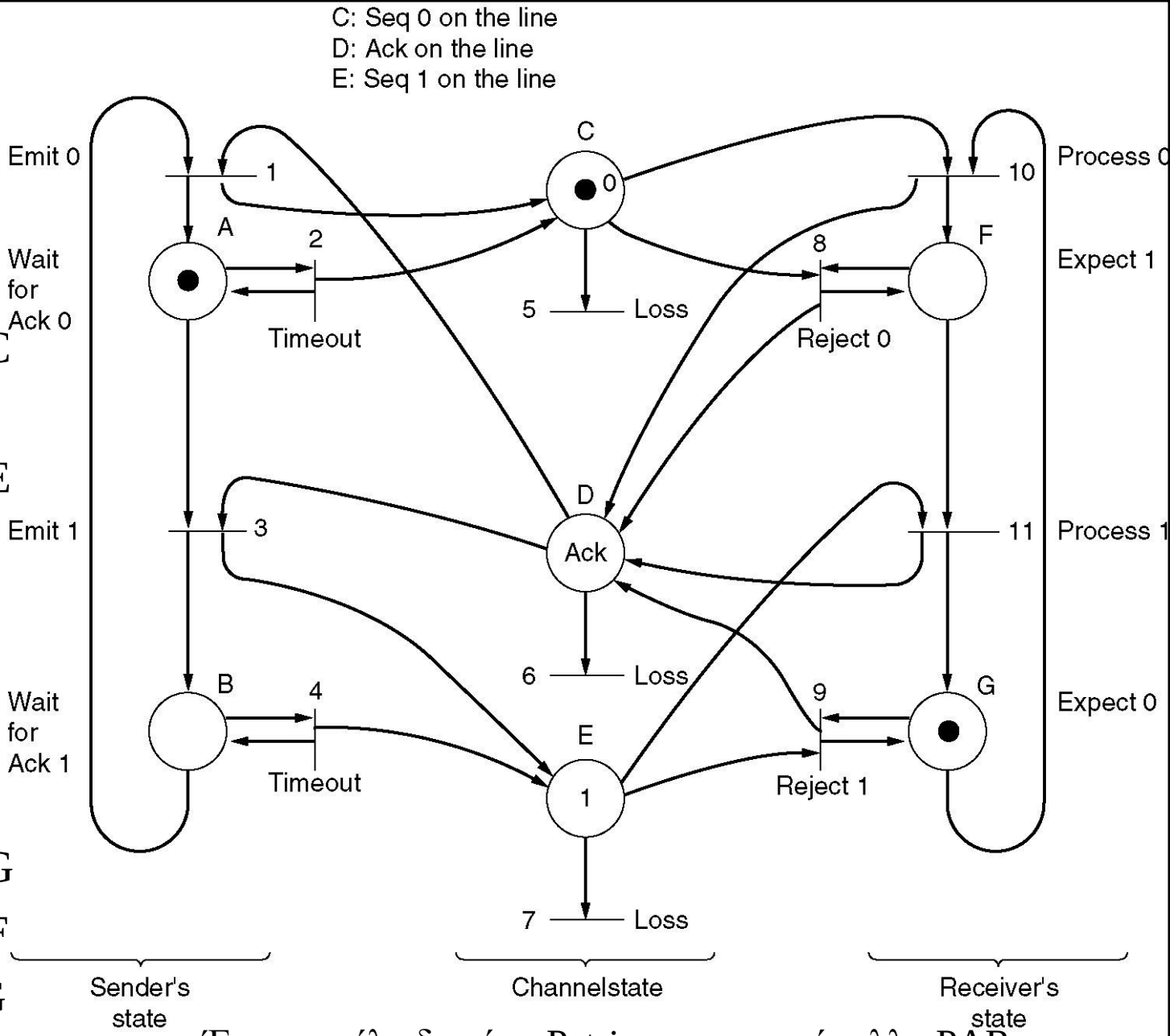
Μτβ.7: E →

Μτβ.8: CF → DF

Μτβ.9: EG → DG

Μτβ.10: CG → DF

Μτβ.11: EF → DG



Ένα μοντέλο δικτύου Petri για το πρωτόκολλο PAR.

Επαναληπτική άσκηση (I)

Αρχείο n byte (με n = πολλαπλάσιο των 1000 byte) πρόκειται να μεταφερθεί μέσω διαδρομής αποτελούμενης από το τερματικό αποστολής, το τερματικό προορισμού, **6 ζεύξεις** σημείου προς σημείο χωρητικότητας **4 Mbps** και 5 κόμβους μεταγωγής. Η **καθυστέρηση διάδοσης** σε κάθε ζεύξη είναι **2 ms**. Οι κόμβοι μεταγωγής υποστηρίζουν και μεταγωγή κυκλώματος και μεταγωγή πακέτου, οπότε το αρχείο μπορεί να μεταφερθεί είτε ως συρμός από bit, αφού προηγουμένως εγκατασταθεί κύκλωμα μέσω των κόμβων μεταγωγής, ή να διαιρεθεί σε πακέτα ίσου μήκους τα οποία αποστέλλονται το ένα μετά το άλλο. Τα πακέτα έχουν **επικεφαλίδα 24 byte** και **ωφέλιμο φορτίο 1000 byte**, η **αποθήκευση και προώθηση** σε κάθε κόμβο μεταγωγής εισάγει **καθυστέρηση 1 ms** σε κάθε πακέτο. Η εγκατάσταση κυκλώματος απαιτεί να σταλεί **μήνυμα μήκους 1024 byte**, να διατρέξει όλη τη διαδρομή με επιστροφή και σε κάθε κόμβο μεταγωγής να εισάγεται **καθυστέρηση 1 ms** στο μήνυμα. Οι κόμβοι μεταγωγής δεν εισάγουν καθυστερήσεις στα δεδομένα μετά την εγκατάσταση του κυκλώματος., ενώ δεν υπάρχουν σφάλματα μετάδοσης.

α) Για ποια τιμή του n είναι ο συνολικός αριθμός των bytes που στέλνονται μέσω του δικτύου μικρότερος για την αποστολή του αρχείου με μεταγωγή κυκλώματος από εκείνον με μεταγωγή πακέτου;

β) Για ποια τιμή του n είναι η συνολική καθυστέρηση για την αποστολή του αρχείου με μεταγωγή κυκλώματος μικρότερη από εκείνη για μεταγωγή πακέτου;

Παραδείγματα πρωτοκόλλων ζεύξης δεδομένων

Ακολούθως θα εξεταστούν δύο διαδεδομένα πρωτόκολλα ζεύξης δεδομένων:

- **HDLC** (Ελεγχος ζεύξης δεδομένων υψηλού επιπέδου ή High Level Data Link Control): κλασσικό πρωτόκολλο προσανατολισμένο σε bit, οι παραλλαγές του οποίου χρησιμοποιούνται επί δεκαετίες σε πολλές εφαρμογές.
- **PPP** (Πρωτόκολλο από σημείο σε σημείο ή Point-to-Point Protocol): είναι το πρωτόκολλο ζεύξης δεδομένων που χρησιμοποιείται για τη σύνδεση οικιακών υπολογιστών στο Internet.

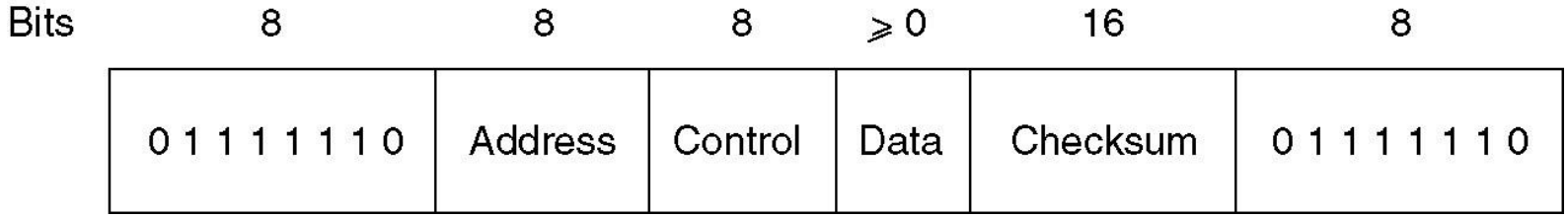
Ιστορία του HDLC

- Η IBM ανέπτυξε το Synchronous Data Link Control (SDLC), από το οποίο προέκυψαν ακολούθως όλα τα επόμενα τέτοια πρωτόκολλα.
- Ο ANSI μετέτρεψε το SDLC στο Advanced Data Communication Control Procedure (ADCCP).
- Ο ISO μετέτρεψε το SDLC στο High-level Data Link Control (HDLC).
- Η CCITT τροποποίησε το HDLC στο Link Access Procedure (LAP), το οποίο χρησιμοποιείται στα δίκτυα X.25.

Όλα αυτά τα πρωτόκολλα βασίζονται στις ίδιες αρχές. Είναι όλα προσανατολισμένα σε bit και χρησιμοποιούν συμπλήρωση με bit για διαφανή μετάδοση δεδομένων.

Έλεγχος ζεύξης δεδομένων υψηλού επιπέδου (I)

Όλα τα προσανατολισμένα σε bit πρωτόκολλα χρησιμοποιούν τη δομή που φαίνεται στο παρακάτω σχήμα.

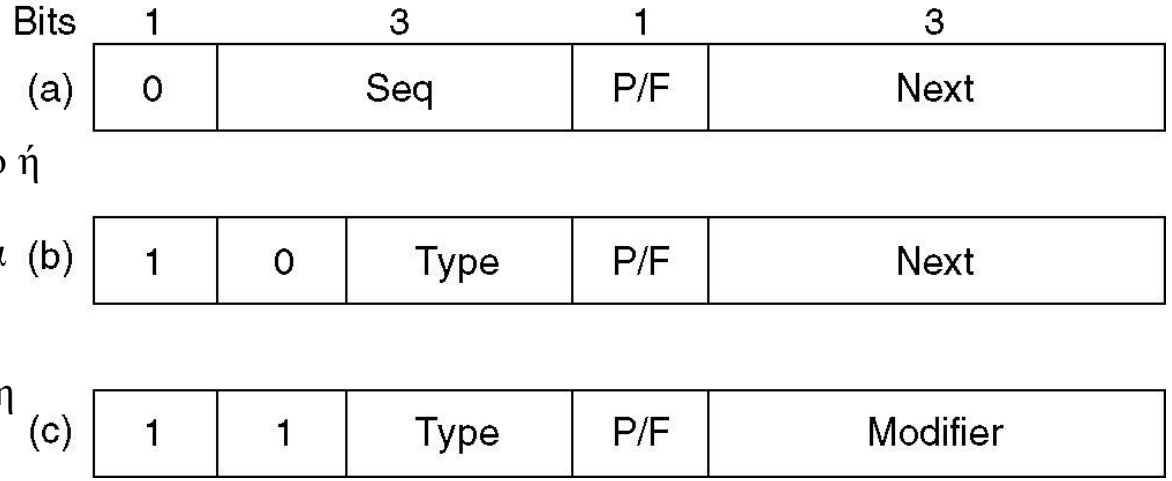


Το πεδίο **Διεύθυνση** (**Address**) έχει νόημα κυρίως στις γραμμές με πολλαπλά τερματικά, οπότε χρησιμοποιείται για να προσδιορίσει ένα από αυτά, ή στις γραμμές από σημείο σε σημείο για να ξεχωρίζουν οι εντολές από τις απαντήσεις. Το πεδίο **Έλεγχος** (**Control**) χρησιμοποιείται για αριθμούς ακολουθίας, επιβεβαιώσεις, κ.λπ. Το πεδίο **Δεδομένα** (**Data**) μπορεί να περιέχει οποιεσδήποτε πληροφορίες και να είναι αυθαίρετα μεγάλο. Το **Άθροισμα Ελέγχου** (**Checksum**) είναι ένας κωδικός κυκλικού ελέγχου πλεονασμού.

Το πλαίσιο οριοθετείται με ακολουθίες σημαίας: 01111110. Στις αδρανείς γραμμές από σημείο σε σημείο, μεταδίδονται συνεχώς ακολουθίες σημαίας. Το ελάχιστο πλαίσιο περιέχει τρία πεδία και έχει μέγεθος 32 bit, εξαιρουμένων των σημαιών των άκρων του.

Έλεγχος ζεύξης δεδομένων υψηλού επιπέδου (ΙΙ)

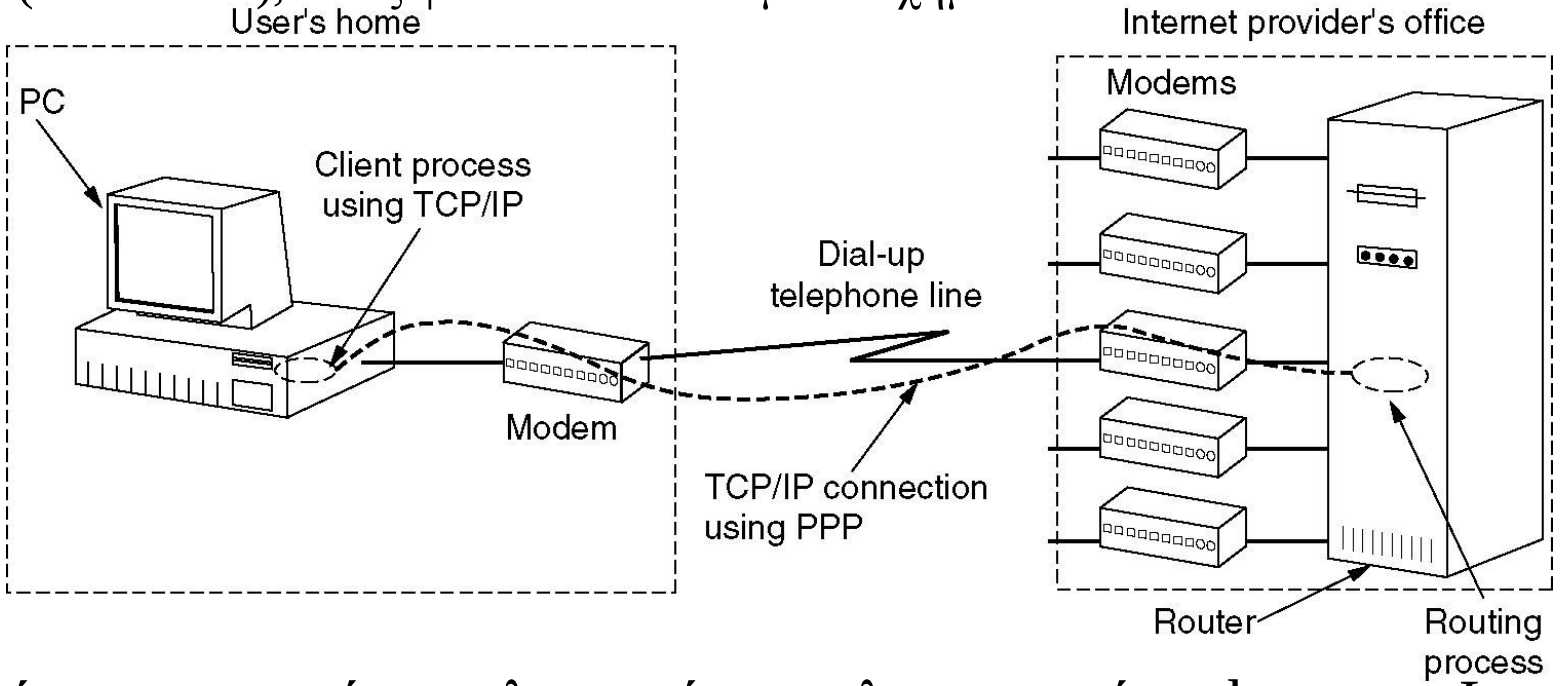
Υπάρχουν τρία είδη πλαισίων: (a) **Πληροφορίας** (**Information**), (b) **Εποπτείας** (**Supervisory**) και (c) **Μη Αριθμημένα** (**Unnumbered**). Η δομή του πεδίου Ελέγχου τους φαίνεται στο ακόλουθο σχήμα. Το πεδίο **Ακολουθία** (**Seq**) στο (a) είναι ο αριθμός ακολουθίας, ενώ το πεδίο **Επόμενο** (**Next**) είναι μια εμβόλιμη επιβεβαίωση που υποδηλώνει τον αριθμό του πλαισίου που αναμένεται. Το bit P/F (poll/final) χρησιμοποιείται όποτε ένα μηχάνημα επικοινωνεί με ομάδα τερματικών με περιόδευση (polling) (P: το μηχάνημα καλεί το τερματικό να στείλει δεδομένα και χρησιμοποιείται από όλα τα πλαίσια που στέλλονται από τερματικό πριν του τελευταίου, F: τελευταίο πλαίσιο). Στο (b) βλέπουμε ότι τα πλαίσια Εποπτείας διακρίνονται βάσει του πεδίου **Τύπος** (**Type**). Τύπος 0 (ή RECEIVE READY)→ δείχνει το επόμενο πλαίσιο που αναμένεται, τύπος 1 (ή REJECT) → δείχνει ότι ανιχνεύτηκε σφάλμα μετάδοσης οπότε το πεδίο Next περιέχει το πρώτο πλαίσιο που δε μεταδόθηκε σωστά, τύπος 2 (ή RECEIVE NOT READY) → δείχνει ότι όλα τα πλαίσια πριν το Next έχουν ληφθεί σωστά, αλλά ο παραλήπτης δεν είναι έτοιμος για λήψη. Ο τύπος 3 (ή SELECTIVE REJECT) ζητά την αναμετάδοση μόνο του πλαισίου που δηλώνεται στο Next. Τα μη αριθμημένα πλαίσια (c) χρησιμοποιούνται για έλεγχο ή δεδομένα, όταν απαιτείται μη αξιόπιστη ασυνδεσμική υπηρεσία. Απαντώνται στα προσανατολισμένα σε bit πρωτόκολλα, αλλά διαφέρουν από πρωτόκολλο σε πρωτόκολλο. Χρησιμοποιούν 5 bit για τη δήλωση του τύπου του πλαισίου.



Το επίπεδο ζεύξης δεδομένων στο Internet

Στην πράξη, υπάρχουν κυρίως δύο περιπτώσεις που έχουμε επικοινωνία από σημείο σε σημείο στο Internet:

- Μισθωμένες γραμμές μεταξύ δρομολογητών στο Internet (router-router).
- Γραμμές dial-up μεταξύ οικιακών προσωπικών υπολογιστών και δρομολογητών των ISPs (host-router), όπως φαίνεται στο επόμενο σχήμα.



Οικιακός προσωπικός υπολογιστής που λειτουργεί ως host στο Internet

PPP – Point to Point Protocol (I)

Το πρωτόκολλο **PPP** (πρωτόκολλο από σημείο σε σημείο ή point-to-point protocol) αποτελείται από τρεις βασικές λειτουργίες:

1. Μέθοδο πλαισίωσης που οριοθετεί μονοσήμαντα αρχή/τέλος πλαισίου και καλύπτει και την ανίχνευση σφαλμάτων.
2. Πρωτόκολλο ελέγχου γραμμής που ονομάζεται **LCP** (Link Control Protocol) για τον έλεγχο και ενεργοποίηση της γραμμής, για διαπραγμάτευση επιλογών και την ομαλή απενεργοποίησή της. Υποστηρίζει σύγχρονα και ασύγχρονα κυκλώματα, και κωδικοποιήσεις προσανατολισμένες σε byte ή σε bit.
3. Μηχανισμό διαπραγμάτευσης των επιλογών του επιπέδου δικτύου, που είναι ανεξάρτητος από το χρησιμοποιούμενο πρωτόκολλο επιπέδου δικτύου. Ο μηχανισμός αυτός ονομάζεται **NCP** (Network Control Protocol).

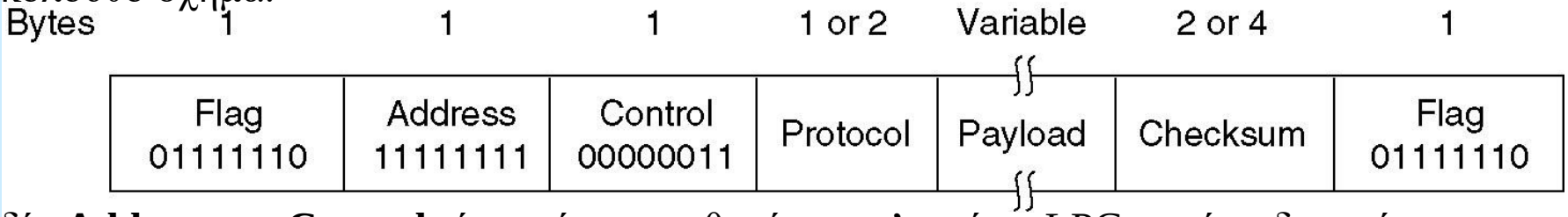
Τυπικό σενάριο χρήσης του PPP

Έστω οικιακός χρήστης που καλεί κάποιον φορέα παροχής υπηρεσιών Internet (ISP) για να κάνει το home PC του έναν προσωρινό host στο Internet. Οι φάσεις που ακολουθούν είναι:

1. Φάση εγκατάστασης φυσικής σύνδεσης:
   - Το PC καλεί το δρομολογητή του ISP μέσω modem.
   - Το modem του δρομολογητή απαντά στην κλήση και εγκαθιστά φυσική σύνδεση.
2. Φάση διαπραγμάτευσης παραμέτρων του επιπέδου ζεύξης δεδομένων:
   - Το PC στέλνει στο δρομολογητή σειρά πακέτων LCP στο πεδίο ωφέλιμου φορτίου ενός ή περισσοτέρων πλαισίων PPP. Αυτά τα πακέτα και οι απαντήσεις τους επιλέγουν τις παραμέτρους PPP που θα χρησιμοποιηθούν.
3. Φάση διαπραγμάτευσης παραμέτρων του επιπέδου δικτύου:
   - Σειρά πακέτων NCP απoοστέλλονται για να διευθετηθεί το επίπεδο δικτύου και για να ανατεθεί μια διεύθυνση IP στο PC (εάν το PC επιθυμεί να εκτελέσει στοίβα πρωτοκόλλων TCP/IP).
4. Φάση επικοινωνίας δεδομένων:
   - Το PC είναι πια host στο Internet και στέλνει/λαμβάνει πακέτα IP μέσω της εγκατεστημένης σύνδεσης.
5. Φάση αποδέσμευσης της σύνδεσης:
   - Μόλις ο χρήστης τελειώσει, το NCP τερματίζει τη σύνδεση στο επίπεδο δικτύου και αποδεσμεύει τη διεύθυνση IP.
   - Το LCP απενεργοποιεί τη σύνδεση στο επίπεδο ζεύξης δεδομένων.
   - Ο υπολογιστής λέει στο modem να κλείσει το τηλέφωνο και να αποδεσμεύσει τη φυσική σύνδεση.

Η πλήρης μορφή πλαισίων του PPP για λειτουργία σε μη αριθμημένη μορφή

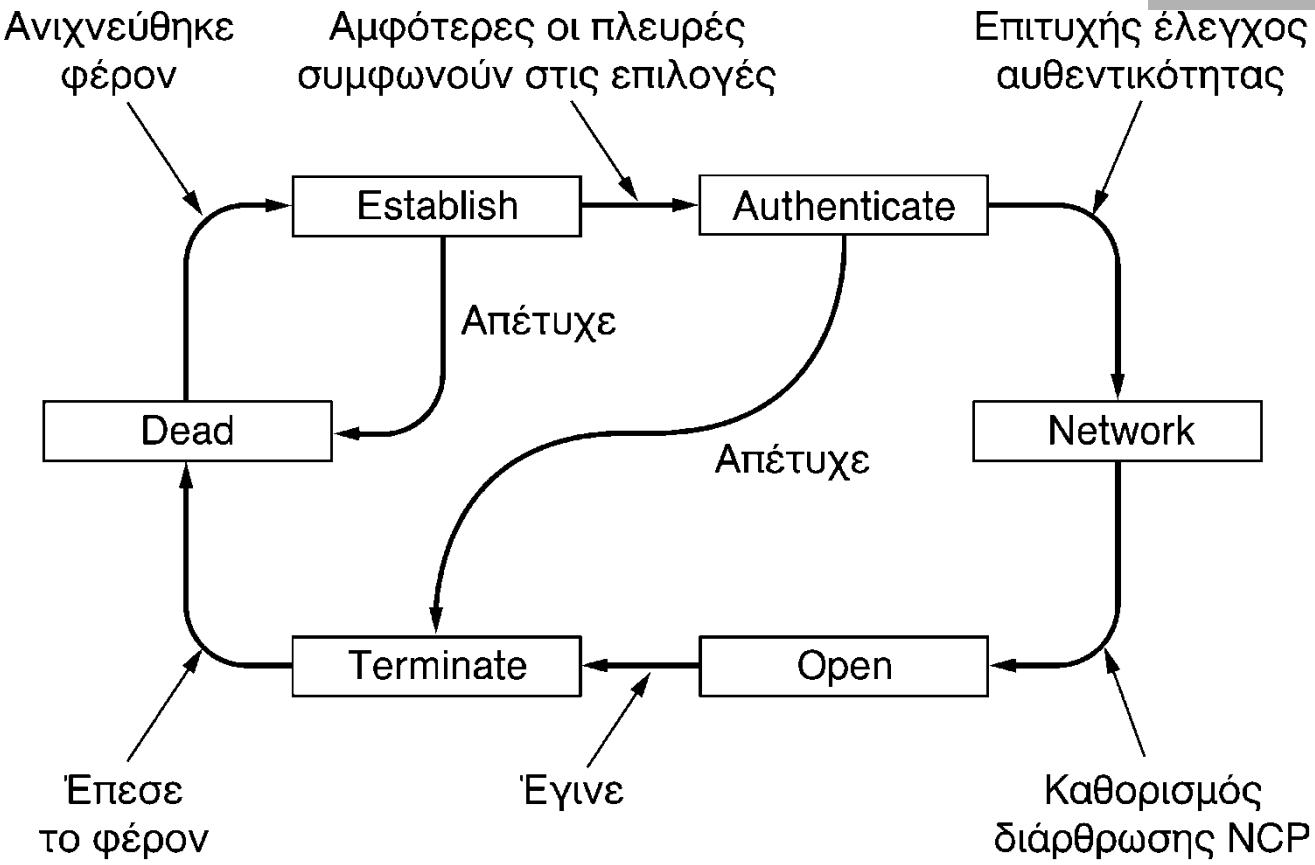
Το πλαίσιο του PPP επελέγη έτσι ώστε να μοιάζει πολύ με αυτό του HDLC frame, όπως φαίνεται στο ακόλουθο σχήμα:



Τα πεδία **Address** και **Control** είναι πάντα σταθερά και γι'αυτό το LPC παρέχει δυνατότητα διαπραγμάτευσης επιλογής ώστε να παραλείπονται τα πεδία αυτά (→οικονομία 2 Bytes ανά πλαίσιο). Το πεδίο **Protocol** δείχνει τι είδους πακέτο περιέχεται στο **Payload** (π.χ. LCP, NCP, IP, IPX, AppleTalk). Έχει προεπιλεγμένο μέγεθος 2 Bytes αλλά μπορεί να μειωθεί σε 1 κατόπιν διαπραγμάτευσης μέσω LCP. Το πεδίο **Payload** έχει μεταβλητό μήκος (με προεπιλεγμένη τιμή 1500 Bytes), μέχρι κάποιο μέγιστο διαπραγματεύσιμο όριο μέσω LCP. Τέλος, ακολουθεί το πεδίο **Checksum** που έχει προεπιλεγμένο μήκος 2 Bytes αλλά μπορεί να μειωθεί σε 1 κατόπιν διαπραγμάτευσης μέσω LCP.

Συνοπτικά, το PPP αποτελεί μηχανισμό πλαισίωσης πολλών πρωτοκόλλων, που είναι κατάλληλος για χρήση πάνω από modem, γραμμές HDLC προσανατολισμένες σε bit, SONET και άλλα φυσικά επίπεδα. Υποστηρίζει ανίχνευση σφαλμάτων, διαπραγμάτευση επιλογών, συμπίεση κεφαλίδων και προαιρετικά αξιόπιστη μετάδοση, ενώ χρησιμοποιεί μορφή πλαισίων παρόμοια με του HDLC.

PPP – Point to Point Protocol (II)



Απλοποιημένο διάγραμμα φάσεων για την ενεργοποίηση και απενεργοποίηση μιας γραμμής.

PPP – Point to Point Protocol (III)

Όνομα	Κατεύθυνση	Περιγραφή
Configure-request	I → P	Κατάλογος προτεινομένων επιλογών και τιμών
Configure-ack	I ← P	Γίνονται δεκτές όλες οι επιλογές
Configure-nak	I ← P	Μερικές επιλογές δεν είναι αποδεκτές
Configure-reject	I ← P	Μερικές επιλογές δεν είναι διαπραγματεύσιμες
Terminate-request	I → P	Αίτηση να κλείσει η γραμμή
Terminate-ack	I ← P	OK, η γραμμή έκλεισε
Code-reject	I ← P	Λήφθηκε άγνωστη αίτηση
Protocol-reject	I ← P	Ζητήθηκε άγνωστο πρωτόκολλο
Echo-request	I → P	Στείλε πίσω αυτό το πλαίσιο παρακαλώ
Echo-reply	I ← P	Να, επιστρέφεται
Discard-request	I → P	Αγνόησε αυτό το πλαίσιο (για έλεγχο)

Οι τύποι πλαισίων του LCP.