Υλοποίηση FEC decoder σε FPGA χαμηλής κατανάλωσης

Ιωάννης Χονδρούλης

Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Πολυτεχνείο Κρήτης

Ιανουάριος 2018

Πίνακας Περιεχομένων

1 Εισαγωγή

1.1 Πρόλογος

Ένα Ασύρματο Δίκτυο Αισθητήρων (ΑΔΑ) αποτελείται από κόμβους που συλλέγουν δεδομένα λαμβάνοντας μετρήσεις από το φυσικό περιβάλλον, τα επεξεργάζονται και τα αποστέλλουν ασύρματα σε άλλους κόμβους-συλλέκτες ή άλλα συστήματα επεξεργασίας (π.χ. Η/Υ). Με αυτό τον τρόπο ο χρήστης αποκτά πρόσβαση στα δεδομένα, μέσω του σταθμού-βάση που συνέλλεξε το σύνολο των μετρήσεων. Χαρακτηριστικό των κόμβων ενός ΑΔΑ είναι ότι λειτουργούν τις περισσότερες φορές με σημαντικούς περιορισμούς, ειδικότερα όσον αφορά τις φυσικές τους διαστάσεις και την ευκολία φυσικής επαφής εκ μέρους του χρήστη. Ως εκ τούτου, υπάρχει η ανάγκη να λειτουργούν αυτόνομα με δικά τους ενεργειακά μέσα (τυπικά μπαταρίες) για όσο το δυνατόν μεγαλύτερα χρονικά διαστήματα.

Αποτέλεσμα των προαναφερθέντων περιορισμών είναι η χρήση υπολογιστικών στοιχείων με εξαιρετικά περιορισμένες δυνατότητες και αντίστοιχα πολύ μικρή κατανάλωση ενέργειας. Κατά συνέπεια, για την πραγματοποίηση της ασύρματης επικοινωνίας χρησιμοποιούνται πολύ απλοί μηχανισμοί και πρωτόκολλα ώστε από τη μία να ελαχιστοποιηθεί η κατανάλωση ισχύος και από την άλλη να μπορούν οι απλοί μικροεπεξεργαστές να διαχειριστούν το υπολογιστικό κόστος. Έτσι, οι ασύρματοι κόμβοι τυπικά οργανώνονται έτσι ώστε να μεταδίδουν δεδομένα σε μικρή απόσταση και με χαμηλή ισχύ, με τον κάθε κόμβο να αποστέλλει πληροφορία σε ένα γειτονικό του κόμβο. Έτσι τα δεδομένα μεταδίδονται μέσω διάφορων ενδιάμεσων κόμβων μέχρι να καταλήξουν στον τελικό τους προορισμό (multi-hop τοπολογίες). Με τον τρόπο αυτό δρομολόγησης των δεδομένων οι ενδιάμεσοι κόμβοι βρίσκονται συνεχώς σε λειτουργία, αφού θα πρέπει να καταγράφουν, να λαμβάνουν, να αποστέλλουν δεδομένα και να είναι συνεχώς διαθέσιμοι να δεχτούν νέα δεδομένα από άλλους κόμβους. Για τη διαδικασία αυτή αφιερώνεται ένα μεγάλο μέρος των ενεργειακών πόρων του δικτύου, ειδικά όταν λόγω ύπαρξης θορύβου ή παρεμβολών απαιτούνται συχνές και πολλαπλές επαναμεταδόσεις πακέτων προκειμένου να φτάσει αξιόπιστα ένα μήνυμα από ένα κόμβο-πηγή στο τελικό προορισμό.

Η εργασία αυτή προτείνει τη χρήση αισθητά πιο περίπλοκων μηχανισμών επικοινωνίας που επιτρέπουν την αποτελεσματική ανταλλαγή δεδομένων σε περιπτώσεις υψηλού θορύβου στο κανάλι επικοινωνίας και τη μετάδοση σε μεγαλύτερες αποστάσεις. Για την εφαρμογή των μηχανισμών αυτών φυσικά απαιτείται περισσότερη επεξεργαστική ισχύς, άρα και ενεργειακοί πόροι, από τους κόμβους του δικτύου για τη μετάδοση ενός πακέτου δεδομένων. Ταυτόχρονα όμως μπορούν να μειωθούν δραματικά οι επαναμεταδόσεις πακέτων, και άρα να προκύψει σημαντική συνολική εξοικονόμηση ενέργειας παρά το αυξημένο κόστος μετάδοσης ανά πακέτο. Σε επίπεδο δικτύου, με χρήση τέτοιων μηχανισμών επικοινωνίας, καθίσταται εφικτή η χρήση απλούστερων τοπολογιών (single-hop), με αποτέλεσμα περαιτέρω ενεργειακά οφέλη λόγω απλούστερων αλγορίθμων δρομολόγησης και διαδικασιών ελέγχου λειτουργίας του δικτύου, ενώ προσφέρονται δυνατότητες να εξοικονομηθεί ενέργεια σε κάθε κόμβο ξεχωριστά εφόσον αυτός μπορεί να τοποθετηθεί σε λειτουργία χαμηλής κατανάλωσης ισχύος (sleep mode) για μεγαλύτερα χρονικά διαστήματα.

1.2 Σκοπός

Το συνηθέστερο σενάριο για την αποστολή δεδομένων με δρομολόγηση multi-hopόπως αναφέρθηκε προηγουμένως, είναι τα πακέτα δεδομένων να αποστέλλονται χωρίς κάποια κωδικοποίηση ή άλλη πληροφορία που να επιτρέπει διόρθωση λαθών. Αν ένα πακέτο δεν φτάσει στον προορισμό του ή παραληφθεί με κάποιο λάθος, ο παραλήπτης αποστέλλει αίτημα επανάληψης της αποστολής. Η τεχνική αυτή (Automatic Repeat Request – ARQ) βελτιώνει σε κάποιο βαθμό την αξιοπιστία στην επικοινωνία, δεν εγγυάται όμως την αποστολή του πακέτου σε δυσμενείς συνθήκες επικοινωνίας καθώς μπορεί να επαναληφθεί μόνο ένα πεπερασμένο αριθμό φορών. Επίσης, μπορεί το μέγεθος του κάθε πακέτου να καθορίζεται μόνο από την καθαρή πληροφορία, και άρα να παραμένει σχετικά μικρό, οι πολλαπλές αποστολές του ίδιου πακέτου όμως αυξάνουν δραματικά την ενεργειακή κατανάλωση κατά τη μετάδοση.

Η ιδέα που επιδιώκεται να υλοποιηθεί σε αυτή την εργασία είναι ότι αν εφαρμοστεί κατάλληλη μορφοποίηση των δεδομένων με βάση κάποιο σχήμα κωδικοποίησης, από τη μία θα αυξηθεί το μέγεθος των πακέτων και η απαιτούμενη επεξεργασία ανά πακέτο, από την άλλη όμως θα μειωθούν σημαντικά οι πολλαπλές αποστολές. Αυτό γιατί ο παραλήπτης θα μπορεί να ανακατασκευάσει τα bitsπου έφτασαν λανθασμένα κατά τη μετάδοση. Η παραδοχή αυτή επιτρέπει στο σχεδιαστή του δικτύου να οργανώσει τους κόμβους χωρίς περιορισμούς όπως ο θόρυβος στο κανάλι, οι μεγάλες αποστάσεις μεταξύ κόμβων ή τα φυσικά εμπόδια να έχουν τόσο έντονη επίδραση στην επικοινωνία, όπως συμβαίνει χωρίς τη κατάλληλη διαμόρφωση των πακέτων.

Ακολουθώντας τη τακτική που χρησιμοποιείται ευρέως σε άλλα ασύρματα δίκτυα, προτείνεται η χρήση κάποιας τεχνικής Forward Error Correction (FEC). Η χρήση ενός τέτοιου σχήματος περιλαμβάνει μια σχετικά χαμηλής πολυπλοκότητας διαδικασία κωδικοποίησης από τη μεριά του αποστολέα και αντίστοιχα μια αρκετά πολύπλοκη διαδικασία αποκωδικοποίησης από τη μεριά του παραλήπτη. Σε τοπολογίες multi-hop ο κάθε ενδιάμεσος κόμβος επιφορτίζεται με τις διαδικασίες κωδικοποίησης/ αποκωδικοποίησης, κάτι που επηρεάζει δραματικά την ενεργειακή και χρονική απόδοση του δικτύου. Επιπλέον, προϋποθέτει ότι οι υπολογιστικές μονάδες των κόμβων θα έχουν αναβαθμισμένες δυνατότητες για να ανταπεξέλθουν στον αντίστοιχο φόρτο εργασίας. Για τους λόγους αυτούς η τεχνική FEC αποφεύγεται σε κλασικές τοπολογίες multi-hop.

Σε τοπολογίες τύπου single-hop,η σχετικά απλή κωδικοποίηση των δεδομένων πραγματοποιείται από τους κόμβους αισθητήρων και η αποκωδικοποίηση των πακέτων από τους αρκετά ισχυρότερους σταθμούς-βάσης. Οι τελευταίοι έχουν τη δυνατότητα να επικοινωνήσουν με τους κόμβους-αισθητήρες πραγματοποιώντας μετάδοση μεγαλύτερης ισχύος και άρα μπορούν να αντιμετωπίσουν καλύτερα την ύπαρξη θορύβου στο κανάλι επικοινωνίας ή τη μεγαλύτερη απόσταση μετάδοσης. Το σημαντικό όμως είναι ότι με αυτόν τον τρόπο, οι κόμβοι αισθητήρων με το απλοϊκό τους επεξεργαστικό σύστημα, δεν απαιτείται να εκτελέσουν αλγορίθμους αποκωδικοποίησης. Αυτή η ασύμμετρη επικοινωνία μεταξύ κόμβων αισθητήρων και σταθμών βάσης έχει αποδειχθεί στη βιβλιογραφία ότι μπορεί να προσφέρει σημαντική εξοικονόμηση ενέργειας.

Στην εργασία [9] υλοποιείται ένα σχήμα Turbo Coding σε έναν απλό κόμβο που αποστέλλει πακέτα δεδομένων. Ο αλγόριθμος του encoderυλοποιείται με λογισμικό στο μικροελεγκτή του κόμβου, αλλά και σε ειδικευμένο hardware (με χρήση επαναδιατασσόμενης λογικής) συνδεδεμένο στον κόμβο. Τα αποτελέσματα που προκύπτουν δείχνουν δραματική βελτίωση στην ενεργειακή απόδοση του κόμβου και την αξιοπιστία στη μετάδοση δεδομένων λόγω της χρήσης του αλγορίθμου.

Ο σκοπός αυτής της εργασίας είναι να επεκτείνει την προαναφερθείσα εργασία μελετώντας το σταθμό βάσης και άρα υλοποιώντας έναν αντίστοιχο αποκωδικοποιητή Turbo Coding. Στόχος είναι να μελετηθεί συνολικά η επίδραση του Turbo Coding στην ενεργειακή και χρονική απόδοση, να προσδιοριστεί το κόστος λειτουργίας ενός τέτοιου σχήματος και να εξαχθούν συμπεράσματα κατά πόσο η χρήση FEC μπορεί να επεκταθεί σε πιο πολύπλοκες τοπολογίες. Οι τοπολογίες αυτές περιλαμβάνουν τη σύνδεση πολλαπλών single-hop clusters (stars) σε ένα δίκτυο καθιστώντας έτσι το δίκτυο πρακτικά multi-hop, επεκτείνοντας έτσι τις δυνατότητες χρήσης του σε περισσότερες εφαρμογές (π.χ. εφαρμογές που απαιτούν μεγάλη γεωγραφική κάλυψη).

1.3 Η Εργασία Αυτή

Η σχεδίαση του αποκωδικοποιητή αναπτύχθηκε με βάση την υπόθεση ότι τα πακέτα δεδομένων αποκωδικοποιούνται σε έναν κόμβο-συλλέκτη, ο οποίος θα αποτελεί την κεφαλή ενός cluster, και θα έχει τη δυνατότητα να επικοινωνεί με κόμβους διαφορετικών clusters στο δίκτυο.

Η προσέγγιση που επιχειρήθηκε είναι η συνολική υλοποίηση Turbo Coding σε διάφορα επίπεδα. Αρχικά χρησιμοποιώντας τα συμπεράσματα της βιβλιογραφίας και έχοντας ως αφετηρία την εργασία [9], έγινε αναζήτηση για τον κατάλληλο αλγόριθμο αποκωδικοποίησης. Έχοντας ως δεδομένο τον αλγόριθμο κωδικοποίησης που είχε ήδη υλοποιηθεί, προτιμήθηκε ο αλγόριθμος Viterbi στη γενική του μορφή, καταλήγοντας σε δύο παραλλαγές του.

Αρχικά, υλοποιήθηκε ο αποκωδικοποιητής BCJR σε κώδικα Matlab, ο οποίος ενσωματώθηκε στη βιβλιοθήκη CML της Matlab ώστε να μελετηθεί η απόδοση του σε ένα κατάλληλο περιβάλλον προσομοίωσης. Τα αποτελέσματα που προέκυψαν ήταν αρχικά ικανοποιητικά, παρόλα αυτά τροποποιώντας τον αλγόριθμο για τις απαιτήσεις του συγκεκριμένου δικτύου ΑΔΑ παρατηρήθηκε μη αποδεκτή λειτουργία του. Έτσι εξετάστηκε μία αυτόνομη υλοποίηση του Max-Log MAP, η οποία ενσωματώθηκε επίσης στη CML. Μετά από μία σειρά προσομοιώσεων, και αφού η υλοποίηση συγκρίθηκε με την αντίστοιχη υλοποίηση του Max-Log MAP που υπάρχει ήδη στη CML, επιλέχθηκε ο συγκεκριμένος αλγόριθμος για τα επόμενα στάδια της εργασίας.

Η πλακέτα που επιλέχθηκε ως σταθμός βάσης είναι το BeagleBone. Η συγκεκριμένη συσκευή διαθέτει σημαντικότερα ανώτερο επεξεργαστή σε σχέση με το μικροελεγκτή του κόμβου-αισθητήρα, με αποτέλεσμα να μπορεί να αναλάβει με ασφάλεια το φόρτο της διαδικασίας αποκωδικοποίησης. Ταυτόχρονα, η πλακέτα επέτρεπε τη διασύνδεση με FPGA που θα μπορούσε να χρησιμοποιηθεί για να αναλάβει τη πραγματοποίηση της αποκωδικοποίησης.

Ο επιλεγμένος αλγόριθμος Max-Log MAP υλοποιήθηκε σε C και δοκιμάστηκε σε Η/Υ αλλά και στην πλακέτα. Τα αποτελέσματα κρίθηκαν ικανοποιητικά, μετά από σύγκριση με αυτά της προσομοίωσης. Το επόμενο στάδιο ήταν η υλοποίηση του αλγορίθμου σε αναδιατασσόμενη λογική. Επιλέχθηκε το κατάλληλο FPGA cape για το BeagleBone, και στη συνέχεια ο Max-Log MAP υλοποιήθηκε σε VHDL. Έτσι έγιναν μετρήσεις για τη χρονική και την ενεργειακή απόδοση του αλγορίθμου, αφενός σε επίπεδο λογισμικού, και αφετέρου με τη χρήση της FPGA. Τέλος, εξετάστηκαν τα αποτελέσματα των μετρήσεων, για τη διεξαγωγή συμπερασμάτων για όλα τα στάδια υλοποίησης του Turbo Code.

Στο Κεφάλαιο 2 παρουσιάζονται γενικά στοιχεία που αφορούν τα ΑΔΑ, ειδικότερα χαρακτηριστικά της αρχιτεκτονικής τους και αναλύονται οι τεχνικές δρομολόγησης δεδομένων καθώς και οι τοπολογίες τους. Στο Κεφάλαιο 3 περιγράφεται η λειτουργία του Forward Error Correction, οι τύποι FEC, καταλήγοντας στους TurboCodes. Στο

Κεφάλαιο 4 αναλύεται η λειτουργία του Convolutional/Turbo Coding που υιοθετήθηκε στη συγκεκριμένη εργασία και παρουσιάζονται οι decodersπου χρησιμοποιήθηκαν. Στο Κεφάλαιο 5 παρουσιάζεται η διαδικασία προσομοίωσης των decodersκαι τα αποτελέσματα που προέκυψαν.

2 Ασύρματα Δίκτυα Αισθητήρων

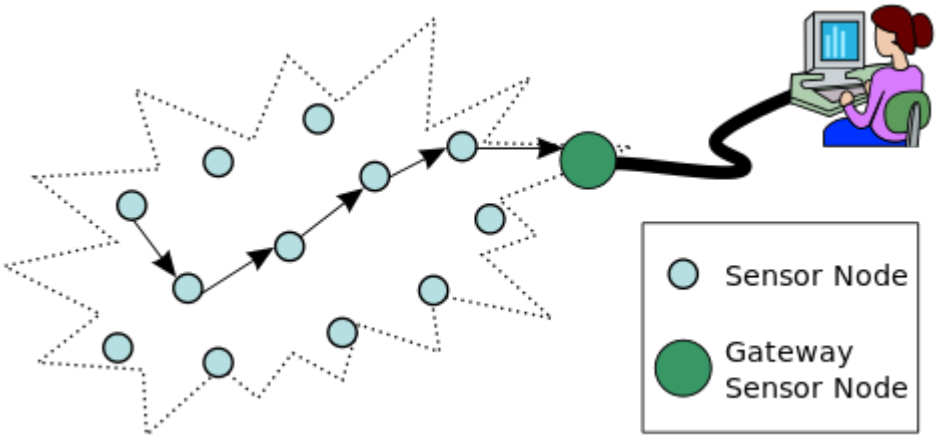
Η ανάγκη του ανθρώπου για παρατήρηση φαινομένων και καταγραφή συνθηκών σε εξωτερικούς ή εσωτερικούς χώρους οδήγησε στην αυτοματοποίηση της διαδικασίας λήψης μετρήσεων φυσικών μεγεθών. Για την βελτιστοποίηση της ακρίβειας στη λήψη τέτοιων μετρήσεων αναπτύχθηκαν ηλεκτρονικοί αισθητήρες που υποστηρίζονται από κατάλληλα ψηφιακά συστήματα. Επιπλέον, για την καταγραφή μεγάλου δείγματος τιμών σε διάφορες περιοχές, χρησιμοποιούνται πολλαπλοί αισθητήρες, τοποθετημένοι στα κατάλληλα σημεία. Η συγκέντρωση των δεδομένων από κάθε αισθητήρα ξεχωριστά θα ήταν μια χρονοβόρα ή και ανέφικτη διαδικασία, αν αναλογιστεί κανείς το πλήθος των αισθητήρων που απαιτούν διάφορες εφαρμογές. Για τον ίδιο λόγο, η ενσύρματη επικοινωνία τους θα είχε μεγάλο κόστος, ενώ σε πολλές περιπτώσεις θα ήταν πρακτικά αδύνατη. Έτσι, για να γίνει εφικτή η συγκέντρωση των δεδομένων που προκύπτουν, κρίθηκε απαραίτητη η ασύρματη επικοινωνία των αισθητήρων, με σκοπό τελικά τη συγκέντρωση της πληροφορίας σε έναν κεντρικό κόμβο. Μέσα από αυτή τη διαδικασία γεννήθηκαν τα Ασύρματα Δίκτυα Αισθητήρων (Wireless Sensor Networks - WSN).

2.1 Γενικά Στοιχεία για τα ΑΔΑ

2.1.1 Περιγραφή

Ένα Ασύρματο Δίκτυο Αισθητήρων (ΑΔΑ) αποτελείται από κόμβους που επικοινωνούν μεταξύ τους ασύρματα, όπου ο κάθε κόμβος αποτελεί έναν «έξυπνο» αισθητήρα. «Έξυπνος» αισθητήρας ονομάζεται μία συσκευή που αλληλοεπιδρά με το φυσικό κόσμο, μέσω ενός ηλεκτρονικού/μηχανικού αισθητήρα. Τα δεδομένα που προκύπτουν αποτελούν είσοδο σε ένα ηλεκτρονικό κύκλωμα που αναλαμβάνει την επεξεργασία τους. Τέλος, χρησιμοποιείται ένα ξεχωριστό ηλεκτρονικό κύκλωμα για να επικοινωνήσει με άλλες συσκευές-κόμβους, μεταδίδοντας τα δεδομένα.

Οι κόμβοι αυτοί είναι ικανοί όχι μόνο να συλλέξουν πληροφορίες όπως αυτές που αναφέρθηκαν παραπάνω, αλλά να τις μεταδώσουν και να τις παραλάβουν από άλλους αντίστοιχους κόμβους. Σε διάφορες περιπτώσεις μπορούν να τις επεξεργαστούν ή να τις αποθηκεύσουν. Λόγω των συνθηκών που επικρατούν στις περιοχές που τοποθετούνται οι αισθητήρες, προέκυψε η ανάγκη να είναι ενεργειακά αυτόνομοι. Έτσι, ο κάθε κόμβος έχει τη δική του αυτόνομη παροχή ρεύματος, πράγμα που διευκολύνει την τοποθέτηση του σε δυσπρόσιτα σημεία. Το πλήθος των κόμβων σε ένα δίκτυο μπορεί να είναι από δεκάδες έως και χιλιάδες, ανάλογα με τις ανάγκες του κάθε δικτύου, οι οποίες καθορίζουν την πυκνότητα τοποθέτησης των κόμβων. Ένας κόμβος μπορεί να συλλέγει δεδομένα και να τα αποστέλλει, να λαμβάνει δεδομένα για να τα αναμεταδώσει, ή και τα δύο. Η αρχιτεκτονική του δικτύου περιγράφει τις διεργασίες αυτές για κάθε κόμβο. Επιπλέον, ένας κόμβος μπορεί να έχει τη δυνατότητα να επεξεργάζεται τα δεδομένα, να τα μορφοποιεί, ή να συγκεντρώνει διάφορα δεδομένα για να τα αποστείλει στον επόμενο κόμβο.



Εικόνα 2.1: : Ένα τυπικό ΑΔΑ όπου τα δεδομένα καταλήγουν από έναν κόμβο στον χρήστη

2.1.2 Εφαρμογές

Τα Ασύρματα Δίκτυα Αισθητήρων χρησιμοποιούνται σε διάφορους τομείς της ανθρώπινης δραστηριότητας. Αρχικά χρησιμοποιήθηκαν για τη καταγραφή περιβαλλοντικών δεδομένων και στη βιομηχανία (βιομηχανικός έλεγχος). Σταδιακά όμως η χρήση τους επεκτείνεται καθώς ενσωματώνονται σε όλων των ειδών τις εφαρμογές από την ιατρική μέχρι τα έξυπνα σπίτια. Με τη βοήθεια του survey [8], παρουσιάζονται κάποιες από τις εφαρμογές των ΑΔΑ, ενδεικτικά:

- Περιβαλλοντικές Εφαρμογές

Σε ότι αφορά το περιβάλλον, τα ασύρματα δίκτυα αισθητήρων είναι ικανά να παρακολουθήσουν την ποιότητα του αέρα και των υδάτων, αλλά και να προβλέψουν φυσικές καταστροφές. Γενικώς η χρήση τους συνηθίζεται λόγω της ικανότητάς τους να τοποθετούνται σε μέρη δυσπρόσιτα και να ελέγχονται από το χρήστη από απόσταση.

- Εφαρμογές στη Γεωργία

Η λειτουργία ΑΔΑ στον κλάδο της γεωργίας συναντάται όλο και πιο συχνά τα τελευταία χρόνια. Οι βασικοί τομείς χρησιμοποίησης τους είναι η παρακολούθηση καλλιέργειας, η διαχείριση της άρδευσης και η εφαρμογή αυτοματισμών σε θερμοκήπια. Παρατηρούμε λοιπόν συνδυασμό συστημάτων αυτοματισμού και αισθητήρων που βοηθούν στην αυτοματοποίηση γεωργικών διεργασιών, και την πραγματοποίηση τους από απόσταση.

- Βιομηχανικές Εφαρμογές

Στον τομέα της βιομηχανίας, τα ΑΔΑ χρησιμοποιούνται σε διάφορα επίπεδα. Αρχικά, χρησιμοποιούνται ασύρματοι αισθητήρες για τη μείωση του κόστους καλωδίων. Ακόμη, υπάρχουν εφαρμογές ΑΔΑ σε κέντρα δεδομένων (data centers). Εκεί, η παρακολούθηση της θερμοκρασίας των server racks είναι πολύ σημαντική και βοηθά στην καλύτερη κατανομή του φόρτου εργασίας στο κέντρο, τη δρομολόγηση των δεδομένων και τη μακροζωία των μηχανημάτων. Όπως και στις περιβαλλοντικές εφαρμογές, έτσι και στη βιομηχανία, τα ΑΔΑ χρησιμοποιούνται για την παρακολούθηση υδάτων και αποβλήτων. Με αυτούς και άλλους τρόπους, τα ΑΔΑ συμβάλλουν στη βελτιστοποίηση της παραγωγικής διαδικασίας.

- Ιατρικές Εφαρμογές

Για ιατρικούς σκοπούς, έχουν αναπτυχθεί δίκτυα WBAN (Wireless Body Area Network), από αισθητήρες που τοποθετούνται στο σώμα και επικοινωνούν μεταξύ τους αποστέλλοντας μέσω ενός κόμβου-πύλη του δικτύου τις μετρήσεις που καταγράφουν στον ιατρό ή το νοσοκομείο του ασθενούς. Με τη μέθοδο αυτή μπορεί άμεσα να προβλεφθεί η επιδείνωση της κατάστασης ενός ασθενούς, καθώς και να ληφθούν αποφάσεις για την αντιμετώπιση της πάθησής του και για την φαρμακευτική του αγωγή.

- Έξυπνο Σπίτι

Ο συνδυασμός ασύρματων αισθητήρων και συστημάτων αυτοματισμού σε ένα οικιακό δίκτυο, συνθέτουν αυτό που ονομάζεται «έξυπνο σπίτι». Οι αισθητήρες επιβλέπουν της συνθήκες στο εσωτερικό και το εξωτερικό του σπιτιού και με τη χρήση των κατάλληλων ενεργοποιητών φροντίζουν για αυτές. Τέτοιου είδους εφαρμογές αποτελούν σύνηθες παράδειγμα διασύνδεσης ενός ασύρματου δικτύου αισθητήρων με το ίντερνετ. Με τη μέθοδο αυτή ο χρήστης μπορεί απομακρυσμένα να παρακολουθήσει τις συνθήκες που επικρατούν στο σπίτι του, καθώς και να τις ρυθμίσει σύμφωνα με τις ανάγκες του.

2.1.3 Αρχιτεκτονική Κόμβου

Σε αυτή την ενότητα θα παρουσιαστούν τα βασικά δομικά στοιχεία ενός ασύρματου κόμβου-αισθητήρα. Τα υποσυστήματα αυτά καθιστούν τον κόμβο ικανό να συλλέξει, να επεξεργαστεί, να αποθηκεύσει και να μεταδώσει τα επιθυμητά δεδομένα.

- Μονάδα Επεξεργασίας

Το υποσύστημα αυτό αποτελείται από μία αυτόνομη μονάδα, συνήθως μικροελεγκτή, και αν χρειάζεται επιπλέον περιφερειακές συσκευές. Η μονάδα επεξεργασίας περιλαμβάνει έναν μικροεπεξεργαστή, δομικά στοιχεία όπως μνήμες ROM ή FLASH και RAM, ψηφιακές εισόδους και εξόδους, και διάφορα κυκλώματα ανάλογα με το μοντέλο και την πολυπλοκότητα του συστήματος. Στο σύστημα αυτό μπορούν να συνδεθούν περιφερειακές συσκευές για να καλύψουν τις ανάγκες του δικτύου, όπως συσκευές εντοπισμού θέσης, εξωτερικές μνήμες για την αποθήκευση των δεδομένων κ.τ.λ.

- Αισθητήρας

Ο αισθητήρας είναι η συσκευή που πραγματοποιεί τη βασική λειτουργία του δικτύου, δηλαδή τη λήψη μετρήσεων από το φυσικό περιβάλλον. Αποτελείται από μία ηλεκτρονική διάταξη που μετατρέπει τα σήματα που δέχεται από το περιβάλλον σε ψηφιακά δεδομένα. Υπάρχουν διαθέσιμοι διάφοροι τύποι αισθητήρων, που αντιστοιχούν στο είδος της μέτρησης που λαμβάνεται σε ένα δίκτυο, αλλά και στις συνθήκες που επικρατούν στην εκάστοτε εφαρμογή.

- Μονάδα Επικοινωνίας

Το υποσύστημα αυτό αναλαμβάνει την ασύρματη επικοινωνία μεταξύ των κόμβων μέσω ραδιοσυχνοτήτων. Συνήθως, αποτελείται από έναν πομποδέκτη RF (radio-frequency) και μία κεραία. Αποτελεί το υποσύστημα με τη μεγαλύτερη ενεργειακή κατανάλωση, γι' αυτό και όπως θα δούμε στη συνέχεια η ελαχιστοποίηση των μεταδόσεων έχει ιδιαίτερη σημασία στη σχεδίαση ενός ΑΔΑ.

- Τροφοδοσία

Η μονάδα τροφοδοσίας σε ένα ΑΔΑ αποτελείται στις περισσότερες περιπτώσεις από μία ή περισσότερες μπαταρίες. Επιπλέον, ανάλογα με το περιβάλλον λειτουργίας του κόμβου, μπορούν να χρησιμοποιηθούν και εναλλακτικές μορφές ενέργειας (π.χ. Φωτοβολταϊκά) για να υποβοηθήσουν την κεντρική μονάδα τροφοδοσίας. Είναι προφανές ότι η επιλογή των μπαταριών γίνεται με βάση το μέγεθος του κόμβου, την κατανάλωση του και τη δυνατότητα συντήρησής του. Ο τρόπος που τροφοδοτούνται οι κόμβοι αποτελεί ένα ιδιαίτερο χαρακτηριστικό των ΑΔΑ, που σχετίζεται με το ζητούμενο της εργασίας. Επειδή η παροχή ρεύματος στις περισσότερες περιπτώσεις γίνεται με χρήση αποθηκευμένης ενέργειας, η παραμονή ενός κόμβου στο δίκτυο εξαρτάται άμεσα από τα αποθέματα της μπαταρίας του. Έτσι, μειώνοντας την κατανάλωση ενέργειας των ηλεκτρονικών διατάξεων, αυξάνεται η διάρκεια ζωής των κόμβων στο δίκτυο.

2.1.4 Χαρακτηριστικά

Η δομή των ΑΔΑ, δηλαδή η παρουσία πολλών κόμβων χαμηλής σχετικά πολυπλοκότητας που επικοινωνούν ασύρματα, παρουσιάζει διάφορα χαρακτηριστικά, βάσει των οποίων τα δίκτυα αυτά προτιμώνται για καταγραφή μετρήσεων σε διάφορες εφαρμογές. Τα χαρακτηριστικά λειτουργίας των Ασυρμάτων Δικτύων Αισθητήρων συνοψίζονται ως εξής:

- Χαμηλή Ενεργειακή Κατανάλωση

Για να μπορούν οι κόμβοι να λειτουργούν αυτόνομα, η τροφοδοσία τους καλύπτεται συνήθως από μπαταρίες. Αυτό προϋποθέτει ότι η ενεργειακή τους κατανάλωση κυμαίνεται σε χαμηλά επίπεδα, ώστε να μην επηρεάζεται η διάρκεια ζωής των κόμβων. Αυτό λαμβάνεται υπ' όψη στη σχεδίαση του δικτύου. Έτσι, γίνεται χρήση υλικού μειωμένων ενεργειακών απαιτήσεων και λογισμικού χαμηλής πολυπλοκότητας.

- Αυτονομία – Προσαρμοστικότητά

Οι κόμβοι έχουν προγραμματιστεί για ένα συγκεκριμένο πλήθος λειτουργιών, τις οποίες και υποστηρίζουν πολλές φορές ανεξάρτητα από την κατάσταση των υπόλοιπων κόμβων του δικτύου. Δηλαδή, ο κάθε κόμβος συνεχίζει να λειτουργεί ανεξάρτητα από το αν προστέθηκαν περισσότεροι κόμβοι στο δίκτυο, ενώ αν κάποιος κόμβος-παραλήπτης είναι εκτός λειτουργίας, ο αποστολέας μπορεί να επιλέξει έναν άλλο κόμβο για να αποστείλει τα δεδομένα. Ακόμη, υπάρχει η δυνατότητα προσαρμογής των χαρακτηριστικών λειτουργίας ενός κόμβου, ανεξάρτητα μάλιστα από τους υπόλοιπους. Για τις ανάγκες του δικτύου ένας κόμβος μπορεί να αλλάξει στοιχεία όπως η συχνότητα

δειγματοληψίας, η συχνότητα αποστολής μετρήσεων, κ.α. Όλα αυτά καθορίζονται από την αρχιτεκτονική του δικτύου. Όπως θα δούμε παρακάτω, η αυτονομία των κόμβων είναι σε κάποιες περιπτώσεις σχετική, αφού εξαρτάται από το ρόλο ενός κόμβου στην τοπολογία του δικτύου. Τέλος, ένα άλλο χρήσιμο χαρακτηριστικό τους είναι η δυνατότητα χρησιμοποίησης διαφορετικών κόμβων από πλευράς υλικού (διαφορετικά μοντέλα μικροελεγκτών, αισθητήρων κ.α.) στο ίδιο δίκτυο, με την προϋπόθεση ότι λειτουργούν με τα ίδια πρωτόκολλα επικοινωνίας και τυποποίησης των δεδομένων που προκύπτουν από τις μετρήσεις.

- Χαμηλή Πολυπλοκότητα – Δυνατότητα Plug & Play

Στους κόμβους ενός ΑΔΑ χρησιμοποιείται χαμηλής πολυπλοκότητας λογισμικό, τόσο σε επίπεδο λειτουργικού συστήματος ή firmware, όσο και σε υψηλότερα επίπεδα. Αυτό δίνει τη δυνατότητα άμεσης εκκίνησης ενός δικτύου και άμεσης προσαρμογής του σε νέα δεδομένα. Επίσης, υπάρχει η δυνατότητα εύκολης πραγματοποίησης αλλαγών όπως η προσθαφαίρεση κόμβων ή ακόμα και η αλλαγή τοπολογίας του δικτύου. Στη συνέχεια θα αναλυθούν οι διάφορες τοπολογίες που εφαρμόζονται, ανάλογα με τις συνθήκες που επικρατούν στην περιοχή τοποθέτησης του δικτύου αλλά και τον φόρτο των δεδομένων. Είναι λοιπόν σημαντικό οι κόμβοι του δικτύου να προσαρμόζονται άμεσα σε αλλαγές που αφορούν τη δρομολόγηση των πακέτων δεδομένων.

- Απομακρυσμένος Έλεγχος

Η ασύρματη λειτουργία του δικτύου παρέχει τη δυνατότητα απομακρυσμένου ελέγχου. Με αυτόν τον τρόπο η γενική επίβλεψη του δικτύου, ακόμα και η πραγματοποίηση αλλαγών για τη προσαρμογή του δικτύου και του τρόπου λειτουργίας του μπορεί να γίνει από απόσταση. Έτσι επιτυγχάνεται η χρήση τέτοιων δικτύων σε περιοχές με δυσμενείς εξωτερικές συνθήκες. Παρόλο που εν γένει το δίκτυο λειτουργεί χωρίς ανθρώπινη επίβλεψη, η φυσική παρουσία του διαχειριστή του δικτύου είναι απαραίτητη σε κάποιες περιπτώσεις. Είναι προφανές ότι εργασίες συντήρησης του δικτύου όπως αλλαγή μπαταριών ή αντικατάσταση κόμβων που είναι εκτός λειτουργίας αλλά και ριζικές αλλαγές στο στήσιμο και την τοπολογία του δικτύου δεν μπορούν να γίνουν από απόσταση.

- Απόδοση Δειγματοληψίας

Η παρουσία πληθώρας κόμβων, άρα και η δειγματοληψία από διάφορες περιοχές μειώνει την πιθανότητα λάθους στη λήψη μετρήσεων. Ανάλογα και με τις συνθήκες λειτουργίας του δικτύου, το εύρος του δείγματος μπορεί να είναι αρκετά μεγάλο, έτσι ώστε να οδηγεί σε ασφαλή συμπεράσματα. Μάλιστα, μέσω της δυνατότητας πυκνής διάταξης των κόμβων, επιτυγχάνεται η βελτίωση της απόδοσης του.

- Απόδοση Αποστολής – Λήψης Μετρήσεων

Το βασικό πρόβλημα στη λειτουργία των ΑΔΑ ήταν αρχικά η αξιοπιστία στην επικοινωνία των κόμβων. Σταδιακά αναπτύχθηκαν πρωτόκολλα για την αντιμετώπιση

του ζητήματος αυτού. Πιο συγκεκριμένα, η χρήση τεχνικών κωδικοποίησης για την αποστολή δεδομένων από ένα κόμβο σε έναν άλλο, μπορεί να πολλαπλασιάσει την απόδοση του, μειώνοντας κατακόρυφα την πιθανότητα λάθους στη λήψη ενός πακέτου από τον παραλήπτη. Φτάνοντας στην ουσία της συγκεκριμένης εργασίας, θα δούμε πως μέσω του ελέγχου λαθών σε ένα πακέτο, αποφεύγονται οι πολλαπλές αποστολές του. Αν αναλογιστεί κανείς την περιορισμένη απόδοση των κυκλωμάτων που υποστηρίζουν την ασύρματη επικοινωνία των κόμβων, και τις συνθήκες υψηλού θορύβου που εν δυνάμει λειτουργεί ένα τέτοιο δίκτυο, τέτοιες τεχνικές μπορούν να αποδειχθούν ευεργετικές για ένα Ασύρματο Δίκτυο Αισθητήρων.

2.2 Αρχιτεκτονική Ασύρματων Δικτύων Αισθητήρων

2.2.1 Χαρακτηριστικά Σχεδίασης

Τα χαρακτηριστικά ενός Ασύρματου Δικτύου Αισθητήρων διαμορφώνονται κατά τη σχεδίαση του από διάφορους παράγοντες. Κατά την εξέταση των παραγόντων αυτών γίνεται αντιληπτό ότι οι περισσότεροι αφορούν άμεσα ή έμμεσα την αξιοπιστία στην επικοινωνία των κόμβων ενός ΑΔΑ, και τον τρόπο με τον οποίο αυτή σχετίζεται με την ενεργειακή κατανάλωση. Η σημασία των χαρακτηριστικών αυτών έγκειται αφενός στις κατευθύνσεις που προσφέρουν για την ανάπτυξη ενός πρωτοκόλλου ή ενός αλγορίθμου για ένα ΑΔΑ και γενικότερα για τη σχεδίαση του. Αφετέρου, αποτελούν κριτήρια σύγκρισης διαφορετικών αρχιτεκτονικών. Αντλώντας πηγές από την εργασία [4] και το survey [8], τα χαρακτηριστικά της αρχιτεκτονικής ενός ΑΔΑ περιγράφονται παρακάτω:

- Περιβάλλον Τοποθέτησης

Οι κόμβοι ενός ασύρματου δικτύου αισθητήρων τοποθετούνται συνήθως σε μεγάλη πυκνότητα, πολύ κοντά ή στο εσωτερικό του φαινομένου που παρατηρούν. Έτσι, λειτουργούν χωρίς ανθρώπινη επίβλεψη σε περιβάλλοντα όπως το εσωτερικό μίας μηχανής, ο βυθός ή η επιφάνεια ενός ωκεανού, σε ένα μεγάλο κλειστό ή προσκολλημένοι σε ένα όχημα ή ένα ζωντανό οργανισμό, κ.α. Γίνεται εύκολα αντιληπτό λοιπόν ότι σε πολλές περιπτώσεις λειτουργούν υπό δυσμενείς συνθήκες. Αυτές μπορεί να περιλαμβάνουν υψηλή πίεση, ακραίες (υψηλές ή χαμηλές) θερμοκρασίες, συνεχής παρεμβολές, κίνηση υπό υψηλές ταχύτητες, κ.α. Αυτές οι συνθήκες μπορεί να αποτελέσουν καθοριστικό παράγοντα στη σχεδίαση και τη συντήρηση του δικτύου, σε ότι αφορά το υλικό, το λογισμικό, την τοπολογία, και άλλες παραμέτρους.

- Διαδικασία Τοποθέτησης των Κόμβων

Η διαδικασία κατά την οποία θα εγκατασταθούν οι κόμβοι ενός δικτύου, καθώς και οι φάσεις πριν και μετά την τοποθέτησή τους επηρεάζουν την σχεδίαση του. Η τοποθέτηση των κόμβων μπορεί να γίνει μαζικά, πράγμα που σημαίνει ότι η ακριβής τοποθεσία στην οποία θα εγκατασταθούν είναι σχετικά τυχαία. Εναλλακτικά, μπορεί να γίνει τοποθέτηση των κόμβων ένας προς έναν, διασφαλίζοντας την ακριβή τοποθεσία όπου αυτοί θα εγκατασταθούν. Η επιλογή αυτή εξαρτάται από το πλήθος των κόμβων, και στις δύο περιπτώσεις όμως ο τρόπος με τον οποίο θα εγκατασταθούν

διαφέρει ανά εφαρμογή (ρίψη από τον αέρα ή εκτόξευση για τη μία περίπτωση, τοποθέτηση από άνθρωπο ή αυτόματα για την άλλη, κ.α.). Την επιλογή της μεθόδου τοποθέτησης των κόμβων επηρεάζουν παράγοντες όπως:

- Κόστος εγκατάστασης
- Ελαχιστοποίηση της προετοιμασίας για την τοποθέτηση
- Αύξηση της ευελιξίας στην διάταξη
- Ενίσχυση της αυτο-οργάνωσης και της ανοχής σφαλμάτων

Μετά τη διαδικασία τοποθέτησης, είναι πιθανό να παρουσιαστούν αστοχίες στην τοποθέτηση, στην προσβασιμότητα, στην επάρκεια ενέργειας ή γενικότερα να παρουσιαστεί κάποια βλάβη. Για τον λόγο αυτό πολλές φορές εκτελείται επανατοποθέτηση των κόμβων του δικτύου. Η διαδικασία αυτή ενδέχεται να συνδυαστεί με αλλαγή της τοπολογίας του δικτύου. Έτσι, η σχεδίαση του δικτύου θα πρέπει να επιτρέπει τέτοιου είδους διαφοροποιήσεις στη δομή του, σε σχέση με την κατάσταση αρχικοποίησης. Μία παρόμοια διαδικασία με αυτή της επανατοποθέτησης είναι η προσθήκη επιπλέον κόμβων. Αντίστοιχα, ένα δίκτυο θα πρέπει να επιδέχεται αλλαγές στην τοπολογία του, όταν για λόγους βλάβης αρχικών κόμβων, ή αλλαγής των απαιτήσεων του δικτύου, προστίθενται νέοι κόμβοι.

- Δυνατότητα Κλιμάκωσης

Ο αριθμός των αισθητήρων σε ένα ασύρματο δίκτυο μπορεί να είναι της τάξης εκατοντάδων ή χιλιάδων, ενώ υπάρχουν εφαρμογές στις οποίες φτάνει τιμές εκατομμυρίων. Σε πολλές περιπτώσεις ένα δίκτυο αρχικοποιείται με έναν πλήθος αισθητήρων και στη συνέχεια αυτό αυξάνεται σταδιακά. Η σχεδίαση του δικτύου θα πρέπει να επιτρέπει την ταυτόχρονη λειτουργία ενός τέτοιου αριθμού αισθητήρων. Η δυνατότητα αυτή κλιμάκωσης του δικτύου αποτελεί δομικό στοιχείο της αρχιτεκτονικής του. Κατά τη διαδικασία κλιμάκωσης θα πρέπει να αξιοποιείται από το δίκτυο η πυκνότητα τοποθέτησης των κόμβων του, στοιχείο που χαρακτηρίζει τα ασύρματα δίκτυα αισθητήρων. Η δυνατότητα κλιμάκωσης του δικτύου σχετίζεται με το είδος της εφαρμογής στην οποία χρησιμοποιείται.

- Περιορισμοί Υλικού

Όπως είδαμε στην ενότητα 2.1.3, οι βασικές μονάδες ενός κόμβου είναι ο αισθητήρας, η μονάδα επεξεργασίας, η μονάδα επικοινωνίας και η τροφοδοσία. Εκτός αυτών διάφορες άλλες υπομονάδες μπορούν να ενσωματωθούν στους κόμβους, αναλόγως την εφαρμογή στην οποία χρησιμοποιούνται. Το είδος και το μοντέλο του υλικού που θα χρησιμοποιηθεί για την κάθε μονάδα του συστήματος καθορίζεται από τους εξής παράγοντες:

- Ενεργειακή κατανάλωση
- Λειτουργία σε συνθήκες υψηλής ογκομετρικής πυκνότητας
- Χαμηλό κόστος παραγωγής και δυνατότητα αντικατάστασης
- Αυτονομία και λειτουργία χωρίς επίβλεψη
- Προσαρμογή στο περιβάλλον

Παρατηρείται ότι οι περισσότεροι παράγοντες σχετίζονται άμεσα με τη βελτιστοποίηση της αξιοπιστίας στην επικοινωνία του δικτύου, με σκοπό την μείωση

της ενεργειακής κατανάλωσης. Όπως θα δούμε στη συνέχεια, η επιλογή υλικών με χαμηλή ενεργειακή κατανάλωση, ιδιαίτερα σε ότι αφορά τη μονάδα επικοινωνίας μπορεί να γίνει χωρίς αυτό να επηρεάζει την αξιοπιστία στην επικοινωνία, αν εφαρμοστούν τεχνικές κωδικοποίησης των δεδομένων. Επίσης, οι τεχνικές αυτές βοηθούν στην αντιμετώπιση παρεμβολών, στην περίπτωση που οι κόμβοι είναι πυκνά τοποθετημένοι σε μία περιοχή. Για τη διαδικασία κωδικοποίησης ενδέχεται να χρησιμοποιηθούν ενσωματωμένες μονάδες επεξεργασίας, οι οποίες αυξάνουν το κόστος παραγωγής. Από την άλλη πλευρά, λόγω της κωδικοποίησης αποφεύγονται ισχυρότερες μονάδες επικοινωνίας που εκτός από υψηλή κατανάλωση ενεργειακών πόρων όπως είδαμε προηγουμένως, σημαίνουν επιπλέον αύξηση του κόστους παραγωγής.

- Κόστος Παραγωγής

Λόγω του μεγάλου αριθμού κόμβου που μπορεί να χρησιμοποιηθεί σε ένα ΑΔΑ, γίνεται εύκολα αντιληπτό ότι το κόστος του κάθε κόμβου αποτελεί τον βασικό γνώμονα για το συνολικό κόστος παραγωγής του δικτύου. Αν το κόστος αγοράς και λειτουργίας των ασύρματων κόμβων – αισθητήρων είναι υψηλότερο από ένα επίπεδο, η επιλογή ανάπτυξης ενός ΑΔΑ δεν μπορεί να προτιμηθεί από την ανάπτυξη ενός παραδοσιακού συστήματος αισθητήρων. Για το λόγο αυτό το κόστος του κάθε κόμβου αισθητήρα θα πρέπει να παραμένει χαμηλό. Αυτό βέβαια δεν εξαρτάται μόνο από το κόστος της βασικής μονάδας του αισθητήρα, αφού σε πολλές περιπτώσεις ενσωματώνονται επιπλέον μονάδες. Αναφορικά, μία τέτοια μονάδα μπορεί να είναι: ένα σύστημα προσδιορισμού θέσης (π.χ. GPS), ένα σύστημα κίνησης (π.χ. σερβομηχανισμός), ή μία ενσωματωμένη πλακέτα επεξεργασίας (όπως στη συγκεκριμένη εργασία που θα αναλυθεί η χρήση ενσωματωμένης FPGA για την αποκωδικοποίηση πακέτων δεδομένων). Για το λόγο αυτό, ο καθορισμός τους κόστους σε συνάρτηση με τις υπηρεσίες που προσφέρει ένα ΑΔΑ αποτελεί ένα ιδιαίτερο ζήτημα κατά τη σχεδίαση του.

- Ανοχή Σφαλμάτων

Σε ένα δίκτυο, κάποιοι κόμβοι μπορούν να βγουν εκτός λειτουργίας, ή να είναι αποκλεισμένοι από το υπόλοιπο δίκτυο. Γι' αυτό μπορεί να ευθύνονται φαινόμενα όπως η ανεπάρκεια ενεργειακών πόρων, η παρουσία έντονων παρεμβολών, φυσικές βλάβες στο υλικό των κόμβων κ.α. Ο αριθμός των κόμβων που βρίσκονται εκτός δικτύου και η συχνότητα με την οποία συμβαίνει αυτό καθορίζουν την αξιοπιστία ή αλλιώς ανοχή σφαλμάτων του δικτύου. Δηλαδή, ανοχή σφαλμάτων είναι η δυνατότητα ενός δικτύου να διατηρεί τη λειτουργικότητά του παρά τις διακοπές που προκαλεί η αναστολή λειτουργίας κάποιων κόμβων. Η χρήση συγκεκριμένων πρωτοκόλλων και η ανάπτυξη αλγορίθμων (όπως αυτοί που θα εξεταστούν σε αυτή την εργασία) μπορούν να συμβάλλουν στην ανοχή ενός δικτύου σε σφάλματα. Αυτές οι βελτιώσεις σε επίπεδο λογισμικού σχετίζονται άμεσα με το περιβάλλον στο οποίο λειτουργεί ένα δίκτυο. Όσο πιο έντονη είναι η αρνητική επίδραση του περιβάλλοντος στη λειτουργικότητα ενός δικτύου, τόσο μεγαλύτερη είναι η ανάγκη ανάπτυξης μεθόδων βελτίωσης της αξιοπιστίας του. Συμπερασματικά, το επίπεδο ανοχής σφαλμάτων σχετίζεται άμεσα με την εφαρμογή στην οποία χρησιμοποιείται ένα ασύρματο δίκτυο αισθητήρων, άρα και το περιβάλλον στο οποίο λειτουργεί.

- Ενεργειακή Κατανάλωση

Αναλύοντας διάφορα χαρακτηριστικά της σχεδίασης του (π.χ. ανοχή σφαλμάτων, περιορισμοί υλικού), γίνεται εύκολα αντιληπτή η επίδραση της ενεργειακής κατανάλωσης ενός κόμβου στη διάρκεια ζωής του. Ένας ασύρματος κόμβος – αισθητήρας, ως μικροηλεκτρονική συσκευή, λειτουργεί με περιορισμένη τροφοδοσία (<0,5 Ah, 1.2V). Σε πολλές περιπτώσεις η χρήση μπαταρίας ως πηγή ηλεκτρικού ρεύματος σημαίνει περιορισμένη διάρκεια ζωής, στην περίπτωση που η μπαταρία δεν δύναται να αντικατασταθεί. Για το λόγο αυτό, αναπτύσσονται συγκεκριμένα πρωτόκολλα και αλγόριθμοι με σκοπό τη μείωση της ενεργειακής κατανάλωσης των κόμβων, όπως οι αλγόριθμοι κωδικοποίησης που εξετάζονται στην εργασία αυτή. Με τον τρόπο αυτό αυξάνεται η διάρκεια ζωής του δικτύου συνολικά, ενώ όπως θα δούμε, σε συγκεκριμένες περιπτώσεις βελτιώνονται οι υπηρεσίες που παρέχει το δίκτυο.

Σε ό,τι αφορά άλλα δίκτυα τηλεπικοινωνιών, η ενεργειακή κατανάλωση είναι ένας σημαντικός παράγοντας στη σχεδίασή τους, αλλά όχι ο πρωταρχικός. Η ποιότητα των υπηρεσιών που προσφέρει ένα δίκτυο αποτελεί συνήθως μεγαλύτερη προτεραιότητα, επειδή οι ενεργειακοί πόροι παρέχονται άμεσα από τον χρήστη. Στην περίπτωση των ΑΔΑ όμως, η λειτουργία τους χωρίς την άμεση επίβλεψη από το χρήστη, άρα και η χρήση αποθηκευμένης ενέργειας που χρήζει αναπλήρωσης είναι το ιδιαίτερο στοιχείο που κάνει την ενεργειακή κατανάλωση τόσο σημαντική για την λειτουργικότητα του δικτύου.

Οι βασικές διεργασίες ενός κόμβου είναι η λήψη μετρήσεων από τον αισθητήρα, η επεξεργασία των τοπικών δεδομένων που προκύπτουν, και η μετάδοσή τους. Έτσι, οι ενεργειακοί πόροι καταναλώνονται στους αντίστοιχους τομείς: την πραγματοποίηση μετρήσεων, την επικοινωνία και την επεξεργασία. Το είδος των αισθητήρων που μπορεί να χρησιμοποιηθούν σε μία εφαρμογή, και η αντίστοιχη ενεργειακή κατανάλωση, δεν αφορά το αντικείμενο της εργασίας. Σε γενικές γραμμές, ο σκοπός της κάθε εφαρμογής καθορίζει τους αισθητήρες που χρησιμοποιούνται. Στην περίπτωση που υπάρχει επιλογή μεταξύ διαφορετικών μοντέλων, αυτή γίνεται κατά τη σχεδίαση του δικτύου, λαμβάνοντας υπ' όψη τους παράγοντες που αναφέρθηκαν στους περιορισμούς υλικού. Οι τομείς της επικοινωνίας και της επεξεργασίας των δεδομένων και ο τρόπος με τον οποίο αυτοί σχετίζονται με την ενεργειακή κατανάλωση του κόμβου, περιγράφονται παρακάτω.

- Επικοινωνία και Μέσα Μετάδοσης

Ο πιο δαπανηρός σε ενέργεια από τους τρεις τομείς είναι αυτός της επικοινωνίας. Μπορεί να αποδειχθεί ότι κατά την επικοινωνία μεταξύ μικρών αποστάσεων, με χαμηλά επίπεδα ακτινοβολίας το ενεργειακό κόστος αποστολής και λήψης είναι σχεδόν το ίδιο. Τα στοιχεία του κυκλώματος ενός πομποδέκτη (ενισχυτές ισχύος, oscillators, PLLs κλπ.) καταναλώνουν πολύτιμη ενέργεια. Κατά τον προσδιορισμό της ενεργειακής κατανάλωσης στον τομέα της επικοινωνίας, είναι σημαντικό να συνυπολογιστεί όχι μόνο η κατανάλωση κατά της διάρκεια της λειτουργίας των κυκλωμάτων ενός πομποδέκτη, αλλά και κατά τη διαδικασία εκκίνησης τους. Μπορεί η εκκίνηση να διαρκεί κάποιες εκατοντάδες μs, το διάστημα αυτό όμως αποτελεί ένα σημαντικό ποσοστό τους συνολικού χρόνου αποστολής ή λήψης ενός πακέτου δεδομένων. Έτσι, η ενέργεια που καταναλώνεται κατά τη διαδικασία εκκίνησης είναι ανάλογη του πλήθους των πακέτων στα οποία θα χωριστεί ο όγκος των δεδομένων προς αποστολή. Ταυτόχρονα, η αξιοπιστία της επικοινωνίας επηρεάζεται από το μέγεθος των πακέτων. Αυτό συμβαίνει επειδή με δεδομένη την πιθανότητα λάθους στη μετάδοση ενός πακέτου, ο όγκος των δεδομένων που αποτυγχάνουν να μεταδοθούν

είναι ανάλογος του μεγέθους του πακέτου δεδομένων. Αντίστοιχα, το ενεργειακό κόστος μετάδοσης ενός πακέτου είναι επίσης ανάλογο του μεγέθους του. Συμπερασματικά, το πλήθος και το μέγεθος των πακέτων, επηρεάζει άμεσα την ενεργειακή κατανάλωση της επικοινωνίας. Αργότερα, το ζήτημα αυτό θα αναλυθεί εκτενέστερα σε τεχνικό επίπεδο.

Η ασύρματη επικοινωνία των κόμβων του δικτύου περιλαμβάνει μία σειρά από επιλογές. Οι βασικοί τρόποι ασύρματης μετάδοσης των δεδομένων είναι οι ραδιοσυχνότητες, οι υπέρυθρες ακτίνες και τα οπτικά μέσα. Ο συνηθέστερος τρόπος από τους τρεις είναι η ψηφιακή μετάδοση στο πεδίο των ραδιοσυχνοτήτων. Για τα ΑΔΑ επιλέγονται υλικά με κύρια χαρακτηριστικά το χαμηλό κόστος, το μικρό μέγεθος και τη χαμηλή ενεργειακή κατανάλωση. Η επιλογή του πομποδέκτη και των συχνοτήτων στις οποίες θα λειτουργεί γίνεται βρίσκοντας τη βέλτιστη ισορροπία μεταξύ της απόδοσης της κεραίας και της ενεργειακής του κατανάλωσης.

Η επιλογή του μέσου μετάδοσης παίζει ιδιαίτερο ρόλο στη σχεδίαση του δικτύου. Όπως διαπιστώθηκε και προηγουμένως, η επικοινωνία κυριαρχεί σε μεγάλο βαθμό την συνολική ενεργειακή κατανάλωση του κόμβου. Ακόμη, το μέσο μετάδοσης επηρεάζει τη λειτουργία των υπόλοιπων δομικών στοιχείων του κόμβου. Ο τρόπος με τον οποίο θα πρέπει να διαμορφωθούν τα δεδομένα για να μεταδοθούν πρέπει να υποστηρίζεται από το υλικό και το λογισμικό του κόμβου. Τελικά, το μέσο μετάδοσης των δεδομένων θα πρέπει να εναρμονίζεται με τη συνολική λειτουργία του κόμβου.

- Επεξεργασία δεδομένων

Το ενεργειακό κόστος επεξεργασίας των δεδομένων είναι σημαντικά μικρότερο από το ενεργειακό κόστος μετάδοσης τους. Όμως, η επεξεργασία των δεδομένων μπορεί να οδηγήσει σε μείωση των ενεργειακών δαπανών κατά την επικοινωνία. Όπως θα δούμε στη συνέχεια, χρησιμοποιώντας την επεξεργαστική ισχύ ενός κόμβου για τη κατάλληλη διαμόρφωση των δεδομένων, μπορεί να αποφευχθεί το ενεργειακό κόστος πολλαπλών αποστολών του ίδιου πακέτου.. Για να είναι αυτή η διαδικασία αποδοτική, θα πρέπει το ενεργειακό κόστος της επιπλέον επεξεργασίας των δεδομένων να μην υπερβαίνει το ενεργειακό όφελος που εξοικονομήθηκε από την επικοινωνία. Στην εργασία αυτή θα εξεταστεί αυτή ακριβώς η διαδικασία. Μάλιστα, η επιπλέον επεξεργασία των δεδομένων και η ενεργειακή της δαπάνη θα εξεταστεί όχι μόνο για την περίπτωση που συμβαίνει στον επεξεργαστή ενός κόμβου, αλλά και για την περίπτωση χρήσης ενσωματωμένου συστήματος, που αναλαμβάνει την επιπλέον επεξεργασία των δεδομένων (συγκεκριμένα την κωδικοποίηση /αποκωδικοποίηση τους) αποκλειστικά.

2.2.2 Δομικά Στοιχεία

Τα δομικά στοιχεία ενός Ασύρματου Δικτύου Αισθητήρων αποτελούν οι συσκευές που αναλαμβάνουν τη συλλογή πληροφοριών και τη μετάδοσή τους (αποστολή – λήψη) τις οποίες θεωρούμε ως κόμβους αλλά και μία κεντρική συσκευή που αναλαμβάνει τη συγκέντρωση όλων των πληροφοριών του δικτύου με σκοπό τη μετάδοση τους στο χρήστη ή σε ένα ευρύτερο δίκτυο. Και οι δύο τύποι συσκευών χωρίζονται σε υποκατηγορίες, ανάλογα με τον τρόπο που μεταδίδουν τα δεδομένα. Έτσι ανάλογα με τη δρομολόγηση και την τοπολογία του δικτύου, οι τύποι συσκευών διαφέρουν ανά περίπτωση.

Οι κόμβοι του δικτύου αποτελούνται συνήθως από μικροηλεκτρονικά συστήματα όπου με την προσθήκη αισθητήρων αποτελούν τους έξυπνους αισθητήρες (smart sensors). Τέτοιο κόμβοι (sensor nodes ή motes) χαρακτηρίζονται διαφορετικά, ανάλογα με το σκοπό που εξυπηρετούν στο δίκτυο. Αρχικά, υπάρχουν οι κόμβοι γενικού σκοπού που συλλέγουν μετρήσεις και τις μεταδίδουν. Επειδή σε αυτούς τους κόμβους δημιουργούνται ουσιαστικά τα δεδομένα μετρήσεων, τους αποκαλούμε πηγές (source nodes). Ύστερα, υπάρχουν οι κόμβοι-αναμεταδότες (routers), που έχουν ως αποκλειστική λειτουργία την λήψη και αποστολή δεδομένων σε άλλους κόμβους. Κλείνοντας, οι κόμβοι-πύλες (gateways) διαβιβάζουν τα δεδομένα ενός δρομολογημένου μονοπατιού του δικτύου στο σταθμό-βάση.

Σταθμός βάσης αποκαλείται η συσκευή που αναλαμβάνει να αποστείλει τα συγκεντρωτικά δεδομένα σε έναν ή περισσότερους χρήστες. Γι' αυτό και συχνά αποκαλείται κόμβος-συλλέκτης (sink node). Ένας τέτοιος συλλέκτης μπορεί να αποτελεί έναν από τους κόμβους του δικτύου, αλλά μπορεί και να βρίσκεται εκτός αυτού. Στη δεύτερη περίπτωση, ο κόμβος συλλέκτης είναι ένας υπολογιστής με δύο ενδεχόμενες λειτουργίες. Αφενός, τη λειτουργία συλλογής των δεδομένων και παρουσίασής τους στο χρήστη, και αφετέρου, την ιδιότητα του ενδιάμεσου κόμβου (gateway) μεταξύ του Ασυρμάτου Δικτύου Αισθητήρων, και ενός μεγαλύτερου δικτύου (π.χ. Internet).

2.2.3 Δρομολόγηση

Σε ό,τι αφορά τη δρομολόγηση των πακέτων δεδομένων σε ένα ΑΔΑ, υπάρχουν δύο βασικοί τύποι. Ο πρώτος αφορά την αποστολή πακέτων δεδομένων απευθείας από ένα κόμβο-πηγή στον κόμβο-συλλέκτη. Η τύπος δρομολόγησης αυτός ονομάζεται one-hop ή single-hop αφού η διαδρομή του κάθε πακέτου γίνεται σε μία διαδρομή (άλμα) από σημείο σε σημείο. Ο δεύτερος τύπος αφορά την αποστολή των πακέτων μέσω διάφορων διαδρομών (πολλαπλά άλματα) από κόμβο σε κόμβο μέχρι να καταλήξουν στον κόμβο-συλλέκτη. Για τον λόγο αυτό, ο συγκεκριμένος τύπος δρομολόγησης ονομάζεται multi-hop. Πιο αναλυτικά:

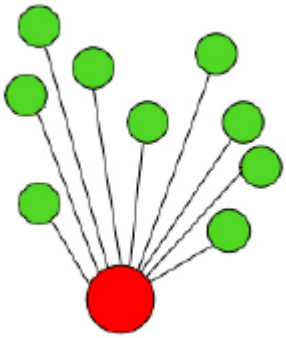
- Single-hop

Στη δρομολόγηση single-hop, ένας κόμβος-πηγή αποστέλλει τα δεδομένα που κατέγραψε απευθείας στον κόμβο-πύλη. Αυτό σημαίνει ότι ο κόμβος-πηγή καταγράφει μετρήσεις από τον αισθητήρα του, τις μορφοποιεί και σχηματίζει έτσι πακέτα δεδομένων. Από τη στιγμή που ένα πακέτο έχει δημιουργηθεί, είναι έτοιμο προς αποστολή.

Ύστερα, ο κόμβος-πηγή μπορεί να μπει σε κατάσταση αδράνειας, μέχρι την επόμενη χρονική στιγμή που θα χρειαστεί να καταγράψει μία μέτρηση για να την αποστείλει. Αυτό δεν ισχύει όμως για τον κόμβο-πύλη. Η χρονική στιγμή που ένα πακέτο θα μεταδοθεί στον κόμβο-πύλη είναι δύσκολο να προβλεφθεί, αφού εξαρτάται από διάφορους παράγοντες. Ενδεικτικά, η απόσταση του κόμβου-πηγή από τον κόμβο-πύλη, και ο θόρυβος στο κανάλι επικοινωνίας επηρεάζουν την ταχύτητα αποστολής. Επιπλέον, η συχνότητα λήψης μετρήσεων δεν είναι πάντα η ίδια για όλους τους κόμβους-πηγές ή ενδέχεται να μην είναι σταθερή, αφού μπορεί να εξαρτάται από

κάποια αλλαγή στο φυσικό περιβάλλον. Για τους λόγους αυτούς, ο κόμβος-πύλη θα πρέπει να βρίσκεται πάντα σε κατάσταση ετοιμότητας, για να παραλάβει ένα πακέτο δεδομένων.

Λόγω των μονών αλμάτων με τα οποία μεταδίδονται τα δεδομένα, υπάρχει ένας σημαντικός περιορισμός στην τοποθέτηση των κόμβων, αυτός της απόστασης από τον κόμβο-πύλη. Όταν ένας κόμβος-πηγή τοποθετηθεί σε μεγάλη απόσταση από τον κόμβο-πύλη, η ενέργεια που θα δαπανηθεί για τη μετάδοση των πακέτων αυξάνεται σημαντικά, αφού δεν υπάρχουν ενδιάμεσοι κόμβοι αναμετάδοσης. Αυτό μάλιστα συμβαίνει τόσο στον κόμβο-αποστολέα, όσο και στον κόμβο-παραλήπτη. Επιπλέον, λόγω των δυσμενών συνθηκών επικοινωνίας που επικρατούν πολλές φορές στα ΑΔΑ, αλλά και των χαμηλής ισχύος συστημάτων επικοινωνίας, η πιθανότητα ανεπιτυχούς αποστολής ενός πακέτου (ολοκληρωτική αποτυχία αποστολής ή αποστολής με λανθασμένα bits)αυξάνεται δραματικά. Καταλήγοντας, η χρήση ενός κόμβου-συλλέκτη και πολλών κόμβων-πηγών που αποστέλλουν πακέτα με μονά άλματα κρίνεται μη αποδοτική για μεγάλης έκτασης περιοχές τοποθέτησης.



Εικόνα 2.1: Single-hop δρομολόγηση. Ο κόμβος-συλλέκτης διακρίνεται με κόκκινο χρώμα, και οι κόμβοι-πηγές με πράσινο

- Multi-hop

Με αυτόν τον τύπο δρομολόγησης, τα πακέτα δεδομένων ακολουθούν ένα μονοπάτι από ενδιάμεσους κόμβους μέσω πολλαπλών αλμάτων, μέχρι να καταλήξουν από τον κόμβο-πηγή στον κόμβο-πύλη. Οι ενδιάμεσοι κόμβοι μπορεί να είναι είτε επίσης κόμβοι-πηγές, είτε κόμβοι-αναμεταδότες.

Η δρομολόγηση με πολλαπλά άλματα αναπτύχθηκε για να επιλύσει το ζήτημα της περιορισμένης απόστασης μεταξύ πηγής και πύλης/συλλέκτη. Έτσι, τα πακέτα δεδομένων μπορούν να αναμεταδίδονται στους ενδιάμεσους κόμβους, οι οποίοι τοποθετούνται σε κοντινές σχετικά αποστάσεις ο ένας από τον άλλο.

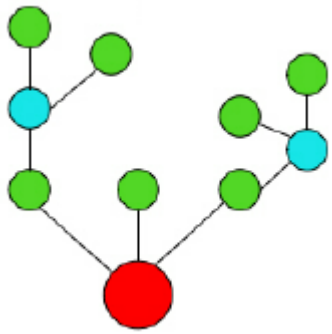
Ο συγκεκριμένος τύπος δρομολόγησης είναι σαφώς πιο πολύπλοκος από τη δρομολόγηση single-hop. Όπως θα αναλυθεί και στην ενότητα 2.2.4, ανάλογα με την τοπολογία του κάθε δικτύου, τα πακέτα δεδομένων ενδέχεται είτε να ακολουθούν ένα προκαθορισμένο μονοπάτι από ένα συγκεκριμένο κόμβο-πηγή προς τον κόμβο-συλλέκτη, είτε να έχουν στη διάθεσή τους ένα πλήθος πιθανών μονοπατιών για να ακολουθήσουν. Γίνεται αντιληπτό ότι σε κάθε περίπτωση τα πακέτα δεδομένων επιβαρύνονται με επιπλέον πληροφορία που σχετίζεται με τη δρομολόγηση τους. Αυτή

η αύξηση του όγκου των δεδομένων συνεπάγεται ενεργειακές δαπάνες τόσο στον τομέα της επεξεργασίας των δεδομένων, όσο και στον τομέα της επικοινωνίας.

Ένα άλλο ζήτημα που προκύπτει κατά την multi-hopδρομολόγηση είναι το κατά πόσο οι κόμβοι του δικτύου έχουν τη δυνατότητα να παραμένουν σε κατάσταση αδράνειας. Σε αντίθεση με τη δρομολόγηση single-hop, οι κόμβοι-πηγές αναλαμβάνουν εκτός από τη λήψη μετρήσεων, την αναμετάδοση δεδομένων, εκτός φυσικά από τους κόμβους-πηγές που βρίσκονται στην αρχή του κάθε μονοπατιού. Αυτό σημαίνει ότι όσο πιο κοντά βρίσκονται στον κόμβο-συλλέκτη, τόσο μεγαλύτερο φόρτο εργασίας αναλαμβάνουν να διεκπεραιώσουν. Το ίδιο ισχύει φυσικά και για τους κόμβους-αναμεταδότες.

Επιπλέον, ακόμα και όταν οι ενδιάμεσοι κόμβοι δεν λαμβάνουν ή αποστέλλουν δεδομένα, θα πρέπει να βρίσκονται σε ετοιμότητα για κάποια νέα λήψη από ένα γειτονικό κόμβο. Έτσι, ένα μέρος των ενεργειακών πόρων των κόμβων αυτών αφιερώνεται στη διαδικασία αυτή, χωρίς αυτοί να εκτελούν κάποια από τις βασικές τους λειτουργίες (καταγραφή μετρήσεων, λήψη, αποστολή δεδομένων).

Από τα παραπάνω γίνεται αντιληπτό ότι με τη multi-hopδρομολόγηση επιλύονται κάποια από τα προβλήματα της single-hopδρομολόγησης, προκύπτουν όμως νέα ζητήματα που αφορούν την ενεργειακή κατανάλωση. Για την κάθε εφαρμογή, ο σχεδιαστής καλείται να διαμορφώσει την τοπολογία του δικτύου έτσι ώστε να αξιοποιούνται στο μέγιστο βαθμό τα πλεονεκτήματα του κάθε τύπου δρομολόγησης.



Εικόνα 2.2: Multi-hop δρομολόγηση. Ο κόμβος-συλλέκτης διακρίνεται με κόκκινο χρώμα, οι κόμβοι-πηγές με πράσινο, και οι κόμβοι-αναμεταδότες με γαλάζιο

- Σύγκριση Short-hops, Long-hops

Η δρομολόγηση multi-hop, λόγω των πολλαπλών αλμάτων εφαρμόζεται σε διάφορες τοπολογίες, με τα αντίστοιχα χαρακτηριστικά, αναλόγως την αρχιτεκτονική του δικτύου. Οι τοπολογίες αυτές εξετάζονται στην επόμενη ενότητα. Ακόμη, υπάρχουν διαφοροποιήσεις μεταξύ των τοπολογιών, που σχετίζονται με το μήκος των αλμάτων με τα οποία μεταδίδονται τα πακέτα δεδομένων. Δηλαδή, σχεδιάζοντας το δίκτυο ενδέχεται να επιλεχθούν δρομολόγια που θα περιέχουν πολλά μικρά άλματα από κόμβο σε κόμβο (short-hops) είτε λιγότερα άλματα που θα διανύουν μεγαλύτερες αποστάσεις (longhops). Αυτό σχετίζεται προφανώς με την πυκνότητα τοποθέτησης

αλλά και το πλήθος των κόμβων σε ένα δίκτυο. Η διαφοροποίηση αυτή στις αποστάσεις των αλμάτων μπορεί να γίνει και στη δρομολόγηση single-hop, όταν υπάρχει η δυνατότητα επιλογής της απόστασης που τοποθετούνται οι κόμβοι-πηγές από τον κόμβο συλλέκτη. Για μία ρεαλιστική προσέγγιση όμως, γίνεται η παραδοχή ότι η δρομολόγηση single-hop θα περιλαμβάνει longhops στις περισσότερες εφαρμογές.

Όπως αναλύθηκε προηγουμένως, το βασικό μειονέκτημα της single-hopδρομολόγησης είναι η περιορισμένη απόσταση μεταξύ των κόμβων. Αντίστοιχα, σε μία τοπολογία με δρομολόγηση multi-hop, η μετάδοση δεδομένων μεταξύ ενδιάμεσων κόμβων που βρίσκονται σε μεγάλη απόσταση ο ένας από τον άλλο (longhops) είναι πιο δαπανηρή σε ενέργεια από αυτή μεταξύ κόμβων τοποθετημένων σε μικρότερη απόσταση (short-hops). Έτσι λοιπόν, επικρατεί μια γενική αντίληψη στη σχεδίαση των ΑΔΑ, ότι η μετάδοση με longhopsείναι πιο αποδοτική απ' ότι με short-hops, και γι' αυτό η δρομολόγηση single-hopσπάνια είναι πιο αποδοτική της multi-hop.

Θεωρητικά , σε ένα απλοποιημένο σενάριο, η μετάδοση με ένα longhopείναι πιο δαπανηρή απ' ότι με πολλά short-hops. Στην εργασία[10] και στο άρθρο [11] αναπτύσσεται μία ισχυρή επιχειρηματολογία που ανατρέπει την αντίληψη αυτή. Παρουσιάζονται έτσι μία σειρά από λόγους για τους οποίους η χρήση longhops είναι αποδοτικότερη των πολλών short-hops σε πολλές περιπτώσεις. Αυτό που συνήθως δεν λαμβάνεται υπ' όψη στη σύγκριση των δύο τεχνικών είναι η εφαρμογή τους σε πραγματικές συνθήκες. Σε πολλές εφαρμογές, ο θόρυβος που εμφανίζεται στα κανάλια επικοινωνίας και τα φυσικά εμπόδια που υπάρχουν ανάμεσα στους κόμβους επηρεάζουν δραματικά τη μετάδοση των πακέτων. Έτσι, η ισχύς του σήματος πρέπει να αυξηθεί αναγκαστικά για τη διατήρηση της αξιοπιστίας στη μετάδοση δεδομένων. Το φαινόμενο αυτό είναι πολύ πιο έντονο κατά τη μετάδοση με short-hops. Αφενός, όταν η ισχύς του σήματος αυξάνεται, η συνολική ενεργειακή κατανάλωση αυξάνεται προσθετικά, όταν ένα πακέτο μεταδίδεται με περισσότερα άλματα. Αφετέρου, αν δεν αυξηθεί η ισχύς του σήματος, η αξιοπιστία στη μετάδοση επηρεάζεται αρνητικά. Έτσι, όσα περισσότερα είναι τα άλματα που ακολουθούν τα πακέτα δεδομένων, τόσο αυξάνεται η συνολική πιθανότητα αποτυχίας αποστολής ενός πακέτου. Έτσι προκύπτουν ορισμένα συμπεράσματα, που οδηγούν στην χρήση longhops έναντι των short-hops. Οι τομείς με τους οποίους σχετίζονται τα συμπεράσματα αυτά αναλύονται στην [10]. Επιγραμματικά, αυτοί που σχετίζονται με την εργασία είναι:

- Παρεμβολές στα κανάλια επικοινωνίας
- Αξιοπιστία στη μετάδοση
- Κωδικοποίηση δεδομένων
- Συνολική κατανάλωση ενέργειας
- Αδρανοποίηση των κόμβων
- Συσσώρευση πακέτων σε ένα κόμβο
- Καθυστέρηση αποστολής

Μπορεί επομένως να αποδειχθεί ότι οι μεγάλες αποστάσεις μεταξύ των κόμβων δεν περιορίζουν κατά κανόνα την ενεργειακή απόδοση ενός ΑΔΑ. Στην [13] εξετάζεται η συσχέτιση ενεργειακής κατανάλωσης με τις αποστάσεις μεταξύ των κόμβων.

Παρατηρείται λοιπόν ότι μέχρι ένα σημείο καμπής (στις συγκεκριμένες περιπτώσεις τα 36 και 125 μέτρα αντίστοιχα) αυξάνοντας το μήκος των αλμάτων, μειώνεται σημαντικά η συνολική ενεργειακή κατανάλωση.

Απλοποιώντας το μοντέλο multi-hopδρομολόγησης με longhops, προκύπτει ένα single-hopμοντέλο. Το επόμενο βήμα είναι να συγκριθεί η single-hopδρομολόγηση με τη multi-hop δρομολόγηση. Στην [14] προτείνεται κωδικοποίηση των δεδομένων με την τεχνική Decode and Forward. Τα αποτελέσματα που προκύπτουν δείχνουν ότι για τη συγκεκριμένη υλοποίηση, η απόσταση μεταξύ των κόμβων και η ανοχή σε σφάλματα καθορίζουν το πότε η multi-hopδρομολόγηση θα είναι πιο αποδοτική από τη single-hop. Όμως, σε όλες τις περιπτώσεις, η δρομολόγηση με κωδικοποίηση των δεδομένων είναι πιο αποδοτική ενεργειακά από κάθε δρομολόγηση στην οποία δεν εφαρμόζεται κωδικοποίηση.

Καταλήγοντας, η single-hopδρομολόγηση μπορεί με τα κατάλληλα πρωτόκολλα να παρουσιάζει μειωμένη ενεργειακή κατανάλωση σε σχέση με τη multi-hopδρομολόγηση. Ταυτόχρονα, μπορεί να εγγυηθεί αξιοπιστία στη μετάδοση δεδομένων, όταν η επικοινωνία των κόμβων είναι δυσχερής. Το ζήτημα που προκύπτει όμως, είναι ότι αναπόφευκτα θα υπάρχει περιορισμός στις αποστάσεις των κόμβων-πηγών από τον κόμβο συλλέκτη. Αυτό σημαίνει ότι η δρομολόγηση των δεδομένων σε μία κλασική single-hopτοπολογία δεν θα είναι αποδοτική για μεγάλες σε έκταση περιοχές τοποθέτησης. Στην επόμενη ενότητα, θα εξεταστούν τοπολογίες με τις οποίες το ζήτημα αυτό μπορεί να επιλυθεί.

2.2.4 Τοπολογίες

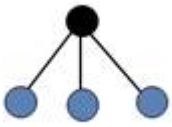
Η τοπολογία ενός ΑΔΑ, αποτελεί ουσιαστικά τον τρόπο με τον οποίο διαμορφώνονται τα μονοπάτια που ακολουθούν τα πακέτα δεδομένων στο δίκτυο. Οι παραπάνω τεχνικές δρομολόγησης μπορούν να χρησιμοποιηθούν σε διάφορες διατάξεις για να διαμορφώσουν συγκεκριμένες τοπολογίες. Οι τοπολογίες αυτές μπορούν να εφαρμοστούν είτε ως έχουν, είτε ομαδοποιημένα, με τη χρήση clusters. Στην πρώτη περίπτωση, ο κόμβος-πύλη κάθε τοπολογίας ταυτίζεται με τον κόμβο-συλλέκτη. Στη δεύτερη, οι κόμβοι-πύλες αποτελούν τις κεφαλές (heads) των clusterskαι αποστέλλουν τα δεδομένα που συγκεντρώνουν στον κεντρικό κόμβο-συλλέκτη.

Η πιο πρωτόλεια τοπολογία είναι αυτή του σημείου προς σημείο, και αποτελείται από μία πηγή και μία πύλη. Είναι οριακά εφαρμόσιμη λόγω των πολύ περιορισμένων δυνατοτήτων της. Παρακάτω παρουσιάζονται οι πιο συνήθεις τοπολογίες για ένα Ασύρματο Δίκτυο αισθητήρων:

- Τοπολογία αστέρα (star)

Η τοπολογία αστέρα είναι ουσιαστικά η υλοποίηση της single-hop δρομολόγησης. Κάθε πηγή αποστέλλει δεδομένα στην πύλη, προσδίδοντας έτσι στο δίκτυο αξιοπιστία , αφού αν ένας κόμβος βρεθεί εκτός λειτουργίας συνεχίζεται κανονικά η συλλογή δεδομένων από τους υπόλοιπους. Αυτό βέβαια ισχύει για την περίπτωση που βρεθεί εκτός λειτουργίας ένας κόμβος-πηγή. Αν κάτι τέτοιο συμβεί στον κόμβο-συλλέκτη τότε

ολόκληρο το δίκτυο τίθεται ουσιαστικά εκτός λειτουργίας. Το βασικό μειονέκτημα της τοπολογίας αστέρα είναι η περιορισμένη απόσταση τοποθέτησης των πηγών από την πύλη.



Εικόνα 2.4: Τοπολογία Αστέρα

Αν λάβουμε υπόψη ότι ένα ΑΔΑ μπορεί να αποτελείται από εκατοντάδες κόμβους, και ότι τα σημεία τοποθέτησης των κόμβων δεν παρουσιάζουν απαραίτητα ομοιογένεια σε ότι αφορά τις συνθήκες επικοινωνίας, πρέπει να εξετάσουμε τη σχέση αξιοπιστίας της επικοινωνίας και ενεργειακής κατανάλωσης των κόμβων στην τοπολογία αστέρα.

Όπως περιγράφεται στην [5], η απόδοση στην επικοινωνία των κόμβων ενός δικτύου μπορεί να αποκλίνει αρκετά από την ιδανική σε μία πραγματική εφαρμογή. Στα σημεία τοποθέτησης των κόμβων παρατηρούνται συνθήκες που δυσκολεύουν την επικοινωνία όπως ανισόπεδο έδαφος, φυσικά εμπόδια, παρεμβολές. Η πιθανότητα ένας κόμβος να αποστείλει τα δεδομένα που συνέλεξε με επιτυχία, υπό αυτές τις συνθήκες, είναι προφανώς μικρότερη σε σχέση με τους υπόλοιπους κόμβους. Έτσι, υποθέτοντας ότι ένα δίκτυο λειτουργεί με πολλαπλές αποστολές των πακέτων δεδομένων που δεν έφτασαν με επιτυχία στον κόμβο-πύλη, η ενεργειακή κατανάλωση μπορεί να αυξηθεί δραματικά σε ορισμένους κόμβους. Άλλωστε, ο βασικός παράγοντας που επηρεάζει την ενεργειακή κατανάλωση ενός κόμβου είναι το κομμάτι της μετάδοσης των δεδομένων, κάτι που κρίνει τη διάρκεια ζωής ενός κόμβου.

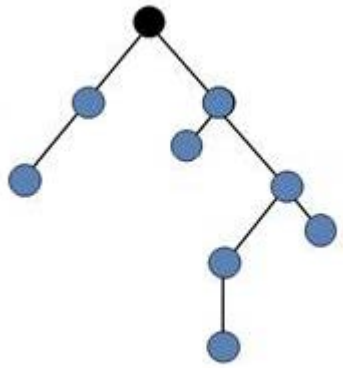
Ακόμη, οι αποστάσεις μεταξύ των κόμβων και η ομοιογενής τοποθέτηση τους δεν μπορεί πάντα να ρυθμίζεται από το χρήστη. Σε πολλές περιπτώσεις οι κόμβοι τοποθετούνται τυχαία (π.χ. λόγω ρίψης τους από απόσταση), ενώ σε άλλες οι κόμβοι μετακινούνται χωρίς τον έλεγχο του χρήστη. Έτσι, σε ένα δίκτυο με τοπολογία αστέρα, ακόμα και αν ο χρήστης έχει περιορίσει την απόσταση των κόμβων από την πηγή, λόγω της τοπολογίας του δικτύου θα υπάρχουν απομακρυσμένοι από την πηγή κόμβοι. Όπως περιγράφεται στην [3], το βασικό μειονέκτημα μιας single-hop τοπολογίας είναι το φαινόμενο της έλλειψης αξιοπιστίας στην επικοινωνία ενός κόμβου όσο αυτός απομακρύνεται από τον κόμβο-πύλη. Η ενεργειακή κατανάλωση ενός κόμβου σε ένα single-hop ασύρματο δίκτυο εξαρτάται άμεσα από την απόσταση του από τον κόμβο-πύλη.

Οι δύο αυτοί παράγοντες που περιγράφηκαν παραπάνω οδηγούν στο συμπέρασμα ότι σε μία τοπολογία αστέρα, υπάρχει η πιθανότητα για ένα κόμβο να λειτουργεί υπό συνθήκες τέτοιες, ώστε η επικοινωνία του με τον κεντρικό κόμβο να είναι εξαιρετικά δυσχερής. Στην περίπτωση αυτή, κάποια πακέτα δεδομένων αποτυγχάνουν να μεταδοθούν στον κόμβο-πύλη ή μεταδίδονται περιέχοντας λάθη (bitsμε λανθασμένη τιμή). Για την αντιμετώπιση αυτού του ζητήματος μπορεί να εφαρμοστεί ένα απλό πρωτόκολλο επανάληψης της αποστολής των πακέτων αυτών. Έτσι η διάρκεια ζωής ενός δικτύου μπορεί να μειωθεί δραματικά, λόγω των απαραίτητων πολλαπλών αποστολών πακέτων δεδομένων. Το αντικείμενο της εργασίας αυτής, είναι η εφαρμογή πιο αναβαθμισμένων τεχνικών ελέγχου λαθών, έτσι ώστε να μειωθούν οι πολλαπλές αποστολές, και οι ενεργειακές δαπάνες που αυτές επιφέρουν. Οι τεχνικές αυτές δίνουν τη δυνατότητα στον παραλήπτη να ανιχνεύσει και να διορθώσει λάθη που υπάρχουν σε ένα πακέτο, αποφεύγοντας την επανάληψη της αποστολής του. Με τον τρόπο αυτό, οι

κόμβοι-πηγές μπορούν να τοποθετηθούν σε μεγαλύτερες αποστάσεις από τον κόμβο-πύλη, εξακολουθώντας ταυτόχρονα να επικοινωνούν αξιόπιστα με τον τελευταίο.

- Τοπολογία δέντρου (tree)

Στην τοπολογία δέντρου γίνεται χρήση multi-hop δρομολόγησης. Οι πηγές διατηρούν ένα αποκλειστικό μονοπάτι προς την πύλη, το οποίο περιλαμβάνει άλλες πηγές και αναμεταδότες. Λόγω της ύπαρξης ενδιάμεσων κόμβων, η εμβέλεια τοποθέτησης των πηγών αυξάνεται, χωρίς όμως να αυξάνονται οι αποστάσεις μεταξύ των κόμβων. Επιλύοντας βέβαια το ζήτημα της εμβέλειας, παρουσιάζεται πρόβλημα αξιοπιστίας, αφού αν βρεθεί ένας ενδιάμεσος κόμβος εκτός λειτουργίας, χάνεται η πρόσβαση στα δεδομένα και των προηγούμενων κόμβων.



Εικόνα 2.5: Τοπολογία δέντρου

Ένα άλλο ζήτημα που προκύπτει στην τοπολογία δέντρου είναι αυτό των αυξημένων ενεργειακών δαπανών των κόμβων που αναμεταδίδουν πακέτα δεδομένων.

Αρχικά, ο όγκος της πληροφορίας σε κάθε πακέτο δεδομένων αυξάνεται λόγω των πολύπλοκων μονοπατιών που ακολουθούν τα δεδομένα. Τα πακέτα δεδομένων θα πρέπει να περιέχουν πληροφορία για τον κόμβο από τον οποίο προέρχονται, έτσι ώστε ο κόμβος-συλλέκτης να γνωρίζει την προέλευσή τους. Αυτό αυξάνει τις ενεργειακές δαπάνες των κυκλωμάτων επικοινωνίας, αλλά και των μικροεπεξεργαστών των κόμβων. Επίσης, η ύπαρξη πολλών διαφορετικών μονοπατιών επιφέρει επιπλέον χρονικές και ενεργειακές δαπάνες για την αρχικοποίηση του δικτύου.

Οι ενδιάμεσοι κόμβοι θα πρέπει συνεχώς να βρίσκονται σε λειτουργία, για να μπορούν να αναμεταδώσουν κάποιο πακέτο. Ακόμα και όταν δεν αναμεταδίδουν δεδομένα ή δεν καταγράφουν κάποια μέτρηση (για την περίπτωση που είναι κόμβοι-πηγές και όχι κόμβοι-αναμεταδότες), χρειάζεται να είναι σε αναμονή για την παραλαβή ενός νέου πακέτου. Αυτό αυξάνει τη συνολική ενεργειακή κατανάλωση του δικτύου.

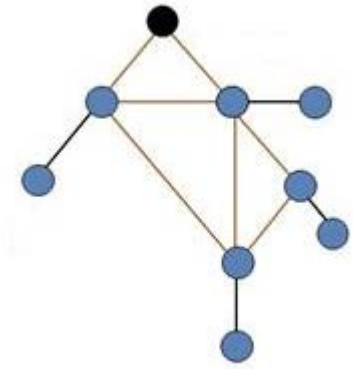
Ένα άλλο χαρακτηριστικό της τοπολογίας δέντρου είναι η υπερφόρτωση δεδομένων στους κόμβους που βρίσκονται κοντινότερα στον κεντρικό κόμβο. Όπως φαίνεται και στην εργασία [1], αυτό συμβαίνει επειδή οι συγκεκριμένοι κόμβοι αναλαμβάνουν να μεταδώσουν όχι μόνο τα δεδομένα που οι ίδιοι συνέλεξαν αλλά και τα δεδομένα που έλαβαν από όλους τους κόμβους που βρίσκονται πίσω τους στο μονοπάτι. Όταν λόγω δυσμενών συνθηκών επικοινωνίας αποτυγχάνει η αποστολή δεδομένων από τους κόμβους αυτούς με το μεγαλύτερο φόρτο δεδομένων (σε σχέση με τους πιο απομακρυσμένους κόμβους), προκύπτει ρήγμα στο μονοπάτι. Αρχικά, η λειτουργία τους επικεντρώνεται στην αποστολή των πακέτων δεδομένων που δεν μεταδόθηκαν με επιτυχία, προκαλώντας συσσώρευση πακέτων στη μνήμη τους. Έπειτα, η ενεργειακή

κατανάλωση τους αυξάνεται πολύ περισσότερο σε σύγκριση με τους υπόλοιπους κόμβους του μονοπατιού, αφού οι επανειλημμένες αποστολές που θα χρειαστεί να εκτελέσουν είναι πολλαπλάσιες. Η υπερκατανάλωση ενέργειας σε αυτούς τους κόμβους μπορεί να περιορίσει τη διάρκεια ζωής τους στο δίκτυο. Στην περίπτωση που οι κόμβοι αυτοί σταματήσουν πλέον να λειτουργούν λόγω έλλειψης ενεργειακών πόρων, ουσιαστικά όλοι οι κόμβοι που βρίσκονται πίσω τους στο μονοπάτι βγαίνουν εκτός λειτουργίας, αφού τα δεδομένα που συλλέγουν δεν μπορούν να φτάσουν στον κόμβο-πύλη.

Σε ό,τι αφορά το ζήτημα αυξημένης κίνησης σε κόμβους που αναμεταδίδουν δεδομένα, η δημιουργία πολλαπλών διαθέσιμων μονοπατιών θα μπορούσε να αποτρέψει τη συσσώρευση δεδομένων σε συγκεκριμένα σημεία του δικτύου.

- Τοπολογία δικτύου βρόγχων – mesh

Για την αντιμετώπιση των περιπτώσεων όπου σε μία τοπολογία δέντρου η επικοινωνία δεν είναι αξιόπιστη, εφαρμόζεται η τοπολογία δικτύου βρόγχων (mesh). Στην τοπολογία αυτή τα δεδομένα αποστέλλονται από τους κόμβους σε ένα κεντρικό κόμβο-πύλη με πολλαπλά άλματα (multi-hop) όπως στην τοπολογία δέντρου, ακολουθούν όμως πολλαπλά μονοπάτια (multipath). Ένας κόμβος μπορεί να αποστείλει τα δεδομένα που συνέλεξε σε πολλούς διαφορετικούς ενδιάμεσους κόμβους, επιλέγοντας τον καταλληλότερο. Έτσι σχηματίζονται διάφορα πιθανά μονοπάτια στο δίκτυο.



Εικόνα 2.6: Τοπολογία δικτύου βρόγχων

Αντίστοιχα με την τοπολογία δέντρου, και σε αυτή την τοπολογία παρουσιάζονται τα ζητήματα επαυξημένης ενεργειακής κατανάλωσης, λόγω της multi-hopδρομολόγησης των πακέτων. Τα πακέτα επιφορτίζονται με πληροφορίες που αφορούν την δρομολόγησή τους, ενώ οι ενδιάμεσοι κόμβοι θα πρέπει να βρίσκονται συνεχώς σε λειτουργία.

Το βασικό ζήτημα που προκύπτει είναι το ενεργειακό κόστος της επιλογής του κόμβου-παραλήπτη από τον κόμβο-αποστολέα. Για τη διαδικασία αυτή εφαρμόζονται διάφορα πρωτόκολλα που βελτιστοποιούν τη διαδικασία επιλογής του ασφαλέστερου ή συντομότερου (ανάλογα με τις ανάγκες του δικτύου) μονοπατιού. Τα πρωτόκολλα αυτά μπορούν να εγγυηθούν αξιόπιστη μετάδοση των δεδομένων, αφού έχουν την δυνατότητα να αποφύγουν την αποστολή πακέτων σε κόμβους που είναι εκτός

λειτουργίας, ή σε κόμβους με υψηλό φόρτο εργασίας. Γενικότερα, το βασικό πλεονέκτημα στην τοπολογία αυτή είναι η δυνατότητα προσαρμογής του δικτύου σε δυσμενείς συνθήκες επικοινωνίας.

Από την άλλη πλευρά, η λειτουργία αυτόματης επιλογής του κατάλληλου μονοπατιού με χρήση των αντίστοιχων πρωτόκολλων επικοινωνίας, επιφέρει αύξηση των ενεργειακών δαπανών του δικτύου. Οι κόμβοι αφιερώνουν ένα μεγάλο μέρος της επεξεργαστικής τους ισχύος σε αυτή τη διαδικασία, κάτι που έχει αντίκτυπο στην κατανάλωση των ενεργειακών τους πόρων.

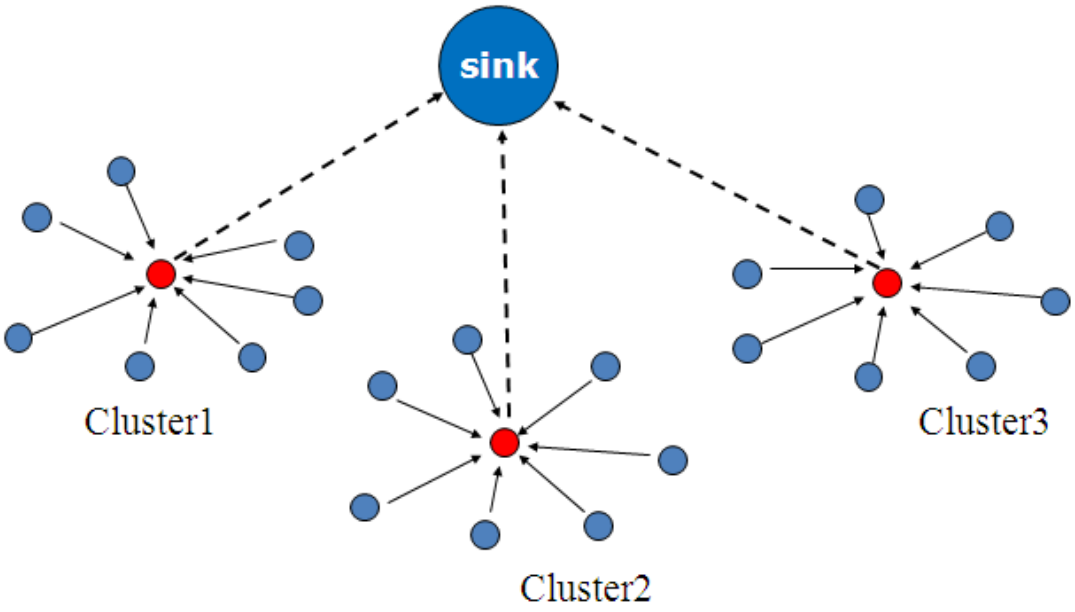
Παρατηρείται πως εφαρμόζοντας πολύπλοκες τοπολογίες με χρήση multi-hopδρομολόγησης, βελτιώνεται η αξιοπιστία στη μετάδοση δεδομένων, επιλύεται το ζήτημα περιορισμών στην απόσταση μεταξύ πηγών και συλλεκτών, αυξάνεται όμως σημαντικά η συνολική ενεργειακή κατανάλωση του δικτύου. Για τη διατήρηση της ενεργειακής κατανάλωσης σε χαμηλά επίπεδα, χωρίς τους περιορισμούς των single-hopτοπολογιών, μπορούν να εφαρμοστούν πολλές επιμέρους απλές τοπολογίες, που επικοινωνούν μεταξύ τους. Κάτι τέτοιο γίνεται δυνατό με τη χρήση clusters, που εξετάζεται στη συνέχεια.

- Τοπολογίες με χρήση clusters

Για τη βελτίωση της επικοινωνίας μεταξύ των κόμβων του δικτύου, μπορεί να γίνει ομαδοποίηση των κόμβων. Οι ομάδες αυτές που δημιουργούνται (clusters) χρησιμοποιούν μία από τις παραπάνω τοπολογίες για να επικοινωνήσουν, έτσι ώστε τα δεδομένα να συγκεντρωθούν στο κόμβο-πύλη του κάθε cluster, που ονομάζεται cluster head. Οι cluster heads αποστέλλουν τα δεδομένα που συγκέντρωσαν είτε στον κόμβο-συλλέκτη του δικτύου, είτε σε κάποιο κόμβο ενός άλλου cluster.

Οι cluster heads, αναλαμβάνουν ουσιαστικά τη διαχείριση των clusters. Σε πολλές περιπτώσεις οι clusters οργανώνονται σε διαφορετικές μεταξύ τους τοπολογίες. Για το λόγο αυτό οι cluster heads αναλαμβάνουν τη μορφοποίηση των δεδομένων που συγκέντρωσαν, συμβάλλοντας έτσι στη δρομολόγηση των πακέτων δεδομένων.

Οι cluster heads που βρίσκονται σε τοπολογίες multi-hop παρουσιάζουν αναβαθμισμένες λειτουργίες για τη διαχείριση των clusters. Φροντίζουν για την αξιόπιστη μετάδοση των δεδομένων σε περιπτώσεις αποτυχίας ενδιάμεσων κόμβων, αυξημένης κίνησης σε συγκεκριμένα μονοπάτια ή γενικότερα δυσμενών συνθηκών επικοινωνίας. Έτσι οι cluster heads ενδέχεται να εκτελέσουν ανασχηματισμό των μονοπατιών που ακολουθούν τα πακέτα δεδομένων στον cluster. Η διαδικασία αυτή επιβαρύνει την ενεργειακή κατανάλωση του δικτύου. Παρατηρείται λοιπόν ότι το ζήτημα της αυξημένης ενεργειακής κατανάλωσης σε τοπολογίες multi-hop λόγω της πολύπλοκης δρομολόγησης των πακέτων, δεν επιλύεται σε απόλυτο βαθμό με την ομαδοποίηση σε clusters.



Εικόνα 2.7: Τοπολογία Single-hop Clusters. Οι κόμβοι με κόκκινο χρώμα αποτελούν τους cluster heads

Από την άλλη πλευρά, η χρήση clusters επιλύει το ζήτημα της περιορισμένης απόστασης τοποθέτησης των κόμβων-πηγών από τον κόμβο-πύλη, σε τοπολογίες single-hop. Οργανώνοντας τους κόμβους του δικτύου σε πολλές επιμέρους τοπολογίες αστέρα, μπορεί να καλυφθούν μεγάλες εκτάσεις της περιοχής τοποθέτησης. Οι κόμβοι-πηγές μεταδίδουν τα πακέτα δεδομένων στους επιμέρους κόμβους-πύλες με μονά άλματα, και οι τελευταίοι αναλαμβάνουν να τα μεταδώσουν σε κάποιο γειτονικό cluster ή στον κόμβο-συλλέκτη του δικτύου, επίσης με μονό άλμα. Με αυτό τον τρόπο δημιουργείται ένα δίκτυο από clusters με τοπολογία αστέρα (single-hop), που σε ανώτερο επίπεδο τα δεδομένα καταλήγουν στον κόμβο-συλλέκτη με πολλαπλά άλματα (multi-hop). Έτσι, αξιοποιείται η λιτότητα της τοπολογίας αστέρα, χωρίς να προκύπτουν τα ζητήματα μιας τοπολογίας multi-hop που αναφέρθηκαν προηγουμένως.

Στο σημείο αυτό πρέπει να αναφερθεί η σημασία της αξιόπιστης επικοινωνίας σε ένα δίκτυο που οργανώνεται σε clusters. Η χρήση clustersδίνει τη δυνατότητα εφαρμογής τοπολογιών αστέρα σε εκτενείς περιοχές τοποθέτησης, αυτό όμως δεν επιλύει απαραίτητα το ζήτημα της αξιοπιστίας στη μετάδοση, σε συνθήκες υψηλού θορύβου και παρεμβολών στα κανάλια επικοινωνίας. Στην περίπτωση που κάποιοι κόμβοι του cluster αδυνατούν να μεταδώσουν δεδομένα, ή είναι συνολικά εκτός δικτύου, ο cluster head επιβαρύνεται επιπλέον με την αρμοδιότητα του re-clusteringγια να διευθετήσει το ζήτημα. Όπως βλέπουμε στην εργασία [2], το re-clustering περιπλέκει το setup του δικτύου και τη διαδικασία επανεκκίνησης του. Οι cluster heads θα πρέπει να διακόψουν την επεξεργασία των δεδομένων και την επικοινωνία με τους υπόλοιπους κόμβους. Έπειτα, θα πρέπει να δημιουργηθούν νέα πρωτόκολλα επικοινωνίας και να σταλθούν σε όλους τους κόμβους. Όταν αυτή η διαδικασία συμβαίνει συχνά, μπορεί να οδηγήσει σε μεγάλη σπατάλη χρόνου και ενεργειακών πόρων στο δίκτυο.

2.2.5 Εφαρμογή Ελέγχου Λαθών σε μία τοπολογία

Στην προηγούμενη ενότητα αναλύθηκαν οι συνήθεις τοπολογίες σε ένα ΑΔΑ. Το συμπέρασμα που προκύπτει είναι ότι η μετάδοση δεδομένων με πολλαπλά άλματα

εγγυάται αξιοπιστία στην επικοινωνία των κόμβων, παρουσιάζει όμως αναβαθμισμένη πολυπλοκότητα. Αυτό επιβαρύνει την ενεργειακή κατανάλωση των ενδιάμεσων κόμβων ενός μονοπατιού μετάδοσης, ιδιαίτερα σε δυσμενείς συνθήκες επικοινωνίας. Από την άλλη, η χρήση μιας απλούστερης τοπολογίας με μετάδοση μονών αλμάτων δεν εγγυάται την αξιόπιστη μετάδοση των πακέτων. Λόγω των μεγάλων αποστάσεων μεταξύ των κόμβων, συχνά τα πακέτα περιέχουν λανθασμένα bits όταν φτάνουν στον κόμβο-παραλήπτη.

Για την αποφυγή κολλήματος στη διαδικασία αυτή σε συνθήκες υψηλού θορύβου, επιλέγεται από το χρήστη ένας μέγιστος αριθμός επαναλήψεων. Η τεχνική αυτή δεν καταναλώνει επιπλέον επεξεργαστική ισχύ σε ένα κόμβο, εμφανίζει όμως δύο προβλήματα. Αφενός, οι πολλαπλές αποστολές του ίδιου πακέτου σημαίνουν πολλαπλάσιες ενεργειακές δαπάνες των κυκλωμάτων επικοινωνίας. Αν αναλογιστεί κανείς ότι το μεγαλύτερο μέρος των ενεργειακών πόρων ενός δικτύου αφιερώνεται στην επικοινωνία, οι πολλαπλές αποστολές επιβαρύνουν ιδιαίτερα την συνολική ενεργειακή κατανάλωση στο δίκτυο. Αφετέρου, όταν οι συνθήκες δεν επιτρέπουν την επιτυχημένη αποστολή ενός πακέτου ακόμα και μετά το πέρας του μέγιστου αριθμού επαναλήψεων, παρουσιάζεται απώλεια πληροφορίας.

Για την βελτιστοποίηση αυτής της τεχνικής ελέγχου λαθών, μπορεί παράλληλα να εφαρμοστεί κωδικοποίηση καναλιού (Channel Coding). Η τεχνική αυτή, που ονομάζεται Forward Error Correction (FEC), περιλαμβάνει την εφαρμογή ενός αλγόριθμου κωδικοποίησης των bits ενός πακέτου πριν την αποστολή του. Κατά την παραλαβή του κωδικοποιημένου πακέτου, ο αντίστοιχος αλγόριθμος αποκωδικοποίησης εξάγει την αρχική μορφή της πληροφορίας που μεταδόθηκε. Κατά την αποκωδικοποίηση, αν υπάρχουν λανθασμένα bits στο πακέτο, μπορούν να ανακτηθούν στην αρχική τους μορφή.

Η διαδικασία αυτή επηρεάζει την ενεργειακή κατανάλωση ενός κόμβου σε δύο επίπεδα. Από τη μία, αφιερώνονται ενεργειακοί πόροι για την επεξεργασία των δεδομένων. Οι αλγόριθμοι κωδικοποίησης, αλλά πολύ περισσότερο οι αλγόριθμοι αποκωδικοποίησης, απαιτούν ένα σημαντικό μέρος της επεξεργαστικής ισχύος του συστήματος. Από την άλλη, η κωδικοποίηση των δεδομένων αυξάνει το μέγεθος των πακέτων. Αυτό επηρεάζει την κατανάλωση ενεργειακών πόρων από τα κυκλώματα επικοινωνίας του συστήματος.

Παρόλα αυτά, η τεχνική FEC βελτιώνει τη μετάδοση των δεδομένων σε τέτοιο βαθμό, που τα ενεργειακά της κόστη πολλές φορές αντισταθμίζονται από την ενεργειακή εξοικονόμηση που αυτή επιφέρει. Δηλαδή, η διαδικασία ανάκτησης των λανθασμένων bits δίνει τη δυνατότητα στον παραλήπτη να ανακατασκευάσει ένα πακέτο που μεταδόθηκε ανεπιτυχώς, χωρίς να χρειάζεται επανάληψη της αποστολής του. Αυτό σημαίνει αφενός δραματική μείωση των πολλαπλών αποστολών ενός πακέτου. Αφετέρου, η τεχνική αυτή μπορεί να εγγυηθεί αξιόπιστη επικοινωνία σε συνθήκες τέτοιες, που χωρίς την εφαρμογή FEC τα πακέτα δεδομένων δεν θα μπορούσαν να μεταδοθούν επιτυχώς ακόμα και μετά από πολλαπλές αποστολές.

Για τους λόγους αυτούς, η εργασία αυτή εξετάζει τη χρήση FEC σε μία single-hop τοπολογία, αλλά και σε μία multi-hop τοπολογία, με τη μορφή single-hop clusters. Ο λόγος που ερευνάται η χρήση FECσε κατά βάση single-hop τοπολογίες έχει να κάνει

με την πολυπλοκότητα των αλγορίθμων αποκωδικοποίησης. Η τελευταία διαδικασία απαιτεί υψηλότερων δυνατοτήτων επεξεργαστές από αυτούς που διαθέτουν οι μικροελεγκτές που χρησιμοποιούνται συνήθως για τους κόμβους-πηγές. Έτσι, το φόρτο της αποκωδικοποίησης αναλαμβάνουν είτε οι κόμβοι-συλλέκτες μιας τοπολογίας αστέρα, είτε οι κόμβοι-πύλες (cluster heads) πολλών επιμέρους αστέρων που επικοινωνούν μεταξύ τους και με τον κόμβο-συλλέκτη. Οι κόμβοι αυτοί συνήθως αποτελούνται από αναβαθμισμένα συστήματα επεξεργασίας, διαθέτοντας αντίστοιχα αναβαθμισμένες πηγές τροφοδοσίας. Έτσι, έχουν τη δυνατότητα να αποστέλλουν τα δεδομένα που συγκέντρωσαν και αποκωδικοποίησαν σε μεγάλες αποστάσεις, αφού χρησιμοποιούν συστήματα επικοινωνίας μεγαλύτερης εμβέλειας από τους απλούς κόμβους-πηγές.

3 Forward Error Correction

Στο κεφάλαιο αυτό περιγράφεται η λειτουργία της τεχνικής Forward Error Correction (FEC) από τη βιβλιογραφία ([16], [17]). Στη συνέχεια παρουσιάζονται οι βασικές κατηγορίες κώδικα FEC (Block Codes, Convolutional Codes) [17]. Επιπλέον, αναλύεται η λειτουργία των Turbo Codes, οι οποίοι συνδυάζουν στοιχεία των Convolutional Codes και Block Codes. Στην κατηγορία αυτή ανήκουν οι αλγόριθμοι που εξετάζονται στη συγκεκριμένη εργασία. Συγκεκριμένα, παρουσιάζεται η λειτουργία ενός Turbo Encoder και ενός Turbo Decoder. Τέλος, γίνεται αναφορά στον βασικό αλγόριθμο αποκωδικοποίησης Convolutional και Turbo Codes, Viterbi.

3.1 Περιγραφή

Η τεχνική Forward Error Correction, που στα τηλεπικοινωνιακά συστήματα εφαρμόζεται με χρήση κωδικοποίησης καναλιού (Channel Coding), χρησιμοποιείται για έλεγχο λαθών σε κανάλια επικοινωνίας με χαμηλή αξιοπιστία ή υψηλές τιμές θορύβου. Συνοπτικά, ο αποστολέας ενός μηνύματος κωδικοποιεί την πληροφορία πριν την αποστείλει, προσθέτοντας πλεονάζουσα πληροφορία με συστηματικό τρόπο, κάνοντας χρήση ενός κώδικα διόρθωσης λαθών. Ο παραλήπτης του μηνύματος χρησιμοποιεί την πλεονάζουσα πληροφορία για να ανιχνεύσει λάθη στο αρχικό μήνυμα και να τα διορθώσει. Η χρήση FEC σε ένα τηλεπικοινωνιακό σύστημα μπορεί να περιορίσει ή ακόμα και να εξαλείψει τις αναμεταδώσεις πακέτων δεδομένων που έφτασαν στον παραλήπτη περιέχοντας λάθη, αφού ο τελευταίος έχει τη δυνατότητα να ανακατασκευάσει τα λανθασμένα πακέτα, εξάγοντας την αρχική πληροφορία.

Τα πλεονάζοντα bits που προστίθενται σε ένα μήνυμα κατά τη χρήση FEC προκύπτουν συναρτήσει των bits της αρχικής πληροφορίας. Ο αλγόριθμος που παράγει τα bits αυτά ενδέχεται να είναι αρκετά περίπλοκος, ενώ ένας αποδοτικός αλγόριθμος μπορεί να εγγυηθεί ότι ο παραλήπτης του μηνύματος θα ανιχνεύσει έναν πεπερασμένο αριθμό λαθών σε ένα μήνυμα, και θα ανακτήσει την αρχική τιμή των bits αυτών. Με τον τρόπο αυτό, τα λανθασμένα πακέτα ανακατασκευάζονται στον παραλήπτη.

Σε μια κωδικοποίηση τύπου FEC, η πληροφορία προς κωδικοποίηση είναι μία ακολουθία k-bits που αποτελεί είσοδο για τον κωδικοποιητή (encoder). Από την έξοδο του encoder προκύπτει μία ακολουθία n-bits, που αποτελεί μία κωδική λέξη.

Ο παραλήπτης θα πρέπει να διαθέτει τον κατάλληλο αποκωδικοποιητή (decoder) για τον συγκεκριμένο encoder, έτσι ώστε να αποκωδικοποιήσει την λέξη. Έτσι, οι κώδικες FEC χαρακτηρίζονται από τη σημειογραφία (n,k), όπου το n προκύπτει από τα bits εξόδου του encoder, και το k από τα bits εισόδου.

Ένας πολύ απλός κώδικας FEC (3,1) έχει ως εξής: κάθε bit εισόδου εισέρχεται στον encoder, όπου προστίθενται 2 επιπλέον bits με την ίδια τιμή. Στον παραλήπτη, ενδέχεται ένα ή περισσότερα bits να έχει αλλοιωθεί από θόρυβο. Έτσι, ο decoder αποφασίζει ποια είναι η αρχική τιμή του κάθε bit, με βάση την πλειοψηφία. Οι κωδικές λέξης αποκωδικοποιούνται ως εξής:

Triplet received	Interpreted as
000	0 (error free)
001	0
010	0
100	0
111	1 (error free)
110	1
101	1
011	1

Πίνακας 1.1: Αποτελέσματα αποκωδικοποίησης λέξεων ενός κώδικα (3,1)

Ο απλός αυτός κώδικας αποτελεί παράδειγμα για το πώς προσθέτοντας περίσσεια πληροφορία από την πλευρά του αποστολέα ενός μηνύματος, ο παραλήπτης του μπορεί να διαβάσει το μήνυμα ακόμα και αν αυτό περιέχει λάθη. Παρόλα αυτά, αποτελεί μία πολύ απλή διαδικασία. Όπως θα δειχθεί στη συνέχεια, οι κώδικες FECχρησιμοποιούν αυξημένης πολυπλοκότητας αλγορίθμους, ιδιαίτερα στο κομμάτι της αποκωδικοποίησης.

Η τεχνική FEC αυξάνει την κίνηση σε ένα κανάλι επικοινωνίας, είτε από την άποψη της επιπλέον πληροφορίας που προστίθεται, είτε από την άποψη της καθυστέρησης στο κανάλι που η FEC επιφέρει. Επιπλέον, ο λόγος της πραγματικής πληροφορίας προς τη συνολική πληροφορία που μεταδίδεται είναι χαμηλός. Για τους λόγους αυτούς η χρήση FEC δεν είναι σε όλες τις εφαρμογές αποδοτική. Χρειάζεται ανάλυση των συνθηκών επικοινωνίας, των διαθέσιμων κυκλωμάτων και των απαιτήσεων ενός συστήματος για να διαπιστωθεί αν η κωδικοποίηση καναλιού βελτιώνει ή δυσχεραίνει την επικοινωνία σε ένα δίκτυο, και σε ποιο τομέα συμβαίνει αυτό (χρονική απόδοση, ενεργειακή κατανάλωση, κόστος παραγωγής και συντήρησης κ.α.). Από την άλλη πλευρά, η χρήση FEC εγγυάται ότι ενώ η ισχύς του σήματος στα συστήματα επικοινωνίας παραμένει σε σταθερά επίπεδα, και χωρίς κάποια καταστροφική για τις συνθήκες επικοινωνίας παρέμβαση στο δίκτυο, ο αριθμός λαθών σε ένα δέκτη μπορεί να παραμείνει χαμηλός.

Στη γενική περίπτωση, η επιπλέον επεξεργαστική ισχύς που απαιτεί η διαδικασία κωδικοποίησης κυμαίνεται σε χαμηλά επίπεδα. Στο περιβάλλον ενός ΑΔΑ, τα ηλεκτρονικά συστήματα που λειτουργούν ως κόμβοι-πομποί δεν χρειάζονται ιδιαίτερη αναβάθμιση για εκτελέσουν τη σχετική επεξεργασία. Οι microcontrollers που χρησιμοποιούνται στις περισσότερες περιπτώσεις μπορούν να διαχειριστούν τη διαδικασία του encoding. Ο φόρτος εργασίας μεταφέρεται όμως στο δέκτη, που αναλαμβάνει τη διαδικασία ανίχνευσης και διόρθωσης λαθών. Η διαδικασία του decoding απαιτεί συνήθως ηλεκτρονικά συστήματα σημαντικής υπολογιστικής ισχύος και περισσότερους ενεργειακούς πόρους. Για τον λόγο αυτό σε δίκτυα αισθητήρων, προτείνονται ασύμμετρες τοπολογίες που περιλαμβάνουν κεντρικούς κόμβους

αναβαθμισμένων δυνατοτήτων που λαμβάνουν και αποκωδικοποιούν δεδομένα από διάφορους, πιο ανίσχυρους κόμβους-πομπούς, που βρίσκονται σε γειτονικές θέσεις.

Τέλος, αξίζει να γίνει μία αναφορά στην τεχνική διάτρησης (Puncturing) που εφαρμόζεται συχνά κατά τη χρήση FEC. Με την τεχνική αυτή, αφού η αρχική πληροφορία κωδικοποιηθεί, αφαιρούνται κάποια από τα bits ισοτιμίας. Η κωδική λέξη που προκύπτει έχει μικρότερο μέγεθος λόγω της εφαρμογής Puncturing, και άρα το αποτέλεσμα προσομοιάζει με τη χρήση αλγόριθμου κωδικοποίησης μεγαλύτερου Rate. Μία τέτοια τεχνική αξιοποιείται σε περιπτώσεις με μικρό αριθμό λαθών στα πακέτα που μεταδίδονται, αφού ο όγκος της μεταδιδόμενης πληροφορίας μικραίνει, χωρίς να επηρεαστεί η αξιοπιστία στην επικοινωνία. Το χαρακτηριστικό της τεχνικής αυτής είναι ότι για την αποκωδικοποίηση μίας punctured κωδικής λέξης χρησιμοποιείται ο ίδιος decoder, ανεξάρτητα από το πόσα bitισοτιμίας αφαιρέθηκαν. Με αυτό τον τρόπο το σύστημα είναι σχετικά ευέλικτο, ενώ ταυτόχρονα η πολυπλοκότητα του αποκωδικοποιητή παραμένει σταθερή. Σε κάποιες εφαρμογές χρησιμοποιείται ένα συγκεκριμένο πρότυπο Puncturing στον encoder, έτσι ώστε να εκτελεστεί η αντίστροφη διαδικασία (depuncturing) στον decoder. Στη συγκεκριμένη υλοποίηση FEC, δεν εξετάστηκε η χρήση Puncturing, λόγω των καναλιών με υψηλές τιμές θορύβου και του αναμενόμενου μεγάλου αριθμού λαθών κατά τη μετάδοση που συναντώνται στις εφαρμογές των ΑΔΑ.

3.2 Τύποι FEC

Η κωδικοποίηση καναλιού εφαρμόζεται με τρεις βασικές μεθόδους [17]. Οι δύο βασικές μέθοδοι είναι δύο κατηγορίες κώδικα. Η πρώτη αφορά τους κώδικες μπλοκ (Block Codes) και η δεύτερη τους συνελικτικούς κώδικες (Convolutional Codes). Η τρίτη μέθοδος είναι η τεχνική Interleaving. Η τεχνική αυτή συνδυάζεται συνήθως με κώδικες από τις δύο βασικές κατηγορίες. Γενικά, και οι τρεις μέθοδοι μπορούν να συνδυαστούν σε μία εφαρμογή. Οι μέθοδοι αυτές παρουσιάζονται στη συνέχεια.

3.2.1 Block Codes

Με τον όρο Block Codes περιγράφεται μία ευρεία κατηγορία διαφόρων τύπων κώδικα, όπου κάθε τύπος διαχωρίζεται από τους υπόλοιπους μέσω διάφορων περιορισμών. Οι περιορισμοί αυτοί καθορίζουν κάποιες παραμέτρους για τον κάθε κώδικα, οι οποίες τον χαρακτηρίζουν. Η βασική αρχή των Block Codes είναι ότι για μια πληροφορία k bits, παράγεται ένα μπλοκ δεδομένων μεγέθους n bits, γι’ αυτό και χρησιμοποιείται η σημειογραφία (n,k). Κλασσικά παραδείγματα Block Codes αποτελούν οι κώδικες Reed–Solomon και Hamming, που χρησιμοποιούνται συχνά στα ΑΔΑ. Οι κώδικες αυτοί λέγονται γραμμικοί, επειδή κάθε γραμμικός συνδυασμός κωδικών λέξεων αποτελεί μία κωδική λέξη. Πιο συγκεκριμένα, αποκαλούνται και αλγεβρικοί ή κυκλικοί, επειδή παράγονται χρησιμοποιώντας δυαδικής μορφής πολυώνυμα. Επιπλέον, οι περισσότεροι Block Codes είναι συστηματικοί. Αυτό σημαίνει ότι η αυθεντική πληροφορία δεν αλλοιώνεται στο κωδικοποιημένο μήνυμα,

αφού προστίθενται bits ισοτιμίας στην αρχή ή στο τέλος του μηνύματος. Οι αλγόριθμοι αποκωδικοποίησης των Block Codes χρησιμοποιούν hard-decisions, δηλαδή παράγουν απευθείας μία τιμή 0 ή 1 για κάθε bit που αποκωδικοποιούν, και όχι μία προσεγγιστική τιμή. Η κατηγορία Turbo Codes που θα παρουσιαστεί στη συνέχεια υιοθετεί πολλά από τα χαρακτηριστικά αυτά που αναφέρθηκαν, και με μία ευρεία έννοια ανήκει στους Block Codes. Όμως, μια πιο ακριβής προσέγγιση τους περιγράφει ως υβρίδιο των Block Codes και Convolutional Codes. Στη συνέχεια παρουσιάζονται οι παράμετροι που χαρακτηρίζουν τους Block Codes:

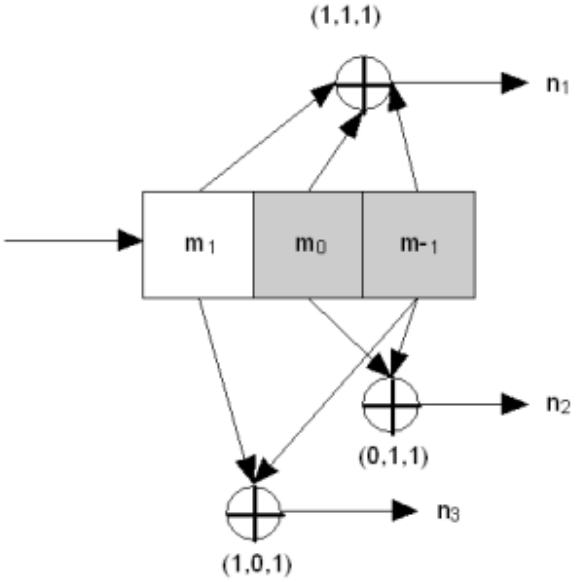
- **Αλφάβητο (∑)**: Η ροή δεδομένων που πρόκειται να κωδικοποιηθεί αποτυπώνεται ως μία συμβολοσειρά ενός αλφαβήτου. Το μέγεθος |∑| του αλφαβήτου συμβολίζεται συνήθως με **q** .
- **Μέγεθος μηνύματος (k)**: Τα μηνύματα είναι στοιχεία του συνόλου $\sum^{k}$, δηλαδή συμβολοσειρές μήκους k.
- **Μέγεθος μπλοκ (n)**: Το μέγεθος του κάθε μπλοκ δεδομένων είναι ο αριθμός συμβόλων που περιέχει ένα κωδικοποιημένο μήνυμα. Έτσι, τα στοιχεία του συνόλου $\sum^{n}$είναι συμβολοσειρές μήκους n που αντιστοιχούν σε μπλοκ που λαμβάνονται από τον δέκτη.
- **Ρυθμός R:** Έτσι ορίζεται ο λόγος **R** του μήκους του μηνύματος **k**, προς το μήκος του μπλοκ **n**: **R=k/n**.
- **Απόσταση d :** Η απόσταση ή ελάχιστη απόσταση **d** ενός Block Code, είναι ο ελάχιστος αριθμών θέσεων στις οποίες διαφέρουν δύο οποιεσδήποτε κωδικές λέξεις, και η σχετική απόσταση **δ** είναι το κλάσμα **d/n**.

3.2.2 Convolutional Codes

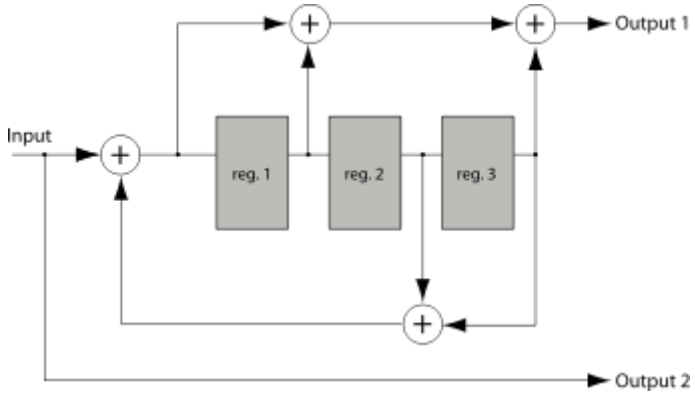
Ένας συνελικτικός κώδικας (Convolutional Code) παράγει bit ισοτιμίας εφαρμόζοντας μία δυαδική πολυωνυμική συνάρτηση που ολισθαίνει σε μία ροή δεδομένων. Η ολίσθηση αυτή δίνει το χαρακτήρα συνέλιξης στη διαδικασία κωδικοποίησης, γι' αυτό και χαρακτηρίζεται συνελικτικός. Η διαφορά των Convolutional Codes από τους Block Codes είναι ότι οι πρώτοι έχουν τη δυνατότητα να κωδικοποιούν μια συνεχιζόμενη ροή δεδομένων (continuous encoding), και δεν παράγουν αποκλειστικά μπλοκ δεδομένων με πεπερασμένο μέγεθος. Στην πρακτική εφαρμογή τους, οι Convolutional Codes κωδικοποιούν μπλοκ δεδομένων συγκεκριμένου μεγέθους. Όμως, το μήκος των μπλοκ δεδομένων καθορίζεται από το χρήστη, σε αντίθεση με τους Block Codes που καθορίζεται από τις αλγεβρικές ιδιότητες του κάθε κώδικα. Επιπλέον, οι Convolutional Codes αποκωδικοποιούν τα δεδομένα χρησιμοποιώντας τον κανόνα μέγιστης πιθανοφάνειας, εφαρμόζοντας soft-decisions. Αυτό σημαίνει ότι για κάθε bit που αποκωδικοποιούν, παράγουν μία τιμή στο διάστημα [0,1] που προσεγγίζει το αν το bit που αποκωδικοποιήθηκε είχε την τιμή '0' ή '1' πριν την κωδικοποίησή του.

Για ένα Convolutional Encoder, χρησιμοποιούνται k καταχωρητές μνήμης μεγέθους 1 bit, που συνήθως είναι αρχικοποιημένοι στο 0. Ακόμα χρησιμοποιούνται n αθροιστές modulo-2, που υλοποιούνται με πύλες XOR, για να παραγάγουν τις τιμές εξόδου n πολυωνύμων. Κάθε bit εισόδου περνάει διαδοχικά από τους καταχωρητές που αποθηκεύουν την τιμή του, την οποία διαδίδουν στις πύλες XOR. Ο αριθμός των

πολυωνύμων καθορίζει τον αριθμό εξόδων του συστήματος άρα και το rate του encoder, ενώ ο αριθμός των καταχωρητών καθορίζει το constraint length. Οι συστηματικοί (systematic) encoders διαθέτουν έξοδο που παράγει αυτούσια τα bits εισόδου, ενώ οι μη συστηματικοί (non-systematic) όχι. Η έξοδος των τελευταίων αλλοιώνει την αρχική πληροφορία, αφού δεν περιέχει αυτούσια τα bits εισόδου. Οι αναδρομικοί (recursive) encoders περιλαμβάνουν ανάδραση της εξόδου στην είσοδο του συστήματος, σε αντίθεση με τους μη αναδρομικούς (non-recursive). Μία κοινή πρακτική για τους Convolutional Encoders είναι να σχεδιάζονται non-systematic, non-recursive encoders και systematic, recursive encoders, χωρίς όμως αυτό να αποτελεί κανόνα. Τέλος, ο αριθμός καταστάσεων του encoder καθορίζεται από το πόσοι κύκλοι ρολογιού χρειάζονται για να κωδικοποιηθεί πλήρως ένα bit, να υπάρξει δηλαδή αποτέλεσμα για το κάθε πολυώνυμο σε όλες τις εξόδους του συστήματος, από τη στιγμή που εισάγεται το πρώτο bit. Στις εικόνες που ακολουθούν παρουσιάζονται δύο encoders που συνδυάζουν τις διαφορετικές παραμέτρους που αναφέρθηκαν προηγουμένως:



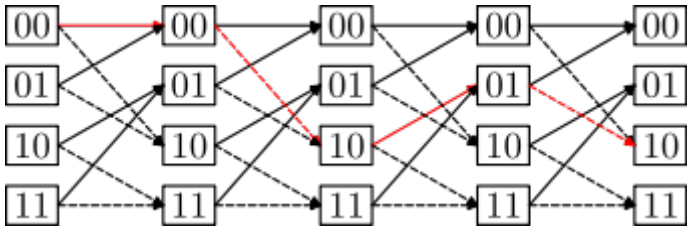
Εικόνα 3.1: Convolutional Encoder με R=1/3, non-recursive, non-systematic, με constraint length 3, τριών καταστάσεων.



Εικόνα 3.2: Convolutional encoder με R=1/2, recursive, systematic, 8 καταστάσεων

Η αποκωδικοποίηση των κωδικοποιημένων μηνυμάτων γίνονται με βάση ένα διάγραμμα Trellis. Στο διάγραμμα αυτό απεικονίζονται όλες οι πιθανές καταστάσεις

των καταχωρητών, ανάλογα με τις τιμές που έχουν αποθηκεύσει, καθώς και οι πιθανές επόμενες καταστάσεις τους. Όταν υπάρχει μετάβαση από μία κατάσταση σε μία άλλη, η οποία δεν απεικονίζεται στο διάγραμμα, σημαίνει ότι το bit που αποκωδικοποιείται είναι λάθος. Τότε, για την ανάκτησή του, επιλέγεται η κοντινότερη πιθανή επόμενη κατάσταση. Αναφορικά, οι μεταβάσεις απεικονίζονται με συμπαγής γραμμή όταν υπάρχει είσοδος '0', και με διακεκομμένη όταν υπάρχει είσοδος '1'. Με βάση το διάγραμμα trellis, δημιουργούνται οι μήτρες ισοτιμίας (parity matrixes), για την υλοποίηση του decoder. Για τον encoder της Εικόνας 3.2, παρουσιάζεται το αντίστοιχο Trellis:



Εικόνα 3.3: Το διάγραμμα Trellis για τον encoder της Εικόνας 3.2. Ένα έγκυρο παράδειγμα μονοπατιού μετάβασης στις καταστάσεις των καταχωρητών σημειώνεται με κόκκινο χρώμα.

3.2.3 Interleaving

Η τεχνική Interleaving χρησιμοποιείται κατά τη χρήση FEC σε τηλεπικοινωνιακά συστήματα και συστήματα αποθήκευσης δεδομένων για τη βελτιστοποίηση του ελέγχου λαθών. Συγκεκριμένα για τα τηλεπικοινωνιακά συστήματα, σε διάφορα κανάλια προκύπτουν λάθη λόγω θορύβου σε πολλά διαδοχικά bits. Ο αριθμός των διαδοχικών λανθασμένων bits μπορεί να είναι τέτοιος, ώστε η ικανότητα διόρθωσης λαθών του αλγορίθμου αποκωδικοποίησης να μην είναι επαρκής για να ανακτήσει την αρχική πληροφορία που μεταδόθηκε. Με την τεχνική Interleaving η κωδικοποιημένη συμβολοσειρά ανακατατάσσεται, δηλαδή η σειρά των bits αλλάζει, με βάση ένα συγκεκριμένο πρότυπο (Interleaver Pattern). Κατά την αποκωδικοποίηση, η σειρά των bits παίρνει ξανά την αρχική της μορφή, με βάση το pattern που χρησιμοποιήθηκε στην κωδικοποίηση. Με τον τρόπο αυτό, τα λανθασμένα bits ακολουθούν μία πιο ομοιόμορφη κατανομή κατά μήκος της συμβολοσειράς που μεταδίδεται.

Η χρήση Interleaver στους σύγχρονους αλγόριθμους κωδικοποίησης FEC είναι αρκετά διαδεδομένη. Για τους Turbo Codes που θα παρουσιαστούν στη συνέχεια, ο Interleaver είναι ένα δομικό στοιχείο τόσο του encoder, όσο και του decoder. Ο τύπος Interleaver που θα χρησιμοποιηθεί εξαρτάται από τις παραμέτρους και τα πρωτόκολλα επικοινωνίας της κάθε εφαρμογής. Ενδεικτικά υπάρχουν οι:

- **Ορθογώνιοι (Rectangular) Interleavers**: Δημιουργείται ένας πίνακας στοιχείων όπου τα bits μιας συμβολοσειράς τοποθετούνται διαδοχικά στις στήλες του πίνακα
- **Συνελικτικοί (Convolutional) Interleavers**: Τα bits εισάγονται σε διαδοχικά σε μία σειρά από καταχωρητές ολίσθησης, από τους οποίους εξάγεται η παλαιότερη τιμή
- **Τυχαίοι (Random) Interleavers**: Τα bits ανακατατάσσονται σύμφωνα με μία γνωστή, τυχαία σειρά.

- Interleavers που χρησιμοποιούν ένα πολυώνυμο μετάθεσης για την ανακατάταξη των bits, όπως αυτοί του 3GPP LTE standard.

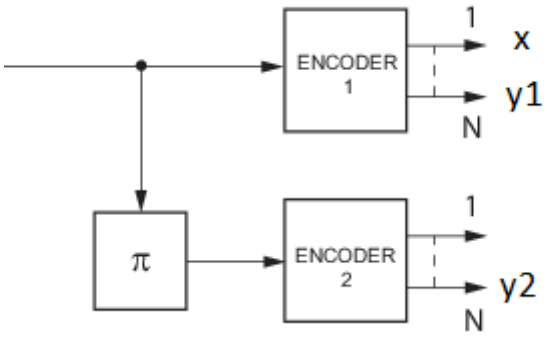
Η χρήση Interleaver προσθέτει έναν περιορισμό στη διαδικασία κωδικοποίησης/αποκωδικοποίησης. Πλέον ο κώδικας δεν μπορεί να εφαρμόζεται με συνεχή τρόπο, και αναπόφευκτα παράγονται μπλοκ δεδομένων με πεπερασμένο μέγεθος.

3.3 Turbo Coding

Η γενική ιδέα του Turbo Coding είναι η χρήση δύο Convolutional Encoders και αντίστοιχα δύο Convolutional Decoders οι οποίοι λειτουργούν παράλληλα. Το βασικό τους πλεονέκτημα είναι ότι ο κάθε decoder έχει τη δυνατότητα να δέχεται ως είσοδο την έξοδο του άλλου decoder. Με αυτόν τον τρόπο η αποκωδικοποίηση ενός πακέτου δεδομένων (frame) μπορεί να εκτελείται ταυτόχρονα από τους δύο decoders και να επαναλαμβάνεται μία ή περισσότερες φορές. Για κάθε επανάληψη, ο αριθμός λανθασμένων bit που διορθώθηκε αυξάνεται. Σε πολλές περιπτώσεις, μερικές επαναλήψεις αρκούν για να ανακτηθεί ολόκληρη η αρχική πληροφορία ενός frame, ακόμα και σε συνθήκες υψηλών τιμών θορύβου.

Οι Turbo Codes χρησιμοποιούνται στις τεχνολογίες κινητής τηλεπικοινωνίας 3G και 4G (UMTS, LTE standards) αλλά και σε δορυφορικές εφαρμογές. Γενικά εφαρμόζονται για την αξιόπιστη μετάδοση δεδομένων σε τηλεπικοινωνιακά συστήματα με περιορισμούς απόκρισης και εύρους ζώνης, σε κανάλια που παρατηρείται απώλεια δεδομένων λόγω θορύβου. Στη συνέχεια παρουσιάζεται η λειτουργία κωδικοποίησης και αποκωδικοποίησης δεδομένων, με χρήση Turbo Code.

3.3.1 Turbo Encoder

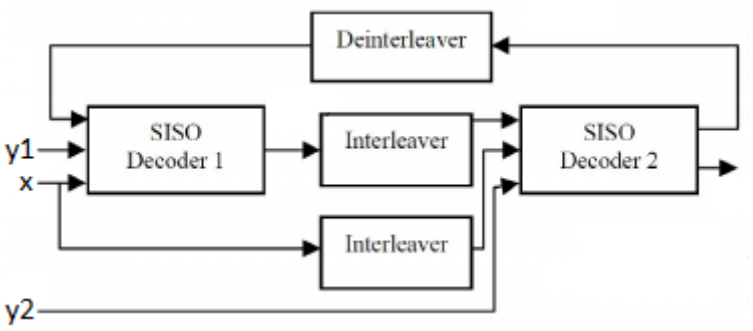


Εικόνα 3.4: Η δομή ενός Turbo Encoder.

Στην Εικόνα 3.4 φαίνεται ο τρόπος με τον οποίο συνδυάζονται δύο Convolutional Encoders για τη δημιουργία ενός Turbo Encoder. Οι encoders 1&2 είναι Recursive, Systematic Convolutional (RSC) Encoders, που διαθέτουν δύο εξόδους ο καθένας. Η μία έξοδος αναπαράγει την αρχική αλληλουχία bits που εισάγεται στον encoder, και από άλλη προκύπτει η κωδικοποιημένη λέξη. Υποθέτουμε ότι όπως και στις περισσότερες εφαρμογές και οι δύο encoders χρησιμοποιούν τον ίδιο αλγόριθμο

κωδικοποίησης, άρα έχουν και οι δύο Rate=1/(1+N). Η αρχική interleaved πληροφορία που εξάγεται από τον Encoder 2 δεν μεταδίδεται. Έτσι το συνολικό rate του Turbo Encoder είναι 1/(1+2N). Το κουτί με την ένδειξη π αποτελεί τον Interleaver του συστήματος. Παρατηρείται ότι ο Encoder 2 κωδικοποιεί την Interleaved έκδοση της αρχικής πληροφορίας. Τελικά στο κανάλι μεταδίδεται η αρχική αλληλουχία bits (x), η κωδικοποιημένη αλληλουχία bits (y1), και η κωδικοποιημένη αλληλουχία των Interleaved bits (y2), παράλληλα. Για το λόγο αυτό οι Turbo Codes χαρακτηρίζονται συχνά ως Parallel Concatenated Convolutional Codes.

3.3.2 Turbo Decoder



Εικόνα 3.5: Η δομή ενός Turbo Decoder

Οι τρεις αλληλουχίες που μεταδόθηκαν εισάγονται στον Turbo Decoder. Όπως φαίνεται στην Εικόνα3.5, η δομή του είναι αντίστοιχη με αυτή του Turbo Encoder. Οι δύο Convolutional Decoders είναι SISO (Soft Input, Soft Output). Αυτό σημαίνει ότι δέχονται τα bits εισόδου διαμορφωμένα σε ένα συγκεκριμένο διάστημα τιμών [-M,M] ενώ τα δεδομένα εξόδου λαμβάνουν τιμές σε ένα αντίστοιχο διάστημα. Όσο πιο κοντά στην μεγαλύτερη τιμή του διαστήματος είναι η τιμή ενός συμβόλου, τόσο πιο πιθανό είναι το αντίστοιχο bit να έχει την τιμή ‘1’. Αντίστοιχα, όσο πιο κοντά στη μικρότερη τιμή του διαστήματος είναι η τιμή ενός συμβόλου, τόσο πιο πιθανό είναι το bit να έχει την τιμή ‘0’. Όσο πιο κοντά είναι η τιμή ενός συμβόλου στην τιμή 0 του διαστήματος [-M,M], τόσο μεγαλύτερη είναι η αβεβαιότητα για την τιμή του bit. Οι τιμές που παίρνουν τα σύμβολα μετά την αποκωδικοποίηση προκύπτουν από το Λογάριθμο του Λόγου Πιθανοφάνειας (Logarithm of the Likelihood Ratio - LLR), ο οποίος υπολογίζεται από τους decoders. Οι τιμές αυτές αφενός χρησιμεύουν ως είσοδο για τους decoders (για την επανάληψη της αποκωδικοποίησης), αφετέρου, χρησιμοποιούνται για την απόφαση της τιμής του κάθε αποκωδικοποιημένου συμβόλου (hard decision).

Έτσι, οι αλληλουχίες συμβόλων x και y1 εισάγονται στον Decoder 1. Από αυτόν παράγεται η πρώτη αποκωδικοποιημένη αλληλουχία. Για την αντίστοιχη διαδικασία του Decoder 2 χρειάζεται όλες οι αλληλουχίες εισόδου να είναι interleaved. Έτσι, η αλληλουχία x που περιέχει την αρχική πληροφορία εισάγεται στον Interleaver, και από την έξοδο του τελευταίου οδηγείται στον Decoder 2. Η αλληλουχία y2 είναι ήδη interleaved, οπότε εισάγεται στον Decoder 2 ως έχει.

Εκτός από τις δύο αλληλουχίες για την αρχική και την κωδικοποιημένη πληροφορία, οι Decoders δέχονται ως είσοδο και την αλληλουχία τιμών LLR που προέκυψαν από την αποκωδικοποίηση. Στην αποκωδικοποιημένη αλληλουχία σε μορφή LLR που

προκύπτει από τον Decoder 1 εφαρμόζεται Interleaving και ύστερα εισάγεται στον Decoder 2. Αντίστοιχα, οι τιμές LLR που προκύπτουν από τον Decoder 2 εισάγονται στον Deinterleaver, ο οποίος εκτελεί την αντίστροφη διαδικασία του interleaving. Μετά το deinterleaving, οι τιμές εισάγονται στον Decoder 1. Με αυτό τον τρόπο δίνεται η δυνατότητα στον Turbo Decoder να επαναλαμβάνει τη διαδικασία αποκωδικοποίησης, διορθώνοντας κάθε φορά τα λανθασμένα σύμβολα που έχουν απομείνει. Εκτός από την έξοδο για τις τιμές LLR, ο Decoder 2 έχει μία δεύτερη έξοδο για τα αποκωδικοποιημένα bits, που πλέον έχουν τιμές ‘0’ ή ‘1’ (hard decision).

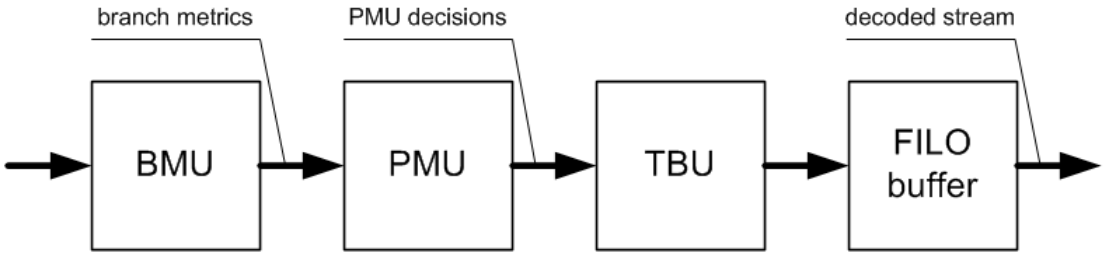
3.3.3 Viterbi Decoder

Ο πιο συνηθισμένος αλγόριθμος για την αποκωδικοποίηση ενός Convolutional κώδικα είναι ο Viterbi. Παρόλα αυτά, η παραδοσιακή εκδοχή του Viterbi δεν μπορεί να χρησιμοποιηθεί σε εφαρμογές Turbo Coding. Ο λόγος που συμβαίνει αυτό είναι ότι η έξοδος του Viterbi παράγει τιμές για τα αποκωδικοποιημένα bits με hard decisions (τιμές ‘0’ ή ‘1’ – Hard Output). Κάτι τέτοιο μπορεί να είναι αποδοτικό σε Convolutional Coding FEC, όμως η φύση των Turbo Codes προϋποθέτει ότι η έξοδος του ενός επιμέρους decoder θα πρέπει να παράγει τιμές πιθανοφάνειας (συνήθως λογαριθμικές), έτσι ώστε αυτές να μπορέσουν να εισαχθούν στον άλλο επιμέρους decoder. Για τον λόγο αυτό έχουν αναπτυχθεί decoders που βασίζονται στον τρόπο λειτουργίας του Viterbi, για να παραγάγουν τιμές που συμβολίζουν την πιθανότητα ένα bit να έχει τιμή ‘0’ ή ‘1’ (Soft Output). Η χρήση αποκωδικοποιητών SISO (Soft Input – Soft Output) είναι απαραίτητη για την λειτουργία δύο Constituent Decoders που αλληλοτροφοδοτούνται, και έχουν τη δυνατότητα να αποκωδικοποιούν το ίδιο frame επανειλημμένα, για τη βελτιστοποίηση της ανίχνευσης και διόρθωσης λανθασμένων bits.

Το κοινό στοιχείο της οικογένειας soft output αποκωδικοποιητών που βασίζονται στον Viterbi είναι λειτουργία τριών μονάδων υπολογισμού. Η πρώτη (Branch Metrics Unit - BMU) υπολογίζει τις ευκλείδειες αποστάσεις μεταξύ πιθανών συμβόλων. Η δεύτερη (Path Metric Unit - PMU), χρησιμοποιεί τις τιμές που υπολόγισε η BMU για να υπολογίσει την πιθανότητα για κάθε βήμα ενός αποδεκτού μονοπατιού, με βάση ένα διάγραμμα trellis. Η τρίτη (Traceback Unit - TBU), εκτελεί την ίδια διαδικασία με τη PMU, όμως ξεκινάει από το τέλος του μονοπατιού για να καταλήξει στην αρχή. Για να μπορεί να εκτελεστεί η τελευταία διαδικασία, θα πρέπει να έχει τελειώσει ο υπολογισμός της PMU, έτσι ώστε να υπάρχει ένα αποδεκτό μονοπάτι με βάση το οποίο θα εκτελέσει τους αντίστοιχους υπολογισμούς η TBU.

Τελικά, μία μονάδα υπολογισμού των τιμών LLR, αναλαμβάνει να συγκρίνει τα δεδομένα που προέκυψαν από τις μετρήσεις των τριών μονάδων που προαναφέρθηκαν, για να παραγάγει τις τιμές LLR που είτε θα χρησιμοποιηθούν για την τελική αποκωδικοποίηση των bit (hard decision), είτε θα ανατροφοδοτηθούν στο σύστημα για περεταίρω έλεγχο λαθών. Οι διαφοροποιήσεις μεταξύ του κάθε decoder που δομείται όπως περιγράφηκε προηγουμένως έχουν να κάνουν με την ακρίβεια στους υπολογισμούς της κάθε ομάδας και τον τρόπο με τον οποίο υπολογίζονται οι τιμές

LLR. Οι αποκωδικοποιητές που εξετάστηκαν και θα παρουσιαστούν στη συνέχεια χαρακτηρίζονται ως απλοποιημένοι (simplified), επειδή χρησιμοποιούν αλγορίθμους χαμηλής σχετικά πολυπλοκότητας. Αυτό έχει σαν αποτέλεσμα τη μειωμένη ακρίβεια στις μετρήσεις που πραγματοποιούν, σε σχέση με τη βέλτιστους υπολογισμούς που θα μπορούσαν να επιτύχουν. Το τελευταίο αντισταθμίζεται μέσω των επαναληπτικών αποκωδικοποιήσεων ενός frame, που βελτιστοποιούν τη διαδικασία ελέγχου λαθών, όπως θα φανεί στη συνέχεια.



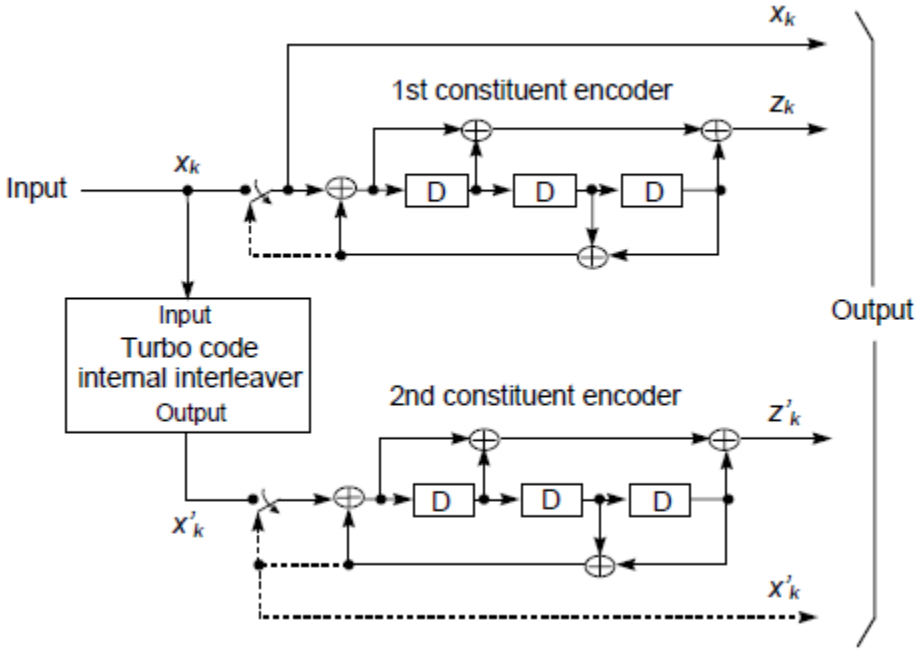
Εικόνα 3.6: Μια κοινή αρχιτεκτονική για την υλοποίηση ενός Viterbi Decoder σε hardware

4 Υλοποίηση Ενός Turbo Code Decoder

Στο κεφάλαιο αυτό παρουσιάζεται αρχικά ο encoder που υλοποιήθηκε στην [9], στον οποίο βασίστηκε η ανάπτυξη ενός αντίστοιχου decoder. Στη συνέχεια αναλύονται μία σειρά από Turbo Decoders, οι οποίοι είναι βασισμένοι στον αλγόριθμο Viterbi που παρουσιάστηκε στο προηγούμενο κεφάλαιο. Από τους decoders αυτούς αναλύεται η λειτουργία των BCJR και Max-Log MAP. Παρουσιάζεται η διαδικασία προσομοίωσής τους, καθώς και τα αποτελέσματα που προέκυψαν. Από τα αποτελέσματα αυτά εξάγονται συμπεράσματα για τους δύο decoders, την αποδοτικότητά τους και τις μεταξύ τους διαφορές.

4.1 Turbo Encoder

Στις [6], [9] προτείνεται η χρήση ενός απλοποιημένου Turbo Encoder για την υλοποίηση Turbo Coding σε περιβάλλον ΑΔΑ. Ως βάση χρησιμοποιείται το σχήμα κωδικοποίησης του πρότυπου UMTS με κάποιες τροποποιήσεις και διαφορετικές παραμέτρους. Αρχικά, επιλέχθηκε σταθερό μήκος εισόδου (frame size) ίσο με 160 bits. Χρησιμοποιήθηκε ένας relative prime interleaver με πολυώνυμο της μορφής i(k)= 3 + 23k mod 160. Όπως αναφέρθηκε σε προηγούμενο κεφάλαιο, στην συγκεκριμένη υλοποίηση δεν προτιμήθηκε να γίνει χρήση puncturing. Η δομή του συγκεκριμένου Encoder φαίνεται στην παρακάτω εικόνα:



Εικόνα 4.1: Η δομή του Turbo Encoder

Ο συγκεκριμένος Turbo Encoder υλοποιεί ένα Παράλληλα Συνδυαζόμενο Συνελικτικό Κώδικα (Parallel Concatenated Convolutional Code – PCCC) όπως αυτοί αναλύθηκαν στο Κεφάλαιο 3. Οι δύο επιμέρους encoders είναι recursive και systematic (RSC), 8 καταστάσεων. Τα δομικά στοιχεία των encoders είναι modulo-2 adders που υλοποιούνται με πύλες XOR και D flip-flops που λειτουργούν ως καταχωρητές ολίσθησης μεγέθους 1 bit. Αξίζει να σημειωθεί ότι οι διακεκομμένες γραμμές του παραπάνω σχήματος αφορούν τη διαδικασία τερματισμού της κωδικοποίησης, ενώ η έξοδος x'_kδεν υπολογίζεται στις εξόδους του συστήματος αφού τα δεδομένα που προκύπτουν από αυτή δεν μεταδίδονται στο κανάλι. Έτσι ο συνολικός ρυθμός κωδικοποίησης του Turbo Encoder είναι 1/3. Τα πολυώνυμα που υλοποιούν τον κάθε επιμέρους encoder είναι της μορφής:

$$g_0(D)=1+D^2+D^3 \quad \text{και} \quad g_1(D)=1+D+D^3,$$

για το feedback και feedforward αντίστοιχα. Από αυτά προκύπτει η εξής συνάρτηση μεταφοράς:

$$G(D) = \left[1, \frac{g_1(D)}{g_0(D)}\right] = \left[1, \frac{1 + D + D^3}{1 + D^2 + D^3}\right]$$

4.2 Turbo Decoders

Η οικογένεια των Turbo Decoders, όπως αναλύθηκε στο προηγούμενο κεφάλαιο κάνει χρήση δύο επιμέρους Convolutional Decoders όπου ο καθένας χρησιμοποιεί ως είσοδο την έξοδο του άλλου με σκοπό τις πολλαπλές επαναλήψεις της διαδικασίας αποκωδικοποίησης. Οι επιμέρους αυτοί Decoders λειτουργούν με βάση τον κανόνα απόφασης Μέγιστης A Posteriori Πιθανότητας (Maximum A Posteriori Probability - MAP) για την αποκωδικοποίηση μίας αλληλουχίας συμβόλων. Συνήθως οι τελικές τους αποφάσεις προκύπτουν από το λογάριθμο του λόγου της πιθανότητας ένα σύμβολο να έχει την τιμή ‘1’ προς την πιθανότητα ένα σύμβολο να έχει την τιμή ‘0’. Για τον λόγο αυτό οι αλγόριθμοι που υλοποιούν τους αποκωδικοποιητές αυτούς χαρακτηρίζονται ως Log MAP. Η χρήση λογαρίθμου για τον υπολογισμό αυτό μειώνει την επεξεργαστική πολυπλοκότητα των αλγορίθμων. Όπως αναφέρεται στην [23] (Κεφ. 4.4.1), οι δύο βασικές διαφοροποιήσεις μεταξύ του Viterbi και των Log MAP αλγορίθμων είναι:

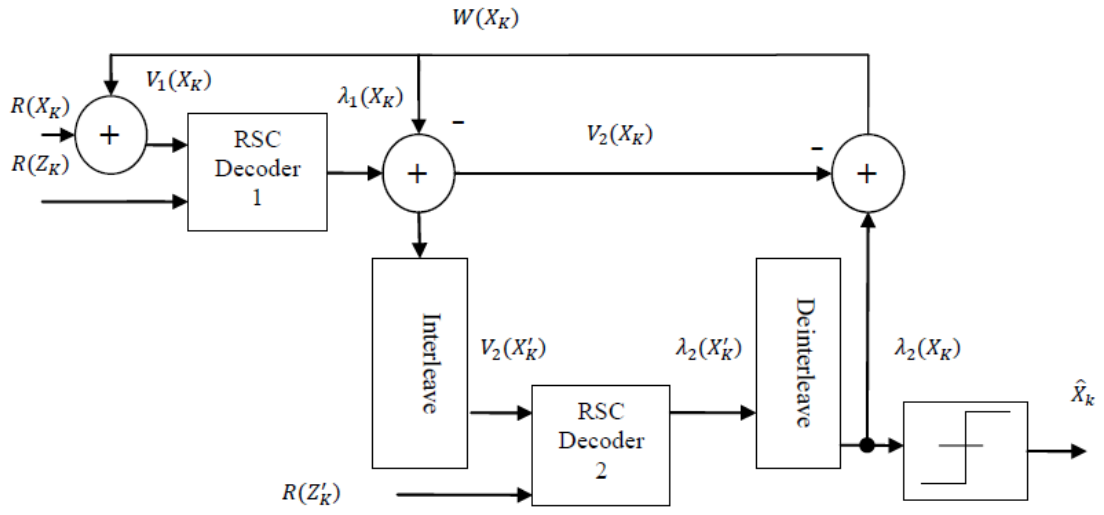
- Το διάγραμμα trellis χρησιμοποιείται όχι μόνο για τον υπολογισμό των Forward State Metrics (τα αντίστοιχα Path Metrics του Viterbi), αλλά και των Backward State Metrics (το path που υπολογίζεται από την Traceback Unit του Viterbi). Άρα στους αλγόριθμους Log MAP το Trellis σαρώνεται από αριστερά προς τα δεξιά και από τα δεξιά προς τα αριστερά.

- Για τον υπολογισμό των Forward και Backward Metrics δεν χρησιμοποιείται μονάδα πρόσθεσης-σύγκρισης-επιλογής (add-compare-select - ACS) όπως

στον Viterbi, αλλά η συνάρτηση max*, γνωστή και ως αλγόριθμος Jacobi [24], της οποίας η γενική μορφή είναι:

$$max*(a,b) = \ln(e^a + e^b) = \max(a,b) + \ln(1 + e^{-|a-b|})$$

Οι διαφοροποιήσεις μεταξύ των διάφορων Log MAP αλγορίθμων έχουν να κάνουν με το πώς εφαρμόζεται ο αλγόριθμος Jacobi. Κάποιοι Decoders υλοποιούν τη συνάρτηση max* με μία πιο αφαιρετική προσέγγιση. Με αυτό τον τρόπο μειώνεται η πολυπλοκότητα στους αλγορίθμους που χρησιμοποιούνται για τον υπολογισμό των Forward και Backward State Metrics. Έτσι στην οικογένεια των Log MAP αλγορίθμων ανήκει ο κλασσικός Log MAP, ο Linear-Log MAP, ο Constant-Log MAP και ο Max-Log MAP. Σε διάφορες περιπτώσεις, ο αλγόριθμος MAP αποκαλείται BCJR [24]. Ο BCJR διαφέρει από τον Viterbi όπως και οι Log MAP, δηλαδή κάνει χρήση του Trellis όπως και οι Log MAP αλγόριθμοι, και δεν χρησιμοποιεί ACS unit (ούτε όμως και τον αλγόριθμο Jacobi). Σε υλοποιήσεις όπως αυτή που θα παρουσιαστεί στη συνέχεια, ο BCJR αποφασίζει για την τιμή των συμβόλων στο πεδίο των λογαρίθμων. Για το λόγο αυτό μπορεί να θεωρηθεί ότι ανήκει στους Log MAP αλγόριθμους. Η δομή των παραπάνω decoders παρουσιάζεται στο παρακάτω σχήμα [23], όπου X^K, X'_K είναι οι συμβολοσειρές εισόδου χωρίς κωδικοποίηση (interleaved και deinterleaved αντίστοιχα), Z_K, Z'_K είναι οι κωδικοποιημένες συμβολοσειρές εισόδου (interleaved και deinterleaved αντίστοιχα), με R συμβολίζονται οι συμβολοσειρές που εισάγονται στο σύστημα, με V οι συμβολοσειρές που παράγονται μετά από αποκωδικοποίηση και αποτελούν είσοδο για τον επόμενο decoder, με λ οι συμβολοσειρές με τιμές στο πεδίο των λογαρίθμων ενώ η συμβολοσειρά $\widehat{X_k}$ προκύπτει μετά από hard decision επί των τιμών λ.



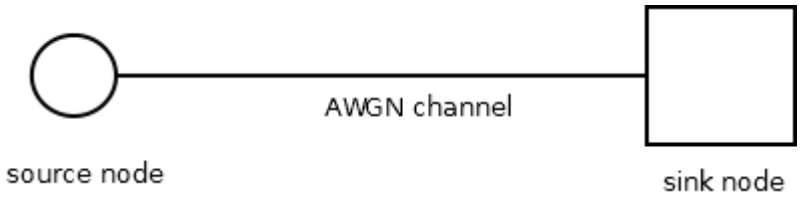
Εικόνα 4.23: Η δομή ενός Turbo Decoder

Για τη διαδικασία προσομοίωσης ενός Turbo Decoder επιλέχθηκαν οι αλγόριθμοι BCJR και Max-Log MAP. Η επιλογή έγινε με βασικό γνώμονα ένα χαμηλό επίπεδο πολυπλοκότητας, έτσι ώστε τα συστήματα που χρησιμοποιούνται σε εφαρμογές ΑΔΑ να μπορούν να υποστηρίξουν τους αλγορίθμους αυτούς. Ο σκοπός της διαδικασίας

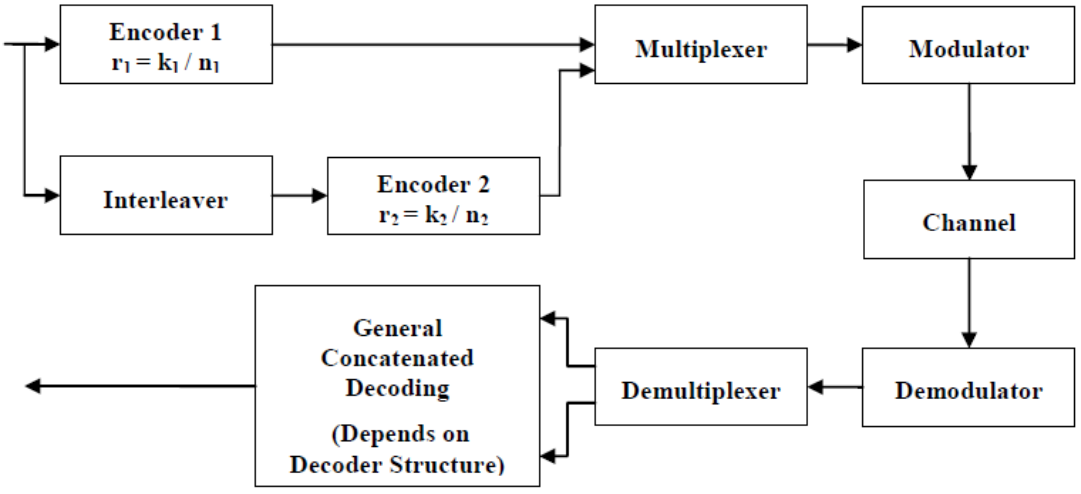
προσομοίωσης συνολικά, ήταν να βρεθεί ένας αλγόριθμος εύκολα υλοποιήσιμος, που θα παρουσίαζε ισορροπία μεταξύ της πολυπλοκότητας και της απόδοσης του. Σημαντικό κριτήριο για την επιλογή του κατάλληλου αλγορίθμου ήταν η αποδοτικότητά του σε συνθήκες αντίστοιχες με αυτές των εφαρμογών ΑΔΑ. Έτσι το περιβάλλον προσομοίωσης των αλγορίθμων διαμορφώθηκε από παραμέτρους αντίστοιχες με αυτές υπό τις οποίες αναπτύχθηκε ο Turbo Encoder της [9].

4.3 Turbo Decoders προς Υλοποίηση και Διαδικασία Προσομοίωσης

Για τη διαδικασία προσομοίωσης των decoders εφαρμόστηκε το εξής πείραμα. Σε ένα ΑΔΑ με δρομολόγηση single-hop, ένας κόμβος-πηγή αποστέλλει frames δεδομένων στον κόμβο-συλλέκτη. Ο αποστολέας χρησιμοποιεί ένα Turbo Encoder για την κωδικοποίηση των frames, και ο παραλήπτης χρησιμοποιεί τον αντίστοιχο Turbo Decoder για την αποκωδικοποίησή τους. Το κανάλι μετάδοσης ακολουθεί το μοντέλο AWGN (Additive White Gaussian Noise), το οποίο σημαίνει ότι στο κανάλι προστίθεται λευκός θόρυβος σταθερής φασματικής πυκνότητας, με Gaussian κατανομή.



Εικόνα 4.4: Το σχήμα της προσομοίωσης



Εικόνα 4.5: Η δομή του Channel Coding με χρήση Turbo Code

Για την προσομοίωση του πειράματος χρησιμοποιήθηκε η βιβλιοθήκη CML (Coded Modulation Library) της MATLAB [21]. Η βιβλιοθήκη αυτή προσφέρει μία σειρά από διάφορες παραμέτρους για την προσομοίωση τηλεπικοινωνιακών συστημάτων. Ενδεικτικά, υπάρχουν διάφορες επιλογές FEC Codes (Convolutional, Turbo, Block Turbo Codes), καθώς και πληθώρα decoders (Log MAP, Max-Log MAP, Constant-

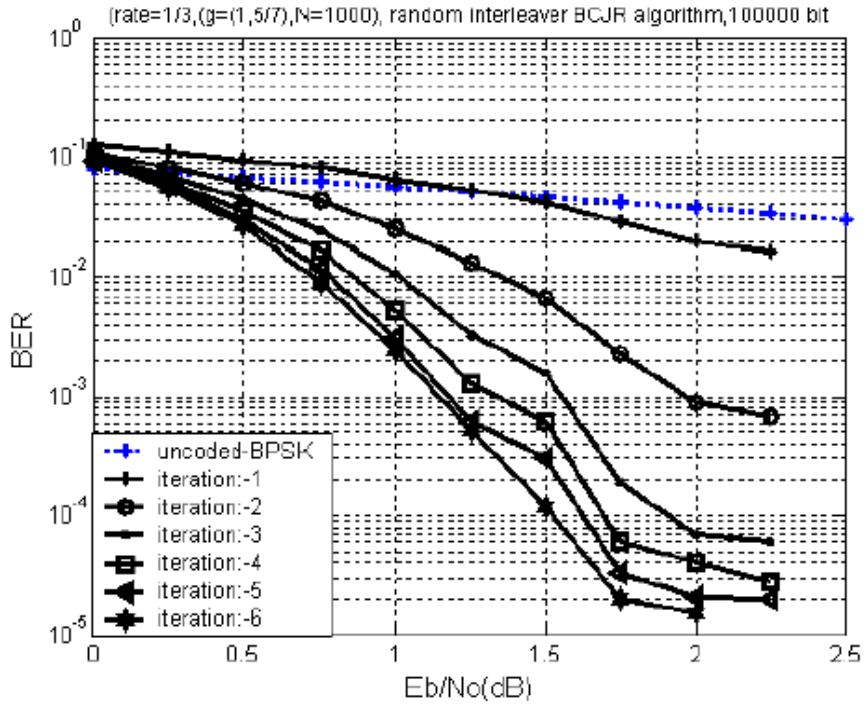
Log MAP, Linear-Log MAP SISO decoder, Viterbi). Επιπλέον, διαθέτει διάφορες εναλλακτικές για τον τύπο καναλιών, τη διαμόρφωση, τη χρήση puncturing και μία σειρά από διαθέσιμους interleavers. Τέλος, για τη διαδικασία ARQ χρησιμοποιήθηκε το κομμάτι κώδικα Matlab που την υλοποιεί, από την [9].

Η διεξαγωγή του πειράματος επαναλαμβάνεται για διάφορες τιμές θορύβου, οι οποίες περιγράφονται από τον λόγο της ισχύος του σήματος προς την ισχύ του θορύβου ($P_{signal}/P_{noise)}$ δηλαδή το Signal-to-Noise Ratio (SNR) σε dB. Συγκεκριμένα, εξετάζονται οι τιμές του διαστήματος 0-9 SNR_{dB} με βήμα 0,5 ενώ ο τύπος SNR που χρησιμοποιείται αφορά τον λόγο της ενέργειας ανά bit προς τη φασματική πυκνότητα ισχύος (E_b/N_o). Η διαμόρφωση του σήματος είναι QPSK και το μέγεθος του κάθε frame είναι 160 bits. Όσον αφορά τον decoder, χρησιμοποιήθηκε ο interleaver με πολυώνυμο i(k) = 3 + 23k mod 160 της [9], ενώ δεν έγινε χρήση puncturing. Η διαδικασία αποκωδικοποίησης συμβαίνει για τέσσερις επαναλήψεις, με την κάθε επανάληψη να σημαίνει επεξεργασία των συμβολοσειρών και από τους δύο επιμέρους decoders. Στην περίπτωση που ένα frame δεν μπορέσει να αποκωδικοποιηθεί σωστά μετά από τέσσερις επαναλήψεις, προσομοιώνεται η επανάληψη της αποστολής του. Η διαδικασία αυτή (Automatic Repeat Request - ARQ) μπορεί να επαναληφθεί έως τέσσερις φορές. Τα αποτελέσματα που παρουσιάζονται αφορούν την αποστολή 10^5 αυθεντικών frames. Έτσι, κάθε αυθεντικό frame ενδέχεται να αποσταλθεί και να αποκωδικοποιηθεί συνολικά πέντε φορές.

Στη συνέχεια παρουσιάζονται οι αλγόριθμοι BCJR και Max-Log MAP, καθώς και τα αποτελέσματα που προέκυψαν μετά την προσομοίωσή τους. Για την αξιολόγηση των αποτελεσμάτων χρησιμοποιούνται οι λόγοι BER και FER, δηλαδή ο λόγος των λανθασμένων bits προς τα συνολικά bits που μεταδόθηκαν, και ο λόγος των λανθασμένων frames (frames που μετά την αποκωδικοποίηση περιέχουν τουλάχιστον ένα λανθασμένο bit) προς τα συνολικά frames που μεταδόθηκαν, αντίστοιχα. Οι τιμές BER και FER παρουσιάζονται στα γραφήματα για κάθε τιμή SNR_{dB} στην οποία αντιστοιχούν. Για την αξιολόγηση της λειτουργίας της τεχνικής ARQ, παρουσιάζεται το επί τοις εκατό ποσοστό των αυθεντικών frames που αποκωδικοποιήθηκαν ανεπιτυχώς (λανθασμένα frames), σε σχέση με το σύνολο των αυθεντικών frames, για κάθε τιμή SNR_{dB}.

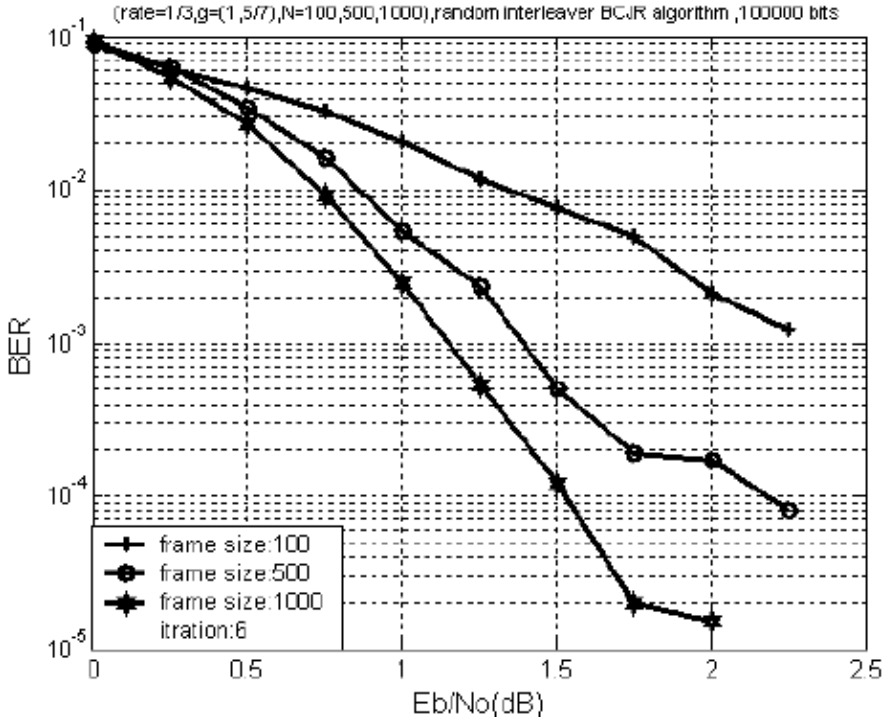
4.3.1 BCJR decoder

Ο Bahl, Cocke, Jelinek and Raviv (BCJR) Decoder, παρουσιάζεται σε διάφορα σημεία της βιβλιογραφίας και συναντάται συχνά σε εφαρμογές Turbo Decoding. Στην [19] γίνεται σύγκριση μεταξύ του αποκωδικοποιητή αυτού και ενός Viterbi Decoder, τροποποιημένου έτσι ώστε να παράγει soft output (Soft Output Viterbi Algorithm - SOVA). Εκεί παρατηρείται αποδοτική λειτουργία του BCJR, ενώ επιβεβαιώνεται η βελτιστοποίηση της διαδικασίας αποκωδικοποίησης, μέσω πολλαπλών επαναλήψεων. Όπως φαίνεται στο παρακάτω γράφημα της [19], ο λόγος BER για AWGN θόρυβο μεγέθους 2dB κυμαίνεται μεταξύ 10^{-3} και 10^{-5}, όσο επαναλαμβάνεται η διαδικασία αποκωδικοποίησης. Οι τιμές αυτές του BER κρίνονται σχετικά ικανοποιητικές, αν αναλογιστεί κανείς το επίπεδο θορύβου στο κανάλι.



Γράφημα 4.1: Απόδοση BER για διάφορες επαναλήψεις του BCJR decoder

Βέβαια, το παραπάνω γράφημα αντιστοιχεί στην προσομοίωση αποστολής frames μεγέθους 10^3 bits, αρκετά μεγαλύτερου από το μέγεθος frame που χρησιμοποιείται σε εφαρμογές ΑΔΑ. Επίσης γίνεται αντιληπτό ότι ο συγκεκριμένος decoder λειτουργεί πιο αποδοτικά όσο το μέγεθος frame αυξάνεται, όπως φαίνεται παρακάτω:



Γράφημα 4.2: Απόδοση BER για διάφορες μεγέθη frame του BCJR decoder

Η απόκλιση στην απόδοση του decoder μεταξύ των διάφορων μεγεθών frame δεν είναι αμελητέα, παρόλα αυτά δεν κρίθηκε απαγορευτική για τη διεξαγωγή της προσομοίωσης. Επιπλέον, η χρήση BCJR decoder παρουσιάζεται πιο αποδοτική όταν συνδυάζεται με συμμετρικό Turbo Encoder, που χρησιμοποιεί δηλαδή δύο όμοιους επιμέρους Convolutional Encoders, από όταν συνδυάζεται με ασύμμετρο Turbo Encoder, όπου οι επιμέρους encoders υλοποιούν διαφορετικά πολυώνυμα, και κωδικοποιούν με διαφορετικό rate. Αντίστοιχα, ο BCJR είναι πιο αποδοτικός όταν συνδυάζεται με Turbo Encoder που κωδικοποιεί με rate 1/3, απ' ότι με άλλα rates. Οι δύο αυτοί παράγοντες συνέβαλαν στην επιλογή του BCJR decoder, αφού παρουσιάστηκε αποδοτική λειτουργία του decoder όταν χρησιμοποιείται σε συνδυασμό με συμμετρικό Turbo Encoder με rate 1/3, όπως αυτός της [9]. Για τους λόγους αυτούς εκτιμήθηκε ως αξιόπιστη η επιλογή ενός χαμηλής πολυπλοκότητας BCJR Turbo Decoder για την προσομοίωση του σε περιβάλλον Matlab με σκοπό τη διεξαγωγή συμπερασμάτων όσων αφορά την υλοποίηση του.

Στη συνέχεια επιλέχθηκε η υλοποίηση σε κώδικα Matlab του BCJR decoder που παρουσιάζεται στην [20]. Ο encoder της συγκεκριμένης υλοποίησης αποτελείται από δύο RSC encoders με την εξής συνάρτηση μεταφοράς:

$$G(D) = \left[1, \frac{1 + D + D^2}{1 + D^2}\right]$$

Παρατηρείται ότι ο encoder της υλοποίησης αυτής διαθέτει διαφορετική συνάρτηση μεταφοράς από τον encoder που αποτελεί βάση για την εργασία. Παρόλα αυτά, είναι και οι δύο συμμετρικοί RSC encoders και έχουν τον ίδιο ρυθμό κωδικοποίησης. Έτσι αποφασίστηκε να χρησιμοποιηθεί ο encoder της [20] για την αξιολόγηση του αντίστοιχου decoder, χωρίς έτσι να χρειαστεί κάποια αλλαγή στη λειτουργία του τελευταίου.

Η ακριβής λειτουργία του αποκωδικοποιητή παρουσιάζεται στην [20]. Συνοπτικά, η απόφαση για το αν το k-οστό bit μιας ακολουθίας εισόδου $\overline{R}$ προκύπτει από την τιμή Log A Posteriori Probability Ratio (LAPPR) που υπολογίζεται ως εξής:

$$\Lambda^k = \log\left\{\frac{P(X^k=1|\overline{R})}{P(X^k=0|\overline{R})}\right\}$$

Για τον υπολογισμό της πιθανότητας ένα σύμβολο να έχει την τιμή i ('0' ή '1'), χρησιμοποιείται η παρακάτω εξίσωση:

$$P(\mathrm{X^k = i|\overline{R}}) = \frac{1}{\mathrm{P(R_0^{N-1})}} \sum_{\mathrm{m=0}}^{\mathrm{M-1}} \sum_{\mathrm{m'=0}}^{\mathrm{M-1}} \alpha_{\mathrm{k}}(\mathrm{m'}) \gamma_{\mathrm{k}}^{\mathrm{i}}(\mathrm{m', m}) \beta_{\mathrm{k}}(\mathrm{m})$$

όπου m είναι η κατάσταση που βρίσκεται ένα σύμβολο, σύμφωνα με το διάγραμμα Trellis (m=0,1,2,3 για τα στάδια 00, 01, 10, 11 αντίστοιχα), M είναι ο συνολικός αριθμός καταστάσεων και η είσοδος $\overline{\mathrm{R}}$ αναλύεται στις δύο ακολουθίες εισόδου $\overline{R_0 R_1}$.

Για τον υπολογισμού της πιθανότητας αυτής, χρειάζονται οι τιμές $\alpha_k(m')$ (forward state metrics) που σχετίζεται με την επόμενη κατάσταση του Trellis, $\beta_k(m)$ (backward state metrics) που σχετίζεται με την προηγούμενη κατάσταση του Trellis, και η $\gamma_k^i(m', m)$ (branch metrics) που συμβολίζει την πιθανότητα μετάβασης από την κατάσταση m' στην m για μία είσοδο i. Για τον υπολογισμό των τιμών α, β, γ χρησιμοποιούνται οι παρακάτω τύποι:

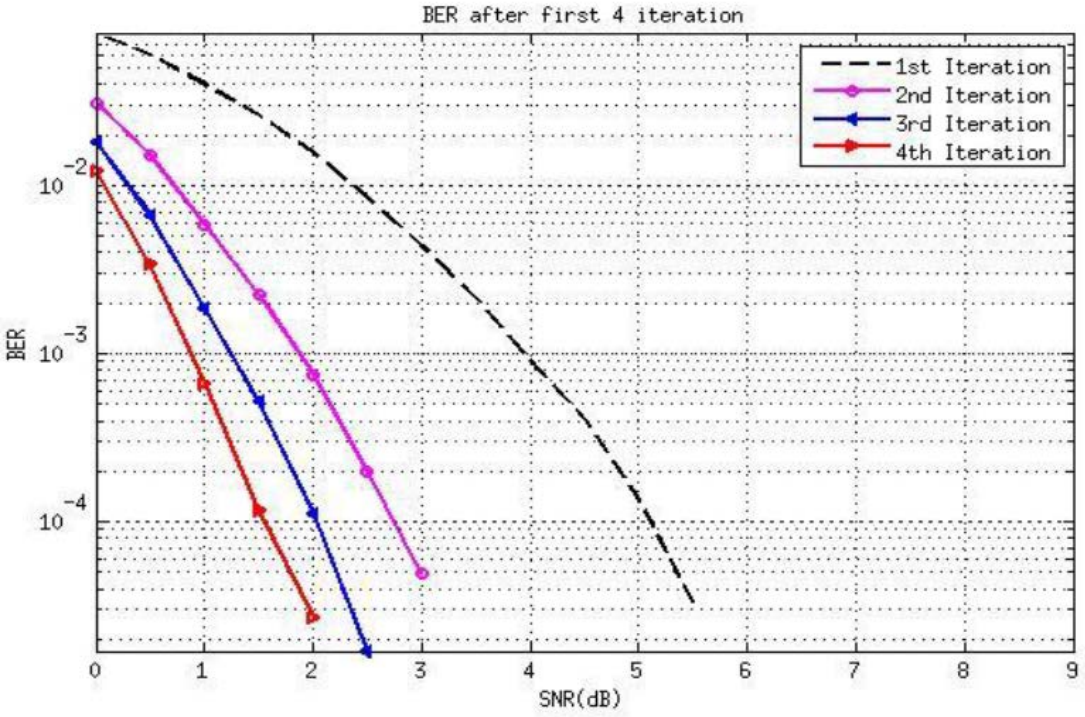
$$\alpha_k(m') = \sum_{m''=0}^{M-1} \sum_{i=0}^{1} \gamma_{k-1}^i(m'', m')\alpha_{k-1}(m'')$$

$$\beta_k(m) = \sum_{m'=0}^{M-1} \sum_{i=0}^{1} \gamma_{k+1}^i(m', m)\beta_{k+1}(m')$$

$$\gamma_k^i(m', m) = \frac{1}{\sqrt{2\pi\sigma^2}} exp\left(-\frac{1}{2\sigma^2}\left(R_0^k - (2i-1)\right)^2\right) X \frac{1}{\sqrt{2\pi\sigma^2}} exp\left(-\frac{1}{2\sigma^2}\left(R_1^k - (2C^k(m', m) - 1)\right)^2\right)$$

Ο υπολογισμός του γ προκύπτει από την a priori πιθανότητα του k-οστού συμβόλου να έχει την τιμή i, την πιθανότητα μετάβασης σε μία κατάσταση με βάση το Trellis, και την πιθανοφάνεια ενός συμβόλου. Για την πρώτη αποκωδικοποίηση από τον πρώτο εκ των δύο επιμέρους decoders, οι a priori πιθανότητες έχουν τιμή 0,5, από τη στιγμή που δεν υπάρχει από πριν κάποια ένδειξη για την τιμή του κάθε συμβόλου. Από την πρώτη αποκωδικοποίηση, εξάγονται a priori πιθανότητες από τις τιμές LAPPR για τις επόμενες επαναλήψεις, οι οποίες μεταδίδονται στον δεύτερο decoder. Με τον τρόπο αυτό, όσο συνεχίζονται οι επαναλήψεις αποκτάται περισσότερη πληροφορία για την πιθανή τιμή ενός bit.

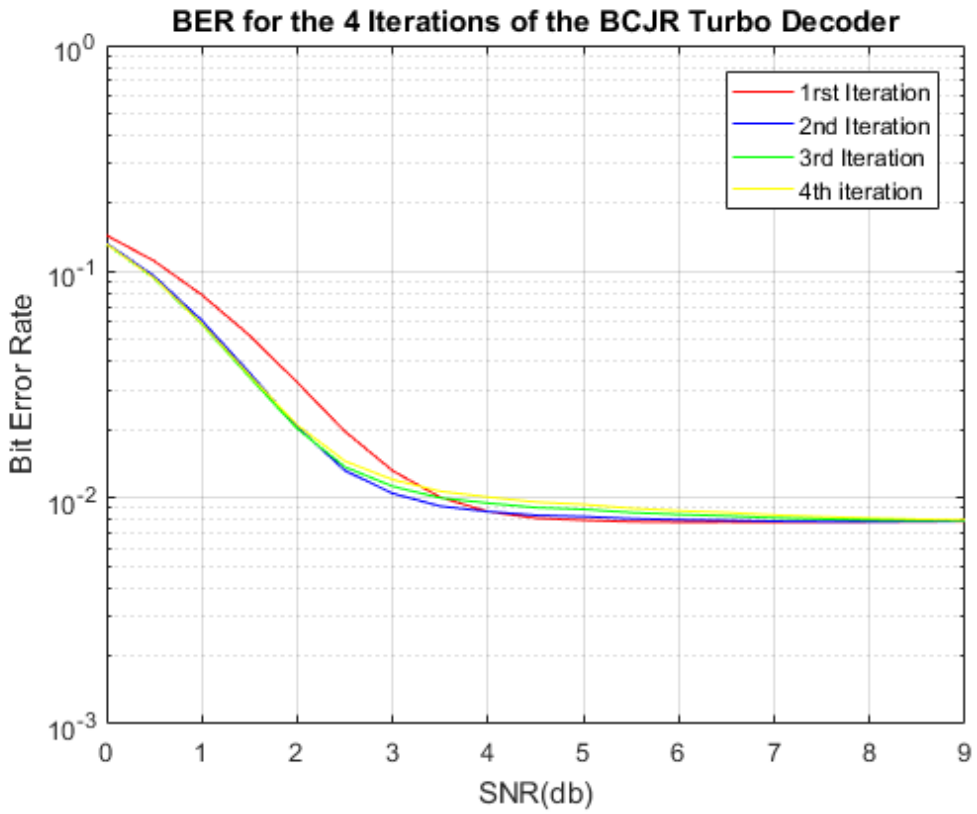
Τα αποτελέσματα που παρουσιάζονται στην [20] και αφορούν μεμονωμένα frames μεγάλου μεγέθους (10^6 bits) είναι ικανοποιητικά, αναφορικά με τον ρυθμό BER (Bit Error Rate):



Γράφημα 4.3: Απόδοση BER για την υλοποίηση BCJR

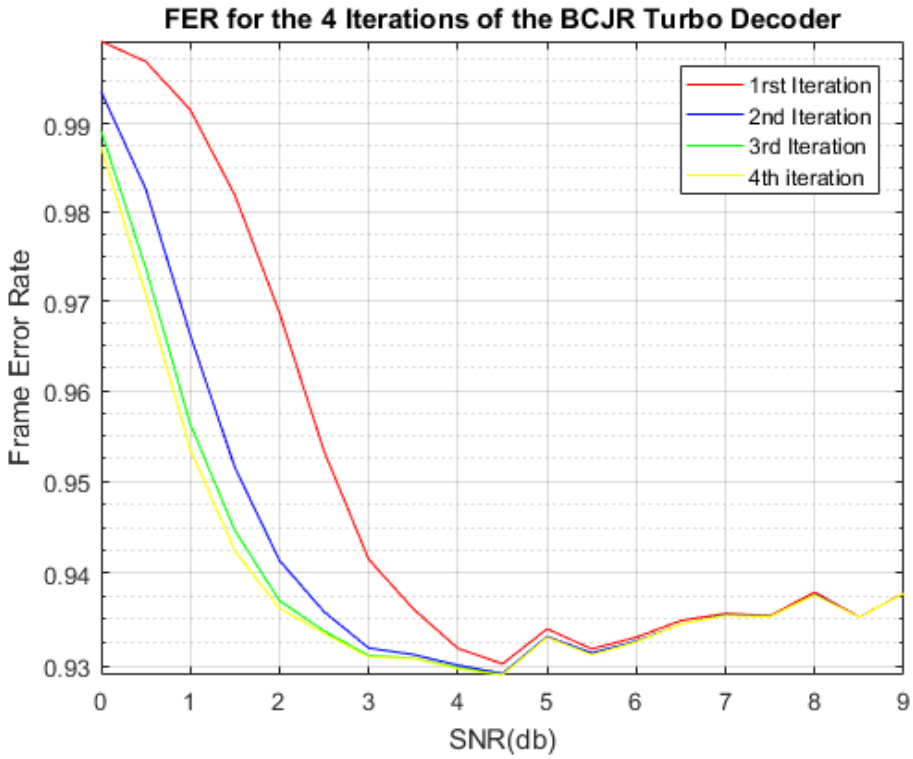
Παρατηρείται λοιπόν ότι για frames μεγάλου μεγέθους, ο decoder έχει τη δυνατότητα να διορθώνει λανθασμένα bits, κρατώντας τον αριθμό τελικών λαθών χαμηλά, σε συνθήκες υψηλού θορύβου (SNR = 1-3 SNR_{dB}, AWGN channel). Επιπλέον, οι πολλαπλές επαναλήψεις στην αποκωδικοποίηση βελτιστοποιούν τη λειτουργία του. Είναι ενδεικτικό ότι σε σχέση με την πρώτη αποκωδικοποίηση, η πρώτη επανάληψη μειώνει αισθητά τη συχνότητα λαθών στο κανάλι.

Για την συνέχεια της προσομοίωσης, ο κώδικας Matlab που υλοποιεί τον BCJR ενσωματώθηκε στη βιβλιοθήκη CML. Στη συνέχεια, ο BCJR decoder προσομοιώθηκε με τις κατάλληλες παραμέτρους, έτσι ώστε να προσεγγίζονται τα πρωτόκολλα επικοινωνίας ενός ΑΔΑ. Τα αποτελέσματα της προσομοίωσης αποστολής 10^4 frames με μέχρι τέσσερις επαναλήψεις αποστολής (ARQ) και τέσσερις επαναλήψεις αποκωδικοποίησης (Iterations) παρουσιάζονται παρακάτω:



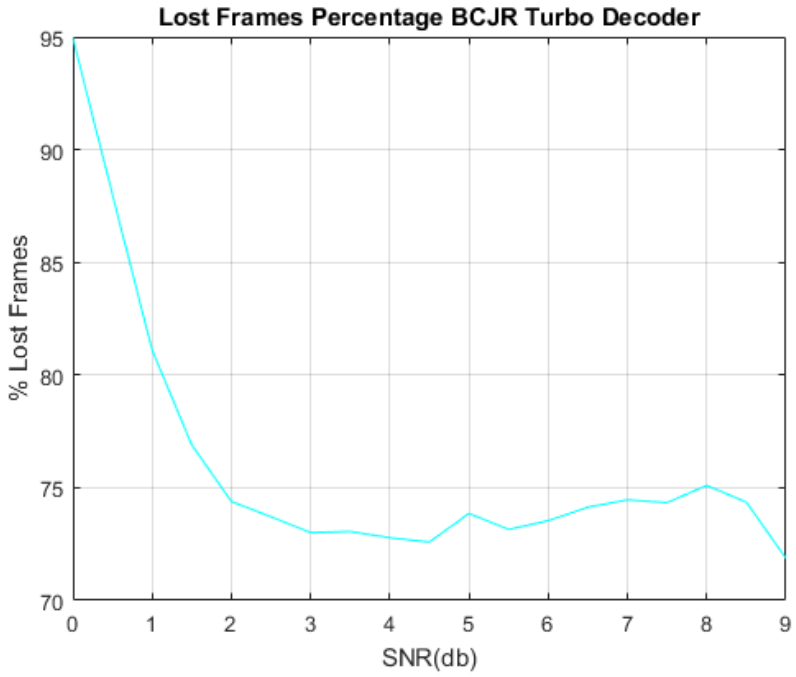
Γράφημα 4.4: Απόδοση BER του BCJR

Τα αποτελέσματα που προέκυψαν δεν ήταν τα αναμενόμενα. Η συγκεκριμένη υλοποίηση παρ' ότι είναι αποδοτική για περιπτώσεις μεγάλων frames, δεν λειτουργεί εξίσου ικανοποιητικά για το μέγεθος frame που αντιστοιχεί στο αντικείμενο της εργασίας. Έτσι παρατηρείται ότι το BER παραμένει σε πολύ υψηλά επίπεδα, ακόμα και για χαμηλές τιμές θορύβου. Κάτι τέτοιο ενδεχομένως οφείλεται στις διαφορετικές παραμέτρους της αυτόνομης υλοποίησης, από αυτές της [9]. Είναι ενδεικτικό ότι ακόμα και για χαμηλές τιμές θορύβου, ο decoder παρουσιάζει ρυθμό σφαλμάτων BER κοντά στο 10^{-2}, πράγμα που σημαίνει ότι για κάθε 100 bits που επεξεργάζονται, παραμένει ένα που δεν μπόρεσε να ανακτηθεί. Με μέγεθος frame 160bits, είναι αναμενόμενο κάθε frame να περιέχει τουλάχιστον ένα λανθασμένο bit, ακόμα και μετά την αποκωδικοποίηση. Κάτι τέτοιο φαίνεται στις τιμές του Frame Error Rate του αποκωδικοποιητή:



Γράφημα 4.5: Απόδοση FERτου BCJR

Στο παρακάτω γράφημα παρουσιάζεται το ποσοστό frames που δεν μπόρεσαν να μεταδοθούν επιτυχώς, ακόμα και μετά από 4 επαναλήψεις της αποστολής:



Εικόνα 4.6: Ποσοστό χαμένων framesγια τον BCJR

Από τα αποτελέσματα της προσομοίωσης έγινε προφανές ότι η συγκεκριμένη υλοποίηση δεν είναι στο ελάχιστο αποδοτική, σε σχέση με τις απαιτήσεις ενός

περιβάλλοντος ΑΔΑ. Είναι χαρακτηριστική η απόκλιση στην απόδοση του αποκωδικοποιητή, μεταξύ προσομοίωσης μεγάλων και μικρών frames, κάτι που είχε φανεί με μικρότερες βέβαια αποκλίσεις και στην [19]. Επειδή οι παράμετροι που αποτελούν τη βάση για τη συγκεκριμένη εργασία επιβάλλουν τηη προσομοίωση αποστολής μικρού μεγέθους frames, καθώς επιχειρείται η προσέγγιση ενός ρεαλιστικού περιβάλλοντος σε ένα Ασύρματο Δίκτυο Αισθητήρων, ο συγκεκριμένος decoder απορρίφθηκε.

4.3.2 Max-Log MAP Decoder

Η χρήση του Max-Log MAP decoder είναι διαδεδομένη σε εφαρμογές Turbo Decoding από συστήματα μειωμένων δυνατοτήτων, λόγω της χαμηλής πολυπλοκότητάς του. Η λειτουργία των Log MAP και Max-Log MAP decoders παρουσιάζεται στην [22]. Εκεί διεξάγονται κάποια πρώτα συμπεράσματα από την προσομοίωση ενός Max-Log MAP decoder για παραμέτρους σχετικά αντίστοιχες με το περιβάλλον της εργασίας. Παρατηρείται αποδοτική λειτουργία του αποκωδικοποιητή με μέγεθος frame 483bits, όχι ιδιαίτερα μεγαλύτερο δηλαδή από το επιθυμητό (160 bits). Επιπλέον, οι Log MAP αλγόριθμοι προσομοιώνονται στην [23] με παραμέτρους frame size 40, 100, 320 bits, AWGN channel, σύμφωνα με το πρότυπο UMTS, και μάλιστα κάνοντας χρήση της CML. Τα αποτελέσματα που προκύπτουν είναι ικανοποιητικά, και αποτελούν βάση για την ανάπτυξη ενός Max-Log MAP decoder. Τέλος, στην [24] αναλύεται θεωρητικά η διαφοροποίηση μεταξύ των BCJR και Max-Log MAP.

Για την προσομοίωση του Max-Log MAP σε περιβάλλον Matlab, βρέθηκε μία αυτόνομη υλοποίηση των Log MAP και Max-Log MAP αποκωδικοποιητών [25]. Σε ό,τι αφορά τον αποκωδικοποιητή Log MAP γενικά, η βασική διαφοροποίηση με τον BCJR είναι ο τρόπος με τον οποίο προκύπτουν οι τιμές α, β, γ και LLR. Η συνάρτηση max, παρουσιάζει τον τρόπο με τον οποίο χρησιμοποιούνται οι τιμές α, β, γ για τον υπολογισμό των LLR [22]:

$$\ln \Lambda_{t+1} = \max_{(m,m',X=1)} [\ln \gamma_{t+1}(m',m) + \ln \beta_{t+1}(m) + \ln \alpha_t(m',m)] - \max_{(m,m',X=-1)} [\ln \gamma_{t+1}(m',m) + \ln \beta_{t+1}(m) + \ln \alpha_t(m',m)]$$

Για τον τρόπο με τον οποίο λειτουργεί η max, παρουσιάζεται ο παρακάτω τύπος, που αφορά τον αλγόριθμο Log MAP, από την [24]:

$$max*(a,b) = \max(a,b) + \ln(1 + e^{-|a-b|})$$

Όπου γίνεται χρήση διορθωτικής συνάρτησης $(\ln(1 + e^{-|a-b|}) = f_c(|a-b|))$ η οποία παίρνει τιμές συνήθως από ένα Look Up Table. Στους αλγόριθμους Constant-Log MAP και Linear-Log MAP γίνεται χρήση μίας αντίστοιχης διορθωτικής

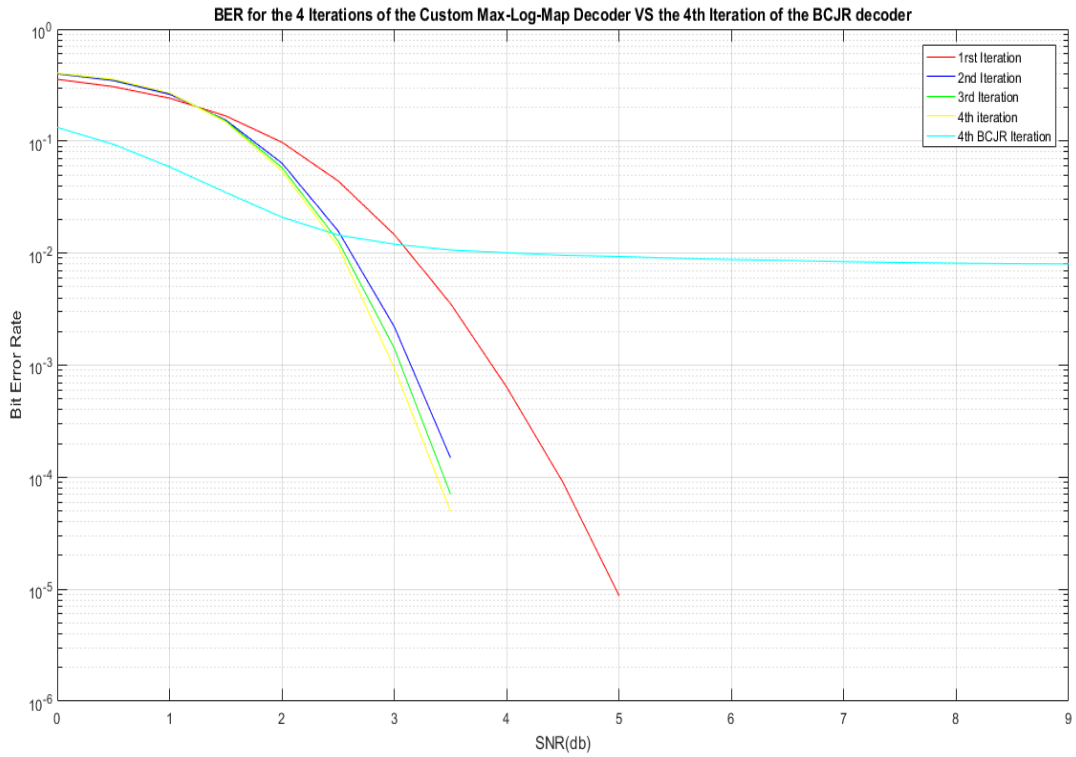
συνάρτησης [23]. Η χρήση διορθωτικής συνάρτησης παραλείπεται στον αλγόριθμο Max-Log MAP, όπου η max παίρνει της εξής απλοποιημένη μορφή [24]:

$$max*(a,b) \approx \max(a,b)$$

Έτσι, ο υπολογισμός των α, β (forward & backward metrics) γίνεται επίσης με χρήση της max. Εκτός από τις παραπάνω αλλαγές, η συγκεκριμένη απλοποιημένη υλοποίηση του Max-Log MAP αλγορίθμου, διαφοροποιείται στον υπολογισμό των τιμών γ. Για τα branch metrics παραλείπεται ο σύνθετος τύπος που χρησιμοποιήθηκε προηγουμένως, και ο υπολογισμός γίνεται με βάση τις ευκλείδειες αποστάσεις μεταξύ των συμβόλων. Στον υπολογισμό αυτό χρησιμοποιούνται ακόμα οι a priori τιμές που προκύπτουν από προηγούμενες επαναλήψεις της αποκωδικοποίησης. Για την πρώτη αποκωδικοποίηση οι τιμές αυτές είναι μηδενισμένες.

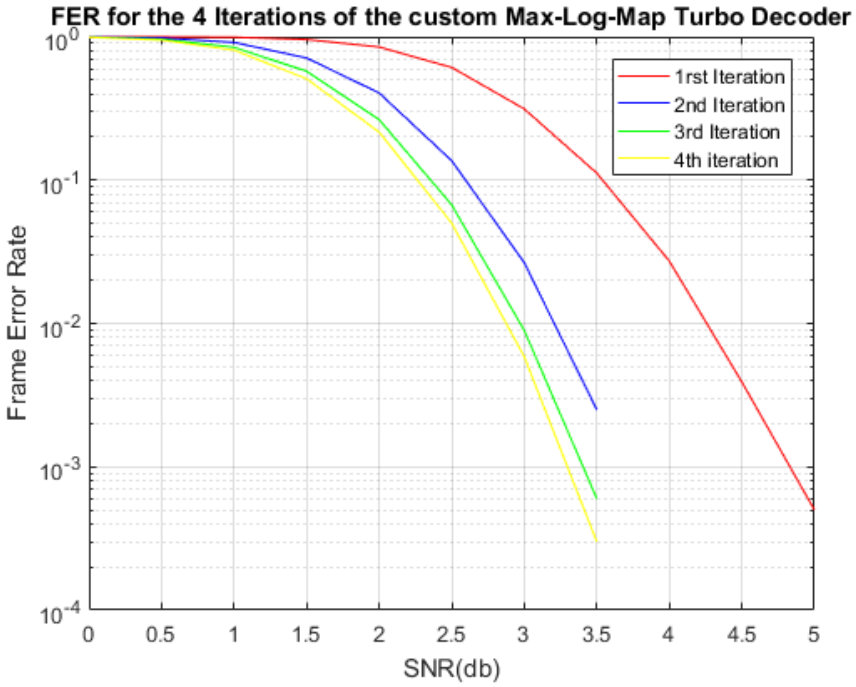
Η υλοποίηση αυτή ενσωματώθηκε στη CML για εκτενή προσομοίωση. Στο σημείο αυτό αξίζει να αναφερθεί ότι η CML διαθέτει υλοποιημένο Max-Log MAP decoder. Παρόλα αυτά, επιλέχθηκε η αυτόνομη υλοποίηση του, λόγω της απλοποιημένης λειτουργίας της, και άρα της χαμηλότερης πολυπλοκότητας της. Έτσι χρησιμοποιήθηκε ο κώδικας Matlab της αυτόνομης υλοποίησης για να προσομοιωθεί η τελευταία στη CML, και να συγκριθεί η απόδοση του με αυτή του BCJR.

Έτσι ο Max-Log MAP προσομοιώθηκε με τις ίδιες παραμέτρους, δηλαδή μέγεθος frame 160bits, QPSK modulation, UMTS interleaver και μη χρήση puncturing. Τα αποτελέσματα της προσομοίωσης αποστολής 10^4frames με μέχρι 4 επαναλήψεις αποστολής (ARQ) και 4 επαναλήψεις αποκωδικοποίησης (Iterations) παρουσιάζονται παρακάτω:



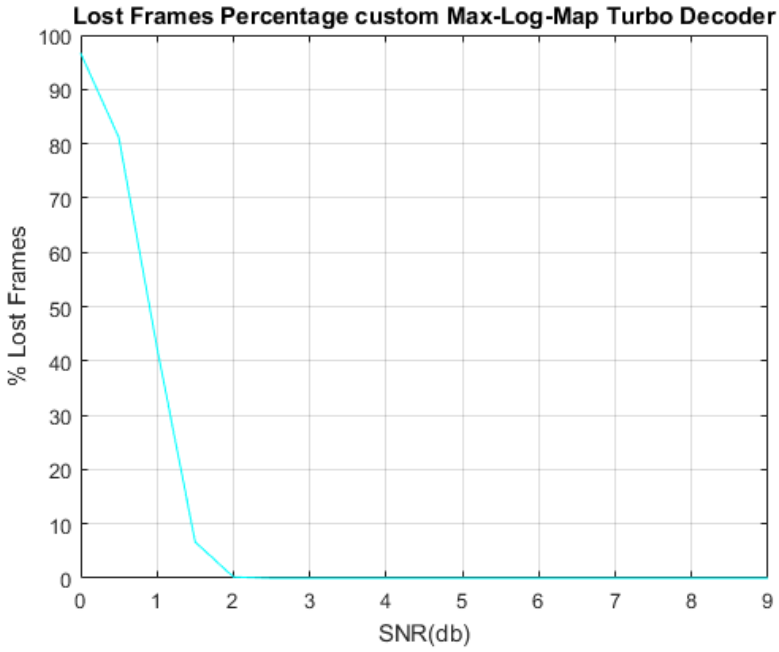
Εικόνα 4.7: Σύγκριση BER μεταξύ BCJR και Max-Log MAP

Παρατηρείται ότι για πολύ υψηλές τιμές θορύβου (0-2,5 SNR_{dB}) ο Max-Log MAP είναι λιγότερο αποδοτικός από τον BCJR. Αυτό ενδεχομένως οφείλεται στο ότι χρησιμοποιεί γενικά απλοποιημένης μορφής αλγορίθμους. Παρόλα αυτά, στο υπόλοιπο φάσμα των τιμών θορύβου είναι εμφανώς πιο αποδοτικός. Αυτό αντανακλάται στην απόδοση του αναφορικά με το FER, όπου για περιοχή τιμών θορύβου <5 SNR_{dB} η απώλεια frames διακόπτεται οριστικά:



Εικόνα 4.8: Απόδοση FER του Max-Log MAP

Σε συνδυασμό με την τεχνική ARQ, η απόδοση του Max-Log MAP βελτιώνεται σε τέτοιο βαθμό, έτσι ώστε να μην απαιτούνται επαναλήψεις αποστολής frames από τα 2 SNR_{dB} και μετά:



Εικόνα 4.9: Ποσοστό χαμένων frames για τον Max-Log MAP

Με βάση τα παραπάνω αποτελέσματα, αποφασίστηκε η συνέχιση της υλοπίησης TurboDecoder, με επόμενο στάδιο την ανάπτυξή του σε επίπεδο software, χρησιμοποιόντας τον αλγόριθμο Max-Log MAP. Ο κύριος παράγοντας που οδήγησε στην απόφαση αυτή, ήταν το tradeoff μεταξύ της απόδοσης του decoder σε συνθήκες υψηλών τιμών θορύβου, και της σχετικά χαμηλής πολυπλοκότητάς του. Η επιλογή μίας απλοποιημένης υλοποίησης του Max-Log MAP θεωρήθηκε ιδανική, με βάση τα συστήματα που χρησιμοποιούνται σε εφαρμογές ΑΔΑ, και τις δυνατότητές τους.

5 Σχεδίαση και Εφαρμογή ενός Turbo Code Decoder

Στο κεφάλαιο αυτό παρουσιάζεται η υλοποίηση του σχήματος που προσομοιώθηκε στο Κεφάλαιο 4, σε επίπεδο λογισμικού. Έτσι, οι διαδικασίες παραγωγής των δεδομένων, κωδικοποίησης, διαμόρφωσης και αποστολής τους υλοποιούνται σε ένα πρόγραμμα C. Στη συνέχεια, παρουσιάζεται η εφαρμογή του λογισμικού σε μία πλακέτα – σταθμό βάσης. Με τον τρόπο αυτό προσομοιώνεται η καταγραφή μετρήσεων, η κωδικοποίηση των δεδομένων που προκύπτουν και η αποστολή τους από έναν κόμβο-πηγή ενός ΑΔΑ, και εφαρμόζεται αποκωδικοποίηση των δεδομένων αυτών, σε έναν σταθμό βάση. Ύστερα παρουσιάζεται η υλοποίηση του Decoder σε γλώσσα περιγραφής υλικού, με σκοπό την παράλληλη χρήση FPGA στο σταθμό βάσης, για τη λειτουργία του Decoding. Μετά την επιτυχή ανάπτυξη του Decoder στην FPGA παρουσιάζονται μετρήσεις που αφορούν τη χρονική και ενεργειακή απόδοση του Decoder σε επίπεδο λογισμικού, αλλά και με τη χρήση της FPGA.

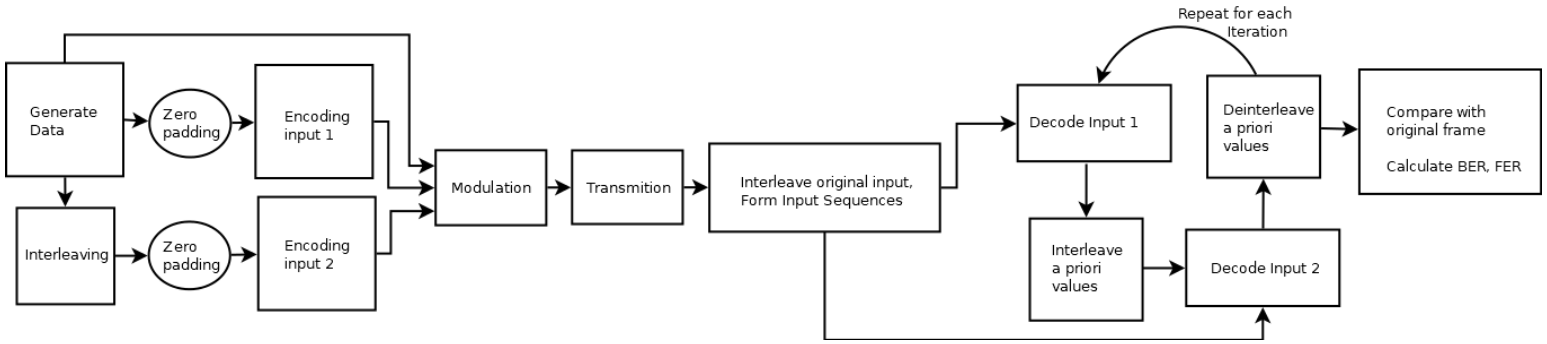
5.1 Σταθμός Βάσης

Για την εφαρμογή Turbo Decoding στον σταθμό βάσης ενός ΑΔΑ, επιλέχθηκε μία πλακέτα BeagleBoard, συγκεκριμένα το BeagleBone. Η συγκεκριμένη low-cost πλακέτα διαθέτει την κατάλληλη επεξεργαστική ισχύ για να υποστηρίξει τη διαδικασία του Decoding, ενώ έχει τη δυνατότητα παράλληλης σύνδεσης με FPGA για την εφαρμογή Hardware Acceleration. Επιπλέον, χαρακτηριστικά όπως η χρήση λειτουργικού Linux, η δυνατότητα σύνδεσης της στο Internet, καθώς και η πληθώρα διαθέσιμων I/O, συνέβαλαν στην επιλογή της ως σταθμό βάση, για την ανάπτυξη λογισμικού προσομοίωσης του Channel Coding σε πραγματικές συνθήκες. Τέλος, η διαθεσιμότητα FPGA cape για την πλακέτα αυτή απλοποιεί τη διαδικασία ανάπτυξης λογισμικού σε HDL με σκοπό τη hardware accelerated λειτουργία του Turbo Decoder. Η διασύνδεση BeagleBone και FPGA board υποστηρίζεται την πλατφόρμα Wishbone, υλοποιημένη σε VHDL, που αναλαμβάνει την επικοινωνία αυτών των δύο. Στη συνέχεια παρατίθενται τα χαρακτηριστικά της πλακέτας BeagleBone:

Επεξεργαστής:	AM335x 720MHz ARM Cortex-A8
Μνήμη RAM:	256MB DDR2
Επιπλέον:	3D graphics accelerator
	ARM Cortex-M3 (power management)
	2x PRU 32-bit RISC CPUs
Συνδεσιμότητα:	USB client: power, debug and device
	USB host
	Ethernet
	2x 46 pin headers
Συμβατότητα Λογισμικού:	4GB microSD card w/ Angstrom Distribution
	Cloud9 IDE on Node.JS w/ BoneScript library

5.2 Λογισμικό

Η ανάπτυξη λογισμικού για την εφαρμογή του συστήματος κωδικοποίησης – αποστολής – αποκωδικοποίησης δεδομένων σε ένα ΑΔΑ έγινε σε περιβάλλον Linux, με λειτουργικό σύστημα Ubuntu, στη γλώσσα C. Χρησιμοποιήθηκαν ανοικτά εργαλεία ανάπτυξης (GNU), συγκεκριμένα τα GNU Compiler Collection (GCC) και GNU Debugger (GDB). Έχοντας κατά νου ότι η πλακέτα που χρησιμοποιείται ως σταθμός βάσης λειτουργεί επίσης με λειτουργικό Linux, η ανάπτυξη του κώδικα έγινε σε Η/Υ με εργαλεία αντίστοιχα με αυτά που υπάρχουν διαθέσιμα και στην πλακέτα BeagleBone. Με τον τρόπο αυτό η διαδικασία της σύνθεσης και αποσφαλμάτωσης του κώδικα γίνεται με τον ίδιο τρόπο τόσο στον Η/Υ όπου αναπτύχθηκε, όσο και στην πλακέτα στην οποία εφαρμόζεται. Στο παρακάτω διάγραμμα παρουσιάζεται η βασική λειτουργία του προγράμματος:



Εικόνα 5.16: Η δομή του λογισμικού υλοποίησης του συστήματος

Στη συνέχεια περιγράφεται η λειτουργία του λογισμικού για τον κάθε τομέα του σχήματος Channel Coding.

5.2.1 Παραμετροποίηση

Αρχικά δημιουργούνται οι απαραίτητες μεταβλητές για τη λειτουργία του προγράμματος, δεσμεύονται περιοχές της μνήμης για τις αντίστοιχες μεταβλητές, ενώ ένα μέρος των μεταβλητών αρχικοποιείται. Στη συνέχεια εισάγονται από το χρήστη οι παράμετροι υπό τις οποίες θα συνεχιστεί η λειτουργία του προγράμματος. Συγκεκριμένα, εισάγεται το μέγεθος frame σε bits, η παρουσία ή όχι θορύβου στο κανάλι επικοινωνίας, το επίπεδο AWGN θορύβου σε SNR_{dB}, ο αριθμός αυθεντικών frames που θα αποσταλθούν, και τέλος ο αριθμός επαναλήψεων της αποκωδικοποίησης από τον Turbo Decoder. Σύμφωνα με τις παραμέτρους που εισήχθησαν, αρχικοποιούνται οι υπόλοιπες μεταβλητές.

5.2.2 Κωδικοποίηση

Κάνοντας χρήση της συνάρτησης *rand*, δημιουργούνται frames τυχαίων δεδομένων με δυαδική τιμή τα οποία στη συνέχεια κωδικοποιούνται. Η συνάρτηση *encoder* υλοποιεί τον κάθε επιμέρους Convolutional Encoder, όπως αυτός παρουσιάστηκε στο Κεφάλαιο 4. Η συνάρτηση *interleaver* υλοποιεί τον relative prime interleaver του συστήματος. Όπως συμβαίνει και στην υλοποίηση του Encoder σε περιβάλλον Matlab, στο τέλος κάθε αλληλουχίας bit που αποτελεί είσοδο για τον κάθε encoder προστίθενται τρία bits με τιμή '0'. Έτσι συνάρτηση *encoder* καλείται δύο φορές με είσοδο τις δύο αλληλουχίες bits διαδοχικά, αυτή με τις αρχικές τυχαίες τιμές του frame και αυτή με τις αναδιατεταγμένες τιμές (μετά από χρήση του interleaver). Ο *encoder* υπολογίζει τις τιμές των εξόδων των adders (Σχ. -) εκτελώντας τις εξής πράξεις, όπου X_k η είσοδος για το χ-οστό bit, A_x οι έξοδοι των τριών adders, D_x οι έξοδοι των τριών D flip-flops και Z_k η τελική έξοδος του encoder:

$$A_3 = D_2 \text{ xor } D_3;$$

$$A_1 = A_3 \text{ xor } X_k;$$

$$A_2 = A_1 \text{ xor } D_1;$$

$$Z_k = A_2 \text{ xor } D_3;$$

Εφόσον παραχθούν οι δύο συμβολοσειρές εξόδου του κάθε encoder, οι τρείς πλέον συμβολοσειρές (αρχικά δεδομένα, κωδικοποιημένη συμβολοσειρά και interleaved κωδικοποιημένη συμβολοσειρά) είναι έτοιμες προς μετάδοση.

5.2.3 Διαμόρφωση και Μετάδοση

Για τη διαμόρφωση των συμβόλων χρησιμοποιείται το πρότυπο BPSK (antipodal) όπου κάθε bit με τιμή '0' αντιστοιχείται στο σύμβολο '-1' και κάθε bit με τιμή '1' αντιστοιχείται στο σύμβολο '1'. Για τη μετάδοση των δεδομένων προστίθεται AWGN θόρυβος εφόσον έχει επιλεχθεί από τον χρήστη, χρησιμοποιώντας συνάρτηση *randn* για την παραγωγή τυχαίων αριθμών κανονικής κατανομής. Με αυτόν τον τρόπο καταλήγουμε στις τρεις συμβολοσειρές που έχουν μεταδοθεί μέσω του καναλιού του συστήματος και αποτελούν είσοδο για τον Turbo Decoder.

5.2.4 Αποκωδικοποίηση

Για τη λειτουργία του Turbo Decoder, η συνάρτηση *decoder*υλοποιεί τον κάθε επιμέρους Max-Log MAP decoder, όπως αυτός παρουσιάστηκε στο Κεφάλαιο 4. Για την απλοποιημένη συνάρτηση $max*(a,b) \approx \max(a,b)$ αναπτύχθηκε η συνάρτηση *max1* για τη σύγκριση δύο μεταβλητών, και η συνάρτηση *max2* για τη σύγκριση περισσότερων τιμών, που εισάγονται με τη μορφή πίνακα στη συνάρτηση. Κατά τα άλλα ο κώδικας C που υλοποιεί τους επιμέρους Decoders λειτουργεί με τον ίδιο ακριβώς τρόπο όπως και ο κώδικας Matlab.

Από τις τρείς συμβολοσειρές που λαμβάνει ο δέκτης, δημιουργούνται δύο συμβολοσειρές για κάθε έναν από τους decoders. Η πρώτη συμβολοσειρά (*in1*) έχει διαδοχικά τα αρχικά bits και τα κωδικοποιημένα bits του ενός encoder, και η δεύτερη (*in2*) έχει διαδοχικά τα αναδιατεταγμένα αρχικά bits (με χρήση interleaver) και τα αναδιατεταγμένα κωδικοποιημένα bits όπως δημιουργήθηκαν από τον δεύτερο encoder. Για κάθε επανάληψη της κωδικοποίησης καλείται αρχικά η συνάρτηση *decoder*με είσοδο την πρώτη συμβολοσειρά εισόδου *in1*, και a priori τιμές από προηγούμενες αποκωδικοποιήσεις του frame, αν υπάρχουν.

Στη συνάρτηση *decoder*, η είσοδος χωρίζεται στα δύο, δημιουργώντας δύο αλληλουχίες, την x_k για τα αρχικά δεδομένα, και την y_k για τα κωδικοποιημένα bits. Χρησιμοποιώντας τις ευκλείδειες αποστάσεις μεταξύ των δύο συμβόλων κάθε φορά (x_k, y_k) αλλά και τις a priori τιμές από προηγούμενες αποκωδικοποιήσεις (αν υπάρχουν), με βάση το αντίστοιχο διάγραμμα Trellis, υπολογίζονται τα Branch Metrics (d). Στη συνέχεια υπολογίζονται οι τιμές των Forward State Metrics (α_k), και Backward State Metrics (β_k), με χρήση της συνάρτησης *max1*. Τέλος από τις τιμές α, β, d και με χρήση των συνάρτησης *max2* υπολογίζονται οι τιμές llr. Από τις τιμές αυτές προκύπτουν δύο αλληλουχίες που ενώνονται για να σχηματίσουν την αλληλουχία εξόδου. Η πρώτη αλληλουχία περιέχει hard decision τιμές (‘0’ αν η τιμή llr για κάθε bit είναι μικρότερη του μηδενός και ‘1’ αν είναι μεγαλύτερη), ενώ η δεύτερη αλληλουχία σχηματίζει a priori τιμές που θα χρειαστούν σε επόμενες αποκωδικοποιήσεις, χρησιμοποιώντας τις τιμές llr, τις a priori τιμές από προηγούμενες αποκωδικοποιήσεις (αν υπάρχουν) και την αρχική συμβολοσειρά.

Μετά την πρώτη διαδικασία αποκωδικοποίησης, συλλέγονται οι a priori τιμές που προέκυψαν, οι οποίες αναδιατάσσονται με χρήση του interleaver. Οι τιμές αυτές μαζί με τη δεύτερη αλληλουχία εισόδου *in2* χρησιμοποιούνται ως είσοδος για τη δεύτερη κλήση της συνάρτησης decoder. Η αποκωδικοποίηση της δεύτερης αλληλουχίας συμβαίνει με αντίστοιχο τρόπο με πριν, ενώ εξάγονται οι αντίστοιχες a priori τιμές για την επανάληψη της αποκωδικοποίησης, και τα αποκωδικοποιημένα bits μετά από hard decision, για την περίπτωση που η αποκωδικοποίηση του frame τελείωσε.

Στο τέλος αποκωδικοποίησης του κάθε frame, η αλληλουχία bits που προκύπτει μετά από hard decision συγκρίνεται με την αυθεντική αλληλουχία που παράχθηκε αρχικά. Έτσι αφενός καταγράφεται ο αριθμός λαθών για τη μέτρηση των ρυθμών BER και FER, ενώ αφετέρου κρίνεται αν το frame θα πρέπει να μεταδοθεί ξανά, σύμφωνα με την τεχνική ARQ, σε περίπτωση που περιέχει λάθη.

5.2.5 Συλλογή Αποτελεσμάτων

Εφόσον η παραγωγή, κωδικοποίηση, αποστολή και αποκωδικοποίηση όλων των frames έλαβε τέλος, συλλέγονται και παρουσιάζονται στο χρήστη τα αποτελέσματα που προέκυψαν. Έτσι τυπώνεται στην οθόνη το σύνολο των παραμέτρων υπό τις οποίες έτρεξε το πρόγραμμα, ενώ υπολογίζεται ο ρυθμός λαθών για τα bits και τα frames που μεταδόθηκαν συνολικά και ο αριθμός των frames που χρειάστηκε να μεταδοθούν ξανά με το ARQ. Τέλος, παρουσιάζονται χρονικές

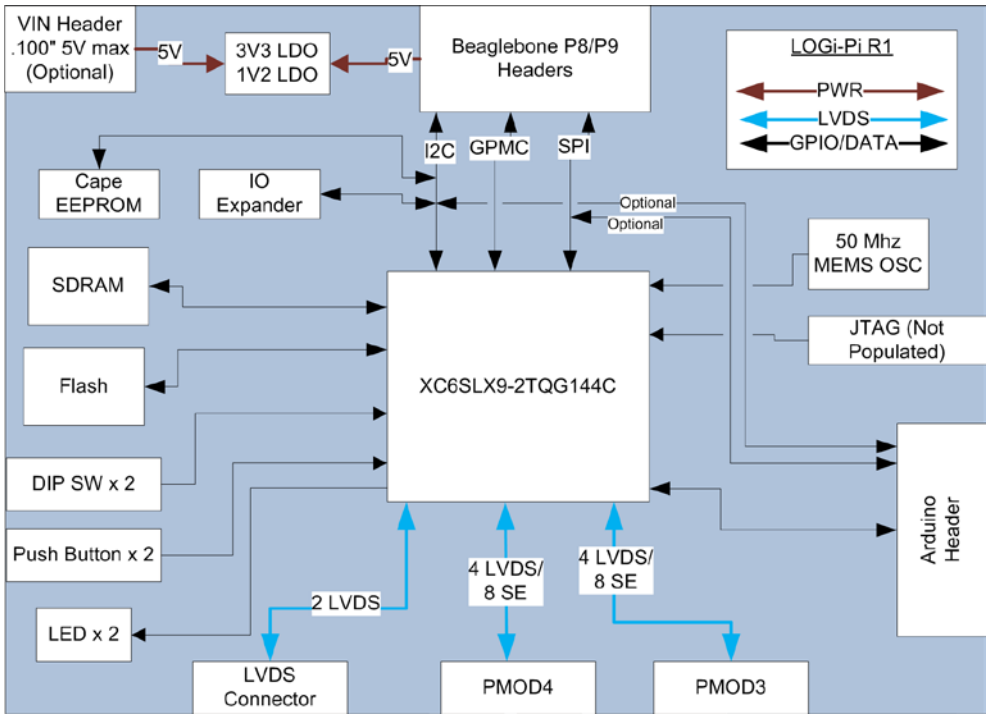
μετρήσεις που σημειώθηκαν με χρήση μεταβλητών τύπου *clock_t* για την καταγραφή timestamps. Οι μετρήσεις αυτές αφορούν τη συνολική διάρκεια λειτουργίας του προγράμματος, καθώς και της μέσης χρονικής διάρκειας αποκωδικοποίησης ενός frame.

5.3 Χρήση Αναδιατασσόμενης Λογικής

Για την υλοποίηση του αλγορίθμου αποκωδικοποίησης σε αναδιατασσόμενη λογική, χρησιμοποιήθηκε το LOGI Bone, δηλαδή το FPGA board cape που αντιστοιχεί στην πλακέτα BeagleBone. Το συγκεκριμένο cape διαθέτει τα εξής χαρακτηριστικά:

- FPGA: Spartan 6 LX9 XC6SLX9-2TQG144C (TQFP144 Package)
- Length tuned GPMC, SDRAM, LVDS signals
- 2x Push buttons
- 2x DIP Switch
- 1x High bandwidth SATA connector expansion port for maximum LVDS bandwidth (SATA protocol not supported)
- 32 FPGA GPIO through PMOD and Arduino headers
- 2x Digilent Inc. PMOD ports supporting 59+ plug and play hardware modules
- 1x Arduino Header supporting 200+ Arduino Shield modules
- Optional GPMC, SPI or I2C port access from the BeagleBone
- 10x length tuned LVDS pairs
- 32 MB SDRAM

Το Block Diagram του LOGI Bone είναι το εξής:



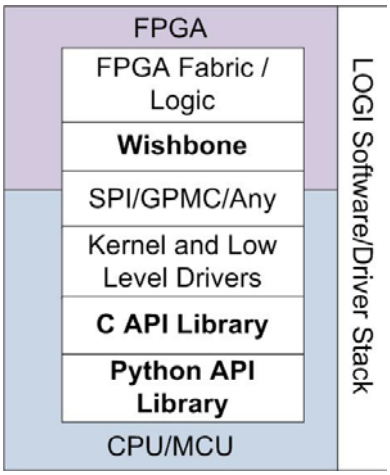
Εικόνα 5.2: LOGI Bone Top Level Block Diagram

Η FPGA Spartan 6 LX9 του board έχει τα εξής χαρακτηριστικά:

- Logic Cells: 9,152
- Configurable Logic Blocks (CLBs)
    - Slices (4 LUTs & 8 flip-flops): 1,430
    - Flip-Flops: 11,440
    - Distributed RAM (Kb): 90
- DSP48A1 Slices (18 x 18 multiplier, 1 adder, 1 accumulator): 16
- Block RAM Blocks
    - 18 Kb: 32
    - Max (Kb): 576
- CMTs (2 DCMs & 1 PLL): 2
- Memory Controller Blocks (Max): 2
- Endpoint Blocks for PCI Express: 0
- Maximum GTP Transceivers: 0
- Total I/O Banks: 4
- Max User I/O: 200

5.3.1 Πλατφόρμα Wishbone

Για την επικοινωνία του board με την FPGA παρέχεται υλοποιημένο ένα σύνολο λογισμικού και οδηγών που συνθέτει το κατάλληλο περιβάλλον για το χρήστη, με σκοπό τον προγραμματισμό της FPGA, χωρίς ο τελευταίος να χρειάζεται να επέμβει σε ό,τι έχει να κάνει με την εγγραφή και την προσπέλαση δεδομένων από και προς την FPGA. Στο παρακάτω διάγραμμα παρουσιάζεται η στοίβα λογισμικού και οδηγών που υλοποιείται από τη CPU του BeagleBone προς την FPGA, από κάτω προς τα πάνω:

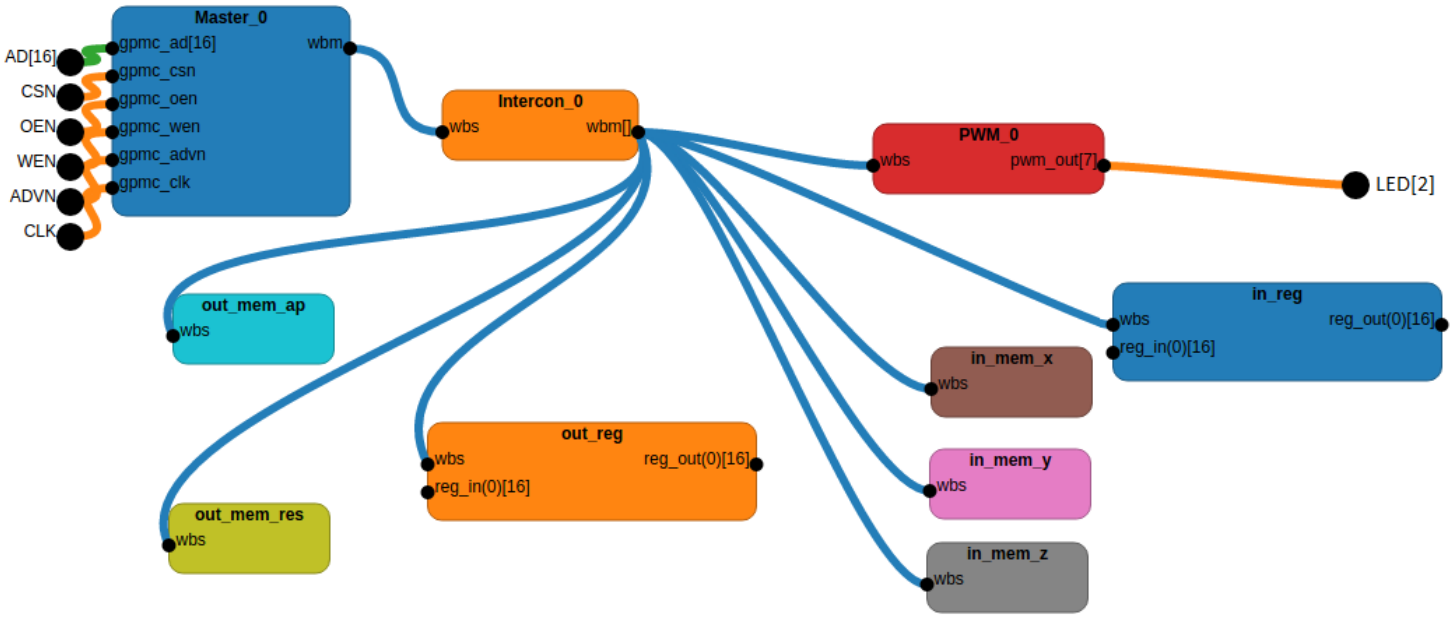


Εικόνα 5.3: LOGI Software/Driver Stack

Ο τομέας της παραπάνω στοίβας που καθορίζει τους πόρους της FPGA που θα χρησιμοποιηθούν από το χρήστη και την επικοινωνία του BeagleBone με αυτούς είναι

το Wishbone. Η πλατφόρμα παρέχει υλοποιημένα σε VHDL όλα τα components που μπορεί ο χρήστης να χρησιμοποιήσει, ενώ συνοδεύεται από την online εφαρμογή Skeleton Editor. Η εφαρμογή αυτή παρέχει στο χρήστη να σχεδιάσει σε γραφικό περιβάλλον το Top Level Block Diagram της εφαρμογής που αναπτύσσει, περιγράφοντας έτσι τους πόρους του FPGA cape που θα χρησιμοποιήσει, και τον τρόπο με τον οποίο αυτοί συνδέονται μεταξύ τους. Έτσι παράγεται αυτόματα από το Block Diagram ο αντίστοιχος κώδικας VHDL που υλοποιεί τα components της FPGA που επέλεξε ο χρήστης.

Για την υλοποίηση του Max-Log MAP Decoder σχεδιάστηκε στον Skeleton Editor το εξής Block Diagram:



Εικόνα5.47: Wishbone Top Level Block Diagram

- Master

Το component *Master_0* υλοποιεί την επικοινωνία του board με την FPGA. Η είσοδος/έξοδος *gpmc_ad* μεγέθους 16 bit χρησιμεύει στην ανάγνωση και εγγραφή δεδομένων, ενώ χρησιμοποιείται τόσο ως είσοδος και έξοδος δεδομένων 16bit, όσο και για τον προσδιορισμό της διεύθυνσης στην οποία διαβάζονται ή εγγράφονται τα δεδομένα. Τα υπόλοιπα σήματα (*gpmc_csn*, *gpmc_oen*, *gpmc_wen*, *gpmc_advn*) αποτελούν σήματα ελέγχου εγγραφής/ ανάγνωσης δεδομένων, ενώ το σήμα *gpmc_clk* αφορά το ρολόι της FPGA που οδηγείται από τη CPU του BeagleBone.

- Intercon

Το component Intercon_0 αναλαμβάνει τη διαχείριση των διάφορων components στο εσωτερικό της FPGA. Συνδέεται με το Master_0 και τα υπόλοιπα component με διαύλους μεγέθους 16bit.

- In_mem

Οι μνήμες in_mem_x, in_mem_y, in_mem_z χρησιμοποιούνται για την εγγραφή των δεδομένων εισόδου. Οι δύο συμβολοσειρές εισόδου με την αρχική και την κωδικοποιημένη συμβολοσειρά εγγράφονται στις μνήμες in_mem_x, in_mem_y αντίστοιχα. Στην μνήμη in_mem_z εγγράφονται οι a priori τιμές που προέκυψαν από προηγούμενες αποκωδικοποιήσεις.

- Out_mem

Στις μνήμες out_mem_res και out_mem_ap εγγράφονται οι συμβολοσειρές εξόδου του Decoder, για να διαβαστούν από το BeagleBone. Στη μνήμη out_mem_res εγγράφονται οι τιμές llr που χρησιμεύουν στην αποκωδικοποίηση των bits μέσω hard decisions. Στη μνήμη out_mem_ap εγγράφονται οι a priori τιμές που χρησιμοποιούνται στις επόμενες επαναλήψεις της αποκωδικοποίησης ενός frame.

- In_reg, Out_reg

Οι δύο καταχωρητές in_reg, out_reg μεγέθους 16 bit χρησιμεύουν στον έλεγχο ανάγνωσης και εγγραφής από και προς την FPGA. Ανάλογα με την τιμή που δίνεται από το λογισμικό του BeagleBone στον in_reg, η FSM της FPGA αντιλαμβάνεται πότε τα δεδομένα εισόδου έχουν εγγραφεί στις αντίστοιχες μνήμες και μπορεί να ξεκινήσει η διαδικασία αποκωδικοποίησης. Αντίστοιχα, όταν η τελευταία λάβει τέλος, η FSM εγγράφει στον out_reg μία συγκεκριμένη τιμή, έτσι ώστε το λογισμικό διαβάζοντάς την να αντιληφθεί ότι τα αποτελέσματα της αποκωδικοποίησης έχουν εγγραφεί στις μνήμες εξόδου και μπορούν να προσπελαστούν.

- PWM

Το component *PWM_0* οδηγεί ένα από τα LED της FPGA έτσι ώστε να είναι εμφανές στον χρήστη πότε αυτή βρίσκεται σε λειτουργία.

5.3.2 Λογισμικό Ελέγχου της FPGA από το BeagleBone

Για την υλοποίηση του Decoder στο FPGA cape αναπτύχθηκε λογισμικό σε γλώσσα C, το οποίο λειτουργεί στο BeagleBone, με βάση το λογισμικό που αναλύθηκε στην Ενότητα 5.2. Το κομμάτι κωδικοποίησης, διαμόρφωσης και μετάδοσης των δεδομένων παρέμεινε ίδιο, ωστόσο έγιναν τροποποιήσεις στο κομμάτι αποκωδικοποίησης, όπου η FPGA πλέον αναλαμβάνει το μεγαλύτερο μέρος της λειτουργικότητας της συνάρτησης *decoder*.

Αρχικά, δημιουργήθηκαν οι συναρτήσεις *lbread, lbwrite* οι οποίες κάνουν χρήση των συναρτήσεων *pread, pwrite* αντίστοιχα, της βιβλιοθήκης *unistd.h* για να αναγνώσουν και να εγγράψουν δεδομένα στις μνήμες της FPGA. Ακόμα, αναπτύχθηκε η συνάρτηση *nsleep*, η οποία δέχεται ως είσοδο μία χρονική διάρκεια σε nsec, και αναλαμβάνει να παγώσει τη ροή του προγράμματος για τη διάρκεια αυτή. Η λειτουργία

αυτής της συνάρτησης είναι ιδιαίτερα σημαντική για τον συγχρονισμό του board με το cape, αφού χωρίς τη χρήση της, σε πολλές περιπτώσεις το λογισμικό ενδέχεται να εγγράφει δεδομένα με μεγαλύτερη ταχύτητα απ' ότι μπορεί η FPGA να διαχειριστεί, με αποτέλεσμα προβληματική ή λανθάνουσα επικοινωνία μεταξύ των δύο board. Τέλος, η συνάρτηση *logi_init* αναλαμβάνει να ενεργοποιήσει την FPGA, και να αρχικοποιήσει τους καταχωρητές και τις μνήμες της. Εγγράφοντας την τιμή '100' στον καταχωρητή in_reg, η FSM της FPGA ξεκινάει τη διαδικασία αρχικοποίησης των μνημών, εγγράφοντας την τιμή '0' σε όλες τις θέσεις μνήμης.

Σε ό,τι αφορά τη λειτουργία της αποκωδικοποίησης, στη συνάρτηση *decoder* έγιναν οι εξής τροποποιήσεις. Αφού διαμορφωθούν οι δύο συμβολοσειρές εισόδου, εισάγονται μαζί με τις a priori τιμές στο σύστημα, χρησιμοποιώντας την *lb_write*για να εγγραφούν τα δεδομένα στις αντίστοιχες μνήμες. Στη συνέχεια, εγγράφεται στον καταχωρητή *in_reg* η τιμή '300', που αποτελεί σήμα έναρξης της αποκωδικοποίησης. Το λογισμικό στη συνέχεια διαβάζει ανά τακτά χρονικά διαστήματα την έξοδο του καταχωρητή out_reg. Όταν η τιμή που διαβάζεται είναι '100', σημαίνει ότι η αποκωδικοποίηση έχει τελειώσει, και τα αποτελέσματα της μπορούν να διαβαστούν από τη μνήμη εξόδου out_reg. Αυτό συμβαίνει κάνοντας χρήση της *lb_read*, ενώ όταν η διαδικασία ανάγνωσης τελειώσει, εγγράφεται στον καταχωρητή εισόδου in_reg η τιμή '400', έτσι ώστε η FSM της FPGA να μπει σε κατάσταση αδράνειας, μέχρι να εγγραφεί στις μνήμες εισόδου το επόμενο frame.

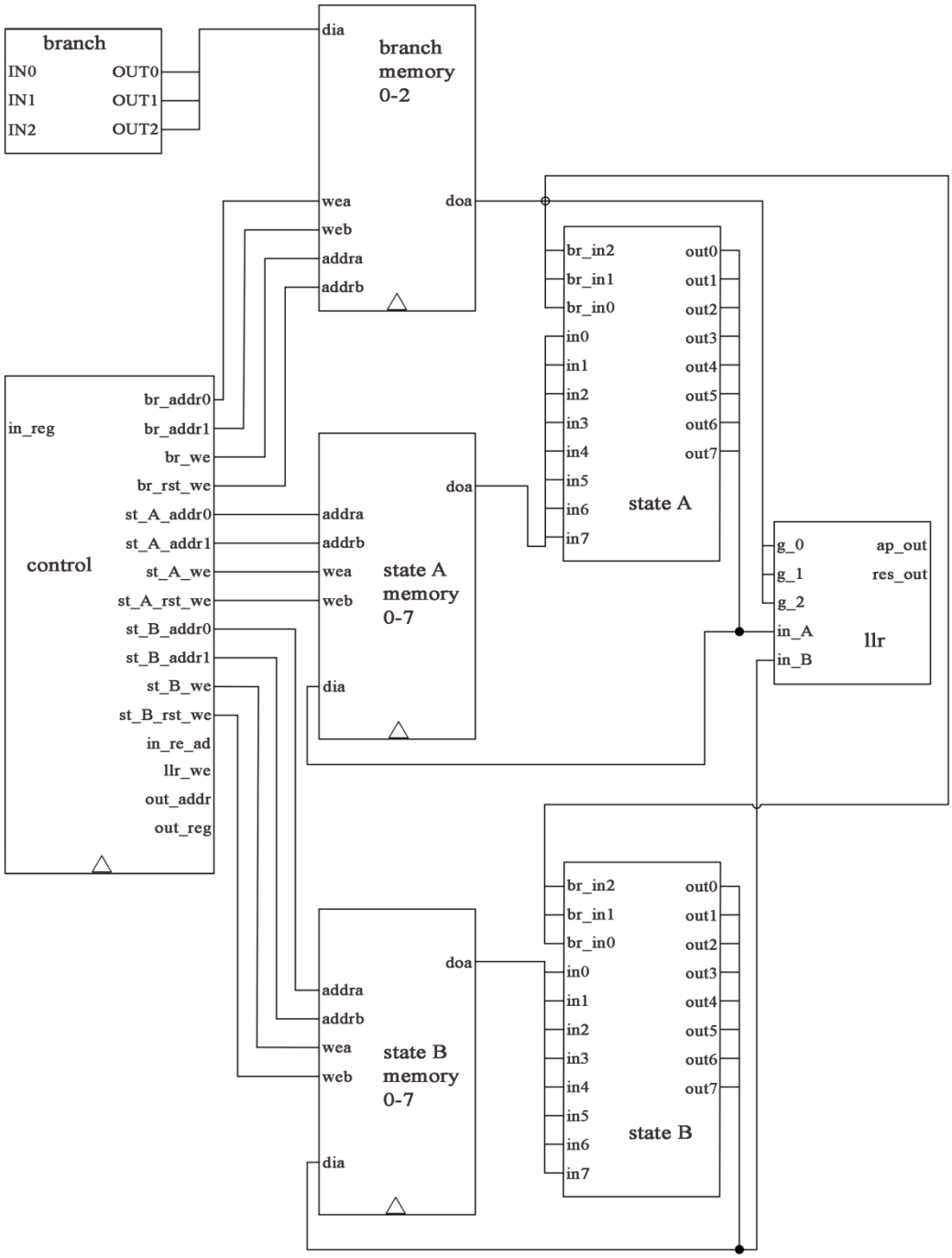
5.3.3 Υλοποίηση του Max-Log MAP σε Υλικό (FPGA)

Για την ανάπτυξη του Max-Log MAP Decoder στην FPGA cape επιλέχθηκε η γλώσσα περιγραφής υλικού VHDL. Η αρχική ιδέα ανάπτυξης του Decoder αφορούσε τη χρήση του Vivado High-Level Synthesis της Xilinx για τη μετατροπή ήδη υλοποιημένου κώδικα C σε HDL. Η διαδικασία σύνθεσης του κώδικα C που χρησιμοποιεί εκτεταμένη διαχείριση μνήμης κατέστησε τη διαδικασία μετατροπής του κώδικα πιο σύνθετη από την εκ νέου ανάπτυξη κώδικα HDL. Η χρήση pointers καθώς και συναρτήσεως δέσμευσης μνήμης της C είναι λειτουργίες που χρειάζονται εκτεταμένη παραμετροποίηση στο Vivado για την επιτυχή τους υλοποίηση σε γλώσσα περιγραφής υλικού. Επιπλέον, η υλοποιημένη πλατφόρμα Wishbone διατίθεται σε γλώσσα VHDL. Έτσι, η υλοποίηση του Decoder σε VHDL απλοποιεί τη διαδικασία ενσωμάτωσης του στην πλατφόρμα Wishbone. Η ανάπτυξη του κώδικα VHDL έγινε στο εργαλείο Xilinx IDE 14.7.

Το πρώτο στάδιο υλοποίησης του Decoder αφορά την ανάπτυξη των μονάδων υπολογισμού των Branch, Forward και Backward Metrics. Οι μονάδες αυτές υλοποιούνται στα components branch, state A και state B αντίστοιχα. Για την αποθήκευση των δεδομένων που προκύπτουν από τις μονάδες αυτές δημιουργήθηκαν μνήμες τύπου RAM, 256 θέσεων, και πλάτους 16 bit. Οι συγκεκριμένες μνήμες είναι διαθέσιμες μεταξύ των διάφορων άλλων υλοποιημένων components στο Wishbone. Διαθέτουν δύο ρολόγια, όπου στη συγκεκριμένη υλοποίηση είναι συγχρονισμένα με το κεντρικό ρολόι του συστήματος, δύο σήματα write enable για την εγγραφή δεδομένων

από τις δύο εισόδους δεδομένων και δύο σήματα enable που ενεργοποιούν δύο εξόδους δεδομένων. Τέλος, υλοποιήθηκε η αντίστοιχη μονάδα για τον υπολογισμό των τιμών LLR και των a priori τιμών.

Στη συνέχεια δημιουργήθηκε η μονάδα Control, που υλοποιεί την FSM η οποία διαχειρίζεται το σύστημα. Εκτός από τη διαχείριση των μονάδων υπολογισμού και των αντίστοιχων μονάδων μνήμης, η FSM του συστήματος αναλαμβάνει την επικοινωνία του υποσυστήματος υλοποίησης του Decoder με την πλατφόρμα Wishbone. Παρακάτω παρουσιάζεται το block diagram του υποσυστήματος του Decoder:



Εικόνα 5.5: Block Diagram του Max-Log MAP Decoder

Το παραπάνω διάγραμμα περιλαμβάνει μόνο τις μονάδες που υλοποιήθηκαν για τη λειτουργία του Decoder. Ο τρόπος με τον οποίο συνδέονται με το Wishbone έχει ως εξής:

- Οι είσοδοι *IN0, IN1, IN2* της μονάδας *Branch* συνδέονται με τις εξόδους των μνημών *in_mem_x, in_mem_y, in_mem_z* του Wishbone, για την εισαγωγή δεδομένων στο σύστημα.
- Η είσοδος *in_reg* της μονάδας *Control* συνδέεται με την έξοδο του καταχωρητή *in_reg* του Wishbone, για τη σηματοδότηση έναρξης και αδρανοποίησης του Decoder.
- Η έξοδος *in_re_ad* του *Control* υποδεικνύει τη διεύθυνση ανάγνωσης των δεδομένων από τις μνήμες *in_mem_x, in_mem_y, in_mem_z* του Wishbone.
- Το σήμα εξόδου *llr_we* του Control ενεργοποιεί την εγγραφή δεδομένων στις μνήμες *out_mem_res* και *out_mem_ap* του Wishbone, για την εξαγωγή των αποτελεσμάτων αποκωδικοποίησης. Η έξοδος *out_addr* υποδεικνύει την διεύθυνση εγγραφής στις μνήμες *out_mem_res* και *out_mem_ap* του Wishbone.
- Οι έξοδοι *ap_out* και *res_out* της μονάδας *LLR*, συνδέονται με τις εισόδους δεδομένων των μνημών *out_mem_ap* και *out_mem_res* του Wishbone, αντίστοιχα.
- Η έξοδος *out_reg* του *Control* συνδέεται με την είσοδο του καταχωρητή *out_reg* του Wishbone, για τη σηματοδότηση τερματισμού της διαδικασίας αποκωδικοποίησης.

Σχετικά με τις μνήμες του συστήματος και τον τρόπο που αναπαριστώνται στο Block Diagram:

- Το κουτί *BranchMemory 0-2* αναπαριστά τις τρείς διαφορετικές μνήμες *branch_memory0, branch_memory1, branch_memory2* που αποθηκεύουν τις εξόδους *OUT0, OUT1, OUT2* της μονάδας *Branch*, αντίστοιχα. Οι τρείς έξοδοι *doa* των μνημών συνδέονται με τις εισόδους *br_in0, br_in1, br_in2* των μονάδων *stateA, stateB* και τις εισόδους *g_0, g_1, g_2* της μονάδας *LLR*.
- Τα κουτιά *stateAmemory 0-7* και *stateBmemory 0-7* αναπαριστούν τις οκτώ μνήμες *st_A_mem0-st_A_mem7* και τις οκτώ μνήμες *st_B_mem0-st_B_mem7* για τις μονάδες *stateA* και *stateB* αντίστοιχα. Για την κάθε μονάδα με τις αντίστοιχες μνήμες τις, οι έξοδοι των οκτώ αντίστοιχων μνημών συνδέονται με τις οκτώ εισόδους της κάθε μονάδας, ενώ οι οκτώ έξοδοι της κάθε μονάδας συνδέονται με τις εισόδους των οκτώ αντίστοιχων μνημών.

Επιπλέον, οι έξοδοι των μονάδων *state A* και *state B* συνδέονται σε concatenated μορφή με τις εισόδους *in_A* και *in_B* της μονάδας *LLR*, αντίστοιχα.

5.3.4 Επιμέρους μονάδες του συστήματος

Σε σχέση με τη λειτουργία του Decoder, στη συνέχεια παρουσιάζονται οι επιμέρους μονάδες υλοποίησής του:

- Branch Unit

Η μονάδα υπολογισμού των Branch Metrics δέχεται ως είσοδο τις τιμές των δύο αλληλουχιών που μεταδόθηκαν στον δέκτη καθώς και των a priori τιμών προηγούμενων αποκωδικοποιήσεων του frame. Οι τιμές αυτές προστίθενται μεταξύ τους όπως στην υλοποίηση σε λογισμικό για να παραχθούν οι αντίστοιχες έξοδοι. Η διαφορά με το λογισμικό είναι ότι από τη στιγμή που οι βασικοί υπολογισμοί είναι τρείς και τα υπόλοιπα branch metrics παίρνουν τιμές σύμφωνα με αυτές τις τρείς προσθέσεις, στη μονάδα branch γίνονται μόνο αυτές οι τρείς πράξεις. Η αντιστοίχιση των υπόλοιπων branch metrics γίνεται στις μονάδες *stateA* και *stateB*, όπου τα αποτελέσματα των τριών πράξεων αποτελούν είσοδο για την κάθε εξίσωση υπολογισμού των Forward και Backward metrics.

- State A Unit

Τα Forward Metrics αποτελούνται από οκτώ διαφορετικές τιμές (a0 – a7) για κάθε σύμβολο. Για τον υπολογισμό τους στο λογισμικό, χρησιμοποιήθηκε μία for loop όπου σε κάθε επανάληψη η συνάρτηση *max1* επιλέγει τη μεγαλύτερη τιμή μεταξύ δύο εισόδων ως αποτέλεσμα του υπολογισμού, για κάθε μία από τις οκτώ τιμές a. Οι είσοδοι αυτές διαμορφώνονται μέσω προσθέσεων μεταξύ των τιμών Forward και Branch Metrics του προηγούμενο συμβόλου του frame. Στην υλοποίηση σε VHDL χρησιμοποιήθηκε μνήμη RAM, συνδεδεμένη με τη μονάδα *stateA*. Η έξοδος της μνήμης αποτελεί είσοδο για τη μονάδα, και αντίστοιχα η έξοδος της μονάδας αποτελεί είσοδο για τη μνήμη. Με αυτόν τον τρόπο, από τη μία τα αποτελέσματα των υπολογισμών που εκτελεί η μονάδα εγγράφονται στη μνήμη, ενώ οι προηγούμενοι υπολογισμοί που εγγράφηκαν στη μνήμη διαβάζονται από τη μονάδα. Η διαχείριση της ανάγνωσης /εγγραφής από και προς τη μνήμη γίνεται από την FSM του συστήματος. Με αυτόν τον τρόπο υλοποιείται άτυπα η αντίστοιχη for loop του λογισμικού. Όσον αφορά τη διαμόρφωση των δύο τιμών μεταξύ των οποίων επιλέγεται η μεγαλύτερη, οι σχετικές προσθέσεις γίνονται κατά την κλήση της συνάρτησης *max1* στο λογισμικό. Στη VHDL, οι προσθέσεις γίνονται εξ’ αρχής για τον υπολογισμό όλων των τιμών προς σύγκριση, οι οποίες αποθηκεύονται προσωρινά σε signals. Στη συνέχεια, μέσα σε ένα process, οι συγκρίσεις πραγματοποιούνται με χρήση βρόγχων if.

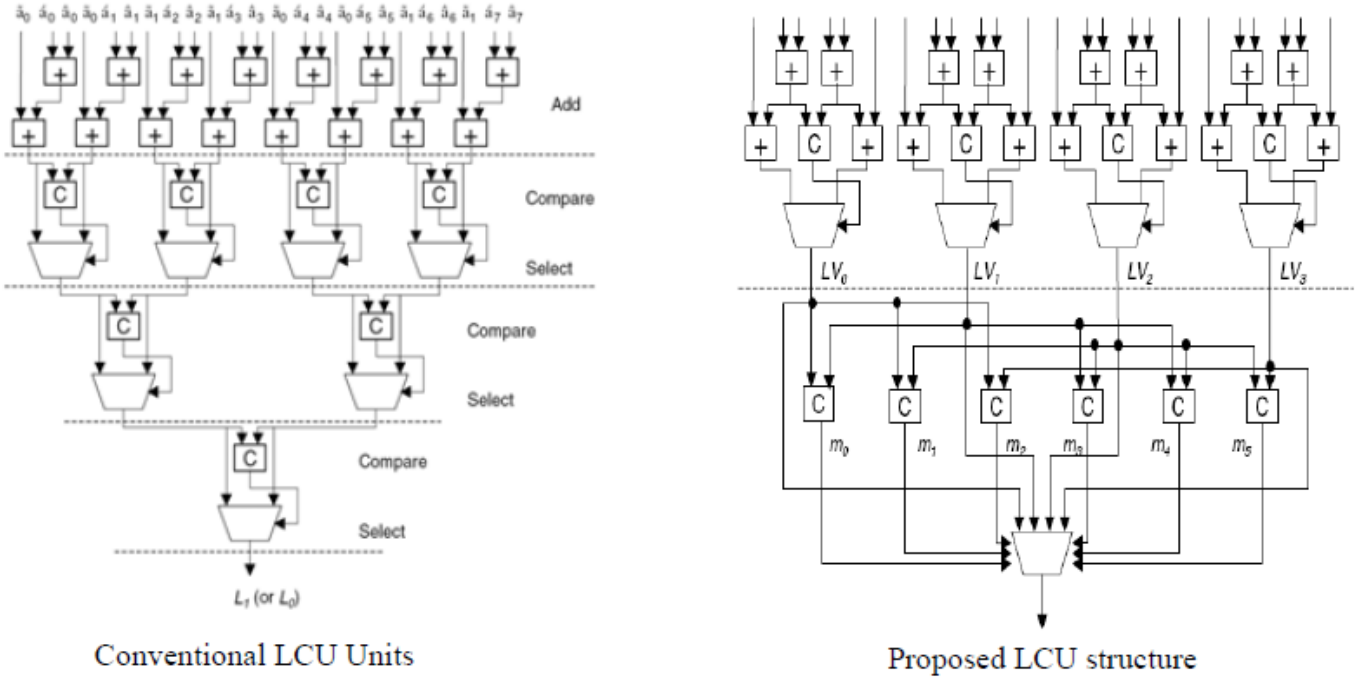
- State B Unit

Ο υπολογισμός των Backward Metrics γίνεται με αντίστοιχο τρόπο με τον υπολογισμό των Forward Metrics. Η βασική διαφορά είναι ότι οι υπολογισμοί γίνονται προς την αντίθετη κατεύθυνση από ότι στη μονάδα *stateA*. Επειδή τα Backward Metrics υπολογίζονται από το τελευταίο σύμβολο του frame προς το πρώτο, αντίστοιχα η FSM του συστήματος φροντίζει έτσι ώστε η αντίστοιχη μνήμη που συνδέεται με τη μονάδα

stateB να γεμίζει από την τελευταία θέση της προς την πρώτη. Κατά τα άλλα οι υπολογισμοί γίνονται όπως και στη μονάδα *stateA*, με τις αντίστοιχες πράξεις για τον υπολογισμό των Backward Metrics.

- LLR Unit

Για τον υπολογισμό των τιμών LLR στο λογισμικό διαμορφώνονται δύο οκτάδες τιμών, χρησιμοποιώντας τα Forward, Backward και Branch Metrics. Με χρήση της συνάρτησης *max2* επιλέγεται σε μία for loop η μεγαλύτερη από τις οκτώ τιμές για κάθε οκτάδα. Έτσι διαμορφώνονται δύο τιμές (*tmp1*, *tmp2*) , όπου το αποτέλεσμα της αφαίρεσης της *tmp2* από την *tmp1* αποτελεί την τιμή LLR για κάθε σύμβολο. Για τον υπολογισμό των τιμών LLR υλοποιήθηκαν δύο επιμέρους μονάδες (*ll0, ll1*) για τον υπολογισμό των δύο τιμών που στη συνέχεια θα αφαιρεθούν. Αρχικά γίνεται ο υπολογισμός των οκτώ τιμών που θα συγκριθούν για κάθε μονάδα. Στη συνέχεια η σύγκριση μεταξύ των τιμών δεν γίνεται σειριακά, όπως στο λογισμικό, αλλά ακολουθείται μία πιο παράλληλη διαδικασία. Για τη μείωση του critical path της κάθε επιμέρους μονάδας *llr*, ακολουθήθηκε η πρόταση της [26], για μία pipelined σχεδίαση. Η διαφορά των παραδοσιακών μονάδων σύγκρισης (LLR Computation Unit - LCU) με αυτή που προτείνεται φαίνεται στην παρακάτω εικόνα [26]:



Conventional LCU Units

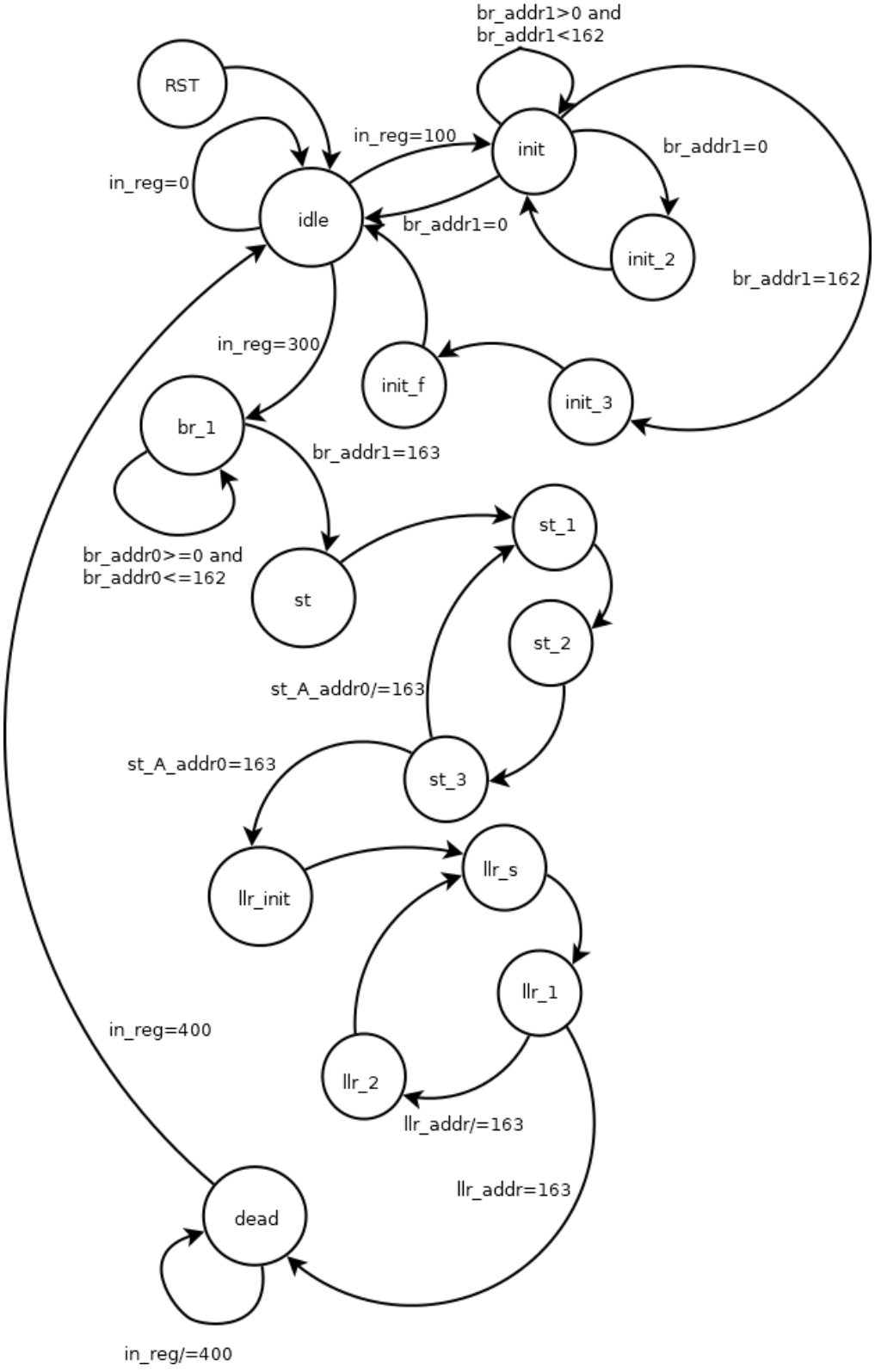
Proposed LCU structure

Εικόνα 5.68: Block Diagram της κλασσικής μονάδας LCU(αριστερά) και αυτής που προτείνεται (δεξιά)

Έτσι, οι τιμές συγκρίνονται αρχικά ανά δυάδες, καταλήγοντας σε τέσσερις τιμές. Η κάθε μία από τις τέσσερις αυτές τιμές συγκρίνεται με τις υπόλοιπες. Ανάλογα με το αποτέλεσμα των έξι συγκρίσεων, διαμορφώνεται η τιμή ενός signal vector πλάτους 6 bit. Στη συνέχεια ένας πολυπλέκτης επιλέγει μία εκ των τεσσάρων τιμών, σύμφωνα με την τιμή του signal vector.

5.3.5 Μονάδα Ελέγχου

Στη μονάδα ελέγχου υλοποιείται η Finite-State Machine που διαχειρίζεται της επιμέρους μονάδες του συστήματος. Στη συνέχεια παρουσιάζεται το διάγραμμα ροής της:



Εικόνα 5.7: Η FSM του συστήματος

- Idle State

Η FSM παραμένει στην κατάσταση αδράνειας μέχρι να διαβαστεί η τιμή ‘100’ στον καταχωρητή *in_reg*, που σηματοδοτεί την έναρξη της αρχικοποίησης των μνημών του συστήματος. Κατά το τέλος της αρχικοποίησης, η FSM επιστρέφει στην κατάσταση αδράνειας, μέχρι να εμφανιστεί η τιμή ‘300’ στον *in_reg*, όπου μεταβαίνει στην κατάσταση branch. Για οποιεσδήποτε άλλες τιμές του *in_reg*, η FSM παραμένει στην κατάσταση αδράνειας.

- Init States

Στις καταστάσεις αρχικοποίησης των μνημών, η FSM μεταβαίνει αρχικά από το initstate στο init_2, εγγράφοντας στις μνήμες την τιμή ‘0’ στην πρώτη θέση μνήμης. Από το init_2 η FSM επιστρέφει στο init, και παραμένει εκεί μέχρι να εγγραφεί η τιμή μηδέν σε όλες τις θέσεις. Στη συνέχεια η FSM μεταβαίνει στην κατάσταση init_3 για τον τερματισμό της εγγραφής, και από εκεί στην init_f, για την επιστροφή στην κατάσταση αδράνειας.

- Branch State

Κατά τη μετάβαση στο branch state, η FSM παραμένει εκεί μέχρι να ολοκληρωθεί η εγγραφή των αποτελεσμάτων της *Branch Unit* στις *Branch Memories*. Ύστερα μεταβαίνει στα ABstates.

- AB States

Λόγω της ανάγνωσης τιμών από τα StateA, StateB Units από τις ίδιες μνήμες στις οποίες εγγράφουν δεδομένα, η FSM μεταβαίνει διαδοχικά σε δύο καταστάσεις καθυστέρησης, για τον συγχρονισμό ανάγνωσης/εγγραφής. Έτσι, προσπελαύνει τις τρείς καταστάσεις υπολογισμού και εγγραφής των Forward, Backward Metrics στις αντίστοιχες μνήμες. Όταν η διαδικασία αυτή ολοκληρωθεί, η FSM μεταβαίνει στις καταστάσεις υπολογισμού των LLR τιμών.

- LLR states

Αντίστοιχα, επειδή η μονάδα LLR διαβάζει τα δεδομένα από όλες τις εσωτερικές μνήμες του συστήματος, για τον υπολογισμό των LLR τιμών, έχει προστεθεί μία κατάσταση καθυστέρησης, για τον συγχρονισμό του συστήματος. Όταν η εγγραφή των αποτελεσμάτων στις εξωτερικές μνήμες *out_mem_res* και *out_mem_ap* ολοκληρωθεί, η FSM μεταβαίνει στην κατάσταση τερματισμού της αποκωδικοποίησης.

- Final State

Στην τελική κατάσταση της FSM, εγγράφεται στον καταχωρητή *out_reg* η τιμή ‘100’, που σηματοδοτεί την έναρξη ανάγνωσης των αποτελεσμάτων αποκωδικοποίησης από το λογισμικό. Όταν από το τελευταίο εγγραφεί στον καταχωρητή *in_reg* η τιμή ‘400’, σηματοδοτείται ο τερματισμός ανάγνωσης των

αποτελεσμάτων. Έτσι η FSM μεταβαίνει στην κατάσταση αδράνειας. Μέχρι να συμβεί αυτό παραμένει στην τελική κατάσταση τερματισμού.

5.3.6 Resources

Στον παρακάτω πίνακα παρουσιάζονται τα resources της FPGA που καταναλώθηκαν κατά τη διαδικασία του τελικού place and route:

Number of Slice Registers	661	11,44	5%
Number of Slice LUTs	1,645	5,72	28%
Number of occupied Slices	534	1,43	37%
Number of MUXCYs used	980	2,86	34%
Number of bonded IOBs	22	102	21%
Number of RAMB16BWERs	0	32	0%
Number of RAMB8BWERs	23	64	35%
Number of BUFIO2/BUFIO2_2CLKs	0	32	0%
Number of BUFIO2FB/BUFIO2FB_2CLKs	0	32	0%
Number of BUFG/BUFGMUXs	3	16	18%
Number of DCM/DCM_CLKGENs	0	4	0%
Number of ILOGIC2/ISERDES2s	2	200	1%
Number of IODELAY2/IODRP2/IODRP2_MCBs	2	200	1%
Number of OLOGIC2/OSERDES2s	16	200	8%
Number of BSCANs	0	4	0%
Number of BUFHs	0	128	0%
Number of BUFPLLs	0	8	0%
Number of BUFPLL_MCBs	0	4	0%
Number of DSP48A1s	0	16	0%
Number of ICAPs	0	1	0%
Number of MCBs	0	2	0%
Number of PCILOGICSEs	0	2	0%
Number of PLL_ADVs	1	2	50%
Number of PMVs	0	1	0%
Number of STARTUPs	0	1	0%
Number of SUSPEND_SYNCs	0	1	0%
Average Fanout of Non-Clock Nets	2.94		

Πίνακας 5.1: Resources used

Το μέγιστο ρολόι για τη συγκεκριμένη σχεδίαση σύμφωνα με τα εργαλεία υλοποίησης της Xilinx ορίζεται στα 56.612 MHz. Παρόλα αυτά επιλέχθηκε αρκετά χαμηλότερη συχνότητα στο ρολόι της υλοποίησης, για λόγους σταθερότητας του συστήματος, από τη στιγμή που η κύρια στόχευση της υλοποίησης ήταν η όσο το δυνατό χαμηλότερη ενεργειακή κατανάλωση, και όχι η χρονική απόδοση. Έτσι, ο χρονισμός της FPGA στο συγκεκριμένο board ορίστηκε στα 24 MHz (περίοδος ρολογιού 40 ns). Για την ολοκλήρωση της διαδικασίας αποκωδικοποίησης απαιτούνται 1.125 κύκλοι ρολογιού από την FSM. Η χρονική διάρκεια αυτή υπολογίζεται από τη στιγμή που έχουν ήδη εγγραφεί από το λογισμικό τα δεδομένα εισόδου στις μνήμες ανάγνωσης της FPGA, και σηματοδοτείται η έναρξη αποκωδικοποίησης. Ως

τελευταίος κύκλος ρολογιού θεωρείται αυτός κατά τον οποίο σηματοδοτείται από την FPGA η έναρξη ανάγνωσης των αποτελεσμάτων από το λογισμικό. Καταλήγοντας για την αποκωδικοποίηση ενός εκ των δύο Decoder απαιτούνται από την FPGA 45 μs ανά frame.

5.3.7 Χρονική και Ενεργειακή Απόδοση του Max-Log MAP Decoder στο Board

Για τη μελέτη της λειτουργικότητας του Max-Log MAP Decoder στον σταθμό-βάση, έγινε λήψη μετρήσεων της χρονικής και ενεργειακής απόδοσης του board κατά τη διάρκεια της αποκωδικοποίησης. Για τη μέτρηση της ενεργειακής κατανάλωσης του board χρησιμοποιήθηκε το Smart Meter της ODROID. Οι παράμετροι της επικοινωνίας που προσομοιώθηκε είναι παρεμφερείς με αυτούς της προσομοίωσης σε περιβάλλον Matlab. Χρησιμοποιήθηκαν τυχαία frames μεγέθους 160bits, ο relative prime interleaver που παρουσιάστηκε στο Κεφάλαιο 4, ενώ δεν έγινε χρήση puncturing. Κατά τη δοκιμή του Decoder δεν έγινε χρήση της τεχνικής ARQ, αφού η διαδικασία λήψης μετρήσεων στόχευσε περισσότερο στη διεξαγωγή συμπερασμάτων από τη μελέτη της απόδοσης του Decoder για κάθε frame που αποκωδικοποιείται. Οι μετρήσεις που καταγράφηκαν αφορούν δύο αποκωδικοποιήσεις για κάθε frame. Η επιλογή αυτή έγινε με βάση τη βελτίωση της απόδοσης του Decoder κατά την πρώτη επανάληψη της αποκωδικοποίησης, σε σχέση με την επιπλέον χρονική καθυστέρηση αλλά και ενεργειακή κατανάλωση που θα επέφεραν επιπλέον αποκωδικοποιήσεις. Τέλος, κατά τη λήψη μετρήσεων, το board τροφοδοτήθηκε με ρεύμα τάσης 5.150V.

- Κατανάλωση Ενέργειας

Αρχικά μετρήθηκε η κατανάλωση ενέργειας στο board κατά την αποκωδικοποίηση σε επίπεδο λογισμικού. Για την αποκωδικοποίηση 10^4frames, καταναλώθηκαν 0,018 mWh.

Κατανάλωση λογισμικού ανά frame: 0,0018 mWh

Στη συνέχεια μετρήθηκε η κατανάλωση κατά τη χρήση hardware acceleration. Για την αποκωδικοποίηση 10^4frames, καταναλώθηκαν 1,537 mWh.

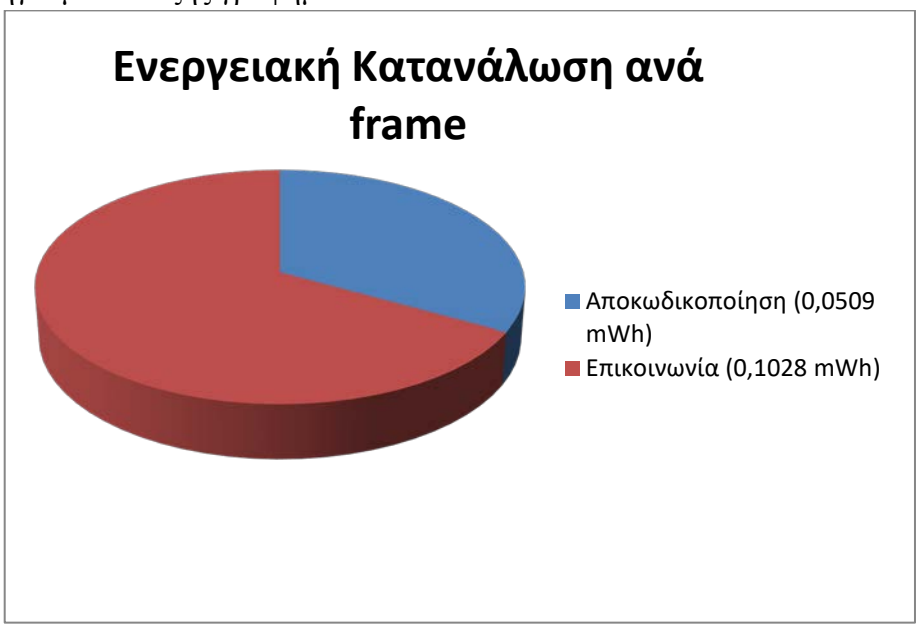
Κατανάλωση hardware accelerated λογισμικού ανά frame: 0,1537 mWh

Παρατηρείται μεγάλη απόκλιση μεταξύ των δύο καταναλώσεων. Για τον σχολιασμό της απόκλισης αυτής, και την αξιολόγηση της ενεργειακής απόδοσης του hardware acceleration, χρειάζεται να παρουσιαστούν μερικά ακόμη στοιχεία.

Για τον υπολογισμό του ενεργειακού κόστους επικοινωνίας board - FPGA cape (I/O), μετρήθηκε η ενεργειακή κατανάλωση κατά την ανάγνωση και εγγραφή κενών frames από και προς την FPGA, χωρίς η τελευταία να εκτελεί κάποια επεξεργασία. Το board κατανάλωσε 0,514 mWh για 10^4 "dummy"frames.

Κατανάλωση I/O ανά frame: 0.0514 mWh

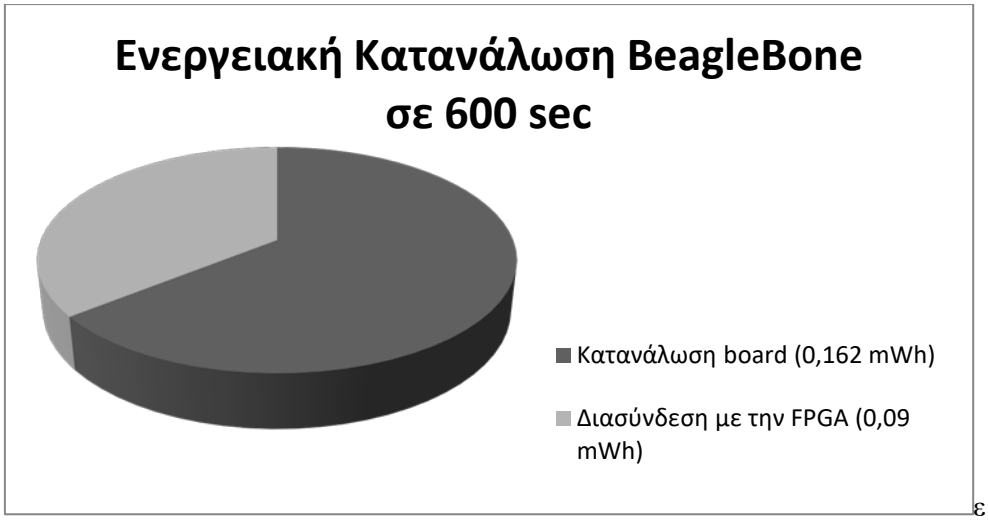
Κατά την αποκωδικοποίηση ενός frame, το κόστος του I/O που αναφέρθηκε προηγουμένως διπλασιάζεται, αφού απαιτείται ανάγνωση και εγγραφή από και προς την FPGA δύο φορές, μία για κάθε επιμέρους Decoder του σχήματος. Έτσι καταλήγουμε στο εξής γράφημα:



Γράφημα 5.1

Γίνεται αντιληπτό ότι το μεγαλύτερο μέρος της ενέργειας που καταναλώνεται αφορά την επικοινωνία board-FPGA, ενώ περίπου το 1/3 της ενέργειας αυτής καταναλώνεται στην επεξεργασία της FPGA.

Το επόμενο βήμα στη διεξαγωγή μετρήσεων ήταν η μελέτη της επίδρασης που έχει η σύνδεση του board με την FPGA στην ενεργειακή κατανάλωση. Για το σκοπό αυτό πραγματοποιήθηκαν μετρήσεις με την FPGA συνδεδεμένη στο board, «ανοιγμένη» από το λογισμικό, σε κατάσταση αδράνειας (με φορτωμένο το bitfile) χωρίς να εκτελείται κάποια διεργασία σε λογισμικό ή υλικό. Για 600 sec, καταναλώθηκαν 0,252 mWh. Στη συνέχεια, πραγματοποιήθηκαν μετρήσεις για με την FPGA αποσυνδεδεμένη. Το board κατανάλωσε 0,162 mWh σε κατάσταση αδράνειας, με το λειτουργικό φορτωμένο. Συνεπώς, παρατηρείται ότι κατά τη σύνδεση και μόνο με την FPGA, χωρίς η τελευταία να εκτελεί κάποια διεργασία, η ενεργειακή κατανάλωση αυξάνεται σχεδόν κατά 55%, όπως φαίνεται στο παρακάτω γράφημα:



ε

Γράφημα 5.2

Τελικά, τα αποτελέσματα των μετρήσεων σε σχέση με τη χρήση hardware acceleration για το συγκεκριμένο σύστημα ήταν απογοητευτικά. Για τον αλγόριθμο που υλοποιήθηκε και τα υλικά που χρησιμοποιούνται, η διασύνδεση με την FPGA για την υλοποίηση του decoding σε υλικό δεν ήταν ενεργειακά αποδοτική. Από την άλλη πλευρά, παρατηρήθηκε ότι η ύπαρξη και μόνο της FPGA στο σύστημα αύξησε κατά πολύ την ενεργειακή κατανάλωση. Επιπλέον, η κατανάλωση της FPGA κατά την αποκωδικοποίηση, δεν αφορούσε κυρίαρχα τη διαδικασία επεξεργασίας, αλλά την επικοινωνία με το board. Είναι ενδεικτικό ότι στο συγκεκριμένο σύστημα, τα 2/3 της ενεργειακής κατανάλωσης για την αποκωδικοποίηση ενός frame δαπανούνται στην εγγραφή του frame στην FPGA cape, και την ανάγνωση των αποτελεσμάτων αποκωδικοποίησης.

- Χρονική Απόδοση

Για τη μελέτη της χρονικής απόδοσης του συστήματος, έγιναν μετρήσεις στο λογισμικό, χρησιμοποιώντας τη συνάρτηση clock της βιβλιοθήκης time.h. Οι μετρήσεις αφορούν τόσο τη χρονική διάρκεια που χρειάζεται το λογισμικό για να προσομοιώσει το συνολικό σχήμα της υλοποίησης (κωδικοποίηση, μετάδοση, αποκωδικοποίηση), όσο και τη μέση χρονική διάρκεια αποκωδικοποίησης ενός frame.

Η χρονική διάρκεια δημιουργίας, κωδικοποίησης, αποστολής και αποκωδικοποίησης 10^4 frames από πλευράς λογισμικού είναι 48,25 sec. Αποκλειστικά για την αποκωδικοποίηση ενός frame από τον Decoder υλοποιημένο σε C, αφιερώνονται κατά μέσο όρο 0,468 ms. Παρατηρείται ότι αν διαιρεθεί η συνολική διάρκεια εκτέλεσης του λογισμικού με τον αριθμό frames, προκύπτει ότι χρειάζονται κατά μέσο όρο 0,482 ms για να δημιουργηθεί ένα τυχαίο frame, να κωδικοποιηθεί, να μεταδοθεί και να αποκωδικοποιηθεί. Λαμβάνοντας υπ' όψη τη μέση χρονική διάρκεια αποκωδικοποίησης, παρατηρείται ότι η διαδικασία αποκωδικοποίησης καταλαμβάνει το μεγαλύτερο ποσοστό του συνολικού χρόνου εκτέλεσης του λογισμικού.

Για την μελέτη των αποτελεσμάτων κατά τη χρήση της FPGA, πρέπει να ληφθεί υπ' όψη η χρήση halting intervals από πλευράς software, κατά την ανάγνωση και εγγραφή

από και προς την FPGA, για λόγους σταθερότητας στην επικοινωνία board - cape. Ενδεικτικά παρουσιάζονται αποτελέσματα μετρήσεων με intervals χρονικής διάρκειας 2ns. Η συνολική διάρκεια της κάθε μέτρησης αφορά τα στάδια data generation, encoding, modulation από πλευράς software, και το decoding για κάθε frame από πλευράς hardware. Αντίστοιχα με τη λήψη χρονικών μετρήσεων κατά τη λειτουργία του λογισμικού, πραγματοποιήθηκαν και κατά τη λειτουργία του συστήματος με hardware acceleration ξεχωριστές μετρήσεις για τη μέση χρονική διάρκεια αποκωδικοποίησης ενός frame. Η μέση χρονική διάρκεια αφορά το κομμάτι decoding, από πλευράς hardware, περιλαμβάνοντας το κομμάτι του I/O σε επίπεδο software.

Η χρονική διάρκεια δημιουργίας, κωδικοποίησης και αποστολής 10^4 frames από πλευράς λογισμικού, και αποκωδικοποίησης του χρησιμοποιώντας τον Decoder υλοποιημένο στην FPGA είναι κατά μέσο όρο 3614,44 sec. Για την αποκωδικοποίηση ενός frame από την FPGA, αφιερώνονται κατά μέσο όρο 350 ms. Αξίζει να σημειωθεί ότι για την αποκωδικοποίηση ενός frame συμβαίνουν 482 halting intervals διάρκειας 2ns, προκαλώντας συνολική χρονική καθυστέρηση 0,964 μs ανά frame. Με βάση την αρχιτεκτονική της υλοποίησης του Decoder στην FPGA, η FSM χρειάζεται 0,045 ms για την επεξεργασία ενός frame, από τη στιγμή που αυτό έχει εγγραφεί στις μνήμες εισόδου της FPGA, μέχρι τη στιγμή που ολοκληρώνεται η εγγραφή των αποτελεσμάτων στις μνήμες εξόδου. Αυτό συμβαίνει για την αποκωδικοποίηση ενός εκ των δύο επιμέρους decoders του συστήματος. Άρα, όταν το λογισμικό αποκωδικοποιεί ένα frame χρησιμοποιώντας τους δύο decoders σειριακά, και εκτελώντας δύο επαναλήψεις, η FSM του συστήματος χρειάζεται συνολικά 0,18 ms για την αποκωδικοποίηση. Σε σχέση με τα 350 ms που χρειάζονται κατά μέσο όρο για την αποκωδικοποίηση ενός frame από το σύστημα, παρατηρείται ότι τα στάδια της FSM της FPGA καταλαμβάνουν ένα πολύ μικρό ποσοστό της συνολικής χρονικής διάρκειας αποκωδικοποίησης. Με βάση τα προηγούμενα, προκύπτει το συμπέρασμα ότι όπως και κατά τη μελέτη της ενεργειακής κατανάλωσης, έτσι και στην περίπτωση της χρονικής απόδοσης, η επικοινωνία μεταξύ board – FPGA είναι ο πιο δαπανηρός τομέας του συστήματος.

Παρότι η χρήση halting intervals δεν επηρεάζει δραματικά τη χρονική απόδοση του I/O, παρατηρείται ότι η διαδικασία εγγραφής/ανάγνωσης μεταξύ λογισμικού και FPGA παρουσιάζει μεγάλη χρονική καθυστέρηση. Ένας παράγοντας που συμβάλει σε αυτό είναι η διαδικασία εγγραφής σε τρείς διαφορετικές μνήμες για την εισαγωγή των δεδομένων. Αντίστοιχα, η διαδικασία ανάγνωσης των αποτελεσμάτων από τις δύο μνήμες εξόδου είναι χρονοβόρα. Η λειτουργία της αποκωδικοποίησης στην FPGA περιλαμβάνει πληθώρα αναγνώσεων και εγγραφών από και προς διάφορες μνήμες RAM. Παρόλα αυτά, γίνεται εύκολα αντιληπτό ότι η διαχείριση των στοιχείων μνήμης εσωτερικά στην FPGA είναι πολύ πιο αποδοτική χρονικά, απ’ ότι η διαχείριση των εξωτερικών μνημών της FPGA από το λογισμικό.

6 Γενικά Συμπεράσματα

Στο κεφάλαιο αυτό παρουσιάζονται τα συμπεράσματα που προέκυψαν κατά την εκπόνηση της εργασίας. Αρχικά, σχολιάζεται η εφαρμογή Turbo Coding στους κόμβους ενός ΑΔΑ, με επίκεντρο τον σταθμό βάσης και τη λειτουργία του Turbo Decoder. Διεξάγονται συμπεράσματα για όλα τα στάδια έρευνας, προσομοίωσης και υλοποίησης του Decoder. Στη συνέχεια, παρουσιάζονται τα ζητήματα που προέκυψαν κατά την ανάπτυξη του Decoder, οι δυσκολίες που αντιμετωπίστηκαν, καθώς και λανθασμένες επιλογές που θα μπορούσαν να έχουν αποφευχθεί. Τέλος, αναφέρονται οι τομείς της εργασίας που επιδέχονται βελτίωσης, καθώς και μερικές κατευθύνσεις που αφορούν την εμβάθυνση της συνολικής ιδέας, τη μελλοντική επέκταση της εργασίας, και τη βελτιστοποίηση της διαδικασίας Turbo Coding.

6.1 Συμπεράσματα σχετικά με την εφαρμογή Turbo Coding σε κόμβους ΑΔΑ

- Σχετικά με την εφαρμογή FEC

Η βασική ιδέα που οδήγησε στην ανάπτυξη του Turbo Encoder της [9], καθώς και του Turbo Decoder που παρουσιάστηκε στη συγκεκριμένη εργασία, είναι πρακτικά ότι το ενεργειακό κόστος των αναμεταδόσεων είναι τελικά πολύ μεγαλύτερο από το ενεργειακό κόστος της επιπλέον επεξεργασίας (χρήση FEC) σε πομπό / δέκτη. Αυτό το χαρακτηριστικό αφορά τοπολογίες single-hop, όπου το κάθε πακέτο δεδομένων επεξεργάζεται μία φορά πριν την αποστολή του (κωδικοποίηση), και μία κατά την παραλαβή του από τον κόμβο-συλλέκτη (αποκωδικοποίηση). Με αυτό τον τρόπο περιορίζεται το ενεργειακό κόστος επεξεργασίας, ενώ ταυτόχρονα μειώνονται οι αναμεταδώσεις πακέτων που μεταδόθηκαν ανεπιτυχώς. Από την άλλη πλευρά, η εφαρμογή FEC σε μία κλασσική τοπολογία multi-hop δεν θα είχε αποτέλεσμα, αφού το κόστος επεξεργασίας τον πακέτων από τον κάθε ενδιάμεσο κόμβο θα επιβάρυνε δραματικά την ενεργειακή κατανάλωση των κόμβων. Από τα αποτελέσματα των προσομοιώσεων έγινε σαφές ότι σε ένα τηλεπικοινωνιακό σύστημα που χρησιμοποιεί αναμεταδόσεις πακέτων (ARQ), η εφαρμογή Turbo Coding είχε αξιοσημείωτη επίδραση στην επικοινωνία των κόμβων, αφού οι αναμεταδόσεις εκμηδενίστηκαν για τιμές θορύβου της τάξης 3-4 SNR_{dB} και άνω. Αυτό σε συνδυασμό με τα οφέλη που προσφέρει η δρομολόγηση single-hop (χαμηλό ενεργειακό κόστος αρχικοποίησης, αξιοπιστία λόγω λιτής δρομολόγησης των πακέτων, χαμηλή συχνότητα συντήρησης του δικτύου, κ.α.), καθιστά το σχήμα που περιγράφεται στην εργασία ιδιαίτερα ελκυστικό προς χρήση σε μία πραγματική εφαρμογή.

- Σχετικά με την τοπολογία που προτείνεται για την αξιοποίηση του Turbo Decoder που αναπτύχθηκε

Έχοντας ως κύριο περιορισμό την ανάπτυξη τοπολογίας που θα χρησιμοποιεί single-hop δρομολόγηση, προέκυψε το ζήτημα της έκτασης που θα μπορεί μία τέτοια τοπολογία να καλύψει. Σε αντίθεση με τοπολογίες multi-hop όπου οι ενδιάμεσοι κόμβοι δίνουν τη δυνατότητα τοποθέτησης των κόμβων-πηγών σε μεγάλες αποστάσεις από τον κόμβο-πύλη, μία κλασσική single-hop τοπολογία δεν θα μπορούσε να αποδώσει σε ρεαλιστικές συνθήκες υλοποίησης. Για τον λόγο αυτό προτείνεται ένα σχήμα πολλαπλών τοπολογιών αστέρα (single-hop clusters), που επικοινωνούν μεταξύ τους. Με αυτό τον τρόπο τα δεδομένα που συγκεντρώνονται στον κάθε κόμβο-συλλέκτη του δικτύου μπορούν να μεταδίδονται μέσω διαφορετικών clusters, καταλήγοντας στον σταθμό βάση. Έτσι προτείνεται η επέκταση της τοπολογίας, υιοθετώντας δρομολόγηση multi-hop, αφού πλέον οι κόμβοι-συλλέκτες έχουν το ρόλο ενδιάμεσων κόμβων, μεταξύ των κόμβων-πηγών και του σταθμού βάσης. Από τα αποτελέσματα της υλοποίησης του αλγόριθμου αποκωδικοποίησης Max-Log MAP στην πλακέτα BeagleBoard, προκύπτει ότι η τοπολογία αυτή είναι εφαρμόσιμη με χρήση συσκευών τέτοιου τύπου. Η αρχική υπόθεση ότι η εφαρμογή Turbo Decoding σε συσκευές αναβαθμισμένης πολυπλοκότητας σε σχέση με αυτές των κόμβων-πηγών, που διαθέτουν όμως τα κύρια χαρακτηριστικά ενός κόμβου ΑΔΑ (ενεργειακή αυτονομία, χαμηλό χρηματικό κόστος, κ.α.) μπορεί να είναι ενεργειακά αποδοτική, επιβεβαιώθηκε. Επιπλέον, η χρήση συσκευών παρόμοιας πολυπλοκότητας με το BeagleBoard για τους κόμβους-συλλέκτες, και συσκευών όπως το MicaZ [9] για τους κόμβους-πηγές σε μία τοπολογία single-hop clusters, προσδίδει μεγάλες δυνατότητες κλιμάκωσης σε ένα ΑΔΑ, για την κάλυψη εκτενών περιοχών τοποθέτησης, σε μία πραγματική εφαρμογή.

- Σχετικά με τους αλγόριθμους αποκωδικοποίησης που προσομοιώθηκαν:

Είναι σαφές ότι τα αποτελέσματα προσομοίωσης του αλγορίθμου BCJR δεν ήταν τα αναμενόμενα. Παρόλα αυτά, το ιδιαίτερο χαρακτηριστικό του αλγορίθμου να αποδίδει καλύτερα όσο μεγαλύτερο είναι το μέγεθος των frames που αποκωδικοποιεί είναι σίγουρα αξιοποιήσιμο. Η εκπόνηση της εργασίας έγινε έχοντας κατά νου εφαρμογές που χρησιμοποιούν συστήματα χαμηλής πολυπλοκότητας. Γι’ αυτό τον λόγο επιλέχθηκε εξ αρχής μικρό μέγεθος frame, που σχετίζεται με τη μετάδοση πληροφορίας μετρήσεων όπως ατμοσφαιρική πίεση, ταχύτητα ανέμου, θερμοκρασία κ.α., που αφορούν γεωργικές, περιβαλλοντικές ή οικιακές εφαρμογές. Όμως, ένα ΑΔΑ μπορεί να χρησιμοποιείται σε εφαρμογές όπου μεταδίδονται μεγάλα πακέτα δεδομένων, όπως η καταγραφή και αποστολή αρχείων βίντεο σε ένα σύστημα παρακολούθησης. Σε τέτοιου είδους εφαρμογές, η υλοποίηση Turbo Coding με χρήση του αλγορίθμου αποκωδικοποίησης BCJR μπορεί να αποδειχθεί ιδιαίτερα αποδοτική.

Σε σχέση με τον αλγόριθμο Max-Log MAP, θα πρέπει να σημειωθεί ότι η επιλογή μίας απλοποιημένης έκδοσης του έγινε συνειδητά, με βάση τη συμβατότητά του με

συστήματα περιορισμένων δυνατοτήτων. Η εφαρμογή ενός αλγορίθμου χαμηλής σχετικά πολυπλοκότητας σε μία πλακέτα με δυνατότητες hardware acceleration, έγινε υπό την κατεύθυνση ανάπτυξης ενός πρωτοτύπου που θα αποτελέσει τη βάση για την περεταίρω εξερεύνηση των δυνατοτήτων ενός τέτοιου σχήματος.

6.2 Ζητήματα που αντιμετωπίστηκαν, διδάγματα που προέκυψαν

- Αναζήτηση αλγορίθμου

Το πρώτο ζήτημα που τέθηκε κατά την εκπόνηση της εργασίας ήταν η εύρεση του κατάλληλου αλγόριθμου αποκωδικοποίησης. Το βασικό κριτήριο για την εύρεση αλγορίθμου ήταν το περιορισμένο area της υλοποίησης του αλγορίθμου, με γνώμονα τις δυνατότητες και τα χαρακτηριστικά των συστημάτων εφαρμογής του. Η αρχική σκέψη ήταν να αποφευχθεί κάποια παραλλαγή του αλγορίθμου Viterbi, μιας και ο συγκεκριμένος αλγόριθμος εμφανίζει υψηλή πολυπλοκότητα. Μετά από εκτεταμένη έρευνα, διαπιστώθηκε όλοι οι διαθέσιμοι αλγόριθμοι Turbo Decoding ακολουθούν τις βασικές αρχές του Viterbi. Στη βάση αυτή, αναζητήθηκε κάποιος εκ των αλγορίθμων αυτών, που να διαθέτει όσο το δυνατόν χαμηλότερη πολυπλοκότητα, και ταυτόχρονα να προσεγγίζει την απόδοση του Viterbi. Έτσι επιλέχθηκε να εξεταστούν δύο απλοποιημένες υλοποιήσεις των αλγορίθμων BCJR και Max-Log MAP.

- Έλλειψη εκτενούς documentation για τη CML

Παρότι η βιβλιοθήκη CML παρέχει πληθώρα διαθέσιμων παραμέτρων και επιλογών όσον αφορά τη διεξαγωγή εξομοιώσεων για τηλεπικοινωνιακά συστήματα, το documentation που προσφέρεται δεν είναι επαρκές για την εμβάθυνση στον τρόπο με τον οποίο λειτουργούν οι συναρτήσεις της. Δηλαδή, ενώ το documentation που διατίθεται είναι αρκετά κατατοπιστικό σε ότι αφορά τη χρήση της και τη διεξαγωγή προσομοιώσεων, δεν επαρκεί για την κατανόηση του κώδικα της βιβλιοθήκης. Έτσι, η διαδικασία ενσωμάτωσης στη βιβλιοθήκη ενός αυτόνομου κώδικα που υλοποιεί έναν Turbo Decoder ήταν πολύπλοκη και ιδιαίτερα χρονοβόρα. Από τη στιγμή που δεν υπάρχουν σχεδιαγράμματα ή σχόλια στον κώδικα Matlab για τον τρόπο με τον οποίο γίνεται η επεξεργασία των δεδομένων, έπρεπε να γίνει εκτενής ανάλυση του κώδικα της CML. Μετά από την καταγραφή του τρόπου λειτουργίας όλων των συναρτήσεων, η ενσωμάτωση του αυτόνομου κώδικα έγινε εφικτή. Στη συνέχεια ακολούθησαν εκτεταμένα test runs για να διαπιστωθεί η σωστή και απροβλημάτιστη λειτουργία του ενσωματωμένου κώδικα.

- Αδυναμία χρήσης των διαθέσιμων ZigBee modules

Προχωρώντας στην υλοποίηση του πειράματος στο board, η αρχική ιδέα ήταν η πλήρης εφαρμογή του πειράματος σε πραγματικές συνθήκες. Έτσι έγινε μία προσπάθεια να χρησιμοποιηθούν ZigBee modules που αναλαμβάνουν την ασύρματη επικοινωνία του board, για την αποστολή των frames σε πραγματικό κανάλι, αντίστοιχο

με τα κανάλια επικοινωνίας ενός ΑΔΑ. Τα modules που υπήρχαν διαθέσιμα ήταν αυτά που χρησιμοποιήθηκαν για τις ανάγκες της [9]. Αυτό που δεν κατέστησε δυνατή τη χρήση τους ήταν η παλαιότητα του υλικού, και άρα του διαθέσιμου λογισμικού υποστήριξής τους. Επειδή οι διαθέσιμοι οδηγοί των modules είναι συμβατοί μόνο με παλαιότερες διανομές λειτουργικών Linux, δεν μπόρεσαν να λειτουργήσουν επιτυχώς ούτε στο λειτουργικό που χρησιμοποιήθηκε για την ανάπτυξη κώδικα, ούτε κάνοντας χρήση Virtual Machine για την δοκιμή λειτουργικών παλαιότερων Kernel. Πολλώ δε μάλλον, η χρήση τους στη διαθέσιμη διανομή Linux για το BeagleBone (Ubuntu 14.04) ήταν ακατόρθωτη. Είναι προφανές ότι η παραμετροποίηση κάποιου παλαιότερου opensource λειτουργικού στα χαρακτηριστικά του BeagleBone δεν υπήρχε σαν επιλογή, αφού ξέφευγε αρκετά από το αντικείμενο της εργασίας. Τελικά, λόγω έλλειψης κάποιας άλλης διαθέσιμης συσκευής ασύρματης επικοινωνίας του αντίστοιχου προτύπου (IEEE 802.15.4) αποφασίστηκε να παραληφθεί υλοποίηση πραγματικού καναλιού επικοινωνίας. Η διεξαγωγή συμπερασμάτων σε σχέση με τη λειτουργία του Decoder σε επίπεδο λογισμικού και υλικού στο board έγινε με βάση τα αποτελέσματα των προσομοιώσεων στη Matlab.

- Έλλειψη documentation για την πλατφόρμα LOGI Bone

Παρότι η opensource διάθεση της πλατφόρμας Wishbone είναι κάτι που διευκολύνει ιδιαίτερα την υλοποίηση κάποιου project στο board, η έλλειψη documentation είτε στις υλοποιημένες μονάδες του Wishbone που αφορούν το LOGI Bone, είτε στα διαθέσιμα project που υπάρχουν online, αποτελεί εμπόδιο στην σύνθεση αυθεντικού κώδικα VHDL με τον κώδικα που υλοποιεί το Wishbone. Λόγω της έλλειψης documentation για τα components που υλοποιούν τη διασύνδεση BeagleBone – LOGI Bone, αρχικά αποπειράθηκε να παραληφθεί η χρήση τους. Στην πορεία έγινε εμφανές ότι η συγγραφή κώδικα από το μηδέν με βάση τα pins του FPGA cape και η διαχείριση του συστήματος συνολικά θα ήταν μία αναβαθμισμένη διαδικασία, κάτι που από μόνο του θα μπορούσε να αποτελέσει αντικείμενο εργασίας. Στη συνέχεια εξετάστηκαν διάφορα από τα διαθέσιμα υλοποιημένα project του LOGI Bone, με σκοπό την ανάλυση του τρόπο λειτουργίας των διαθέσιμων υλοποιημένων μονάδων. Έτσι έγινε ανάλυση του κάθε βασικού component και της λειτουργικότητας του, με αποτέλεσμα να καταγραφεί η συνολική λειτουργία της πλατφόρμας. Στο τέλος της διαδικασίας αυτής έγινε πλέον εφικτή η επικοινωνία του Decoder και των επιμέρους μονάδων του, με τις μονάδες του Wishbone που αναλαμβάνουν την αξιοποίηση των resources της FPGA και της επικοινωνίας της με το board.

- Διαχείριση πληθώρας μνημών στη VHDL

Η φύση του Decoder που υλοποιήθηκε στη VHDL είναι τέτοια, που η χρήση πολλών διαφορετικών στοιχείων μνήμης για την αποθήκευση των αποτελεσμάτων διάφορων υπολογισμών είναι απαραίτητη. Λόγω έλλειψης εμπειρίας στη διαχείριση τόσο μεγάλου αριθμού μνημών, η διαδικασία δημιουργίας, παραμετροποίησης και συγχρονισμού των μνημών μεταξύ τους ήταν περίπλοκη και χρονοβόρα. Αρχικά έγινε απόπειρα δημιουργίας όλων των μνημών με χρήση του Core Generator του Xilinx. Στη

συνέχεια, για να υπάρχει η δυνατότητα μεταφοράς και σύνθεσης του κώδικα σε διαφορετικές πλατφόρμες, προτιμήθηκε η υλοποίηση των μνημών με χρήση VHD components. Σε κάθε περίπτωση, η διαχείριση των μνημών, ιδιαίτερα από τη στιγμή που καταναλώνουν ένα σημαντικό ποσοστό των διαθέσιμων resources, ήταν μία σύνθετη διαδικασία. Βέβαια, το ζήτημα αυτό δεν ήταν σχετικό με τον αλγόριθμο που υλοποιήθηκε, ή το διαθέσιμο υλικό, αλλά είχε να κάνει με υποκειμενικούς παράγοντες που αφορούν την εκπόνηση της εργασίας.

- Συγχρονισμός BeagleBone – LOGI Bone

Ο συγχρονισμός του board με την FPGA, ιδιαίτερα σε ότι έχει να κάνει με την ανάγνωση και εγγραφή δεδομένων από και προς την FPGA αποτέλεσε μία εκτενή διαδικασία του τελικού σταδίου υλοποίησης. Παρόλο που στα διαθέσιμα open source projects που υπάρχουν online συμπεριλαμβάνεται κώδικας διαχείρισης της FPGA από πλευράς BeagleBone (κυρίως σε Python), κανένα από τα projects αυτά δεν περιέχει την προσπέλαση τόσων θέσεων μνήμης, ή υλοποίησης τέτοιου αριθμού μνημών στην FPGA γενικότερα. Παρότι ο Decoder λειτουργούσε επιτυχώς σε επίπεδο προσομοίωσης, και το λογισμικό στο Board διάβαζε και έγραφε επιτυχώς μεμονωμένες τιμές στην FPGA, η συνολική τους επικοινωνία ήταν αρχικά ανέφικτη. Στη συνέχεια, και μετά από πληθώρα δοκιμών και πειραμάτων, διαπιστώθηκε ότι μόνο με τη χρήση halting interval θα μπορούσε να επιτευχθεί ο μεταξύ τους συγχρονισμός. Μέσω των intervals επιτεύχθηκε τελικά ή ομαλή προσπέλαση των στοιχείων μνήμης από το board, με αποτέλεσμα την επιτυχή εφαρμογή hardware acceleration στο BeagleBone.

6.3 Μελλοντική Δουλειά, Προτάσεις Βελτίωσης

- Βελτίωση της υπάρχουσας υλοποίησης

Σε ό,τι αφορά τη συγκεκριμένη υλοποίηση, υπάρχει χώρος για βελτίωση τόσο του λογισμικού στο board, όσο και της υλοποίησης του Decoder στην FPGA. Αρχικά, υπάρχει η δυνατότητα περαιτέρω δοκιμής και βελτιστοποίησης του κώδικα C. Από τα αποτελέσματα που παρουσιάστηκαν στο Κεφάλαιο 5, προκύπτει ότι η επικοινωνία board – FPGA cape επηρεάζει δραματικά τη χρονική απόδοση του hardware acceleration. Είναι χαρακτηριστικό ότι το χρονικό διάστημα από τη στιγμή που έχουν ήδη εισαχθεί τα δεδομένα στις μνήμες εισόδου της FPGA μέχρι τη στιγμή που ολοκληρώνεται η εγγραφή των αποτελεσμάτων στις μνήμες εξόδου είναι μικρότερο από το αντίστοιχο διάστημα που χρειάζεται ο Decoder του λογισμικού στο BeagleBoard. Παρόλα αυτά, η μέση χρονική διάρκεια αποκωδικοποίησης ενός frame με εφαρμογή hardware acceleration είναι πολλαπλάσια από την αντίστοιχη διάρκεια αποκωδικοποίησης αποκλειστικά στο software. Κάτι τέτοιο υποδεικνύει ότι χρειάζεται εκτεταμένη διαδικασία βελτιστοποίησης του λογισμικού που διαχειρίζεται την FPGA, έτσι ώστε να αξιοποιηθούν επαρκώς οι δυνατότητές της. Σε γενικές γραμμές, η ανάπτυξη του λογισμικού έγινε με βάση τη σταθερότητα του συστήματος και όχι τη χρονική απόδοση, έτσι ώστε να διεξαχθούν κάποια πρώτα συμπεράσματα για το συνολικό εγχείρημα.

Επιπλέον, η υλοποίηση του Decoder στη VHDL (ή σε HDL γενικότερα) είναι κάτι που σίγουρα επιδέχεται βελτίωσης. Αξίζει να σημειωθεί ότι η φύση του decoder είναι τέτοια που δεν επιτρέπει τη διεξαγωγή υπολογισμών με παράλληλο τρόπο. Ο τρόπος λειτουργίας του επιβάλλει την σειριακή επεξεργασία δεδομένων, αφού για κάθε στάδιο αποκωδικοποίησης, είναι απαραίτητη η πρόσβαση στα αποτελέσματα του προηγούμενου σταδίου (Branch Metrics, Forward & Backward Metrics, LLR values). Παρόλα αυτά, η λειτουργία του Decoder θα μπορούσε να βελτιστοποιηθεί αυξάνοντας τη συχνότητα του ρολογιού του συστήματος. Στη συγκεκριμένη υλοποίηση δεν αξιοποιήθηκαν πλήρως οι δυνατότητες της FPGA. Επιλέχθηκε να οριστεί περίοδος ρολογιού αρκετά μεγαλύτερη από την ελάχιστη δυνατή από πλευράς υλικού. Ενδεικτικά η περίοδος τέθηκε στα 40 ns ενώ η ελάχιστη δυνατή, όπως προέκυψε από τα reports του Xilinx IDE, ήταν περίπου 17 ns. Στόχος της υλοποίησης ήταν η σταθερότητα του συστήματος, ιδιαίτερα από τη στιγμή που ακόμα και αν διπλασιαζόταν η συχνότητα του ρολογιού της FPGA, η βελτίωση στη χρονική απόδοση του hardware acceleration θα ήταν ελάχιστή, λόγω των χρονικών καθυστερήσεων του I/O. Καταλήγοντας, αν ξεπεραστούν τα ζητήματα επικοινωνίας μεταξύ board και FPGA, υπάρχουν πολλά περιθώρια βελτίωσης της υλοποίησης από πλευράς χρονισμού στο υλικό.

- Εφαρμογή του Decoder σε διαφορετικές συσκευές

Από τη στιγμή που διαπιστώθηκε η ομαλή λειτουργία του Decoder στο BeagleBone και αποδείχθηκαν έγκυρες οι αρχικές εκτιμήσεις για την απόδοση του Turbo Coding στην προτεινόμενη τοπολογία, η εφαρμογή του Decoder σε συσκευές με διαφορετικά χαρακτηριστικά είναι σίγουρα ένας τομέας άξιος προς εξερεύνηση. Ένα board με την επεξεργαστική ισχύ του BeagleBone μπορεί να εκτελεί τη λειτουργία αποκωδικοποίησης με ιδιαίτερα καλή απόδοση από τη μία, από την άλλη όμως τα αναβαθμισμένων δυνατοτήτων κυκλώματά του επιφέρουν υψηλά επίπεδα ενεργειακής κατανάλωσης. Η εκτέλεση του αλγορίθμου αποκωδικοποίησης σε ένα board με πιο περιορισμένα τεχνικά χαρακτηριστικά θα μπορούσε να παρουσιάσει σημαντικά βελτιωμένη ενεργειακή κατανάλωση, χωρίς να επηρεάζεται σε ανάλογο βαθμό η χρονική του απόδοση.

Αντίστοιχα, η υλοποίηση του αλγορίθμου σε αναδιατασσόμενη λογική, όπως φαίνεται στην κατανάλωση των resources της Spartan 6 που παρουσιάστηκε στο Κεφάλαιο 5, μπορεί να πραγματοποιηθεί σε συσκευή FPGA με λιγότερα resources και άρα μικρότερη συνολική κατανάλωση. Φυσικά το μεγαλύτερο κέρδος θα μπορούσε να προκύψει αν η hardware-accelerated υλοποίηση του αλγορίθμου αποκωδικοποίησης συνδυαζόταν με ένα σύστημα πολύ χαμηλότερων δυνατοτήτων (και αντίστοιχα χαμηλότερης κατανάλωσης ισχύος) από το BeagleBoard, όπως για παράδειγμα μια συσκευή τύπου MicaZ ή Arduino. Θα πρέπει να σημειωθεί βέβαια ότι κάτι τέτοιο απαιτεί επιπλέον διερεύνηση καθότι η συσκευή αυτή θα πρέπει να είναι επαρκώς ισχυρή ώστε να μπορεί να ανταποκριθεί στο ρόλο της ως σταθμός βάσης.

- Ανίχνευση Θορύβου

Η τελική πρόταση βελτιστοποίησης της εφαρμογής Turbo Coding στους κόμβους ενός ΑΔΑ αφορά την ανίχνευση θορύβου. Η πρόταση αυτή αφορά την ανάπτυξη τεχνικών αξιολόγησης των επιπέδων θορύβου στα κανάλια επικοινωνίας είτε από τους κόμβους πηγές ή κόμβους αναμεταδότες, είτε από το σταθμό βάση του δικτύου. Με τον τρόπο αυτό θα υπάρχει η δυνατότητα αυτοματοποιημένης επιλογής μεταξύ της αποστολής πακέτων δεδομένων χωρίς κωδικοποίηση και της αποστολής κωδικοποιημένων πακέτων στο δίκτυο. Από τα αποτελέσματα της προσομοίωσης διαφαίνεται ότι η τεχνική αυτή μπορεί να αποδειχθεί ευεργετική για την ενεργειακή κατανάλωση των κόμβων. Σε συνθήκες χαμηλών τιμών θορύβου, θα μπορεί να αποφευχθεί η αποστολή πλεονάζουσας πληροφορίας που επιφέρει η εφαρμογή FEC. Σε συνθήκες υψηλού θορύβου η αξιοπιστία στη μετάδοση βελτιώνεται δραματικά, ενώ η ενεργειακή κατανάλωση μειώνεται λόγω των περιορισμένων πολλαπλών αποστολών πακέτων, όπως έχει ήδη αποδειχθεί. Μία υβριδική υλοποίηση που θα συνδυάζει τα πλεονεκτήματα της κωδικοποίησης δεδομένων με την «έξυπνη» διαχείριση των συνθηκών επικοινωνίας σε ένα πραγματικό κανάλι διαφαίνεται πολλά υποσχόμενη, αφού θα μπορούσε να παρουσιάσει αξιοσημείωτα αποτελέσματα σε διάφορες εφαρμογές.

7 Βιβλιογραφία

[1] Vivek Mhatre, Catherine Rosenberg, "Design guidelines for wireless sensor networks: communication, clustering and aggregation", Ad Hoc Networks 2 (2004) 45–63

[2] Gaurav Gupta, Mohamed Younis, "Fault-Tolerant Clustering of Wireless Sensor Networks"

[3] Gurwinder Kaur, Rachit Mohan Garg, "Energy Efficient Topologies for Wireless Sensor Networks", International Journal of Distributed and Parallel Systems (IJDPS) Vol.3, No.5, September 2012

[4] Kay Romer, Friedemann Mattern, "The Design Space of Wireless Sensor Networks"

[5] Rex Min and Anantha Chandrakasan, "Energy-Efficient Communication for High Density Networks"

[6] Daniel Schmidt, Matthias Berning, Norbert Wehn, "Error Correction in Single-Hop Wireless Sensor Networks - A Case Study"

[7] Jennifer Yick, Biswanath Mukherjee, Dipak Ghosal, "Wireless sensor network survey", Computer Networks 52 (2008) 2292–2330

[8] I.F. Akyildiz, W. Su*, Y. Sankarasubramaniam, E. Cayirci, "Wireless sensor networks: a survey", Computer Networks 38 (2002) 393–422

[9] Andreas Brokalakis, Ioannis Papaefstathiou, "Using Hardware-Based Forward Error Correction to Reduce the Overall Energy Consumption of WSNs"

[10] Martin Haenggi, "Twelve Reasons not to Route over Many Short Hops"

[11] Martin Haenggi, Daniele Puccinelli, "Routing in Ad Hoc Networks: A Case for Long Hops", IEEE Communications Magazine, October 2005

[12] L.C. Zhong, J.M. Rabaey, A. Wolisz, "Does proper coding make single hop wireless sensor networks reality: the power consumption perspective", IEEE Communications Society / WCNC 2005

[13] MEKKAOUI Kheireddine, RAHMOUN Abdellatif, "Short-hops vs. Long-hops - Energy efficiency analysis in Wireless Sensor Networks"

[14] Marcos Tomio Kakitani, Glauber Brante, Richard Demo Souza and Anelise Munaretto, "Comparing the Energy Efficiency of Single-Hop, Multi-Hop and Incremental Decode-and-Forward in Multi-Relay Wireless Sensor Networks", 2011 IEEE 22nd International Symposium on Personal, Indoor and Mobile Radio Communications

[15] Kewei Sha, Jegnesh Gehlot, Robert Greve, "Multipath Routing Techniques in Wireless Sensor Networks: A survey", Wireless Pers Commun (2013) 70:807–829

[16] Sui Luyi, Fu Jinyi, Yang Xiaohua, "Forward Error Correction", 2012 Fourth International Conference on Computational and Information Sciences

[17] Charan Langton, "Intuitive Guide to Principles of Communications"

[18] Bernard Sklar, "A Primer on Turbo Code Concepts", IEEE Communications Magazine, December 1997

[19] Maha George Zia, Khaleel Abdul Khader Saleh, "Comparison between Modified Bahl, Cocke, Jelinek, and Raviv (BCJR) and Soft Output Viterbi Algorithm (SOVA) Decoders Over Additive White", Journal of Engineering and Development, Vol. 11, No. 3, December (2007)

[20] Yogesh Kumar Soniwal, " TURBO encoder & decoder implementation and its performance over AWGN channel for BPSK modulated symbols "

[21] Iterative Solutions Coded Modulation Library, *http://www.iterativesolutions.com/Matlab.htm*

[22] Mihai Timis, "Design of LOG-MAP/MAX-LOG MAP Decoder", Annals. Computer Science Series. 5th Tome 1st Fasc. - 2007

[23] Anum Imran, "SOFTWARE IMPLEMENTATION AND PERFORMANCE OF UMTS TURBO CODE"
[24] Silvio A. Abrantes, "From BCJR to turbo decoding: MAP algorithms made easier"
[25] Vinay kumar Reddy, Log-MAP and Max-Log MAP
[26] M.Mathana , Dr.P.Rangarajan, "FPGA Implementation of High Speed Architecture for Max Log Map Turbo SISO Decoder"