ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΜΕ ΕΦΑΡΜΟΓΕΣ ΣΤΗ ΒΙΟΙΑΤΡΙΚΗ

«Ολοκληρωμένο ψηφιακό σύστημα επεξεργασίας βιοσημάτων του εγκεφάλου για την ανάδειξη ατραπών»

Γκόντα Μαρία

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Υπεύθυνος

Κακαρούντας Αθανάσιος

Λαμία, 2019

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΜΕ ΕΦΑΡΜΟΓΕΣ ΣΤΗ ΒΙΟΙΑΤΡΙΚΗ

«Ολοκληρωμένο ψηφιακό σύστημα επεξεργασίας βιοσημάτων του εγκεφάλου για την ανάδειξη ατραπών»

Γκόντα Μαρία

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Υπεύθυνος

Κακαρούντας Αθανάσιος

Λαμία, 2019

Με ατομική μου ευθύνη και γνωρίζοντας τις κυρώσεις [1], που προβλέπονται από της διατάξεις της παρ. 6 του άρθρου 22 του Ν. 1599/1986, δηλώνω ότι:

1. *Δεν παραθέτω κομμάτια βιβλίων ή άρθρων ή εργασιών άλλων αυτολεξεί* ***χωρίς να τα περικλείω σε εισαγωγικά*** *και χωρίς να αναφέρω το συγγραφέα, τη χρονολογία, τη σελίδα. Η αυτολεξεί παράθεση χωρίς εισαγωγικά χωρίς αναφορά στην πηγή, είναι λογοκλοπή. Πέραν της αυτολεξεί παράθεσης, λογοκλοπή θεωρείται και η παράφραση εδαφίων από έργα άλλων, συμπεριλαμβανομένων και έργων συμφοιτητών μου, καθώς και η παράθεση στοιχείων που άλλοι συνέλεξαν ή επεξεργάσθηκαν, χωρίς αναφορά στην πηγή. Αναφέρω πάντοτε με πληρότητα την πηγή κάτω από τον πίνακα ή σχέδιο, όπως στα παραθέματα.*

2. *Δέχομαι ότι η αυτολεξεί* ***παράθεση χωρίς εισαγωγικά****, ακόμα κι αν συνοδεύεται από αναφορά στην πηγή σε κάποιο άλλο σημείο του κειμένου ή στο τέλος του, είναι αντιγραφή. Η αναφορά στην πηγή στο τέλος π.χ. μιας παραγράφου ή μιας σελίδας, δεν δικαιολογεί συρραφή εδαφίων έργου άλλου συγγραφέα, έστω και παραφρασμένων, και παρουσίασή τους ως δική μου εργασία.*

3. *Δέχομαι ότι υπάρχει επίσης περιορισμός στο μέγεθος και στη συχνότητα των παραθεμάτων που μπορώ να εντάξω στην εργασία μου εντός εισαγωγικών. Κάθε μεγάλο παράθεμα (π.χ. σε πίνακα ή πλαίσιο, κλπ), προϋποθέτει ειδικές ρυθμίσεις, και όταν δημοσιεύεται προϋποθέτει την άδεια του συγγραφέα ή του εκδότη. Το ίδιο και οι πίνακες και τα σχέδια*

4. *Δέχομαι όλες τις συνέπειες σε περίπτωση λογοκλοπής ή αντιγραφής.*

Ημερομηνία:25/09/2019

Η δηλούσα

Γκόντα Μαρία

(1) «Όποιος εν γνώσει του δηλώνει ψευδή γεγονότα ή αρνείται ή αποκρύπτει τα αληθινά με έγγραφη υπεύθυνη δήλωση του άρθρου 8 παρ. 4 Ν. 1599/1986 τιμωρείται με φυλάκιση τουλάχιστον τριών μηνών. Εάν ο υπαίτιος αυτών των πράξεων σκόπευε να προσπορίσει στον εαυτόν του ή σε άλλον περιουσιακό όφελος βλάπτοντας τρίτον ή σκόπευε να βλάψει άλλον, τιμωρείται με κάθειρξη μέχρι 10 ετών.

«Ολοκληρωμένο ψηφιακό σύστημα επεξεργασίας βιοσημάτων του εγκεφάλου για την ανάδειξη ατραπών»

Γκόντα Μαρία

Τριμελής Επιτροπή:

Αθανάσιος Κακαρούντας, Επίκουρος Καθηγητής (επιβλέπων)

Ιωάννης Αναγνωστόπουλος, Αναπληρωτής Καθηγητής

Θανάσης Λουκόπουλος, Επίκουρος Καθηγητής

Περίληψη

Η μελέτη των βιολογικών σημάτων αποτελεί σημαντικό και χρησμό κομμάτι για την διάγνωση πολλών ασθενειών και δυσλειτουργιών του ανθρώπου. Ωστόσο θα επικεντρωθούμε σε αυτές που αφορούν τον ανθρώπινο εγκέφαλο όπως η επιληψία. Μελετώντας τα εγκεφαλικά σήματα (EEG) διαπιστώθηκε ότι στις καταγραφές αυτών εμφανίζονται φαινόμενα θορύβου και παρασίτων. Τα σήματα που είναι χρήσιμα για τον εντοπισμό επιληπτικών κρίσεων προέρχονται από καταγραφές κατά την διάρκεια του ύπνου. Επομένως είναι γεμάτα με πληροφορίες που δεν αφορούν την επιληψία και διαστρεβλώνουν το περιεχόμενο της.

Κατά αυτόν τον τρόπο προέκυψε η ιδέα δημιουργίας ενός συστήματος προ επεξεργασίας του ακατέργαστου σήματος με στόχο την παραγωγή καθαρής πληροφορίας απαλλαγμένη από άχρηστές τιμές. Αυτό το σύστημα αποτελείται από τρία βασικά φίλτρα: ένα Βαθυπερατό φίλτρο, ένα υψιπερατό φίλτρο και ένα φίλτρο κινούμενου μέσου όρου. Και τα τρία αυτά φίλτρα αποτελούν βασική αλλά ιδιαίτερη σημαντική επεξεργασία σήματος, απαλλάσσοντας το από φαινόμενα θορύβου και διατηρώντας τα τμήματα που είναι απαραίτητα για την διεξαγωγή αποτελεσμάτων.

Το σύστημα σχεδιαστικέ με σκοπό τη χρήση του σε διάφορα βιολογικά σήματα καθώς και να παρέχει προσαρμοστικότητα στις απαιτήσεις του κάθε χρήστη. Σημαντικό κομμάτι της υλοποίηση του αποτελεί η ικανότητα αποθήκευσης των δεδομένων σε διάφορα στάδια της ροής του συστήματος, καθώς ο χρήστης μπορεί να επιλέξει να εφαρμόσει παραπάνω από το ένα φίλτρο.

Λέξεις κλειδιά: επιληψία, EEG, ύπνος, Βαθυπερατό φίλτρο, Υψιπερατό φίλτρο, φίλτρο κινούμενου μέσου όρου

Abstract

The study of biological signals is an important and useful part of the diagnosis of many human diseases and dysfunctions. However, we will focus on those having to do with the human brain such as epilepsy. A study of brain signals (EEGs) revealed that noise phenomena appear in these recordings. The signals that are useful for detecting seizures come from recordings during sleep. It is therefore full of information that is not connected with epilepsy and distorts its content.

This gave rise to the idea of creating a signal processing system for the purpose of producing pure information free of deform. This system consists of three basic filters: a high pass filter, a low pass filter and a moving average filter. All three of these filters provide basic but very important signal processing, freeing it from noise and keeping the parts necessary for having conclusions.

The system is designed to be used in a variety of bio-signals as well as to adapt to each user's requirements. An important part of its implementation is the ability to store data at various stages of the system, since the user can choose to apply more than one filter.

Key words: epilepsy, EEG, sleep, high pass filter, low pass filter, moving average filter

Ευχαριστίες

Η παρούσα πτυχιακή εργασία εκπονήθηκε στο πλαίσιο των προπτυχιακών μας σπουδών στο Τμήμα Πληροφορικής με Εφαρμογές στην Βιοϊατρική τεχνολογία του Πανεπιστημίου Θεσσαλίας. Με την ολοκλήρωση της πρώτης επιστημονικής προσπάθειας, επιθυμώ να εκφράσω την εκτίμηση, τον σεβασμό και τις ευχαριστίες στον επιβλέποντα καθηγητή, κύριο Κακαρούντα Αθανάσιο για την πολύτιμη βοήθεια και την καθοδήγησή του καθ' όλη τη διάρκεια της φοίτησης μου.

Επίσης, θέλω να ευχαριστήσω την οικογένεια μου, που αποτελεί ανεκτίμητος αρωγός στην πορεία της ζωής μου, καθώς και τους φίλους μου για την εκτίμηση, την εμπιστοσύνη τους και την αμέριστη ψυχολογική τους ενίσχυση. Η συμβολή και η αρωγή του καθενός από τους προαναφερθέντες υπήρξε καθοριστικής σημασίας στη διεκπεραίωση της πτυχιακής.

Περιεχόμενα

Ευρετήριο Εικόνων

Ευρετήριο Πινάκων

1. Εισαγωγή

Παγκοσμίως περισσότερο από 50 χιλιάδες άνθρωποι πάσχουν από επιληψία. Η επιληψία είναι μια από τις πιο συνηθισμένες νευρολογικές διαταραχές που σχετίζονται με παροδικές, επαναλαμβανόμενες και απρόβλεπτες επιληπτικές κρίσεις. Με την παρακολούθηση των ηλεκτροεγκεφαλικών σημάτων (EEG) μπορούν να διαγνωστούν και να εντοπιστούν οι επιληπτικές κρίσεις.

Στην καταγραφή αυτών των σημάτων υπάρχουν χρήσιμες πληροφορίες για τους νευρολόγους, ώστε να προβούν σε κατάλληλες θεραπείες για τους ασθενείς. Ωστόσο η παρακολούθηση των ηλεκτροεγκεφαλικών σημάτων είναι μια ακριβή και χρονοβόρα διαδικασία. Έτσι έχει κριθεί σημαντική η ύπαρξη ενός συστήματος αυτόματης αναγνώρισης επιληψιών είναι ιδιαίτερα χρήσιμο και μπορεί να οδηγήσει σε βελτιώσει της διαδικασίας θεραπείας των επιληψιών.

Η ύπαρξη ενός τέτοιου συστήματος απαιτεί ένα μεγάλο κομμάτι προ διεργασίας, καθώς η καταγραφή των ηλεκτοεγκεφαλικών σημάτων προσφέρει ένα αδιαμόρφωτο σήμα. Το οποίο σήμα χρειάζεται να περάσει από διάφορα στάδια επεξεργασίας ώστε εν τέλει να είναι χρήσιμο για την διάγνωση των επιληψιών.

1.1 Κίνητρο, σκοπός και στόχος πτυχιακής εργασίας

Έπειτα από μελέτη έχει διαπιστωθεί η χρησιμότητα των ηλεκτροεγκεφαλικών σημάτων για τον εντοπισμό και την διάγνωση πολλών νευρολογικών ασθενειών μεταξύ των διαταραχών του ύπνου και της επιληψίας όπως προαναφέρθηκε. Η καταγραφή αυτών των σημάτων γίνεται με ηλεκτροεγκεφαλογράφημά. Οι ιδιότητες του ΗΕΓ χαρακτηρίζονται ως πολύπλοκες, εξαιτίας της πολυπλοκότητας του νευρικού συστήματος. Ένα ΗΕΓ θεωρείται μία γραμμική στοχαστική διαδικασία με πολλές ομοιότητες με το θόρυβο. Χαρακτηριστικά όπως ο θόρυβος, παράσιτα τα οποία οφείλονται στην καρδιακή λειτουργία, σε μυϊκές κινήσεις, σε οφθαλμικές κινήσεις και στο ίδιο το σύστημα καταγραφής, μπορεί επίσης να αλλοιώσουν το σήμα. Επίσης οι κυματομορφές μπορεί να περιέχουν μια σύνθεση κανονικών ημιτονοειδών κυμάτων, ακανόνιστων αιχμών ή πολυαιχμών, ατράκτων ή πολυατράκτων. Στις περισσότερες παθολογικές καταστάσεις όπως είναι οι επιληπτικές κρίσεις το ΗΕΓ επιδεικνύει εμφανή παραδοξότητα. Πολύ συχνά στις Βιοχαρτικές εφαρμογές η λήψη του σήματος δεν είναι αρκετή. Απαιτείται η επεξεργασία του σήματος για την εξαγωγή της χρήσιμης πληροφορίας που είναι «κρυμμένη» σε αυτό. Επομένως υπάρχει η ανάγκη για την δημιουργεί ενός συστήματος επεξεργασίας βιολογικών σημάτων για την εξασφάλιση της καθαρότητας της πληροφορίας.

Μελετώντας τις ανάγκες για καθαρισμό και επεξεργασία των σημάτων, στόχος της συγκεκριμένης πτυχιακής εργασίας είναι η δημιουργία και η σχεδίαση ενός

συστήματος επεξεργασίας των ηλεκτροεγκεφαλικών σημάτων με στόχο τη χρήση τους στη διάγνωση επιληπτικών κρίσεων.

Το σύστημα που υλοποιείται στην συγκεκριμένη πτυχιακή εργασία θα επιτρέπει στον χρήστη να επιλέγει ην επεξεργασία που θέλει να εφαρμόσει στο σήμα. Έτσι το σύστημα θα μπορεί να εφαρμοστεί σε διάφορα είδη σημάτων εκτός των ηλεκτροεγκεφαλικών. Σκοπός είναι να αποτελεί ένα εύχρηστο, ευέλικτο τεχνολογικό εργαλείο ώστε να είναι χρήσιμο σε διάφορους τομείς που απαιτούν την χρήση πληροφοριών από βιοσήματα. Η πτυχιακή εργασία έχει δομηθεί σύμφωνα με τα παρακάτω:

Στο δεύτερο κεφάλαιο, επιχειρείται η εννοιολογική προσέγγιση των εννοιών βιοσήμα και ύπνος. Ακόμη παρουσιάζονται και αναλύονται τα βιοσήματα που εμφανίζονται κατά την διάρκεια του ύπνου. Στη συνέχεια γίνεται αναλυτική περιγραφή του ύπνου μέσω των σταδίων από τα οποία αποτελείται. Στο τρίτο κεφάλαιο, αναλύετε η έννοια της επιληψίας, δίνεται έμφαση στο συσχετισμό της με τον ύπνο και τα εγκεφαλικά σήματα. Συγκεκριμένα περιγράφεται πως η καταγραφή των εγκεφαλικών σημάτων μπορεί να βοηθήσει στην διάγνωση επιληπτικών κρίσεων. Εν συνέχεια στο κεφάλαιο 4 γίνεται περιγραφή του συστήματος, των φίλτρων που θα χρησιμοποιηθούν για την επεξεργασία των σημάτων και αποτυπώνονται οι απαιτήσεις του συστήματος. Τα επόμενα δύο κεφάλαια είναι ιδιαίτερα σημαντικά για την εργασία καθώς παρουσιάζεται η σχεδίασή και η υλοποίηση του συστήματος. Στο τελευταίο τμήμα της εργασίας, παρουσιάζεται το κριτικό μέρος του υπό μελέτη συστήματος και τέλος γίνεται η διεξαγωγή των συμπερασμάτων. Να επισημανθεί ότι στο παράθεμα Α βρίσκεται ο κώδικάς υλοποίησης του συστήματος

2. Βιοσήματα και ύπνος

Ο εγκέφαλος είναι ένα περίπλοκο σύστημα νευρώνων και νευρωνικών κυττάρων, το οποίο χωρίζεται σε τμήματα. Όλα αυτά τα τμήματα παράγουν EEG κυματομορφές οι οποίες καταγράφονται από εγκεφαλογράφους και η διαδικασία καταγραφής των κυματομορφών αυτών, εγκεφαλογράφημα. Οι κυματομορφές αυτές είναι ιδιαιτέρα σημαντικές καθώς περιέχουν χρήσιμες πληροφορίες όπως ενδείξεις για την κατάσταση της υγείας ενός ανθρώπου.[1]

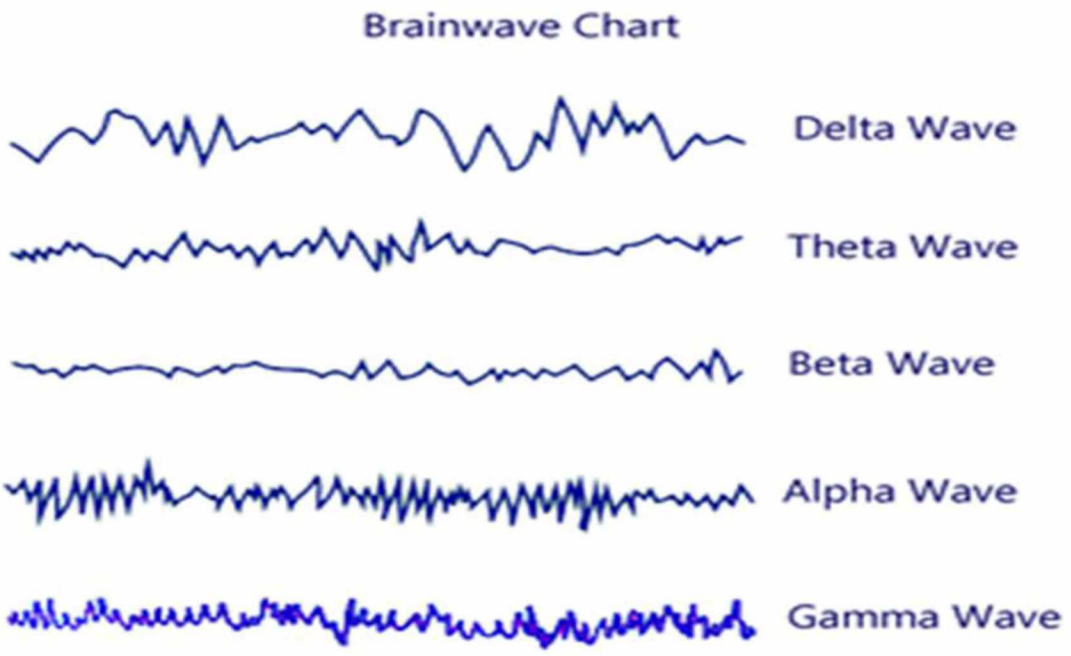
Ως βιοσήμα ορίζεται οποιοδήποτε σήμα που περιγράφει κάποιο φυσιολογικό φαινόμενο του οργανισμού και μπορεί να παρατηρηθεί και να μετρηθεί οποιαδήποτε στιγμή. Τα σήματα αυτά σχετίζονται με την αλλαγή του ρεύματος, η οποία παράγεται από την συνολική διαφορά δυναμικού σε ένα συγκεκριμένο ιστό, οργανικό ή κυτταρικό σύστημα. Έτσι, μεταξύ των πολλών αυτών σημάτων, τα πιο γνωστά είναι: EEG (electroencephalography), EOG (electrooculography) και EMG (electromyography).[2] [3]

2.1 EEG και ύπνος

2.1.1 EEG σήμα

Το ηλεκτροεγκεφαλογράφημα είναι μια καταγραφή ηλεκτρικής δραστηριότητας στην επιφάνεια του κρανίου. Το EEG μετρά διακυμάνσεις τάσης που προκύπτουν από ροές ιοντικού ρεύματος εντός των νευρώνων του εγκεφάλου. Σε κλινικά πλαίσια, το ΗΕΓ αναφέρεται στην καταγραφή του αυθόρμητων ηλεκτρικών δραστηριοτήτων του εγκεφάλου σε σύντομο χρονικό διάστημα, καταγραφόμενα από πολλά ηλεκτρόδια τοποθετημένα στο τριχωτό της κεφαλής. Οι διαγνωστικές εφαρμογές επικεντρώνονται γενικά στο φασματικό περιεχόμενο των EEG, δηλαδή, το είδος των νευρωνικών ταλαντώσεων που μπορούν να παρατηρηθούν στα σήματα EEG. Στη νευρολογία, η κύρια διαγνωστική εφαρμογή του EEG είναι στην περίπτωση της επιληψίας, καθώς η επιληπτική δραστηριότητα μπορεί να δημιουργήσει σαφείς ανωμαλίες σε μια τυποποιημένη μελέτη EEG. Μια δευτερεύουσα η κλινική χρήση του EEG είναι στη διάγνωση κώματος, εγκεφαλοπαθειών και εγκεφαλικού θανάτου.

Αναλυτικότερα τα ηλεκτροεγκεφαλικά σήματα είναι στοχαστικά, θορυβώδη και αρκετά ασθενή. Έχουν χαμηλό πλάτος το οποίο κυμαίνεται περίπου από 1μV έως 100μV. Το φάσμα των ηλεκτροεγκεφαλικών σημάτων εντοπίζεται στο 0 έως 100Hz, ενώ διακρίνονται σε συγκεκριμένες κυματομορφές βάσει των συχνοτήτων των αρμονικών από τις οποίες αποτελούνται, δηλαδή βάσει του φασματικού περιεχομένου τους. Με βάση αυτόν τον τρόπο, χωρίζονται σε πέντε θεμελιώδεις κατηγορίες: δ, θ, α, β και γ.[1],[2],[4]



Εικόνα 1: Εγκεφαλικοί ρυθμοί

2.1.2 Ύπνος

Ο ύπνος είναι μια κύρια, κυκλική, φυσιολογική κατάσταση που ορίζεται από σημαντική μείωση της συνείδησης, μείωση στην κίνηση των μύων και μια γενετική επιβράδυνση του μεταβολισμού. Είναι σημαντικός για την ποιότητα της ζωής, την καλή υγεία, προστατεύοντας την νοητική και φυσική υγεία του ανθρώπου. Οι ανάλυση του ύπνου μπορεί να γίνει μελετώντας τα ηλεκτροεγκεφαλικά σήματα (EEG), τα οποία χωρίζονται σε εποχές. [5] Ο ύπνος έχει δύο ξεχωριστές και διακριτές καταστάσεις με βάση τα συμπεριφορικά και φυσιολογικά χαρακτηρίστηκα του ανθρώπου. Οι καταστάσεις αυτές είναι: μη ταχεία κίνηση των ματιών (NREM) και ταχεία κίνηση των ματιών (REM). Ο NREM επιπλέον χωρίζεται σε 4 στάδια, το στάδιο I,II,III και IV.

2.1.3 Στάδια του ύπνου

1. **Στάδιο ξύπνιου**

Το στάδιο αυτό χαρακτηρίζεται από χαμηλό εύρος και αναμιγμένες συχνότητες EEG σημάτων. Το στάδιο είναι εμφανές στην αρχή του ύπνου και μπορεί να χαρακτηριστεί ως μεταβατικό από την πλήρη επαγρύπνηση σε σχεδόν κατάσταση ύπνου. Το στάδιο αποτελείται από α και β κύματα, μαζί με κίνηση ματιών και υψηλό μυϊκό τόνο.

2. NREM στάδιο 1

Στο στάδιο 1, το EEG σήμα έχει το υψηλότερο εύρος περίπου 50-70μV με συχνότητα που κυμαίνεται σε 2-7 Hz. Είναι μεταβατικό μεταξύ του ξύπνιου και του ύπνου και διαρκεί 1 με 5 λεπτά. παρατηρείται χαμηλή τάση EEG μαζί με α και θ δραστηριότητα, περιστασιακές αιχμές κορυφής και αργή κίνηση των ματιών. Το N1 στάδιο αποτελεί το 2% με 5% του συνολικού ύπνου με απουσία των ατράκτων ύπνου.

3. NREM στάδιο 2

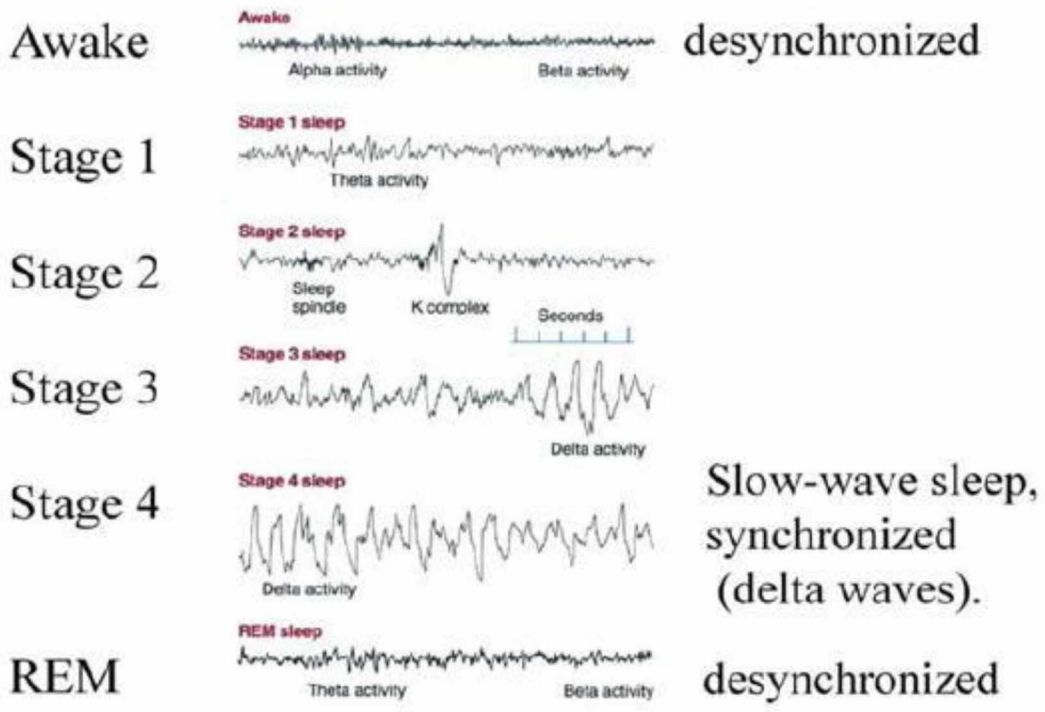
Το στάδιο N1 ακολουθείται από το στάδιο 2 το οποίο καλείται επίσης και ως στάδιο βαθύτερου ύπνου και διαρκεί 10 με 20 λεπτά. Σε αυτό το στάδιο εμφανίζονται οι κυματομορφές του ύπνου και K-συμπλέγματα. Το K-σύμπλεγμα (σύμπλοκο) είναι μεταβατικές κυματομορφές με αρνητική εκτροπή ακολουθούμενο από ένα θετικό κύμα. Είναι μεγαλύτερο από 75μV εύρος και χαμηλή τάση μικτής συχνότητας EEG. Εναλλακτικά, τα κύματα δ υψηλής τάσης μπορεί να περιλαμβάνουν έως 205 των εποχών του $2^{ου}$ σταδίου.

4. NREM στάδιο 3

Το στάδιο αυτό αναφέρετε στην περίοδο του ύπνου όπου αποτελείται από EEG κύματα με συχνότητα τον 2Hz η μικρότερη και εύρος όχι μεγαλύτερο των 75mV (δ κύματα). Το στάδιο 4 είναι παρόμοιο με το στάδιο 3 με μόνη διαφορά ότι σε αυτό το στάδιο εμφανίζονται περισσότερα δ κύματα

5. REM στάδιο

Το στάδιο αυτό χαρακτηρίζεται από ταχεία κίνηση των ματιών κάτω από τα κλειστά βλέφαρα και αποτελείται από χαμηλής τάσης μικτά EEG κύματα και χαμηλό εύρος.[6]



Εικόνα 2: Κυματομορφές σταδίων ύπνου

.

Κατηγορία	**Συχνότητα**	**Εύρος**	**Στάδια**
Άλφα	8-13 Hz	20-60 μV	Στάδιο ξύπνιου, στάδιο I και REM
Βήτα	13+ Hz	2-20 μV	ξύπνιος
Θήτα	4-8 Hz	50-75 μV	Στάδιο I,II,III και IV
Δέλτα	0-4 Hz	75+ μV	Στάδιο III και IV
Άτρακτοι ύπνου	12-14 Hz		Στάδιο II
Κ σύμπλεγμα	0,5-1 Hz		Στάδιο II

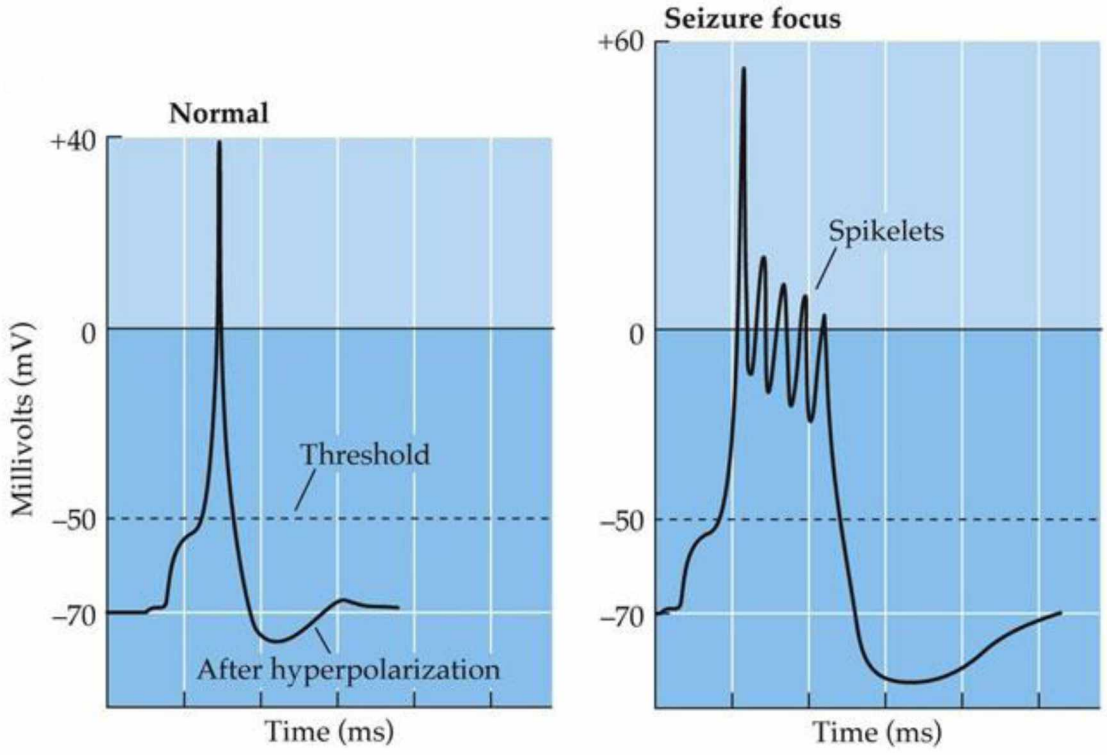
Πίνακας 1: στάδια του ύπνου και εγκεφαλικά σήματα

3. Επιληψία και ύπνος

3.1 Επιληψία

Ο ορός «ΕΠΙΛΗΨΙΑ» εμφανίστηκε στη γαλλική γλώσσα το 1503. Σχηματίστηκε από την λατινική λέξη epilepsia, η οποία προέρχεται από το ελληνικό ρήμα επιλαμβάνειν (καταλαμβάνω, προσβάλλω αιφνίδια).

Βασική προϋπόθεση της επιληψίας αποτελεί μια ιδιοπαθής ή επίκτητη διαταραχή της νευρωνικής διεγερσιμότητας. Κυρίο σημείο των επιληπτικών νευρώνων είναι : η υπερδιεγερσιμότητα, η οποία ορίζεται ως η τάση ενός νευρώνα να δίνει γένεση σε επαναλαμβανόμενες εκ φορτίσεις σαν απάντηση σε ένα ερέθισμα, το οποίο προκαλεί ένα δυναμικό ενέργειας και ο υπερσυγχρονισμός, ο οποίος ορίζεται ως η ιδιότητα μιας ομάδας νευρώνων να παράγουν συγχρόνως σειρές δυναμικών.[7]



Εικόνα 3: Νευρώνας με ένα δυναμικό ενέργειας και "επιληπτικός" υπερευρέθιστος νευρώνας

3.2 Οι επιδράσεις του ύπνου στις επιληπτικές κρίσεις

Ο ύπνος και η επιληψία είναι αλληλένδετα. Η κατανόηση της σχέσης μεταξύ ύπνου και επιληπτικών κρίσεων είναι σημαντική για την βελτιστοποίηση της διαχείρισης του επιληπτικού ατόμου σε διάφορους τομείς. Αρχικά ο ύπνος ρυθμίζει την έκφραση των επιληπτικών κρίσεων και των εσωτερικών επιληπτικών εκκρίσεων. Καθώς επίσης η θεραπεία διαταραχών του ύπνου και των επιληπτικών κρίσεων είναι αλληλοεπηρεαζόμενη, η μια μπορεί να συμβάλει στην βελτιστοποίηση της άλλης.

Οι επιληπτικές κρίσεις επηρεάζονται από τα διάφορα στάδια του ύπνου. Το στάδιο NREM είναι γνωστό και ως κατάσταση σχετικού "συγχρονισμού νευρώνων" και χαρακτηρίζεται από συγχρονισμένη συνοπτική δραστηριότητα, στην οποία για να εμφανιστεί μια επιληπτική κρίση χρειάζεται η παρουσία μιας κριτικής μάζας νευρώνων. Κατά τον NREM ύπνο και συγκεκριμένα στα στάδια III και IV χαλινεργικά και μονοαμινεργικά κομμάτια του εγκεφάλου μειώνουν την παραγωγή προοδευτικών υπερπολώσεων στους θάλαμο φλοιώδεις νευρώνες ρελέ . Ενώ στο REM στάδιο η ενεργοποίηση αυτή εμφανίζεται όταν χαλινουργικοί προσαγωγοί του εγκεφαλικού στελέχους αυξάνουν τους ρυθμούς πυροδότησης προκαλώντας απόπλυση των νευρώνων. Ωστόσο επιληπτικές κρίσεις συναντάμε κυρίως κατά το NREM ύπνο.[7]

3.3 ΗΕΓ ύπνου

Η πιο στοιχειώδης έκφραση της επιληπτικής εκ φόρτισης στο ηλεκτροεγκεφαλογράφημα επιφάνειας είναι το σύμπλεγμα αιχμή-κύμα. Η αιχμή προέρχεται από ένα άθροισμα νευρωνικών παροξυσμών εκ πολώσεων (PDS, paroxysmal depolarization shift), οι οποίες δίνουν σε αυλούς δυναμικών ενεργείας. Το βραδύ κύμα μπορεί να θεωρηθεί ότι αποτελεί την ηλεκτροφυσιολογική έκφραση ενός προστατευτικού φαινομένου περιορισμού της διάχυσης διεγερτικών εκ φορτίσεων.

Είναι αποδεκτό από πολύ καιρό τώρα ότι στην επιληπτογόνο ζώνη, μια περιορισμένη περιοχή του εγκεφαλικού φλοιού, δημιουργούνται οι εστιακές κρίσεις. Από την ενεργοποίηση αυτής της ζώνης εξαρτώνται τα ήλεκτρο-κλινικά χαρακτηριστικά των κρίσεων. Για αυτό το λόγο η καταγραφή των εγκεφαλικών δυναμικών που συναθροίζονται στην επιφάνεια του κρανίου ή ηλεκτροεγκεφαλογράφημα (ΗΕΓ) είναι μια αναντικατάστατη εξέταση για την διάγνωση των επιληπτικών κρίσεων.

Η καταγραφή της εγκεφαλικής ηλεκτρικής δραστηριότης γίνεται μέσω ηλεκτροδίων επιφάνειας, τα οποία τοποθετούνται στο κρανίο με τυποποιημένο τρόπο. Η διάρκεια ενός απλού ΗΕΓ είναι 20 λεπτά και πρέπει να καταγράψει τρεις έως τέσσερίς διαφορετικές διατάξεις. Η καταγραφή απαιτεί την συστηματική προσφυγή σε πολυγραφική μελέτη για να προσδιοριστούν καλύτερα τα διάφορα στάδια του ύπνου.

Δώδεκα με δεκαέξι ΗΕΓ επαγωγές συνδυάζονται με καταγραφή των αναπνευστικών κινήσεων. Δύο τρόποι είναι δυνατοί: καταγραφή ολόκληρη τη νύχτα και καταγραφή του μεσημεριανού ύπνου με ή χωρίς προηγούμενη στέρησή του ύπνου.

Τα εγκεφαλικά κύματα ή ΗΕΓ δραστηριότητες χαρακτηρίζονται από τη συχνότητα τους, το εύρος τους, τη μορφολογία τους, τη σταθερότητα τους, την εντόπιση τους και την αντιδραστικότητα τους.

Με βάση την συχνότητα τους κατατάσσονται σε:

- Δέλτα δραστηριότητα: συχνότητά μικρότερη των 3,5 Hz
- θήτα δραστηριότητα: συχνότητά ανάμεσα στα 4 και 7,5 Hz
- άλφα δραστηριότητα: συχνότητά ανάμεσα στα 8 και 13 Hz
- βήτα δραστηριότητα: συχνότητά μεγαλύτερη των 13 Hz

σε έναν υγιή και σε εγρήγορση ενήλικα, που βρίσκεται σε φυσική και πνευματική χαλάρωση καταγράφονται φυσιολογικά δύο ρυθμοί, ο ρυθμός άλφα και ο ρυθμός βήτα. Ο ρυθμός άλφα αποτελείται από μια σειρά ημιτονοειδών κυμάτων των οποίων η συχνότητα βρίσκεται ανάμεσα στα 8 και 13 Hz και το εύρος ανάμεσα στα 20 και 100 microvolts, κατανέμεται στις οπίσθιες περιοχές του κρανίου με τρόπο αμφοτερόπλευρο και σύγχρονο. Το εύρος του είναι μέγιστο όταν τα μάτια είναι κλειστά και αποκλείεται προσωρινά με άνοιγμα ματιών. Ο ρυθμός βήτα έχει μια συχνότητα περίπου 16 με 18 Hz και καταλαμβάνει τις μέσες περιοχές των δύο ημισφαίριων με ασύγχρονο τρόπο

Ένας ορισμένος αριθμός ΗΕΓ δραστηριοτήτων απομακρύνεται από τον κανόνα και αποτελούν ασυνήθιστες φυσιολογικές ΗΕΓ δραστηριότητες. Ορισμένες είναι συχνές και εύκολες να αναγνωριστούν. Άλλες, πιο σπάνιες, πρέπει να είναι καλά γνωστές . Λόγω της μορφολογίας τους ή του ρυθμού τους μπορούν πραγματικά να θυμίσουν τις ΗΕΓ ανωμαλίες που απαντώνται στους επιληπτικούς ανθρώπους.

Αυτές οι στοιχειώδεις παροξυσμικές ανωμαλίες οργανώνονται ρυθμικά σύμφωνα με διάφορους τρόπους στη διάρκεια μιας επιληπτικής κρίσης στα μεσοδιαστήματα των κρίσεων η παρουσία τους στο ΗΕΓ επιφάνειας αποτελούν μια σημαντική βοήθεια στη διάγνωση της επιληψίας .

- η βραδεία αιχμή ή αιχμηρόμορφο κύμα είναι ένα κύμα του οποίου η διάρκεια βρίσκεται ανάμεσα στα 70 και 200 ms και εξαιτίας αυτού παίρνει μια μορφή λιγότερο οξεία. Εάν η απότομη κλίση καταλαμβάνει το πρόσθιο μέρος του σχήματος καλείται κύμα με οξεία κορυφή
- το σύμπλεγμά αιχμή-κύμα αποτελείται από μια αιχμή που ακολουθείται αμέσως από ένα βραδύ κύμα. Εάν η συχνότητα του συμπλέγματος αιχμή-κύμα

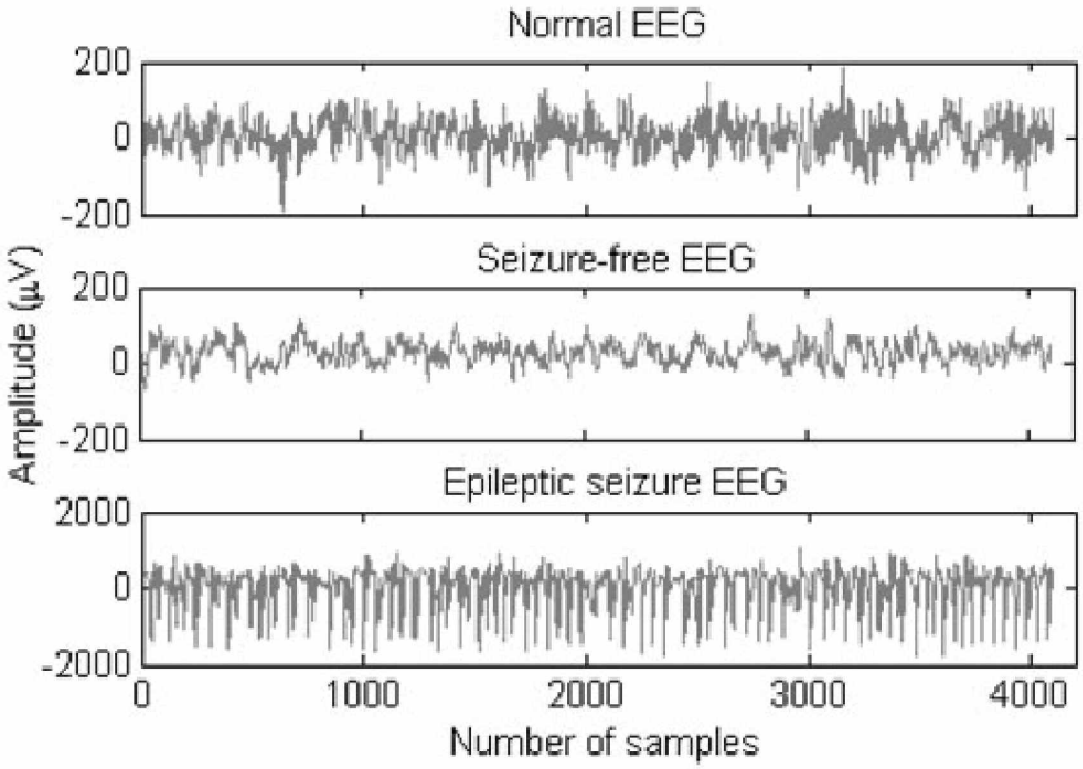
είναι μικρότερη από 2,5 Hz ονομάζεται βραδύ σύμπλεγμα αιχμή-κύμα, που η συστατική αιχμή του έχει συχνότητα μεγαλύτερη η μικρότερη των 70 ms.

- Η πολυαιχμή είναι μια ακολουθία πολλών αιχμών.
- Το σύμπλεγμα πολυαιχμή-κύμα είναι μια ακολουθία μίας πολυαιχμής και ενός η περισσότερων βραδέων κυμάτων.

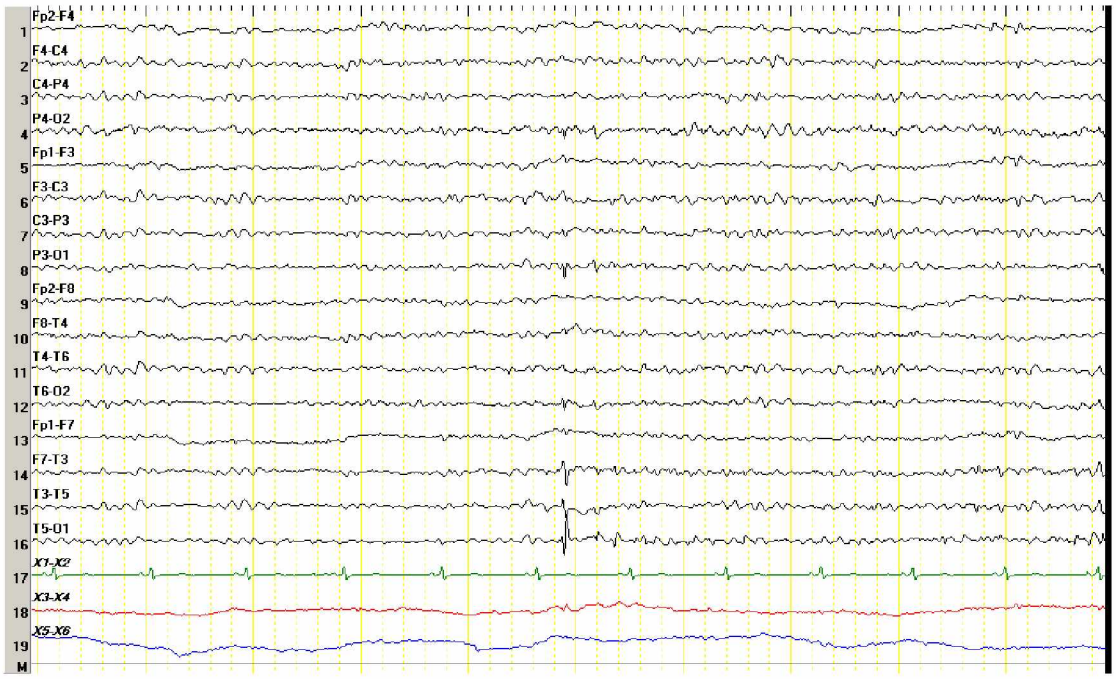
Η καταγραφή κατά την διάρκεια του ύπνου αποτελεί πολύτιμο συμπλήρωμα ,συχνά αναντικατάστατο, της καταγραφής σε εγρήγορση. Ο ελαφρύς βραδύς ύπνος είναι πράγματι ένας πολύ αποτελεσματικός ενεργοποιητής των μεσοκριτικών παροξυσμικών ανωμαλιών, ιδιαίτερα στις ιδιοπαθείς επιληψίες των παιδιών και των εφήβων.

Συστηματικό για ορισμένους, το ΗΕΓ ύπνου συνιστάται ιδιαίτερα εάν το ΗΕΓ εγρήγορσης δε δείχνει ανωμαλίες, εάν αυτές δεν είναι καλά προσδιορισμένες ή εάν οι κρίσεις συμβαίνουν εκλεκτικά κατά την υπνηλία, στη διάρκεια του ύπνου ή στην αφύπνιση. Σε ορισμένες περιπτώσεις, μόνο το ΗΕΓ ύπνου επιτρέπει τη διάγνωση του υπεύθυνου επιληπτικού συνδρόμου. Το ΗΕΓ ύπνου είναι απαραίτητο για την ακριβή διάγνωση ορισμένων επιληπτικών συνδρόμων, όπως το σύνδρομο Lennox-Gastaut.

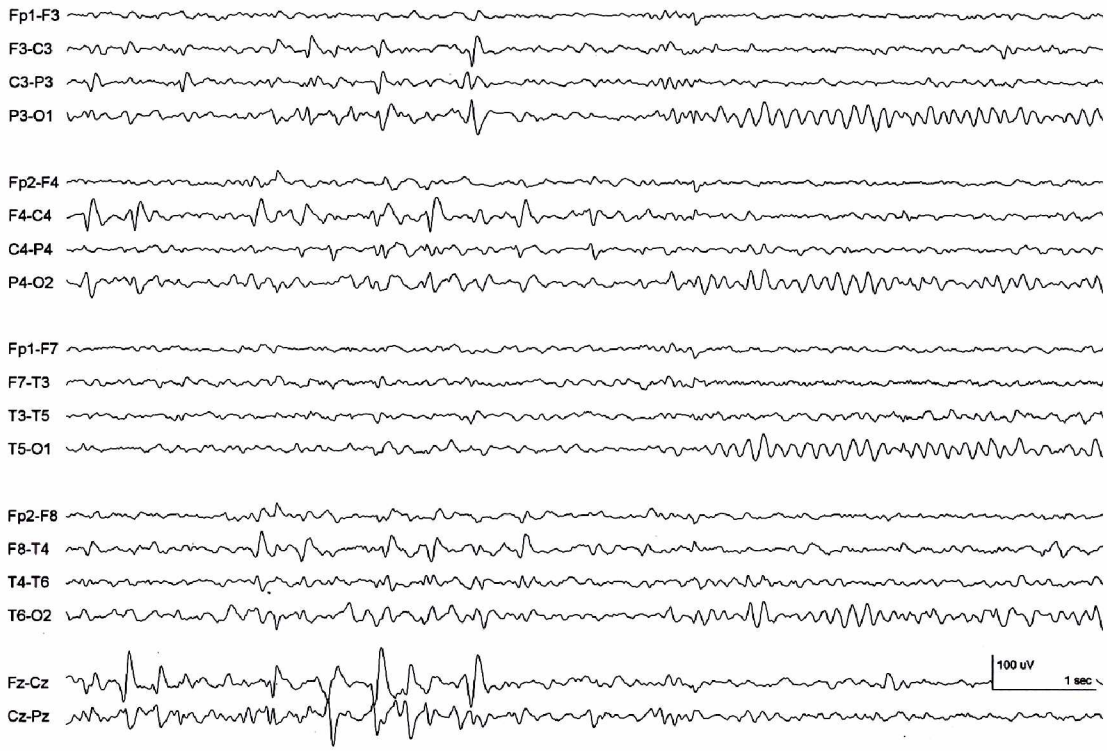
Εκτός από την καταγραφή ορισμένων λειτουργιών κατά την διάρκεια του ύπνου, η πολυγραφική καταγραφή επιτρέπει την ανάλυση της σχέσης ανάμεσα στις δραστηριότητες που καταγράφει το ΗΕΓ. [7]



Εικόνα 4: :φυσιολογική, μη επιληπτική και επιληπτική κυματομορφή



Εικόνα 5: καλοήθη επιληπτόμορφα μεταβατικά ευρήματα ύπνου



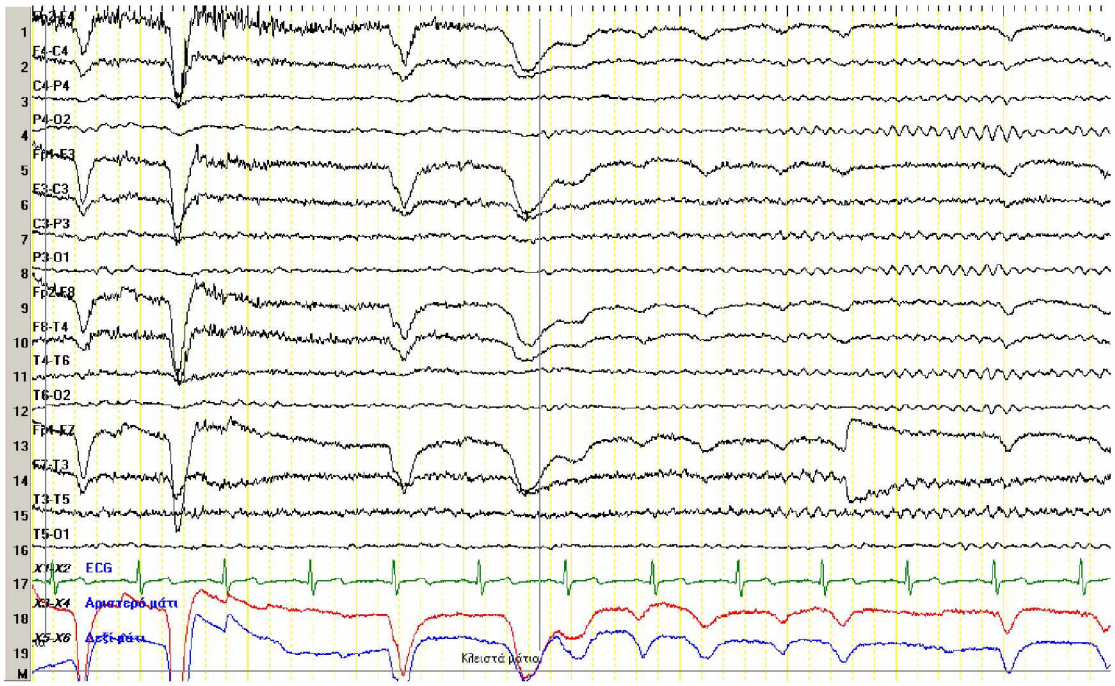
Εικόνα 6: Επιληψία- υπνηλία αιχμηρά κύματα στις κεντρικές περιοχές (C3 και C4) με επικράτηση δεξιά.

4. Περιγραφή συστήματος

Το υπό διαπραγμάτευση θέμα της παρούσας εργασίας αφορά την ανάπτυξη ενός ψηφιακού συστήματος, για την επεξεργασία βιοσημάτων του εγκεφάλου, αξιοποιώντας ολοκληρωμένα αναδιατασσόμενα κυκλώματα (Reconfigurable Integrated Systems). Συγκεκριμένα το σύστημα θα υλοποιηθεί στην πλακέτα ανάπτυξης PYNQ z2 και θα δέχεται ως είσοδο βιοσήματα προερχόμενα από τον εγκέφαλο, με σκοπό την επεξεργασία τους και κατόπιν εφαρμογή τους σε Βιοχαρτικούς αλγορίθμους. Συγκεκριμένα χρησιμοποιούνται 3 είδη φίλτρων : βαθυπερατό φίλτρο (Low pass filter), Υψιπερατό φίλτρο (High pass filter) και φίλτρο κινούμενου μέσου όρου (Moving average filter). Με την βοήθεια αυτών των φίλτρων το αδιαμόρφωτο ψηφιακό σήμα μετατρέπεται σε κατάλληλη μορφή απαλλαγμένο από θόρυβο, ενισχυμένο κατάλληλα και διορθωμένο. Καθώς επίσης το βιοσήμα αναλύεται σε συχνότητα πεδίου και αναδεικνύεται η ισχύ του ώστε να είναι ευκολότερος ο εντοπισμός των ατραπών. Η προ επεξεργασία των εγκεφαλικών σημάτων (EEG) είναι ιδιαίτερα σημαντική καθώς την καταγραφή των ηλεκτοεγκεφαλικών σημάτων μπορεί να προκύψουν διάφορα παράσιτα όπως:

1. **Φυσιολογικά**

- **Δυναμικά από μάτια**
  Η διαφορά δυναμικού μεταξύ του πρόσθιου και του οπίσθιου τμήματος του ματιού είναι αρκετά mV. Το δίπολο αυτό δίνει ρεύμα δια των ιστών και στα δυναμικά στο δέρμα του κεφαλιού που αλλάζουν όταν κινούνται τα μάτια.



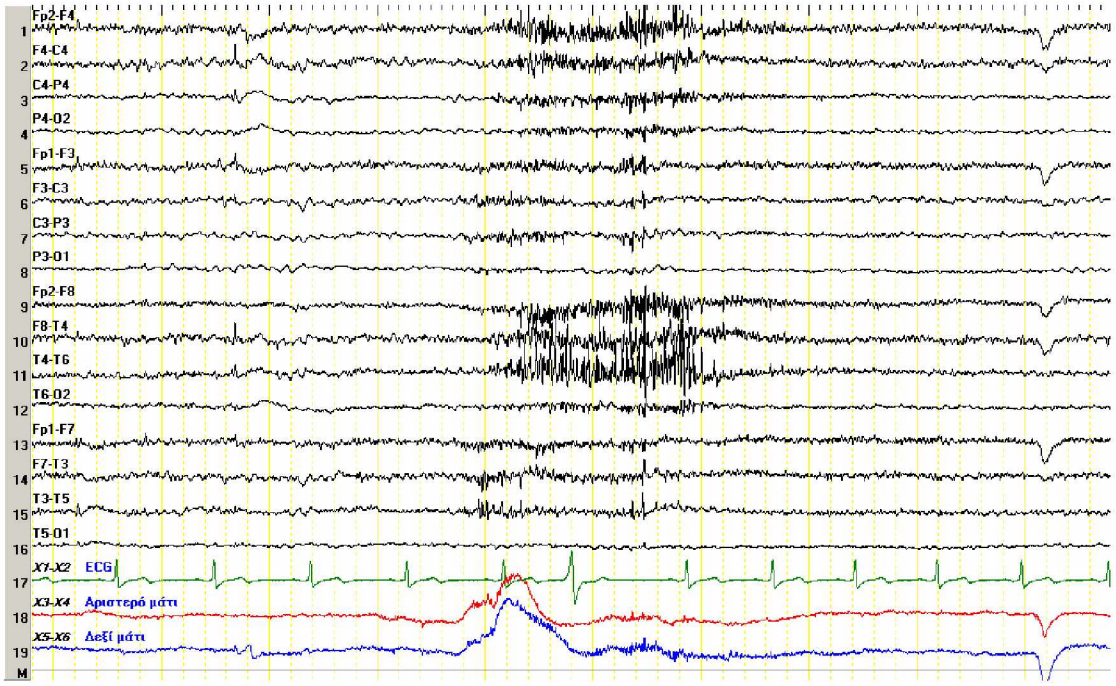
Εικόνα 7:οφθαλμικές κινήσεις- παράσιτα

- **Δυναμικά από καρδία**

  Η απόσταση μεταξύ καρδίας και εγκεφάλου είναι μεγάλη, έτσι το σήμα που παίρνουν τα ηλεκτρόδια είναι πολύ μικρό. Όταν είναι απαραίτητο το καρδιακό σήμα δίνει στο EEG οξείες αιχμές με την περιοδικότητα της καρδίας (~0,8s). Η παρεμβολή που προκαλείται στο EEG όταν υπάρχει εμφυτευμένος βηματοδότης της καρδίας είναι ισχυρότατη και δεν μπορεί να επαλειφθεί.

Εικόνα 8: ΗΚΓ-παράσιτα

- Γλωσσοκινητικό παράσιτο, γαλβανικό-δέρματος και από φυσιολογικές κινήσεις. Τα παράσιτα που οφείλονται σε δραστηριότητα διαφορετικών μυών, όπως στους μύες του προσώπου ή του λαιμού αποτελούνται από ένα μεγάλο εύρος συχνοτήτων και είναι δυνατόν να εμφανίζονται σε πολλά ηλεκτρόδια ανάλογα με τη θέση των μυών



Εικόνα 9: μυϊκό παράσιτο

2. Μη φυσιολογικά

- **Παράσιτα ηλεκτροδίων**

Η κίνηση είτε του εξεταζόμενου είτε των καλωδίων που συνδέουν τα ηλεκτρόδια με το μηχάνημα μπορεί να προκαλέσει διαταραχή της ηλεκτροχημικής ισορροπίας μεταξύ ηλεκτροδίων και δέρματος. Με αποτέλεσμα να προκληθούν αλλαγές δυναμικού που θα αποτελούν σήματα για το EEG. Αυτά ενισχύονται μαζί με τα ανεπιθύμητα σήματα επηρεάζοντας την γραμμή βάσεως του EEG.

Εικόνα 10: παράσιτα από ηλεκτρόδια

- **Μηχανικά, ψηφιακά παράσιτα και παράσιτα από το περιβάλλον** [8]

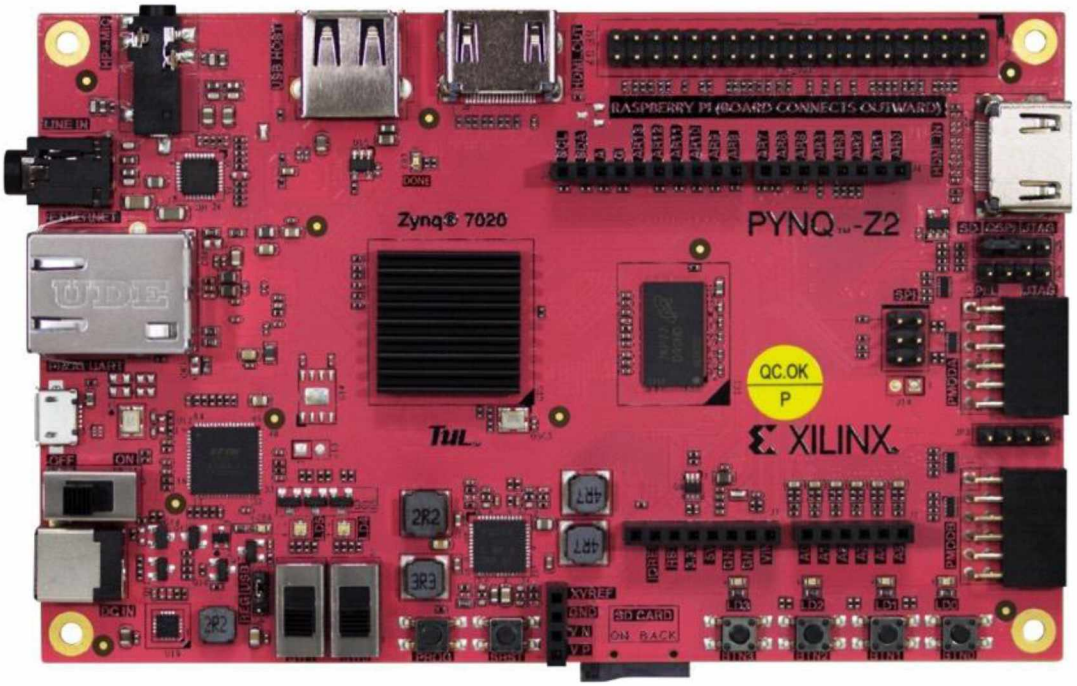
PYNQ z2

Το PYNQ-z2 είναι μια πλατφόρμα ανάπτυξης για την υποστήριξη του πλαισίου PYNQ και της ανάπτυξης ενσωματωμένου συστήματος. Το PYNQ είναι ένα έργο ανοικτού κώδικα από την Xilinx, με το οποίο οι χρήστες μπορούν να εκμεταλλευτούν τα πλεονεκτήματα της προγραμματιζόμενης λογικής και μικροεπεξεργαστών των Zynq SoCs. Τα κυκλώματα προγραμματιζόμενης λογικής παρουσιάζονται ως βιβλιοθήκες υλικού και ονομάζονται overlays. Έτσι μπορούν να επιλεχθούν τα overlay που ταιριάζουν καλύτερα με την εφαρμογή και να υπάρχει πρόσβαση σε αυτό ευκολότερα μέσο ενός Application Programming Interface (API). Με PYNQ οι χρήστες μπορούν να δημιουργήσουν υψηλής απόδοσης ενσωματωμένα συστήματα με:

- Παράλληλη εκτέλεση hardware
- Αλγόριθμους επιτάχυνσης hardware
- Επεξεργασία σημάτων σε πραγματικό χρόνο
- Υψηλού εύρους ζώνης IO
- Έλεγχος χαμηλής λανθάνουσας κατάστασης

Η επιλογή της συγκεκριμένης πλακέτας έγινε βάση των δυνατοτήτων για ευρύτερη χρήση του συστήματος σχεδίασης. Η χρήση της συγκεκριμένης πλακέτας στην

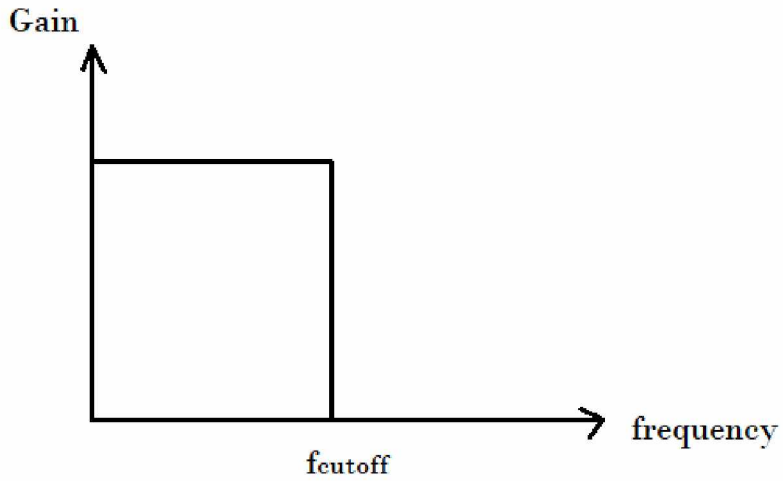
παρούσα πτυχιακή εργασία είναι ιδιαίτερα βοηθητική στην επεξεργασία των σημάτων καθώς δίνει δυνατότητες παρακολούθησης σε συνθήκες πραγματικού χρόνου. [9]



Εικόνα 11: πλακέτα PYNQ-z2

4.1.1 Βαθυπερατό φίλτρο (Low pass filter)

Από ένα βαθυπερατό φίλτρο επιτρέπονται να περάσουν χαμηλές συχνότητες του σήματος στο οποίο εφαρμόζεται, ενώ απορρίπτονται οι υψηλές. Χαρακτηρίζεται από την συχνότητα αποκοπής (cutoff frequency), η οποία καθορίζει ποιες συχνότητες θα περάσουν.

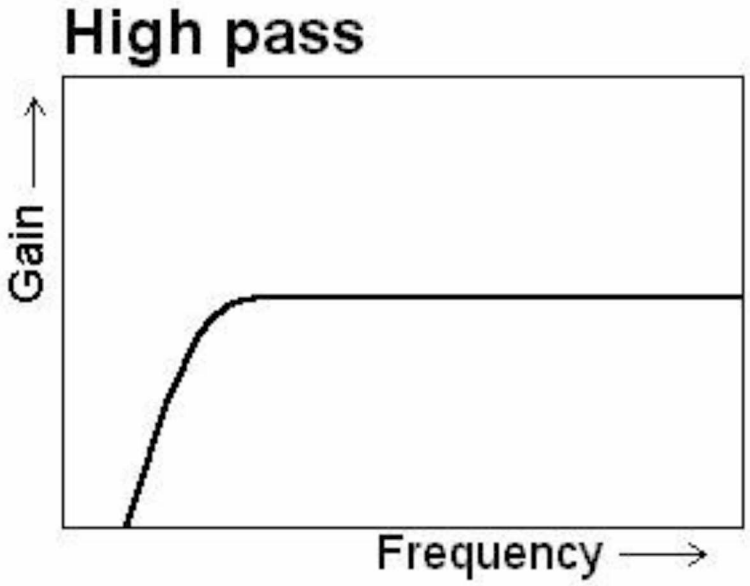


Εικόνα 12: Ααπλουστευμένο σχήμα ιδανικού βαθυπερατού φίλτρου

Η εφαρμογή του βαθυπερατού φίλτρου σε ένα σήμα αποτρέπει την εμφάνιση φαινομένων επικάλυψης. Να σημειωθεί ότι σε μια καταγραφή εγκεφαλικών σημάτων είναι έντονες η παρεμβολές θορύβων. Ένα βαθυπερατό φίλτρο μπορεί να εξομαλύνει χαμηλές συχνότητες θορύβου.[10],[11],[12]

4.1.2 Υψιπερατό φίλτρο (High pass filter)

Ένα Υψιπερατό φίλτρο επιτρέπει να περάσουν συχνότητες υψηλότερες από μια ορισμένη συχνότητα αποκοπής και εξασθενεί συχνότητες χαμηλότερες από την συχνότητα αυτή. Οι τιμές που δεν επιτρέπονται να περάσουν εξαρτώνται από τον σχεδιασμό του φίλτρου.



Εικόνα 13: Σχήμα υψιπερατού φίλτρου

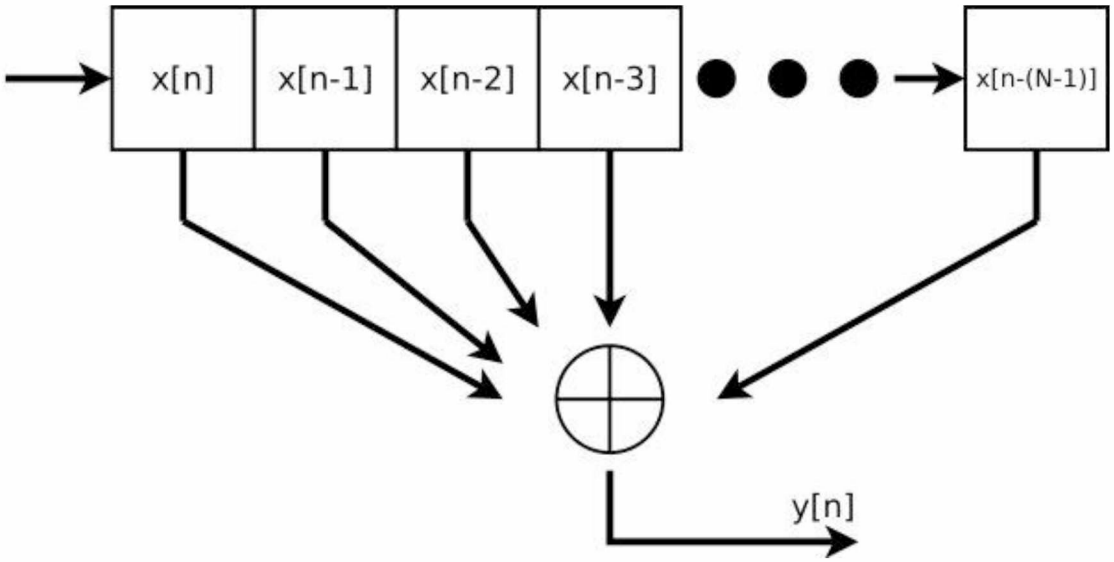
Καθώς θέλουμε το σήμα μας να είναι απαλλαγμένο από κάθε είδους θόρυβο, είτε αυτός βρίσκεται σε υψηλές είτε σε χαμηλές τιμές. Αυτό γίνεται με την βοήθεια των δύο παραπάνω σημάτων , διατηρώντας μόνο τις επιθυμητές τιμές που είναι χρήσιμές για την μελέτη μας. [13]

4.1.3 Φίλτρο κινούμενου μέσου όρου (Moving average filter)

Στην στατιστική, ο μέσος όρος είναι ένας υπολογισμός για την ανάλυση δεδομένων με τη δημιουργία μιας σειράς μέσων όρων διαφορετικών υποσυνόλων του πλήρους συνόλου δεδομένων. Λαμβάνοντας μια σειρά αριθμών και ένα σταθερό μέγεθος υποσυνόλου, το πρώτο στοιχείο του κινούμενου μέσου λαμβάνεται λαμβάνοντας τον μέσο όρο του αρχικού σταθερού υποσυνόλου των σειρών αριθμών. Στη συνέχεια, το υποσύνολο τροποποιείται με "μετατόπιση προς τα εμπρός". Δηλαδή, εξαιρώντας τον πρώτο αριθμό της σειράς και συμπεριλαμβάνοντας την επόμενη τιμή στο υποσύνολο. Ένας κινητός μέσος όρος χρησιμοποιείται συνήθως με δεδομένα χρονοσειρών για την εξομάλυνση των βραχυπρόθεσμων διακυμάνσεων και την επισήμανση μακροπρόθεσμων τάσεων ή κύκλων. Το όριο μεταξύ βραχυπρόθεσμης και μακροπρόθεσμης διάρκειας εξαρτάται από την εφαρμογή και οι παράμετροι του κινητού μέσου όρου θα καθοριστούν αναλόγως. Το φίλτρο κινούμενου μέσου όρου είναι ένα από τα κοινά ψηφιακά φίλτρα. Είναι ιδιαίτερα χρήσιμο για μείωση τυχαίου θορύβου διατηρώντας παράλληλα απότομη απόκριση βημάτων.

Εξίσωση του κινητού μέσου φίλτρου: $y[i] = \frac{1}{M} \sum_{j=0}^{M-1} x[i+j]$

Σε αυτή η εξίσωση, x [] είναι το σήμα εισόδου, y [] είναι το σήμα εξόδου και το M είναι ο αριθμός των σημεία που χρησιμοποιούνται στον κινητό μέσο όρο.



Εικόνα 14: σχήμα φίλτρου κινούμενου μέσου

Με την χρήση φίλτρου κινούμενου μέσου όρου καθώς αντιμετωπίζονται όλες οι τιμές εισόδου με την ιδία σημασία, επιτυγχάνεται η εξομάλυνση της θορυβώδους εισόδου.[14],[15],[16]

4.2 Εξαγωγή απαιτήσεων

Απαίτηση μπορεί να είναι από μια υψηλού επιπέδου αφηρημένη δήλωση μιας υπηρεσίας ή ενός περιορισμού του συστήματος μέχρι ένας λεπτομερής μαθηματικός ορισμός μιας λειτουργίας του συστήματος. Οι απαιτήσεις διακρίνονται σε λειτουργικές και μη λειτουργικές. Συγκεκριμένα οι λειτουργικές απαιτήσεις αφορούν δηλώσεις που ορίζουν ποιες υπηρεσίες θα πρέπει να παρέχει το σύστημα, πως πρέπει να αντιδράσει στις εισόδους και τι είδους συμπεριφορά θα πρέπει να έχει στις διάφορες καταστάσεις. Παράλληλα οι μη λειτουργικές απαιτήσεις είναι περιορισμοί στις υπηρεσίες και τις λειτουργίες που παρέχει το σύστημα, για παράδειγμα χρονικοί περιορισμοί, περιορισμοί στη διαδικασία ανάπτυξης κ.λπ. τέλος υπάρχουν οι απαιτήσεις πεδίου που αφορούν το πεδίο εφαρμογής του συστήματος.

Λειτουργικές απαιτήσεις

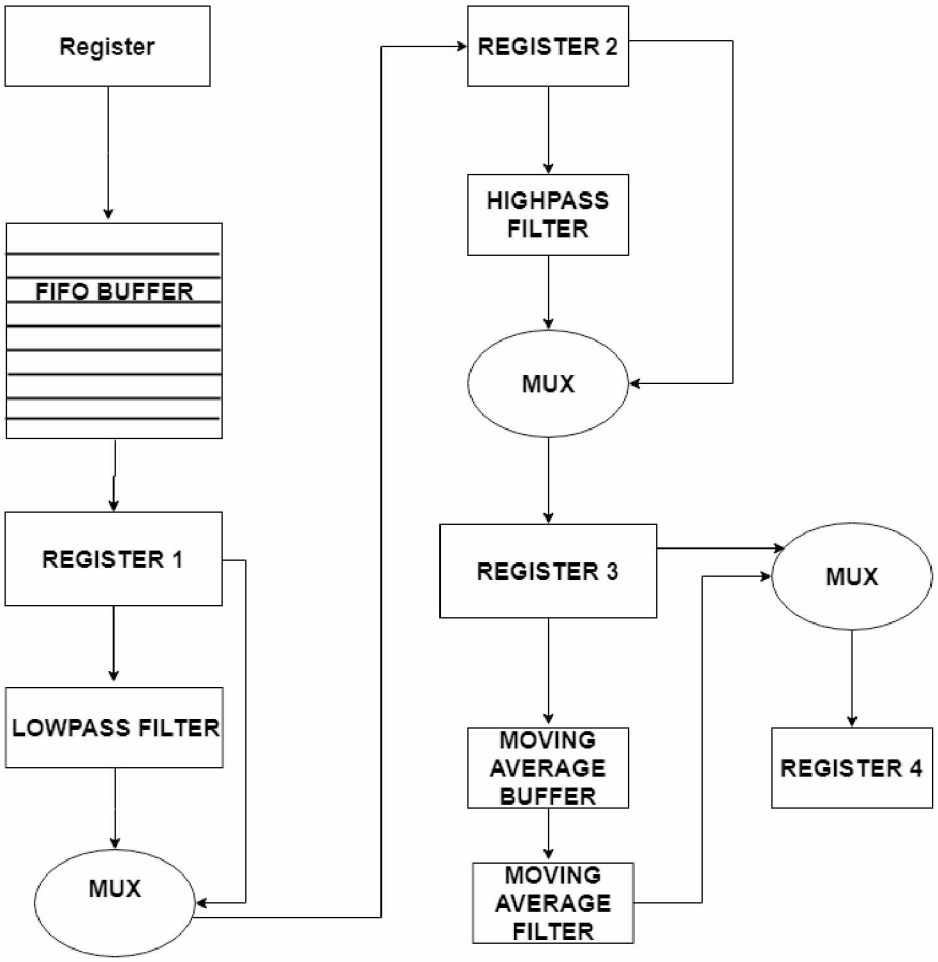
- Δυνατότητα επιλογής του χρήστη ενός, δύο ή τριών φίλτρων
- Έλεγχός ορθής λειτουργίας του υψιπερατού φίλτρου
- Έλεγχός ορθής λειτουργίας του βαθυπερατού φίλτρου
- Έλεγχος ορθής λειτουργίας φίλτρου κινούμενου μέσου όρου

Μη λειτουργικές απαιτήσεις

- Επιλογή αναλογικών φίλτρων
- Επιλογή φίλτρων επεξεργασίας που δεν αλλοιώνουν την χρήσιμη πληροφορία
- Φίλτρα για ευρεία χρήση βιοσημάτων
- Γρήγορος χρόνος απόκρισης συστήματος
- Δημιουργία συστήματος σε συμβατή γλώσσα προγραμματισμού (VHDL)
- Το σύστημα να είναι εύχρηστο

5. Σχεδίαση συστήματος

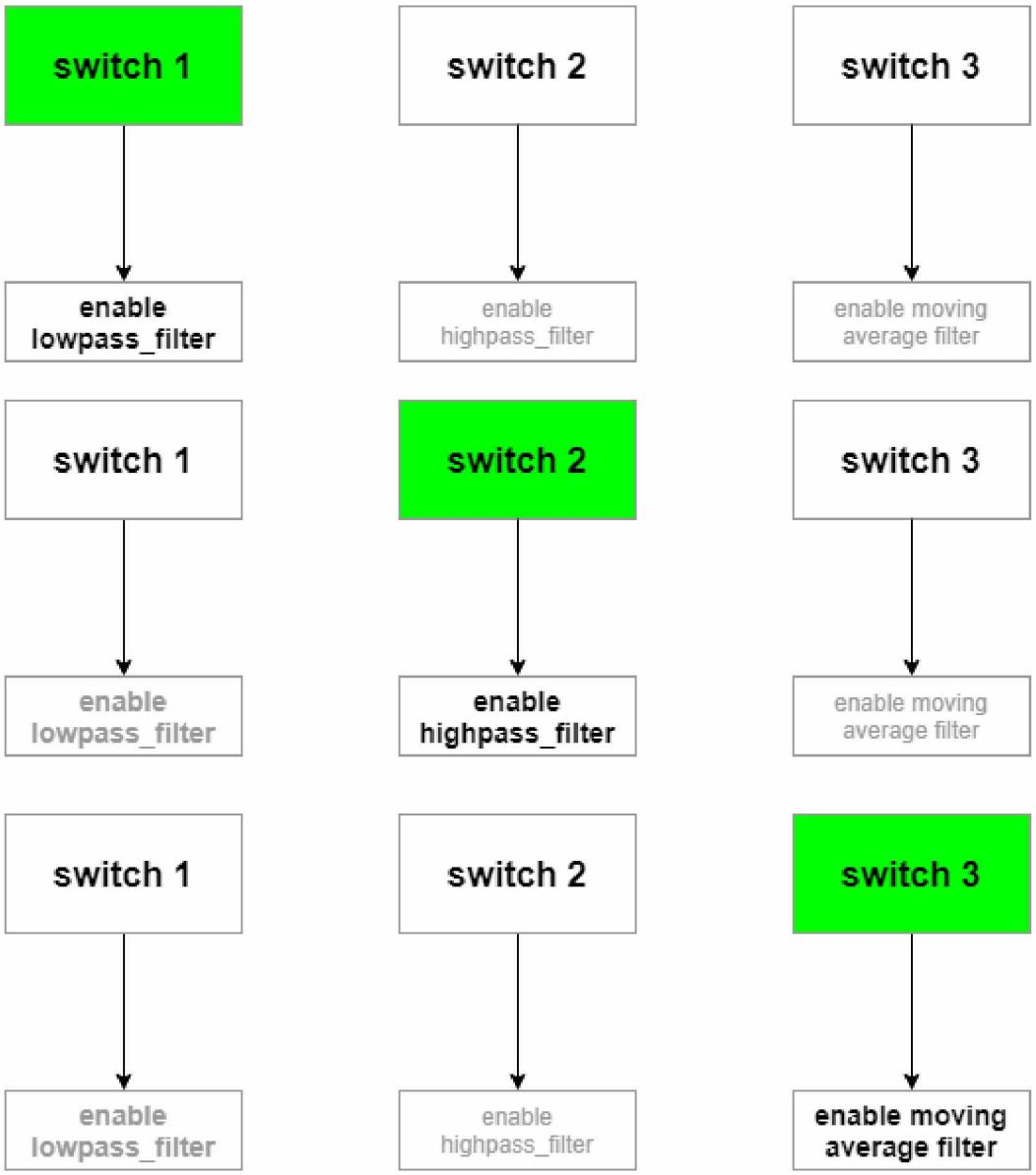
Η παρούσα σχεδίαση του συστήματος έχει γίνει με σκοπό την εύχρηστη εφαρμογή του σε ποίκιλα βιολογικά σήματα. Ο χρήστης μπορεί να προσαρμόζει το σύστημα ανάλογα με τις ανάγκες και τις απαιτήσεις του. Αναλυτικότερα αυτό θα αποτελείται από καταχωρητές (registers), προσωρινή μνήμη FIFO (buffer FIFO), καθώς και έναν απλό buffer, πολυπλέκτες (mux), ένα βαθυπερατό φίλτρο, ένα υψιπερατό φίλτρο και ένα φίλτρο κινούμενου μέσου όρου. Το σύστημα θα λειτουργεί με την εξής μορφή:



Εικόνα 15: Σχεδιάγραμμα τελικού συστήματος

Συγκεκριμένα οι τιμές που είναι αποθηκευμένες στον πρώτο καταχωρητή περνάνε σε έναν fifo buffer και από εκεί μία μία αποθηκεύονται σε έναν δεύτερο καταχωρητή μαζί με ένα bit εγκυρότητας που τοποθετείται στο τέλος και δείχνει ότι προήλθαν από τον fifo buffer. Στη συνέχεια αν έχει επιλεχθεί το πρώτο φίλτρο (low pass filter) ενεργοποιείται ένα enable_f1 το οποίο μαζί με τις φλιταρισμένες τιμές και τις αρχικές περνάνε από ένα πολυπλέκτη για να αποθηκευτούν πάλι σε έναν καταχωρητή. Η επιλογή στον πολυπλέκτη για το ποιες τιμές θα αποθηκευτούν στον καταχωρητή γίνεται με το enable του φίλτρου, αν είναι ενεργό αποθηκεύονται οι φιλτραρισμένες αλλιώς αποθηκεύονται οι αρχικές. Υπάρχουν αντίστοιχα enable_f2 και enable_f3 για το δεύτερο και τρίτο φίλτρο (high pass filter, moving average filter). Πάλι με την βοήθεια αυτών των σημάτων αν έχει γίνει επιλογή των φίλτρων οι τιμές τους αφού περάσουν από πολυπλέκτη καταγράφονται εκ νέου σε καταχωρητές. Οι ενδιάμεσοι καταχωρητές μας δίνουν την δυνατότητα τα δεδομένα αφού έχουν περάσει από ένα φίλτρο να καταγραφούν και να μπορέσουν να χρησιμοποιηθούν και στα αλλά φίλτρα. Καθώς στην πορεία του συστήματος ως τελευταίο φίλτρο είναι του κινούμενου μέσου όρου, οι τιμές πριν περάσουν από αυτό πρέπει να ελεγχθούν για την εγκυρότητα τους. Αυτό γίνεται με την βοήθεια ενός buffer, ο οποίος περιέχει ένα D flip flop και με την διαδικασία που θα αναλυθεί παρακάτω ελέγχεται η εγκυρότητα τους. Τέλος οι τιμές αποθηκεύονται σε έναν τελευταίο καταχωρητή και βγαίνουν στην έξοδο.

Όπως προαναφέραμε δίνεται η δυνατότητα επιλογής των φίλτρων από τον χρήστη, καθώς αυτός μπορεί να επιλέξει να εφαρμόσει ένα ή δύο ή και τα τρία από τα φίλτρα που περιέχει το σύστημα . Αυτό επιτυγχάνεται με την βοήθεια διακοπτών (switches) που υπάρχουν για το κάθε φίλτρο.



Εικόνα 16: επιλογή φίλτρων μέσο διακοπτών

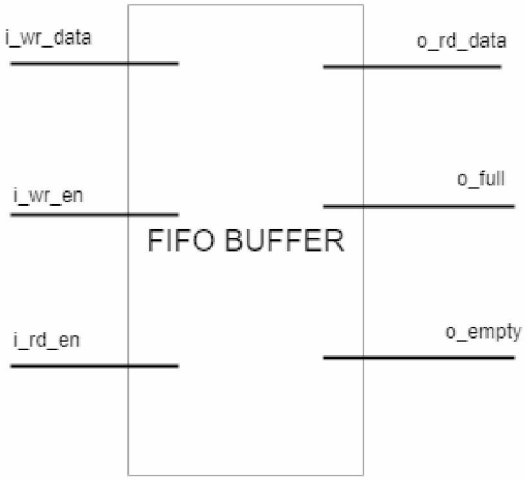
Οι τιμές όπου στην περίπτωση μας προέρχονται από εγγραφές εγκεφαλικών σημάτων θα αποθηκεύονται αρχικά σε έναν καταχωρητή και στη συνέχεια θα εισέρχονται σε έναν buffer όπου θα ακολουθεί την λογική First in First out (FIFO). Έπειτα οι τιμές αν έχει επιλεχθεί το πρώτο φίλτρο θα φιλτράρονται μία μία με βαθυπερατό φίλτρο. Αλλιώς με την βοήθεια του πολυπλέκτη θα περνάνε σε έναν δεύτερο καταχωρητή για την εφαρμογή του υψιπερατού φίλτρου. Η ίδια η διαδικασία ακολουθείται και για το φίλτρο κινούμενου μέσου όρου. Ωστόσο σε αυτή την περίπτωση οι τιμές περνάνε από έναν buffer για ελεγχθεί η εγκυρότητα τους Οι ενδιάμεσοι καταχωρητές βοηθούν στο να διατηρούνται οι τιμές που έχουν φιλτραριστεί και να περάσουν στο επόμενο

επιλεγμένο φίλτρο. Επιπλέον μαζί με τους πολυπλέκτες, δίνουν την δυνατότητα να περάσουν αυτούσιες από κάποιο μη επιλεγμένο φίλτρο κατά την ροή του συστήματος.

5.1 Υποσυστήματα

Register: Ο καταχωρητής είναι τύπος μικρής αλλά πολύ γρήγορης μνήμης. Η μνήμη αυτή χρησιμοποιείται για την βελτίωση της ταχύτητας εκτέλεσης των προγραμμάτων, αφού σε αυτήν συνήθως αποθηκεύονται δεδομένα που χρησιμοποιούνται συνέχεια από τα προγράμματα. Στην περίπτωση αυτή ο καταχωρητής παρέχει πολύ γρήγορη πρόσβαση σε αυτά τα δεδομένα και έτσι το πρόγραμμα εκτελείται πιο γρήγορα. Ένας καταχωρητής αποθηκεύει bits πληροφορίας με τέτοιον τρόπο ώστε το σύστημα να μπορεί να διαβάζει ή να γράφει όλα τα bits ταυτόχρονα. Πιο αναλυτικά ένας n-bit καταχωρητής είναι ένα σύνολο από n flip-flops, ικανό να αποθηκεύσει n bits δυαδικής πληροφορίας. Ένας απλός καταχωρητής είναι χρήσιμος όταν η πληροφορία πρέπει να καταχωρηθεί σε κάθε κύκλο ρολογιού. Στο παρόν σύστημα χρησιμοποιούνται δύο απλή καταχωρητές των 16 και 17 bits. [17][18]

Buffer FIFO: Χρησιμοποιείται για προσωρινή αποθήκευση δεδομένων, μέχρι η επόμενη διαδικασία να είναι έτοιμη να τα διαβάσει. Όπως και το όνομα τους (First In First Out) υποδηλώνει ότι το πρώτο bit γραμμένο σε έναν FIFO buffer θα είναι το πρώτο που θα εμφανιστεί στην έξοδο. Το μέγεθος των στοιχείων που μπορείς να αποθηκεύσεις δείχνει το βάθος του buffer. Όταν υλοποιείται σε υλικό (hardware) κάποιες προϋποθέσεις θα πρέπει να ληφθούν υπόψιν. Συγκεκριμένα το βάθος του buffer θα πρέπει να είναι δύναμη του δύο. Επίσης χρειάζονται δύο σημαίες (flag) FIFO_full και FIFO_empty που να δείχνουν πότε ο buffer είναι γεμάτος ή άδειος.[19][20]



Εικόνα 17: FIFO buffer του συστήματος

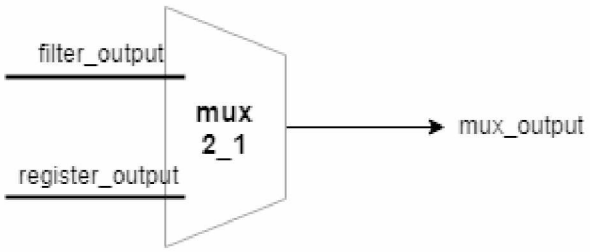
Μόλις οι τιμές από τον καταχωρητή περάσουν στον buffer θα είναι έτοιμές να περάσουν από τα φίλτρα. Για να αποφευχθούν άχρηστες τιμές που μπορεί να υπάρχουν στον buffer μετά από αυτές των σημάτων, μόλις ο buffer γεμίσει οι τιμές θα συνοδεύονται από ένα bit με την τιμή 1,που θα δηλώνει την εγκυρότητα τους. Επομένως πλέων χρησιμοποιούνται 17_bit συστήματα, 16_bit που είναι η πληροφορία και 1_bit για την εγκυρότητα όπου αυτό χρειάζεται. Πλέον οι τιμές αποθηκεύονται σε έναν δεύτερο καταχωρητή μαζί με το bit εγκυρότητας των 17_bit και στη συνέχεια περνάνε στα φίλτρα που έχουν επιλεχθεί.

Low pass filter: το βαθυπερατό φίλτρο χρησιμοποιείται κυρίως για την εξομάλυνσή του θορύβου, καθώς δεν αφήνει να περάσουν κάποιες τιμές. Στο παρόν σύστημα οι τιμές περνάνε από τον καταχωρητή για φιλτράρισμα και συγκρίνονται με επιλεγμένη τιμή. Αν είναι χαμηλότερες από αυτήν τότε γίνονται αποδεκτές και καταγράφονται σε έναν επόμενο καταχωρητή. Αν είναι υψηλότερες τότε σαν επιθυμητή τιμή διατηρείται η τιμή συγκρίσεις.

Mux: Ένας πολυπλέκτης (Mux) έχει πολλές εισόδους, ένα σήμα ελέγχου και ένα αποτέλεσμα. Η έξοδος λαμβάνει την τιμή που υπάρχει σε μία από τις εισόδους της, σύμφωνα με το σήμα ελέγχου. Το μπλοκ διάγραμμα ενός γενικού mux έχει n σήματα εισόδου (I) και m σήματα ελέγχου (S). Μια είσοδος 2n mux, χρειάζεται n σήματα ελέγχου. Ένα κύκλωμα mux, γνωστό και ως " επιλογέας δεδομένων ", επιτρέπει μόνο

μία είσοδο που διατίθεται στην έξοδο του κάθε φορά, σύμφωνα με τα σήματα ελέγχου.[21]

Με την βοήθεια του πολυπλέκτη δίνεται η δυνατότητα να περάσουν τα δεδομένα αυτούσια (αν δεν έχει επιλεχθεί το φίλτρο) είτε να περάσουν φιλτραρισμένα από το εκάστοτέ φίλτρο. Αν έχει επιλεχθεί ένα φίλτρο μέσω των διακοπτών τότε ενεργοποιείται ένα σήμα και τα δεδομένα από τον καταχωρητή περνάνε στο φίλτρο. Από εκεί είτε οι νέες τιμές ή οι αρχικές περνάνε σε έναν δεύτερο καταχωρητή για να γίνει χρήση τους από το επόμενο φίλτρο.



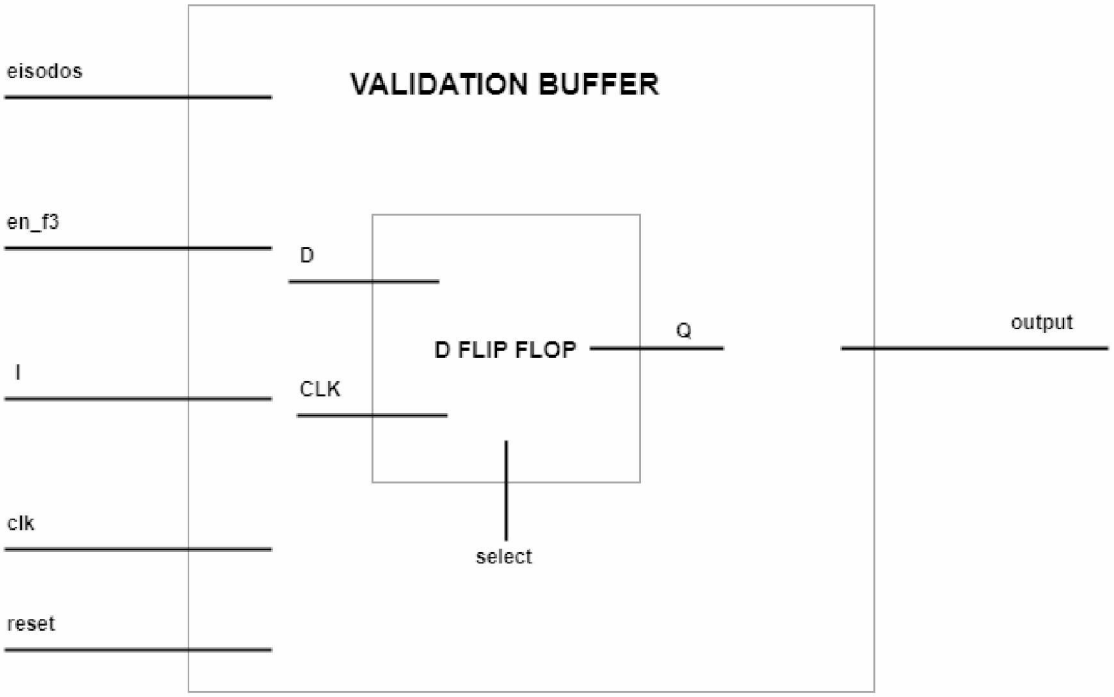
Εικόνα 18: πολυπλέκτης που επιλέγει ανάμεσα στις φιλτραρισμένες τιμές και τις αρχικές

High pass filter: το Υψιπερατό φίλτρο χρησιμοποιείται επίσης για την εξομάλυνση χαμηλών τιμών θορύβου. Και σε αυτή την περίπτωση οι τιμές εισέρχονται από τον καταχωρητή και συγκρίνονται με μια επιλεγμένη τιμή. Εάν είναι υψηλότερες από αυτήν τότε γίνονται αποδεκτές και περνάνε στον πολυπλέκτη ώστε να αποθηκευτούν σε καταχωρητή. Αν είναι χαμηλότερες από την επιλεγμένη τιμή τότε διατηρείται αυτή.

Στη συνέχεια ακολουθεί το τρίτο φίλτρο που είναι φίλτρο κινούμενου μέσου όρου. Ωστόσο πριν περάσουν οι τιμές σε αυτό το φίλτρο πρέπει να ελεγχθεί η εγκυρότητα τους. Αυτό γίνεται με την βοήθεια ενός buffer, ο οποίος έχει σχεδιαστεί ώστε να δέχεται κάποια σήματα ελέγχου και να περνάει μόνο τις έγκυρες τιμές στο φίλτρο κινούμενου μέσου όρου.

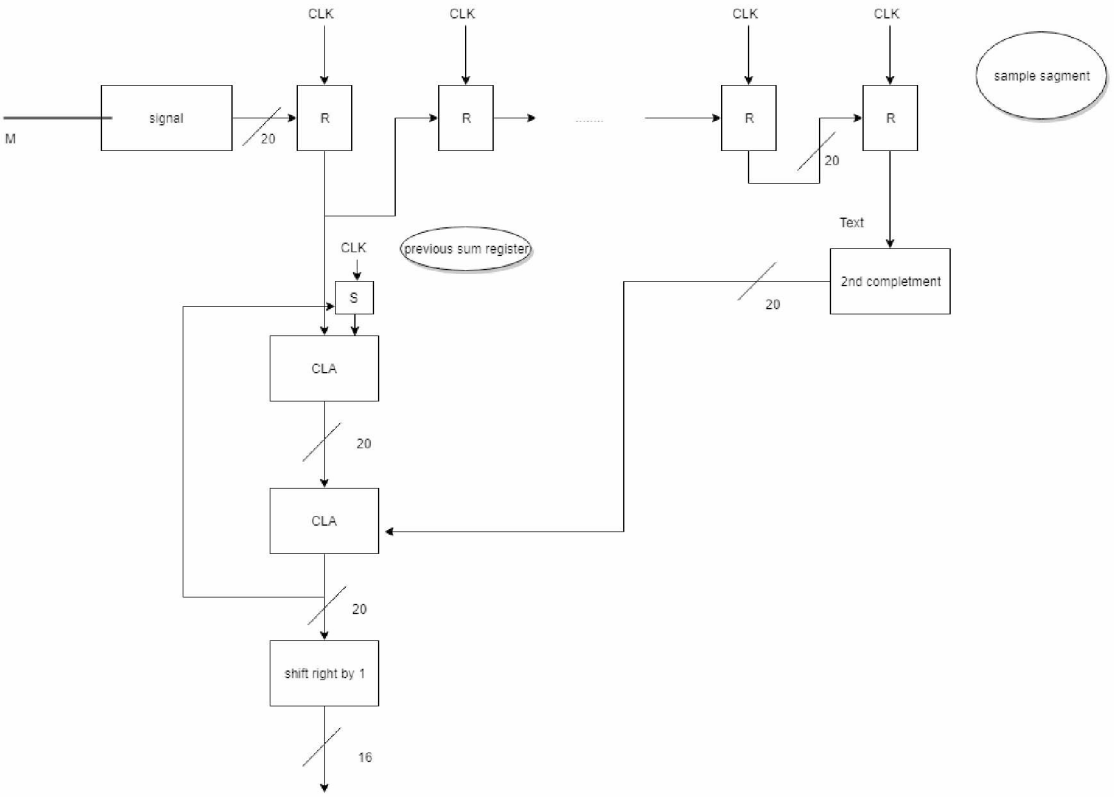
Validation buffer: είναι ένας buffer με 16_bit είσοδο, η οποία περνάει από ένα d Flip Flop. Η επιλογή για το αν η τιμή θα περάσει στην έξοδο του flip flop γίνεται με τη βοήθεια ενός select. Το select παίρνει τις τιμές 0 και 1, όπου προκύπτουν από δύο σήματα ελέγχου που υποδεικνύουν την εγκυρότητα των τιμών. Το πρώτο σήμα είναι η ενεργοποίηση του φίλτρου (enable moving average) και το δεύτερο είναι το σήμα που προκύπτει όταν γεμίσει ο αρχικός fifo buffer. Αυτά τα δυο σήματα συνδέονται με ένα λογικό και (and) και δίνουν τιμή στο select. Η επιλογή των συγκεκριμένων σημάτων για την εγκυρότητα των τιμών έγινε γιατί το πρώτο δείχνει ότι το φίλτρο έχει ενεργοποιηθεί για χρήση και το δεύτερο είναι το bit που ακολουθεί τις τιμές για να

βεβαιώσει ότι προέρχονται από τον αρχικό buffer. Έτσι περνάνε μόνο οι έγκυρες τιμές στην έξοδο του buffer και απορρίπτονται τυχόν τιμές που έχουν προκύψει ύστερα.



Εικόνα 19: περιγραφή του validation buffer

Moving average filter: το φίλτρο κινούμενου μέσου όρου είναι ένα από τα φίλτρα που χρησιμοποιούνται σε ευρεία ποικιλία σημάτων παρέχοντας δυνατότητες εξομάλυνσης. Στην παρούσα πτυχιακή το φίλτρο υλοποιείται ψηφιακά με 16_bit είσοδο. Καθώς το φίλτρο δέχεται προσημασμένες τιμές τα 16_bit μετατρέπονται σε 20_bit. Το φίλτρο δουλεύει με τον εξής τρόπο: αρχικά ένας καταχωριστής που έχει την μορφή πίνακα μεγέθους 20 γεμίζει με μηδενικές τιμές και το άθροισμα (sum) αρχικοποιείται με μηδενικές τιμές. Όταν ξεκινήσει ο κύκλος του ρολογιού οι τιμές του καταχωρητή μετακινούνται προς τα δεξιά κατά ένα. Έτσι η πρώτη θέση του καταχωρητή είναι άδεια και εκεί αποθηκεύεται η πρώτη τιμή. Στη συνέχεια στο sum αποθηκεύονται οι τιμές των αθροισμάτων. Έπειτα γίνεται η επέκτασή των πρόσημων οπού ανατίθονται στην τιμή first και ενώνονται οι υπόλοιπες τιμές. Με φορά από τα αριστερά στα δεξιά αρχικά έχουμε τα bit των πρόσημων και μετά τα υπόλοιπα 16_bit. Στην τελευταία τιμή ανατίθεται η τιμή που βρίσκεται στην τελευταία θέση του καταχωρητή. Τέλος με την βοήθεια αθροιστών πρόβλεψης κρατουμένου αθροίζονται οι πρώτες τιμές και αφαιρείται η τελευταία. [22]



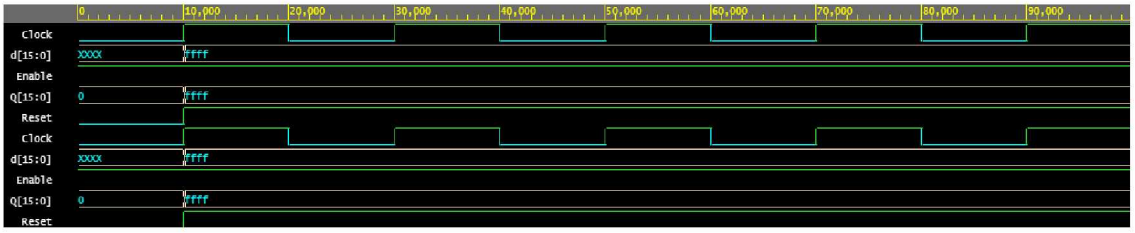
Εικόνα 20: σχεδίαση φίλτρου κινουμένου μέσου όρου

6. Υλοποίηση συστήματος

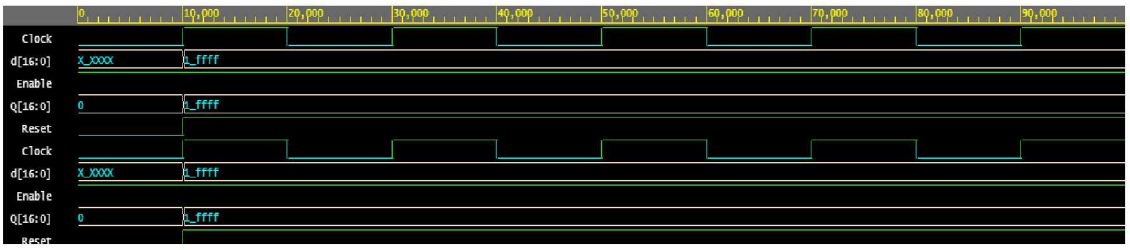
Το σύστημα που αναπτύχθηκε στο πλαίσιο εκπόνησης της πτυχιακής εργασίας, υλοποιείται με την βοήθεια της γλώσσας περιγραφής υλικού (VHDL). Η VHDL είναι μια γλώσσα που χρησιμοποιείται στον αυτοματισμό ηλεκτρικών σχεδιάσεων για την περιγραφή ψηφιακών και μεικτών συστημάτων [22]. Στο παρών κεφάλαιο θα παρουσιαστούν τα αποτελέσματα από την υλοποίηση όλων των υποσυστημάτων καθώς και κάθε περίπτωση εφαρμογής των φίλτρων του τελικού συστήματος

6.1 Προσομοίωση καταχωρητή (Register)

Οι καταχωρητές αποτελούν σημαντικό κομμάτι του συστήματος καθώς σε αυτό χρησιμοποιούνται 5 και αποθηκεύουν τις πληροφορίες σε διάφορα στάδια του συστήματος. Βασικό χαρακτηριστικό των καταχωρητών αυτών είναι ότι με την άνοδο του ρολογιού η είσοδος μεταφέρεται στην έξοδο. Παρακάτω παρατίθεται η προσομοίωση των 16_bit και 17_bit καταχωρητών, οι οποίοι έχουν υλοποιηθεί με τον κώδικα που βρίσκεται στο παράρτημα της εργασίας.



Εικόνα 21: Κυματομορφή προσομοίωση 16_bit καταχωρητή

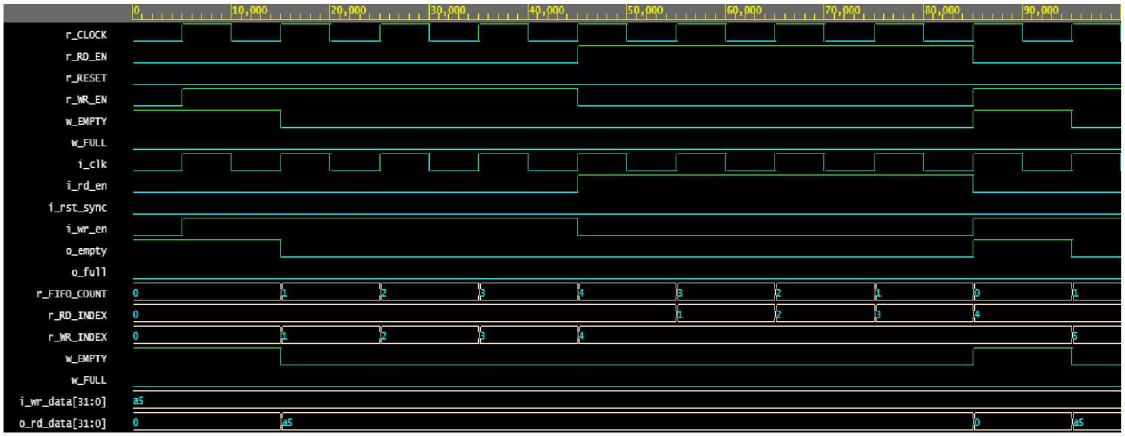


Εικόνα 22: Κυματομορφή προσομοίωσης 17_bit καταχωρητή

Στις προσομοιώσεις μπορούμε να δούμε ότι με το που ο κύκλος του ρολογιού πάρει την τιμή 1, η είσοδός εξέρχεται στην έξοδο.

6.2 Προσομοίωση FIFO buffer

Σε αυτόν τον buffer αποθηκεύονται οι τιμές από τον πρώτο καταχωρητή και στη συνέχεια εισέρχονται μία μία σε έναν δεύτερο μαζί με ένα bit που τις συνοδεύει αφού γεμίσει ο buffer από αυτές . Καθώς ο buffer είναι FIFO υπάρχουν δύο ιδιότητες που μπορούν να διαμορφωθούν για να ρυθμίσουν το πλάτος και το βάθος του. Το g_WIDTH ρυθμίζει το μέγεθος της εισόδου και της εξόδου, ώστε να μπορούν να γραφτούν διαφορετικά μεγέθη. Το g_DEPTH ρυθμίζει πόσο μεγάλη είναι η εσωτερική μνήμη του FIFO. Αυτά τα δύο είναι χαρακτηριστικά που υπάρχουν στον κώδικα που παρατίθεται παρακάτω.

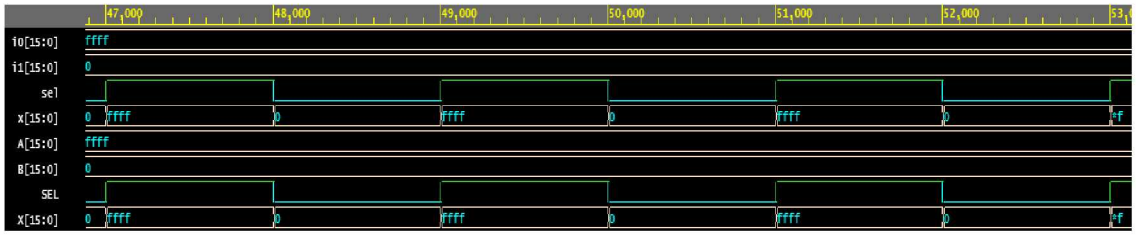


Εικόνα 23 Κυματομορφή προσομοίωσης FIFO buffer

Βλέπουμε ότι μόλις ενεργοποιηθεί το σήμα i_wr_en αρχίζουν να γράφονται οι τιμές στον buffer. Ενώ όταν το i_rd_en πάρει την τιμή ένα, δηλαδή ενεργοποιηθεί τότε τα δεδομένα γράφονται στο o_rd_data. Έτσι είναι έτοιμα να βγουν στην έξοδο του.

6.3 Προσομοίωση πολυπλέκτη (Mux)

Οι πολυπλέκτες στο σύστημα μας επιλέγουν ποια δεδομένα θα καταγράφουν στους καταχωρητές. Αποτελούνται από δύο εισόδους και με την βοήθεια ενός select επιλέγουν ποια θα βγει στη έξοδο. Η πρώτη είσοδος είναι οι τιμές που προέρχονται από την εφαρμογή κάποιου από τα φίλτρα και η δεύτερη οι τιμές που υπάρχουν τα πριν την εφαρμογή αυτών. Ως select κάθε φορά παίρνουμε το enable των φίλτρων. Μπορούμε να δούμε την κυματομορφή από την προσομοίωση που προέκυψε από κώδικα που παρατίθεται παρακάτω.

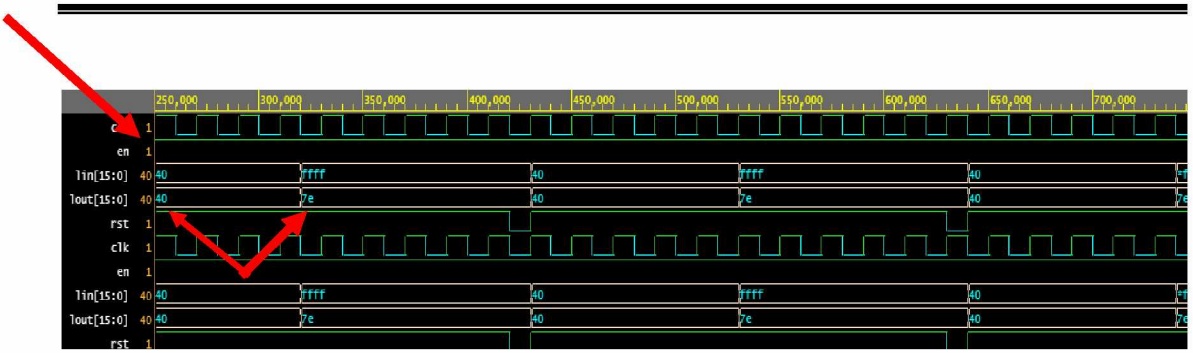


Εικόνα 24: Κυματομορφή προσομοίωσης πολυπλέκτη

Βλέπουμε ότι μόλις το select πάρει την τιμή 1 περνάει στην έξοδο την πρώτη είσοδο που είναι αυτή που προέχεται από την εφαρμογή του φίλτρου. Αν το select είναι 0 τότε περνάει η δεύτερη είσοδος.

6.4 Προσομοίωση βαθυπερατού φίλτρου (Low pass filter)

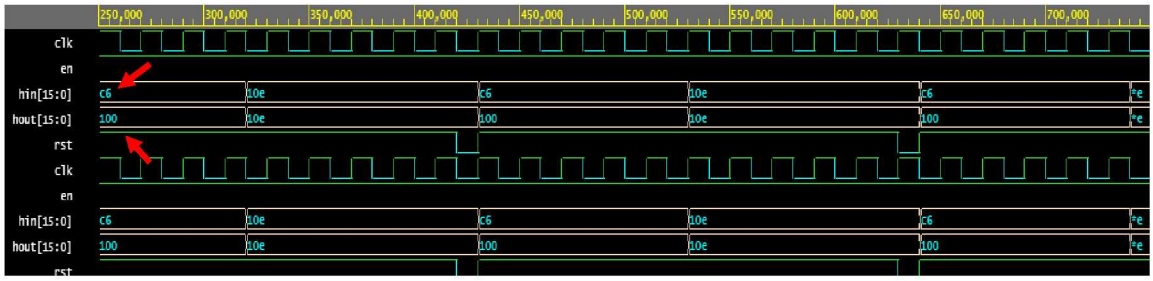
Το φίλτρο αποτελείται από μια είσοδο (lin), μια έξοδο (lout) και ένα enable (en). Το enable ενεργοποιείται κάθε φορά που το φίλτρο επιλέγεται από τους διακόπτες μέσω ενός σήματος en_f1. Έπειτα ελέγχεται η τιμή και είτε περνάει αυτούσια στην έξοδο είτε περνάει η τιμή που αντιστοιχεί στην συχνότητα αποκοπής. Όπως βλέπουμε και στο παρακάτω σχήμα το en έχει την τιμή 1, έτσι η τιμή 40 βρίσκεται εντός των ορίων και περνάει η ίδια στην έξοδο. Αντιθέτως η τιμή ffff (65535) βρίσκεται εκτός των ορίων με αποτέλεσμα να περνάει η τιμή 7e(126) που ορίστηκε ως όριο.



Εικόνα 25: Κυματομορφή προσομοίωσης βαθυπερατού φίλτρου

6.5 Προσομοίωση υψιπερατού φίλτρου (High pass filter)

Το υψιπερατό φίλτρο όπως και το βαθυπερατό αποτελείται από μία είσοδο (hin), ένα enable (en) και μια έξοδο (hout). Η διαφορά με το βαθυπερατό φίλτρο είναι ότι εδώ επιτρέπονται να περάσουν οι τιμές που είναι υψηλότερες από το όριο και στην θέση αυτών που είναι χαμηλότερες η τιμή του ορίου. Και σε αυτή την περίπτωση το enable παίρνει τιμή από την επιλογή του φίλτρου μέσο των διακοπτών. Στην προσομοίωση μπορούμε να δούμε ότι η τιμή c6 (198) που είναι χαμηλότερη από το όριο απορρίπτεται και στη θέση της βγαίνει στην έξοδο η τιμή 100 (256). Αντιθέτως η τιμή 10e (270) που είναι υψηλότερη από το 256 περνάει κανονικά στην έξοδο.

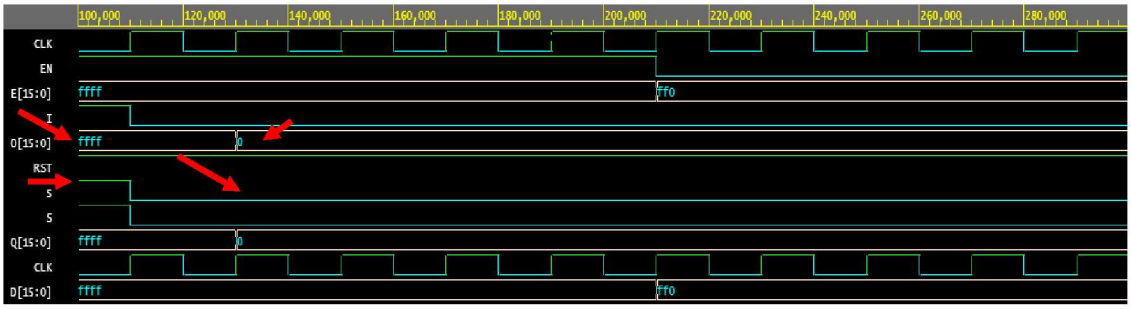


Εικόνα 26: Κυματομορφή προσομοίωσης υψιπερατού φίλτρου

6.6 Προσομοίωση Validation buffer

Ο buffer αυτός αποτελείται από ένα D Flip Flop, του οποίου η λειτουργία είναι να επιτρέπει ανάλογα με το select την είσοδο να βγει στην έξοδο. Δύο από τις εισόδους του buffer είναι αυτές που δίνουν τιμή στο select του D flip Flop. Η πρώτη είναι το EN που υποδηλώνει την ενεργοποίηση του φίλτρου κινούμενου μέσου όρου και η δεύτερη το I που είναι το bit που ακολουθεί τις τιμές και υποδηλώνει την προέλευση τους από τον αρχικό buffer. Έτσι στην έξοδο του buffer περνάνε μόνο οι έγκυρες τιμές μέσο του

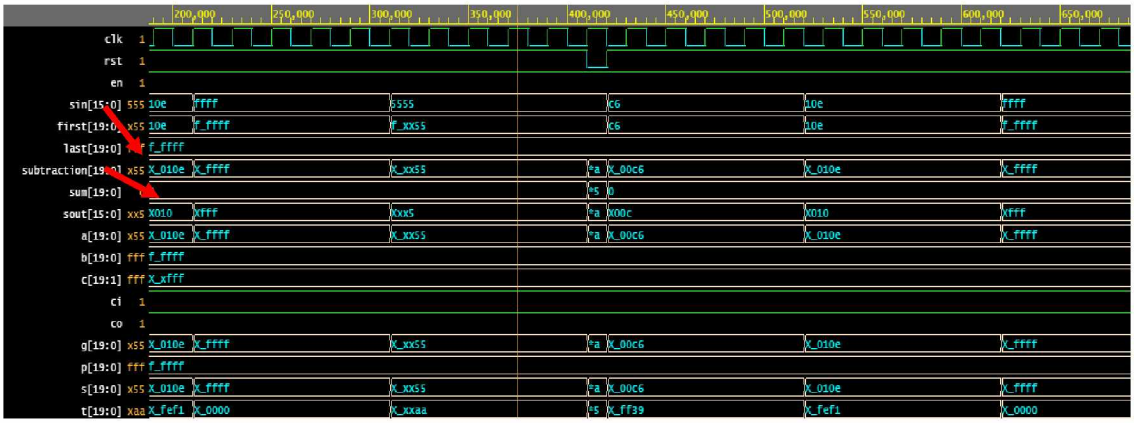
D Flip Flop. Όπως φαίνεται και παρακάτω αρχικά τα σήματα EN I έχουν την τιμή 1 και επομένως το S (EN and I) ισούται με 1 και η είσοδος περνάει στην έξοδο. Έπειτα το EN διατηρεί την τιμή 1 ενώ το I γίνεται 0, έτσι και το S μηδενίζεται και δεν περνάει τίποτα στην έξοδο.



Εικόνα 27: Κυματομορφή προσομοίωσης validation buffer

6.7 Προσομοίωση φίλτρου κινουμένου μέσου όρου (Moving average filter)

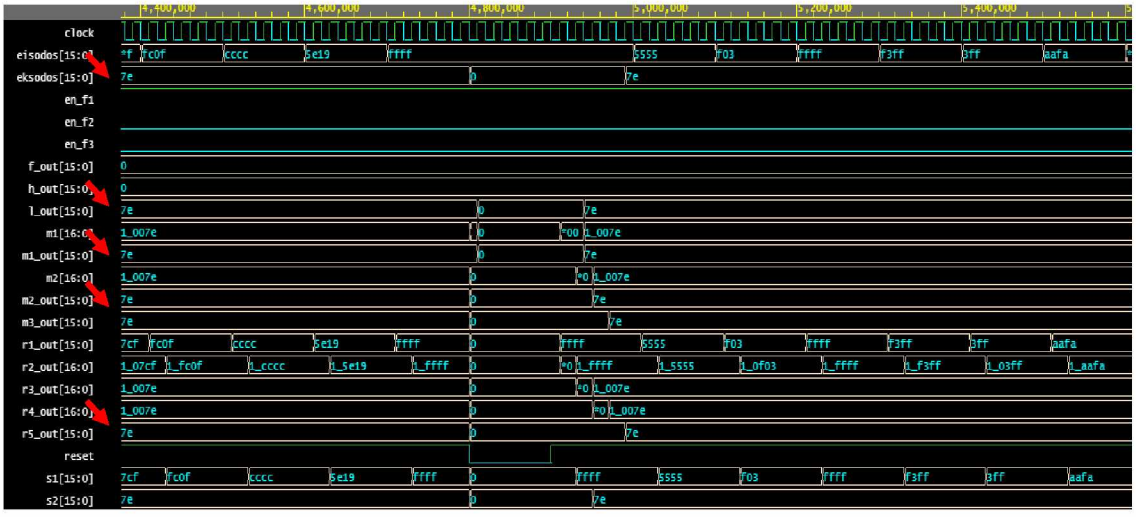
Η προσομοίωση που βλέπουμε παρακάτω προέρχεται από κώδικα του φίλτρου κινούμενου μέσου όρου που παρατίθεται παρακάτω. Τα σήματα εισόδου είναι clock, reset, enable και η είσοδος των τιμών (sin). Το enable παίρνει την τιμή από την ενεργοποίηση του φίλτρου, επομένως όταν αυτό έχει την τιμή ένα τότε ξεκινάει η διαδικασία φιλτραρίσματος των τιμών. Στο γράφημα μπορούμε να δούμε ότι το first έχει ως πρώτη την τιμή που προκύπτει από το άθροισμα των τεσσάρων πρώτων τιμών και έπειτα ακολουθούν οι υπόλοιπες τιμές. Το last έχει αποθηκευμένη την τελευταία τιμή. ενώ στο sum έχει την τιμή του subtraction. Η τιμή subtraction προέρχεται από την έξοδο του carry look ahead adder που σαν είσοδο έχει τα addition και last. Η τιμή addition επίσης προέρχεται από carry look ahead adder που οι είσοδοι του είναι το first και το sum. Με αυτή την διαδικασία προκύπτουν τα αθροίσματα των τιμών, που θα είναι η τελική έξοδος του φίλτρου. Ουσιαστικά η έξοδος του φίλτρου όπως μπορούμε να δούμε έχει την τιμή του subtraction μη προσημασμένη.

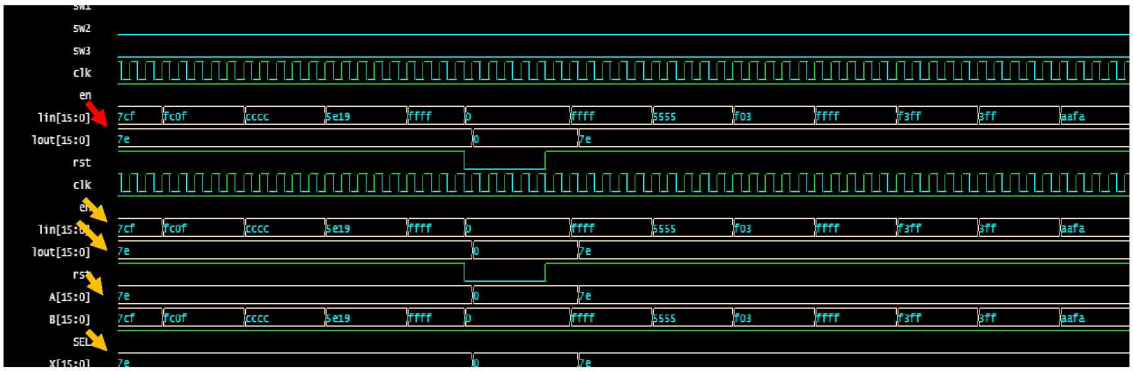


Εικόνα 28: Κυματομορφή προσομοίωσης φίλτρου κινούμενου μέσου όρου

6.8 Προσομοίωση τελικού συστήματος

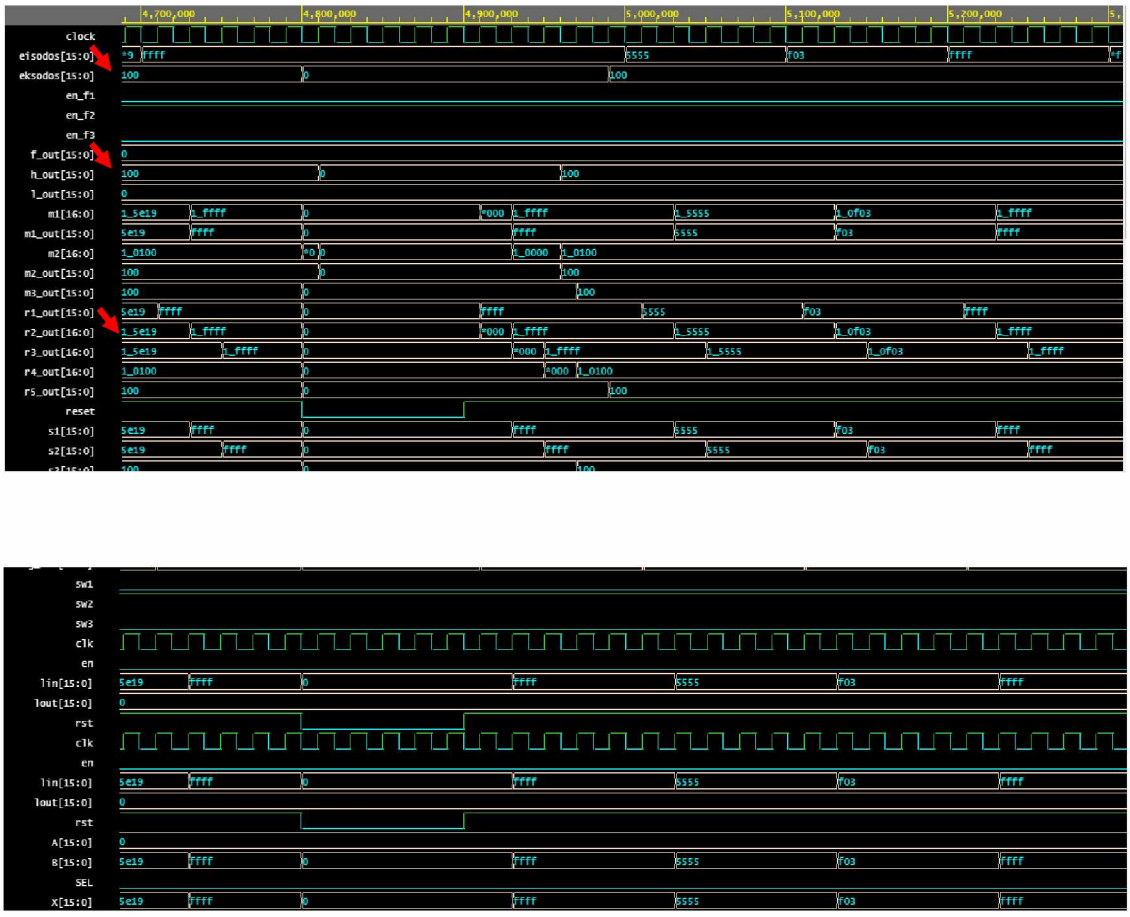
Καθώς είδαμε την υλοποίηση και τα αποτελέσματα των υποσυστημάτων που συντελούν το τελικό σύστημα που αναλύεται στην παρούσα πτυχιακή. Τώρα θα παρουσιαστεί το τελικό αποτέλεσμα που προκύπτει από τον συνδυασμό και την ενοποίηση αυτών. Όπως έχει προαναφερθεί μπορούν να εφαρμοστούν περισσότερα του ενός φίλτρου με την βοήθεια διακοπτών. Επομένως θα δούμε τις κυματομορφές που προκύπτουν από την προσομοίωση κάθε πιθανού σεναρίου επιλογής φίλτρων. Δηλαδή θα δούμε την εφαρμογή μεμονωμένου φίλτρου, για κάθε ένα από αυτά lowpass, highpass και moving average filter. Επιπλέον θα γίνει συνδυασμός αυτών, του πρώτου με του δευτέρου, με το τρίτο, το δεύτερο με το τρίτο καθώς και τα τρία μαζί.





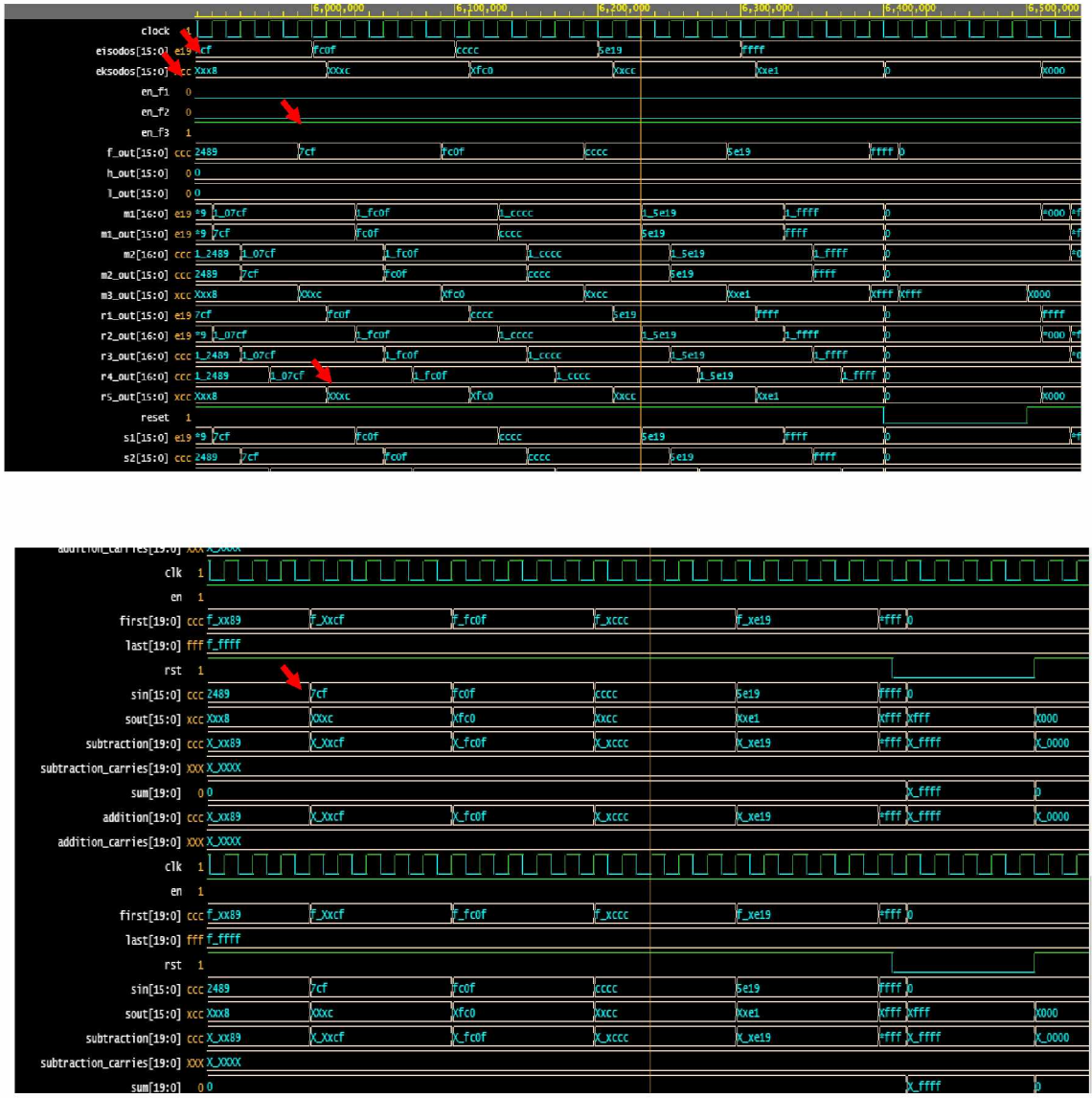
Εικόνα 29: Κυματομορφή προσομοίωσης τελικού συστήματος – επιλογή πρώτου φίλτρου

Αρχικά βλέπουμε την είσοδο που έχει την τιμή 7cf, η οποία περνάει στον καταχωρητή r1_out. Στη συνέχεια το r2_out έχει την τιμή του πρώτου καταχωρητή μαζί με το validation bit (1_07cf). Καθώς το en_f1 έχει την τιμή ένα, το οποίο έχει ενεργοποιηθεί από τον sw1, το 7cf περνάνε στο lin και βγαίνει η τιμή 7e στο lout. Έπειτα στο αποτέλεσμα του πρώτου πολυπλέκτη m1_out έχουμε το 7e καθώς το select έχει την τιμή 1. Η ίδια τιμή περνάει και στους υπόλοιπούς ενδιάμεσους καταχωρητές και πολυπλέκτες r3_out, r4_out, m2_out m3_out. Το f_out που είναι το αποτέλεσμα του validation buffer έχει την τιμή μηδέν καθώς το select του D Flip Flop του είναι μηδέν (en_f3=0 and I=1). Τέλος στον τελευταίο καταχωρητή r5_out 7e που προήλθε από το low pass filter.όλα αυτά μπορούμε να τα δούμε πάνω στην εικόνα με την βοήθεια των κόκκινων βελακίων. Ενώ με τα κίτρινα βελάκια βλέπουμε τις τιμές στην είσοδο lin (7cf) του φίλτρου,A (7e), B (7cf) και SEL (1) του καταχωρητή καθώς και την έξοδο του x που έχει ίδια τιμή με την έξοδο lout (7e) του φίλτρου. Επομένως μπορούμε να δούμε ότι όλη η ροή του συστήματος λειτουργεί σύμφωνα με το απαιτούμενο αποτέλεσμα.



Εικόνα 30: Κυματομορφή προσομοίωσης τελικού συστήματος – επιλογή δεύτερου φίλτρου

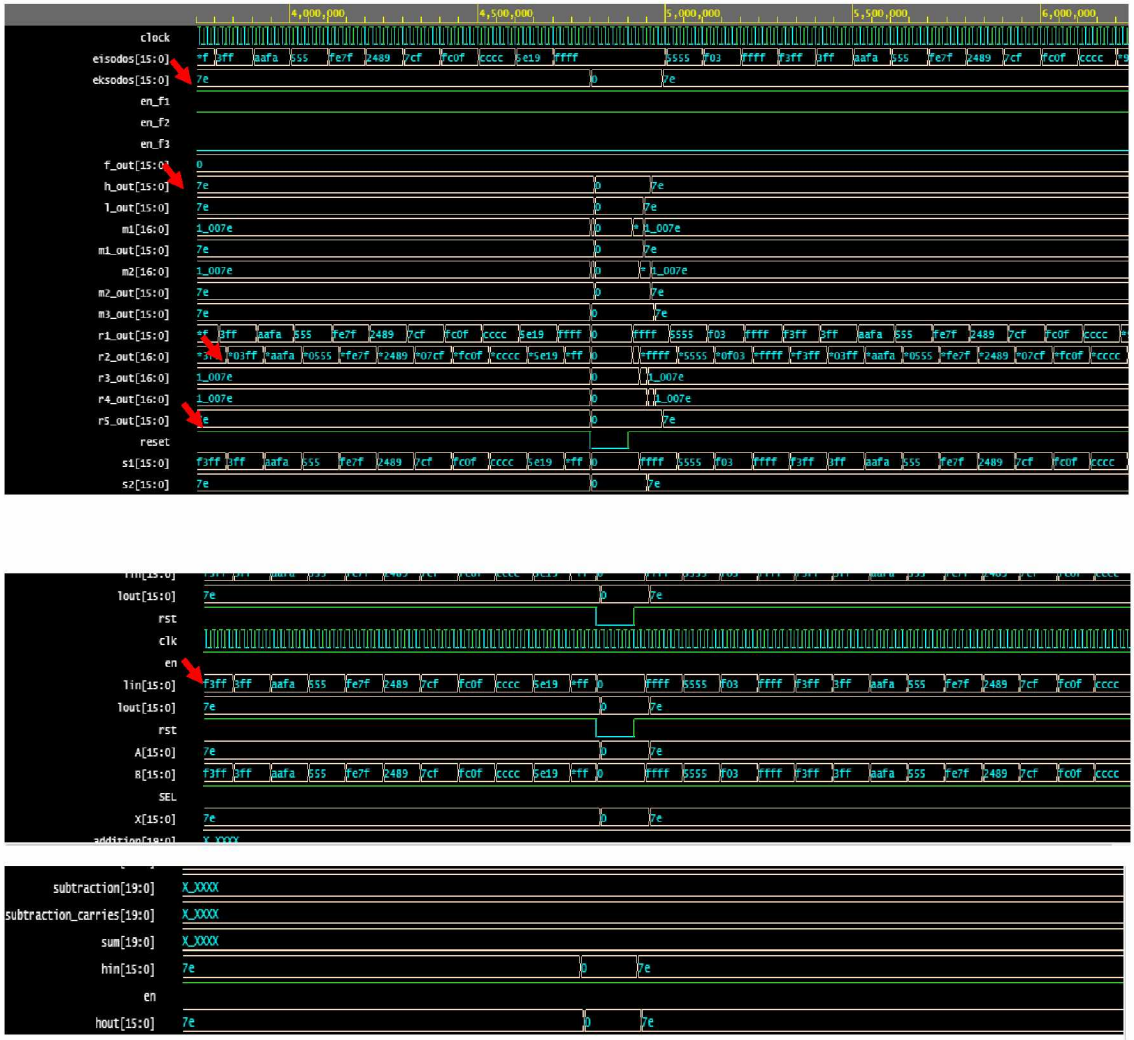
Ξεκινάμε με την είσοδο που έχει την τιμή 5c19, η οποία περνάει στον καταχωρητή r1_out. Στη συνέχεια το r2_out έχει την τιμή του πρώτου καταχωρητή μαζί με το validation bit (1_5c19). Έπειτα στο αποτέλεσμα του πρώτου πολυπλέκτη m1_out έχουμε το 5e19, καθώς το select έχει την τιμή 0. Η τιμή αυτή καταγράφεται και στον καταχωρητή r3_out. Καθώς το en_f2 έχει την τιμή ένα, το οποίο έχει ενεργοποιηθεί από τον sw2, το 5c19 περνάνε στο hin και βγαίνει η τιμή 100 στο hout, αφού η είσοδος ήταν χαμηλότερη από το όριο που θέσαμε Η ίδια τιμή περνάει και στους υπόλοιπούς ενδιάμεσους καταχωρητές και πολυπλέκτες r4_out, m2_out m3_out. Το f_out που είναι το αποτέλεσμα του validation buffer έχει την τιμή μηδέν καθώς το select του D Flip Flop του είναι μηδέν (en_f3=0 and I=1). Τέλος στον τελευταίο καταχωρητή r5_out περνάει το 100 που προήλθε από το high pass filter. Όλα αυτά μπορούμε να τα δούμε πάνω στην εικόνα με την βοήθεια των κόκκινων βελακίων. Ενώ βλέπουμε και τις τιμές στην είσοδο lin (5c19) του φίλτρου,A (0) προερχόμενο από το lout, B (5c19) και SEL (0) του καταχωρητή καθώς και την έξοδο του x που έχει την τιμή (5c19). Επομένως μπορούμε να δούμε ότι όλη η ροή του συστήματος λειτουργεί σύμφωνα με το απαιτούμενο αποτέλεσμα.



Εικόνα 31: Κυματομορφή προσομοίωσης τελικού συστήματος – επιλογή τρίτου φίλτρου

Πρώτα βλέπουμε την είσοδο που έχει την τιμή 7cf, η οποία περνάει στον καταχωρητή r1_out. Στη συνέχεια το r2_out έχει την τιμή του πρώτου καταχωρητή μαζί με το validation bit (1_07cf). Έπειτα στο αποτέλεσμα του πρώτου πολυπλέκτη m1_out έχουμε το 7cf, καθώς το select έχει την τιμή 0 Η ίδια τιμή περνάει και στους υπόλοιπούς ενδιάμεσους καταχωρητές και πολυπλέκτες r3_out, r4_out, m2_out. Καθώς το en_f3 έχει την τιμή ένα, το οποίο έχει ενεργοποιηθεί από τον sw3, το 7cf περνάει σαν είσοδος στον validation buffer. Το f_out που είναι το αποτέλεσμα του validation buffer έχει την τιμή 7cf καθώς το select του D Flip Flop του είναι ένα (en_f3=1 and I=1). Έτσι ελέγχθηκε η εγκυρότητα της εισόδου και μπορεί να περάσει

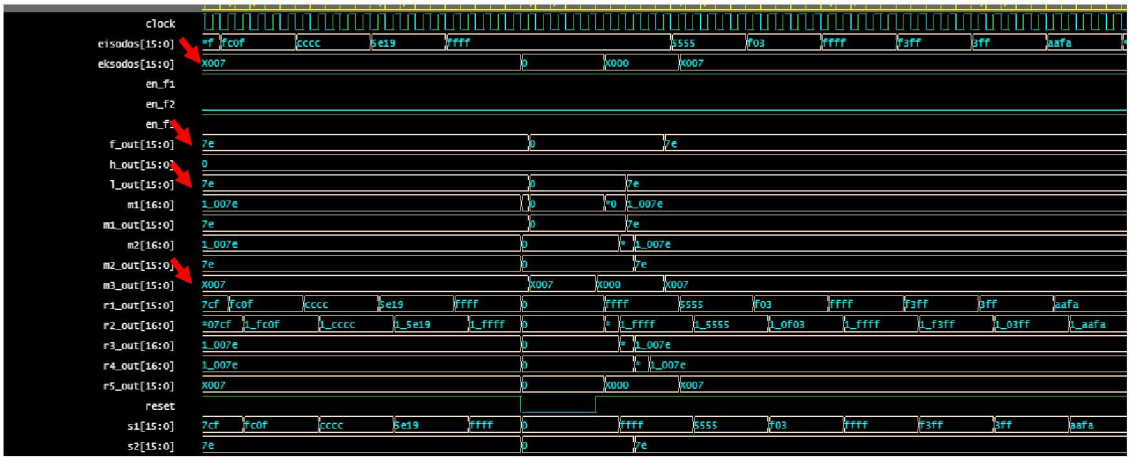
το φίλτρο κινούμενου μέσου όρου. Από το οποίο βλέπουμε το αποτέλεσμα στο sout. Τέλος στον τελευταίο καταχωρητή r5_out περνάνε οι τιμές που προήλθαν από το moving average filter. Συνεπώς μπορούμε να δούμε ότι όλη η ροή του συστήματος λειτουργεί σύμφωνα με το απαιτούμενο αποτέλεσμα.



Εικόνα 32: Κυματομορφή προσομοίωσης τελικού συστήματος – επιλογή πρώτου και δεύτερου φίλτρου

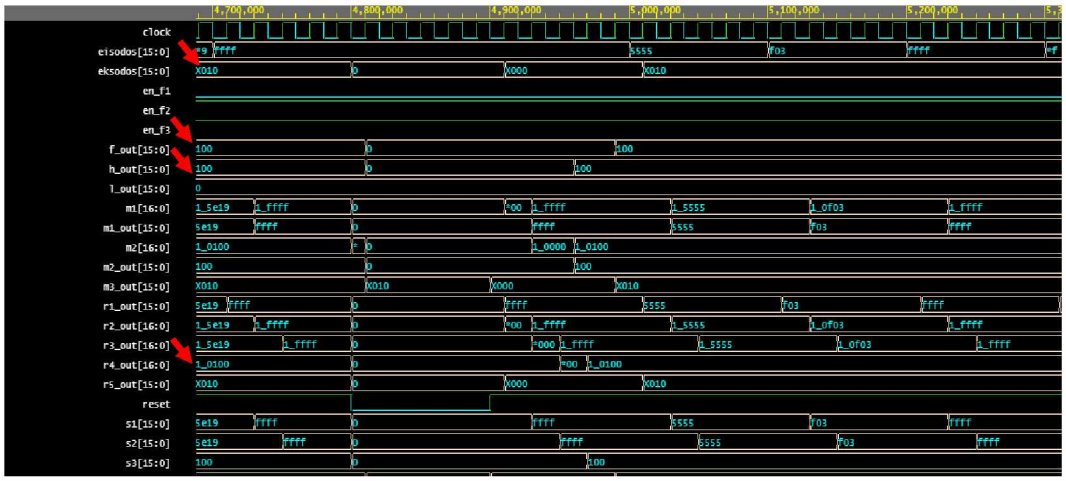
Σε αυτή την περίπτωση έχουμε την ενεργοποίηση δύο φίλτρων en_f1 και en_f2 μέσο των διακοπτών sw1και sw2. Αρχικά στην είσοδο έχουμε την τιμή 7e, η οποία περνάει αυτούσια στους δύο πρώτους καταχωρητές r1_out r2_out με την μόνη διαφορά ότι εδώ συνοδεύεται από το bit εγκυρότητας (1_07e). Από εκεί περνάει σαν είσοδος στο πρώτο φίλτρο, όπου και είναι εντός ορίων και βγαίνει στην έξοδο. Στην συνέχεια με την βοήθεια του πρώτου πολυπλέκτη καταγράφεται στον τρίτο καταχωρητή και από εκεί περνάει στην είσοδο του δεύτερου φίλτρου hin. Και πάλι είναι εντός όριων, άρα

βγαίνει στην έξοδο hout. Εν τέλει, αφού προσπεράσει τον validation buffer ο οποίος περνάει την τιμή μηδέν στην έξοδο του, καταλήγει στο τελευταίο καταχωρητή r5_out και από εκεί στην έξοδο. Όλα αυτά μπορούμε να τα διακρίνουμε πάνω στο διάγραμμα. Επίσης έχουν σημανθεί με βελάκια τα σημαντικά σημεία.



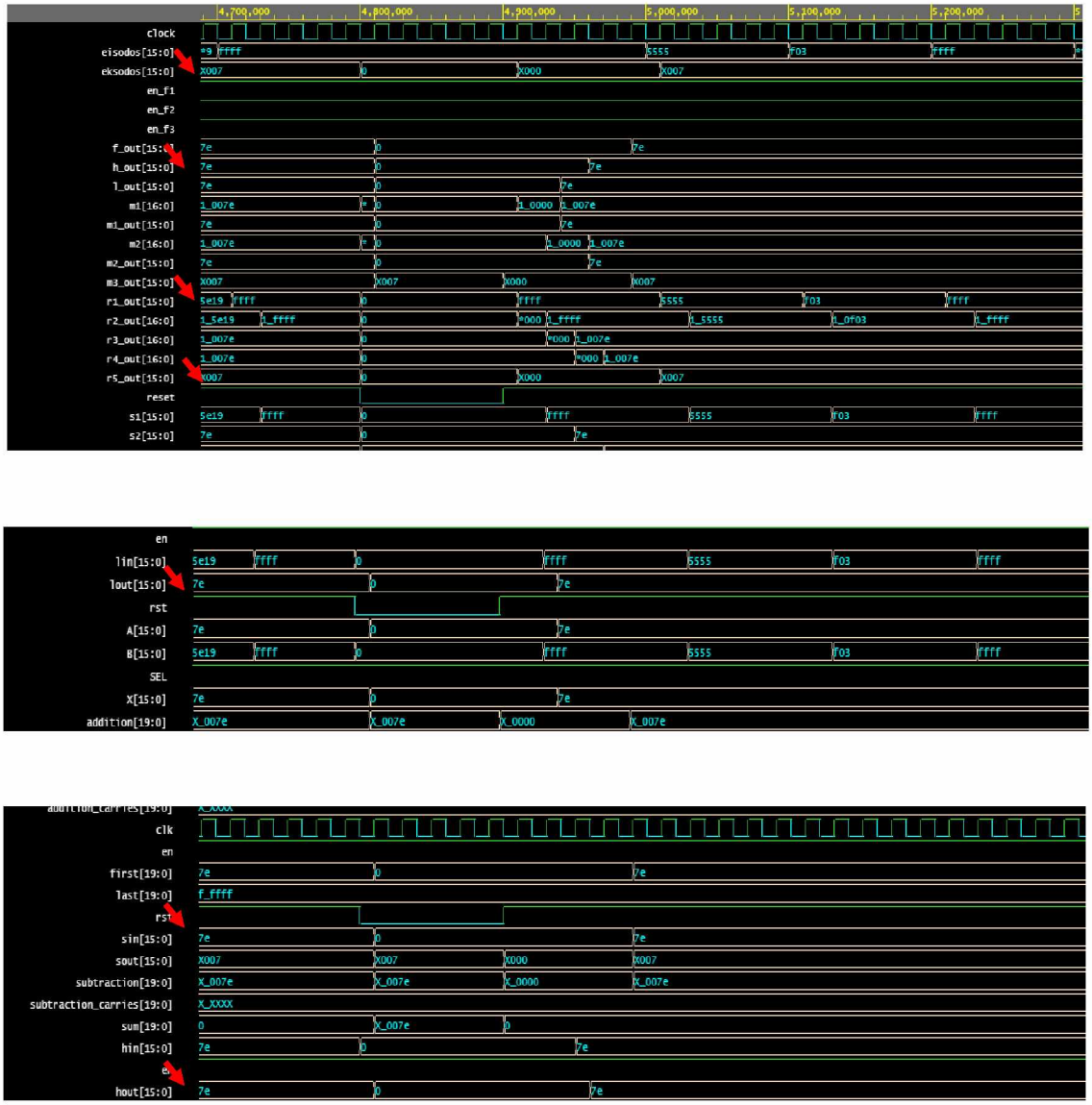
Εικόνα 33: Κυματομορφή προσομοίωσης τελικού συστήματος – επιλογή πρώτου και τρίτου φίλτρου

Η τιμή 7e της εισόδου περνάει στους δύο πρώτους καταχωρητές r1_out και r2_out. Έπειτα καθώς τα sw1 και sw3 έχουν την τιμή 1 ενεργοποιούνται τα en_f1και en-f3. Έτσι η τιμή 7e περνάει από το r2_out στο lin (7e) από όπου βρίσκεται εντός ορίων και βγαίνει στην έξοδο. Στη συνέχεια καταγράφεται στα r3_out, m1_out, m2_out, r4_out. Αφού το en_f3 και το I έχουν την τιμή ένα, η τιμή περνάει από το validation buffer (sel = en-f3 and I =1) και κρίνεται έγκυρη να εισέλθει στο φίλτρο κινούμενου μέσου όρου. Από εκεί βγαίνουν οι νέες τιμές (x007,x000) όπου μέσου του τρίτου πολυπλέκτη καταγράφονται στον καταχωρητή r5 (r5_out) και βγαίνουν στην έξοδο του συστήματος.



Εικόνα 34: Κυματομορφή προσομοίωσης τελικού συστήματος – επιλογή δεύτερου και τρίτου φίλτρου

Σε αυτή την περίπτωση έχουν επιλεχθεί το δεύτερο και τρίτο φίλτρο, επομένως τα sw2, sw3, en_f2, en_f3 έχουν την τιμή ένα. Η αρχική τιμή 5c19 καταγράφεται στους καταχωρητές r1_out και r2_out συνοδευόμενη από το bit εγκυρότητας. Στην συνέχεια καθώς το en_f1 δεν είναι ενεργοποιημένο προσπερνάει το Low pass filter, αποθηκεύεται στον r3_out και με την βοήθεια του πρώτου πολυπλέκτη κατευθύνεται στην είσοδο του high pass filter hin. Εκεί η τιμή κρίνεται εκτός ορίου και επομένως στον έξοδο hout καταγράφεται η τιμή ορίου (100). Στην συνέχεια η τιμή αυτή αποθηκεύεται στον τέταρτο καταχωρητή r4_out,περνάει στον δεύτερο πολυπλέκτη m2_out και περνάει από τον validation buffer μαζί με το bit εγκυρότητας της και κρίνεται αποδεκτή να εισέλθει στο φίλτρο κινούμενου μέσου όρου (sel = en_f3(1) and I(r4(16)). Τέλος από το φίλτρο κινούμενου μέσου όρου προκύπτει η τιμή x010, η οποία και αποθηκεύεται στον τελευταίο καταχωρητή. Όσα προαναφέρθηκαν μπορούν να παρατηρηθούν στην παραπάνω προσομοίωση.



Εικόνα 35: Κυματομορφή προσομοίωσης τελικού συστήματος – επιλογή και των τριών φίλτρων

Σαν τελευταία επιλογή έχουμε αυτή της εφαρμογής όλων των φίλτρων. Επομένως sw1, sw2, sw3, en_f1, en_f2, en_f3 έχουν την τιμή ένα. Αυτό σημαίνει ότι το σύστημα λειτουργεί κατά την ροή σχεδίασης του και είναι όλα ενεργοποιημένα. Ξεκινάμε με την είσοδο 5e19, η οποία καταγράφεται στο r1_out και στο r2_out μαζί με το bit εγκυρότητας (1_5e19). Από εκεί περνάει στο Low pass filter και βρίσκεται εκτός ορίου, επομένως στην έξοδο του lout περνάει η τιμή ορίου 7e (126). Στη συνέχεια η νέα τμή περνάει από τον πρώτο πολυπλέκτη και καταγράφεται στον τρίτο καταχωρητή r3_out. Στη συνέχεια και το δεύτερο φίλτρο είναι ενεργοποιημένο, έτσι το 7e βρίσκεται εντός των ορίων του και περνάει στην έξοδο του hout. Εκεί μέσο του δεύτερου πολυπλέκτη καταγράφεται στο r4_out. Εκεί περνάει στον Validation buffer όπου το sel του D Flip Flop του έχει την τιμή 1 (en_f3 = 1 and I=r4_out(16)=1) και

επομένως κρίνεται έγκυρο να περάσει στην έξοδο του. Τέλος φιλτράρεται από το φίλτρο κινούμενου μέσου όρου και βγαίνει στην έξοδο αυτού sout (x007) και αποθηκεύεται στον τελευταίο καταχωρητή. Μπορούμε να τα δούμε σημειωμένα με βελάκια στην παραπάνω εικόνα.

7 Συμπεράσματα

Στο κεφάλαιο αυτό θα παρουσιαστούν τα συμπεράσματα που προέκυψαν από τον σχεδιασμό και την υλοποίηση του συστήματος της πτυχιακής εργασίας. Συγκεκριμένα παρουσιάζονται τα κυριότερα συμπεράσματα που προέκυψαν από το σύστημα.

Αρχικά, μελετήθηκαν τα βιολογικά σήματα και ιδιαίτερα τα σήματα του ανθρώπινου εγκεφάλου και δόθηκε έμφαση στον συσχετισμό τους με την επιληψία, καθώς αποτελεί κομμάτι μελέτης της συγκεκριμένης πτυχιακής εργασίας. Παρατηρήθηκε ότι κατά την καταγραφή των εγκεφαλικών σημάτων δημιουργούνται διάφορα παράσιτα που μπορούν να αλλοιώσουν την χρήσιμή πληροφορία. Η πληροφορία που μας αφορά στην παρούσα πτυχιακή εργασία είναι μη φυσιολογικές κυματομορφές που ωστόσο δεν αποτελούν αποτέλεσμα θορύβου ή κάποιου παρασίτου αλλά σχετίζονται με την εμφάνιση επιληπτικών κρίσεων. Έτσι κρίθηκε απαραίτητη η δημιουργία ενός συστήματος που παρέχει την δυνατότητα επεξεργασίας τέτοιων σημάτων με αποτέλεσμα τον καθαρισμό τους από άχρηστες πληροφορίες. Γενικά, προσφέρεται ευελιξία ως προς την χρήση του συστήματος. Η βασική ιδέα μπορεί να προήλθε για χρήση της σε εγκεφαλικά σήματα, ωστόσο τα φίλτρα που την αποτελούν μπορούν να εφαρμοστούν σε κάθε είδος βιολογικών σημάτων και γενικότερων σημάτων. Το σύστημα δημιουργήθηκε στη γλώσσα VHDL και προσομοιώθηκε για την εξακρίβωση της σωστής λειτουργίας του. Διαπιστώθηκε λοιπόν ότι η λειτουργία του κάθε φίλτρου καθώς και ο συνδυασμός τους ανταποκρίνεται στον σχεδιασμό. Τα φίλτρα αυτά είναι ιδιαίτερα χρήσιμα για την απομάκρυνση τιμών που οφείλονται στον θόρυβο καθώς και την εξομάλυνση του σήματος. Με την εφαρμογή του βαθυπερατού φίλτρου παρατηρήθηκε απομάκρυνση υψηλών τιμών που μπορεί να οφείλονται σε θόρυβο. Με το υψιπερατό φίλτρο απομακρύνθηκαν οι χαμηλής κλίμακας τιμές. Το φίλτρο κινούμενου μέσου όρου μείωσε σημαντικά το εύρος των τιμών και διατήρησε αυτές που είναι σημαντικές για την διεξαγωγή αποτελεσμάτων. Συμπεραίνουμε λοιπόν ότι το σύστημα μπορεί να παρέχει μια καλή προ επεξεργασία των σημάτων εφαρμόζοντας μεμονωμένα τα φίλτρα ή συνδυάζοντας τα για διατήρηση λιγότερής και πιο ουσιώδης πληροφορίας. Τέλος καθώς το σύστημα δίνει δυνατότητα επιλογής των φίλτρων που θα εφαρμοστούν μέσω διακοπτών, ο χρήστης πολύ απλά μπορεί να το προσαρμόσει κάθε φορά στις ανάγκες και στις απαιτήσεις του, εφαρμόζοντας διαφορετικά είδη εισόδων.

ΠΑΡΑΡΤΗΜΑ Α

Κώδικας τελικού συστήματος

```
library IEEE;
use IEEE.std_logic_1164.all;
use ieee.numeric_std.all;

entity signal_filtering is
    port (clock,reset :in std_logic;
          sw1 : in std_logic;
          sw2 : in std_logic;
          sw3 : in std_logic;
          eisodos : in std_logic_vector(15 downto 0);
          eksodos : out std_logic_vector(15 downto 0));
end entity;

architecture behavioral of signal_filtering is


component registern is
        Port ( d : in std_logic_vector (15 downto 0);
               Clock : in std_logic;
               Reset : in std_logic;
               Q : out std_logic_vector (15 downto 0)
      );
end component ;

component register2 is
        Port ( d : in std_logic_vector (16 downto 0);
             Clock : in std_logic;
```

```
                    Reset : in std_logic;
                    Q : out std_logic_vector (16 downto 0));
end component ;

component module_fifo_regs_no_flags is
  generic (
    g_WIDTH : natural := 16;
    g_DEPTH : integer := 14
    );
  port (
    i_rst_sync : in std_logic;
    i_clk      : in std_logic;

    -- FIFO Write Interface
    i_wr_en   : in  std_logic;
    i_wr_data : in  std_logic_vector(g_WIDTH-1 downto 0);
    o_full    : out std_logic;

    -- FIFO Read Interface
    i_rd_en   : in  std_logic;
    o_rd_data : out std_logic_vector(g_WIDTH-1 downto 0);
    o_empty   : out std_logic
    );
end component;

component mux_2to1_top is
    Port ( SEL : in  STD_LOGIC;
           A   : in  STD_LOGIC_VECTOR (15 downto 0);
           B   : in  STD_LOGIC_VECTOR (15 downto 0);
           X   : out STD_LOGIC_VECTOR (15 downto 0));
end component;
```

```
component moving_average_filter is
   Port (sin: in std_logic_vector (15 downto 0);
         clk: in  std_logic;
         en: in std_logic;
         sout: out std_logic_vector (15 downto 0);
         rst: in  std_logic);
end component moving_average_filter;

component lowpass_filter is
    Port( lin : in std_logic_vector( 15 downto 0);
          clk : in std_logic;
          en: in std_logic;
         lout: out std_logic_vector(15 downto 0);
         rst: in std_logic
    );
end component lowpass_filter;

component highpass_filter is
    Port( hin : in std_logic_vector(15 downto 0);
              clk :in std_logic;
              en: in std_logic;
              hout: out std_logic_vector(15 downto 0);
              rst : in std_logic
    );
end component highpass_filter;

component mov_buf is
            Port ( E : in std_logic_vector(15 downto 0);
```

```
                    EN : in std_logic;
                    I  : in  std_logic;
                    CLK : in std_logic;
                    RST : in std_logic;
                    O : out std_logic_vector(15 downto 0)
            );
end component;

signal en_f1,en_f2,en_f3 : std_logic;
signal r1_out,b_out,s1,s2,s3 : std_logic_vector(15 downto 0);
signal   l_out,m1_out,h_out,m2_out,m3_out,s_out,f_out,r5_out:   std_logic_vector(
15 downto 0);
signal o_full: std_logic; --
signal r2_out ,r3_out,r4_out,sig_eis,m1,m2 : std_logic_vector(16 downto 0);

begin
process (clock,reset)
begin
if (sw1='1') then    en_f1 <='1';
else en_f1 <='0';
end if;
if (sw2='1') then en_f2 <='1';
else en_f2 <='0';
end if;
if (sw3='1') then en_f3 <='1';
else en_f3 <='0';
end if;

end process;
reg1 : registern port map ( d => eisodos,
                            Clock => clock,
```

```
                                        Reset => reset,
                                        Q => r1_out
                    );

bf : module_fifo_regs_no_flags port map ( i_rst_sync => reset,
                                          i_clk =>clock,
                                          i_wr_en =>s1,
                                          i_wr_data => r1_out ,
                                          o_full => o_full,
                                          i_rd_en => s2,
                                          o_rd_data =>b_out,
                                          o_empty => o_empty);

sig_eis <=   (o_full & b_out);

reg2: register2 port map (d => sig_eis,
                          Clock => clock,
                          Reset => reset,
                          Q => r2_out);
s1 (15 downto 0) <= r2_out (15 downto 0);
lp: lowpass_filter port map (lin =>s1 ,
                             clk =>clock,
                             lout => l_out,
                             en => en_f1,
                             rst => reset);

mx1 : mux_2to1_top port map (SEL => en_f1,
                             A => l_out,
                             B => s1,
                             X => m1_out);
```

```
m1 <= (r2_out(16) & m1_out);

reg3 : register2 port map(d =>m1,
                          Clock => clock,
                          Reset => reset,
                          Q => r3_out);

s2(15 downto 0) <= r3_out(15 downto 0);

hp : highpass_filter port map (hin =>s2,
                          clk =>clock,
                          hout =>h_out,
                          en => en_f2,
                          rst =>reset);

mx2 : mux_2to1_top port map (SEL => en_f2,
                    A => h_out,
                    B =>s2,
                    X => m2_out);

m2 <= (r3_out(16) & m2_out);

reg4 : register2 port map( d =>m2,
                           Clock => clock,
                           Reset => reset,
                           Q => r4_out);
s3(15 downto 0) <= r4_out(15 downto 0);

buf: mov_buf port map (  E => s3,
                          EN => en_f3,
                          I  => r4_out(16),
```

```
                    CLK => clock,
                    RST =>reset,
                    O =>f_out);

ma : moving_average_filter port map (sin =>f_out,
                                     clk =>clock,
                                     en => en_f3,
                                     sout =>s_out,
                                     rst =>reset);

mx3 : mux_2to1_top port map (SEL => en_f3,
                             A => s_out,
                             B => s3,
                             X => m3_out);




reg5 : registern port map( Clock => clock,
                           d =>m3_out,
                           Reset => reset,
                           Q => r5_out
                    );


eksodos <= r5_out;

end architecture;
```

ΠΑΡΑΡΤΗΜΑ Β

Κώδικες υποσυστημάτων τελικού συστήματος

Registers

```
-- Code your design here
library IEEE;
use IEEE.std_logic_1164.all;


entity registern is
                Port ( d : in std_logic_vector(15 downto 0);
                    Clock : in std_logic;
                        Reset : in std_logic;
                        Q : out std_logic_vector(15 downto 0)
            );
end registern;


architecture Behavioral of registern is
begin
 process(Clock,Reset)
begin
  if (Reset='0') then
    Q <=x"0000";
      elsif (Clock'event and Clock= '1') then
                Q <= d;
      end if;
 end process;
```

```
end architecture;


-- Code your design here
library IEEE;
use IEEE.std_logic_1164.all;


entity register2 is
                Port ( d : in std_logic_vector(16 downto 0);
                    Clock : in std_logic;
                        Reset : in std_logic;
                        Q : out std_logic_vector(16 downto 0));
end register2;


architecture Behavioral of register2 is
begin
 process(Clock,Reset)
begin
  if (Reset='0') then
      Q <="00000000000000000";
       elsif (Clock'event and Clock= '1') then


                Q <= d;
   end if;


end process;
end architecture;
```

FIFO buffer

```
library IEEE;
use IEEE.std_logic_1164.all;
use ieee.numeric_std.all;

entity module_fifo_regs_no_flags is
  generic (
    g_WIDTH : natural := 16;
    g_DEPTH : integer := 14
    );
  port (
    i_rst_sync : in std_logic;
    i_clk      : in std_logic;

    -- FIFO Write Interface
    i_wr_en   : in  std_logic;
    i_wr_data : in  std_logic_vector(g_WIDTH-1 downto 0);
    o_full    : out std_logic;

    -- FIFO Read Interface
    i_rd_en   : in  std_logic;
    o_rd_data : out std_logic_vector(g_WIDTH-1 downto 0);
    o_empty   : out std_logic
    );
end module_fifo_regs_no_flags;
```

```
architecture rtl of module_fifo_regs_no_flags is

  type t_FIFO_DATA is array (0 to g_DEPTH-1) of std_logic_vector(g_WIDTH-1
downto 0);
  signal r_FIFO_DATA : t_FIFO_DATA := (others => (others => '0'));

  signal r_WR_INDEX   : integer range 0 to g_DEPTH-1 := 0;
  signal r_RD_INDEX   : integer range 0 to g_DEPTH-1 := 0;

  -- # Words in FIFO, has extra range to allow for assert conditions
  signal r_FIFO_COUNT : integer range -1 to g_DEPTH+1 := 0;

  signal w_FULL  : std_logic;
  signal w_EMPTY : std_logic;

begin

  p_CONTROL : process (i_clk) is
  begin
    if rising_edge(i_clk) then
      if i_rst_sync = '1' then
        r_FIFO_COUNT <= 0;
        r_WR_INDEX   <= 0;
        r_RD_INDEX   <= 0;
      else
```

```
-- Keeps track of the total number of words in the FIFO
if (i_wr_en = '1' and i_rd_en = '0') then
  r_FIFO_COUNT <= r_FIFO_COUNT + 1;
elsif (i_wr_en = '0' and i_rd_en = '1') then
  r_FIFO_COUNT <= r_FIFO_COUNT - 1;
end if;

-- Keeps track of the write index (and controls roll-over)
if (i_wr_en = '1' and w_FULL = '0') then
  if r_WR_INDEX = g_DEPTH-1 then
    r_WR_INDEX <= 0;
  else
    r_WR_INDEX <= r_WR_INDEX + 1;
  end if;
end if;


-- Keeps track of the read index (and controls roll-over)
if (i_rd_en = '1' and w_EMPTY = '0') then
  if r_RD_INDEX = g_DEPTH-1 then
    r_RD_INDEX <= 0;
  else
    r_RD_INDEX <= r_RD_INDEX + 1;
  end if;
end if;
```

```
      -- Registers the input data when there is a write
      if i_wr_en = '1' then
        r_FIFO_DATA(r_WR_INDEX) <= i_wr_data;
      end if;

    end if;                    -- sync reset
  end if;                      -- rising_edge(i_clk)
end process p_CONTROL;

o_rd_data <= r_FIFO_DATA(r_RD_INDEX);

w_FULL  <= '1' when r_FIFO_COUNT = g_DEPTH else '0';
w_EMPTY <= '1' when r_FIFO_COUNT = 0       else '0';

o_full  <= w_FULL;
o_empty <= w_EMPTY;

-- ASSERTION LOGIC - Not synthesized
-- synthesis translate_off

p_ASSERT : process (i_clk) is
begin
  if rising_edge(i_clk) then
    if i_wr_en = '1' and w_FULL = '1' then
report "ASSERT FAILURE - MODULE_REGISTER_FIFO: FIFO IS FULL AND
BEING WRITTEN " severity failure;
```

```
    end if;


    if i_rd_en = '1' and w_EMPTY = '1' then
report "ASSERT FAILURE - MODULE_REGISTER_FIFO: FIFO IS EMPTY AND BEING READ " severity failure;
    end if;
  end if;
 end process p_ASSERT;


 -- synthesis translate_on
end rtl;
```

Mux

```
-- Code your design here
library IEEE;
use IEEE.std_logic_1164.all;
entity  mux_2to1_top is
   Port ( SEL : in  STD_LOGIC;
          A   : in  STD_LOGIC_VECTOR (15 downto 0);
          B   : in  STD_LOGIC_VECTOR (15 downto 0);
          X   : out STD_LOGIC_VECTOR (15 downto 0));

end mux_2to1_top;

architecture Behavioral of mux_2to1_top is
begin
```

```
   X <= A when (SEL = '1') else B;
end Behavioral;
```

Low_pass Filter

```
-- Code your design here
library IEEE;
use IEEE.std_logic_1164.all;
entity lowpass_filter is
        Port( lin : in std_logic_vector(15 downto 0);
                clk :in std_logic;
       en:in std_logic;
                lout : out std_logic_vector(15 downto 0);
                rst : in std_logic
        );
end entity lowpass_filter;

architecture behavioral of lowpass_filter is
begin
process(clk,rst)
begin
if (en='1') then
if (lin < "0000000001111110")then
lout <= lin ;
else
lout <="0000000001111110"; --don't care
end if;
```

```
else lout <="0000000000000000";
end if;
end process;
end;
```

High pass filter

```
-- Code your design here
library IEEE;
use IEEE.std_logic_1164.all;
entity highpass_filter is
        Port( hin : in std_logic_vector(15 downto 0);
                clk :in std_logic;
      en:in std_logic;
                hout : out std_logic_vector(15 downto 0);
                rst : in std_logic
        );
end entity highpass_filter;

architecture behavioral of highpass_filter is
begin
process(clk,rst)
begin
if( en='1') then
if (hin < "0000000100000000")then
hout <= hin ;
else
hout <="0000000100000000";
```

```
end if;
else hout<="0000000000000000";
end if;
end process;
end;
```

validation buffer

```
-- Code your design here
library IEEE;
use IEEE.std_logic_1164.all;
entity mov_buf is
                Port ( E : in std_logic_vector(15 downto 0);
                    EN : in std_logic;
                        I  : in  std_logic;
                        CLK : in std_logic;
                        RST : in std_logic;
                        O : out std_logic_vector(15 downto 0)
                );
end mov_buf;
architecture behavioral of mov_buf is

component d_ff_ns port ( D : in std_logic_vector(15 downto 0);
                S : in std_logic;
                CLK : in std_logic;
                Q : out std_logic_vector(15 downto 0));
end component;
```

```
signal s : std_logic;
--signal CLK: std_logic;
signal q : std_logic_vector(15 downto 0);
begin
s <= (EN and I);
D1: d_ff_ns port map ( D => E,
                       S => s,
                       CLK => CLK,
                       Q => q);

O <= q;
end architecture;
```

D Flip-Flop

```
library ieee;
use ieee.std_logic_1164.all;
entity d_ff_ns is
port ( D : in std_logic_vector(15 downto 0);
S : in std_logic;
CLK : in std_logic;
Q : out std_logic_vector(15 downto 0));
end entity;
-- architecture
architecture my_d_ff_ns of d_ff_ns is
begin
```

```
dff: process (CLK)
begin
if (rising_edge(CLK)) then
if (S = '0') then
Q <= "0000000000000000";
else
Q <= D;
end if;
end if;
end process dff;
end my_d_ff_ns;
```

Moving average filter

```
library ieee;
use ieee.std_logic_1164.all;

entity moving_average_filter is
   Port ( sin  : in  std_logic_vector (15 downto 0);
          clk  : in  std_logic;
          en:in std_logic;
          sout : out std_logic_vector (15 downto 0);
          rst  : in  std_logic);
end moving_average_filter;

architecture behavioral of moving_average_filter is
   -- Registers
```

```
    type tregisters is array (0 to 20) of std_logic_vector(19 downto 0);
    signal registers : tregisters;
    signal sum : std_logic_vector(19 downto 0);

    -- Wires
    signal addition             : std_logic_vector(19 downto 0);
    signal addition_carries     : std_logic_vector(19 downto 0);
    signal subtraction          : std_logic_vector(19 downto 0);
    signal subtraction_carries  : std_logic_vector(19 downto 0);
    signal first                : std_logic_vector(19 downto 0);
    signal last                 : std_logic_vector(19 downto 0);

    component carry_lookahead_adder is
        Port ( a : in  std_logic_vector (19 downto 0);
               b : in  std_logic_vector (19 downto 0);
               ci : in  std_logic;
               s : out  std_logic_vector (19 downto 0);
               co : out  std_logic);
    end component;
begin
    process (sin, clk)
    begin
    if(en ='1') then
        if rst = '1' then
            -- Reset register
            for i in 0 to 20 loop
```

```
            registers(i) <= "00000000000000000000";
        end loop;
        sum <= "00000000000000000000";
    elsif rising_edge(clk) then
        -- Shift operands
        for i in 20 downto 1 loop
            registers(i) <= registers(i-1);
        end loop;
        -- Store first operand
        registers(0) <= first;

        -- Store sumious value
        sum <= subtraction;
    end if;
    end if;
end process;

-- Sign extension
sign_extension_loop: for i in 19 downto 11 generate
    first(i) <= sin(10);
end generate;

-- Connect lower 11-bits
input_loop: for i in 15 downto 0 generate
    first(i) <= sin(i);
end generate;
```

```
    -- Last operand
    last <= not (registers(20));

    -- Add first operand
    addition_stage: carry_lookahead_adder port map(a=>first, b=>sum, ci=>'0',
s=>addition);

    -- Subtract last operand
    subtraction_stage: carry_lookahead_adder port map(a=>addition, b=>last, ci=>'1',
s=>subtraction);

    -- Write output
    division_loop: for i in 15 downto 0 generate
        sout(i) <= subtraction(i+4);
    end generate;

end Behavioral;
```

Carry look ahead adder

```
library ieee;
use ieee.std_logic_1164.all;

-- 20-bit carry look-ahead adder
entity carry_lookahead_adder is
    port (
        a  : in  std_logic_vector (19 downto 0);
```

```
        b  : in  std_logic_vector (19 downto 0);
        ci : in  std_logic;
        s  : out std_logic_vector (19 downto 0);
        co : out std_logic
    );
end carry_lookahead_adder;

architecture behavioral of carry_lookahead_adder is
    signal t : std_logic_vector(19 DOWNTO 0);
    signal g : std_logic_vector(19 DOWNTO 0);
    signal p : std_logic_vector(19 DOWNTO 0);
    signal c : std_logic_vector(19 DOWNTO 1);
begin
    -- Product stage
    g <= a and b;
    p <= a or b;

    -- Sum stage
    t <= a xor b;

    -- Carry stage
    c(1) <= g(0) or (p(0) and ci);
    carry_loop: for i in 1 to 14 generate
        c(i+1) <= g(i) or (p(i) and c(i));
    end generate;
```

```
    co <= g(15) or (p(15) and c(15));
    s(0) <= t(0) xor ci;
    s(15 downto 1) <= t(15       downto 1) xor c(15 downto 1);
end behavioral;
```

Βιβλιογραφία

[1] [Ηλεκτρονικό]. Khald Ali I.Aboalayon,Miad Faezipour,Wafaa S. Almuhammadi and Saeid Moslehpour «Sleep Stage Classification Using EEG Signal Analysis: A Comprehensive Survey and New Investigation» Available: https://www.mdpi.com/1099-4300/18/9/272. [Πρόσβαση 20 7 2019].

[2] Άρτεμης Ζωγράφου «Ανάλυση EEG και EOG σημάτων με ευφυείς τεχνικές για τηλεκίνηση RC αμαξιδίου» [Ηλεκτρονικό]. Available: http://artemis.cslab.ece.ntua.gr:8080/jspui/handle/123456789/13749. [Πρόσβαση 20 7 2019].

[3] Τσιπούρας Μάρκος, Γιαννακέας Νικόλαος, Καρβούνης Ευάγγελος και Τζάλλας Αλέξανδρος Βιοϊατρικά Σήματα.

[4] Mohd. Maroof Siddiqui, Saifur Rahman, Syed Hasan Saeed , Anurag Banodia «EEG Signals Play Major Role to diagnose Sleep Disorder» [Ηλεκτρονικό]. Available: www.ijecse.org. [Πρόσβαση 20 7 2019].

[5] Sudhansu Chokroverty, Sleep Disorders Medicine- Basic Science, Technical, Considerations and Clinic Aspects.

[6] Bharti Kamlesh Kanoje, Ashwini S Shingare « Automatic Sleep Stage Detection of an EEG Signal Using an Ensemble Method» [Ηλεκτρονικό]. Available: https://pdfs.semanticscholar.org/486f/e58c8883a5a7c918ab3a9d2f39014d8828b7.pdf. [Πρόσβαση 20 7 2019]

[7] P.Thomas, A. Arzimanoglou «Επιληψίες»]

[8] [Ηλεκτρονικό]. Available: https://biomig.ntua.gr/courses.html. [Πρόσβαση 23 7 2019].

[9] [Ηλεκτρονικό]. Available: http://www.pynq.io/board [Πρόσβαση 23 7 2019].

[10] Implementing a Low-Pass Filter on FPGA with Verilog [Ηλεκτρονικό]. Available: https://www.allaboutcircuits.com/technical-articles/implementing-a-low-pass-filter-on-fpga-with-verilog/. [Πρόσβαση 24 7 2019].

[11] [Ηλεκτρονικό]. Available: https://www.elprocus.com/what-is-low-pass-filter-lpf-using-op-amp-applications/. [Πρόσβαση 24 7 2019].

[12] «Wikipedia,» [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/Low-pass_filter. [Πρόσβαση 24 7 2019].

[13] «Wikipedia,» [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/High-pass_filter. [Πρόσβαση 24 7 2019]

[14] [Ηλεκτρονικό]. Available: https://wirelesspi.com/moving-average-filter/. [Πρόσβαση 24 7 2019].

[15] [Ηλεκτρονικό]. Available: https://newonlinecourses.science.psu.edu/stat510/lesson/2/2.1. [Πρόσβαση 24 7 2019].

[16] [Ηλεκτρονικό]. Available: https://tomroelandts.com/articles/the-moving-average-as-a-filter. [Πρόσβαση 24 7 2019].

[17] «Wikipedia,» [Ηλεκτρονικό]. Available: https://el.wikipedia.org/wiki/%CE%9A%CE%B1%CF%84%CE%B1%CF%87%CF%89%CF%81%CE%B7%CF%84%CE%AE%CF%82. [Πρόσβαση 2 8 2019].

[18] [Ηλεκτρονικό]. Available: https://www.xilinx.com/support/documentation/university/Vivado-Teaching/HDL-Design/2015x/VHDL/docs-pdf/lab6.pdf. [Πρόσβαση 2 8 2019]

[19] [Ηλεκτρονικό]. Available: http://www.deathbylogic.com/2013/07/vhdl-standard-fifo/. [Πρόσβαση 3 8 2019].

[20] [Ηλεκτρονικό]. Available: https://surf-vhdl.com/what-is-a-fifo/. [Πρόσβαση 3 8 2019].

[21] [Ηλεκτρονικό]. Available: https://lauri.xn--vsandi-pxa.com/hdl/arithmetic/moving-average-filter.html. [Πρόσβαση 5 8 2019].

[22] [Ηλεκτρονικό]. Available: https://el.wikipedia.org/wiki/VHDL. [Πρόσβαση 10 8 2019].

Πηγές Εικόνων

Εικόνα 1: Εγκεφαλικού ρυθμοί:

http://apothetirio.teiep.gr/xmlui/bitstream/handle/123456789/10145/%20%20%20%20%20%20%20%20.pdf?sequence=1

Εικόνα 2: Κυματομορφές σταδίων ύπνου:

https://robertlovespi.net/tag/reprogramming/

Εικόνα 3: Νευρώνας με ένα δυναμικό ενέργειας και "επιληπτικός" υπερευρέθιστος νευρώνας :

https://oup-arc.com/access/content/psychopharmacology-3e-student-resources/psychopharmacology-3e-web-box-2-1-clinical-applications-epilepsy

Εικόνα 4: Φυσιολογική, επιληπτική και μη επιληπτική κυματομορφή:

https://www.researchgate.net/figure/Waveform-of-three-EEG-signals-normal-set-A-seizure-free-set-D-and-epileptic-seizure_fig5_270523871

Εικόνα 5: κλοήθη επιληπτόμορφα μεταβατικά ευρήματα ύπνου:

https://docplayer.gr/51575517-Synithi-iegrafika-sfalmata-sti-diagnosi-tis-epilipsias.html

Εικόνα 6: Επιληψία- υπνηλία αιχμηρά κύματα στις κεντρικές περιοχές (C3 και C4) με επικράτηση δεξιά:

https://docplayer.gr/51575517-Synithi-iegrafika-sfalmata-sti-diagnosi-tis-epilipsias.html

Εικόνες 7: Οφθαλμικές κινήσεις- παράσιτα:

https://docplayer.gr/51575517-Synithi-iegrafika-sfalmata-sti-diagnosi-tis-epilipsias.html

Εικόνα 8: ΗΚΓ-παράσιτα:

https://docplayer.gr/51575517-Synithi-iegrafika-sfalmata-sti-diagnosi-tis-epilipsias.html

Εικόνα 9: μυϊκό παράσιτο:

https://docplayer.gr/51575517-Synithi-iegrafika-sfalmata-sti-diagnosi-tis-epilipsias.html

Εικόνα 10: παράσιτα από ηλεκτρόδια:

https://docplayer.gr/51575517-Synithi-iegrafika-sfalmata-sti-diagnosi-tis-epilipsias.html

Εικόνα 11: πλακέτα PYNQ-z2:

http://www.pynq.io/board

Εικόνα 12: *Ααπλουστευμένο σχήμα ιδανικού βαθυπερατού φίλτρου*:

https://www.researchgate.net/figure/The-four-common-filters-a-Low-pass-filter-passes-signals-with-a-frequency-lower-than_fig3_315801831

Εικόνα 13: Σχήμα υψιπερατού φίλτρου:

https://www.colinhemphill.com/files/live-sound-reinforcement-module-01-church-volunteers.pdf

Εικόνα 14: σχήμα φίλτρου κινούμενου μέσου:

https://zipcpu.com/dsp/2017/10/16/boxcar.html

Εικόνα 15: Σχεδιάγραμμα τελικού συστήματος:

https://www.draw.io/#G11Wy4uD71PHsvE9KO0xyYkrrtjuYZR0MT

Εικόνα 16: επιλογή φίλτρων μέσο διακοπτών:

https://www.draw.io/#G11Wy4uD71PHsvE9KO0xyYkrrtjuYZR0MT

Εικόνα 17: FIFO buffer του συστήματος:

https://www.draw.io/#G11Wy4uD71PHsvE9KO0xyYkrrtjuYZR0MT

Εικόνα 18: *πολυπλέκτης που επιλέγει ανάμεσα στις φιλτραρισμένες τιμές και τις αρχικές*:

https://www.draw.io/#G11Wy4uD71PHsvE9KO0xyYkrrtjuYZR0MT

Εικόνα 19: περιγραφή του validation buffer:

https://www.draw.io/#G11Wy4uD71PHsvE9KO0xyYkrrtjuYZR0MT

Εικόνα 20: σχεδίαση φίλτρου κινουμένου μέσου όρου:

https://www.draw.io/#G11Wy4uD71PHsvE9KO0xyYkrrtjuYZR0MT

Εικόνα 21: Κυματομορφή προσομοίωσης 16_bit καταχωρητή:

https://www.edaplayground.com/x/4yss

Εικόνα 22 Κυματομορφή προσομοίωση 17_bit καταχωρητή:

https://www.edaplayground.com/x/4yss

Εικόνα 23: Κυματομορφή προσομοίωσης FIFO buffer:

https://www.edaplayground.com/x/4yss

Εικόνα 24: Κυματομορφή προσομοίωσης πολυπλέκτη:

https://www.edaplayground.com/x/4yss

Εικόνα 25: Κυματομορφή προσομοίωσης βαθυπερατού φίλτρου:

https://www.edaplayground.com/x/4yss

Εικόνα 26: Κυματομορφή προσομοίωσης υψιπερατού φίλτρου:

https://www.edaplayground.com/x/4yss

Εικόνα 27: Κυματομορφή προσομοίωσης validation buffer:

https://www.edaplayground.com/x/4yss

Εικόνα 28: Κυματομορφή προσομοίωσης φίλτρου κινούμενου μέσου όρου:

https://www.edaplayground.com/x/4yss

Εικόνα 29: Κυματομορφή προσομοίωσης τελικού συστήματος – επιλογή πρώτου φίλτρου:

https://www.edaplayground.com/x/4yss

Εικόνα 30: Κυματομορφή προσομοίωσης τελικού συστήματος – επιλογή δεύτερου φίλτρου:

https://www.edaplayground.com/x/4yss

Εικόνα 31: Κυματομορφή προσομοίωσης τελικού συστήματος – επιλογή τρίτου φίλτρου:

https://www.edaplayground.com/x/4yss

Εικόνα 32: Κυματομορφή προσομοίωσης τελικού συστήματος – επιλογή πρώτου και δεύτερου φίλτρου:

https://www.edaplayground.com/x/4yss

Εικόνα 33: Κυματομορφή προσομοίωσης τελικού συστήματος – επιλογή πρώτου και τρίτου φίλτρου:

https://www.edaplayground.com/x/4yss

Εικόνα 34: Κυματομορφή προσομοίωσης τελικού συστήματος – επιλογή δεύτερου και τρίτου φίλτρου:

https://www.edaplayground.com/x/4yss

Εικόνα 29: Κυματομορφή προσομοίωσης τελικού συστήματος – επιλογή και των τριών φίλτρων:

https://www.edaplayground.com/x/4yss

Οι εικόνες 15-20 αποτελούν προσωπικά δικαιώματα και έχουν δημιουργηθεί με την βοήθεια του draw.io.

Οι εικόνες 21-35 αποτελούν προσωπικά δικαιώματα και έχουν δημιουργηθεί με την βοήθεια του edaplayground.