Ονοματεπώνυμο: Σωτήριος Λουκάς Καμπύλης

ΑΕΜ: 3805

Σημειώσεις - Διευκρινήσεις: Για την πραγματοποίηση της εργασίας χρησιμοποιήθηκε το προγραμματιστικό περιβάλλον του Codeblocks, ***γλώσσα C++****. Επίσης, για την υλοποίηση της εργασίας δημιουργήθηκε και η* **κλάση CRC** *(.cpp και .h).*

ΚΛΑΣΗ CRC: Η κλάση αυτή αναπαριστά το δυαδικό σήμα. Δηλαδή, σε κάθε μια θέση του βρίσκονται '0' ή '1'.Βασίζεται στην υλοποιημένο από την C++ δομή ***vector***.Περιέχει στο *.h* έναν κενό κατασκευαστή, ένα καταστροφέα , έναν κατασκευαστή αντιγράφων , ένα κατασκευαστή που δέχεται ένα vector bool και έναν που δέχεται μια συμβολοσειρά. Ακόμα, υπάρχει ακόμα ένας κατασκευαστής που επιτελεί τη σημαντικότερη λειτουργία, δέχεται ως όρισμα έναν ακέραιο αριθμό **k** (που στην προκειμένη περίπτωση δίνεται ως είσοδο από το χρήστη το 20) και ύστερα παράγει ένα δυαδικό σήμα που αποτελείται από **k** ψηφία και κάθε ψηφίο έχει ίση πιθανότητα να είναι '0' ή '1', καθώς έχει χρησιμοποιηθεί συνάρτηση παραγωγής τυχαίων αριθμών (*μέσω της βιβλιοθήκης* **<ctime>)**, ώστε να παράγονται νέα αποτελέσματα μετά από κάθε εκτέλεση του προγράμματος. Μάλιστα, εκτός κλάσης έχει υπερφορτωθεί ο τελεστής "***!=***".Τέλος υπάρχουν άλλες τέσσερις μέθοδοι (η **void shift**, η **Crc division**, η **Crc noise** και η **bool zero**) οι οποίες θα αναλυθούν παρακάτω. Αρχικά, η μέθοδος **shift** δέχεται ως όρισμα έναν ακέραιο αριθμό, ο οποίος είναι ο αριθμός των ψηφίων του σταθερού διαρέτη μείον ένα, δημιουργώντας τόσες θέσεις με μηδενικά στο τέλος του vector έτσι ώστε το σήμα μας να είναι έτοιμο για την διαδικασία της διαίρεσης (modulo 2) κάθε φορά (όπως αναφέρεται και στην εκφώνηση).Ύστερα, παρατηρούμε την μέθοδο **division** που δέχεται ως παράμετρο ένα αντικείμενο *CRC* και εκτελεί τη modulo 2 διαίρεση μεταξύ των δύο δυαδικών σημάτων, χρησιμοποιώντας επαναλήψεις και κάνοντας τις απαραίτητες

πράξεις. Αμέσως από κάτω, βρίσκεται υλοποιημένη η συνάρτηση **noise** που δέχεται ως όρισμα ένα αντικείμενο τύπου CRC από τον μεταδότη και ανάλογα με το BER (Bit Error Rate) που δίνεται ως είσοδος από το χρήστη, υπάρχει τόση πιθανότητα κάθε ψηφίου ενός σήματος να αλλάξει. **Προσοχή:το BER δίνεται ως ακέραιος αριθμός, για παράδειγμα το BER 10^-3 θα πρέπει να δοθεί ως 1000 (αφού δηλώνει την πιθανότητα πως 1 στα 1000 ψηφία μπορεί να αλλάξει λόγω του θορύβου του καναλιού).** Τέλος, έχει υλοποιηθεί και μία τελευταία συνάρτηση (η empty) η οποία χρησιμοποιείται για να ελέγξουμε αν το υπόλοιπο από τη διαίρεση δύο σημάτων είναι 0.

ΜΕΘΟΔΟΣ main: Στην μέθοδο main εκτελούνται όλες οι βασικές λειτουργίες έτσι ώστε να μην προκύψει λανθασμένη παραγωγή σήματος. Το πρόγραμμα μας κατά την εκτέλεση του θα εμφανίσει τα απαραίτητα μηνύματα (Tips) και θα περιμένει τις αντίστοιχες εισόδους από το χρήστη, οι οποίες θεωρούνται εξαρχής ορθές και δεν γίνεται έλεγχος τους. (οπότε πρέπει να τοποθετούνται ακριβώς σύμφωνα με τις συμβουλές!) Οι *βασικές μεταβλητές* που έχουμε στην main είναι **4** και είναι οι εξής: Η **Signal** που είναι το αρχικό σήμα, η **P** που είναι ο σταθερός διαιρέτης του σήματος, η **T** που σε αυτήν φτιάχνεται και αποθηκεύεται το τελικό σήμα πριν περάσει από το ενθόρυβο κανάλι και η **Noised** στην οποία αποθηκεύεται το σήμα που έχει περάσει από το ενθόρυβο κανάλι. Αρχικοποιούνται επίσης δύο μετρητές. Ένας για την καταμέτρηση των σημάτων που ο κυκλικός κώδικας ανίχνευσης CRC θα τον εντόπιζε ως λανθασμένα, το οποίο επαληθεύεται καθώς το υπόλοιπο της διαίρεσης θα ήταν διάφορο του μηδέν και ένας για την καταμέτρηση των σημάτων που θα έφταναν στον αποδέκτη που αν και λανθασμένα δεν θα εντόπιζε ο CRC, αυτό συμβαίνει καθώς το ενθόρυβο κανάλι μπορεί να μετατρέψει το σήμα σε ένα διαφορετικό το οποίο όμως πάλι θα δίνει υπόλοιπο μηδέν, αυτό ελέγχεται συγκρίνοντας το σήμα που παράγει η μέθοδος **noise** με το αρχικό σήμα **T**. Τέλος, κάνει τις απαραίτητες πράξεις και ελέγχους για την επεξεργασία των σημάτων και την αύξηση των μετρητών. Οι πράξεις οι οποίες

γίνονται είναι οι εξής: Πρώτα κάνουμε το σήμα **T** ίσο με το σήμα **Signal**, έπειτα χρησιμοποιούμε τη συνάρτηση **shift** που στην πραγματικότητα εκτελεί την πράξη: *2^(n-k)*Signal*, όπου n είναι ο αριθμός των bits του **T** και **k** ο αριθμός των bits του **Signal** που ταυτόχρονα είναι και τα πρώτα bits του **T**. Τέλος, κάνουμε τη διαίρεση και αντικαθιστούμε τα μηδενικά στα τελευταία bits του **T** με το υπόλοιπο. Περνάμε το **T** από το ενθόρυβο κανάλι, ελέγχοντας το σήμα και μετά το πέρασμα του συγκρίνοντας τα δύο σήματα που έχουμε παράγει.

Δοκιμή Προγράμματος: Ακριβώς από κάτω υπάρχει ένα στιγμιότυπο εκτέλεσης του προγράμματος σύμφωνα με τους αριθμούς που λέει η εκφώνηση (k=20, P=110101 και BER=10^-3 (αλλά με το αναπαράσταση στο 1000)).

```
Give k(Tip:Give 20 for the exercise):20
Give P(Tip:Give 110101 for the exercise):110101
Give (BER) Bit Error Rate(Tip:Give 1000 for 10^-3):1000
Wait some seconds for the wrong messages to get detected by the CRC...

The number of wrong messages get detected by the CRC:24138
The number of wrong messages that reached the receiver and were not get detected by the CRC:15

Process returned 0 (0x0)   execution time : 32.045 s
Press any key to continue.
```

Στιγμιότυπο προγράμματος

Συμπεράσματα: Τα μηνύματα που φθάνουν με σφάλμα στο αποδέκτη και **δεν ανιχνεύονται** από το CRC είναι 15 (παρατηρούμε μικρό αριθμό μηνυμάτων που δεν ανιχνεύονται γιατί οι πιθανότητες είναι πάρα πολύ λίγες να μην εντοπιστούν τα σφάλματα). Το ποσοστό των μηνυμάτων που ανιχνεύονται ως εσφαλμένα από το CRC έχοντας φτάσει στον αποδέκτη είναι 24138. Το ποσοστό των μηνυμάτων που φθάνουν με σφάλμα στον αποδέκτη, ανιχνεύσιμα και μη από τον CRC, είναι το άθροισμα (των ποσοστών) των δύο παραπάνω. Ωστόσο, με τις συγκεκριμένες εισόδους κάθε φορά που εκτελείται το πρόγραμμα, το ποσοστό των μηνυμάτων που φθάνουν με σφάλμα στο αποδέκτη και δεν ανιχνεύονται από το CRC θα είναι πάλι αρκετά μικρό!