Εθνικο Μετσοβιο Πολυτεχνειο

Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων

Τομεας Τεχνολογιας Πληροφορικης και Υπολογιστων

Αξιολόγησης της απόδοσης και της κατανάλωσης ενεργειας μίας εφαρμογής ανάλυσης Ηλεκτροκαρδιογραφήματος στη **Myriad 2**

Διπλωματικη Εργασια

του

καραβαλακη νικολαου

Επιβλέπων: Σούντρης Δημήτριος
Αναπληρωτής Καθηγητής Ε.Μ.Π.

Εργαστηριο Μικροϋπολογιστων και Ψηφιακων Συστηματων

Αθήνα, Ιούλιος 2017

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών
Εργαστήριο Μικρουπολογιστών και Ψηφιακών Συστημάτων

Power Consumption and Performance Evaluation of ECG Analysis Flow on Myriad 2 Multicore System-on-Chip

Διπλωματικη Εργασια

του

ΚΑΡΑΒΑΛΑΚΗ ΝΙΚΟΛΑΟΥ

Επιβλέπων: Σούντρης Δημήτριος
Αναπληρωτής Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 3η Ιουλίου 2017.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Σούντρης Δημήτριος	Πεσκμεντζη Κιαμάλ	Γκούμας Γεώργιος
Αναπληρωτής Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Επίκουρος Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2017

(Υπογραφή)

.....

Καραβαλακης Νικολαος

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών
Εργαστήριο Μικρουπολογιστών και Ψηφιακών Συστημάτων

Acknowledgements

The current thesis is the result of my work in collaboration with the Microprocessors and Digital Systems Laboratory (MicroLab) of NTUA. I would like to thank my supervisor, Prof. Dimitrios Soudris for the trust he showed in me and for his guidance and encouragement throughout the conduction of the thesis. The educational opportunities that he offered me undoubtedly helped me evolve both on a professional and personal level. I would also like to sincerely thank Doctoral Researcher for his contribution and precious guidance. His insightful comments and constructive criticism were determinative for the completion of my work. I am also grateful to Doctoral Student Konstantina Koliogewrgi and student Dimitra Azariadi for algorithm they made. Both their work has been a true inspiration for me. I would also like to thank all the members of the laboratory with whom I have interacted during the course of my thesis conduction. Their friendliness and sincere will to help have made these last months one of the fondest memories I have from my studies in NTUA. I would also like to thank my fellow students and especially my closest friends for being an integral part of my life during these last five years. Finally I would like to wholeheartedly thank my family for their love and support throughout all the challenges of my life.

Abstract

One of the most essential biological signals for the diagnosis of the heart is the Electrocardiogram (ECG). The need of constant monitoring and on-time heart condition assessment have imposed new requirements for acceleration and power consumption of ECG Analysis Flow. Due to complexity of assessing and predicting heart's condition, machine learning techniques have become dominant on the field of ECG analysis.

Support Vector Machine classifiers is the most efficient way to predict accurately the heart's condition. Based on multiple computational operations, SVM classifiers lead to excessive power consumption and high execution time. The approach of this thesis is to implement and optimize the algorithm in respect of performance and power consumption. Working towards meeting these specifications, the filter and the Support Vector Machine classifier, the first and the last part of the ECG Analysis Flow respectively, were accelerated, as the most time consuming and power demanding parts of the flow. In this thesis, the ECG Analysis Flow was implemented into an ultra-low-power multicore System-on-Chip Myriad 2, designed for high computational tasks for mobile, wearable and embedded applications, provided by Movidius. Firstly, the original filter and classifier codes are modified, in order to utilize the micro-architectural features and the memory hierarchy of Myriad 2. Afterwards, the computational tasks are delivered to the VLIW micro-processors of Myriad 2, where they are executed in parallel for higher performance efficiency.

Finally, our implementation proved that Myriad 2 can achieve up to 97% and 99% latency gain compared to the original filter code and SVM code respectively, coming with significant energy efficiency.

Keywords

Medical embedded system design, ECG analysis, machine learning, Support Vector Machines, Movidius Myriad 2, Very Long Instruction Word operations, Vectorization

Περίληψη

Ένα από τα πιο σημαντικά και βασικά βιολογικά σήματα είναι το Ηλεκτροκαρδιογράφημα (ΗΚΓ). Η ανάγκη για συνεχή παρακολούθηση και έγκαιρη διάγνωση της κατάστασης της καρδιάς έχει αυξήσει τις απαιτήσεις για επιτάχυνση και μειωμένη κατανάλωση ισχύος στην Ροή Ανάλυσης Ηλεκτροκαρδιογραφήματος. Εξαιτίας της πολυπλοκότητας για την ανάκτηση στοίχειων για την κατάσταση της καρδιάς, καθώς και η πρόβλεψη τους, οι τεχνικές μηχανικής μάθησης έχουν καταστεί κυρίαρχες στο πεδίο της ανάλυσης Ηλεκτροκαρδιογραφήματος.

Οι ταξινομητές διανυσμάτων υποστήριξης είναι ο πιο αποτελεσματικός τρόπος για την ασφαλή πρόβλεψη της κατάστασης της καρδιάς. Βασιζόμενοι σε πολλαπλές υπολογιστικές διαδικασίες, οι ταξινομητές αυτή οδηγούν σε πολύ υψηλό χρόνο εκτέλεσης, καθώς και υψηλή κατανάλωση ενέργειας. Για το λόγο αυτό, ο σκοπός αυτής της διπλωματικής είναι η υλοποίηση και βελτιστοποίηση του αλγορίθμου ανάλυσης ΗΚΓ από την άποψη του χρόνου και της κατανάλωσης ισχύος. Δουλεύοντας προς αυτή την κατεύθυνση, επιταχύναμε το φίλτρο, καθώς και τον ταξινομητή διανυσμάτων υποστήριξης, το πρώτο και το τελευταίο κομμάτι του αλγορίθμου αντίστοιχα, καθώς ήταν τα δύο πιο απαιτητικά κομμάτια του κώδικα. Στην διπλωματική αυτή, η Ροή Ανάλυσης ΗΚΓ υλοποιήθηκε σε ένα πολυπύρηνο επεξεργαστή, πολύ χαμηλής κατανάλωσης το Myriad 2 της Movidius, ο οποίος έχει σχεδιαστεί για πολύ υπολογιστικές διαδικασίες για φορητές, φορετές και ενσωματωμένες συσκευές. Αρχικά, το φίλτρο και ο ταξινομητής τροποποιήθηκαν, έτσι ώστε να χρησιμοποιήσουμε τα μικρο-αρχιτεκτονικά χαρακτηριστικά και την ιεραρχία μνήμης της Myriad 2. Στην συνέχεια, τα υπολογιστικά μέρη τα περάσαμε στους υποπυρήνες της Myriad 2, όπου εκεί εκτελούνται παράλληλα για καλύτερη απόδοση.

Τελικά, η υλοποίηση μας δείχνει ότι η Myriad 2 μπορεί να πετύχει μέχρι και 97% και 99% κέρδος σε χρόνο συγκριτικά με τους αρχικούς χρόνους του φίλτρου και του ταξινομητή, αντίστοιχα, ενώ ταυτόχρονα έχουμε πολύ σημαντικά κέρδη σε κατανάλωση ενέργειας.

Λέξεις Κλειδιά

Σχεδιασμός Ιατρικών Ενσωματωμένων Συστημάτων, Ανάλυση Ηλεκτροκαρδιογραφήματος, Τεχνικές Μηχανικής Μάθησης, Πολυπύρηνοι επεξεργαστές, Movidius Myriad 2

Εκτεταμένη Περίληψη

ΠΕΡΙΕΧΟΜΕΝΑ

ΕΙΣΑΓΩΓΗ

Οι καρδιαγγειακή νόσος είναι πλέον η επικρατούσα αιτία θνησιμότητας παγκοσμίως [1]. Οι ασθένειες που σχετίζονται με το καρδιαγγειακό σύστημα προσβάλλουν κυρίως την καρδιά ή τα αιμοφόρα αγγεία και περιλαμβάνουν την καρδιακή προσβολή, την υπερτασική καρδιακή νόσο, τη φλεβική θρόμβωση και άλλες καρδιακές παθήσεις. Σύμφωνα με τον Παγκόσμιο Οργανισμό Υγείας, ο αριθμός των θανάτων λόγω καρδιαγγειακής νόσου έχει αυξηθεί δραματικά, περίπου 41% μέσα στα τελευταία 30 χρόνια. Λόγω της επικρατούσας κατάστασης, η ακριβής και έγκυρη διάγνωση κρίνεται ως διαδικασία μέγιστης σημασίας.

Ένα από τα πιο σημαντικά βιολογικά σήματα για την παρακολούθηση και διάγνωση της κατάστασης της καρδιάς είναι το ηλεκτροκαρδιογράφημα (ΗΚΓ). Το ΗΚΓ είναι στενά συνδεδεμένο με τη μορφολογία και τη φυσιολογία της καρδιάς. Ως εκ τούτου, το ηλεκτροκαρδιογράφημα έρευνας (ΗΚΓ) έχει χρησιμοποιηθεί ευρέως για τη διάγνωση πολλών καρδιακών παθήσεων. Για την αξιολόγηση και την εξαγωγή, ταχύτερα και ακριβέστερα, πληροφορίας από το ΗΚΓ, είναι απαραίτητη και μπορεί να επιτευχθεί με τη συμβολή της τεχνολογίας. Τα τελευταία χρόνια, πολλές ηλεκτρονικές φορητές συσκευές έχουν κατασκευαστεί για ιατρικούς σκοπούς. Αυτές οι συσκευές είναι σε θέση να παρακολουθούν και να καταγράφουν την καρδιακή κατάσταση σε ημερήσια βάση. Ως αποτέλεσμα, είναι ευκολότερο και ταχύτερο για τον γιατρό να εξετάσει τον ασθενή, ενώ έχει βελτιωθεί επίσης η ποιότητα ζωής του ασθενούς, ελαχιστοποιώντας το κόστος της υγειονομικής περίθαλψης, μειώνοντας την ανάγκη για νοσηλεία και αποφεύγοντας πολλές ώρες αναμονής στις ουρές στο νοσοκομείο. Επιπλέον, ο κίνδυνος θανάτου, εξαιτίας τέτοιων ασθενειών, έχει εκμηδενιστεί, εξαιτίας της από-απόστασης έγκαιρης διάγνωσης, ακόμα και όταν ο ασθενής βρίσκεται μακριά από το ιατρικό κέντρο.

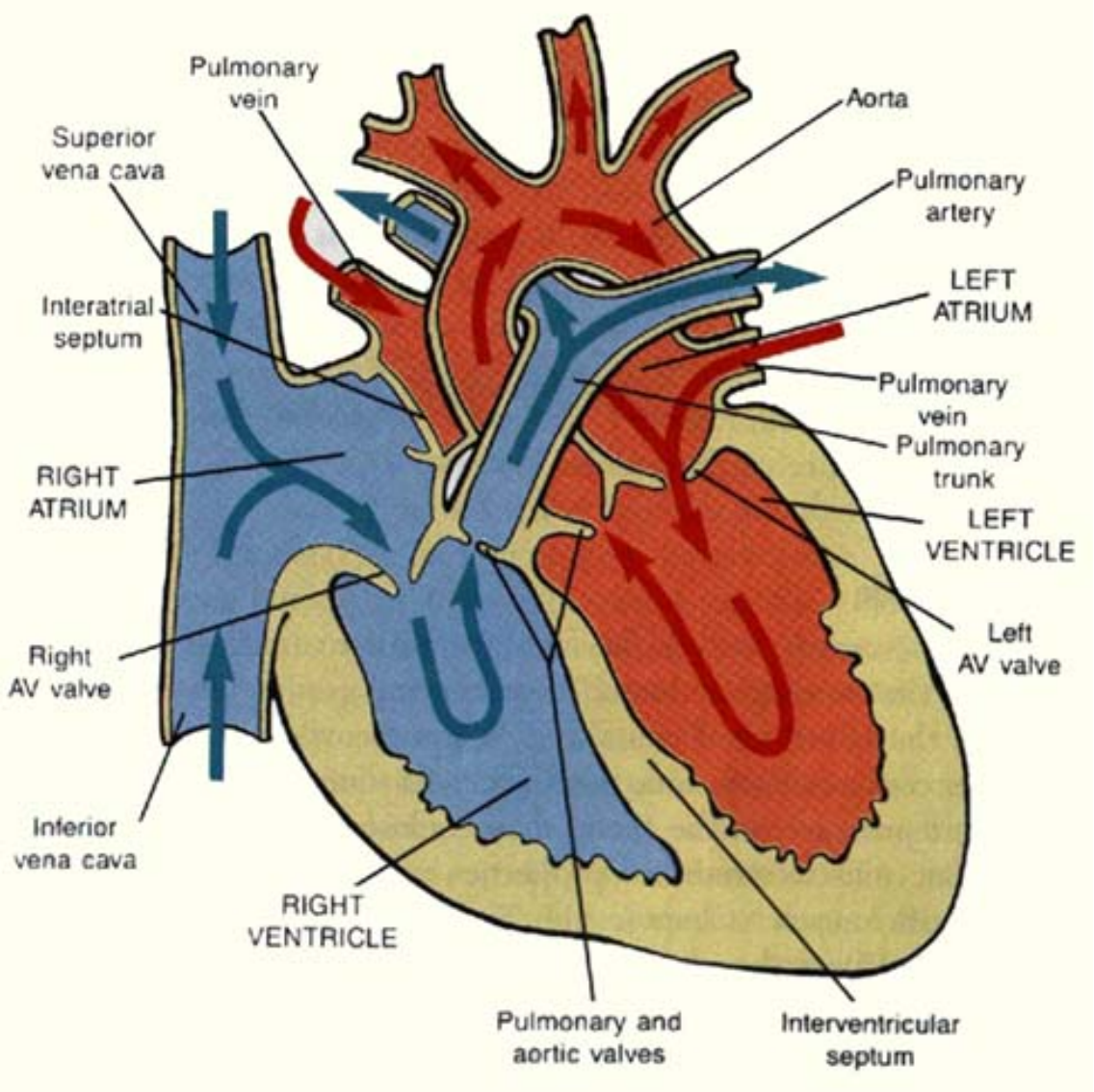
Λόγω της πολυπλοκότητας της εξαγωγής ακριβών μοντέλων του σήματος ΗΚΓ για την εκτίμηση και την πρόβλεψη της κατάστασης της καρδιάς, οι τεχνικές μηχανικής μάθησης κατέχουν κυρίαρχη θέση στο πεδίο της ανάλυσης ΗΚΓ. Τα Support Vector Machine [2] έχει αποδειχθεί ότι είναι ο πιο αποτελεσματικός τρόπος για να εξετάσει καποιος και να εξαγάγει χαρακτηριστικά σήματος ενός ΗΚΓ. Από τη μία πλευρά, τα Support Vector Machine εξασφαλίζουν πολύ υψηλή ακρίβεια ταξινόμησης ακόμη και σε πολύπλοκες μη γραμμικές κατανομές στο χώρο των εξαγόμενων λειτουργιών. Από την άλλη, βασίζονται σε πολλαπλούς υπολογισμούς, οι οποίοι οδηγούν σε υψηλό χρόνο εκτέλεσης καθώς και σε υπερβολική κατανάλωση ενέργειας. Στο [6], έχει αποδειχθεί ότι, όπου οι υπολογιστικές απαιτήσεις και οι απαιτήσεις ισχύος του ταξινομητή SVM πολλαπλασιάζονται, η επιτάχυνση υλικού μπορεί να είναι το κλειδί για την ικανοποίηση των περιορισμών χρόνου και ισχύος της ροής ανίχνευσης σήματος ECG. Στο [8], έχει αναπτυχθεί μια επιτάχυνση υλικού, έτσι ώστε η ροή ανάλυσης του ηλεκτροκαρδιογραφήματος να πληροί τα πρότυπα χρονισμού και ισχύος για μια συσκευή παρακολούθησης σε πραγματικό χρόνο. Η ροή ανάλυσης ΗΚΓ εφαρμόστηκε στο λογισμικό, εκτός από τον ταξινομητή Υποστηρικτικού μηχανήματος υποστήριξης ο οποίος υλοποιήθηκε ως επιταχυντής υλικού που στοχεύει FPGAs.

Σε αυτή τη διπλωματική εργασία, προσπαθούμε να βελτιστοποιήσουμε την απόδοση της ροής ανάλυσης του ΗΚΓ και συγκεκριμένα εστιάζουμε στην επιτάχυνση του πρώτου και του

τελευταίου σταδίου του αλγορίθμου, του φίλτρου αφαίρεσης θορύβου και του ταξινομητή των Support Vector Machine. Η υπερβολική ανάγκη για ένα σύστημα ανάλυσης του ΗΚΓ σε πραγματικό χρόνο σχετικά με τις φορητές ή εμφυτεύσιμες συσκευές οδηγεί σε αναπόφευκτους περιορισμούς χρόνου και ισχύος. Η προσέγγιση σε αυτή της διατριβής σε σχέση με τη βελτιστοποίηση και επιτάχυνση της ροής ανάλυσης ΗΚΓ είναι η εφαρμογή του αλγορίθμου σε ένα πολύπλευρο σύστημα πολύ μικρής ισχύος σε τσιπ. Προσπαθώντας να ανταποκριθούμε σε αυτές τις προδιαγραφές, προτείνουμε τον επεξεργαστή Movidius Myriad 2, έναν πολυπύρηνο SoC συγκεκριμένου σκοπού, στον οποίο υλοποιείται όλος ο αλγόριθμος, αλλά βελτιστοποιούνται τα πιο απαιτητικά και ενεργειακά απαιτητικά μέρη της ροής, χρησιμοποιώντας τα μικροαρχιτεκτονικά χαρακτηριστικά του επεξεργαστή. Ο επεξεργαστής Movidius Myriad 2 παρέχει 12 μίνι επεξεργαστές, οι οποίοι υποστηρίζουν λειτουργίες Very Long Instruction Word (VLIW), όσο και υψηλή απόδοση και βελτιστοποιημένες βιβλιοθήκες C, για να είναι δυνατή η διανυσμάτωση των αλγορίθμων. Λαμβάνοντας αυτό υπόψη, ξεκινήσαμε να παρουσιάζουμε διάφορες βελτιστοποιήσεις, επιτυγχάνοντας μέγιστη απόδοση. Αρχικά, έγινε αναδιάρθρωση του αρχικού κώδικα προκειμένου να αξιοποιηθούν πλήρως οι εγγενείς δυνατότητες παραλληλισμού των αλγορίθμων. Επιπλέον, οι βελτιστοποιήσεις βασίστηκαν στη μείωση της επιβάρυνσης της μνήμης της εφαρμογής. Με τον περιορισμό του αριθμού των απαιτούμενων προσπελάσεων στη μνήμη, μπορέσαμε να επιτρέψουμε στους επεξεργαστές VLIW του Myriad 2 να τρέξουν αποτελεσματικά τον αλγόριθμο. Επιπλέον, εφαρμόστηκε ένας αλγοριθμικός μετασχηματισμός για να ευθυγραμμιστούν οι πιο απαιτητικές λειτουργίες με το σχήμα επεξεργασίας των SHAVEs. Το Myriad 2, αντίθετα με τους παραδοσιακούς επεξεργαστές, έχει σχεδιαστεί για να λειτουργεί παράλληλα, προσπελαυνοντας τονους πληροφοριών ταυτόχρονα. Επομένως, κάνοντας μια σχετική βελτιστοποίηση στις βασικές λειτουργίες, επιτύχαμε ακόμη υψηλότερα επίπεδα απόδοσης. Αυτές οι στρατηγικές έχουν αποδειχθεί ότι είναι σε θέση να παράγουν όχι μόνο πολύ υψηλά κέρδη, αλλά και χαμηλή κατανάλωση ενέργειας.

Ανάλυση Ηλεκτροκαρδιογραφήματος. Η ροή των σημάτων του ΗΚΓ είναι μία από τις σημαντικότερες πηγές διαγνωστικών πληροφοριών. Ένα σήμα ηλεκτροκαρδιογραφήματος (ΗΚΓ) είναι η εκδήλωση ηλεκτρικής δραστηριότητας του μυοκαρδίου στην επιφάνεια του σώματος, η οποία εμφανίζεται ως ένα σχεδόν περιοδικό σήμα [6]. Οι λεπτές μεταβολές στο εύρος και τη διάρκεια αυτών των κυμάτων υποδηλώνουν διάφορες παθολογικές καταστάσεις της καρδιάς, μερικές από τις οποίες φαίνονται στο σχήμα ;;.

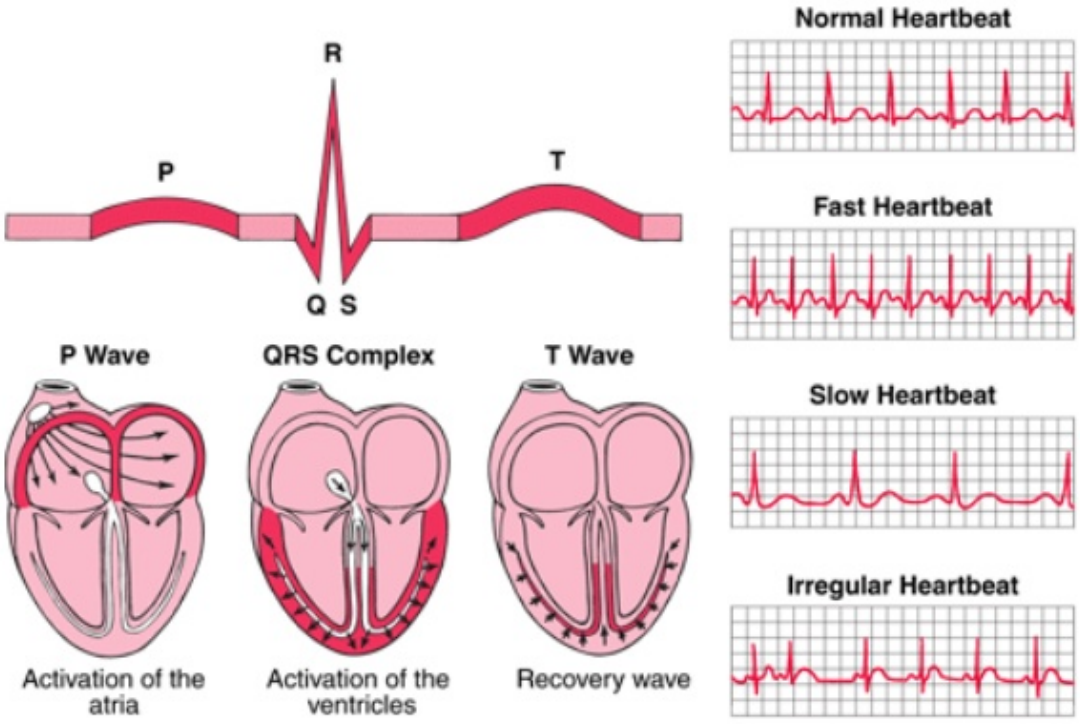
Μορφολογία της καρδιάς.



Σχήμα 1: Φυσιολογία της καρδιας και η ροή του αίματος στους κόλπους και στις κοιλίες [3]

Η καρδιά είναι ένα μυϊκό όργανο, το οποίο αντλεί αίμα μέσω των αιμοφόρων αγγείων του κυκλοφοριακού συστήματος. Στον άνθρωπο, η καρδιά, η οποία αποτελείται από το δεξί και το αριστερό μέρος, χωρίζεται σε τέσσερις θαλάμους: επάνω αριστερό και δεξή κόλπο και κάτω αριστερή και δεξιά κοιλία. Οι δύο άνω κόλποι είναι υπεύθυνοι για τη λήψη του αίματος και οι δύο κάτω κοιλίες είναι υπεύθυνες για την εκκένωση των θαλάμων. Ο δεξιός κόλπος δέχεται αίμα από τις κύριες φλέβες του σώματος. Το αίμα ρέει από το δεξιό κόλπο στη δεξιά κοιλία μέσω της τριγλωχίνας βαλβίδας και της δεξιάς κοιλίας και αντλεί το αίμα προς τους πνεύμονες μέσω του πνευμονικού κορμού και των δεξιών πνευμονικών αρτηριών. Στη συνέχεια, το νέο οξυγονωμένο αίμα επιστρέφει στον αριστερό κόλπο και, μέσω της μιτροειδούς βαλβίδας, διέρχεται στην αριστερή κοιλία, η οποία αντλεί το αίμα σε ολόκληρο το σώμα μέσω της αορτικής βαλβίδας. Η φυσιολογία της καρδιάς απεικονίζεται στο Σχήμα 1.

Ο καρδιακός κύκλος αναφέρεται στην ακολουθία των μηχανικών και ηλεκτρικών συμβάντων που επαναλαμβάνεται με κάθε κτύπο της καρδιάς και αποτελείται από τη φάση της χαλάρωσης, που ονομάζεται «διαστολή» και η φάση της σύσπασης, που ονομάζεται «συστολή». Όπως αναφέρθηκε προηγουμένως, η ανθρώπινη καρδιά είναι τετράγωνο όργανο, συνεπώς υπάρχει κολπική συστολή, κολπική διαστολή, κοιλιακή συστολή και κοιλιακή διαστολή. Η συχνότητα του καρδιακού κύκλου περιγράφεται από τον καρδιακό ρυθμό, τυπικά χτυπά ανά λεπτό. Κάθε καρδιακός κύκλος μπορεί να χωριστεί σε τέσσερα κύρια στάδια: τη φάση εισροής, την ισομετρική συστολή, τη φάση εκροής και την ισομετρική χάλαση. Το πρώτο και το τέταρτο στάδιο αποτελούν τη φάση της κοιλιακής διαστολής, η οποία αρχίζει όταν οι κοιλίες αρχίζουν να χαλαρώνουν. Όταν η πίεση στον πνευμονικό κορμό γίνεται υψηλότερη από την πίεση μέσα στις κοιλίες, καθώς το αίμα του προηγούμενου κύκλου συνεχίζει να εξέρχεται, η ημιτελική βαλβίδα θα κλείσει. Στη συνέχεια, οι βαλβίδες ημιτελικού και Α΄ είναι κλειστές, η οποία είναι η φάση της ισομετρική χαλάρωσης. Τελικά, οι κολποκοιλιακές βαλβίδες ανοίγουν και οι κοιλίες γεμίζονται γρήγορα με αίμα από τους κόλπους. Στο τέλος αυτής της φάσης, οι κόλποι συστέλλονται, αντλώντας περισσότερο αίμα στις κοιλίες. Ως αποτέλεσμα, η πίεση αυξάνεται στις κοιλίες και οι κολποκοιλιακές βαλβίδες θα κλείσουν, προκειμένου να αποφευχθεί η εισροή των αρθρώσεων. Αυτή η φάση ονομάζεται ισομετρική συστολή, λόγω της αύξησης της πίεσης. Η φάση εκροής αρχίζει, όταν η πίεση στις κοιλίες γίνεται υψηλότερη από το αίμα στον πνευμονικό κορμό (αορτή). Αυτό το γεγονός οδηγεί σε ημιτελικές βαλβίδες για να ανοίξει και το αίμα ρέει έξω από την πνευμονική αρτηρία. Στο τέλος του καρδιακού κύκλου, οι αορτικές και πνευμονικές βαλβίδες θα κλείσουν, καθώς οι κοιλίες εξαντλούνται.



Σχήμα 2: Ηλεκτροκαρδιογράφημα και μερικές παθολογικές καταστάσεις της καρδιάς που μπορούν να εξαχθουν από αυτό. [2]

Ο καρδιακός κύκλος, που περιγράφεται παραπάνω, συνδέεται στενά με ένα ηλεκτρικό σήμα, το οποίο παράγεται από τον φλεβόκομβο που βρίσκεται στο πάνω μέρος του δεξιού αίθριου. Το σήμα αυτό εξαπλώνεται μέσω των κόλπων, περνά τον κολποκοιλιακό κόμβο και μέσω των ινών Purknje καταλήγει σε όλες τις κοιλίες. Το πρότυπο ΗΚΓ βασίζεται γενικά στην αποπόλωση και την επαναπόλωση της καρδιάς. Η γενική κατεύθυνση της αποπόλωσης και της επαναπόλωσης παράγει ένα διάνυσμα που δημιουργεί θετική ή αρνητική παραμόρφωση στο ΗΚΓ. Μια κανονική κυματομορφή του ΗΚΓ αποτελείται από 3 κύριες οντότητες: ένα κύμα P, ένα κύμα QRS και ένα κύμα T. Το κύμα P αντιπροσωπεύει την κολπική αποπόλωση. Κατά τη διάρκεια αυτής της φάσης, το ηλεκτρικό σήμα φθάνει στο αριστερό και το δεξί αίτιο, το οποίο στη συνέχεια συστέλλεται και αντλεί επιπλέον αίμα μέσα στις κοιλίες. Τελικά, το ηλεκτρόνιο κύμα φτάνει στον κολποκοιλιακό κόμβο, οδηγώντας σε σύμπλεγμα QRS. Το σύμπλεγμα QRS αντιπροσωπεύει την κοιλιακή αποπόλωση, η οποία ακολουθείται από την κοιλιακή συστολή. Η ηλεκτρική συστολή των κοιλιών αρχίζει στην αρχή του συμπλέγματος QRS. Κατά τη διάρκεια του συμπλέγματος QRS, οι κοιλίες συστέλλονται για να εκτοξεύσουν το αίμα σε κυκλοφορία. Η κοιλιακή επαναπόλωση (ανάκτηση) ακολουθεί συστολή των κοιλιών, η οποία αντιπροσωπεύεται από το κύμα T. Σε αυτή τη φάση, οι κοιλίες ανακτούν και περιμένουν να ξαναγεμίσουν με το αίμα κυκλοφορίας.

Όπως αναφέρθηκε παραπάνω, μπορεί να εξαχθεί μια μεγάλη ποσότητα πληροφοριών σχετικά με τη δομή της καρδιάς και τη λειτουργία του ηλεκτρικού συστήματος αγωγιμότητας. Μεταξύ άλλων, ο ρυθμός και ο ρυθμός των καρδιακών παλμών μπορούν να μετρηθούν μέσω του σήματος του ΗΚΓ. Όλες οι οντότητες της κυματομορφής του ΗΚΓ και τα διαστήματα μεταξύ τους έχουν μια προβλέψιμη χρονική διάρκεια, ένα εύρος αποδεκτών μεγεθών και μία τυπική μορφολογία . Οποιαδήποτε απόκλιση από το τυπικό πρότυπο είναι κλινικής σημασίας. Η αρρυθμία θεωρείται ως μία από τις πιο κακοήθεις δυσλειτουργίες της καρδιάς. Η καρδιακή αρρυθμία, επίσης γνωστή ως καρδιακή δυσρυθμία ή ακανόνιστος καρδιακός παλμός, είναι μια ομάδα καταστάσεων στις οποίες ο καρδιακός παλμός είναι ακανόνιστος, πολύ γρήγορος ή πολύ αργός. Ωστόσο, ορισμένες ασυμπτωματικές αρρυθμίες σχετίζονται με ανεπιθύμητες ενέργειες. Παραδείγματα περιλαμβάνουν υψηλότερο κίνδυνο πήξης αίματος στην καρδιά και υψηλότερο κίνδυνο ανεπαρκούς μεταφοράς αίματος προς την καρδιά λόγω ασθενούς καρδιακού παλμού. Άλλοι αυξημένοι κίνδυνοι είναι η εμβολή και το εγκεφαλικό επεισόδιο, η καρδιακή ανεπάρκεια και ο αιφνίδιος καρδιακός θάνατος. Η ιατρική αξιολόγηση της ανωμαλίας με ηλεκτροκαρδιογράφημα είναι ένας τρόπος διάγνωσης και εκτίμησης του κινδύνου οποιασδήποτε αρρυθμίας.

Λαμβάνοντας υπόψη την κρίσιμη κατάσταση ενός ατόμου που πάσχει από επεισόδια αρρυθμίας, το πεδίο ανίχνευσης σημείων αρρυθμίας σε σήμα του ΗΚΓ έχει διερευνηθεί σε μεγάλο βαθμό. Δεδομένου ότι το σήμα ΗΚΓ είναι ένα μη στατικό σήμα, οι αρρυθμίες μπορεί να εμφανιστούν τυχαία σε χρονική κλίμακα. Επομένως, η μελέτη του σήματος του ηλεκτροκαρδιογραφήματος και της ένδειξης μεταβλητότητας του καρδιακού ρυθμού ενδέχεται να χρειαστεί

να υποβληθεί σε επεξεργασία αρκετές ώρες. Αυτό σημαίνει ότι πρέπει να εκτελεστεί ένα τεράστιο σύνολο δεδομένων προκειμένου να επιτευχθεί αποτελεσματικά η διάγνωση. Έτσι, οι τεχνικές μηχανικής μάθησης[7] είναι ιδανικές για την επίλυση του προβλήματος διάγνωσης. Το σύνολο δεδομένων χρησιμοποιείται ως το σετ κατάρτισης που απαιτείται από λύσεις μηχανικής μάθησης, οι οποίες μπορούν να δώσουν μια διάγνωση μετά την ολοκλήρωση της εκπαίδευσής τους. Στις περισσότερες μελέτες, χρησιμοποιείται ως πηγή εγγραφών ΗΚΓ, η βάση δεδομένων αρρυθμιών MIT-BIH. Έτσι, η βάση δεδομένων MIT-BIH χρησιμοποιείται για να διαμορφώσει το σύνολο εκπαίδευσης. Η βάση δεδομένων αρρυθμιών MIT-BIH αποτελείται από 48 μισαωρα απόσπασματα ηλεκτροκαρδιογραφήματων διπλών καναλιών από ασθενείς με διαφορετικά ιατρικά αρχεία και τύπους αρρυθμιών, ψηφιοποιημένα σε 360 δείγματα ανά δευτερόλεπτο ανά κανάλι με ανάλυση 11 bit σε περιοχή 10 mV. Επιπλέον, δύο ή περισσότεροι καρδιολόγοι σχολίασαν ανεξάρτητα κάθε εγγραφή και σχολιασμοί αναφοράς για κάθε κτύπο συμπεριλήφθηκαν στη βάση δεδομένων. Ως αποτέλεσμα, δημιουργήθηκαν σύνολα δεδομένων εκπαίδευσης για το πρόβλημα ανίχνευσης αρρυθμιών, χρησιμοποιώντας τη βάση δεδομένων MIT-BIH. Πρώτον, ένας εκπαιδευτής που ταξινομήθηκε με βάση τις τεχνικές μάθησης μηχανών εκπαιδεύτηκε χρησιμοποιώντας αυτό το σύνολο δεδομένων και, στη συνέχεια, χρησιμοποιήθηκε για την ανίχνευση αρρυθμιών σε μεμονωμένους ρυθμούς. Ο αλγόριθμος απόκτησης και επεξεργασίας του σήματος ΗΚΓ προκειμένου να εξαχθεί και να ταξινομηθεί κάθε κτύπος αποτελείται από διάφορα στάδια. Διαχωρίζεται σε τρία κύρια στάδια: στάδιο φιλτραρίσματος, ανίχνευση καρδιακού παλμού και στάδιο εξαγωγής χαρακτηριστικών και φάση ταξινόμησης. Μια επισκόπηση της ροής ανάλυσης ECG απεικονίζεται στο σχήμα 3 .

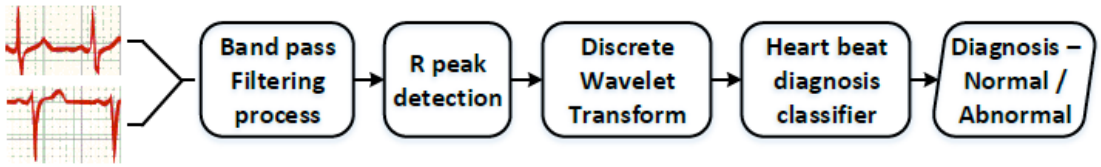
Αφαίρεση θορύβου. Σ' αυτό το στάδιο το σήμα φιλτράρεται για την απομάκρυνση θορύβου, συνήθως χρησιμοποιώντας φίλτρο διέλευσης ζώνης. Τα σήματα τεχνούργου που αφαιρούνται περιλαμβάνουν την περιστροφή της γραμμής βάσης, την παρεμβολή της γραμμής ισχύος και τον θόρυβο υψηλής συχνότητας. Τα τεχνουργήματα που προκύπτουν από την αναπνοή και την κίνηση των ασθενών πρέπει επίσης να αφαιρεθούν. Τα φιλτραρισμένα σήματα ΗΚΓ χρησιμοποιούνται σε όλες τις επακόλουθες επεξεργασίες.

Ανίχνευση κορυφής R. Σε αυτό το στάδιο ο απώτερος στόχος είναι να ανιχνευθούν οι καρδιακοί παλμοί που συνθέτουν ένα σήμα ΗΚΓ μεγαλύτερης διάρκειας. Για να γίνει αυτό συνήθως πρέπει να αναγνωριστούν κορυφές του συμπλέγματος QRS και πιθανώς P κύματα, T κύματα και QRS αντιστάσεις και αντισταθμίσεις. Η καρδιά χτύπησε την ακριβή τοποθεσία και η διάρκεια μπορεί να αφαιρεθεί από αυτές τις πληροφορίες. Η βάση δεδομένων αρρυθμιών MIT-BIH παρέχει στο χρήστη λειτουργίες υλοποίησης που εντοπίζουν αυτά τα βασικά σημεία. Αυτές οι λειτουργίες μπορούν να εφαρμοστούν στα διαθέσιμα σήματα ΗΚΓ και τα αποτελέσματα μπορούν να επαληθευτούν με τη βοήθεια των αρχείων σχολιασμού των γιατρών. Έτσι καταφέρουμε να απομονώσουμε τους ρυθμούς και έτσι να κατασκευάσουμε τους ρυθμούς που θα συμπεριληφθούν στο σύνολο δεδομένων εκπαίδευσης. Σε ένα σύστημα απόκτησης ΗΚΓ σε πραγματικό χρόνο, ο καρδιακός παλμός δεν είναι γνωστός, όπως και τα σημεία ενδιαφέροντος, όπως η κορυφή P, πρέπει να καθοριστούν με βάση μόνο τους ανιχνευτές QRS που είναι διαθέσιμοι προκειμένου να εντοπιστεί ένας νέος καρδιακός παλμός. Μικρότερα

κλάσματα του σήματος ΗΚΓ επεξεργάζονται τώρα για τμηματοποίηση του ρυθμού και ως εκ τούτου ανιχνεύονται λιγότεροι ρυθμοί κάθε φορά.

Εξαγωγή χαρακτηριστικών. Έχοντας καθορίσει νέο καρδιακό ρυθμό, επιβάλλεται μια διαδικασία εξαγωγής χαρακτηριστικών προκειμένου να εξαχθούν τα χαρακτηριστικά της. Αυτά τα χαρακτηριστικά αναφέρονται σε ειδικές παραμέτρους σήματος που είναι ενδεικτικές της φυσιολογικής κατάστασης ενδιαφέροντος. Για την ανίχνευση αρρυθμίας υπάρχει μια ποικιλία τύπων χαρακτηριστικών που μπορούν να χρησιμοποιηθούν, καθένα από τα οποία εισάγει διάφορα κλινικά συμπτώματα. Ο πιο πλήρης τρόπος για την εμφάνιση των πληροφοριών που περιλαμβάνονται στο σήμα ΗΚΓ είναι η πραγματοποίηση φασματικής ανάλυσης. Ο μετασχηματισμός της κυματομορφής(ΩΤ), μια επέκταση του κλασικού μετασχηματισμού [Fourier, μπορεί να εφαρμοστεί για την εξαγωγή των συντελεστών wavelet διακριτών σημάτων χρόνου, όπως το ΗΚΓ. Το WT λειτουργεί τόσο στον τομέα χρόνου όσο και συχνότητας και επιτρέπει την αποσύνθεση ενός σήματος σε έναν αριθμό κλιμάκων, όπου κάθε κλίμακα αντιπροσωπεύει μια συγκεκριμένη τραχύτητα του υπό μελέτη σήματος. Το WT έχει επίσης τη δυνατότητα να υπολογίζει και να χειρίζεται δεδομένα σε συμπιεσμένες παραμέτρους που ονομάζονται χαρακτηριστικά. Έτσι, χρησιμοποιώντας το μετασχηματισμό WT, το σήμα ΗΚΓ, που αποτελείται από πολλά σημεία δεδομένων, μπορεί να συμπιεστεί σε μερικές παραμέτρους που χαρακτηρίζουν τη συμπεριφορά του και δεν είναι εμφανείς από το αρχικό σήμα του χρονικού πεδίου. Οι καρδιακές παλμοί που ανιχνεύθηκαν στο προηγούμενο στάδιο, αποσυντίθενται σε παραστάσεις χρονικής συχνότητας χρησιμοποιώντας διακεκριμένο μετασχηματισμό ωαελετ (ΔΩΤ) και οι συντελεστές wavelet υπολογίζονται για να αντιπροσωπεύουν τα σήματα. Οι έξοδοι που παράγονται σε αυτό το στάδιο αποτελούν τους διανύσματα χαρακτηριστικών που χρησιμοποιούνται για την ταξινόμηση.

Ταξινόμηση-Διάγνωση. Το τελικό στάδιο της ροής ανάλυσης είναι ανίχνευση αν ο καρδιακός παλμός εμφανίζει σημάδια αρρυθμίας ή όχι. Αυτό γίνεται χρησιμοποιώντας έναν αλγόριθμο ταξινόμησης, ο οποίος ανιχνεύει το πρότυπο προβληματικού ρυθμού. Ο εκπαιδευτής έχει εκπαιδευτεί στο σύνολο δεδομένων που περιλαμβάνει τους διανύσματα χαρακτηριστικών των απομονωμένων κτυπημάτων. Δεδομένου ενός νέου διάνυσμα χαρακτηριστικών, ο ταξινομητής μπορεί να αποφασίσει εάν το αντίστοιχο κτύπημα εμφανίζει σημάδια αρρυθμίας. Υποστηρικτικοί μηχανισμοί υποστήριξης είναι οι βασισμένοι στη μάθηση μηχανισμοί ταξινόμησης που χρησιμοποιούνται σε αυτή τη μελέτη. Τα SVM είναι δημοφιλείς ταξινομητές μηχανικής μάθησης για μοντελοποίηση και ταξινόμηση βάσει δεδομένων και μπορούν να εκπαιδευτούν αποτελεσματικά εκτός σύνδεσης. Η διαδικασία κατάρτισης τους έχει σαν αποτέλεσμα ένα σύνολο φορέων, που ονομάζονται φορείς υποστήριξης (SVs), οι οποίοι χρησιμοποιούνται για να μοντελοποιούν τα δεδομένα αντιπροσωπεύοντας ένα όριο απόφασης. Αυτό το όριο απόφασης χρησιμοποιείται στη συνέχεια για να ταξινομήσει μια νέα εμφάνιση, το διάνυσμα χαρακτηριστικών ενός αταξινόμητου ρυθμού. Ο αριθμός των διανυσμάτων στήριξης και η διαστασιολόγηση των χαρακτήρων μπορούν να έχουν σημαντικό αντίκτυπο στην πολυπλοκότητα του ταξινομητή.



Σχήμα 3: Ροή Ανάλυσης ΗΚΓ [11]

Σχετική έρευνα.

Η έρευνα για το σήμα ηλεκτροκαρδιογραφήματος (ΗΚΓ) έχει χρησιμοποιηθεί ευρέως για την παρακολούθηση και τη διάγνωση πολλών καρδιακών παθήσεων. Τα τελευταία χρόνια έχουν διεξαχθεί πολυάριθμες έρευνες για την ανάλυση και ταξινόμηση του σήματος ΗΚΓ. Οι περισσότερες βιοϊατρικές συσκευές που χρησιμοποιούνται για την ανάλυση ΗΚΓ πρέπει να παρέχουν ακριβή αποτελέσματα σε πραγματικό χρόνο. Συνεπώς, πρέπει να επεξεργαστεί ένα μεγάλο μέρος δεδομένων με εξαιρετικά πολύπλοκες συσχετίσεις. Σύμφωνα με [7], οι τεχνικές μοντελοποίησης που βασίζονται σε δεδομένα εμφανίζονται ως μια ισχυρή προσέγγιση για την υπέρβαση των αναφερθέντων προκλήσεων [15], καθώς αναπτύχθηκαν γρήγορα πολλές τεχνικές εκμάθησης μηχανών, οι οποίες είναι σε θέση να διαχειρίζονται μεγάλες ποσότητες δεδομένων για να μοντελοποιήσουν συγκεκριμένους συσχετισμούς και στη συνέχεια, χρησιμοποιήστε μοντέλα μέσα σε μια λειτουργία απόφασης [14]. Επιπλέον, οι βιοϊατρικές συσκευές απαιτούν πολύ χαμηλή κατανάλωση ενέργειας, επειδή είναι συνήθως φορητές συσκευές. Προς αυτή την κατεύθυνση, οι συγγραφείς προτείνουν να χρησιμοποιηθούν αρχιτεκτονικές για εφαρμογές χαμηλής ενέργειας [20], [21], όπου χρησιμοποιούν ανίχνευση αρρυθμίας και ροή ανάλυσης ΗΚΓ. Χρησιμοποιούν επίσης ταξινομητές που βασίζονται στον φορέα υποστήριξης στη βαθμίδα ταξινόμησης και στη λειτουργία του πυρήνα RBF (εκθετική) στον πυρήνα ταξινόμησης, δεδομένου ότι οι γραμμικοί πυρήνες δεν είναι κατάλληλοι για το βαθμό πολυπλοκότητας της συσχέτισης των ιατρικών σημάτων. Η συνάρτηση του πυρήνα μαζί με τον αριθμό των διανυσμάτων υποστήριξης και τη διάσταση των χαρακτηριστικών διανυσμάτων μπορεί να έχει σημαντικό αντίκτυπο στην πολυπλοκότητα του ταξινομητή. Αυτό αποδείχθηκε με την εφαρμογή ολόκληρου του αλγορίθμου ανίχνευσης αρρυθμίας σε ενσωματωμένο επεξεργαστή χαμηλής ισχύος χρησιμοποιώντας μοντέλα ανίχνευσης υψηλής ανάλυσης για ακριβή ταξινόμηση σήματος και ανάλυση ενεργειακής απόδοσης. Διαπιστώνεται ότι η ταξινόμηση θέτει τη συμφόρηση της ενέργειας λόγω της πολυπλοκότητας των απαιτούμενων μοντέλων. Έτσι, στη μελέτη τους στοχεύουν στη βελτιστοποίηση του σταδίου ταξινόμησης όσον αφορά την απόδοση και την αποδοτικότητα της ενέργειας. Για να επιτευχθεί αυτό, διερευνούν την ανάπτυξη μιας αρχιτεκτονικής με βάση τον επεξεργαστή που είναι κατάλληλη για τη ροή ανάλυσης διαφόρων βιοϊατρικών σημάτων που απαιτούν ταξινόμηση. Ένας επεξεργαστής γενικού σκοπού χρησιμοποιείται για τον υπολογισμό χαρακτηριστικών, ενώ ένας βελτιστοποιημένος συν-επεξεργαστής χρησιμοποιείται για ταξινόμηση SVM βάσει πυρήνα. Οι προδιαγραφές για την πλατφόρμα πληρούν τους περιορισμούς για την ανίχνευση, την ενεργειακή απόδοση και την ευελιξία σε πραγματικό χρόνο, έτσι ώστε να υποστηρίζει διάφορες βιοϊατρικές εφαρμογές.

Η αρχιτεκτονική έχει τρία κύρια τμήματα: buffer για τους φορέα υποστήριξης και δοκιμής, μονάδες MAC και προγραμματιζόμενο πυρήνα πολυωνυμικού πυρήνα. Οι φορείς υποστήριξης φορτώνονται στα buffer μετά την ολοκλήρωση της εκπαίδευσης εκτός σύνδεσης και οι φορείς δοκιμής φορτώνονται δυναμικά στα αντίστοιχα προσωρινά τους δεδομένα. Κάθε μονάδα MAC είναι υπεύθυνη για τον υπολογισμό των προϊόντων κουκίδων του ταξινομητή SVM μεταξύ ενός διανύσματος δοκιμής και των φορέων υποστήριξης σε ένα προσωρινό buffer φορέα υποστήριξης. Μόλις ολοκληρωθεί ο πολλαπλασιασμός πάνω σε όλους τους φορείς υποστήριξης, τα προϊόντα κουκκίδων πολυπλέκονται στον προγραμματιζόμενο πυρήνα πολυωνυμικού πυρήνα, όπου υπολογίζεται ο μετασχηματισμός πολυωνύμου δεύτερης, τρίτης ή τέταρτης τάξης καθώς και ο πυρήνας του ταξινομητή SVM που επιλέγεται Ως RBF, sigmoid, κλπ). Τα αποτελέσματα κλιμακώνουν και αθροίζονται από έναν τελικό συσσωρευτή του οποίου το σήμα εξόδου καθορίζει το αποτέλεσμα ταξινόμησης. Οι υπολογισμοί εκτελούνται σε ακέραιες τιμές. Οι περιορισμοί σε πραγματικό χρόνο επιτυγχάνονται με παραλληλισμό των υπολογισμών των κουκίδων με τη χρήση πολλαπλών μονάδων MAC και η ενεργειακή απόδοση επιτυγχάνεται μέσω της κλιμάκωσης της τάσης στις μονάδες MAC.

Επιπλέον, σε [8], αναπτύχθηκε ένας co-processor FPGA υλικού για το στάδιο ταξινόμησης, προκειμένου να βελτιστοποιηθεί η απόδοση και η ενεργειακή αποδοτικότητα. Δημιούργησαν έναν συν-επεξεργαστή χρησιμοποιώντας εργαλεία σύνθεσης υψηλού επιπέδου και έτσι η αρχιτεκτονική καθορίζεται σύμφωνα με τη λειτουργία του πυρήνα και τα δεδομένα που σχετίζονται με το μοντέλο SVM, όπως ο αριθμός των φορέων υποστήριξης και οι συντελεστές, είναι hard-coded αντί να φορτώνονται. Η βελτιστοποίηση εφαρμόζεται με την τροποποίηση της δομής του κώδικα, την αύξηση του παραλληλισμού επιπέδου διδασκαλίας και τη χρήση των οδηγιών βελτιστοποίησης του εργαλείου. Σε αυτή τη διατριβή, ο στόχος είναι η υλοποίηση και βελτιστοποίηση της ροής ανάλυσης ΗΚΓ από την άποψη του κέρδους λανθάνουσας κατάστασης και της ενεργειακής απόδοσης σε ένα σύστημα πολλαπλών εντολών πολύ χαμηλής ισχύος, το Movidius Myriad 2, σχεδιασμένο για ειδικές εφαρμογές επεξεργασίας όρασης. Προκειμένου να επιτευχθεί η βελτιστοποίηση, η ανάπτυξη της ροής στο VPU Myriad 2 βασίστηκε στα δικά της αρχιτεκτονικά χαρακτηριστικά. Ο επεξεργαστής Μψριαδ 2 αποτελείται από συνολικά 14 πυρήνες. Λαμβάνοντας αυτό υπόψη και το γεγονός ότι το Myriad 2 έρχεται με μνήμη DRAM πολλαπλών δίσκων 2MB, το καταστήστε κατάλληλο για την εκτέλεση υψηλών υπολογιστικών εργασιών σε αυτό, όπως οι SVM classifiers. Επιπλέον, οι 12 ¨μίνι' πυρήνες υποστηρίζουν την αρχιτεκτονική SIMD (single instruction multiple data), επιτρέποντας τον υπολογισμό αριθμών 128 bit σε έναν κύκλο (περισσότερες πληροφορίες για την αρχιτεκτονική Myriad 2 περιγράφονται στο κεφάλαιο 4). Λαμβάνοντας υπόψη αυτά τα χαρακτηριστικά, εφαρμόστηκε παραλληλισμός επιπέδου διδασκαλίας, προκειμένου να μεγιστοποιηθεί ο παράλληλος υπολογισμός σε κάθε SHAVE και να γίνει δυνατή η χρήση του χαρακτηριστικού SIMD του επεξεργαστή. Επιπλέον, η αρχιτεκτονική του συστήματος μνήμης έχει μεγάλη σημασία στο Myriad 2. Έτσι, οι φορείς υποστήριξης χαρτογραφούνται στη 2-MB DRAM, προκειμένου να επιτευχθεί μεγαλύτερη καθυστέρηση κέρδους, ελαχιστοποιώντας τις διασταυρούμενες γέφυρες μνήμης SHAVE.

Θεωρητικό υπόβαθρο

Όπως έχει ήδη αναφερθεί, έχουν αναφερθεί δεδομένα από τη μυϊκή αρτηρία MIT-BIH χρησιμοποιήθηκε βάση δεδομένων. Αυτή η βάση δεδομένων είναι αποτέλεσμα της συνεργασίας του Ιατρικού Κέντρου Beth Israel Deaconess και του MIT και είναι μία από τις πιο χρησιμοποιούμενες βάσεις δεδομένων για ερευνητικούς σκοπούς. Η βάση δεδομένων αποτελείται από 48 μισάωρα αποσπάσματα διπλών καναλιών καταγραφής ΗΚΓ που προέρχονται από 47 άτομα. Από αυτά, εικοσιτέσσερις εγγραφές επιλέχθηκαν τυχαία από μια συλλογή πάνω από 4.000 24-ωρών καταγραφές ΗΚΓ, που χρησίμευαν ως αντιπροσωπευτικό δείγμα κλινικών αρχείων ρουτίνας. Οι υπόλοιπες είκοσι καταγραφές επιλέχθηκαν από το ίδιο σετ για να συμπεριλάβουν μια ποικιλία σπάνιων αλλά κλινικά φέροντων φαινομένων (σύνθετες κοιλιακές, διακλαδικές και υπερκοιλιακές αρρυθμίες), οι οποίες δεν θα εκπροσωπούνταν καλά σε ένα μικρό τυχαίο δείγμα. Τα θέματα περιελάμβαναν 25 άνδρες ηλικίας 32 έως 89 ετών και 22 γυναίκες ηλικίας 23 έως 89 ετών. Τα δεδομένα έχουν φιλτραριστεί από ένα ζωνοπερατό φίλτρο και έχουν ψηφιοποιηθεί με συχνότητα δειγματοληψίας 360 kHz το δευτερόλεπτο ανά κανάλι. Η MIT-BIH παρέχει επίσης σχολιασμούς για κάθε εγγραφή, όπου καρδιολόγοι έχουν ταξινομήσει κάθε παλμό. Υπάρχουν περίπου 110.000 σχολιασμοί παλμών.

Στην αλγοριθμική ανάλυση που ακολουθεί, έχουν χρησιμοποιηθεί δεδομένα απο όλα τα πρώτα κανάλια όλων των εγγραφών εκτός των 102, 104, 114. Έχουν εξετασθεί δύο είδη αρρυθμιών, οι κανονικοί και οι μη κανονικοί.

Μετασχηματισμός Διακριτού Χρόνου. Ο μετασχηματισμός διακριτού χρονου είναι παρόμοιος με το μετασχηματισμό Fourier, με τη διαφορά ότι μπορεί να παρέχει ταυτόχρονα πληροφορίες και για τον χρόνο αλλά και για την συχνότητα. Αυτό είναι σημαντικό όταν αναλύεις μη-στατικά σήματα, όπως το ηλεκτροκαρδιογράφημα. Γενικότερα, ο μετασχηματισμός διακριτού χρόνου μπορεί να εκφρασθεί από την παρακάτω συνάρτηση:

$$F(a,b) = \int_{-\infty}^{\infty} f(x)\psi^*_{(a,b)}(x)dx \tag{1}$$

όπου το * είναι το σύμβολο του συμπληρώματος και η συνάρτηση ψ είναι η συνάρτηση μετασχηματισμού. Ο μετασχηματισμός διακριτού χρόνου για να εξάγει ενα σήμα στην κλίμακα του χρόνου, χρησιμοποιεί διαφορετικά φίλτρα αποκοπής συχνοτήτων. Το σήμα περνάει από ενα υψηπερατό φίλτρο για να πάρει όλες τις υψηλές συχνότητες και από ενα βαθυπερό για τις χαμηλές. Έτσι, η ανάλυση του σήματος μπορεί να αλλάξει ανάλογα με τα φίλτρα που θα χρησιμοποιηθούν.

Θεωρία Μηχανών Διανυσμάτων Υποστήριξης. Οι Μηχανές Διανυσμάτων Υποστήριξης (Support Vector Machine - SVM) είναι μοντέλα επιβλεπόμενης μάθησης που εκπαιδεύονται με ένα μεγάλο σύνολο δεδομένων και είναι κατάλληλη για την ταξινόμηση των νέων εισόδων σε δύο υποψήφιες κλάσεις συμπληρωματικές μεταξύ τους. Το σύνολο εκπαίδευσης αποτελείται από διανύσματα με συγκεκριμένα χαρακτηριστικά καθένα από τα οποία διαθεται μια ετικέτα δηλωτικής της κλάσης στην οποία ανήκει. Ένα σύνολο από άλλα διανύσματα με τα ίδια χαρακτηριστικά και γνωστές τις ετικέτες χρησιμοποιείται για να ελεχθεί η

ακρίβεια της πρόβλεψης.

Τα SVM εφαρμόζουν αρχικά μια συνάρτηση πυρήνα που ανάγει τα διανύσματα σε ενα χώρο περισσότερων διαστάσεων, όπου είναι πιο εύκολος ο διαχωρισμός τους. Στο χώρο αυτό βρίσκουν ένα υπερεπίπεδο το οποίο αποτελείται από τα διανύσματα που απέχουν μέγιστα από τα διανύσματα που ανήκουν σε κάθε κλάση. Κάθε νέο διάνυσμα ανάγεται σε αυτόν το χώρο, υπολογίζεται η απόσταση του από το υπερεπίπεδο και άρα με βάση τη θέση του σε σχέση με αυτό ταξινομείται στην αντίστοιχη κλάση. Η συνάρτηση πυρήνα είναι καθοριστική για την ακρίβεια και την πολυπλοκότητα του μοντέλου. Λόγω των μη γραμμικών σχέσεων μεταξύ των χαρακτηριστικών του διανύσματος κάθε παλμού χρησιμοποιούμε μη γραμμική συνάρτηση πυρήνα και συγκεκριμένα εκθετικής φύσης.

Ακολουθεί η μαθηματική εξίσωση που περιγράφει τον υπολογιστικό πυρήνα του ταξινομητή και ο αντίστοιχος κώδικας [7] που την υλοποιεί:

$$Class = sgn(\sum_{i=1}^{N\_sv} (y_i * a_i * K(x, sup\_vector_i)) - b) \tag{2}$$

όπου K είναι η συνάρτηση πυρήνα, x είναι το διάνυσμα του παλμού προς ταξινόμηση, sup_vector_i είναι το i-οστό διάνυσμα ταξινόμησης και y_i, q_i είναι τιμές διαφορετικές για κάθε διάνυσμα υποστήριξης και προέκυψαν κατά την εκπαίδευση. Η μεταβλητή b είναι μια μεταβλητή σύγκρισης, αποτέλεσμα της εκπαίδευσης και σταθερή για όλα τα διανύσματα υποστήριξης.

Listing 1: Αρχικός κώδικας SVM

```
const float sv_coef[N_sv];
const float sup_vectors[D_sv][N_sv];

void SVM_predict (int *y, float test_vector [D_sv]){

for(i=0; i<N_sv; i++){
        for(j=0; j<D_sv; j++){
                diff=test_vector[j] - sup_vectors[j][i];
                norma = norma + diff*diff;
                }
        sum = sum + exp(-gamma*norma)*sv_coef[i];
        norma = 0;
}

sum = sum - b;

if (sum<0)
        *y = -1;
else
        *y = 1;
}
```

Στον παραπάνω κώδικα 1, η μεταβλητή *sv_coef* ισοδυναμεί με το γινόμενο το y_i και a_i της εξίσωσης του υπολογιστικού πυρήνα. Ο αριθμός των διανυσμάτων υποστήριξης *N_sv* και ο αριθμός των χαρακτηριστικών *D_sv*, όπως και η επιλογή συνάρτησης πυρήνα έχουν μεγάλη επίδραση στην πολυπλοκότητα.

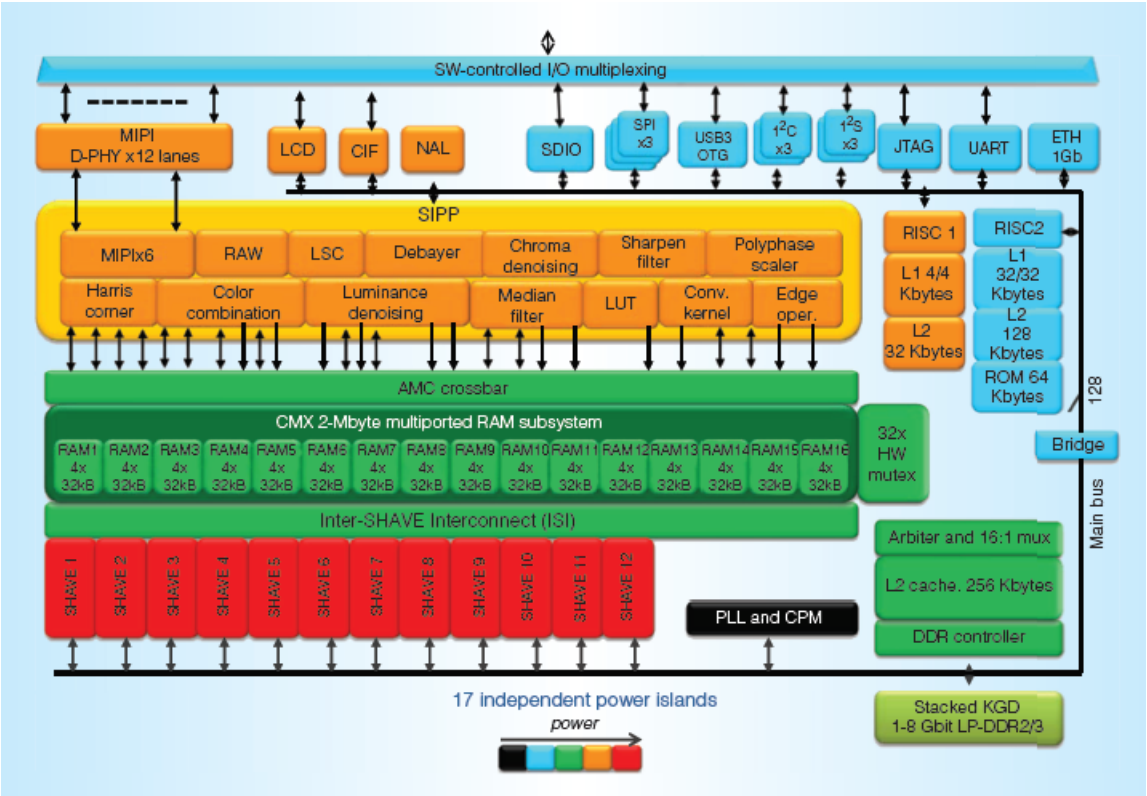
Παρουσίαση του Myriad 2 System-on-Chip

Η Myriad 2 είναι ένας επεξεργαστής τύπου System-on-Chip, ο οποίος είναι σχεδιασμένος για να πραγματοποιεί πολύπλοκες υπολογιστικές διαδικασίες. Συνδυάζει την παραλληλία και μικροαρχιτεκτονικά χαρακτηριστικά, ώστε να παρέχει πολύ καλές επιδόσεις σε πολλές εφαρμογές.

Οι περισσότεροι επεξεργαστές System-on-Chip (SoC) είναι συνήθως βασισμένοι σε έναν ή περισσότερους επεξεργαστές αρχιτεκτονικής RISC, οι οποίοι περιβάλλονται από διάφορους επιταχυντές υλικού και μοιράζονται μια πολυεπίπεδη cache και ένα περιβάλλον από DRAM. Αυτές οι κοινές δομές είναι ελκυστικές από άποψη κόστους, αλλά δημιουργεί πολλά προβλήματα από την πλευρά της προσπέλασης της μνήμης. Μία καλύτερη λύση είναι να φτιάξουμε μία προγραμματιζόμενη αρχιτεκτονική, ως δευτερεύον επεξεργαστή σε ένα κύριο, ο οποίος θα αναλαμβάνει όλο το βαρύ φόρτο εργασίας σε πραγματικό χρόνο, καθώς και θα χειρίζεται όλα τα άλλα συστήματα εικόνας και ήχου.

Η Myriad 2 ως κλασσικό SoC, συνδυάζει 2 επεξεργαστές LEON αρχιτεκτονικής RISC. Επίσης, προσφέρει 12 SHAVEs, τα οποία προσφέρουν καλύτερες επιδόσεις. Οι δύο κύριοι επεξεργαστές μπορούν να ελέγχουν τους 12 ή ένα αριθμό από τους ενσωματωμένους πυρήνες. Ο LEON OS μπορεί να ελέγχει όλες τις περιφερειακές εισόδους και εξόδους και γιαυτό έχει μία L1 cache 32KB και μια L2 cache 256KB, δίνοντας του τη δυνατότητα να τρέξει ένα λειτουργικό σύστημα. Ο LEON RT είναι υπεύθενος για όλες τις συσκευές ήχου και εικόνας, και έχει μια L1 cache 4KB και L2 cache 32KB.

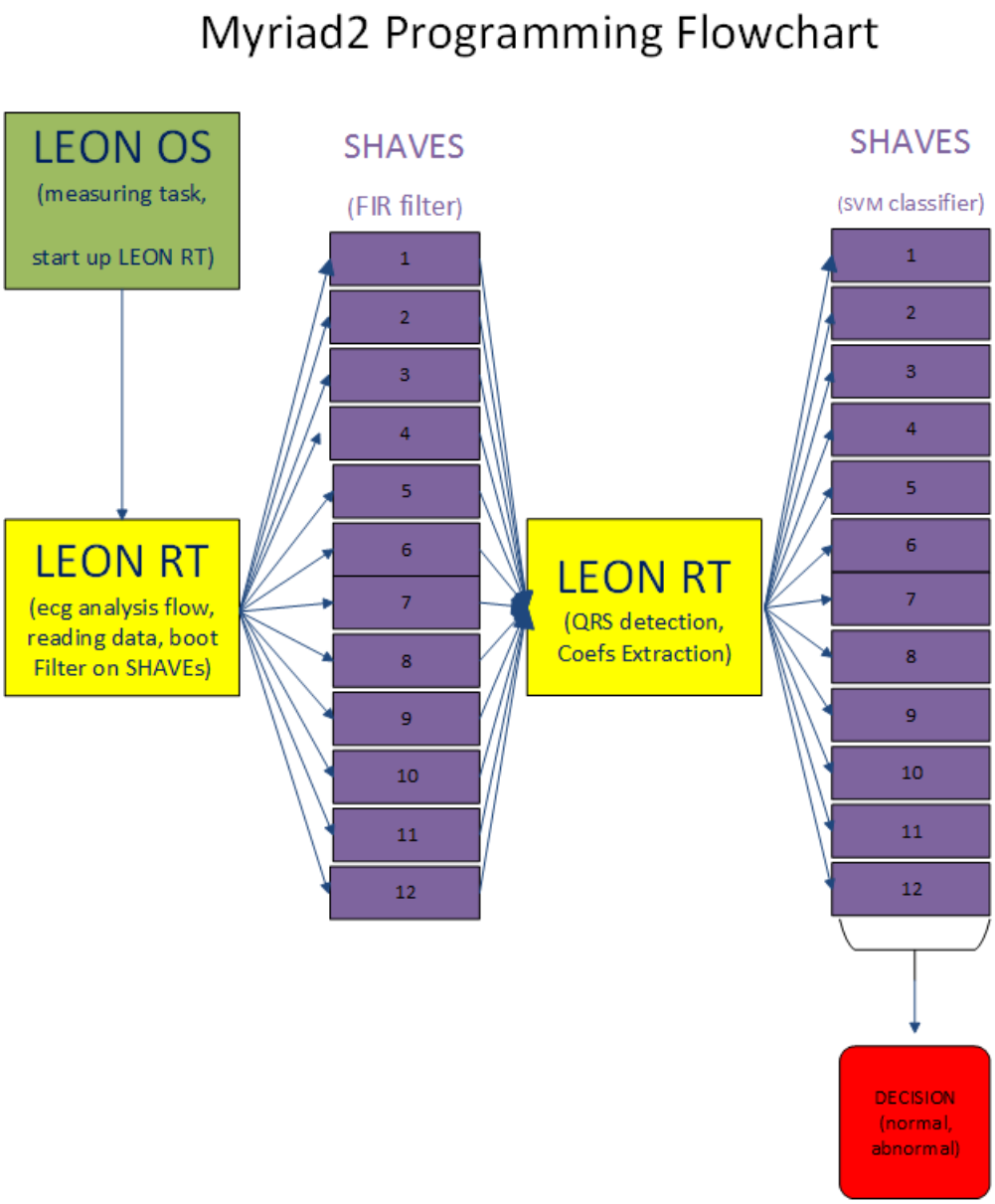
Επιπλέον, η Myriad 2 περιλαμβάνει μία ελεγχόμενη από το λογισμικό, πολυπύρηνη μνήμη, μεγέθους 2MB, η οποία μπορεί να υποστηρίξει του 12 επεξεργαστες, καθώς και τους 2 ΡΙΣ$^{\text{n}}$ επεξεργαστές. Επίσης, μία μηχανή άμεσες προσπέλασης της μνήμης είναι διαθέσιμη, έτσι ώστε να είναι δυνατή η διεπαφή των επεξεργαστών τόσο μεταξύ τους, όσο και με όλα τα περιφερειακά συστήματα της πλατφόρμας.



Σχήμα 4: Η Myriad 2 ως μονάδα επεξεργασίας εικόνας (VPU) στο λεπτομερές διάγραμμα(SoC) δείχνει τους 12 πυρήνες SHAVE και τη σχετική διασύνδεση των SHAVE με το υποσύστημα μνήμης πολλαπλών εγγραφών. Πάνω από τη μνήμη CMX είναι οι συσκευές επιτάχυνσης υλικού για την όραση και την επεξεργασία εικόνας, καθώς και άλλες συσκευές πολυμέσων, που ελέγχονται από τον επεξεργαστή επεξεργαστή υπολογιστών RISC με μειωμένη οδηγία LEON RT. Πάνω από αυτό, είναι τα περιφερειακά I/O, τα οποία ελέγχονται από άλλο επεξεργαστή RISC (LEON OS).[7]

Η Myriad 2 προσφέρει τους 12 SHAVE πυρήνες, οι οποίοι υποστηρίζουν την αρχιτεκτονική 128-bit VLIW, τα οποία προσφέρονται για μεγάλες υπολογιστικές διαδικασίες. Επειδή η κατανάλωση ενέργειας είναι ένας κύριος παράγοντας, ο επεξεργαστής χρησιμοποιεί 17 νησίδες ισχύος, έτσι ώστε να μπορούμε να βελτιστοποιούμε την κατανάλωση ισχύος ανάλογα με την εφαρμογή μας. Η Myriad 2 περιλαμβάνει τα εξής 3 κύρια αρχιτεκτονικά τμήμα: Media Sub System(MSS), CPU Sub System(CSS) και το Microprocessor Array(UPA).

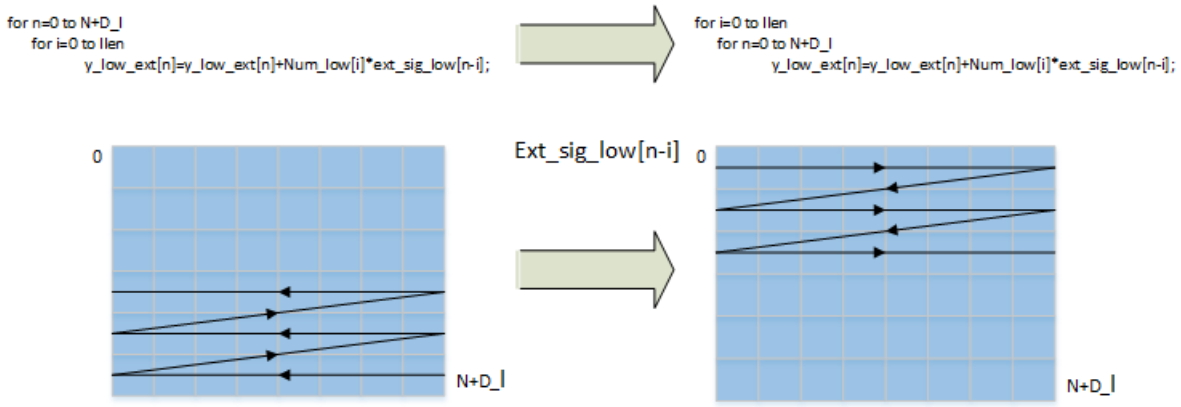
Υλοποίηση της Ροής Ανάλυσης ΗΚΓ στην Myriad 2Αρχικά, ανοίγει ο LEON OS, ο οποίος τρέχει ένα λειτουργικό σύστημα, ενώ παράλληλα τρέχει έναν αλγόριθμο για τη λήψη μετρήσεων τόσο για την κατανάλωση ισχύος και τον χρόνο εκτέλεσης του αλγορίθμου. Αυτός με τη σειρά του ανοίγει τον LEON RT, ο οποίος ξεκινάει τον αλγόριθμο της Ροής Ανάλυσης ΗΚΓ, χρησιμοποιώντας τα SHAVEs για να παραλληλοποιήση και να επιταχύνει τον αλγόριθμο του φίλτρου και του SVM classifier.



Σχήμα 5: Διάγραμμα Ροής της Ροής Ανάλυσης ΗΚΓ στη Myriad 2

Φίλτρο απαλοιφής θορύβου. Αρχικά, υλοποιήσαμε και βελτιστοποιήσαμε το φίλτρο. Η πρώτη υλοποίηση έγινε στον LEON RT. Στη συνέχεια, περάσαμε τον κώδικα σε ένα SHAVE. Αρχικά παρατηρήσαμε ότι ο κώδικας δεν μπορεί να διανυσματοποιηθεί, ώστε να μπορούμε να κάνουμε περισσότερες από μία πράξεις σε ένα κύκλο του ρολογιού. Αυτό συνέβαινε, καθώς δεν διασχίζοταν όλοι οι πίνακες με την ίδια φορά. Για το λόγο αυτό, πραγματοποιήσαμε ένα loop interchange στις δύο εμφωλευμένες επαναλήψεις, όπως φαίνεται στην εικόνα 6. Με αυτό τον τρόπο καταφέραμε να διασχίζονται όλοι οι πίνακες με την ίδια φορά, ώστε να μπορούν να παραλληλοποιηθούν οι πράξεις, πραγματοποιώντας 4 πράξεις σε ένα κύκλο του ρολογιού, αξιοποιώντας έτσι την αρχιτεκτονική των SHAVEs. Στην συνέχεια, αφού καταφέραμε να επιταχύνουμε τον κώδικα σε ένα SHAVE, η κύρια ιδέα της παραλληλοποίησης ήταν να μοιράσουμε το φόρτο του φίλτρου σε περισσότερα από ένα SHAVEs. Αυτό θα γινόταν, μειώνοντας το

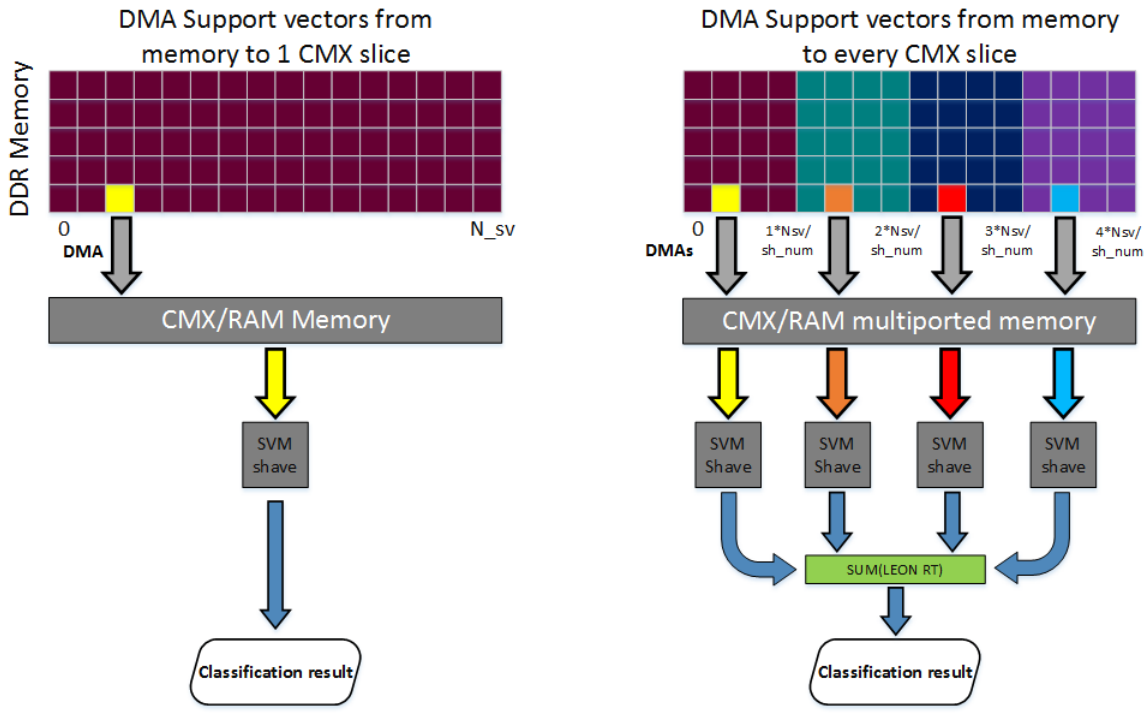
σήμα εισόδου που θα πήγαινε σε κάθε ένα από τα SHAVEs. Σε αυτό το σημείο παρατηρήθηκε ότι δεν μπορεί να μοιραστεί ως έχει το σήμα εισόδου, καθώς θα επηρεαστεί η ορθότητα του αλγορίθμου. Για αυτό το λόγο, ήταν απαραίτητο να έχουμε μερικά σημεία του σήματος μας πριν το επιλεγμένο προς επεξεργασία σήμα.



Σχήμα 6: Loop interchange

SVM ταξινομητής.Αφού βελτιστοποιήσαμε τον αλγόριθμο του φίλτρου, προσπαθούμε να επιταχύνουμε τον Support Vector Machine classifier, το πιο απαιτητικό τμήμα του αλγορίθμου μας. Η αρχική μορφή του κώδικα φαίνεται στον κωδικα 1.

Όπως παρατηρούμε, ο κώδικας αποτελείται από 2 εμφωλεύμενες επαναλήψεις, όπου η εσωτερική επανάληψη υπολογίζει την Ευκλείδεια απόσταση μεταξύ των διανυσμάτων υποστήριξης και των χαρακτηριστικών του παλμού και στην συνέχεια, στην εξωτερική επανάληψη υπολογίζεται η συνάρτηση πυρήνα του ταξινομητή. Η τιμή αυτή πολλαπλασιάζεται με ένα παράγοντα βάρους του εκάστοτε διανύσματος υποστήριξης. Στο τέλος, το αποτέλεσμα προστίθεται σε ένα συνολικό άθροισμα, το οποίο στο τέλος συγκρίνεται με το μηδέν, ώστε να βγεί η τελική απόφαση. Παρατηρούμε ότι δεν έχουμε εξαρτήσεις δεδομένων κατά τους υπολογισμούς και άρα οι υπολογισμοί μπορούν να γίνουν παράλληλα, όπως φαίνεται στην εικόνα 7.



Σχήμα 7: Παραλληλισμός SVM ταξινομητή

Όπως και στο φίλτρο, ο πίνακας με τα διανύσματα υποστήριξης μπορεί να χωριστεί σε μικρότερους πίνακες με λιγότερα διανύσματα. Αυτοί οι πίνακες θα έχουν το ίδιο μέγεθος. Κάθε πίνακας θα συνεισφέρει στον υπολογισμό ενός μερικού αθροίσματος και όλοι μαζί στον υπολογισμό του συνολικού αθροίσματος. Οι υπολογισμοί που απαιτούνται για τον υπολογισμώ των μερικών αθροισμάτων μπορούν να τρέχουν παράλληλα. Έτσι, καταφέραμε να μετατρέψουμε το αρχικό μας πρόβλημα σε επιμέρους μικρά προβλήματα. Όλες οι μικρότερες διαδικασίες μπορούν να τρέξουν παράλληλα και ανεξάρτητα μεταξύ τους και στο τέλος, όλα τα μερικά αθροίσματα επιστρέφονται στον LEON RT, όπου υπολογίζεται το συνολικό άθροισμα.

Αρχικά, όλο το τμήμα του αλγορίθμου υπολογιζόταν στον LEON RT, αλλά στη συνέχεια, χωρίζοντας το πρόβλημα σε μικρότερες υπο-διεργασίες, μοιράστηκαν οι υπολογισμοί στα SHAVEs. Στις αρχικές εκτελέσεις, τα δεδομένα του πίνακα με τα διανύσματα υποστήριξης βρισκόντουσαν στην DDR μνήμη και διαβάζονταν απευθείας από τη μνήμη. Αυτό, όμως, είχε ως αποτέλεσμα να έχουμε πάρα πολλές προσπελάσεις στη μνήμη, άρα και αυξημένο χρόνο εκτέλεσης του αλγορίθμου και αυξημένη κατανάλωση ισχύος. Για να αποφύγουμε αυτή την κατάσταση, αποφασίσαμε να χρησιμοποιήσουμε μια μηχανή Άμεσης Προσπέλασης της Μνήμης, η οποία μετέφερε τα δεδομένα απο την DDR μνήμη στην CMX-RAM μνήμη, όπου τα SHAVEs έχουν ταχύτερο χρόνο προσπέλασης και μικρότερη κατανάλωση ενέργειας. Στην αρχή, είχαμε πολλές μεταφορές δεδομένων από την μία μνήμη στην άλλη. Δημιουργώντας ένα διπλό buffer, καταφέραμε να μειώσουμε τις μεταφορές σε δύο. Έτσι, φέρναμε στην αρχή τα πρώτα δεδομένα που θα επεξεργαζόταν ο αλγόριθμος μας και στη συνέχεια, καθώς αυτός έκανε τους πρώτους υπολογισμούς, φέρναμε και τα υπόλοιπα δεδομένα στο δεύτερο μισό του buffer. Τέλος, καθώς κάθε SHAVE έχει αποκλειστικότητα σε ένα κομμάτι της CMX μνήμης, αποθηκέυσαμε με την

μορφή ενός αρχείου header τα διανύσματα υποστήριξης στην μνήμη. Έτσι, έχουμε ακόμα ταχύτερη προσπέλασης της μνήμης και καμία μεταφορά των δεδομένων από την DDR στην CMX μνήμη.

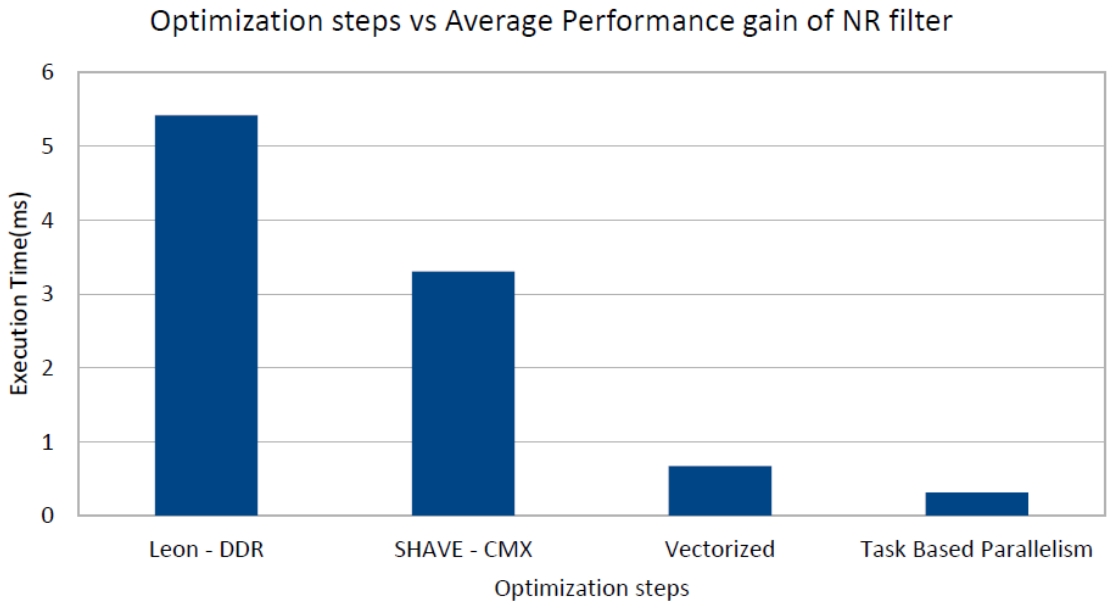
Στην συνέχεια, πέρα από την παραλληλοποίηση σε επίπεδο SHAVE, παρατηρήσαμε ότι μπορούμε να παραλληλοποιήσουμε τις πράξεις στην εσωτερική επανάληψη του κώδικα, όπου υπολογίζονται οι Ευκλείδειες αποστάσεις. Για να το πετύχουμε αυτό, κάναμε ένα loop unroll κατά 2, έτσι ώστε να πραγματοποιούνται δύο πράξεις ταυτόχρονα. Στο τέλος, αποφασίσαμε να αλλάξουμε την συνάρτηση πυρήνα του ταξινομητή και να χρησιμοποιήσουμε μία προσέγγιση της εκθετικής συνάρτησης με τη σειρά Taylor της συνάρτησης, όπως φαίνεται στην εικόνα 8 . Το κύριο πλεονέκτημα αυτής της επιλογής είναι ότι η ακρίβεια του αλγορίθμου μπορει να ελεγθεί ρυθμίζοντας τον παράγοντα της σειράς. Έτσι, χρησιμοποιώντας το ανάπτυγμα 5ης τάξης καταφέραμε να ελαχιστοποιήσουμε κατά πολύ τις πράξεις και να έχουμε ένα κέρδος της τάξης του 30%.

$$e^x = \sum_{k=1}^{\infty} \frac{x^k}{k!} = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$$

Σχήμα 8: Ανάπτυγμα της Σειράς Taylor της εκθετιής συνάρτησης

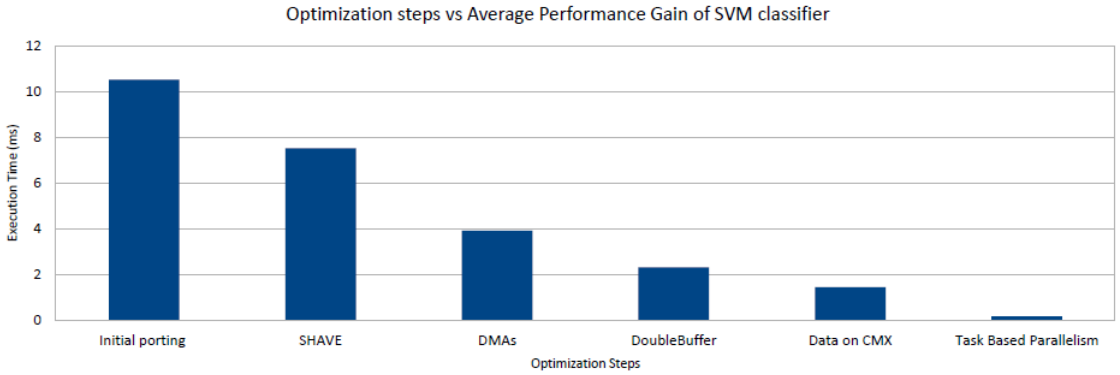
Αποτελέσματα

Παρακάτω παρουσιάζονται τα αποτέλεσματα των βελτιστοποιήσεων που εφαρμόστηκαν στον κώδικα της Ροής Ανάλυσης ΗΚΓ. Αρχικά, παρουσιάζονται οι χρόνοι του φίλτρου κατά τα επιμέρους βήματα της βελτιστοποίησης του. Πρώτα, είναι ο χρόνος της αρχικής υλοποίησης του φίλτρου, έπειτα η εκτέλεση σε ένα SHAVE με τα δεδομένα του φίλτρου στην CMX μνήμη. Στην συνέχεια, ο χρόνος εκτέλεσης του κώδικα όταν αυτός είναι διανυσματοποιημένος και τέλος, ο τελικός χρόνος του φιλτραρίσματος του αλγορίθμου όταν παραλληλοποιείται σε όλα τα SHAVEs.



Σχήμα 9: Απόδοση στα επιμέρους βήματα επιτάχυνσης του φίλτρου

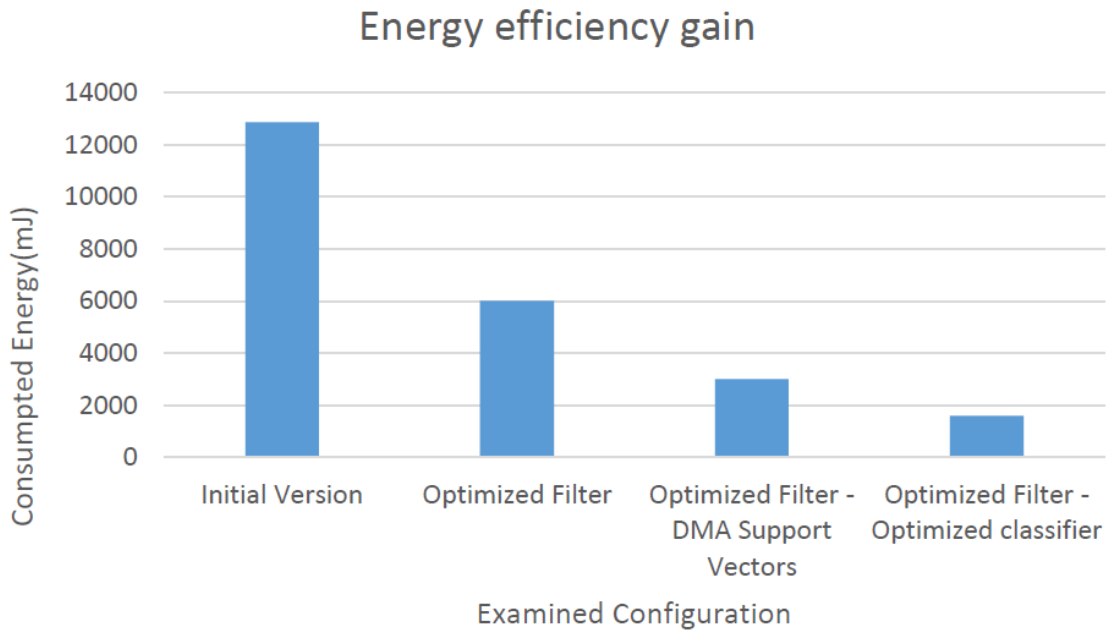
Στη συνέχεια, βλέπουμε τους χρόνους εκτέλεσης του SVM ταξινομητή στα κάθε επιμέρους βήμα υλοποίησης του. Αρχικά, τα δεδομένα βρίσκονται στην DDR μνήμη, στην συνέχεια τα φορτώνουμε με DMAs στην CMX μνήμη. Έπειτα, προσπαθούμε να μειώσουμε τα DMAs, φτιάχνοντας ένα διπλό buffer και τέλος, παραλληλοποιούμε τον κώδικα σε όλα τα SHAVEs.



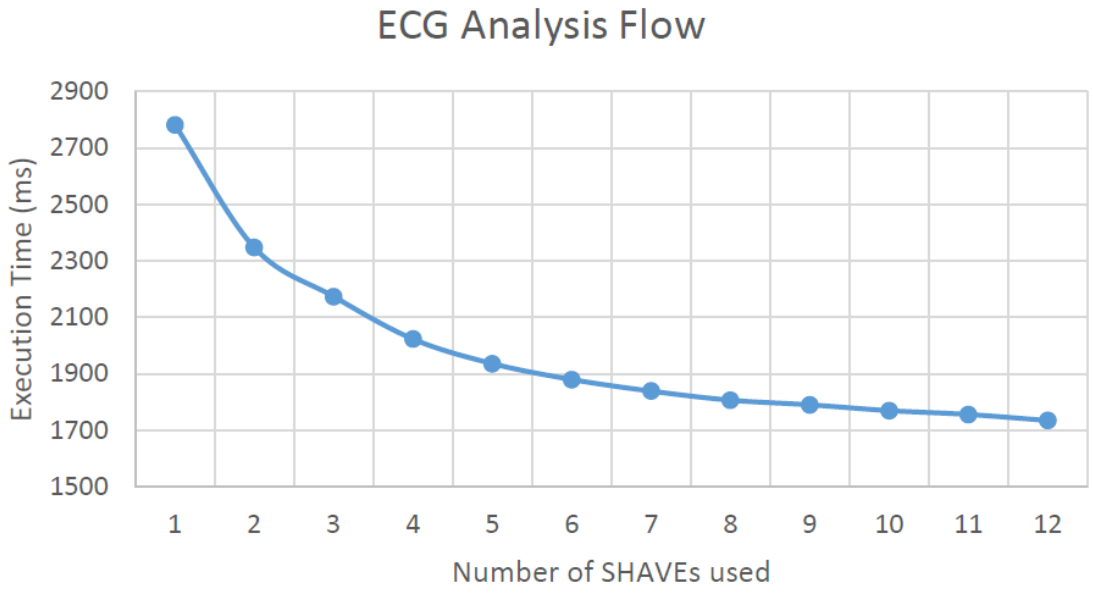
Σχήμα 10: Απόδοση στα επιμέρους βήματα επιτάχυνσης του SVM ταξινομητή

Στις παρακάτω γραφικές παραστάσεις, βλέπουμε την κατανάλωση ενέργειας της ροής ανάλυσης ΗΚΓ σε επιμέρους βήμα υλοποίησης του αλγορίθμου. Επίσης, βλέπουμε την κατανάλωση ενέργειας, ανάλογα με το πόσα SHAVEs χρησιμοποιούμε, καθώς και το χρόνο εκτέλεσης ανάλογα με το πόσα SHAVEs χρησιμοποιούμε. Σε αυτές τις δύο γραφικές παραστάσεις παρατηρούμε ότι καθώς ο χρόνος εκτέλεσης συνεχίζει να μειώνεται όσο χρησιμοποιούμε περισσότερα SHAVEs, δεν συμβαίνει το ίδιο και με την κατανάλωση ενέργειας. Αυτό συμβαίνει, καθώς όσο ανοίγουμε περισσότερα SHAVEs, αυτά καταναλώνουν περισσότερη ισχύ και, ταυ-

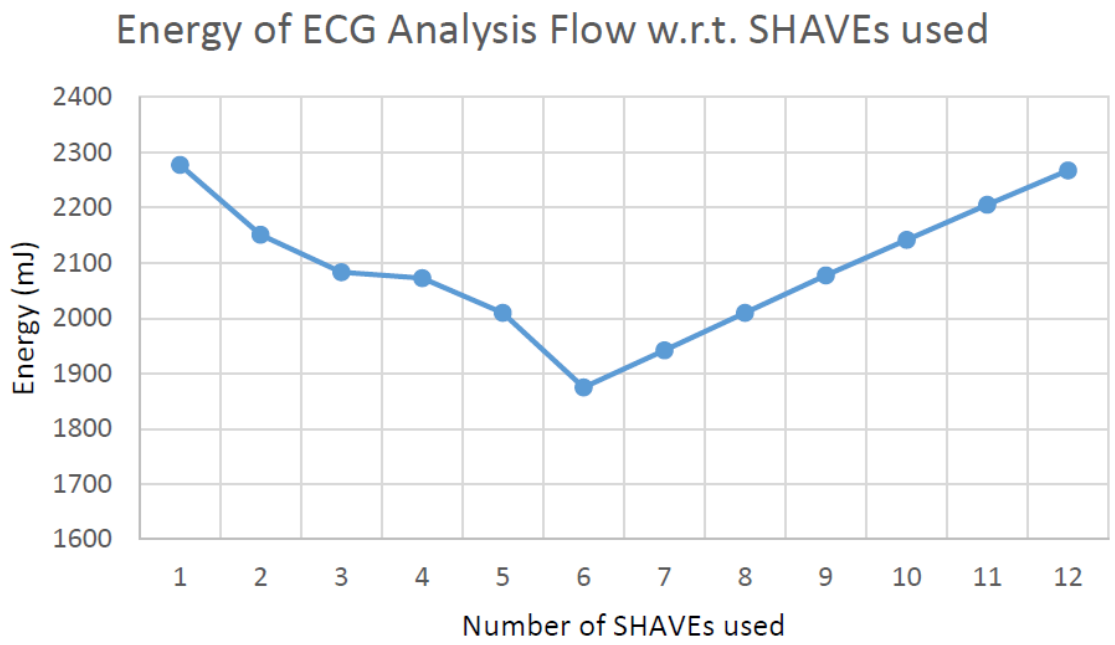
τόχρονα, το κέρδος σε χρόνο δεν είναι τόσο ώστε να αντισταθμίσει την καταναλυσκώμενη ισχύ.



Σχήμα 11: Η μείωση στην κατανάλωση ενέργεια σε επιμέρους υλοποιήσεις

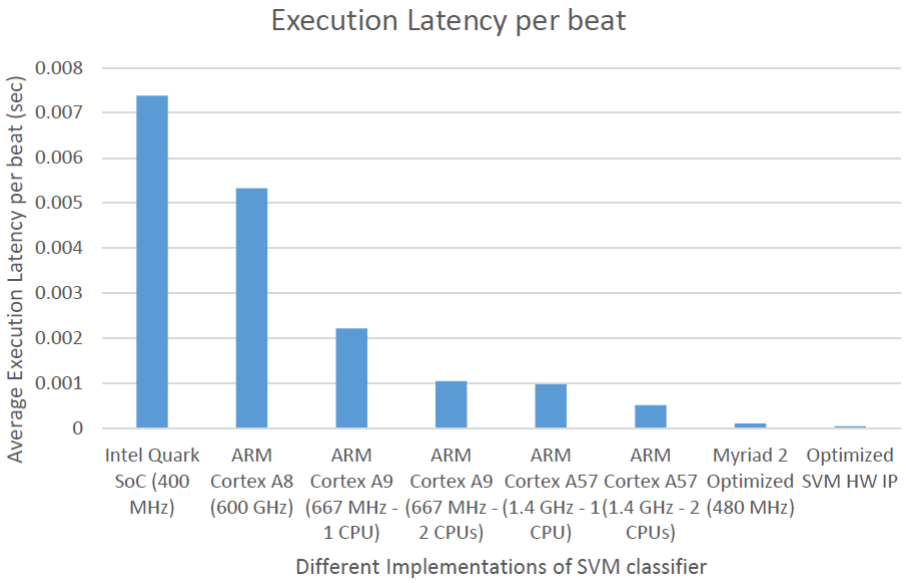


Σχήμα 12: Χρόνος εκτέλεσης της Ροής Ανάλυσης ΗΚΓ ανάλογα με τα SHAVEs που χρησιμοποιούνται



Σχήμα 13: Η κατανάλωση ενέργειας της Ροής Ανάλυσης ΗΚΓ ανάλογα με τον αριθμό των SHAVEs που χρησιμοποιούνται

Παρακάτω βλέπουμε τον χρόνο εκτέλεσης του SVM ταξινομητή σε διαφορετικούς επεξεργαστές. Παρατηρούμε ότι ο χρόνος εκτέλεσης στη Myriad 2 είναι συγκρίσιμος με το χρόνο που έχουμε σε ένα FPGA.



Σχήμα 14: Ο χρόνος εκτέλεσης του ταξινομητή σε διαφορετικούς επεξεργαστές

Συμπεράσματα

Στην εργασία αυτή εξετάσαμε μια μεθοδολογία για την ανάπτυξη και τη βελτιστοποίηση μιας ροής ανάλυσης ΗΚΓ σε ενσωματωμένες αρχιτεκτονικές. Ως μελέτη περίπτωσης, εργαστήκαμε στην αποτελεσματική εφαρμογή του αλγόριθμου ροής ανάλυσης ΗΚΓ σε πλατφόρμα που ειδικεύεται κυρίως σε επεξεργασία εικόνας και βίντεο, όπως η Movidius Myriad 2. Ο αλγόριθμος επιτάχυνσης της ανάλυσης ηλεκτροκαρδιογραφήματος και οι αλγόριθμοι βαθιάς μάθησης σε μια ενσωματωμένη αρχιτεκτονική είναι μια ενδιαφέρουσα ιδέα για πολλούς επιστήμονες και μηχανικούς λογισμικού Οι οποίοι έχουν πρόσβαση σε σχετικές βιβλιοθήκες λογισμικού για μηχανές γενικής χρήσης και θέλουν να γνωρίζουν την προσπάθεια που απαιτείται για την ανάπτυξή τους σε μια ενσωματωμένη πλατφόρμα. Η μεθοδολογία βασίζεται, πρώτον, στη διαρθρωτική μετατροπή του κώδικα, προκειμένου να παραλληλισθεί σε πολλούς πυρήνες και, δεύτερον, στην εφαρμογή χειρωνακτικών τροποποιήσεων στον κώδικα, προκειμένου να βοηθήσει τον μεταγλωττιστή να δημιουργήσει έναν διανυσματικό κώδικα ή να διανείμει την κωδικοποίηση . Οι κυριότερες προκλήσεις αυτής της εφαρμογής δημιουργήθηκαν από τους περιορισμούς στο μέγεθος της μνήμης, ένα κοινό χαρακτηριστικό των ενσωματωμένων συστημάτων. Εκτός από τις τροποποιήσεις που μείωσαν την επιβάρυνση της μνήμης της εφαρμογής, εφαρμόστηκαν επίσης πολλές βελτιστοποιήσεις προκειμένου να επιτευχθούν υψηλά επίπεδα απόδοσης.

Μελλοντικές Επεκτάσεις

Σήμερα, μια έγκυρη και έγκαιρη διάγνωση του προβλήματος ενός ασθενούς είναι υψίστης σημασίας. Για να μπορέσουν να εκτελέσουν τους περίπλοκους αλγορίθμους που απαιτούνται για τέτοιες εργασίες, οι παραδοσιακοί τρόποι βελτίωσης των επιδόσεων σε αυτές τις συσκευές πρέπει να επανεξεταστούν. Ο νόμος του Moore επιβραδύνεται, γεγονός που μειώνει τη δύναμη και τα κέρδη απόδοσης κατά τη μετάβαση στον επόμενο κόμβο διεργασίας. Ως εκ τούτου, οι ενσωματωμένοι σχεδιαστές πλατφόρμας πρέπει να καταλήξουν σε πιο έξυπνους τρόπους για να δημιουργήσουν ισχυρές και οικονομικά αποδοτικές συσκευές.

Σε αυτή τη διατριβή δουλέψαμε με μια συσκευή σχεδιασμένη με αυτόν τον τρόπο, με βάση την κατανόηση ότι υπάρχει βαθιά αλληλεξάρτηση μεταξύ αλγορίθμων και της αρχιτεκτονικής των επεξεργαστών. Η Myriad 2 είναι μια συσκευή που ειδικεύεται στην αποτελεσματική εργασία μηχανικής όρασης. Ωστόσο, χρησιμοποιήθηκε σε αυτή τη διατριβή για να αναπτύξει έναν αλγόριθμο ανάλυσης ΗΚΓ, τη ροή ανάλυσης ΗΚΓ, η οποία περιλαμβάνει επίσης τον αλγόριθμο μηχανικής μάθησης. Η Myriad 2 είναι πολύ ελπιδοφόρο για την επιτάχυνση των αλγορίθμων Deep Learning. Οι βασικές τεχνικές που περιγράφονται σε αυτή τη διατριβή μπορούν επίσης να χρησιμοποιηθούν για τη βελτίωση άλλων παρόμοιων εφαρμογών. Αν και αυτοί οι αλγόριθμοι αναπτύχθηκαν για να τρέχουν κυρίως σε υπερυπολογιστές, μπορούν τώρα να μεταφερθούν σε ισχυρές ενσωματωμένες συσκευές και να χρησιμοποιηθούν σε πραγματικό χρόνο.

Επιπλέον, ένα ενδιαφέρον βήμα προς τα εμπρός θα ήταν η περαιτέρω επιτάχυνση ενός Support Vector Machine ταξινομητή χρησιμοποιώντας την ενσωματωμένη γλώσσα προγραμματισμού. Επιπροσθέτως, εκτός από τα τμήματα φίλτρου και SVM, θα ήταν ενδιαφέρον να βελτιστοποιηθούν τα άλλα μέρη του αλγορίθμου σε συστήματα πολλαπλών εντολών και να εφαρμοστούν τεχνικές για την αξιοποίηση των χαρακτηριστικών του SIMD. Το βήμα εξαγωγής χαρακτηριστικών βάσει του διακριτού μετασχηματισμού κύματος θα ήταν ιδανικός υποψήφιος.

Τέλος, η ροή ανάλυσης ΗΚΓ θα μπορούσε να είναι εξ ολοκλήρου σε SHAVEs, έτσι ώστε οι αλγόριθμοι των διαφορετικών συχνών δειγματοληψιών να μπορούν να χρησιμοποιηθούν ταυτόχρονα.

Contents

List of Figures

Chapter 1

Introduction

Cardiovascular diseases has become the ascendant cause of mortality in the world [4]. That group of diseases are closely related with the heart or the blood vessels and include heart attack, hypertensive heart disease, venous thrombosis and many other heart diseases. According to the World Health Organization, the number of deaths due to cardiovascular diseases has grown dramatically, about 41% over the past 30 years. Consequently, valid and accurate diagnosis of heart condition is of utmost importance.

One of the most crucial biological signals for monitoring and diagnosing the condition of the heart is the Electrocardiogram (ECG). The ECG is closesy related to the heart's morphology and physiology. Therefore, the investigation electrocardiogram (ECG) has been extensively used for the diagnosis of many heart diseases. For assessing and deriving, faster and more accurate, information from the ECG signal is essential the contribution of technology. Over the past years, many electronic wearable devices have been manufactured for medical purposes. These devices are able to monitoring and record heart condition on day long basis. As a result, it is easier and faster for the doctor to examine the patient, while also patient's life quality has been improved, minimizing healthcare cost by reducing the need for his hospitalization and avoiding many hours waiting in queues in the hospital. Additionally, the risk of death, due to such diseases, has been deminished, because of the remote on-time diagnosis, even when the patient is far away from the medical center.

Due to the complexity of deriving exact models of the ECG signal for assessing and predicting heart's condition, machine learnig techniques have become dominant on the field of ECG analysis. Support Vector Machine [5] based classifiers specifically have been proved to be the most efficient way to examine and extract ECG signal features. On the one hand, Support Vector Machines ensure very high classification accuracy even on complex non-linear distribution in the extracted features space. On the other hand, they are based on multiple computation operations, which leads to high execution time as well as, excessive power consumption. In [9], has been shown that, where the computational and power requirements of the SVM classifier are multiplied, hardware acceleration can

be the key for meeting both time and power constraints of the ECG signal analysis flow. In [11], a hardware acceleration has been developed, so that the ECG analysis flow meet timing and power standards for a real-time monitoring device. The ECG analysis flow was implemented in software, except for the Support Vector Machine classifier which was implemented as a hardware accelerator targeting FPGAs.

In this diploma thesis, we try to optimize the performance of the ECG analysis flow, and specifically we focus on the acceleration of the first and the last stage of the algorithm, the noiseremoval filter and the Support Vector Machine classifier. The excessive need for a real-time ECG analysis system on wearable or implantable devices leads to inevitable time and power constraints. The approach in this thesis in respect of the optimization and acceleration of the ECG analysis flow is to implement the algorithm on an ultra-low-power multicore system-on-chip. Working towards meeting these specifications, we propose Movidius Myriad 2 processor, a specific purpose multicore SoC , in which all the algorithm is implemented, but the most time consuming and energy demanding parts in the flow are optimized, utilizing the processor's microarchitectural features. Movidius Myriad 2 processor provides 12 mini-processors, which support Very Long Instruction Word(VLIW) operations, as long as, high throughput and optimized C libraries, in order to be able to vectorize the algorithms. Taking that into account, we started introducing several optimizations, reaching a maximum performance. Initially, manually restructuring of the original code was made in order to fully utilize the inherent parallelization capabilities of the algorithms. Moreover, the optimizations were based in reducing the memory overhead of the application. By limiting the number of required memory accesses, we were able to allow the VLIW processors of Myriad 2 to run the algorithm efficiently, without too many stalls. In addition, an algorithmic transformation was applied in order to align the most demanding functions with the processing scheme of the SHAVEs. Myriad 2, instead of traditional processors, is designed for operating in parallel on tons of information that they are all coming through at once. Therefore, by making a relevant optimization in the basic functions, we were able to achieve even higher efficiency levels. These strategies have proven to be capable of producing not only very high latency gains, but also low power consumption.

In Chapter 2, a brief overview of the ECG Analysis Flow is presented. The stages of the flow are explained to inform the reader of the steps required for a successful arrhythmia detection. Related work progress on the field is also included, to highlight our approach and its contribution. In Chapter 3, the theoritical backround of the thesis is presented, such as the MIT database, from which the tested samples was taken. Additionally, extensive backround information on Support Vector Machines is included, in order to explain its functionality and complexity and stresses the need of efficient acceleration. In Chapter 4, the development board, Movidius Myriad 2, is presented. Its technical features are detailed and programming paradigms avalaible for the developer was also presented. In Chapter 5, our implementation on Myriad 2 was presented. Every development option is justified and

the impact of their application is analysed. The chapter concludes with a comparative study of the results of all implementation during the development, as well as, with a comparison of a hardware accelerated flow and our development. In last chapter, Chapter 6, imporant conclusions are discussed, as well as, ideas for improving and extending the existing work in the future.

Chapter 2

Problem Overview

2.1 ECG Analysis Flow

ECG signals are one among the most important sources of diagnostic information. An electrocardiogram (ECG) signal is the manifestation of myocardium electrical activity on the body surface, which appears as a nearly periodic signal [**?**]. The subtle changes in amplitude and duration of these waves indicate various pathological heart conditions, some can be seen in Fig. 2.3.

Heart Morphology

The heart is a muscular organ, which pumps blood through the blood vessels of the cirulatory system. In humans, the heart, which consists of right and left part, is divided into four chambers: upper left and right atria and lower left and right ventricles. The two upper atria are responsible for receiving the blood, and the two lower ventricles are responsible for discharging the chambers. The right atrium receives blood from the major veins of the body. The blood flows from the right atrium to the right ventricle via tricuspid valve and right ventricle contracts and pumps the blood towards the lungs through the pulmonary trunk and the right pulmonary arterties. Subsequently, the newly oxygenated blood returns to the left atrium, and, through the mitral valve, passes to the left ventricle, which pumps the blood to the entire body through the aortic valve. The phusiology of the heart is depicted in Fig.2.1.

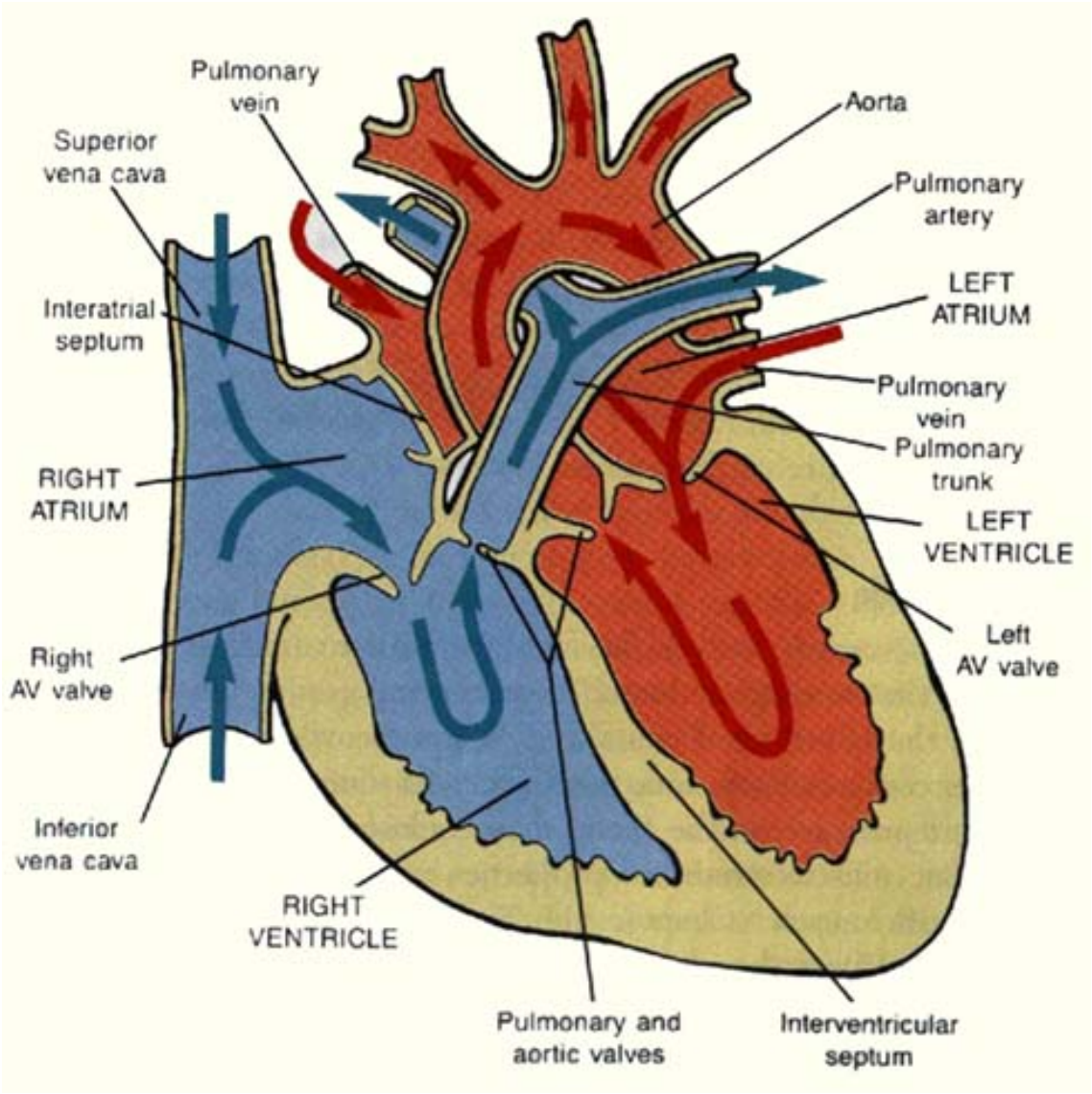


Figure 2.1: Heart physiology and blood flow through the atria and ventricles.[3]

The cardiac cycle refers to the sequence of mechanical and electrical events that repeats with every heartbeat and it consists of the phase of relaxation, called "diastole" and the phase of contraction, called "systole". As mentioned before, the human heart is four chambered organ, thus there are atrial systole, atrial diastole, ventricle systole and atrial systole. The frequency of the cardiac cycle is described by the heart rate ,typically beats per minute. Each cardiac cycle can be divides into four major stages: the inflow phase, isovolumetric contraction, outflow phase and isovolumetric relaxation. The first and the fourth stages constitute the ventricular diastole phase, which begins when the ventricles start to relax. When the pressure in the pulmonary trunk becomes higher than the pressure inside the ventricles, as the blood of the previous cycle is still flowing out, the semilunar valve will shut. Then, semilunar and AV valves are closed, which is the phase of isovolumetric relaxation. Eventually, the atrioventricular valves (AV) open, and the ventricles are being rapidly filled with blood from atria. At the end of this phase,

atria contract, pumping more blood into the ventricles. As a result, the pressure rises in the ventricles and the atrioventricular valves will shut, in order to prevent backflow into the atria. This phase is called isovolumic contraction, due to the rise of pressure. The outflow phase begins, when the pressure in the ventricles becomes higher that the blood in the pulmonary trunk (aorta). This fact leads semilunar valves to open and the blood flows out the pulmonary artery. At the end of the cardiac cycle, the aortic and pulmonary valves will close, as the ventricles deplete.

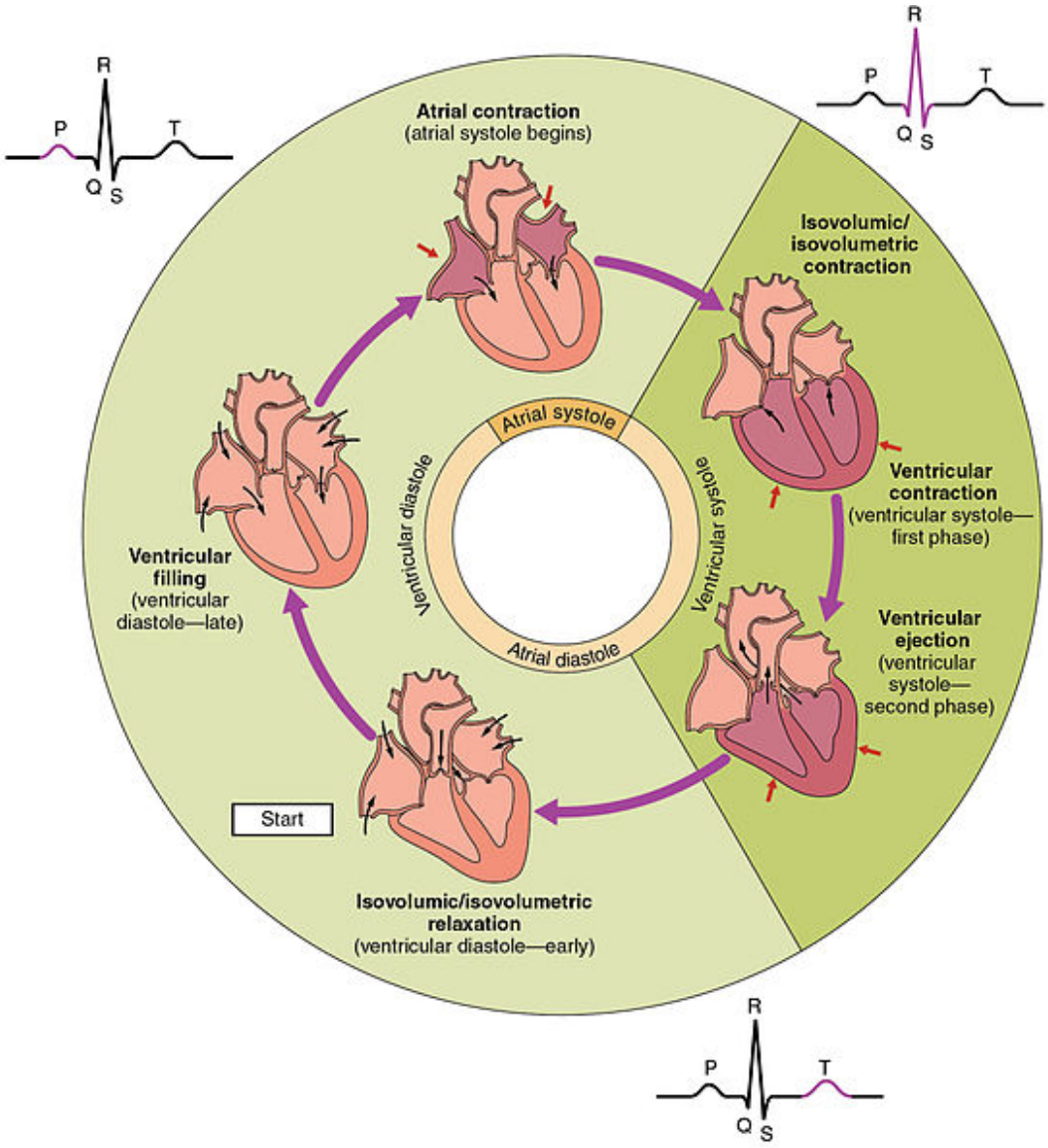


Figure 2.2: Heart cycle correlated to ECG signal.[1]

The cardiac cycle, described above, is closely connected with an electrical signal, that is generated by the sinoatrial node, the heart's peace maker, located in the upper part of the right atrium. That signal spreads out through the atrium, passes the atrioventricular node and via the Purknje fibers ends up throughout the ventricles. ECG pattern is generally based on the depolarization and repolarization of the heart. The overall direction

of depolarization and repolarization produces a vector that creates a positive or negative deflection on the ECG. A normal ECG waveform consists of 3 major entities: a P wave, a QRS wave and a T wave. The P wave represents atrial depolarization. During this phase, electrical signal reaches left and right atria, which subsequently contract and pump additional blood inside the ventricles. Eventually, the electircal wave reaches the atrioventricular node, leading to QRS complex. The QRS complex represents the ventricular depolarization, which is followed by the ventricular systole. The electrical systole of the ventricles begins at the onset of the QRS complex. During QRS complex, the ventricles contract in order to eject the blood into circulation. The ventricular repolarization (recovery) follows ventricles contraction, which is represented by the T wave. In this phase, ventricles recover and wait to be refilled with the circulation blood.

As mentioned above, a large amount of information about the structure of the heart and the function of its electrical conduction system can be extracted. Among other things, the rate and the rhyhm of heartbeats can be measured throught ECG signal. All entities of the ECG waveform and the intervals between them have a predictable time duration, a range of acceptable amplitudes and a typical morpholoy. Any deviation from the typical pattern is of clinical significance. Arrhythmia is considered as one of the most commly encountered heart malfunctions. Cardiac arrhythmia, also known as cardiac dysrhythmia or irregular heartbeat, is a group of conditions in which the heartbeat is irregular, too fast, or too slow. However, some asymptomatic arrhythmias are associated with adverse events. Examples include a higher risk of blood clotting within the heart and a higher risk of insufficient blood being transported to the heart because of weak heartbeat. Other increased risks are of embolisation and stroke, heart failure and sudden cardiac death. Medical assessment of the abnormality using an electrocardiogram is one way to diagnose and assess the risk of any given arrhythmia.

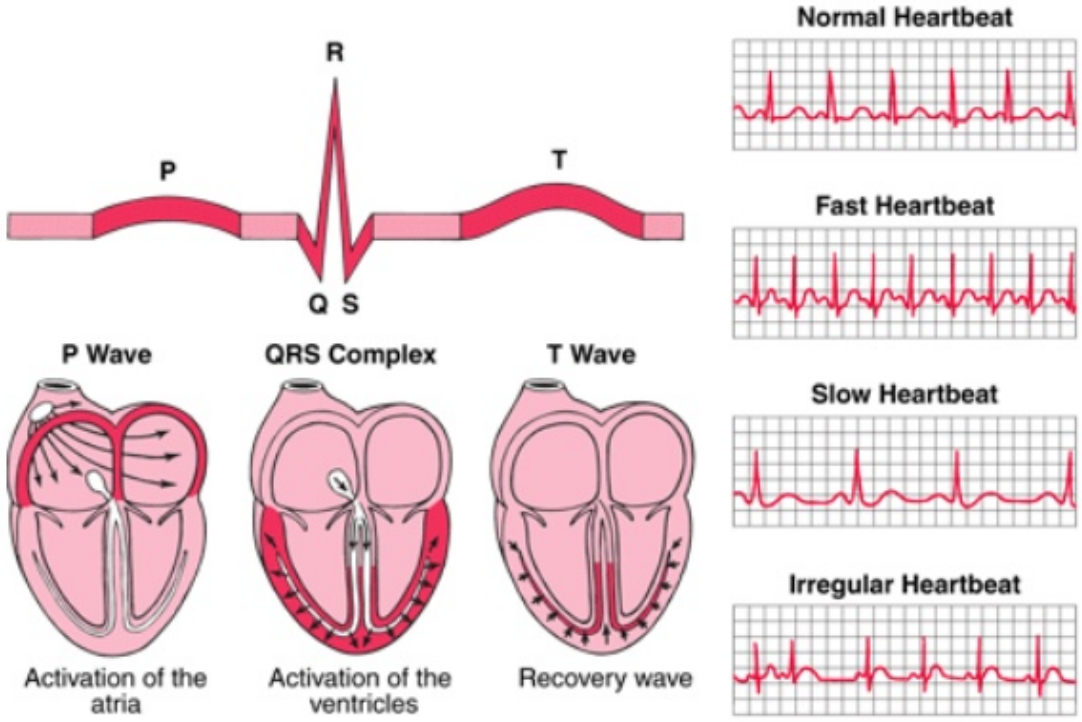


Figure 2.3: ECG signal and some pathological heart conditions that can be derived from the ECG signal.[2]

Taking into account the critical condition of a person suffering from arrhythmia episodes, the field of detecting signs of arrhuthmia in an ECG signal has been highly investigated. Since, the ECG signal is a non-stationary signal, arrhythmias may occur at random in time scale. Therefore, the study of the ECG pattern and heart rate variability signa may have to be processed several hours. This means that an enormous data set needs to be carried out in order for the diagnosis to be reached effectively. Thus, machine learning techniques [13] are ideal for solving the diagnosis problem. The data set is used as the training set required by machine learning solutions, that can deliver a diagnosis after their training is completed.

In most of the studies, MIT-BIH Arrhythmia Database [14] is used as the source of ECG recordings. So, MIT-BIH database is used to form the training set. The MIT-BIH Arrhythmia Database consists of 48 half-hour excerpts of two-channel ambulatory ECG recordings from patients with different medical files and types of arrhythmias, digitized at 360 samples per second per channel with 11-bit resolution over a 10 mV range. Furthermore, two or more cardiologists independently annotated each record and computer-readable reference annotations for each beat was included in the database. As a result, training data sets for the arrhythmias' detection problem was created, using MIT-BIH database. Firstly, a classifier based on machine learning techniques was trained using this

data set and, then, was used to detect arrhythmias in individual beats.

The algorithm of acquiring and processing the ECG signal in order to extract and classify each beat was composed of varius stages. It is divides into three main stages: a filtering stage, a heartbeat detection and feature extraction stage and a classification stage. An overview of the ECG analysis flow is depicted in Fig.2.4.

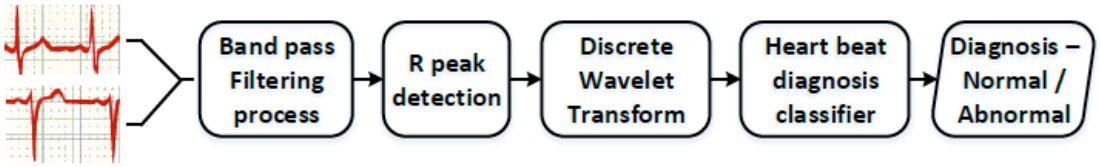


Figure 2.4: ECG Analysis Flow [11]

The following key features are included:

Noise removal.

In this stage the signal is filtered for noise removal, usually using a band-pass filter. The artifact signals that are removed include baseline wander, power line interference and high-frequence noise. Artifacts resulting from patient breathing and movement also have to be removed. The filtered ECG signals are used in all subsequent processing.

R peak detection.

In this stage the ultimate goal is to detect the heart beats that compose an ECG signal of a longer duration. In order to do that usually peaks of the QRS complex have to be identified and possibly P wave, T wave and QRS onsets and offsets [15]. The heart beat exact location and duration can be deducted from this information. MIT-BIH Arrhythmia Database [14] provides the user with function implementations that locate these fiducial points . These functions can be applied to the ECG signals available and the results can be verified with the help of the doctors' annotation files. That way we manage to isolate beats and thus construct the beats which will be included in the training data set. In a real-time ECG acquisition system, the heart beat is not known a priori and points of interest such as the R peak have to be defined relying solely on the QRS detectors available in order for a new heart beat to be identified. Smaller fractions of the ECG signal are now processed for beat segmentation and as a result fewer beats are detected each time.

Feature extraction process.

Having determined a new heart beat, a feature extraction process is imposed on it in order to extract its characteristics. These characteristics refer to speciffic signal parameters

that are indicative of the physiological state of interest. For arrhythmia detection there is a variety of types of characteristics that can be used, each of them introducing various clinical trade-offs. The most complete way to display the information included in the ECG signal is to perform spectral analysis. The wavelet transform (WT), an extension of the classic Fourier transform, can be applied to extract the wavelet coefficients of discrete time signals, such as the ECG. The WT works on both time and frequency domain and allows the decomposition of a signal into a number of scales, each scale representing a particular coarseness of the signal under study. WT also has the ability to compute and manipulate data in compressed parameters which are called features. Thus using the WT transform, the ECG signal, consisting of many data points, can be compressed into a few parameters that characterize its behaviour and are not apparent from the original time domain signal. The heart beats detected at the previous stage, are decomposed into timefrequency representations using discrete wavelet transform (DWT) and wavelet coefficients are calculated to represent the signals [16]. The outputs derived at this stage form the feature vectors that are used for classiffication.

Diagnosis classiffication.

The final stage of the analysis flow is actually detecting whether the heart beat exhibits arrhythmia signs or not. This is performed using a classification algorithm, which detects the pattern of problematic beat. The classiffier has been trained on the data set that includes the feature vectors of the isolated beats. Given a new feature vector the classifier can decide on whether the corresponding beat displays signs of arrhythmia.

Support Vector Machines [5] are the machine-learning based classiffiers that are used in this study. SVMs are popular machine-learning classifiers for data-driven modeling and classification and can be efficiently trained offline. Their training process results in a set of vectors, called support vectors (SVs), which are used to model the data by representing a decision boundary. This decision boundary is then used to classify a new instance, the feature vector of an unclassified beat. The number of support vectors and the feature-vector dimensionality can have a major impact on classifier complexity [9].

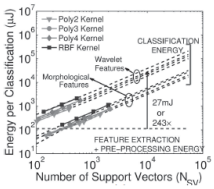


Figure 2.5: Classiffication energy scales with N_sv [11]

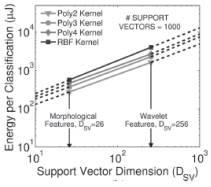


Figure 2.6: Classification energy scales with D_sv [11]

2.2 Related Work

The investigation of the electrocardiogram (ECG) signal has been extensively used for monitoring and diagnosing many cardiac diseases. In recent years, numerous research have been conducted for analyzing and classifying the ECG signal. Most biomedical devices used for ECG analysis have to provide accurate results in real time. Therefore, a large amount of data with extremely complex correlations have to be processed. According to [9], data-driven modeling techniques are emerging as a powerful approach for overcoming the mentioned challenges [18] since a lot of machine-learning techniques have been rapidly developed, that are capable of managing large amounts of data to model specific correlations and then use models within a decision function [17]. Additionally, biomedical devices require very low power consumption,because they are mostly wearable devices. To that direction the writers propose to employ application-specific architectures for low energy [19], [20], where they use arrhythmia detection and the ECG analysis flow, stated in section 2.1. They also use Support Vector Machine based classiffiers in the classiffication stage and RBF (exponential) kernel function in the classiffication core since linear kernels are not suitable for the degree of complexity of the correlation of medical signals. The kernel function along with the number of support vectors and feature vector dimensionality can have a major impact on classiffier complexity. This was proven by implementing the entire arrhythmia detection algorithm on an embedded low-power base processor using high-order detection models for accurate signal classiffication and performing energy analysis. Figures 2.5 and 2.6 show the energy of classiffication versus number of support vectors and feature vector dimensionality, respectively. It can be seen that, because of energy scaling, classiffication energy rapidly dominates that of feature extraction. It is found that classiffication poses the energy bottleneck due to the complexity of the models required.

Thus, in their study they aim to optimize the classification stage in terms of throughput and energy efficieny. To achieve that they explore the development of a co-processor based architecture suitable for the analysis flow of various biomedical signals that require classification. A general-purpose processor is employed for feature computation, while an optimized co-processor is employed for kernel-based SVM classification. The specifications for the platform are meeting the constraints for real-time detection, energy efficiency and

exibility so that it supports various biomedical applications. The architecture has three main blocks: buffers for the support and test vectors, MAC units and a programmable polynomial kernel core. The support vectors are loaded to the buffers after the offline training is finished and the test vectors are dynamically loaded to their respective buffers. Each MAC unit is responsible for the dot product computations of the SVM classifier between a test vector and the support vectors in one support vector buffer. Once multiplication over all the support vectors is complete, the dot products are multiplexed to the programmable polynomial kernel core, where a second-, third-, or fourth-order polynomial transformation is computed as well as the kernel of the SVM classifier selected (such as RBF,sigmoid, etc). The results are scaled and summed by a final accumulator whose output sign determines the classification result. The computations are performed on integer values. The real-time constraints are met by parallelizing the dot product computations with the use of multiple MAC units and energy efficiency is achieved through voltage scaling in the MAC units. Furthermore, in [11], a hardware FPGA co-processor is developed for the classification stage in order to optimize throughput and energy efficiency. They built a co-processor using High Level Synthesis tools and, thus, the architecture is fixed according to kernel function and the data related to the SVM model, such as the number of support vectors and the coefficients, are hardcoded instead of being loaded. Optimization is applied by modifying the structure of the code, increasing instruction level parallelism and utilizing the optimization directives of the tool. In this thesis, the aim is to implement and optimize the ECG analysis flow in terms of latency gain and energy efficiency on an ultra-low-power multicore system-on-chip, Movidius Myriad 2, designed for specific vision processing applications. In order to achieve the optimization, the development of the flow on the VPU Myriad 2 was based on its own architectural characteristics. Myriad 2 processor consists of a total of 14 cores. Taking that in mind and the fact that Myriad 2 comes with a 2MB multiported DRAM memory, make it suitable for implement high computational tasks on it, such as the SVM classifiers. Moreover, the 12 "mini" cores support Single Instruction Multiple Data (SIMD) architecture, allowing computations of 128-bit number to be performed in one cycle (more information about Myriad 2 architecture is described in chapter 4). Taking these characteristics into account, instruction level parallelism was applied, in order to maximize the computations parallelism on each SHAVE and to make it possible of using the SIMD feature of the processor. Additionally, memory system architecture is of great importance in Myriad 2. Thus, the support vectors are mapped in the 2-MB DRAM, in order to achieve greater gain latency through minimizing the SHAVE-memory bridges crossed.

Chapter 3

Theoritical Backround on ECG analysis flow

As mentioned in chapter 2, a typical structure of the algorithm for the heartbeat classification, which is implemented in this study, consists of four main parts. The first lead of the digitized ECG signal is applied as the input to the system. A *filtering* unit is used as a preprocessing stage, to remove baseline wander and noise from the ECG signal. The filtered signal is then passed to the *heartbeat detection* unit, which attempts to locate all the heartbeats contained in the input ECG signal. Since the QRS complex is located, the ECG signal is segmented into single heartbeats. Next in the flow, a *feature extraction* is included, in order to achieve greater classification performance. In this phase, a feature vector is extracted for every single heartbeat with a smaller number of elements. This feature vector is expected as an input for the *classification stage*, where the heartbeat will be labeled as nornal or abnormal by a single classifier.

3.1 MIT Database

As already stated, for the purposes of this study, data from the MIT-BIH arrhythmia database [14] were used. This database is a result of the collaboration of Beth Israel Deaconess Medical Center and MIT, and it is one of the most utilized databases for research purposes.

The database is composed of 48 half-hour excerpts of two-channel (two leads) ambulatory ECG recordings, obtained from 47 subjects. Of these, twenty three recordings were chosen at random from a collection of over 4000 24-hour ambulatory ECG recordings, serving as a representative sample of routine clinical recordings. The remaining twenty five recordings were selected from the same set to include a variety of rare but clinically important phenomena (complex ventricular, junctional and supraventricular arrhythmias), which would not be well represented in a small random sample. The subjects included 25

men aged 32 to 89 years and 22 women aged 23 to 89 years. Approximately 60% of the subjects were inpatients and 40% outpatients. The data are bandpass filtered at 0.1-100Hz and digitized at 360 samples per second per channel with 11-bit resolution over a 10 mV range. In 45 recordings, the first lead is a modified limb lead II (MLII), and for the resting 3 recordings it is lead V5. The second lead is lead V1 for 40 of the recordings, and it is either lead II,V2, V4 or V5 for the other recordings.

The MIT-BIH database also provides annotations for each record, where cardiologists placed a label for every beat detected in the record. There are approximately 110.000 annotations. Table 2.1 lists the heartbeat types included in the database and their mapping to the American Heart Association (AHA) heartbeat classes (N, V, F, E, P, Q and O). Table 2.2 shows the percentages of beat labels that correspond to each heartbeat class. The percentages are disproportionate, as the largest beat class, 'N' (normal beat), covers 84,8% of the beats found in the database.

In the algorithmic analysis that follows, data from the first-channel lead (MLII) of all records of the database were used, apart from records 102, 104, 114, whose first-channel lead is not a MLII. Two arrhythmia groups are examined, 'Normal' (N), and 'Abnormal' (V, F, E, P, Q, O).

3.2 Discrete Wavelet Transform

The Wavelet Transform (WT) is similar to the Fourier transform, with the extension that it is capable of providing the time and frequency information simultaneously, hence giving a time-frequency representation of the signal. This is essential when analyzing non-stationary signals (whose frequency response varies in time), such as the ECG signal, where the time localization of the frequency spectral components are needed. Generally, the wavelet transform can be expressed by the following equation:

$$F(a,b) = \int_{-\infty}^{\infty} f(x)\psi^*_{(a,b)}(x)dx \tag{3.1}$$

where the * is the complex symbol and function ψ is the transforming function called the mother wavelet. Dilation, also known as scaling, compresses or stretches the mother wavelet and translation shifts it along the time axis. The WT can be categorized into continuous and discrete. Continuous, in the context of the WT, implies that the scaling and translation parameters change continuously. However, calculating wavelet coefficients for every possible scale can represent a considerable effort and result in a vast amount of data. Therefore, discrete wavelet transform (DWT) is often used. In the DWT, a time-scale representation of a digital signal is obtained using filtering techniques. Filters of different cutoff frequencies are used to analyze the signal at different scales. The signal is passed through a series of high pass filters to analyze the high frequencies, and it is passed through

AHA heartbeat class	N	V	F	E	P	Q	O
MIT-BIH heartbeat types	N (normal)	V (premature ventricular contraction)	F (fusion of ventricular and normal)	E (ventricular escape)	P (paced)	Q (unclassifiable)	! (ventricular flutter wave)
	L (left bundle branch block)		f (fusion of paced and normal)				P (non-conducted P wave)
	R (right bundle branch block)						
	A (atrial premature)						
	a (aberrated atrial premature)						
	J (nodal premature)						
	S (supraventricular contraction)						
	e (atrial escape)						
	j (nordal escape)						

Table 3.1: Mapping the MIT-BIH heartbeat types to the AHA heartbeat classes [10]

Heartbeat class	N	V	F	E	P	Q	O	total
number of beats	93411	7129	1785	106	7028	33	665	110157
% of total beats	84.8	6.47	1.62	0.096	6.38	0.03	0.6	100

Table 3.2: Beats of full database corresponding to each class [10]

a series of low pass filters to analyze the low frequencies. The resolution of the signal, which is a measure of the amount of detail information in the signal, is changed by the filtering operations, and the scale is changed by upsampling and downsampling operations. Downsampling a signal corresponds to reducing the sampling rate, or removing some of the samples of the signal. Upsampling a signal corresponds to increasing the sampling rate by adding new samples to it (usually zeros or interpolated values). The DWT analyzes the signal at different frequency bands with different resolutions by decomposing the signal into a coarse approximation and detail information. DWT employs two sets of functions, called scaling functions and wavelet functions, which are associated with low pass and high pass filters, respectively.

The decomposition of the signal into different frequency bands is simply obtained by successive highpass and lowpass filtering of the time domain signal. The original signal x[n] is first passed through a halfband highpass filter g[n] and a lowpass filter h[n]. After filtering, half of the samples can be eliminated according to the Nyquist's rule, since the signal now has a highest frequency of p/2 radians instead of p. The signal can therefore be downsampled by 2, simply by discarding every other sample.

This constitutes one level of decomposition and can mathematically be expressed as follows:

$$y_{high}[k] = \sum_n x[n] * g[2k - n] \tag{3.2}$$

$$y_{low}[k] = \sum_n x[n] * h[2k - n] \tag{3.3}$$

where $y_{high}[k]$ and $y_{low}[k]$ are the outputs of the highpass and lowpass filters, respectively, after downsampling by 2. This decomposition halves the time resolution since only half the number of samples now characterizes the entire signal. However, this operation doubles the frequency resolution, since the frequency band of the signal now spans only half the previous frequency band, effectively reducing the uncertainty in the frequency by half. The above procedure can be repeated for further decomposition.

3.3 Background Information on SVM Classifiers

Support vector machines (SVMs) are machine learning algorithms, based on the statistical learning theory, that analyze data and recognize patterns. They are used for classification and regression analysis. Given a set of training examples, each labeled for belonging to one of two categories (supervised learning), an SVM training algorithm builds a model that assigns new examples into one category or the other, making it a non-probabilistic binary linear classifier. SVMs can only handle binary classification problems. Multiclass classification can be obtained through the combination of multiple binary classifiers.

An essential component of SVMs is the separating hyperplane. In a binary classification task, the hyperplane is the geometrical division or separation between the two categories. In a one- dimensional space, this is a single point, in a two-dimensional space a line, in a three-dimensional space a plane. We can extrapolate this procedure mathematically to higher dimensions. The general term for a separator in such a high dimensional space is a hyperplane. The SVM algorithm will try to find the optimal hyperplane, called maximum margin hyperplane that offers the best classification. This is achieved by the hyperplane that has the largest distance to the nearest training-data point of any class, since in general the larger the margin the lower the generalization error of the classifier. The decision of the optimal hyperplane is fully specified by a (usually small) subset of the data which defines the position of the separator. These points are referred to as the support vectors. In order for the SVM to be able to deal with errors in the data by allowing a few misclassiffication, soft margins can be set around the hyperplane. They determine the number of examples that are allowed to push their way through the margin of the hyperplane at a certain distance without affecting the final result.

A SVM is a kernel-based technique that makes use of a kernel function. While the original problem may be stated in a finite dimensional space, it often happens that the categories to discriminate are not linearly separable in that space. For this reason, it is proposed that the original finite-dimensional space be mapped into a much higher-dimensional space, making the separation easier in that space. A kernel function will add a dimension to data, in order to obtain the most optimal classiffication. Any given dataset with consistent labels can be brought into a dimension where it can be linearly separated by a hyperplane.

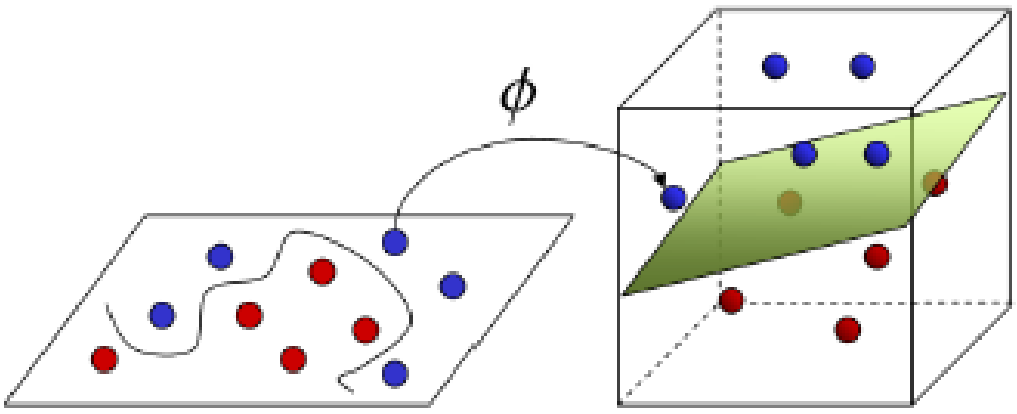


Figure 3.1: Separation of data classes possible in a higher-dimensional space [10]

The hyperplane decision function for classifying feature vector x is of the following form:

$$Class = sgn(\sum_{i=1}^{N\_sv}(y_i * a_i * K(x, sup\_vector_i)) - b) \tag{3.4}$$

where K is the kernel function, x is the feature vector, sup_$vector_i$ is the i-th support vector and y_i, a_i are values related to it and result from the classifier training process. Coefficient b is a bias value, also a result of the training process and is constant for all support vectors. The kernel function is very important to the accurate prediction of testing data. Depending on the characteristics of the dataset, different kernel functions are able to provide the desired classification accuracy.

The most popular kernels are the following:

- linear: K(x, sup_$vector_i$) = $x^T * sup\ \_vector_i$
- polynomial: K(x, sup_$vector_i$) = $\tanh(\gamma * x^T * sup\_vector_i + r)^d$
- radial basis function (RBF): K(x, sup_$vector_i$) = $\exp(-\gamma * \|x - sup\_vector_i\|^2)$
- sigmoid: K(x, sup_$vector_i$) = $\tanh(\gamma * sup\_vector_i + r)$

If the feature vectors of our data set were linearly separable, a linear kernel function could be used for classification. The test vector **x** could be pulled out of the summation in 3.1, allowing the summation to be precomputed over all of the support vectors into a classification energy would remain constant. However, biomedical applications have shown to perform poorly when linear decision functions are used with medical datasets [9]. Non-linear functions, such as high-order polynomials, RBFs, or sigmoidal kernels, are needed for acceptable classifier accuracies.

In this study, RBF kernel function is implemented, since the complex correlations between the attributes of our feature vector and the physiological states of interest typically require the flexibility afforded by non linear kernel functions. The advantage of the RBF kernel over the other non linear kernels is that RBF has fewer parameters and fewer numerical difficulties [22]. The RBF kernel for test vector x and the i-th support vector sup_vector_i is defined as:

$$K(x, sup\_vector_i) = \exp(-\gamma * \|x - sup\_vector_i\|^2) \tag{3.5}$$

Combining the equations , the final function, which will be implemented, is the following:

$$Class = sgn(\sum_{i=1}^{N\_sv}(y_i * a_i * \exp(-\gamma * \|x - sup\_vector_i\|^2)) - b) \tag{3.6}$$

The original kernel code of this equation is provided in C language in 3.1. In **??**, *sv*

_coef for the product of y_i and a_i of equation 3.1. The complexity of the algorithm depends on the number of the support vectors *N_sv* and the length of the feature vector *D_sv*. The support vectors were trained using the Matlab interface of LIBSVM library for Support Vector Machines [21].

Listing 3.1: SVM original kernel code

```
const float sv_coef[N_sv];
const float sup_vectors[D_sv][N_sv];

void SVM_predict (int *y, float test_vector [D_sv]){

for(i=0; i<N_sv; i++){
        for(j=0; j<D_sv; j++){
                diff=test_vector[j] - sup_vectors[j][i];
                norma = norma + diff*diff;
                }
        sum = sum + exp(-gamma*norma)*sv_coef[i];
        norma = 0;
}

sum = sum - b;

if (sum<0)
        *y = -1;
else
        *y = 1;
}
```

Chapter 4

Myriad 2 System-on-Chip Presentation

This chapter describes the functionality and use of the Myriad 2 multiprocessor SoC. In addition, it introduces details regarding the architecture of this chip as well as the programming paradigms it supports.

4.1 Myriad 2 Implementation Board

Myriad 2 is a multiprocessor SoC, that is designed to perform highly computational tasks for mobile, wearable and embedded applications. Myriad 2 incorporates parallelism, instruction set architecture and microarchitectural features to provide highly sustainable performance efficiency for a wide range of applications.

4.1.1 Myriad 2 System Architecture

Most typical application processors systems-on-chip (SoCs) are typically based on one or more 32-bit reduced-instruction-set computing (RISC) processor, surrounded by hardware accelarators, that share a common multilevel cache and DRAM interface. This shared infrastructure is attractive from a cost perspective, but creates major bottlenecks in terms of memory access, where highly computational tasks demand real-time performance but must contend with user applications and a platform OS such as Android. A better solution is to build a software programmable architecture as a coprocessor to an application processor, which can take over all the hard real-time workload, dealing with multimedia systems and accelerometers. As a result, the architecture presented in [6] focuses on power-efficient operation, as well as area efficiency, allowing the programmer to choose between software or hardware implementation of his product.

Myriad 2 as a typical SoC combines two 32-bit reduced-instruction-set-computing(RISC)

processors LEON. Also offers twelve SHAVES to provide exceptional performance efficiency and flexibility. Both 32-bit processors can control the 12 or a number of the intergrated SHAVE processors algorithms. The first LEON is called LEON OS and is designed to communicate with the outside world. LEON OS can control all external peripherals, such as Universal Asynchronous Receiver Transmitter (UART), Serial Peripheral Interface (SPI), ETHERNET, GPIO, USB 3.0, etc, connected to a reduced number of I/O pins using a tree of software-controlled multiplexers. In this way, these interfaces support a broad range of use cases in a low-cost plastic ball-grid array package with integrated 2 to 4 Gbit low-power DDR2/3 synchronous DRAM stacked in package using a combination of flip-chip bumping for the VPU die and wire bonding for the stacked DRAM.LEON OS comes with bigger caches, a L1 cache of 32KB and a L2 cache of 256 KB, making it possible to run a simple RTOS on it. The other LEON, called LEON RT, is responsible of controlling the media devices of Myriad 2, such as camera sensors, LCDs, HDMI controllers etc, as well as, of using high performance video hardware filters for vision processing applications. Hardware accelerators help speed up hard-to-parallelize functions required by video codecs. Up to 12 independent high-definition cameras can be connected to 12 programmable MIPI D-PHY lanes supporting CSI-2 organized in six pairs, each of which can be independently clocked. In order to be able to control all these devices and the dataflow, LEON RT has a 32KB L2 cache and 4KB L1 cache.

Additionally, Myriad 2 comes with a software controlled, multicore and multiported memory subsystem with the size of 2MB, which can support the 12 processors and two RISC processors with high, sustainable on-chip data. In order to be able the processors and hardware accelarators to communicate with the memory and offload data movement between them and a large range of peripherals, a multichannel direct memory access engine is available to move the data between them.

Myriad chip offers Streaming Hybrid Architecture Vector Engine (SHAVE) 128bit Very Long Instruction Word (VLIW) cores with a particular instruction set, able to process high computational tasks. Myriad 2 intergrates 12 SHAVE processors as depicted in Fig.4.1.

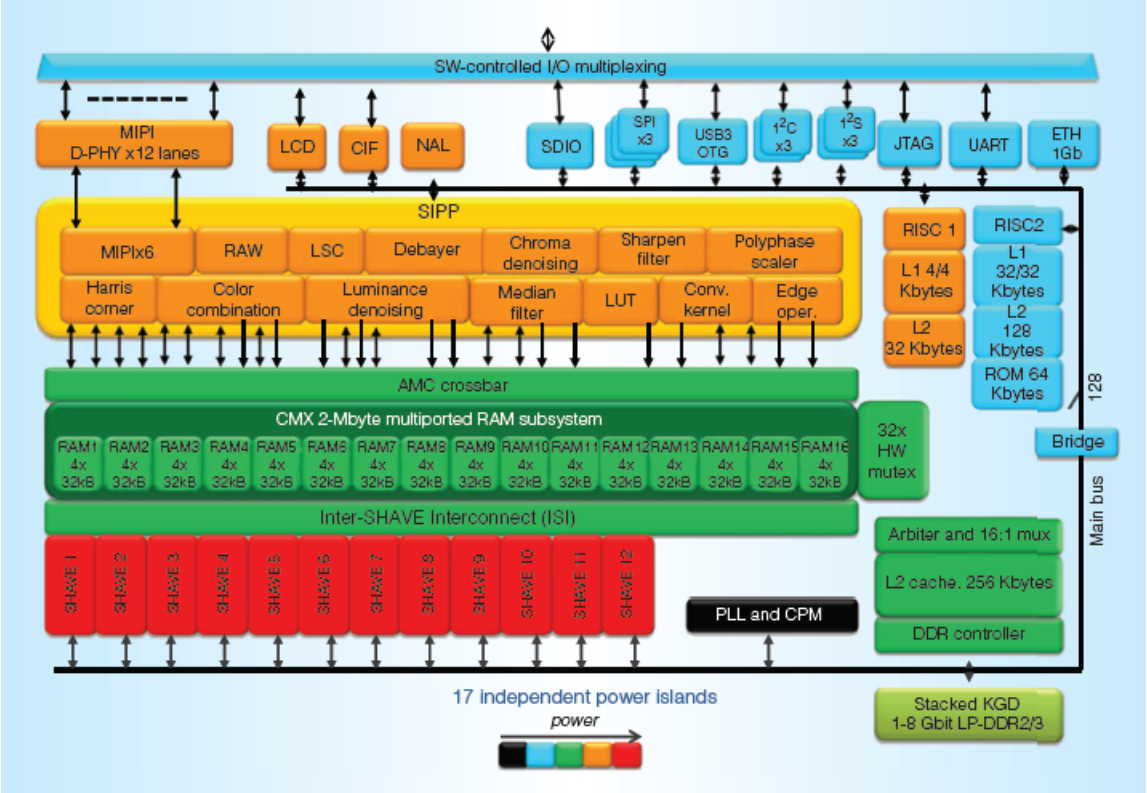


Figure 4.1: Myriad 2 vision processing unit (VPU) system on chip (SoC) detailed block diagram shows the 12 SHAVE cores and associated Inter-SHAVE interconnect with the multicore memory subsystem. Above the CMX memory are the hardware accelarators for computer vision and image processing, as well as other media devices, controlled by the LEON RT reduced-instruction-set computing processor (RISC) processor. Above that, are the I/O peripherals, which are controlled by other RISC processor (LEON OS).[7]

Because power efficiency is an ascendent factor, the implementation board utilizes 17 power islands, including one for each of the 12 SHAVE cores, providing a tailormade power control using software.The device supports 8-, 16-, 32- and 64-bit integer operations as well as fp16 and fp32 arithmetic. Thus, Myriad's architecture offers an increased performance per watt across a wide range of highly computational applications. Myriad 2 family consists of the following two series:

- MA2100, the one we use on this thesis, and
- MA2x5 - MA2150/MA2155/MA2450/MA2455

MA2100 is the first generation Myriad 2 SoC with 500 Mhz system clock and 128 MB DDR2 at 500 Mhz.

As Figure 4.1 shows, there are 3 major architectural units in the Myriad 2 processor: the Media Sub System (MSS), the CPU Sub System (CSS) and the Microprocessor Array (UPA).

The Media Sub System (MSS)

The MSS is the subsystem, that is responsible for the external connections between Myriad 2 processor and the camera sensors, LCDs and HDMI controllers, and, at the same time, for making use of hardware filters, located on Myriad 2. Therefore, MSS consists of the MIPI, LCD, CIF interfaces, the SIPP filters, as long as the AMC block which is designed for making it feasable all these to communicate with the CMX memory.

Main part of MSS is LEON RT RISC processor, which is responsible for all above stated parts. LEON RT has a 32 KB of L2 cache memory, sufficient enough to support the coordination, LEON RT is responsible for. Furthermore, LEON RT is the only one, which can change any parameters of MSS block with the minimum delay due to fewer bridge crossing.

The CPU Sub System (CSS)

The CSS have been designed to be the main communication and control unit with the outside world via the external communication peripherals: I2C blocks, I2S blocks, SPI blocks, UART, GPIO, ETH and USB3.0. The control unit of CSS is the Leon OS RISC processor, but in this block the Leon has much bigger L1 (32 KB) and L2 (256 KB) cache memories, which allows to run a modern RTOS on it. This block also offers an AHB DMA engine for offloading large amounts of data from DDR memory or peripherals to the processors. Beside handling the external interfaces and communication Leon OS could also control SHAVE processors imaging algorithms.

The Microprocessor Array (UPA)

The UPA subsystem consists of the 12 VLIW SHAVE vector processors, the 2 MB CMX SRAM memory and a few other blocks from which we list: the specialized DMA engine, the 256 KB L2 cache memory available to the SHAVE cores. The main purpose of UPA is to support the 2 LEON processors by taking over many imaging or computer vision tasks as well as any other general computation intensive algorithms. Each SHAVE processor has preferential ports into a 128 KB slice of the CMX memory, which will be detailed in a following subsection. As such, 12x128 KB = 1536 KB are preferentially used by SHAVE cores but the remaining 512 KB of CMX memory are generally usable by any other resources. The recommended usage for these 512 KB is for HW SIPP filters usage or Leon OS timing critical code which would otherwise not be able to be kept in DDR.

4.1.2 Streaming Hybrid Access Vector Engine processor overview

To achieve high performance and low power consumption, Myriad 2 employs 12 SHAVE processor, which contain wide register files with couples with a Very Long Instruction Word

(VLIW). VLIW packets can control multiple functional units, which support SIMD for high parallelism and throughput. Each one of these SHAVE cores can be launched in parallel. SHAVE's architecture is shonw in Fig. 4.2.

SHAVE supports SIMD instruction on multiple type, such as 16 bits integer, 32 bits integer, 16 bits float and 32 bits float. Assembly, C and C++ can be used for SHAVE programming. As a result, SHAVEs are easy to program and can manage higher performance per watt.

As mentioned before, VLIW packets can control multiple functional units, supporting SIMD for high parallelism. The functional units are 32-bt Integer Arithmetic Unit, 32-bit Scalar Arithmetic Unit, 128-bit Vector Arithmetic Unit, which supports 8, 16, 32-bit of both floating point and integer types. Besides, there is a 128-bit Compare Move unit (CMU), load-store units and predicated execution units. Each functional unit can be used either from a Vector Register File with 32 registers of 128 bits each or from Integer Register File with 32 registers of 32-bits each. The SHAVE processors have access both to L1 and L2 cache memories, 1 KB and 256 KB respectively. L2 cache memory can be divided up to 16 of 16 KB each.

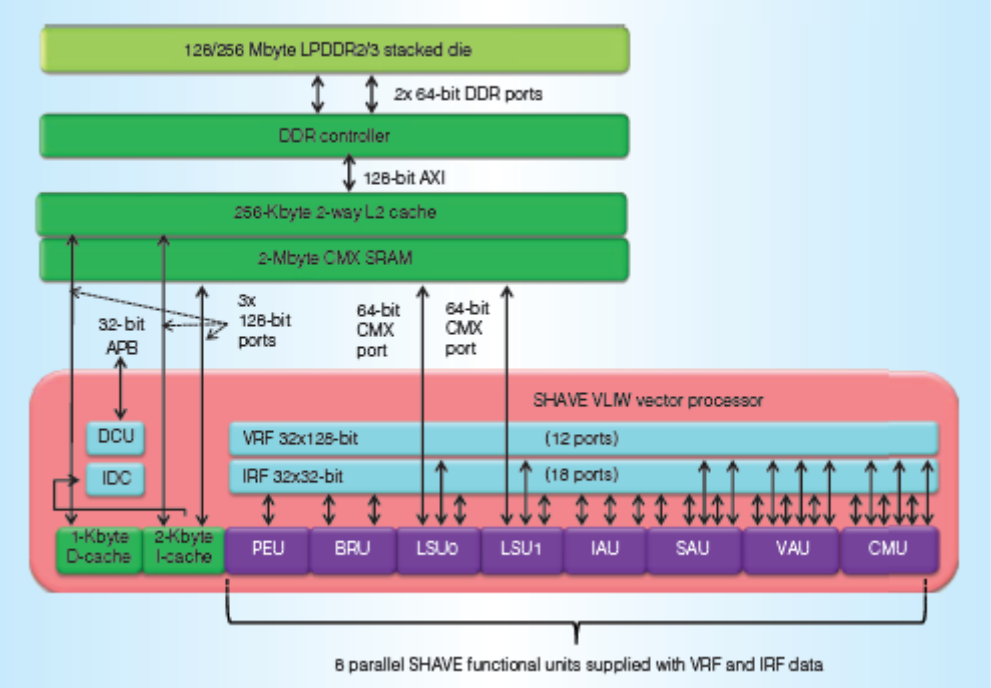


Figure 4.2: The Streaming Hybrid Architecture Vector Engine (SHAVE) core microarchitecture, able to perform 128-bit vector load/store, floating-point, integer and control-flow operations in one clock cycle [7]

The SHAVE processors are easily programmable using C, C++ and/or Assembly languages, and being compiles via Movidius internally developed tools. Multiple vector types are supported by Movidius C/C++ compiler, such as int4, uint4, float2, short4 etc. Additionally, numerous standard C and C++ libraries are delivered along with the Movidius

compiler.

Multiple core systems are usually equiped with internal locks, in order to give exclusive access to a particular resource for a single processor. So, Myriad 2 comes with eight mutexes, thus SHAVEs can predicate-stall on mutex availability in order to avoid multiple memory accesses by multiple processors.

The SHAVE processor is typical used for performing intensive computational tasks. Usually, data would be loaded from DDR to CMX memory piece by piece and it would be processed there. In order to achieve the best results and minize the development time, C level design is prefered for control code on SHAVEs and optimized routines for the inner loops of the computational tasks. It is better to let LEON handle various interrupts, while SHAVEs are processing the data. Typically, the Streaming Image Processing Pipeline (SIPP) engine is handled by the SHAVE processors, in order to achieve optimized scheduling of its functionality.

4.1.3 Memory Overview

The most ascedent factor for image processing and computer vision algorithms is the ability of combing pipelines of hardware and software. Therefore, the connection, as well as, data sharing between SHAVE processors and hardware accelerators was the key of designing Myriad 2 multicore system-on-chip. In order to achieve that, Myriad 2 was designed around a 2-MB multiported memory block called Connection Matrix (CMX) memory(as shown in Fig.4.3), which can be configured to manage different instructions and workloads.

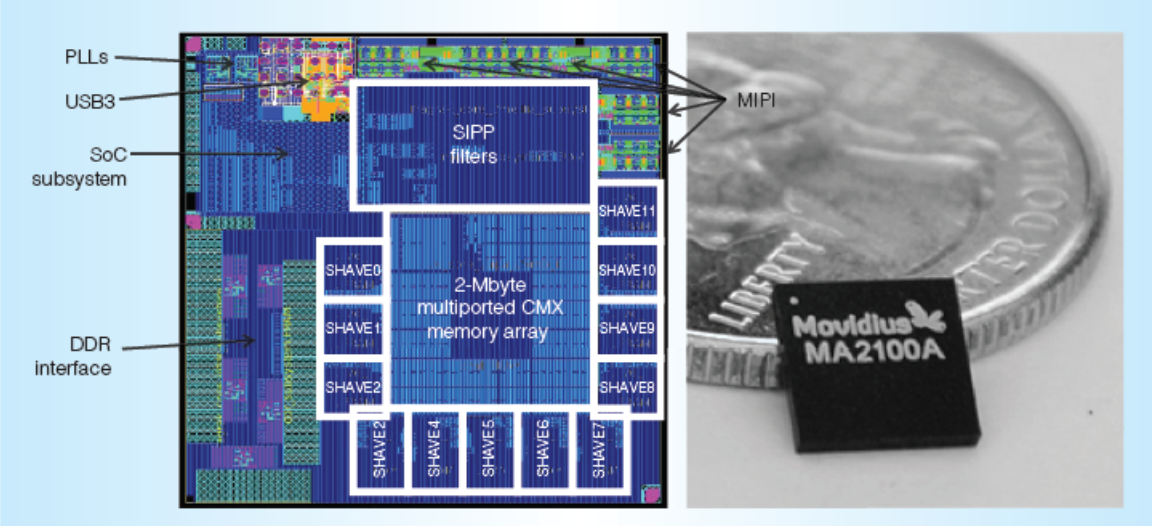


Figure 4.3: Myriad 2 VPU SoC die plot and 6 x 6 mm ball-grid array packaging. As you can see, SHAVE processors and hardware accelerators are built around the CMX memory subsystem. By the SHAVE processors and the SIPP filters, the DRAM interface, MIPI, USB and other peripherals was placed, with the remainder of the die plot being occupied by the RISC processors and other subsystems.[6]

The CMX memory can be divided to 16 blocks of 128 KBytes, which are independently arbitrated, allowing each RAM block to be accessed independently. Theoritical, the 12 SHAVE processors can move 12 X 128 bits of code and 24 X 64 bits of data. This software-controlled memory can be configured to allow many workloads to be handled, providing high sustainable on-chip bandwidth of 307 GBps to support data and instruction offload to 12 SHAVE processors. Additionally, the CMX memory can support multiple traffic classes for latency-tolerant hardware accelerators to latency-intolerant SHAVE vector processors, allowing construction of pipelines from a mix of software running on SHAVEs and hardware accelerators, which can perform simultaneously without performance loss [6]. Finally, each SHAVE has higher bandwidth and lower power access to "its" own local slice. The time penalty of accessing different slice is about 10 percent of SHAVEs running time, as measured in lab. As such, each SHAVE is more energy-efficient to its local slice, and this is worth keeping in mind at design time for optimal performance.

4.2 Myriad 2 Basic Programming Paradigms

There are three different programming paradigms supported by Myriad 2 platform. Each one offers features suitable for specific applications and is selected based on the relevant requirements.

Standard Programming Paradigm

The standard programming paradigm for Myriad 2 involves using RTEMS running on LeonOS and the SIPP scheduler on Leon RT. In this way, it provides parallelization in an environment that is easily used and configured. The SIPP scheduler is designed to ensure parallel pipeline configurations for managing the HW filters and exterior interfaces with a low footprint. Hence, LeonRT optimized utilization is guaranteed. The number of SHAVEs used for SIPP applications is configurable, so those that are not used for line based pipelines will remain free to be used by the RTEMS operating system running on Leon OS for various other purposes such as computer vision algorithms.

The One Leon Programming Paradigm

This paradigm is suitable for applications that might not require heavy line based processing. Such applications might choose to make the Leon RT processor completely

inactive and instead use only the LeonOS with or without RTEMS. HW filters can still be used in this paradigm. In this programming model, Leon OS would control all of the applications running on the 12 SHAVE cores.

Bare Metal Programming Paradigm

It is often wanted by developers to write applications which will not be affected by any operating system overhead. For this reason, a bare metal programming paradigm is also be supported by Myriad 2. This allows the usage of both LEON cores without any operating system. Only minimal schedulers are required in order to control the pipelines application. Even though this paradigm requires more integration efforts, it offers the developers a model in which operating systems cost is absent.

4.3 Myriad 2 Applications

The capabilities that were described previously make this embedded platform ideal for various computer vision applications, as stated in [6]. The SHAVE DSP supports streaming workloads from the ground up, making decisions about pixels or groups of pixels as they are processed by the 128-bit VAU. Because 128-bit comparison using the CMU and predication using the PEU can be performed in parallel with the VAU, higher performance can be achieved compared to a GPU in which decisions must be made about streaming data because GPUs suffer from performance loss due to branch divergence. Futhrermore, Myriad 2 System-on-Chip is capable of running machine learning algorithms, due to high level of parallelism that can be achieved, high workloads that can be processed and all above Myriad 2 features that help accelerate any data analysis, such as VLIW and float4 or float16 data type support.

For example, the SHAVE processor excels when processing the FAST9 algorithm, where 25 pixels on a Bresenham circle around the center pixel must be evaluated for each pixel in, for instance, a 1080p frame. The FAST9 algorithm looks for nine contiguous pixels out of 25 on a Bresenham circle around the center that are above or below a center-pixel luminance value, meaning hundreds of operations must be computed for each pixel in a high-definition image. This requires hundreds of instructions on a scalar processor, and performance optimization requires the use of machine learning and training to improve detector performance [15].

In general, the application domain of Myriad 2 is Intelligent Machine Vision with some examples being (Source: www.movidius.com) :

Robotics

Drones and household robots are increasingly small and affordable enough to become serious consumer product categories. As new types of service, companion and collaborative robots emerge, these devices are demanding visual intelligence in order to navigate, understand and proactively assist us in our daily lives. Movidius provides the platform to create visually intelligent drones and robots without sacrificing size, battery life or performance.

Augmented and Virtual Reality

Virtual Reality (VR) and Augmented Reality (AR) devices are hitting the market and technological demands on the hardware are huge: gesture recognition, head tracking and object recognition are just a few of the necessary technologies to convincingly blend the real world with the digital. Myriad 2 allows VR and AR devices to crunch huge amounts of data at low power and ultra-low latency, two absolute musts in compact, immersive head-worn devices.

Wearables

Wearables are emerging as a category of devices that can augment our lives in meaningful ways. By passively filtering visual information and acting on cues relevant to their user, the dream of a truly capable digital assistant is in sight. Ultra-low power, high performance vision processors mean that even the smallest wearable devices can benefit from visual intelligence. The Myriad 2 platform allows devices to remain small and battery efficient, yet provide powerful new applications based on the rich variety of visual information available as users go about their daily lives. Smart Security Security and surveillance technology is getting a huge boost from visual intelligence. Imagine, a doorbell camera that not only alerts you to a visitor, but has already identified them as a courier. Visually intelligent cameras can detect fires from heat maps and alert authorities long before a fire builds up enough smoke to trigger a smoke detector; and motion detection cameras will be able to differentiate potential burglar from house pet. By bringing Myriad's visual intelligence to our security and surveillance, these new systems can detect and then intelligently act on data in real-time, providing safe and personalized security to homeowners and businesses alike.

Smart Security

Security and surveillance technology is getting a huge boost from visual intelligence. Imagine, a doorbell camera that not only alerts you to a visitor, but has already identified them as a courier. Visually intelligent cameras can detect fires from heat maps and alert

authorities long before a fire builds up enough smoke to trigger a smoke detector; and motion detection cameras will be able to differentiate potential burgler from house pet. By bringing Myriad's visual intelligence to our security and surveillance, these new systems can detect and then intelligently act on data in realtime, providing safe and personalized security to homeowners and businesses alike.

Chapter 5

Implementation of ECG Analysis flow on Myriad 2

In this chapter we introduce the main focus of this thesis, the software acceleration of the ECG Analysis flow algorithm. As mentioned above, ECG analysis flow algorithm consists of 4 different parts, the Low-pass Filter, QRS detection algorithm, discrete wavelet transform and the support vector machine classifier. In this study, we try to accelerate the low pass filter and the SVM classifier, as their execution time is 80x times slower than the QRS detection and the discrete wavelet transform.

In the following sections, the C-code was implemented from scratch on Myriad 2 and the architecture capabilities of Myriad 2 are presented, in order to achieve the best possible performance of the ECG analysis flow. The results of this thesis, as well as, the development step, that were taken, are detailed below.

5.1 Initial porting

In the first approach, the original code was tested on the LEON RT. Standard programming paradigm has been used and the main block diagram of the flow is shown in Fig.5.1. The execution starts from the LeonOS processor, which runs the RTEMS operation system. This processor, then, boots up the LeonRT, and simultaneously, starts measuring the execution time, as well as, the power and energy consumption of the code. That decision was made, in order the time measurement and the code execution to be independent. In turn, LeonRT starts executing the ECG flow analysis, using the SHAVEs to accelerate the filter, as well as, the SVM classifier. More details about the development will be given in each of the following steps. In the first stage of the implementation, we try to develop the filter execution time.

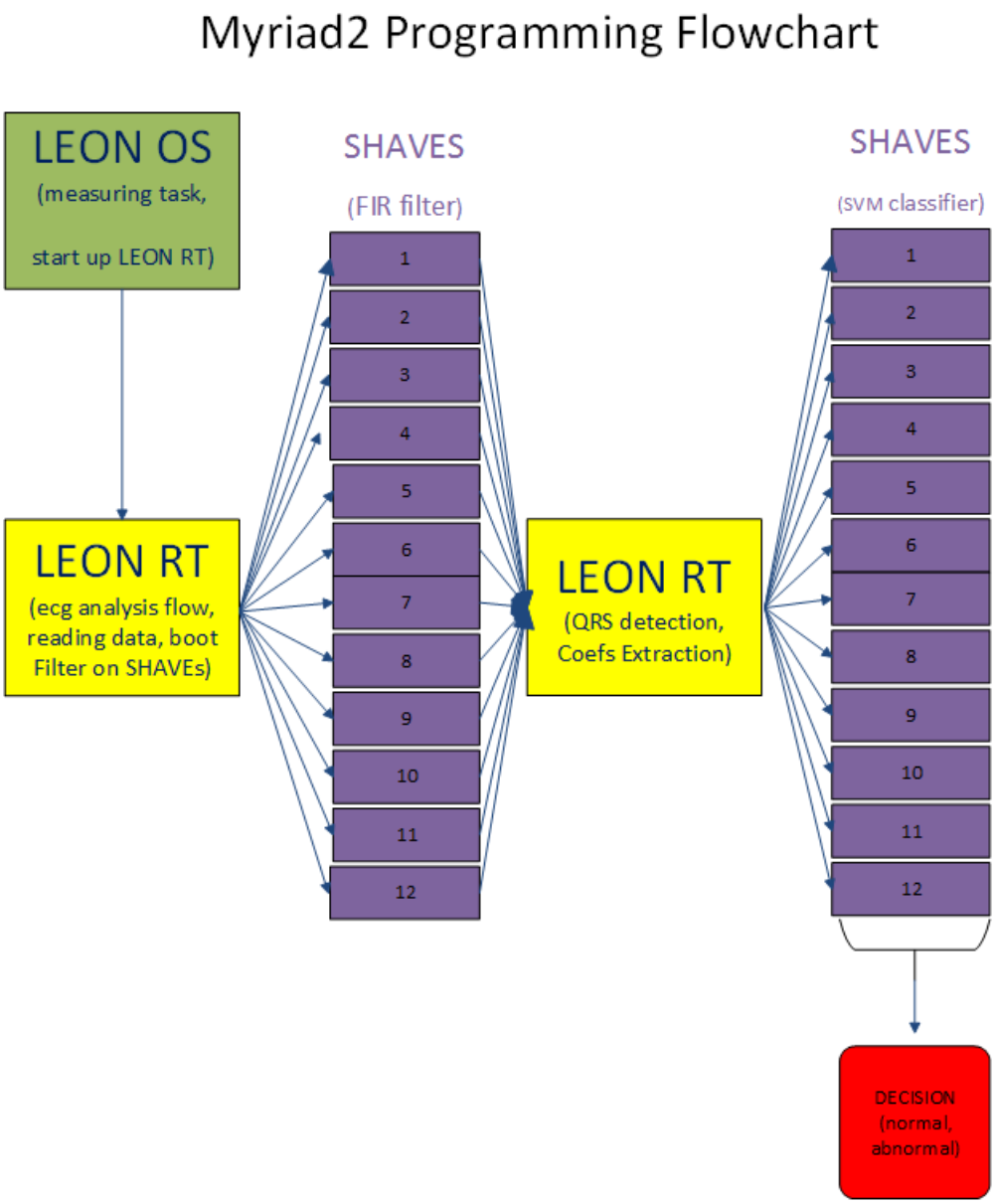


Figure 5.1: ECG Analysis Flowchart on Myriad 2

5.2 Lowpass filtering process

Noise removal filter had a large share of the execution time of the original code, about 40 percent of the total execution time. An extended signal window is normalized, and then, the signal is filtered, by calculating the Eucledes distance of the low-pass filter coefficients and the signal.

At first, we implement the original code on Myriad 2, composed by a band-pass filter. After an extensive analysis, it was decided that a removal of the high pass filter had no impact on the decision's accuracy of the analysis flow. The removal of the high pass filter

resulted to a 50% latency gain, compared to the original code. The C-code of the low pass filter is provided below in Listing 5.1.

Listing 5.1: Low pass filter main

```
for (j=0; j<len;j++){
        ext_sig[j] = (bufferedSig[j]-baseline)/gain;
}

for (i=len; i<len+D_l; i++){
    ext_sig_low[i] = 0;
}
for (n=0; n<(len+D_l); n++){
  y_low[n]=0;
  for (i=0; i<llen; i++){
    if ((n-i)>=0) {
                y_low[n]=y_low[n]+Num_low[i]*ext_sig[n-i];
    }
  }
}
for (i=D_l; i<(D_l+len); i++) {
                        y_low[i-D_l] = y_low[i]*gain + baseline;
}
```

The signal window's form, which is processed by the filter, is shown below.

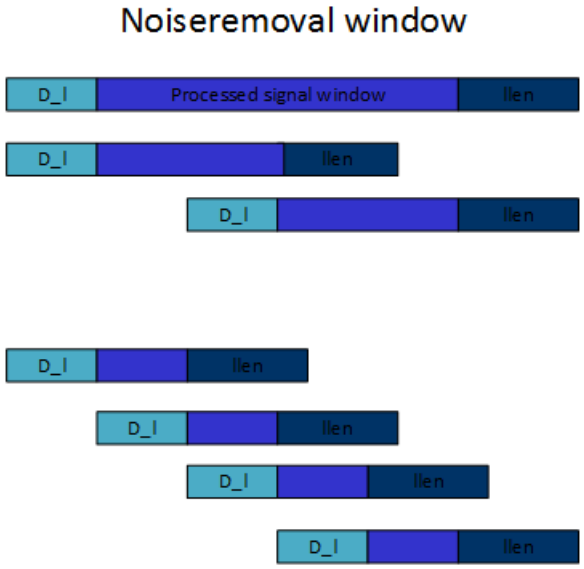


Figure 5.2: The filtered window of the signal is smaller than the input signal window of the noiseremoval filter, because of the functionality of the low pass filter. As a result, the input signal window can not be divided. So the execution time is not the expected time.

Noiseremoval window can not be divided in just half parts, since the filter loses its precision and depth and does not behave the same way, when running on multiple SHAVEs, resulting on wrong classification results. Therefore, the input signal window keeps its original form, only the processed part is being divided, depending on how many SHAVEs are being used (Fig.5.2).

At first we try to increase the vectorization of the filter algorithm. To do that successfully, a good understanding of the algorithm is required. In that direction, a detailed explanation of the filter is provided below.

In the code in Listing 5.1, Array *Num_low* represents the features of the lowpass filter. Array *Ext_sig* contains the normalized pre-filtered signal, the input of the noiseremoval algorithm. Its size depends on the sampling frequency of the ECG analysis flow. *D_l* is the depth of the filter and *llen* represents the size of the lowpass filter. The array *y_low* contains the newly filtered signal, which is returned back to the LEON RT as an output. At the beginning of the filter algorithm, *gain* and *baseline* are parameters of the flow, which is used in order to normalize the ECG signal around zero.

According to chapter 5, the signal has to be multiplied with the filter's coefficients in order the filtered signal to be computed. The resulting algorithm is an array of sums of the multiplication of input signal and the lowpass filter's features. Each time in every loop this sum is computed for every sample of the input signal of the filter. In order a code to be vectorizable, it should be transformed, so as the data in the memory to be accessed continiously. Each filtered signal's sample is calculated, using the *D_l* previous samples. As a result, the computations can be performed simultaneously, although, in every partition, we need to take a window of the previous samples of the signal as input, in order to deduct the same results. This is illustrated in Fig 5.2, where the noiseremoval window is presented, as a combination of the *D_l* window, the processed window(SHAVE window) and the *llen* window.

This is the main idea of this parallelization technique. The signal, that will be filtered, can be partitioned in every shave, each one containing fewer signal samples. Different part of the signal is given as input in the filter. So, multiple samples of the filtered signal can be computed simultaneously and independently. The calculated samples of the signal is returned to different addresses and their combination result to the filtered signal. Thus, we have accomplished to extract coarse level parallelism from the initial problem and the same result is being solved faster. The coarse level parallelism results are presented in Chap.5.4.

So far, we examined how to parallelize the ECG analysis flow algorithm in every SHAVE core on Myriad2, by performing the computations simultaneously. In our next step, we tried to exploit the architectural features of Myriad2 processors, and specifically, the microarchitectural features of the SHAVE cores. As it was mention in Chapter 4, SHAVEs support SIMD instructions on multiple data types. Very Long Instruction Words can

control multiple functional units, such as 128-bit Vector Arithmetic Unit, which supports 8, 16, 32-bit of both floating point and integer data types. Each Vector Register File consists of 32 register of 128 bits each. In our algorithm, signal samples are of floating point data type. Thus, we tried to modify the code structure in order to achieve 4 floating point(float4*) vectorization. 32-bit floating point vector size was implemented, as high computational precision was demanded and this size was proved to fit better in flow's requirements.

At first, we examined the code to find any data dependences or data locality, in order to create opportunities for further optimizations, such as vectorization of the array assignments. It was noticed that, in the second loop (i_loop), the access of the array elemenents of *y_low*, *ext_sig* and *Num_low* was not in the same order. Thus, we chose to make a loop interchange between the inner loop(n_loop) and the outer loop(i_loop). This optimization technique will allow the use of the VLIW features, as not only the elements are accessed in the same order, which are present in memory, improving the locality of reference, but also vectorization of the mathematical operations is now feasible. The result of the application of the loop interchange on the nested loop and array access are displayed in Fig. 5.3

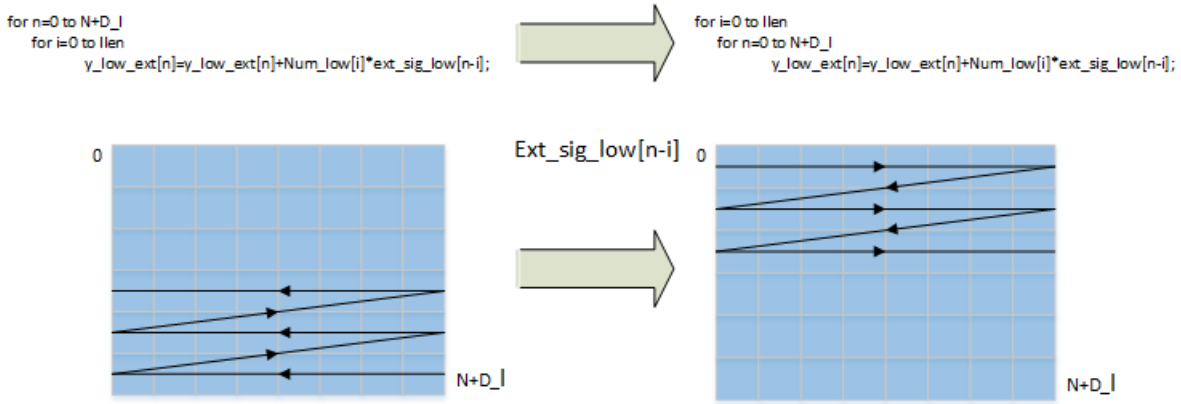


Figure 5.3: Loop interchange and array access

Subsequently, explicit vectorization techniques was applied on the filter's code. First, we vectorized the normalization process of the filter. We built 2 arrays of 4 integers, one for the *gain* and one for the *baseline*. Building these arrays of integers allows to vectorize the first loop. Accordingly, we vectorized the main loops of the filter. After applying loop interchange, we built an array of 4 floats, containing the same coefficient of the lowpass filter. Then, we implemented explicit vectorization on the arrays. The final code and mathematical processed is displayed in Listing 5.2.

Listing 5.2: Final filter code

```
float    gain[4] =          {gain,gain,gain,gain};
float    baseline[4] =      {baseline,baseline,baseline,baseline};

float4* v_8 = (float4*)&(gain[0]);
float4* v_9 = (float4*)&(baseline[0]);
for (j=0; j<noiseremoval_window;j+=4){
        float4* v_1 = (float4*)&(bufferedSig[j]);
        float4* v_2 = (float4*)&(ext_sig[j]);
        *v_2 = ((*v_1) - (*v_9))/(*v_8);
}

int k = D_filter;
for (i=0; i<llen; i++){
        float Nlow[4] = {Num_low_360[i], Num_low_360[i],
                         Num_low_360[i],Num_low_360[i]};
                float4* v_nl = (float4*)&(Nlow[0]);
        if((i-D_filter)>0){
                D_filter=i;
        }
        for (n = D_filter;n < noiseremoval_window;n++){
           if(i==0)
                   y_low[n]=y_low[n+1]=y_low[n+2]=y_low[n+3]==0.0;
           float4* v_7 = (float4*)&(y_low[n]);
           float4* v_6 = (float4*)&(ext_sig[n-i]);
           *v_7 = *v_7 + (*v_nl)*(*v_6);
           n+=3;
        }
        D_filter = k ;
}

float4* v_3 = (float4*)&(sig_filt[D_l]);
float4* v_4 = (float4*)&(y_low[D_filter + D_l]);

for (i = D_filter + D_l; i < D_filter+ D_l + input_window; i+=4){
        *v_3 = ((*v_4)*(*v_8)) + (*v_9);
        v_3++;
        v_4++;
}
```

5.3 SVM classification

After optimizing the FIR filter, we try to accelerate the Support Vector Machine classifier, the most power and time demanding port of the ECG analysis flow. The original code of the SVM classifier is presented in chapter 3, in Listing 3.1.

Firstly, a detailed explanation of the SVM kernel code classifier is required for the better understanding of the reader. In the code in Listing 3.1, test vector represents the feature vector of the pulse to be classified and has been created at the previous stage of the ECG analysis flow, feature extraction. It is implemented as an array of D_sv elements, as many as the attributes of interest are. The classification and differentiation of beats into normal and abnormal is based on these chosen features of the pulse. Array sup_vectors contains the support vectors of the hyperplane that divides the space into two classes. It has N_sv columns, one for each support vector, and each column-support vector has D_sv elements-attributes. Array sv_coef holds the values of the coefficients of the support vectors, and thus has N_sv elements, one for each support vector. Constant b is also a parameter of the derived SVM classifier and represents the bias to which the final result is compared, so as to decide the class to which the current beat belongs.

According to the decision function given in Chapter 3, the squared euclidean distance between the test vector and each support vector is computed and then the RBF kernel function is applied on it. This value is then multiplied by a weighting factor equal to the coefficient of the current support vector. The resulting value is finally added to the total sum, which is compared to the bias in order for the class to be deducted. The contribution of each support vector to the total sum is irrelevant to the contribution of the other support vectors. This means that there are no data dependencies regarding the computations performed between the test vector and each column of the support vector's array. As a result, these computations can be performed simultaneously. This is illustrated in Fig.4.1 where the use of different colours indicates that the computations performed between each coloured column and the test vector can happen in parallel with the computations of the other columns.

Just like in filter section, array sup_vectors can be partitioned in smaller arrays, containing fewer support vectors. These arrays will be of the same size, since the number of attributes does not change. Each array will contribute a partial sum to the total sum used for classiffication. The calculations required for the partial sums to be computed can be performed in parallel. We have thus managed to divide the initial large problem into smaller ones, which are then solved at the same time. Each smaller problem is not a subtask of the initial task but the same task performed on a smaller dataset. All the smaller-scale problems are processed independently and simultaneously and their results are combined for the final computations. Thus, we have accomplished to extract coarse level parallelism from the initial problem. The coarse level parallelism is presented schematically in Fig 5.5.

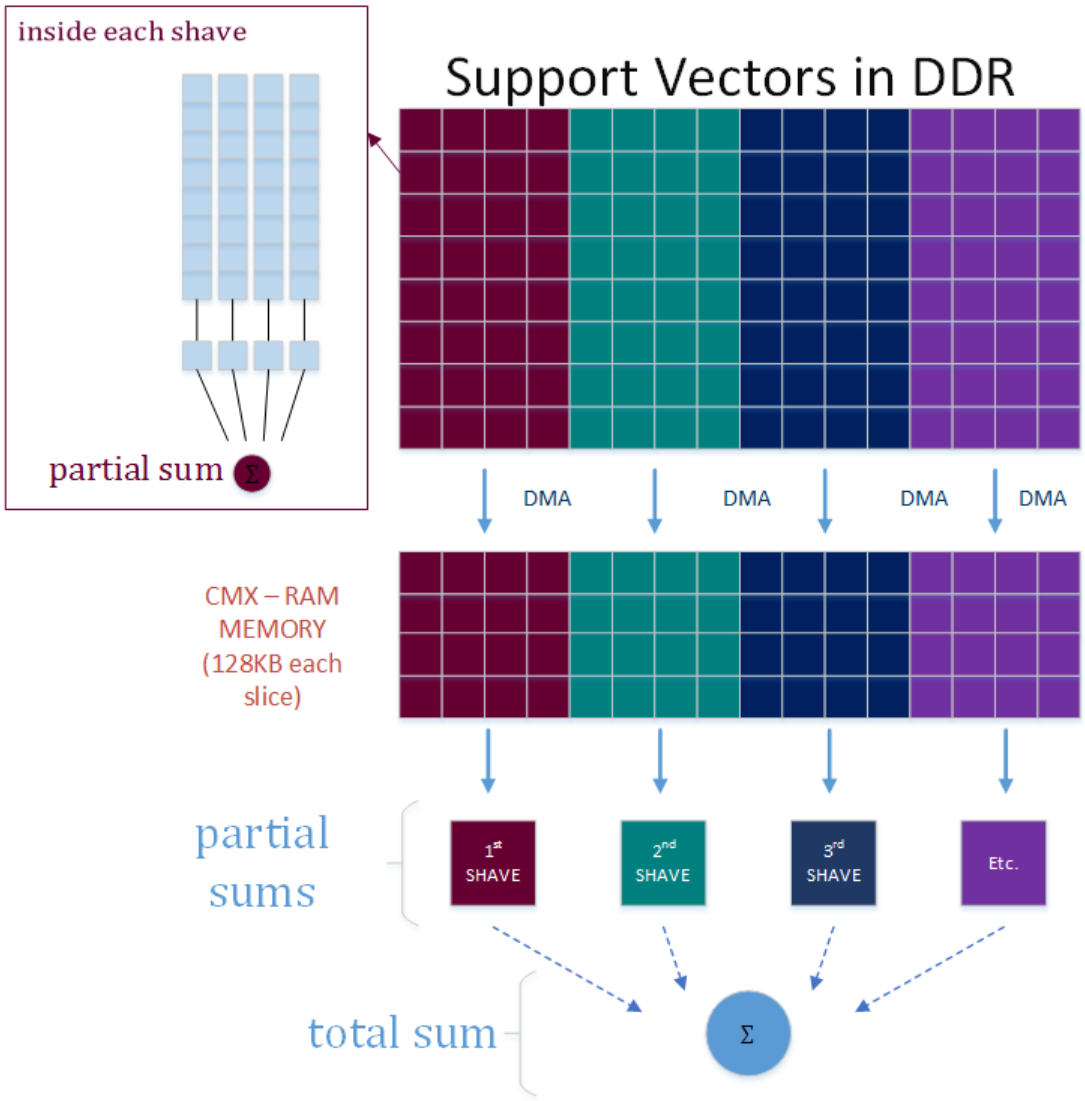


Figure 5.4: Coarse Level Parallelism 2

In the initial porting of the SVM code in Myriad 2, the main part of the code, that is responsible for computing the total sum, is going to be implemented as a separate function, running on SHAVEs. That function will be called by the LEON processor, as many times as many SHAVEs run each time. A different partition is assigned each time on each SHAVE core. Every SHAVE returns the partial sum of its partition to LEON processor, where the total sum is computed and the classification result is exported. Array sup_vectors is initially saved in the DDR memory and its elements are read and written directly from the memory, which means thousands memory accesses resulting to increased execution time and huge power consumption.

Firstly, because array sup_vectors can not fit into each SHAVE's memory slice in our first try on Myriad 2(MA2100 model), the array is transfered into CMX/RAM memory using Direct Memory Accesses(DMAs). DMAs should not being used in abundance, because they consume not only time, but also power. So, we tried to reduce the number of DMAs.

At first, we increase the number of the elements, that are transfered into the memory. As a result, we achieve to reach 100 DMA transfers from 5140 initially. But that idea was not enough, as DMAs are still too many. Our next thought was to implement a double-buffer transfer. Thus, we create an array that is half-loaded with the first support vectors at the beginning of the execution of the code. As the support vectors are processed, the second half of the array is filled with the elements, which will be processed next. This way, only 1 DMA is going to delay the execution of the code, and the execution time will be minimum.

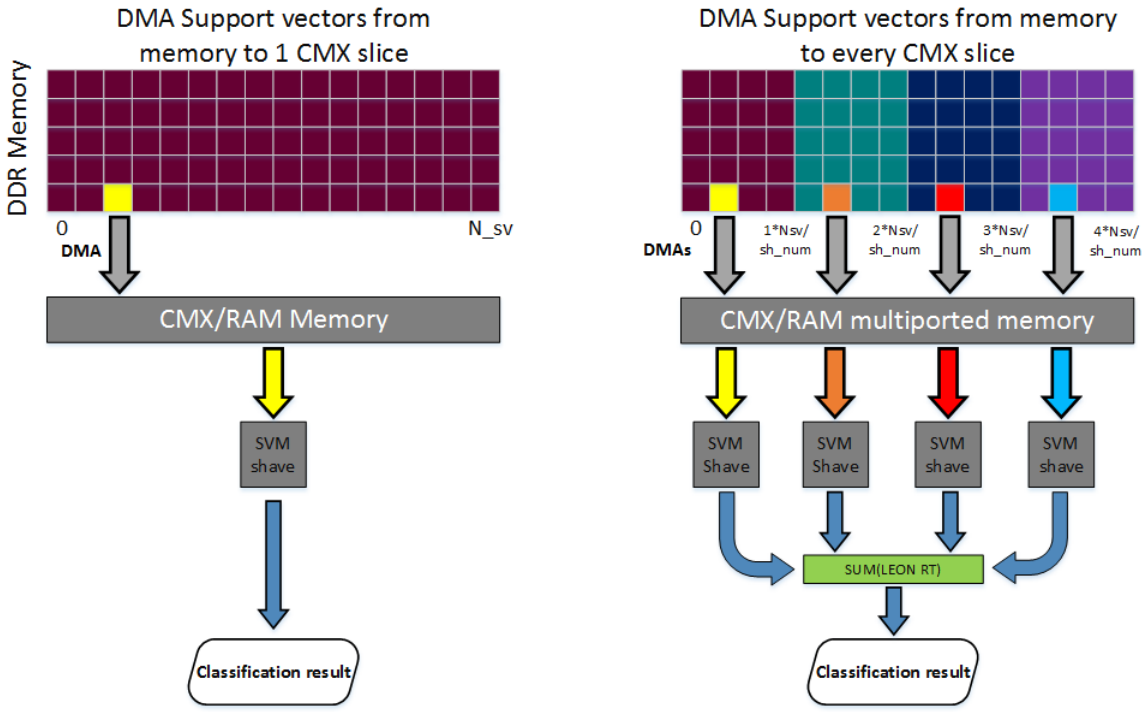


Figure 5.5: Coarse Level Parallelism

Normally, each SHAVE has 128KB of memory in the CMX, 96KB for data usage and 32KB for its code. However, this size is not enough to fit the support vectors array in one or two SHAVEs. Therefore, we have used a configuration LD script that changes the memory that is available for each SHAVE only in Myriad2 MA2150 model. So, since only a single SHAVE was used at first, we could allocate to it the size of the CMX memory, that can afford the size of the sup_vector array. Now, there is enough memory for the data, which are cached into the CMX memory, and so, DMAs are not required.

Below a part of the ldscript, that makes the described configuration, is avalaible in Listing 5.3. The memory, allocated to all SHAVEs, can be seen in this script. LENGTH is the only thing, that we have to change, in order to change the memory size of each SHAVE. CMX memory addresses start from 0x70000000, while the addresses, that point to DDR, start from 0x80000000.

Listing 5.3: LD script Memory Configuration

```
MEMORY
{
SHV0_CODE (wx): ORIGIN = 0x70000000 + 0 * 128K, LENGTH = 32K
SHV0_DATA (w) : ORIGIN = 0x70000000 + 0 * 128K + 32K, LENGTH = 96K

SHV1_CODE (wx): ORIGIN = 0x70000000 + 1 * 128K, LENGTH = 32K
SHV1_DATA (w) : ORIGIN = 0x70000000 + 1 * 128K + 32K, LENGTH = 96K

SHV2_CODE (wx): ORIGIN = 0x70000000 + 2 * 128K, LENGTH = 32K
SHV2_DATA (w) : ORIGIN = 0x70000000 + 2 * 128K + 32K, LENGTH = 96K

SHV3_CODE (wx): ORIGIN = 0x70000000 + 3 * 128K, LENGTH = 32K
SHV3_DATA (w) : ORIGIN = 0x70000000 + 3 * 128K + 32K, LENGTH = 96K

SHV4_CODE (wx): ORIGIN = 0x70000000 + 4 * 128K, LENGTH = 32K
SHV4_DATA (w) : ORIGIN = 0x70000000 + 4 * 128K + 32K, LENGTH = 96K

SHV5_CODE (wx): ORIGIN = 0x70000000 + 5 * 128K, LENGTH = 32K
SHV5_DATA (w) : ORIGIN = 0x70000000 + 5 * 128K + 32K, LENGTH = 96K

SHV6_CODE (wx): ORIGIN = 0x70000000 + 6 * 128K, LENGTH = 32K
SHV6_DATA (w) : ORIGIN = 0x70000000 + 6 * 128K + 32K, LENGTH = 96K

SHV7_CODE (wx): ORIGIN = 0x70000000 + 7 * 128K, LENGTH = 32K
SHV7_DATA (w) : ORIGIN = 0x70000000 + 7 * 128K + 32K, LENGTH = 96K

SHV8_CODE (wx): ORIGIN = 0x70000000 + 8 * 128K, LENGTH = 32K
SHV8_DATA (w) : ORIGIN = 0x70000000 + 8 * 128K + 32K, LENGTH = 96K

SHV9_CODE (wx): ORIGIN = 0x70000000 + 9 * 128K, LENGTH = 32K
SHV9_DATA (w) : ORIGIN = 0x70000000 + 9 * 128K + 32K, LENGTH = 96K

SHV10_CODE (wx): ORIGIN = 0x70000000 + 10 * 128K, LENGTH = 32K
SHV10_DATA (w) : ORIGIN = 0x70000000 + 10 * 128K + 32K, LENGTH = 96K

SHV11_CODE (wx): ORIGIN = 0x70000000 + 11 * 128K, LENGTH = 32K
SHV11_DATA (w) : ORIGIN = 0x70000000 + 11 * 128K + 32K, LENGTH = 96K

CMX_DMA_DESCRIPTORS (wx) : ORIGIN = 0x78000000 + 12*128K , LENGTH = 3K
CMX_OTHER (wx): ORIGIN = 0x70000000 + 12*128K + 3K, LENGTH = 128K - 3K
DDR_DATA (wx) : ORIGIN = 0x80000000 , LENGTH = 128M
LOS (wx) : ORIGIN = 0x80000000 + 100M , LENGTH = 3*128K-10K
LRT (wx) : ORIGIN = 0x80000000 + 100M + 3*128K-10K , LENGTH = 10K
}
```

In the previous paragraphs, we examined ways to amplify parallelization by modifying the structure of the original code and ways to minimize the memory accesses' effect. These are first level optimizations. This means that the performance can be further improved by combining these modification with the features of the SHAVE cores.

The directives chosen for optimization differ from one application to another, depending on the nature of the algorithm under study. For the SVM classifier, the selection of the applied modifications is based on the instruction level parallelism that can be accom-

plished. Most of the directives used, aim at optimizing the inner loop that computes the euclidean distance between the test and support vector. As mention above, the squared differences that compose the euclidean distance can be computed in parallel, as there are no data dependences between each support vector. That means that the inner loop can be vectorized. As a result, we try to evaluate the best vectorization techniques.

The optimization techniques applied on the classifier are loop unrolling and scalar expansion. The first one performs loop unrolling, a well known standard transformations, that duplicates loop bodies by a given unroll factor. In this study, a factor of 2 and 4 (64- and 32-bit floating point values) was tested, in order to achieve the required precision of the floating point operations. The second technique serves to split accumulators within unrolled loop bodies into multiple variables that are added after the loop execution.

Eventually, the compiler could automatically vectorize the inner loop by the factor of 2. However, in the case of 4-element vectors, it could not. This concept requires the addition of *padding* to each support and test vector to ensure that they are multiples of 4. At the end, Taylor expansion of the exponential function was used, in order to achieve further development.

$$e^x = \sum_{k=1}^{\infty} \frac{x^k}{k!} = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$$

Figure 5.6: Taylor Expansion of Exponential

The main advantage of this strategy is that the accuracy of the exponential function can be controlled by varying k . In the limit (k towards infinity) the sum converges to the exact value of the exponential function. However, this approach has a very slow convergence rate for increasing values of k, unless x is close to zero. In our example, the x is very close to zero and k is chosen to be 5. Even though, k is not big enough, the declination between the approximation and the exponential function of libc.h is about 10^{-5}, which does not affect the results of our code. The results of every step and method are presented in section 5.4.

5.4 Results

In this section, the result of the optimizations applied on the ECG Analysis Flow are presented. Firstly, the results of the implementation of the filter are displayed in Fig.5.7. *LEON-DDR* is the initial implementation on LEON RT. Afterwards, *SHAVE-CMX* is the code on SHAVEs, as well as, the filter's features are on CMX-RAM memory. The third column shows the execution time of the vectorized code, and, the last column shows the execution time of the filter running on 12 SHAVEs. After our optimizations, we managed to reach a 17x speedup, regarding the execution time of the filter.

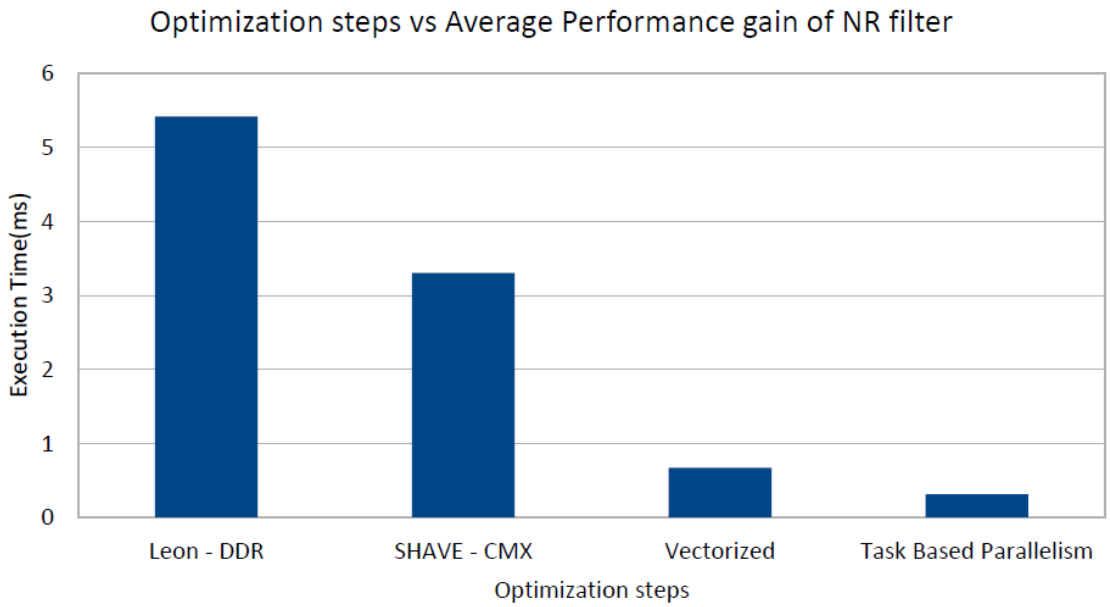


Figure 5.7: Optimization steps vs Average performance gain of the Noise removal filter

Afterwards, in Fig. 5.8, the performance gain of the SVM classifier can be derived. After the initial porting, the Support Vectors are transfered from DDR to CMX memory, using DMA. The next step is to reduce the DMA transfers. The fifth column displays the execution time of the classifier, when the Support Vectors are loaded directly to CMX memory and the last displays the SVM classifier running on all SHAVEs. The speedup, compared to the initial implementation, is about 105x.

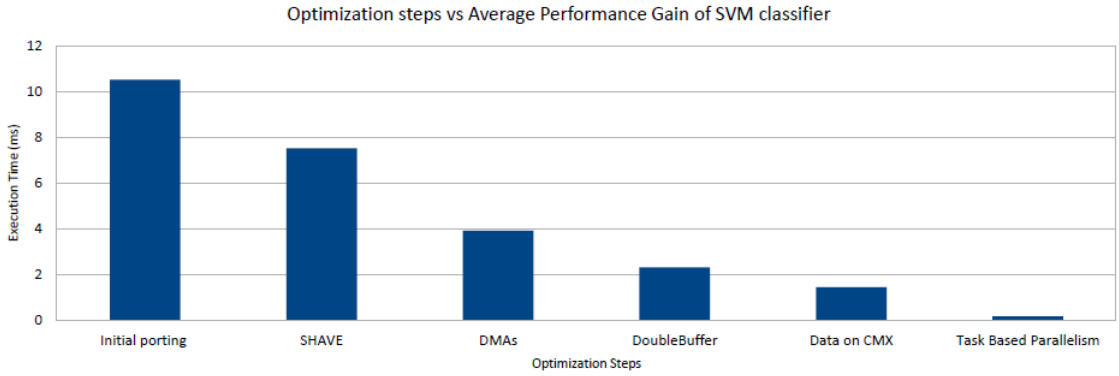


Figure 5.8: Optimization steps vs Average performance gain of the SVM classifier

In Fig.5.9, the average instant power consumption of the ECG analysis flow can be seen. The maximum power consumption is below 950mW, and the minimum is about 450mW.

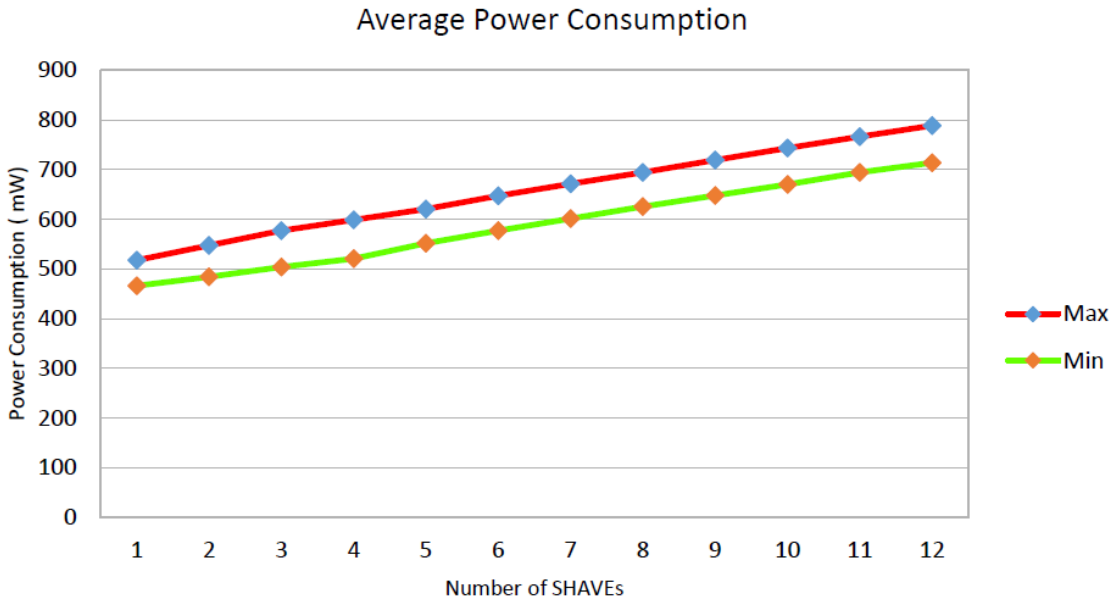


Figure 5.9: Average power consumption w.r.t SHAVEs used

Below, in Fig.5.10, the energy efficiency gain is shown. The consumpted energy in different configurations during the development of the ECG analysis flow can be seen. The energy reduction is about 85%, compared to the initial version of the ECG analysis

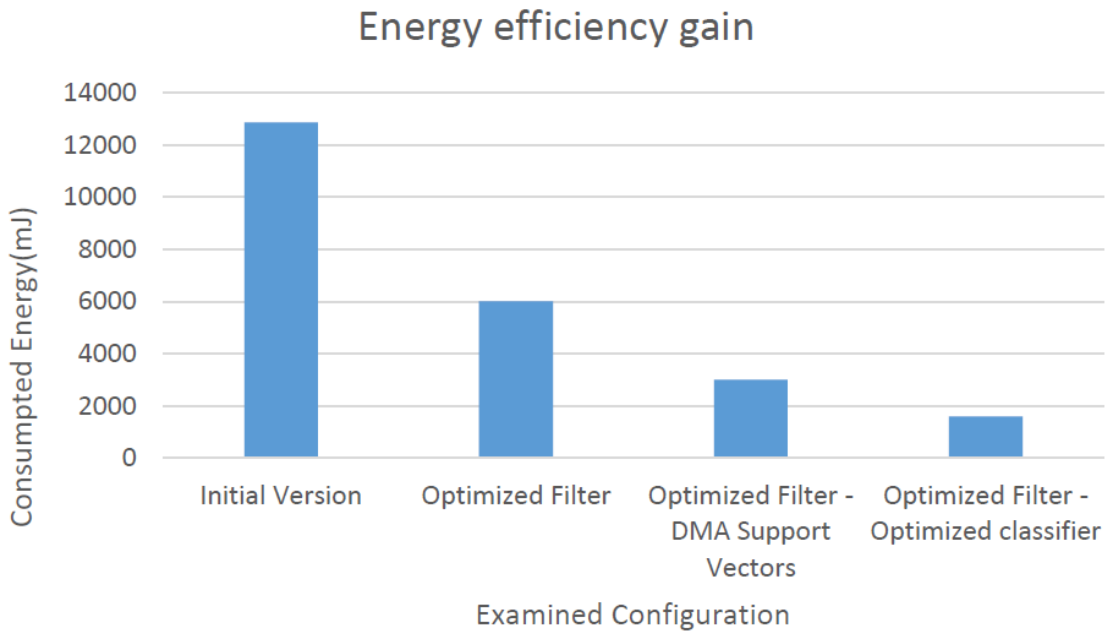


Figure 5.10: Energy Efficiency Gain in different examined configurations

In Fig.5.11 and in Fig.5.12, we can see the ECG Analysi flow execution time and energy consumption per SHAVEs used. The deduction, that can be derived from these two graphs, is that when more and more SHAVEs are used, the execution time reduction is not beneficial in terms of energy. As it can be seen, the minimum energy consumption is achieved, when 6 SHAVEs are employed.

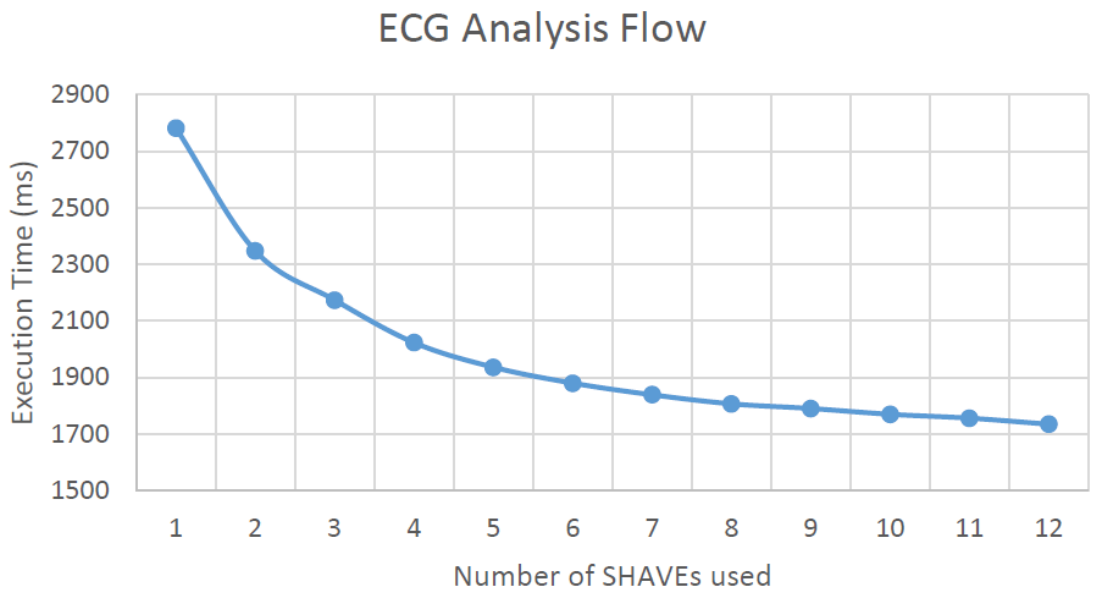


Figure 5.11: ECG Analysis flow execution time w.r.t. SHAVEs used

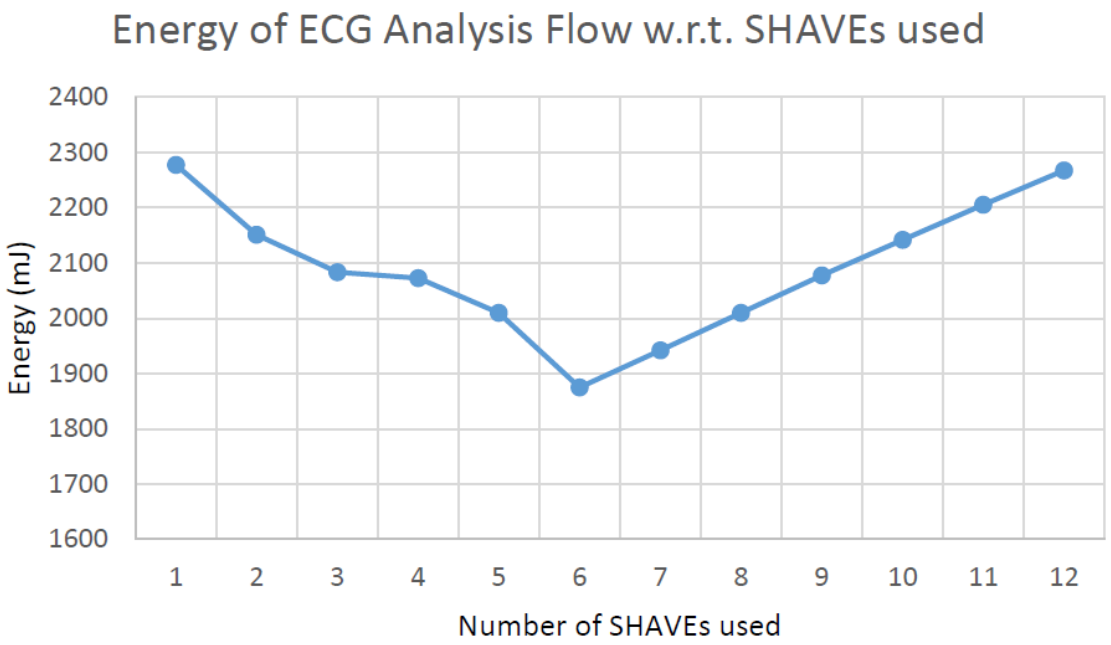


Figure 5.12: ECG Analysis flow energy consumption w.r.t. SHAVEs used

In the next graph, Fig.5.13, the execution time of the SVM classifier on different implementation boards is displays. As we can see, the execution time on Myriad 2 is very low and comparable with the execution time on a HLS based HW implementation.

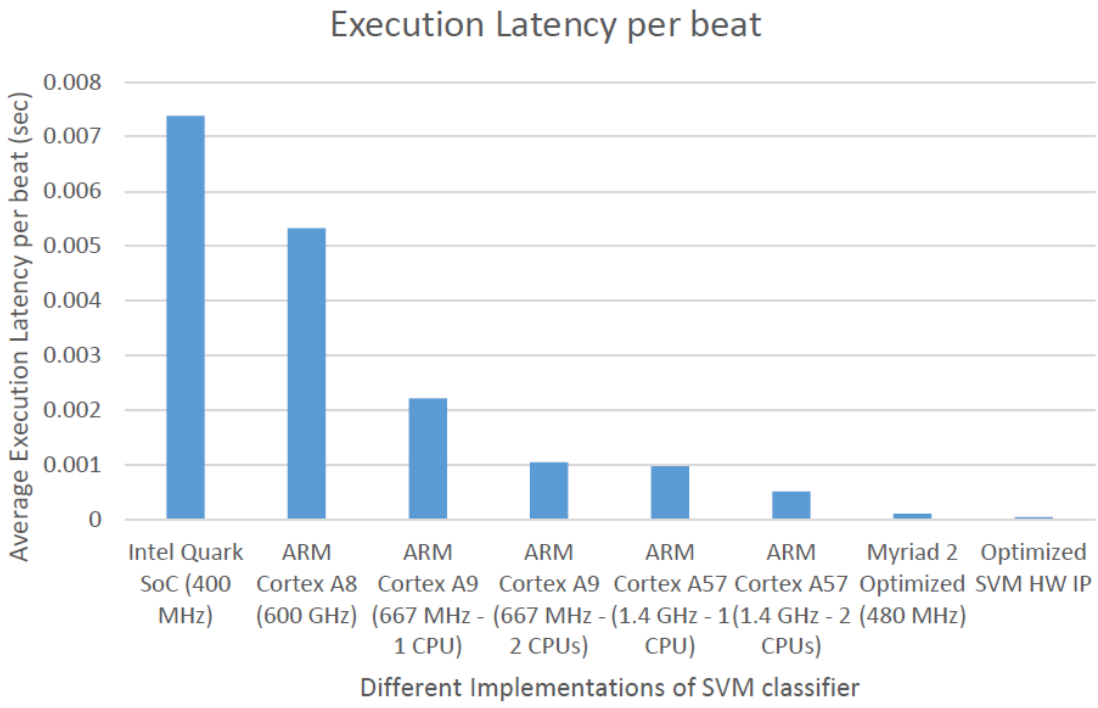


Figure 5.13: Execution time of SVM classifier on different implementation boards

Specifically, in Fig.5.14 and in Fig.5.15, a more detailed comparison between Myriad 2 and Zynq-7000 is presented. The execution time on Myriad 2 is still double the execution time on Zynq-7000, but the instant power consumption is 80% lower than the Zynq-7000.

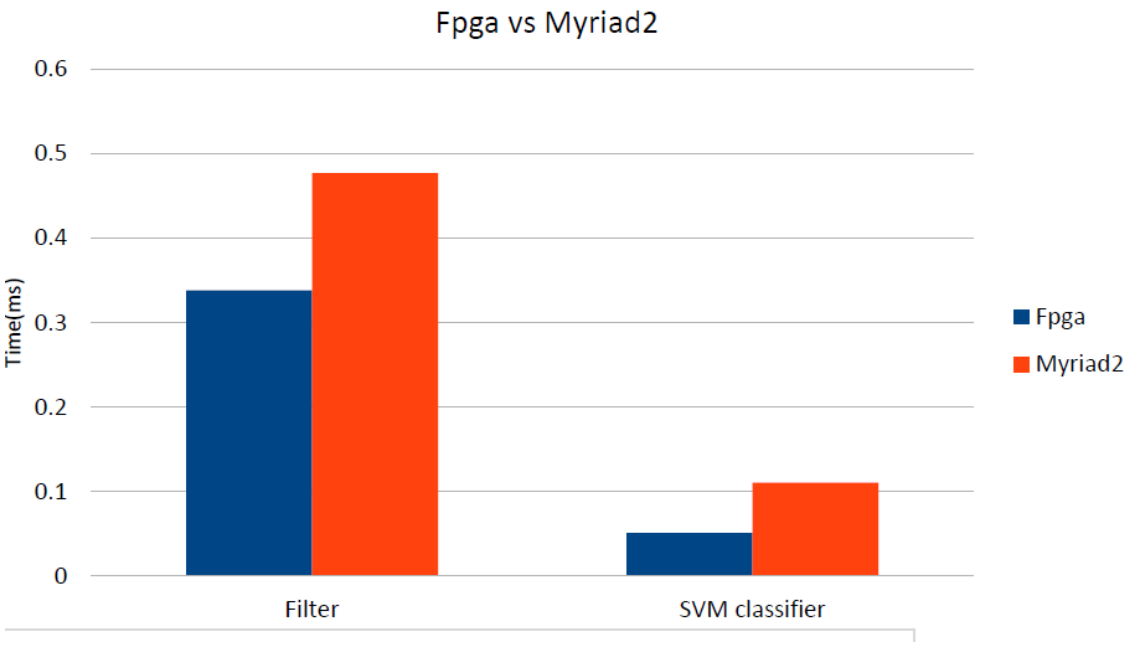


Figure 5.14: Execution Time of SVM on Myriad 2 and Zynq-7000

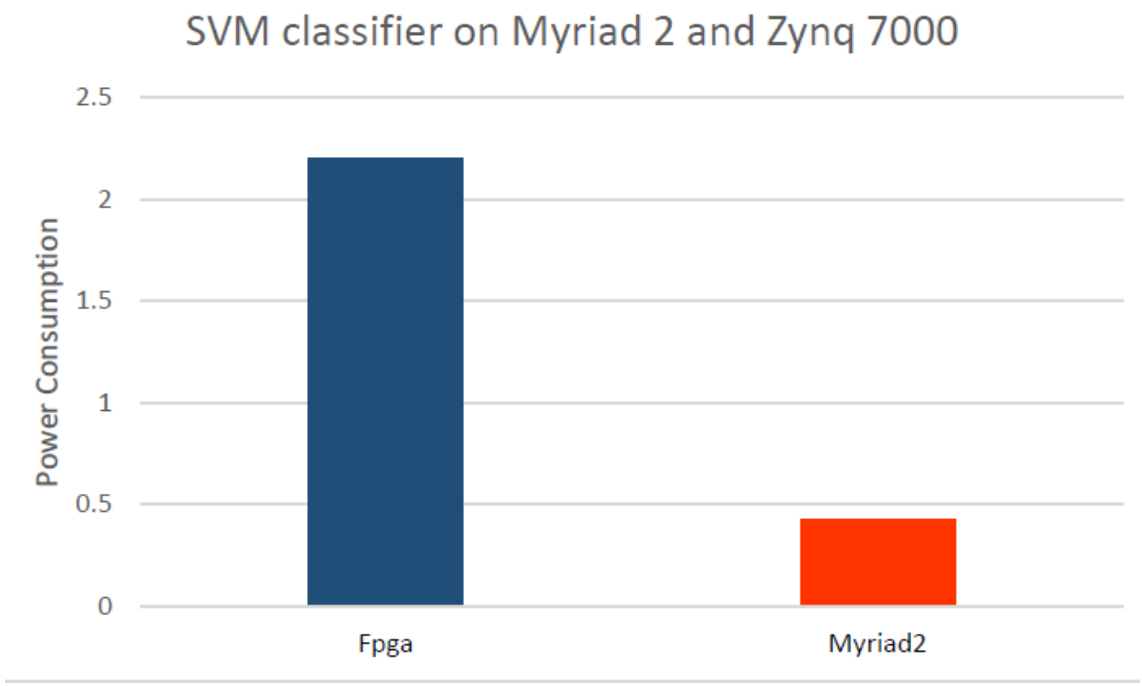


Figure 5.15: Instant power consumption of the SVM on Myriad 2 and Zynq-7000

Chapter 6

Conclusion

6.1 Summary

Arrhythmia detection for chronic patients suffering from various cardiovascular problems requires constant monitoring by recording the ECG signal and thus processing an enormous data set characterized by complex non-linear distribution among its samples. Given the complexity of deriving exact models for assessing the ECG signals and predicting the heart's condition, machine learning techniques have recently dominated the field of ECG analysis. Support Vector Machines particularly are widely used as classifiers and are often incorporated in ECG arrhythmia detection flow. In the detection algorithms, classification is found to pose the primary energy and performance bottleneck and is thus targeted for optimization.

In this thesis, we examined a methodology for developing and optimizing an ECG Analysis Flow into embedded architectures. As a case study, we worked on efficiently implementing the ECG Analysis flow algorithm in a platform specialized mainly for vision tasks, Movidius' Myriad 2. Accelerating electrocardiogram analysis algorithm and deep learning algorithms into an embedded architecture is an interesting concept for many scientists and software engineers who have access to relevant software libraries for general purpose machines and want to know the effort needed to deploy them into an embedded platform. The methodology relies, firstly, on structurally transform the code, in order to parallelize it on many cores, and, secondly, on implementing manually modifications to the code, in order to assist the compiler to build a vectorized code or to vectorize the code explicity. The main challenges of this implementation were posed by the constraints in the memory size, a common feature of embedded systems. Apart from modifications that reduced the memory overhead of the application, several optimizations have also been applied in order to reach high efficiency levels.

6.2 Future Work

Nowadays, valid and timely diagnosis of a patient's problem is of utmost importance. In order to be able to run the complex algorithms needed for such tasks, the traditional ways of performance improvement on these devices have to be reconsidered. Moore's law is slowing down which is causing the power and performance benefits in transitioning to the next process technology node to decrease. Therefore, designers of embedded platforms have to come up with more artful ways of creating powerful and at the same time, power efficient devices.

In this thesis we worked with a device which was designed in this way, based on the understanding that there is a deep interdependency between algorithms and chip architecture. Myriad 2 is a device specialized for efficiently performing machine vision tasks. However, it was used in this thesis to develop a ecg analysis algorithm, the ECG Analysis flow, that includes also machine learning algorithm. Myriad 2 is very promising for accelerating Deep Learning algorithms. The basic techniques described in this thesis can also be used to improve other similar applications. Even though such algorithms were developed to run mainly on supercomputers, they can now be ported into powerful embedded devices and be used on real time basis.

In addition, an interesting step forward would be to further accelerate Support Vectors Machine classifier, using builtin assembly programming language. Additionally, apart from the parts of the filter and the SVM classifier, it would be interesting to optimizing the other parts of the algorithm, as well, on multicore systems and apply techniques, in order to take advantage of the SIMD features. The feature extraction stage that is based on Discrete Wavelet Transform would be an ideal candidate.

Finally, the ECG analysis flow could be entirely on the SHAVEs, so that the algorithms of different sampling frequences can be used simultaneously.

Bibliography

[1] Heart, `en.wikipedia.org/wiki/Heart`, 2017-05-28

[2] Normal and Abnormal ECG readings, `galura-basa.blogspot.gr/2011/06/normal-and-abnormal-ecg-readings.html`, 2017-06-12

[3] Human heart physiology, `anatomybody101.com/human-heart-physiology/human-heart-physiology-heart-physiology3/`, 2017-06-15

[4] Evangelos B. Mazomenos, Dwaipayan Biswas, Amit Acharyya, Taihai Chen, Koushik Maharatna, "A Low-Complexity ECG Feature Extraction Algorithm for Mobile Healthcare Applications", *IEEE Journal of Biomedical And Health Informatics*, VOL. 17, NO. 2, March 2013, pp 459-469

[5] C. Cortes and V. Vapnik, "Support-vector networks", *Machine learning*, vol.20, no. 3, pp. 273-297, 1995

[6] B. Barry, D. Moloney et al., "Always-on Vision Processing Unit for Mobile Applications", *IEEE Computer Society*, 2015

[7] D. Moloney et al., "Myriad 2: Eye of the Computational Vision Storm", *Hot Chips 26*, 2014

[8] R.J. Martis et al., "Characterization of ECG beats from cardiac arrhythmia using discrete cosine transform in PCA framework", *Knowledge-Based Systems*,45 (2013)

[9] M. Shoaib, N. K. Jha, and N. Verma, "Algorithm-driven architectural design space exploration of domain-specific medical-sensor processors", *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 21, no. 10, pp. 1849-1862, 2013

[10] D.Azariadi, "Software Design and Optimization of ECG Signal Analysis and Diagnosis for Embedded IoT Devices"

[11] K.Koliogeorgi, "Optimizing ECG Signal Analysis by building FPGA-based accelerators using High Level Synthesis", *Diploma Thesis*

[12] Tsoutsouras, Vasileios, et al. "An Exploration Framework for Efficient High-Level Synthesis of Support Vector Machines: Case Study on ECG Arrhythmia Detection for Xilinx Zynq SoC." *Journal of Signal Processing Systems (2017): 1-21*

[13] A. Gacek and W. Pedrycz, "ECG signal processing, classification and interpreta-

tion: a comprehensive framework of computational intelligence", *Springer Science & Business Media, 2011*

[14] G. B. Moody and R. G. Mark, "The impact of the mit-bih arrhythmia database", *Engineering in Medicine and Biology Magazine, IEEE, vol. 20, no. 3, pp. 45-50, 2001*

[15] J. Pan and W. J. Tompkins, "A real-time qrs detection algorithm", *Biomedical Engineering, IEEE Transactions on, no. 3, pp. 230-236, 1985*

[16] E. D. Ubeyli,"Ecg beats classification using multiclass support vector machines with error correcting output codes",*Digital Signal Processing, vol. 17, no. 3, pp. 675-684, 2007*

[17] G. Meyfroidt, F. Guiza, J. Ramon, and M. Bruynooghe, "Machine learning techniques to examine large patient databases",*Best Practice & Research Clinical Anaesthesiology, vol. 23, no. 1, pp. 127-143, 2009*

[18] D. T.-W. Hau and E. W. Coiera, "Learning qualitative models from physiological signals", *Master's thesis, Massachusetts Institute of Technology, Dept. of Electrical Engineering and Computer Science, 1994*

[19] B. Gyselinckx, R. J. Vullers, C. Van Hoof, J. Ryckaert, R. F. Yazicioglu, P. Fiorini, and V. Leonov, "Human++: Emerging technology for body area networks.", in *VLSI-SoC, pp. 175-180, 2006*

[20] Z. Nie, L. Wang, W. Chen, T. Zhang, and Y. Zhang, "A low power biomedical signal processor asic based on hardware software codesign", in *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE, pp. 2559-2562, IEEE, 2009*

[21] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines", *ACM Transactions on Intelligent Systems and Technology, vol. 2, pp. 27:1-27:27, 2011. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm*

[22] C.-W. Hsu, C.-C. Chang, C.-J. Lin, et al., "A practical guide to support vector classification", 2003