Οδηγίες για το AWGN καναλι, υλοποιημένο σε VHDL.

Βασικά η παρακάτω αναφορά δεν έχει σκοπό να δείξει την αρχή λειτουργίας του component LLR\_chanel αλλά τον τρόπο και τους χρονισμούς του, ώστε να φτιάξετε μια FSM απλά και γρήγορα.

Αρχικά αυτό το component απαρτίζεται από επιμέρους components. Ο θόρυβος θα πρέπει α μπαίνει σε κάθε bit της κωδικοποιημένης λέξης και η έξοδος θα πρέπει να είναι ένας πίνακας όπου η μια διάσταση είναι ίση με την μήκος της εισόδους και η άλλη διάσταση είναι ο πλήθος των bits που εκφράζουν το εν λογω LLR ενός bit.

Τα παραπάνω ορίζονται ως είσοδοι , έξοδοι αντίστοιχα στο LLR\_channel καθώς και κάποια άλλα σήματα, τα οποία αναλύω παρακάτω.

Για να μπει θόρυβος σε κάθε bit της λέξης εισόδου θα πρέπει να γίνει parallel-2-serial και αυτό γίνεται με το component “mux” το οποίο παίρνει σαν είσοδο την κωδική λέξη και βγάζει 2 bit (llr2bit) αυτής κάθε clk. To σημα llr2bit θα μπει στην είσοδο του component “mysystem” το οποίο θα παράξει το διαμορφωμένο σύμβολο για κάθε bit με διαμόρφωση BPSK.

Η διαμόρφωση BPSK είνια γνωστό ότι μπορεί να περιγραφεί με την εξής γραμμική εξίσωση

BPSK = 1 – 2\*X , όπου Χ είναι ένα από bit της λέξης.

Η σχέση που υπολογίζει τα LLR είναι : 2y/σ2 , y = x+σ ,χ διαμορφωμένο bit.

Οι τιμές των σ και σ2 , για διάφορες στάθμες θορύβου έχουν υπολογιστεί μέσω matlab και έχουν εισαχθεί σε ένα LUT (snrlut.vhd) όπου για κάθε στάθμη θορύβου εξάγονται αυτές οι δυο ποσότητες και χρησιμοποιούνται από το “mysystem.vhd”. Για λόγους debugging για 0db SNR έχει οριστεί το σ = 0 και το σ2 = -1, το οποίο θα παράξει LLR μέγιστης αξιοπιστιας και ο decoder θα βρεί την κωδικη λέξη χωρίς κανένα λάθος. Αν στην ίδια περίπτωση το σ = 1 με αντίστοιχο format (8.32) θα αναμένεται να γίνει μέτρηση στα 0dB SNR.

Έχοντας το σ = 0 και το σ2 = -1, θα πρέπει η διαμόρφωση να γίνεται όχι όπως αναφέρεται παραπάνω αλλά ως BPSK = 2\*X – 1 το οποίο στην έξοδο του καναλιού θα έχουμε την σωστά διαμορφωμένη λέξη.

Το parallel-2-serial γίνεται χρησιμοποιώντας εναν counter του οποίου η έξοδος θα χρησιμοποιηθεί για να επιλεχθεί στο “mux.vhd” ποια δυαδα bits θα παρθούν για να προστεθεί θόρυβος. Επομένως αν ο counter έχει σητν έξοδο του την τιμή 5, το 10 και το 11 bit της λέξης εισόδου θα επιλεχθεί για να του προστεθεί θόρυβος.

Η αντίστροφη διαδικασία γίνεται και για την έξοδο, όπου κάθε φορά που υπολογίζονται τα llr1 και llr2 από το “mysystem”, δηλαδή όταν παραχθεί ένα ζεύγος απο llr θα αποθηκευτούν σε ένα register-file όπου το σημα enable για κάθε καταχωρητή παράγεται από εναν shift-register. O shift-register θα πρέπει να γίνει enable για ένα clk και μονο καθώς σε κάθε clk πρέπει να ενεργοποιείται μόνο ενας καταχωρητής του register-file. Όταν τελειώσει το shiftαρισμα και μόνο τότε είναι σωστά τα llr και έτοιμα για το decoder.

Το component “LLR\_channel” έχει σαν είσοδο όπως αναφέρθηκε παραπάνω την λέξη εισόδου, llr\_count\_en : ενεργοποιεί τον llr\_counter για το parallel-2-serial

dinredy : να είναι μόνιμα στο ‘1’

start\_shift : ενεργοποιει το shift, να γίνει ‘1’ για ένα clk μονο

SNR : σταθμη θορύβου

llr\_counter : ο counter για το p-2-s

shift\_reg :

readyout : όταν γίνει ‘1’ μετά το rst του συστήματος, τοτε μπορεί να βγάλει σωστά LLR

Το πρόβλημα που γεννιέται είναι τότε θα πρέπει να γίνει το llr\_count\_en =’1’ και πότε το start\_shift=’1’, τα οποία σήματα θα πρέπει να ελέγχονται από την FSM.

Όταν είναι έτοιμη η κωδικοποιημένη λέξη και διαθέσιμη στην είσοδο του καναλιού θα πρέπει να γίνει στο επόμενο clk το llr\_count\_en =’1’.

Θα πρέπει όταν γίνει ο llr\_counter = “7” θα πρεπει να γίνει το start\_shift =’1’ για ένα clk.

Το πιο σημαντικό είναι ότι δεν θα πρέπει ο llr\_counter να πάθει overflow ενώ γίνεται shifting. Άρα θα πρέπει να παραμείνει στην τιμή “FF” (το πόσα F εξαρτάται απο το μήκος της λέξης), μέχρι να είναι διαθέσιμη η επόμενη κωδικοποιημενη λέξη ή να μην χρειάζεστε άλλο τα LLR.

Το πως θα γίνει η παραπάνω διαδικασία είναι στην δικιά σας ευχέρια για το γράψιμο της FSM.