University of Science and Technology at Zewail City
Department of Nanotechnology Engineering

Detecting Arrhythmia from ECG Using Hardware Accelerated Feature Extraction Algorithm and Producing Artificial Data Set by Comsol to Enhance SVM Training

By:

Ahmed Elsayed Mokhles
Essam Radwan Helmy
Islam Alaa Elsadek
Mohamed Abdallah Abdelgawad
Mohamed Saleh Khalafallah
Mostafa Ibrahim Elagouz

Supervised by:
Dr. Hassan Mostafa Hassan
Dr. Hamdy Abdelhameed

Abstract

According to World Health Organization (WHO), cardio vascular diseases (CVD) are the major reason responsible for human death. In 2018, CVDs were responsible for 31% of total deaths all over the world annually [11]. Electrocardiogram (ECG) is a representation that measures cardiac conduction system. ECG measures any abnormality in human heart according to changes in depolarization and repolarization of heart. Arrhythmia is one common disease of CVDs which is disorder in heartbeat and cardiac conduction system. Arrhythmia needs to be identified in its early stages because this prevents the chance of development of other heart diseases.

MIT BIH Arrithmya database was used through the project for training and testing. A supervised machine learning model was used for classification which is Support Vector Machine (SVM) which is a robust classification Algorithm. The feature extraction algorithms used are Autoregressive model, and Shanon Entropy for Maximal Overlab Discrete Wavelet Packet Transform of the data. A low power Hardware implementation of these complex feature extraction algorithms was done in order to do hardware acceleration that would be useful for wearable technologies and IoT applications in the near future. A real-time FPGA implementation was done on Zynq Ultrascale+ FPGA with dynamic power less than 1mW under operating frequency of 10 KHz.

Moreover, this work extends to attempting generating the first artificial ECG database through solving the FitzHugh–Nagumo (FHN) model for cardiac tissues on COMSOL Multiphysics. This artificial database is aimed to have controllable biasing and infinitely long records. The generated records are correlated to the database records (correlation coefficients > 0.7) and then they are subjected to the same training and testing methodology. The generated records' accuracies are higher but comparable to accuracies calculated for database records. Additionally, the proposed arrhythmia detection system examines the left bundle branch block condition which was not possible by the existing database.

Acknowledgment

In the name of Allah, the most be beneficent, the most merciful
Firstly, we would like to express our gratitude to our supervisors Dr. Hassan Mostafa and Dr. Hamdy Abdelhamid for their guidance, advice and sharing innovative ideas throughout this year. If this work is a success, it would have never been possible without their aid and support.
Next, we would like to express our sincere gratitude to Dr. Amr Zaher, adjunct assistant professor at the University of Science and Technology at Zewail City and a staff surgeon in the Egyptian national heart center. Dr. Amr was our first and main medical reference, his support, and efforts have undeniably contributed to the work presented in this thesis and chapter 5 in particular.
Our sincere thanks also go to every professor who has taught us at Zewail city and the whole team of the nanotechnology department in particular. Without their aid, we would have never reached that far.
We thank all our colleagues at Zewail city for sharing ideas and cooperating with us to help any problem we have faced. Moreover, we have shared a lot of funny moments during courses' projects and hanging out that cheered us up and motivated us to work more on the project.
Last but not least, we are thanking our beloved families who supported us spiritually to the most and exerted all their efforts to educate us properly.

Eventually, we all recognize the impact of Dr. Ahmed Zewail and how he had enhanced our lives, we all pray that Allah blesses his soul and preserves Zewail city.

Contents

List of Figures

List of tables

Chapter1: Introduction, Medical Background

The main objective of this project is to build a complete hardware system that is capable of detecting arrhythmia from a single lead, real-time electrocardiogram (ECG) signal. In this chapter, the medical concepts are introduced which includes human heart cycle, definition and types of arrhythmias, and electrocardiogram interpretation. Additionally, the ECG database used in training and testing is introduced, the details of training and testing are explained in the next chapters.

Electrophysiology of the Heart:

The human heart anatomy is shown in Figure 1- 1, it is composed of four blood chambers: the left atrium, right atrium, left ventricle, and right ventricle, in addition to electrical nodes: sinoatrial node (SAN) and atrioventricular node (AVN). The atrium, upper chambers, is connected to the ventricles, lower chambers, with a special type of electrically insulating tissues; hence, electrical pulses cannot be transferred between chambers directly. The right atrium receives unoxygenated blood from the body through the coronary sinus, superior vena cava, and vena cava. Once the right atrium is filled by unoxygenated blood, right atrium pumps this blood into the right ventricle. Then, this unoxygenated blood is pumped to the lungs through the pulmonary arteries to get oxygenated. The oxygenated blood enters the heart back from the left atrium which pumps it into the left ventricle. Eventually, this oxygenated blood is pumped back to the whole body after the contraction of the left ventricle. The heart with a normal rhythm shall undergo atriums relaxation and contraction simultaneously, and ventricles relaxation and contraction simultaneously [2].

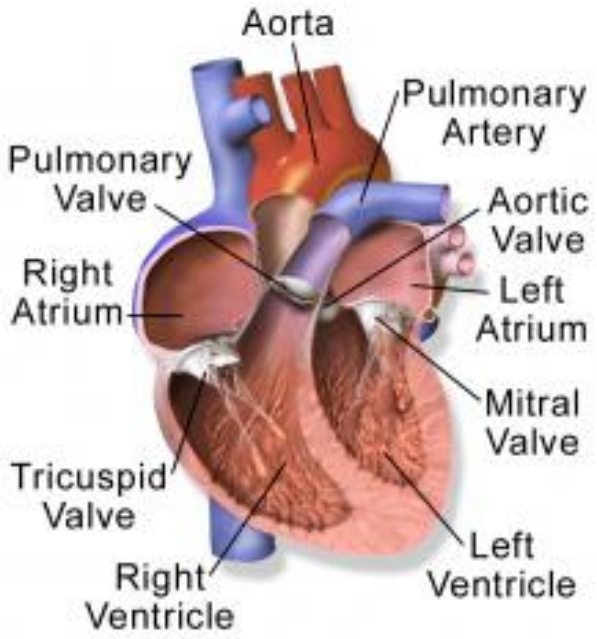


Figure 1- 1 Human heart anatomy [1].

The electrical conduction system of the heart regulates the blood cycle in the blood. There are two important terms that are used in describing the heart's conduction system: depolarization when the region's potential becomes more negative compared to outside of the heart, and repolarization when the potential is neutralized due to the balance between depolarization inward ionic currents and outward flux. The cardiac conduction system defines the ECG signal recorded from different positions on the human body. The conduction cycle starts through the activation of the SAN, which generates an electrical pulse that depolarizes the atria (contraction). This electric pulse is delayed at the AVN and then it propagates through the bundle of His to both right and left bundle branches, these bundle branches are terminated with Purkinje fibers that excite the muscle cells of the ventricles causes their contraction [3]. The conduction system of the heart is illustrated in Figure 1- 2. These electric pulses flow through the body tissues which results in the voltage drop that appears in the ECG.

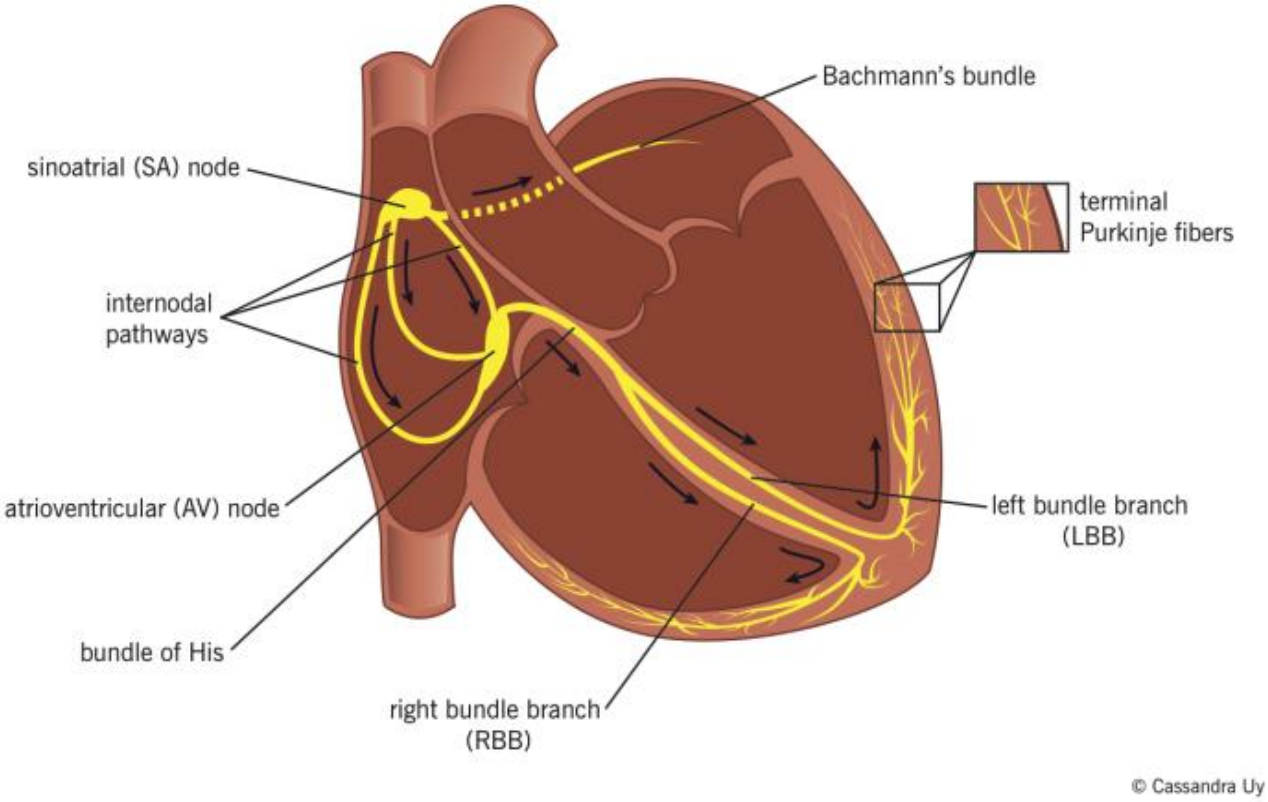


Figure 1- 2 Electrical Conduction system of the heart and electrical pulses pathways [3].

Electrocardiogram (ECG):

Electrocardiogram (ECG) is a visualization for the cardiac electrical activity that can be used in diagnosis for a variety of diseases [4]. It is a noninvasive way of monitoring the activity of the human heart. Any abnormality or disorder in the activity sequence of the heart affects the morphology of the ECG and can be identified based on the changes of ECG features. The ECG measures the depolarization and repolarization signals, it also helps in identifying [5]:

- Effect of different drugs on cardiac activity.
- Disorders in the cardiac system (Arrhythmias).
- Abnormality in electrical conduction pathways inside the heart.

The normal rhythm ECG is composed of a sequence of peaks as indicated in Figure 1- 3. The time domain features of the ECG are the P-QRS-T waves' amplitudes and intervals between these waves reveal necessary information on the heart's

condition. Fig 1.4 illustrates the relationship between the cardiac system cycle and the ECG features.

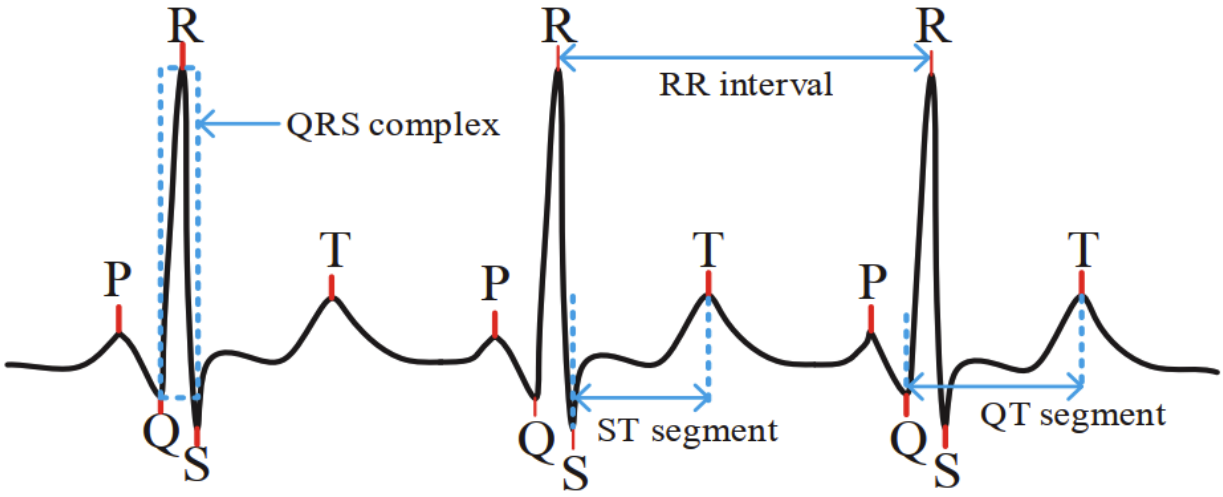


Figure 1- 3 Schematic of ECG signal and its features [6].

The P wave corresponds to atrial contraction, its amplitude is smaller than that of the R peak, its existence reveals that the SAN is activated and that it is a normal rhythm, and its absence is a symptom of atrial fibrillation, or premature atrial contraction (PAC) [3]. QRS complex corresponds to ventricular contraction, its duration and amplitudes reveal information about the depolarization of the ventricles and electrical pulses conduction through bundle branches and Purkinje fibers, and its prolongation is always a symptom of bundle branch block [7]. The repolarization signal of the atria is masked by the ventricular depolarization signal. The T wave corresponds to the ventricular repolarization and its disturbance is a symptom of bundle branch block [7] and ischemia [8]. Any disorder in the heart conduction system, arrhythmias, corresponds to abnormalities in these waves' existence amplitudes or durations. This is the basis of heart diseases' detection systems based on the ECG.

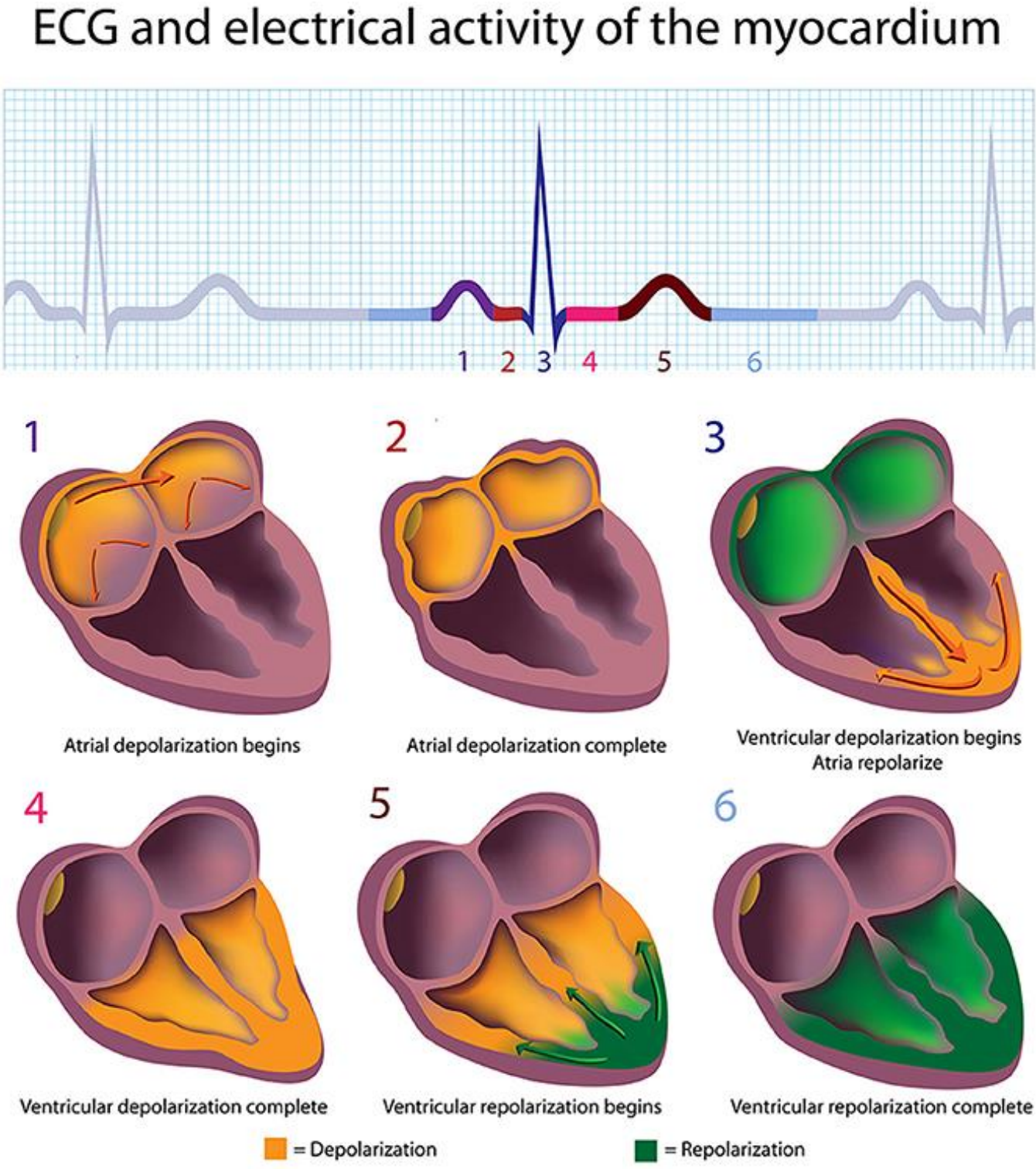


Figure 1- 4 Cardiac system cycle and its correspondence on the ECG signal. Figure adapted from Shutterstock.com by Alila Sao Mai.

The 12-Lead Standard Electrocardiogram (ECG):

The ECG signal differs according to the position of the electrodes relative to the heart, there are different configurations; however, the most commonly used configuration is the 12-lead standard ECG configuration. The12-lead standard ECG configuration is composed of 3 Einthoven leads (bipolar), 3-limb leads (unipolar), and 6 chest leads. The 3 Einthoven leads (I, II, III) signal is a difference between two different leads, the 3-limb leads (AVR, AVL, AVF) and the 6 chest leads (V1, V2, V3, V4, V5, V6) record the potential from a single point only. Each of these leads has a different view of the heart according to their placement as shown in Figure 1- 5. These leads' signals are derived from the placement of 10 electrodes on human heart noninvasively as shown in Figure 1- 6.

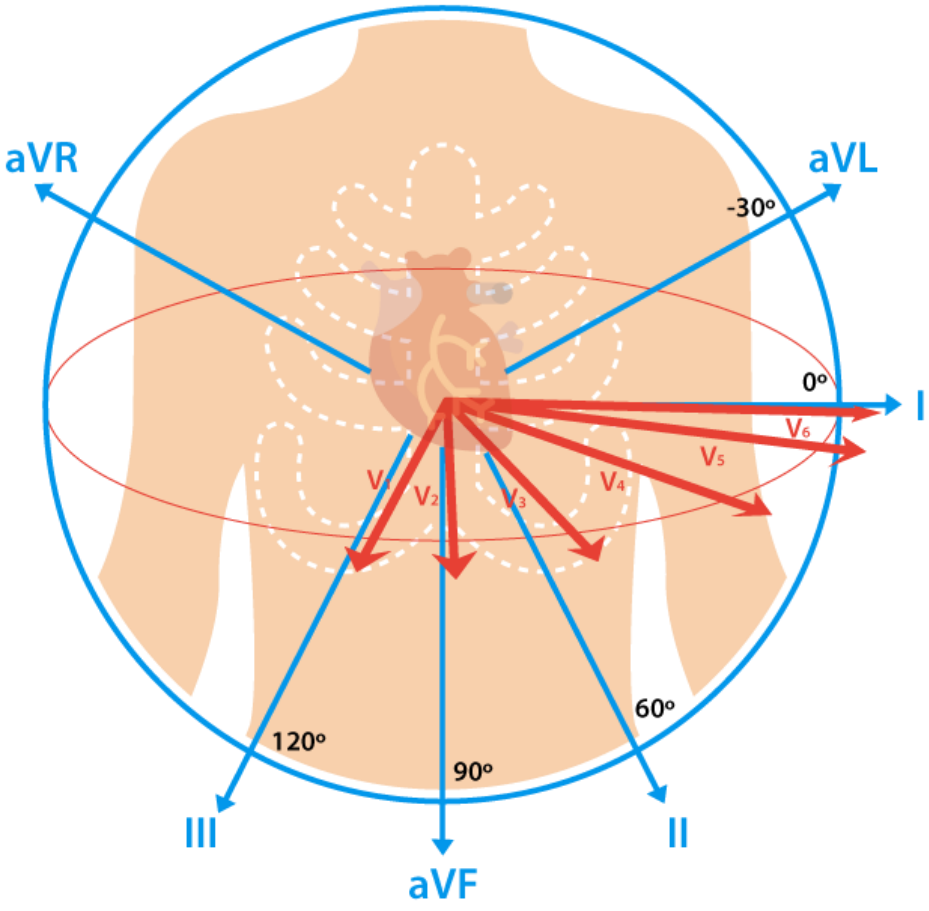


Figure 1- 5 12-lead ECG angles and their different views for the heart.

In our work, we have focused only on lead II, the potential between the right arm (negative electrode) and left leg (positive electrode), due to its availability in the records of the database. Additionally, lead II has the best heart view that yields high upright QRS complex and high amplitude P wave compared to the other leads of the 12-lead standard system. Thus, lead II is the first and most commonly used lead for diagnosis of different heart diseases [10].

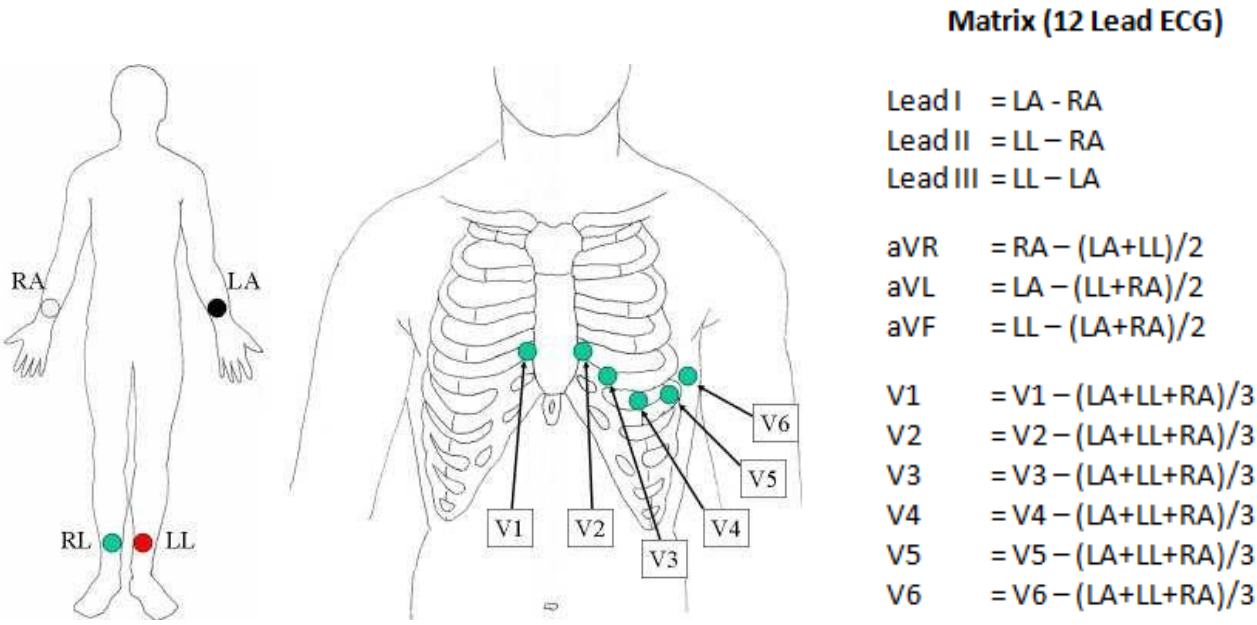


Figure 1- 6 Electrodes placement in 12-lead ECG configuration and the transformation used for extracting the 12-lead signals [9].

Arrhythmias:

In 2018, the World Health Organization (WHO) has reported that cardiovascular diseases (CVDs) are the major (number 1) reason for human death globally and contributes to 31% of total deaths annually [11]. One of the most common CVDs is cardiac arrhythmia which is a disorder in the heartbeat rate or cardiac conduction system. The focus on arrhythmia comes from the fact that early detection of arrhythmia prevents the development of more serious conditions.

Moreover, some types of arrhythmias can cause sudden death as severe ventricular fibrillation (VF) [3].

There are many types of arrhythmias, each of them is different in its symptoms, causes, severity, and ECG morphology. These types can be categorized in general into four categories: abnormality in heartbeat rate (either fast or slow), irregular heartbeat with abnormal ECG morphology, blockage of the cardiac conduction system (either right or left bundle branch), and early beats not triggered by the SAN [3]. Each of these categories includes different diseases according to the region of occurrence and condition. In the following context, the main types of cardiac arrhythmias are introduced briefly [3].

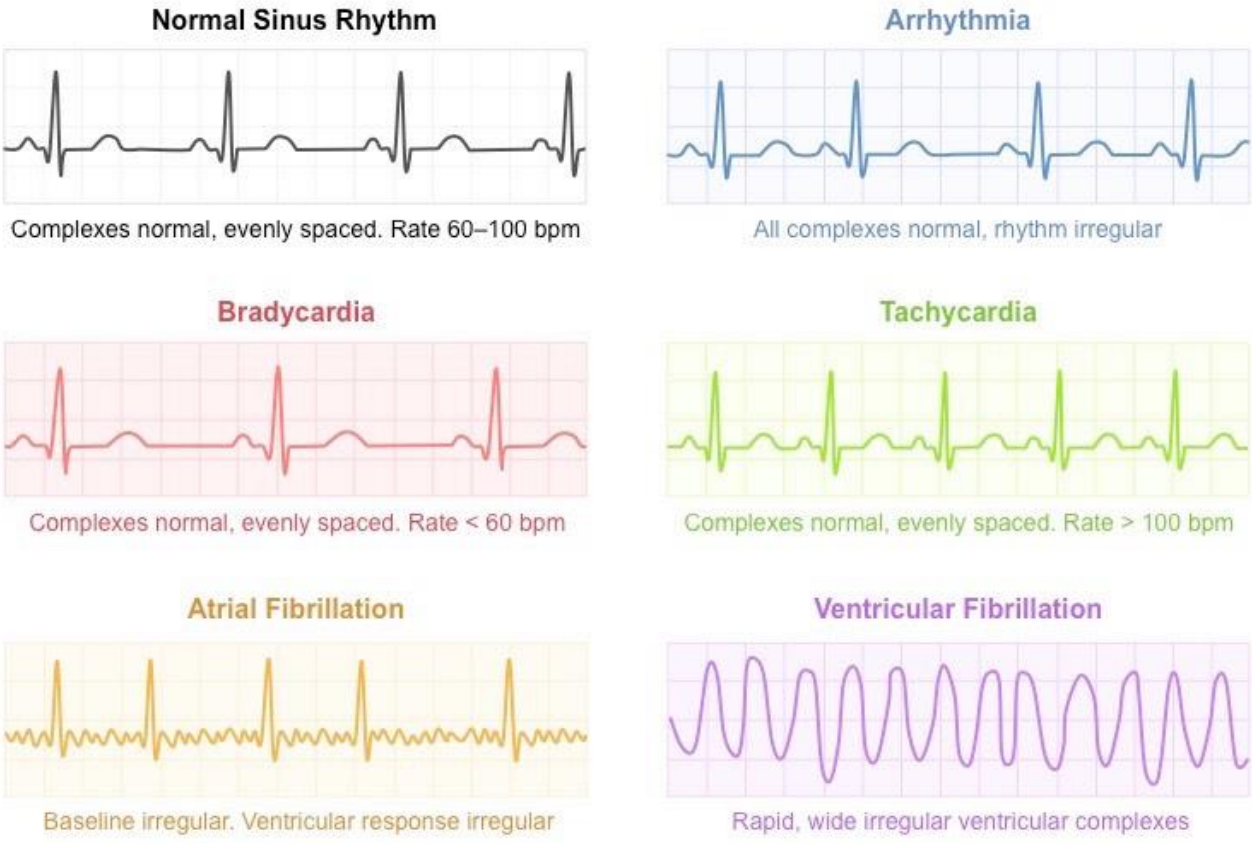


Figure 1- 7 ECG morphology for different cardiac arrhythmia types [13].

- **Bradycardia** is the condition of having slow heartbeats, it delays the pumping of the oxygenated blood to the body and might affect vital organs in the body. Nevertheless, resting for too long or sleeping slowers the heartbeat; hence, the diagnosis of bradycardia involves monitoring the patient's heartbeats for a long duration.
- **Tachycardia** is the condition of having fast heartbeats, these fast beats hinder the complete filling of the ventricles before pumping the oxygenated blood to the body; hence, some organs might suffer from oxygenated blood deficiency.
- **Atrial flutter** is the condition of having a disorder in the SAN that causes the atria to depolarize very fast compared to the ventricles which lead to an ECG single composed of successive P waves (flutter waves).
- **Premature atrial contractions (PAC),** it is a condition that occurs when a group of cells in the atria gather together and become electrically active

  (ectopic node) that it triggers the depolarization of the atria prior to the SAN. It changes the morphology of the P wave in the ECG in a way that differs according to the position of this ectopic node.
- **Atrial fibrillation (AF)** is the development of untreated atrial flutter condition, it is a disorder in the SAN that leads to dysfunctionality and lacking coordination between different regions in the atria. It results in abnormal pumping of blood from atria to the ventricles which leads to abnormal pumping of oxygenated blood to all body organs.
- **Premature ventricular contractions (PVC)** is a condition similar to that of PAC; however, the ectopic node is formed inside one of the ventricles. The ectopic node depolarizes the ventricles before the atria and results in an abnormal morphology of the QRS complex.

- **Ventricular fibrillation (VF)** is a developed condition of PVC where several uncoordinated ectopic nodes are created inside the ventricles; hence, different parts of the ventricles are depolarized and contracts at different times. This uncoordinated and abnormal rhythm might cause sudden death [12].
- **Bundle branch block (BBB)** is a disorder in the cardiac conduction system when one of the bundles or its Purkinje fibers is blocked and does not conduct the electrical signal from the AVN to the ventricles; hence, it delays the activation of the ventricle on the side of the blockage as it gets activated by conduction of the electrical signal through myocardium walls of the other ventricle [7].

MIT-BIH Arrhythmia Database:

Despite the existence of many types of cardiac arrhythmias, our analysis is limited to conditions available in the dataset. MIT-BIH Arrhythmia dataset [14] available on PhysioNet platform [15] has been used for training and testing, it is the most commonly used dataset in literature [16]. MIT-BIH Arrhythmia dataset includes 48 half-hour records of two-channel ECG leads, only 46 records have the ECG signal of the modified limb lead II (MLII). MLII is similar to normal lead II; however, the electrodes on the arms and legs are positioned nearer to the torso to amplify the ECG signal. The dataset records are prepared by the Boston's Beth Israel Hospital (BIH) arrhythmia laboratory, the first 23 records (100 series) are chosen randomly from their recordings for the arrhythmia patients. The rest 25 records (200 series) are picked and chosen from the records to include important, but uncommon arrhythmia conditions in the dataset [17]. The dataset provides annotation for each beat to identify its type based on its morphology. However, this dataset suffers from records biasing, as most of the records are extremely bias

either for normal or abnormal heartbeats. Additionally, it suffers from the noise and artifacts due to the electrodes used in measuring the ECG signal.

Detection of Arrhythmia based on the ECG is still challenging due to the fact that arrhythmia symptoms and their correspondence on the ECG morphology might vary from one person to another. The physical condition, age, mental status, stress and many other factors affect the morphology of the ECG; hence, detection of arrhythmia requires long monitoring of the ECG signal. Additionally, the available dataset is small and does not cover all cardiac arrhythmias conditions; hence, all designed systems and reported accuracies are only valid for conditions reported in the dataset.

References:

[1] "Medical gallery of Blausen Medical 2014", *WikiJournal of Medicine*, vol. 1, no. 2, 2014. Available: 10.15347/wjm/2014.010.

[2] "Handbook of Cardiac Anatomy, Physiology, and Devices", 2009. Available: 10.1007/978-1-60327-372-5.

[3] H. Griffiths, "Basic arrhythmias by Gail Walraven R. J. Brady Co., Maryland, USA, (1980) 512 pages, many figures, $19.45 ISBN 0-87619-627-X", *Clinical Cardiology*, vol. 5, no. 6, pp. 376-378, 1982. Available: 10.1002/clc.4960050606.

[4] R. Acharya, J. S. Suri, J. A.E. Spaan and S .M. Krishnan, *Advances in Cardiac Signal Processing*, springer, pp.1-50.

[5] A. J. Moss and S. Stern., "Noninvasive Electrocardiology," *Clinical Aspects of Holter*, London, Philadelphia, W.B. Saunders, 1996.

[6] S. Pudukotai Dinakarrao and A. Jantsch, "ADDHard", *Proceedings of the 2018 on Great Lakes Symposium on VLSI - GLSVLSI '18*, 2018. Available: 10.1145/3194554.3194647.

[7] K. Bilchick, S. Kamath, J. DiMarco and G. Stukenborg, "Bundle-Branch Block Morphology and Other Predictors of Outcome After Cardiac Resynchronization Therapy in Medicare Patients", *Circulation*, vol. 122, no. 20, pp. 2022-2030, 2010. Available: 10.1161/circulationaha.110.956011.

[8] T. Chow et al., "Microvolt T-Wave Alternans Identifies Patients With Ischemic Cardiomyopathy Who Benefit From Implantable Cardioverter-Defibrillator Therapy", *Journal of the American College of Cardiology*, vol. 49, no. 1, pp. 50-58, 2007. Available: 10.1016/j.jacc.2006.06.079

[9] M. Lemay and J. Vesin, "Data processing techniques for the characterization of atrial fibrillation," EPFL Thesis, DOI: 10.5075/epfl-thesis-3982

[10] "12-lead ECGs, part I: Recognizing normal findings", *Nursingcenter.com*, 2019. [Online]. Available: https://www.nursingcenter.com.

[11] "World Health Organization Cardiovascular Disease Facts ". World Health Organization (2019). [Online]. Available: https://www.who.int/cardiovascular_diseases/en/.

[12] L. Dekker et al., "Familial Sudden Death Is an Important Risk Factor for Primary Ventricular Fibrillation", *Circulation*, vol. 114, no. 11, pp. 1140-1145, 2006. Available: 10.1161/circulationaha.105.606145.

[13] "Electrocardiography", BioNinja. [Online]. Available: https://ib.bioninja.com.au/

[14] G. Moody and R. Mark, "The impact of the MIT-BIH Arrhythmia Database", IEEE Engineering in Medicine and Biology Magazine, vol. 20, no. 3, pp. 45-50, 2001. Available: 10.1109/51.932724.

[15] G. Moody, R. Mark and A. Goldberger, "PhysioNet: a Web-based resource for the study of physiologic signals", IEEE Engineering in Medicine and Biology Magazine, vol. 20, no. 3, pp. 70-75, 2001. Available: 10.1109/51.932728.

[16] N. Flores, R. L. Avitia, M. A. Reyna, and C. García, “Readily available ECG databases,” *Journal of Electrocardiology*, vol. 51, no. 6, pp. 1095–1097, 2018.

[17] M. Llamedo and J. P. Martínez, "QRS detectors performance comparison in public databases," Computing in Cardiology 2014, Cambridge, MA, 2014, pp. 357-360

Chapter 2: Machine Learning & Features Theory

Section 1: Machine Learning:

Machine Learning (ML) is the field of computer science that uses Artificial Intelligence (AI) to provide systems with the ability to adaptively identify data and improve its identification ability without the need of specific programming for the problem. There are different types of ML, however, the two main types are supervised and unsupervised learning. In supervised learning the ML model is to be trained with labeled data in order to be able to classify future data according to the knowledge it gained from the labeled data. On the other hand, unsupervised learning deals with unlabeled data and it is up to the ML model to identify hidden patterns, cluster the data points into groups, or lower the dimensionality. Usually unsupervised learning algorithms are more complex than supervised learning algorithms.

Supervised learning models are usually used for classifying new data or for regression (predicting future data). Since our problem is a classification problem between normal and arrhythmic subjects, we need a classification model and the problem is called a binary classification problem. The most commonly used algorithm in our problem is Support Vector Machine (SVM) algorithm due to its robustness and ability to deal with large datasets efficiently.

Support Vector Machine:

Support vector machine is a supervised learning algorithm that is built according to Statistical Learning Theory (SLT) and it is introduced by Vladimir Vapnik [1]. SVM is usually used for binary classification problem, as it is in our case, and it uses the old labeled data to build a model that would be able to categorize new test points to one of the two categories. A great advantage of SVM is that it has a very good generalization ability, that is, if it is trained on a certain

data sample the resulting hyperplane will most probably be able to classify any test point with the same features. This results from the dependence of SVM on Support Vectors (SV) that are the closest points of the two classes to the hyperplane. These support vectors contribute greatly to the optimization of the classification parameters and play an effective role in the classification operation. As a result, after some training, data points other than SVs become redundant and barely change the fitting parameters. So, we can note that the generalization properties of SVM are good.

SVM Idea:

In a binary classification problem, the training data consists of data points x_i each of width m corresponding to the dimension of the feature space (number of features). Each data point x_i has a corresponding data label y_i that is +1 or -1 according to the class to which this data point belongs. If the data is linearly separable, then there is an infinite number of hyperplanes that can separate the two classes as shown in **Error! Reference source not found.**. The target of SVM algorithm is to choose the hyperplane that maximizes the margin that is the distance between the hyperplane and the closest points of the two classes known as support vectors.

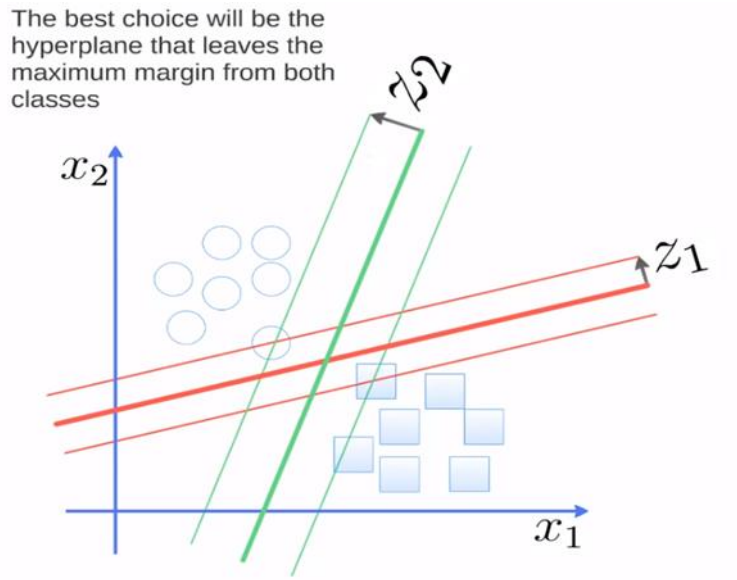


Figure 2- 1 Linearly separable example with different possible hyperplanes.

The hyperplane equation consists of a vector $\vec{w}$ that is normal to the hyperplane and a constant b that moves the hyperplane over this direction. When we have a test point x, it can be classified according to the equation: [2]

$$f(x) = sign(\vec{w}.\vec{x} + b) \qquad (2\text{-}1)$$

So, for positive samples$f(x) \geq +1$, and for negative samples$f(x) \leq -1$. Thus, it can be written that$y_i(\vec{w}.\overrightarrow{x_i} + b) = 1$, where y_i is the data label of test point x_i. Training points that results in exactly ± 1 are called support vectors and the planes passing through these SVs parallel to the hyperplane are called canonical hyperplanes as shown in **Error! Reference source not found. Error! Reference source not found.**[33] The distance between the two canonical hyperplanes is called the margin needs to be maximized and can be calculated as follows:

$$(\vec{w}.\overrightarrow{x_+} + b) - (\vec{w}.\overrightarrow{x_-} + b) = 2$$

$$\vec{w}.(\overrightarrow{x_+} - \overrightarrow{x_-}) = 2 \qquad (2\text{-}2)$$

Normalizing to the unit vector of $||\vec{w}||$, it can be shown that $(\overrightarrow{x_+} - \overrightarrow{x_-}) = \frac{2}{||\vec{w}||}$ and we need to maximized this value or equivalently minimize $\frac{||\vec{w}||}{2}$ subject to the condition that $y_i(\vec{w}.\overrightarrow{x_i} + b) = 1$.

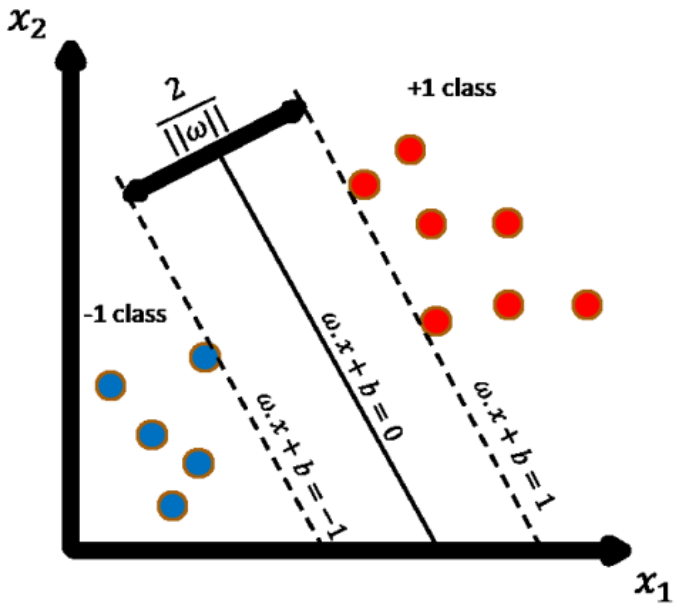


Figure 2- 2: Hyperplane and Canonical hyperplanes and their equations

This is called a constraint optimization problem that can be reduced to minimization of LaGrange function defined as:

$$L(w,b) = \frac{1}{2}\left|\left|\vec{w}\right|\right|^2 - \sum_{i=1}^{N} \alpha_i [y_i(\vec{w_i}.\vec{x_i} + b) - 1] \qquad (2\text{-}3)$$

This new equation consists of the target function to be minimized minus the constraint function multiplied by the Lagrange multipliers (α_i). To minimize this equation, it should be differentiated with respect to b and $\vec{w}$ and to be set equal to zero.

$$\frac{\partial L(w,b)}{\partial b} = -\sum_{i=1}^{N} \alpha_i y_i = 0 \qquad (2\text{-}4)$$

$$\frac{\partial L(w,b)}{\partial \vec{w}} = \vec{w} - \sum_{i=1}^{N} \alpha_i y_i \vec{x_i} = 0 \qquad (2\text{-}5)$$

Solving for the Lagrange multipliers, $\vec{w}$ *and* b can be calculated through:

$$\vec{w} = \sum_{i=1}^{N} \alpha_i y_i \vec{x_i} \qquad (2\text{-}6)$$

$$b = y_i - \vec{w}.\vec{x_i} \qquad (2\text{-}7)$$

Kernel Function:

The previous analysis assumed that the data points are linearly separable, however, if the data is not separable the trick of kernel function can be used. Kernel function is a transformation that is applied over the data set in order to

make it separable as shown in **Error! Reference source not found.**. The notation of the kernel function is $k(x_i, x_j) = \Phi(x_i).\Phi(x_j)$ where Φ is the transformation function. There are different kinds of kernel functions namely:

Linear Kernel Function: $k(x_i, x_j) = x_i.x_j$

Polynomial Kernel function: $k(x_i, x_j) = (x_i.x_j + y)^n$, n = order

Laplacian Kernel: $k(x_i, x_j) = e^{-\psi||x_i - x_j||}$

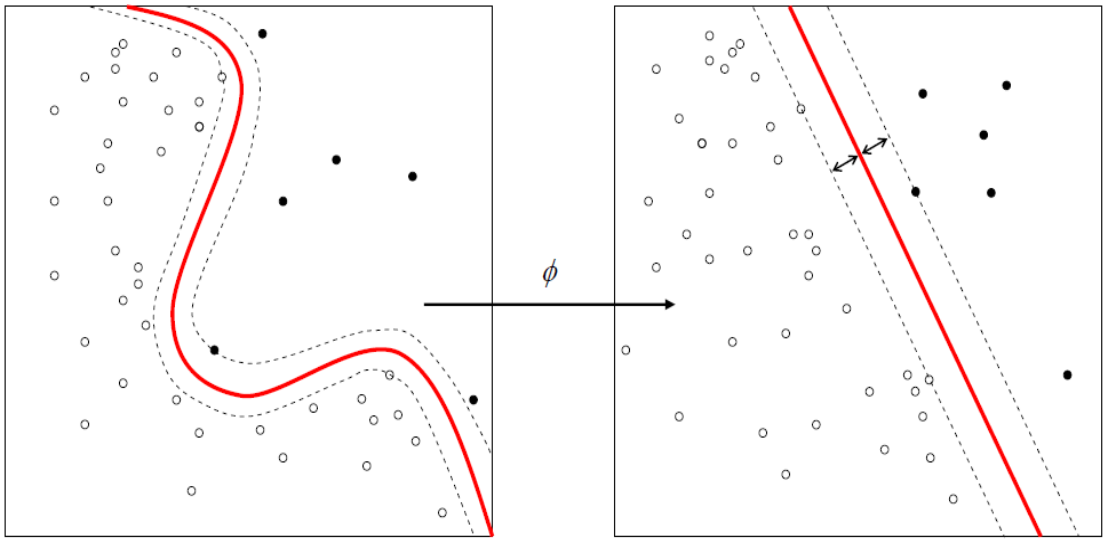


Figure 2- 3: Non-Linearly separable data before and after the application of kernel function

Radial Basis Kernel: $k(x_i, x_j) = e^{\frac{||x_i - x_j||}{\sigma}}$ (2-8)

Appling the kernel function will help making the data separable in addition to changing the minimization function to the following form:

$$\sum_{i=1}^{N} \alpha_i - 0.5 \sum_{i=1}^{N} \sum_{i=1}^{N} y_j y_i k(\overrightarrow{x_j}.\overrightarrow{x_i}) \alpha_i \alpha_j \quad (2\text{-}9)$$

Doing the math, it can be shown that the classification function of the SVM is:

$$f(x) = \sum_{i=1}^{N} y_i k(\overrightarrow{x_i}.\vec{x}) \alpha_i + b \quad (2\text{-}10)$$

Where x is the new test point after training.

Sequential Minimum Optimization (SMO) SVM:

The training process of SVM takes a huge amount of time specifically when the data is large and are in very high dimension. In this case, the process of optimization and solving for Lagrange multipliers is a very large Quadratic

Programming (QP) problem and requires unreasonable amount of memory. Hence, SMO was proposed by Platt [4] in 1998 as an optimized training algorithm for SVM. SMO SVM training algorithm is shown briefly in **Error! Reference source not found.**.

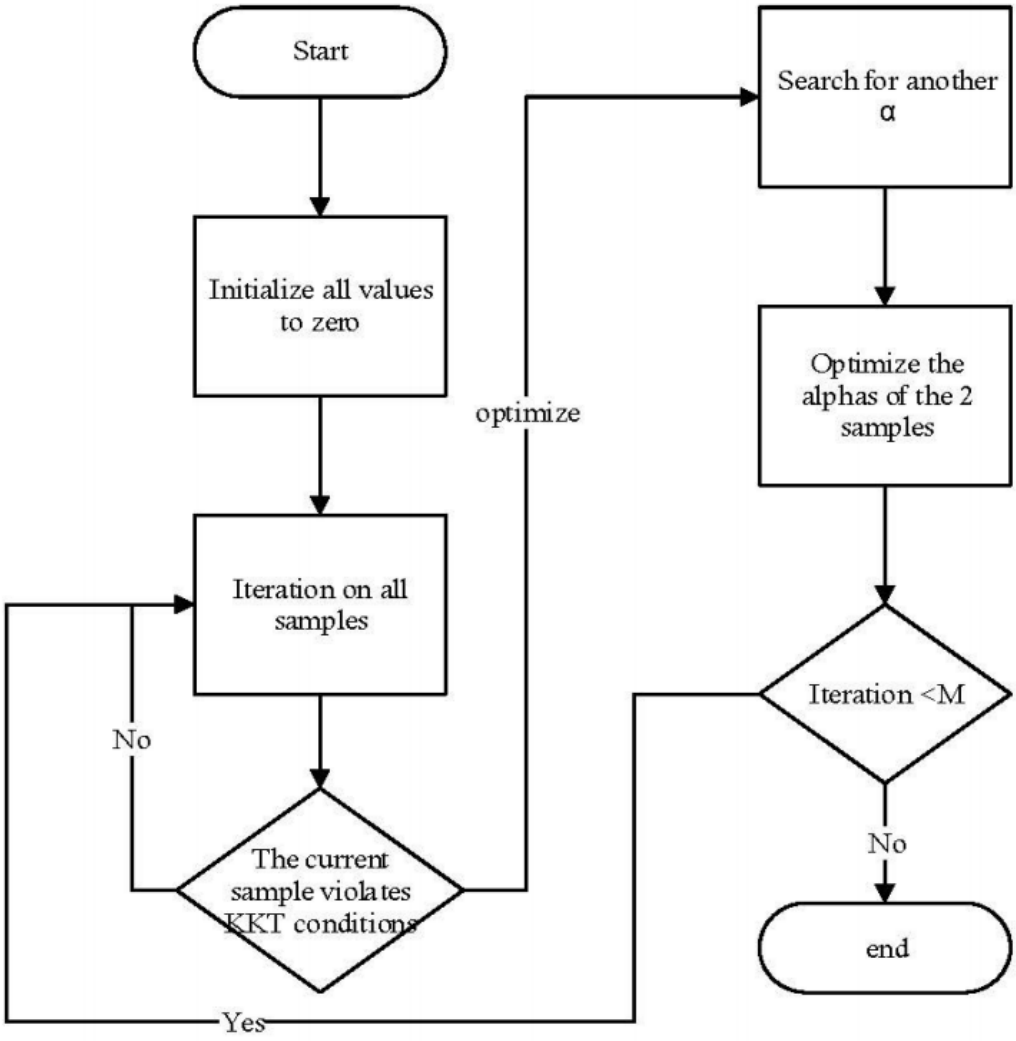


Figure 2- 4: SMO algorithm flowchart

Section 2: Feature Extraction Concepts:

Sub-section1: AR Theory and Mathematics

Autoregressive model introduction:

Autoregressive (AR) analysis is an effective algorithm for the estimation of the complex signals. Moreover, these signals can be modeled on the base of their AR analysis. Thus, AR analysis is widely used in the biomedical applications, medical diagnostics, economics and voice recognition. [5] In our thesis, it is used to extract features of ECG signals to detect certain diseases.

Our dataset is considered a time series. A time series is a sequence of measurements through time of a certain variable. Thus, in our case its voltage measured over time with a time step 1/360 second. An autoregressive model (AR) is applied on time series to produce a value which is regressed from previous value. So, it predicts a new value by using a linear combination of previous values as indicated in equation 2-11.

$$y_t = \beta_0 + \beta_1 y_{t-1} + \epsilon_t \qquad (2\text{-}11)$$

It is used as subscript instead of i to emphasize that values are measured over time. In this equation, y_{t-1} has become a predictor for new y_t . ϵ_t represents an error term which has same error assumptions in a linear regression model. AR model has order which represents how many previous values in time are used to predict the present value. So, the above equation (2-11) is a first order AR model which is written as AR(1). A second order will depend on two previous values as shown in equation 2-12. Generally, a k^{th} order autoregressive model is a multiple

linear regression that predicts the value of present variable by using previous values from a time series at times $t-1, t-2, \dots, t-k-1, t-k$ and this model is written as AR(k). [6]

$$y_t = \beta_0 + \beta_1 y_{t-1} + \beta_1 y_{t-2} + \epsilon_t \qquad (2\text{-}12)$$

Autocorrelation coefficient:

The previous values are multiplied by coefficients. These coefficients (β) are called autocorrelation coefficients. Autocorrelation is a useful guide to the properties of a time series, provide useful descriptive information and has unique importance in AR model. Thus, autocorrelation mathematics are shown in this subsection.

The concept of autocorrelation coefficient is based on ordinary correlation coefficient. It's assumed that ordinary correlation coefficient is known. Given N pairs of measurements of two distinct variables x and y, their sample correlation coefficient is shown in equation 2-13.

$$r = \frac{\sum(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum(x_i-\bar{x})^2 \sum (y_i-\bar{y})^2}} \qquad (2\text{-}13)$$

x_i , y_i : represents the i[th] element of the N elements.

$\bar{x}$, $\bar{y}$: represents the average of the N elements in each pair.

This sample correlation coefficient (r) measures the strength of the linear association between the two variables and its range lies between [-1,1]. Also, the coefficient has no units which fits with the previously mentioned equations (Eq. 2-11 & Eq. 2-12). This concept is applied on time series. However, in this case only one variable x is present. Given N observations in time series x1, x2, …, xN, (N-1) pairs of observations are produced; (x1,x2), (x2,x3), … , (xN-1, xN). Each pair of observations is separated by only one-time step. The first variable in each pair is considered as variable and the other is considered another variable. Then,

these two variables are substituted in equation 2-13 which gives the below equation:

$$a = \frac{\sum_{t=1}^{N-1}(x_t - \bar{x}_{(1)})(x_{t+1} - \bar{x}_{(2)})}{\sqrt{\sum_{t=1}^{N-1}(x_t - \bar{x}_{(1)})^2 \sum_{t=1}^{N-1}(x_{t+1} - \bar{x}_{(2)})^2}} \qquad (2\text{-}14)$$

Where

$$\bar{x}_{(1)} = \sum_{t=1}^{N-1} \frac{x_t}{N-1} \qquad (2\text{-}15)$$

$$\bar{x}_{(2)} = \sum_{t=2}^{N} \frac{x_t}{N-1} \qquad (2\text{-}16)$$

As the coefficient in equation 2-14 measures the correlation between successive observations, it is called an autocorrelation coefficient or a serial correlation coefficient at lag one instead of sample correlation coefficient. The two means are very close in value to each other especially when having a large number of elements (N), thus it is approximated that $\bar{x}_{(1)} \approx \bar{x}_{(2)}$. This approximation results in a reduced equation:

$$a = \frac{\sum_{t=1}^{N-1}(x_t - \bar{x})(x_{t+1} - \bar{x})}{(N-1)\sum_{t=1}^{N}(x_t - \bar{x})^2/N} \qquad (2\text{-}17)$$

Where $\bar{x}$ is the sample mean and is calculated by the given equation:

$$\bar{x} = \sum_{t=1}^{N} \frac{x_t}{N} \qquad (2\text{-}18)$$

As assumed above, the N is large number (in our implementation N is equal to 64 or higher). Thus the $\frac{N-1}{N}$ can be neglected. The expression will be:

$$a = \frac{\sum_{t=1}^{N-1}(x_t - \bar{x})(x_{t+1} - \bar{x})}{\sum_{t=1}^{N}(x_t - \bar{x})^2} \qquad (2\text{-}19)$$

Equation 2-19 is the final used equation to calculate the autocorrelation coefficient in our software model and hardware implementation which are discussed in the following chapters. [7]

Burg algorithm for implementing AR model:

AR model can be implemented by various methods such as Burg, Covariance, modified covariance and Yule-Walker AR Estimators. Covariance and modified covariance AR estimator may produce unstable models. However, Burg and Yule-Walker AR Estimators always produce a stable model. [8]

Burg algorithm is preferred over Yule-Walker algorithm in hardware implementation as Burg has major advantages such as: [9]

- Modularity and less computational complexity as it uses lattice structure to perform recursive operations.
- Any order can be added for an existing AR order by adding more lattice stages without affecting earlier computations for lower orders.

Burg method is based on minimizing the sum of the squared forward and backward prediction errors:

$$\epsilon_k = \sum_{n=k+1}^{N} f_k^2(n) + b_k^2(n) \qquad (2\text{-}20)$$

Where f and b are the forward and backward prediction errors respectively. K is the order of the AR model. N is the length of the input data which the number of data elements.

forward and backward predictions can be calculated using a recursive formula:

$$f_k(n) = f_{k-1}(n) - a_k\, b_{k-1}(n-1) \qquad (2\text{-}21)$$

$$b_k(n) = b_{k-1}(n-1) - a_k\, f_{k-1}(n) \qquad (2\text{-}22)$$

The (a) parameter is the autocorrection coefficient calculated in the previous section. These formulas can be further explained by using Figure 2- 5below.

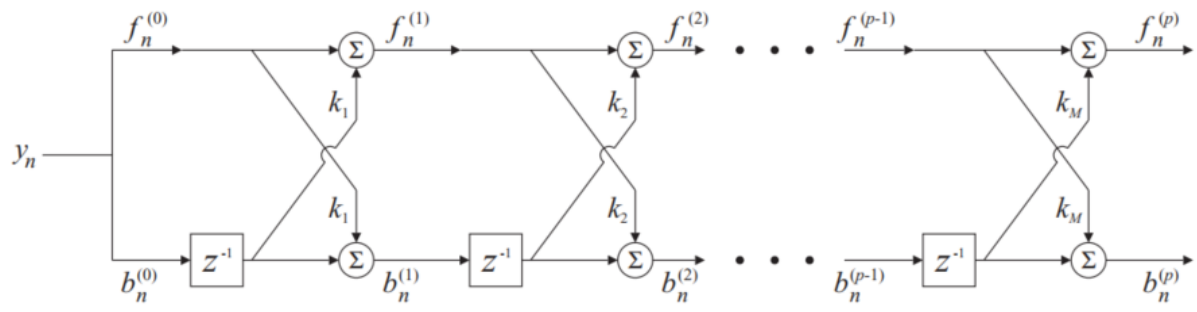


Figure 2- 5: Lattice structure for forward and backward prediction error [10]

Forward and backward predictions are initialized by input data from the time series where every array is less than the input data by one element (N-1) as specified in autocorrelation derivation. Then, these two predictions are used to generate autocorrelation coefficient (represented by k in the Figure 2- 5). This autocorrelation coefficient is used to produce next forward, backward and AR coefficient.

The AR coefficients can be obtained from the auto-reflection coefficient using Levinson-Durbin algorithm [10]:

$$AR_m^{(l)} = AR_m^{(l-1)} + a_l\, AR_{l-m}^{(l-1)} \qquad (2\text{-}23)$$

Where $m = 1,2,3,\dots,l-1$ and is repeated for $l = 1,2,\dots,k$

At the end of the k iteration for AR(k), $AR^{(k)}$ is produced.

In this thesis, the procedure of implementing AR model either Software or Hardware depends completely on the equation of autocorrelation coefficient, forward, backward prediction and AR coefficients equations: 2-19, 2-21, 2-22 and 2-23.

Wavelet Packet Entropy:

Entropy

The concept of entropy in information theory was first introduced by ClaudeE. Shannon in 1948[11], and now it is used in various applications of signal and image processing. The concept of Shannon entropy depends on the probability of a point or a character to appear in a stream of points or characters.

$$H = \sum_{i=1}^{M} -p_i \log(p_i) \qquad (2\text{-}24)$$

p_i is the probability of point I to occur in the stream of data.[11]

The Figure 2- 6 shows the values for H function for two possibilities with probabilities p and(1-p).

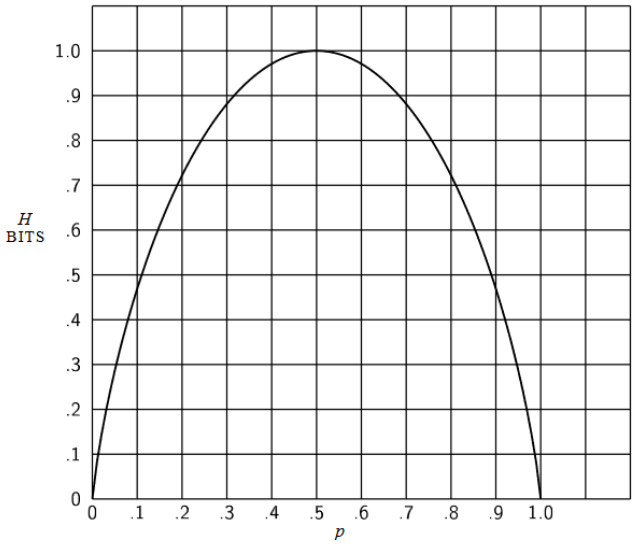


Figure 2- 6: Entropy for two possibilities with probabilities p and(1-p) [11]

Wavelet Packet Decomposition

When the testing signal is changed from the normal state to abnormal state or fault state, the amplitude, frequency or both are significantly changed. The change in Shannon entropy may not cope with abnormal signals, however, wavelet entropy can be used to solve this as it is based on time-frequency analysis, featured with multi- resolution analysis and can express the signal's localized feature at time-frequency domains[12]. So, the features calculated from entropy are from the ECG signals directly, however the signal is firstly processed by MODWT (Maximal overlap discrete wavelet packet transform).

The decomposing process splits the original signal into two subspaces, V and W, which are orthonormal and Complementary to each other [13], as shown schematically in Figure 2- 8.

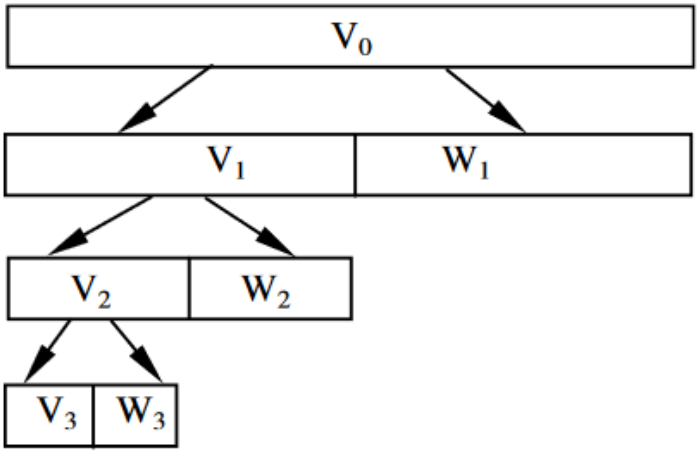


Figure 2- 7: Decomposing algorithm of DWT. [13]

The decomposed packet is calculated as follows by these equations:[13]

$$U_{j,k}^{n}(t) = 2^{-j/2}u^{n}\left(2^{-j}t - k\right) \text{ (2-25)}$$

$$U_{j,0}^{n}(t) = \sum_{k} h_0(k)u_{j-1,k}^{i} \quad (n \text{ is even}) \text{ (2-26)}$$

$$U_{j,0}^{n}(t) = \sum_{k} h_1(k)u_{j-1,k}^{i} \quad (n \text{ is odd}) \text{ (2-27)}$$

Where k is a shift factor related to the current branch of the decomposing to be calculated, and $h_0(k)$, $h_1(k)$ is a couple of quadruple mirror filters (QMF). This decomposition results in the following chart [13]:

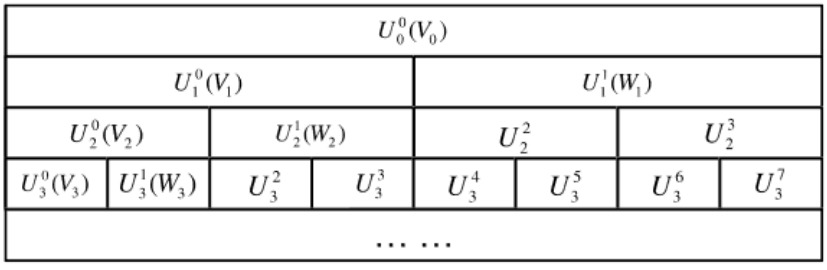


Figure 2- 8: Decomposing result of DWT [13]

For classification, MODWT is usually used for extracting features. This is due to its great time and frequency localization ability; it can expose the local features of the input ECG signal. Moreover, the multi-level decomposition of an ECG

signal into provides multi-scale features with each one to represent a specific characteristic of the ECG signal.[14]

The following is an example of a level 3 DWT for an ECG signal performed by T. Li and M. Zhou with each row is a depth in the decomposition:

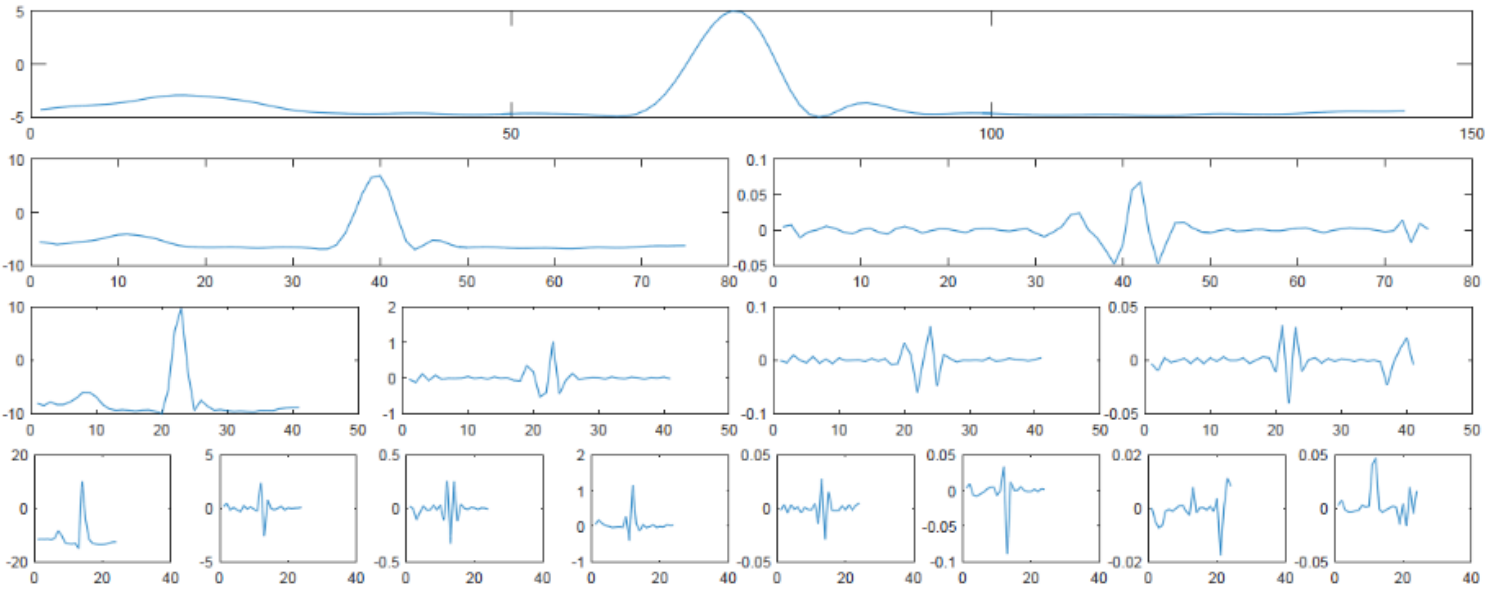


Figure 2- 9: level 3 DWT for an ECG signal [14]

FFT and Inverse FFT

Discrete Fourier transform (DFT) is a member of Fourier analysis family that converts finite digitized sequence of points in time domain to finite digitized sequence of points in frequency domain [15]. DFT is used in a wide range of applications in digital signal processing. To extract features from ECG signal, features must be separable enough. In time domain, features are not highly separable, and accuracy decreases drastically. That's why DFT is used widely in medical applications [16].

DFT for a sequence X_n is in the form:

$$X_k = \sum_{n=0}^{N-1} X_n \, e^{\frac{-2\pi i}{N} nk} \quad ; k = 0,1,2,\ldots,N-1 \qquad (2\text{-}28)$$

Computation complexity of DFT is Θ (N^2) because DFT needs (N) multiplications and N-1 additions, where N is the data size. This makes computational complexity is extremely high especially if the data size (N) is enormous [17]. Modern applications uses Fast Fourier Transform (FFT) to compute DFT because it reduces the computation complexity to Θ ($N\ log\ N$).
Fast Fourier transform (FFT) is an algorithm published by mathematicians J.W. Cooley and John Tukey in 1965 [18]. Cooley-Tukey algorithm reformulate discrete Fourier transform (DFT) to reduce time complexity from Θ (N^2) for DFT to Θ (N log N) for FFT. This reduction in time complexity is highly effective when the number of points are in thousands or even millions [19].

FFT divides DFT of length (N) to two DFTs of lengths $N_1 N_2$. If N_1= 2 then this is called radix-2. Radix-2 means that input data set of length (N) is reduced to two datasets of lengths ($\frac{N}{2}$). For example, 16-point DFT is reduced to two 8-point DFTs that are even reduced to four 4-point DFTs which are reduced to eight 2-point DFTs, this approach is called divide and conquer. In radix-2 method, there are ($\log_2 N$) stages as explained in the above example. Moreover, each stage requires N operations, that's why the complexity of FFT is Θ ($N\ log\ N$). DFT could be decomposed to two sets; a set of even points and a set of odd points, this is called decimation-in-time (DIT). Another approach is to decompose DFT to two sets: first half and second half, this is called decimation-in-frequency (DIF). DIT is the approach used in this project [20].

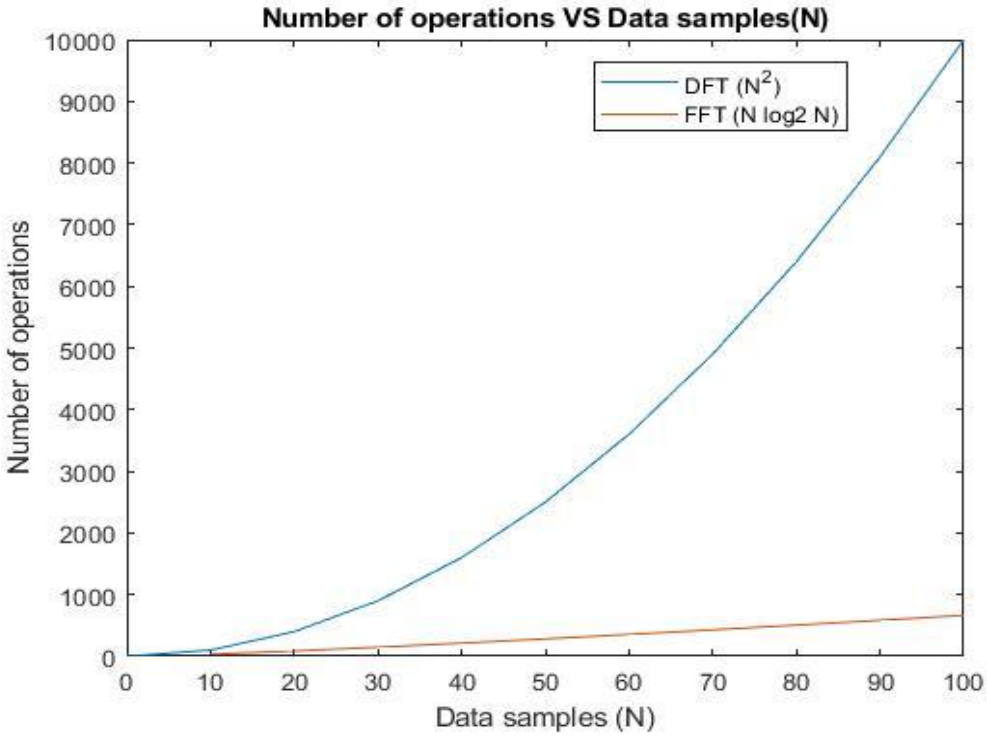


Figure 2- 10: Matlab generated graph representing number of operations versus data sample of size (N)

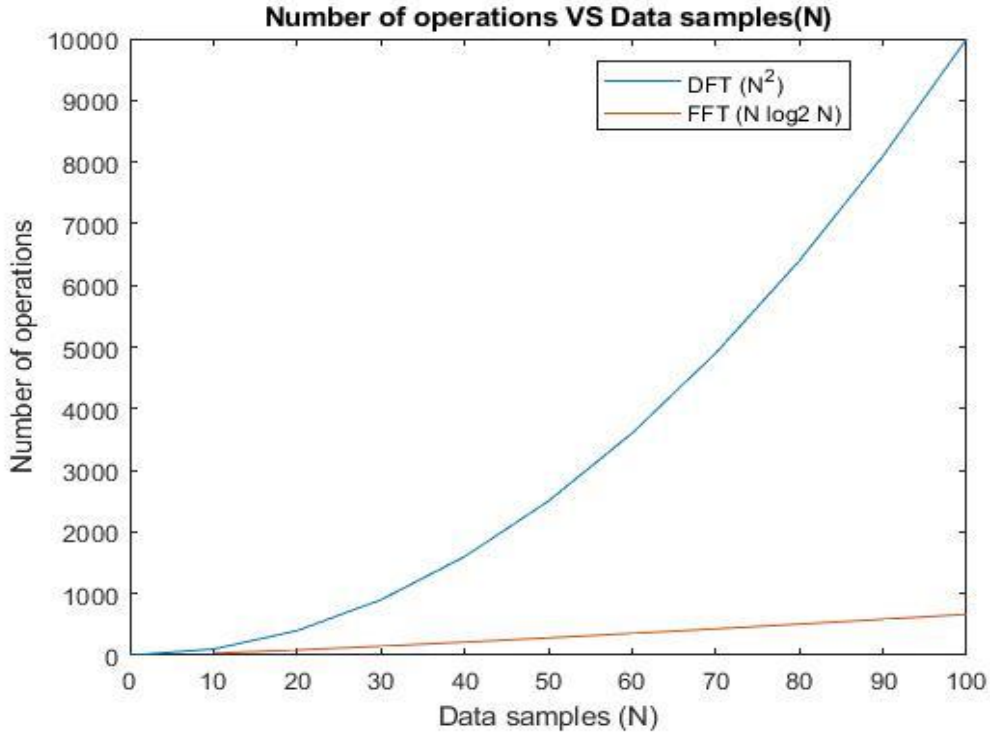


Figure 2- 11: Matlab generated graph representing number of operations versus data sample of size (N)

compares between DFT and FFT according to number of operations versus data size (N). It's shown that computational complexity increases explosively as the

number of operations increase relative to $N \log_2 N$. In conclusion, FFT is extremely faster than DFT that's why it is implemented in this project.

Cooley-Tukey algorithm

DFT for a sequence X_n is in the form:

$$X_k = \sum_{n=0}^{N-1} X_n \, e^{\frac{-2\pi i}{N} nk} \quad ; k = 0,1,2,\ldots N-1 \quad (2\text{-}29)$$

Radix 2 Decimation in Time (DIT) Cooley – Tukey algorithm divides DFT of size N to two smaller DFTs of size $\frac{N}{2}$, the first DFT is summation of even indexed points (n =2m) and the second DFT is summation of odd indexed points (n = 2m+1).

$$X_k = \sum_{m=0}^{N/2-1} x_{2m} \, e^{\frac{-2\pi i}{N}(2m)k} + \sum_{m=0}^{N/2-1} x_{(2m+1)} \, e^{\frac{-2\pi i}{N}(2m+1)k} \quad (2\text{-}30)$$

Factorizing out the term $e^{\frac{-2\pi i}{N} k}$ from odd part

$$X_k = \sum_{m=0}^{N/2-1} x_{2m} \, e^{\frac{-2\pi i}{N}(2m)k} + e^{\frac{-2\pi i}{N} k} \sum_{m=0}^{N/2-1} x_{(2m+1)} \, e^{\frac{-2\pi i}{N}(2m)k} \quad (2\text{-}31)$$

$$X_k = X_{ev} + e^{\frac{-2\pi i}{N} k} X_{odd} \quad (2\text{-}32)$$

Where, X_{ev} is DFT for even indexed inputs, X_{odd} is DFT for odd indexed inputs, $e^{\frac{-2\pi i}{N} k}$ is the N^{th}primitive roots of the unity which is called also twiddle factor (W_N).

For even part,

$$X_{ev(k+\frac{N}{2})} = \sum_{m=0}^{N/2-1} x_{2m} \, e^{\frac{-2\pi i}{N}(2m)(k+\frac{N}{2})} \quad (2\text{-}33)$$

$$X_{ev(k+\frac{N}{2})} = \sum_{m=0}^{N/2-1} x_{2m} \, e^{\frac{-2\pi i}{N}(2m)k} \, e^{-2\pi m i} \quad (2\text{-}34)$$

$$\because e^{-2\pi m i} = 1$$

$$X_{ev(k+\frac{N}{2})} = \sum_{m=0}^{N/2-1} x_{2m} \, e^{\frac{-2\pi i}{N}(2m)k} = X_k \quad (2\text{-}35)$$

The results proofs that, even part is periodic, $X_{ev(k+\frac{N}{2})} = X_k$. The same result for odd part, $X_{odd(k+\frac{N}{2})} = X_k$

For the twiddle factor,

$$e^{\frac{-2\pi i}{N}(k+\frac{N}{2})} = e^{\frac{-2\pi i}{N}k} e^{-i\pi} = -e^{\frac{-2\pi i}{N}k} \quad (2\text{-}36)$$

It's shown that twiddle factor is also periodic.

From equations (2-35) and (2-36),

$$X_{k+N/2} = X_{ev(k)} - e^{\frac{-2\pi i}{N}k} X_{odd(k)} \quad (2\text{-}37)$$

According to equations (2-32) (2-36) (2-37), the periodic nature of even part, odd part and twiddle factors decreases the computation complexity drastically as FFT uses intermediate results previously calculated. Figure 2- 12 below illustrates clearly radix-2 DIT for N points FFT.

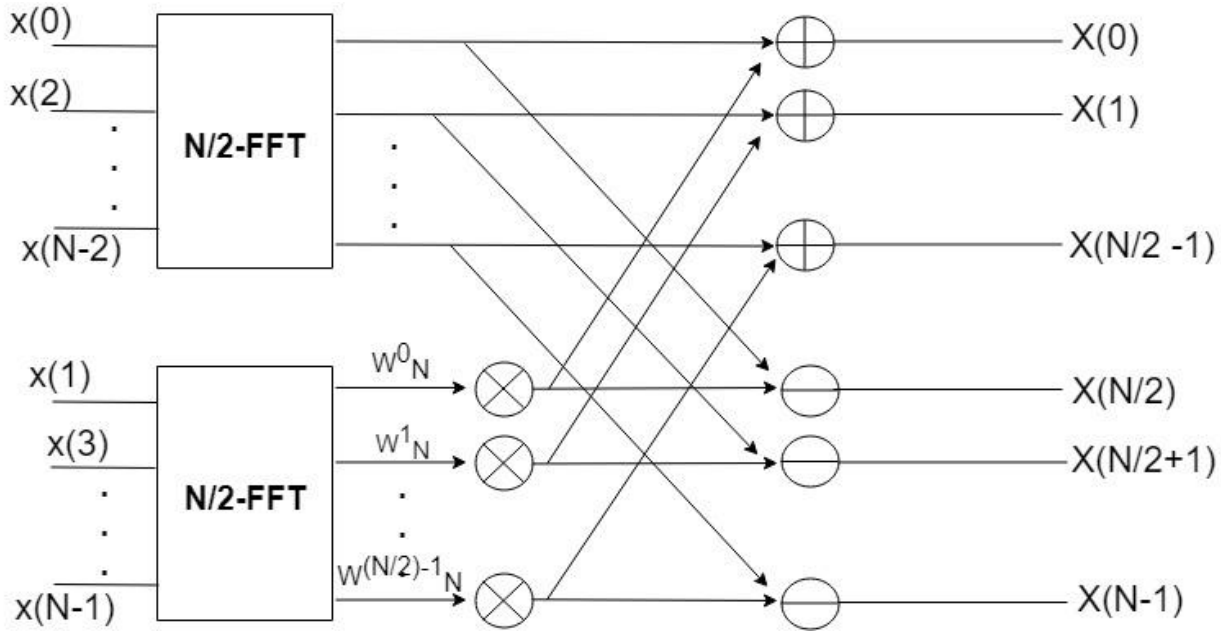


Figure 2- 11: Radix-2 DIT N-FFT graphical representation

Inverse FFT

Inverse fast Fourier transform (IFFT) could be implemented easily and directly from original forward FFT. The following is a proof for this fact [21]. First, inverse DFT (IDFT) equation is,

$$X_n = \frac{1}{N}\sum_{N=0}^{N-1} X(k)e^{\frac{j2\pi kn}{N}} \quad (2\text{-}38)$$

$e^{\frac{j2\pi kn}{N}}$ Could be written in the form $\cos\left(\frac{n\pi kn}{N}\right) + j\sin(\frac{n\pi kn}{N})$, and $X(k)$could be written in the form $X_{real}(k) + j\,X_{im}(k)$, substituting in (2-38).

$$X_n = \frac{1}{N}\sum_{N=0}^{N-1}[X_{real}(k) + j\,X_{im}(k)] * [\cos\left(\frac{n\pi kn}{N}\right) + j\sin(\frac{n\pi kn}{N})] \quad (2\text{-}39)$$

Multiplication from equation (2-39) is carried out.

$$X_n = \frac{1}{N}\sum_{N=0}^{N-1}\left[X_{real}(k)\cos\left(\frac{n\pi kn}{N}\right) - X_{im}(k)\sin\left(\frac{n\pi kn}{N}\right)\right] + j$$
$$\left[X_{im}(k)\cos\left(\frac{n\pi kn}{N}\right) + X_{real}(k)\sin\left(\frac{n\pi kn}{N}\right)\right] \quad (2\text{-}40)$$

Equation (2-40) is the final form of IDFT.
If $X_{real}\ and\ X_{im}\ are\ swapped, then\ forward\ DFT\ is\ applied.$

$$X_n = \sum_{N=0}^{N-1}[X_{im}(k) + j\,X_{real}(k)] * [\cos\left(\frac{n\pi kn}{N}\right) - j\sin(\frac{n\pi kn}{N})] \quad (2\text{-}41)$$

Multiplication from equation (2-41) is carried out.

$$X_n = \sum_{N=0}^{N-1}\left[X_{im}(k)\cos\left(\frac{n\pi kn}{N}\right) + X_{real}(k)\sin\left(\frac{n\pi kn}{N}\right)\right] + j$$
$$\left[X_{real}(k)\cos\left(\frac{n\pi kn}{N}\right) - X_{im}(k)\sin\left(\frac{n\pi kn}{N}\right)\right] (2\text{-}42)$$

Swapping real part with imaginary part from equation (2-42) and divide by N,

$$X_n = \frac{1}{N}\sum_{N=0}^{N-1}\left[X_{real}(k)\cos\left(\frac{n\pi kn}{N}\right) - X_{im}(k)\sin\left(\frac{n\pi kn}{N}\right)\right] + j$$
$$\left[X_{im}(k)\cos\left(\frac{n\pi kn}{N}\right) + X_{real}(k)\sin\left(\frac{n\pi kn}{N}\right)\right] (2\text{-}43)$$

Equation (2-43) is the same exactly as equation (2-40), this means that forward FFT calculations can be manipulated resulted in IFFT.

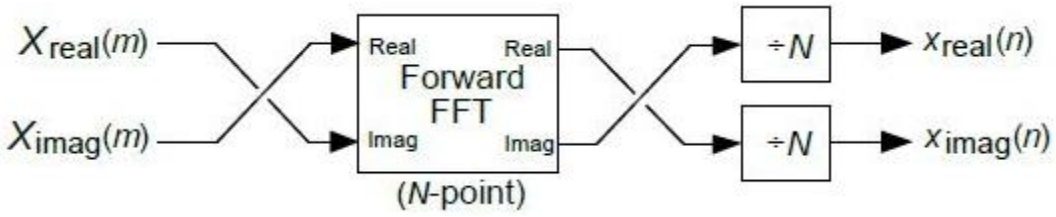


Figure 2- 12: How to make IFFT from FFT source

The following Figure 2- 13 [21] summarize how to obtain IFFT from FFT to save hardware resources.

The final algorithm for Wavelet Packet Entropy is as follows:

- Perform FFT on the ECG signal.
- Perform wavelet packet decomposition.
- Perform inverse Fourier transform for the decomposed data.
- Calculate the entropy of the wavelet packet.

This is summarized in the following flowchart.

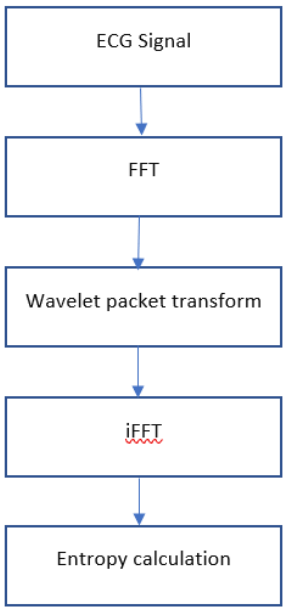


Figure 2- 13: Flowchart for wavelet packet entropy

References

[1] V. Vapnik, S. E. Golowich, and A. Smola, "Support vector method for function approximation, regression estimation, and signal processing," Advances in neural information processing systems,
pp. 281-287, 1997.
[2] R. S. Shah, "Support vector machines for classification and regression," McGill University, 2007
[3] C. Campbell and Y. Ying, "Learning with support vector machines," Synthesis lectures on artificial intelligence and machine learning, vol. 5, no. 1, pp. 1-95, 2011.
[4] J. Platt, "Sequential minimal optimization: A fast algorithm for training support vector machines," 1998.
[5] A. Sergiyenko, D. Ivanov and J. Vinogradov, "High Speed AR Analysis Based on FPGA."
[6] "14.1 - Autoregressive Models | STAT 501", Newonlinecourses.science.psu.edu. [Online]. Available: https://newonlinecourses.science.psu.edu/stat501/node/358/.
[7] C. Chatfield, The analysis of time series An Introduction, 6th ed. pp. 22 - 24.
[8] "Compute estimate of autoregressive (AR) model parameters using Burgmethod - Simulink", Mathworks.com. [Online]. Available: https://www.mathworks.com/help/dsp/ref/burgarestimator.html

[9] B. Jiao, "High Level FPGA Implementation Of Adaptive Signal Segmentation And Autoregressive Modeling Techniques", 2019. [Accessed 12 June 2019].

[10] I. Kauppinen, V. Välimäki and P. Esquef, "Frequency warped Burg's method for AR-modeling", 2003. [Accessed 12 June 2019].

[11]C. Shannon, "A Mathematical Theory of Communication", Bell System Technical Journal, vol. 27, no. 4, 1948. Available: 10.1002/j.1538-7305.1948.tb00917.x.

[12]Z. He, X. Chen and Q. Qian, "A study of wavelet entropy measure definition and its application for fault feature pick-up and classification", Journal of Electronics, vol. 24, no. 5, pp. 628-634, 2007. Available: 10.1007/s11767-005-0253-0.

[13]W. Ting, Y. Guo-zheng, Y. Bang-hua and S. Hong, "EEG feature extraction based on wavelet packet decomposition for brain computer interface", Measurement, vol. 41, no. 6, pp. 618-625, 2008. Available: 10.1016/j.measurement.2007.07.007.

[14]T. Li and M. Zhou, "ECG Classification Using Wavelet Packet Entropy and Random Forests", Entropy, vol. 18, no. 8, p. 285, 2016. Available: 10.3390/e18080285.

[15] S. Smith, The scientist and engineer's guide to digital signal processing. [San Diego, Calif.]: California Technical Pub., 2002, p. Ch8.

[16] R. Amirfattahi, “Calculation of computational complexity for radix-2p fast fourier transform algorithms for medical signals,” Journal of medical signals and sensors, vol. 3, no. 4, p. 217, 2013.

[17] S Haykin and M Moher, Communication Systems, 5th ed. (New Jersey: John Wiley & Sons Inc., 2009).

[18] J. W. Cooley and J. W. Tukey. ”An algorithm for the machine calculation of complex Fourier series.” Math. ofComput.,vol. 19, pp. 297- 301. April 1965.

[19] P Embree, C Algorithms For Real-Time DSP (Upper Saddle River, NJ.: Prentice-Hall, 1995).

[20] “Fast Fourier Transform (FFT) FAQ,” dspGuru. [Online]. Available: https://dspguru.com/dsp/faqs/fft/. [Accessed: 5-Jun-2019].

[21] R. G. Lyons, Understanding digital signal processing. Upper Saddle River, NJ: Prentice Hall, 2010

Chapter3: Software Implementation by Matlab

Introduction:

Depending on what discussed earlier, Matlab was used to implement the feature extraction algorithms, SVM training and testing. Our goal is finding the highest possible accuracy, but some definitions should be discussed before defining the accuracy. After the 2 algorithms extracts the features, these features are entered to the SVM to be trained. After training, the test vectors are entered to the classifier to identify the class of each vector. All the results of the classifier are extracted in a matrix called Binary classification confusion matrix.

Figure 3- 1 shows how the Binary classification confusion matrix is. It consists of 4 sections.

1- **True Positive (TP):** This section represents the number of the **correct** predictions of the first class. In other words, the data which is actually positive and predicted to be positive.
2- **False Negative (FN):** This section represents the number of the **wrong** predictions of the first class. In other words, the data which is actually positive and predicted to be negative.
3- **False Positive (FP):** This section represents the number of the **wrong** predictions of the second class. In other words, the data which is actually negative and predicted to be positive.
4- **True Negative (TN):** This section represents the number of the **correct** predictions of the second class. In other words, the data which is actually negative and predicted to be negative. [1]

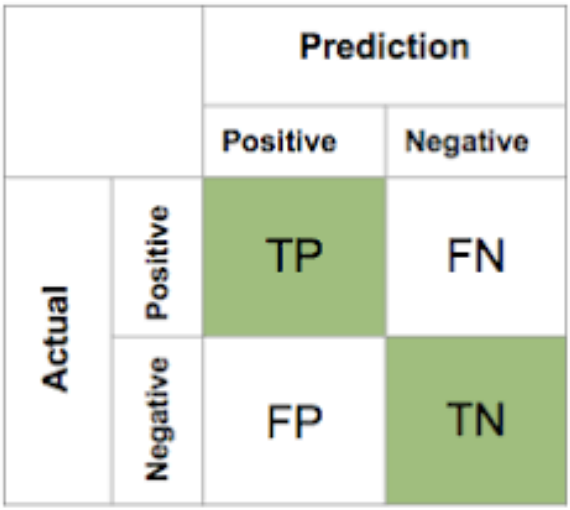


Figure 3- 1 Binary classification confusion matrix

This table is used to calculate other parameters, which will be used to reach the definition of the accuracy. Firstly, the precision of a class, which is the number of correct predicted vectors of this class divided by the total number of the predicted vectors for the same class. For example, the precision of the positive class=**TP/(TP+FP),** and for the negative class= **TN/(TN+FN).** Another definition is the recall of a class, which is the number of the correct predicted vectors of this class divided by the actual number of vectors that already belongs to this class. For example, the recall of the positive class= **TP/(TP+FN),** and for the negative class=**TN/(TN+FP).** F1 score is another important definition, which is the average of the precision and recall of the class. Now the accuracy can be defined as the average of F1 scores of the classes.

In Table 3- 1 and Table 3- 2, an example of a binary classification confusion matrix, precision, recall and F1 score of patient number 212, using window length=1 second, and using the specified features which will be explained later.

	Predicted Positive	Predicted Negative
True Positive	337	23
True Negative	20	161

Table 3- 1 Binary classification confusion matrix

	Precision (%)	Recall (%)	F1 score (%)
Positive Class[ARR]	94.39	93.61	94.00
Negative Class[N]	87.50	88.95	88.22

Table 3- 2 Precision, Recall and F1 score example

The test accuracy is calculated by averaging the F1 scores. So accuracy= 91.11 %, which is the average of 94.00% and 88.22%.

The code implementation

With the help of model "Signal Classification Using Wavelet-Based Features and Support Vector Machines", the Matlab code was implemented [2]. Signal processing toolbox, statistics and machine learning toolbox and wavelet toolbox were used in this code. Firstly, the data entered to the matlab. Then, number of windows is calculated depending on the winnow size and time length of the input data. Using the annotation of the data input, the data is classified into normal and Arrhythmia, and the labels are extracted by the same way. In each window, if there are only one abnormal beat, the window is considered Arrhythmia.

All the windows are entered to the feature extraction function "helperExtractFeatures", which contains the two algorithms, AR and Shanon. The output of this function is 20 features for each window. All the windows are split into two sections, 70 % training and 30 % for testing. The training features are entered to the SVM for traing. The training was done using "fitcecoc" model. On the other hand, the remaining 30% of the features was entered to the classifier using "predict" function. The output of the classifier is compared to the annotation of these 30% of the data, and the results were reported.

During testing, it was expected to use cross validation technique but because it is time consuming, the 70-30 % strategy was used instead. Cross validation technique depends on randomizing the data, splitting it into k folds and make the training-testing step k times. Finally, averaging the results of the k folds which represent the accuracy. [3]

After defining the accuracy and implementing the code, all the patients form the MIT-BIH Arrhythmia database were tested using different training-testing strategies. Firstly, train the SVM by 70% of the patients and test by the remaining patients. Secondly, training the SVM by 70% of each patient and test by the remaining data of it, and repeat this for each patient.

The first strategy got 90% accuracy, but it is very time consuming. So the 6 patients that achieved the highest individual accuracy, were chosen to continue all the software simulations with. Table 3- 3 represents the accuracies of the 6 patients chosen with average accuracy 90.2%.

Patient number	106	119	205	212	213	215
Accuracy (%)	91.1	94.3	86.3	90.1	88.8	90.86

Table 3- 3the accuracies of the best 6 patients

Optimizations

There are many restrictions in the hardware that forced us to optimize many parameters. One of the most significant factors in the hardware is the area to fit the design on FPGA. To achieve so, some parameters should be optimized.

Window size

Window size is the period of ECG that is annotated and entered to the SVM to train with. The smaller the window size is, the larger the number of the training and testing data are.

Window size is a significant factor that affects the accuracy and also highly affects the area of the design. Extracting the features from 1 second ECG is so easier in terms of time and area than 10 seconds. Sweeping on the

window size was done versus the accuracy using AR features only. The result is shown in Figure 3- 2.

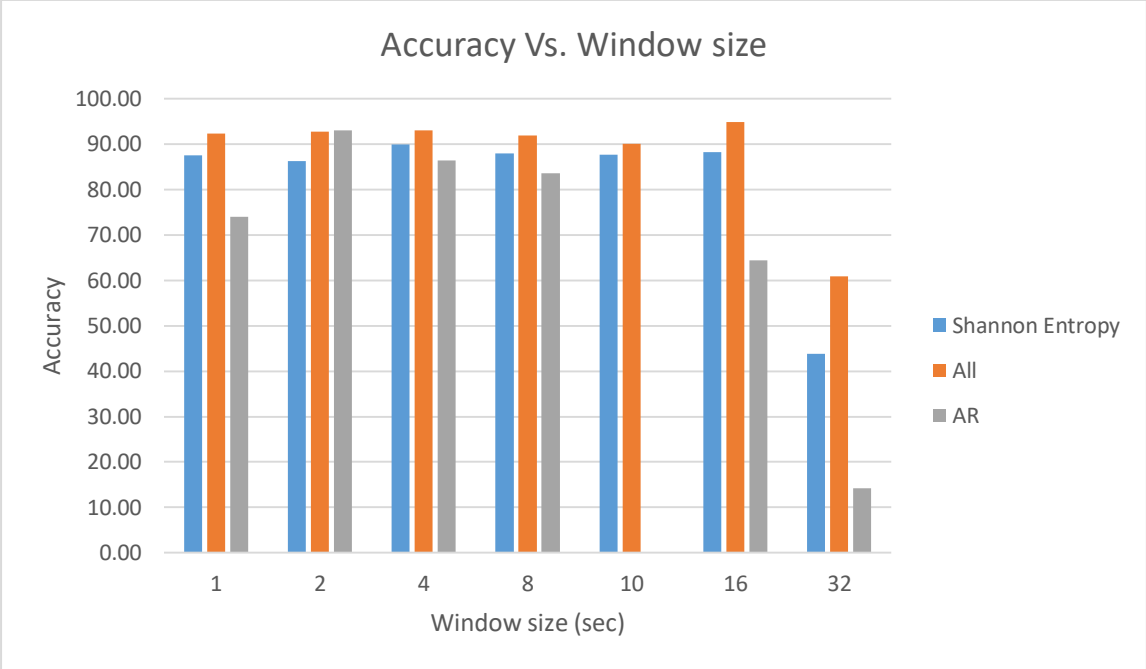


Figure 3- 2 sweeping on the window size for shanon only, AR only and both together

As shown in the figure, 2 seconds represents the best window length because it has the highest accuracy. On the other hand, 4 seconds window size had the second highest accuracy with the highest standard deviation. In fact, 1 second window length was implemented on the hardware due to the huge area of the 4 seconds window length. The accuracy dropped from 88.824 % to 88.10607 which is insignificant comparing to the area saved.

Number of features that entered to the SVM

Using all the features that extracted from the extraction algorithms leads to huge area and also does not lead to the highest accuracy. So sweeping on the features that entered to the SVM was a must. This sweeping is so time consuming as all the possible combinations of all the 20 features should be tested.

N (number of possible combinations) =

20C10+ 2(20C1+20C2+20C3+20C4+20C5++20C6++20C7+20C8+20C9) =

1,048,574 iteration.

Each iteration consumes 27 seconds on average as it changes with changing the number of features. So the total time for this task would be expected to consume 327 days. So it was a must in terms of time, hardware area and simplicity to lower the degree of the SVM from polynomial of degree 2 to linear.

In Figure 3- 1, sweeping on the number of features that entered to the SVM was done. The x-axis represents the number of features n, and the y-axis represents the mean of the accuracies for the 6 patients for the best combination of n features.

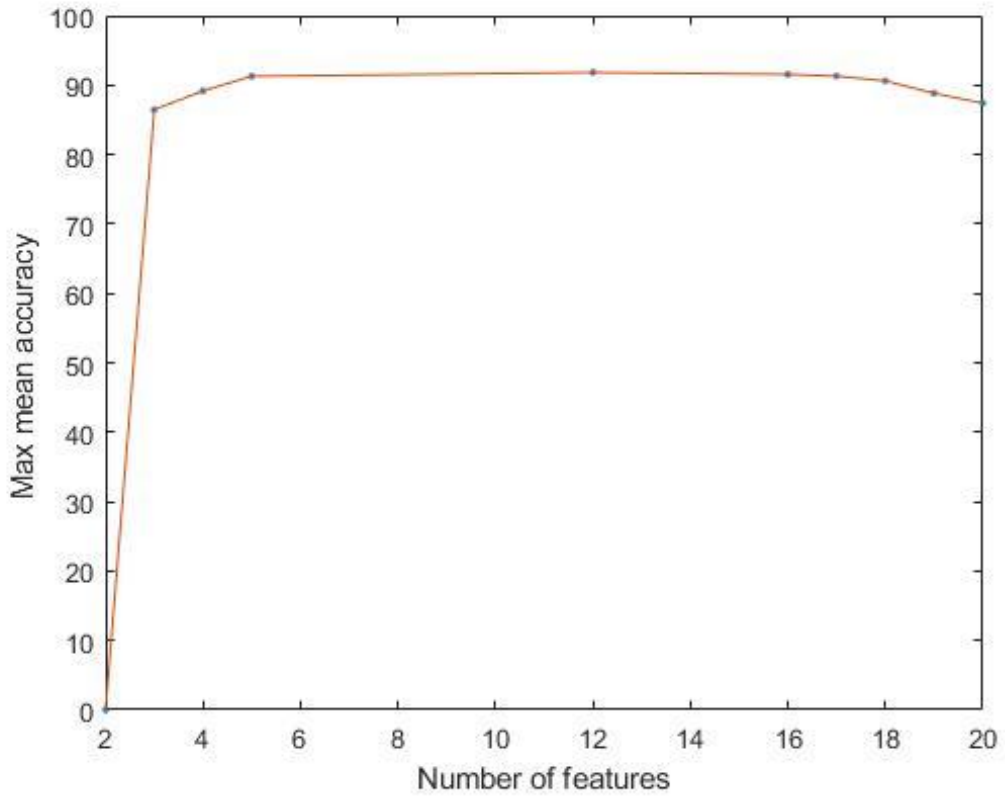


Figure 3- 3 The accuracy for the best combination of n features Vs the number of features n

From Figure 3- 1, it is concluded that around 12 features leads to the highest accuracy 91.83 %, but 12 is still a big number of features to be

implemented on the FPGA. Because Auto Regression algorithms extracts only 4 features, and its area is not sensitive to the number of features, all the 4 features of AR were chosen. On the other hand, Shanon algorithm's area depends on the number of features. So the best 4 features for Shanon were chosen to be implemented on the hardware, which are features number 1,2,3,9.

The accuracy of those 8 features reached 91.0957 % which is acceptable comparing to the highest accuracy with 12 features which is 91.83 %.

Decimal points for hardware implementation

The number of the decimal points of the data should be defined to be implemented on the hardware.

In Figure 3- 4, sweeping on the number of decimal points Vs average accuracy was done.

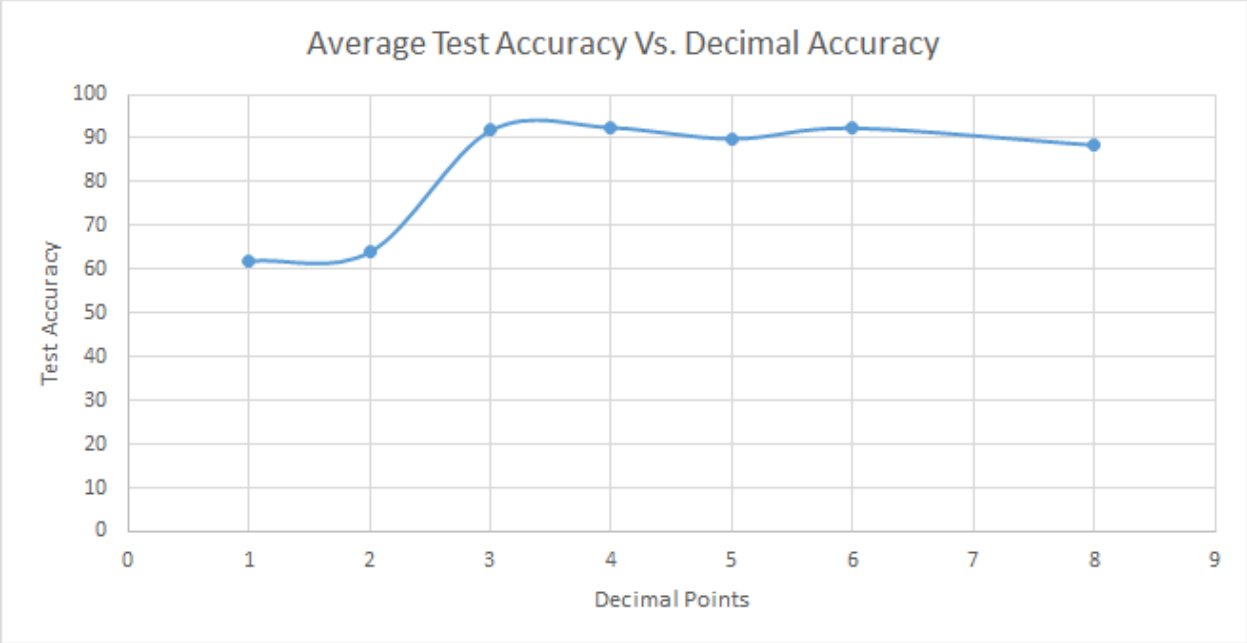


Figure 3- 4 Sweeping on the decimal points Vs the average accuracy of the 6 patients

It is concluded that 3 decimal points is the most suitable number as it has the highest accuracy. More than 3 decimal points will increase the area a lot, and will not increase the accuracy much.

Data resampling for reducing the area of the hardware

Because Shanon algorithm consumes huge area in the hardware, the sampling frequency had to be lower than 360 Hz. So the data was resampled with different frequencies and the accuracies were reported.

In Figure 3- 5, the sampling rate of Shanon extraction algorithm was fixed with some frequencies, which are 128 Hz, 64 Hz and 32 Hz. The sampling frequency of AR algorithm was swept, and the average accuracy of the 6-patients was reported.

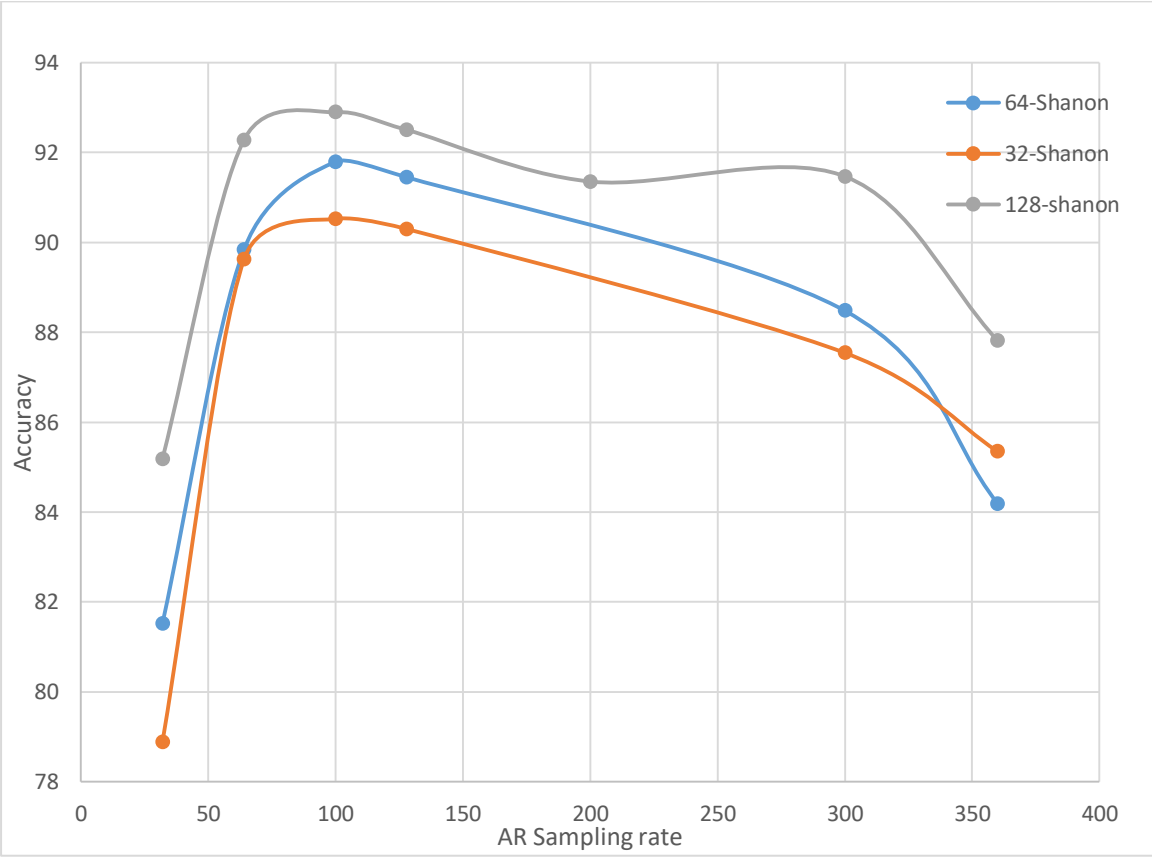


Figure 3- 5 sweeping on the sampling rate of Shanon and AR Vs the average accuracy

It is concluded that the highest accuracy is at 128 Hz sampling rate for Shanon, and 100 Hz sampling rate for AR. The accuracy at these frequencies reached 92.8969 %.

In Figure 3- 6, the raw data was graphed with the resampled data with frequency 64 Hz to confirm that there is no much difference between them.

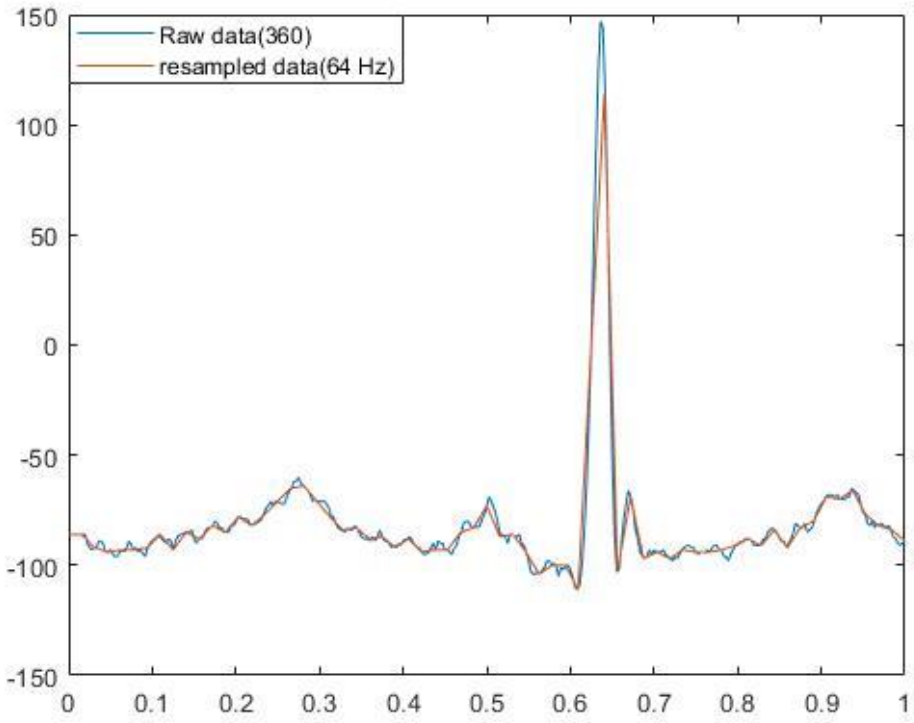


Figure 3- 6 *360 Hz unresampled data plot and 64 Hz resampled data*

References

[1] J. Davis and M. Goadrich, “The relationship between Precision-Recall and ROC curves,” *Proceedings of the 23rd international conference on Machine learning - ICML 06*, 2006.

[2]“modwpt,” *Signal Classification Using Wavelet-Based Features and Support Vector Machines - MATLAB & Simulink Example*. [Online]. Available: https://www.mathworks.com/help/wavelet/examples/ecg-classification-using-wavelet-features.html. [Accessed: 01-Dec-2018].

[3] “Selecting a classification method by cross-validation.” [Online]. Available: https://link.springer.com/content/pdf/10.1007/BF00993106.pdf. [Accessed: 12-Apr-2019].

Chapter 4: Hardware Implementation

Detrend Hardware Implementation

Detrend basic idea:

Detrend is a submodule which calculates the mean of input data, then subtract this mean from the input data.

As shown in Eq. 4.1 shown below, in order to calculate autocorrelation coefficient; the sample mean is calculated first, then it is subtracted from the sample elements. Detrend turns equation 4.1 into a simpler equation shown in equation 4.2.

$$a = \frac{\sum_{t=1}^{N-1}(x_t - \bar{x}\,)(x_{t+1} - \bar{x})}{\sum_{t=1}^{N}(x_t - \bar{x})^2} \qquad (Eq.\ 4.1)$$

$$a = \frac{\sum_{t=1}^{N-1}(detrendOutput_t\,)(detredOutput_{t+1})}{\sum_{t=1}^{N}(detrendOutput_t)\,(detrendOutput_t\,)} \qquad (Eq.\ 4.2)$$

Detrend is needed for both AR and SE blocks. Thus, instead of implementing it inside the algorithms, it is shared between the hardware for better resource management.

Detrend can be divided into two submodules; the first one for mean calculation, the second one for subtraction this mean from the data. In mean calculation, as the total number of elements is well-known before the synthesis, we divide first then sum these divided signals (Eq. 4.3) instead of summing the data then dividing by the total number of elements (Eq. 4.4). If addition is done first, the bit width should be increased be one bit to accommodate the result of adding two large numbers. However, this method produces smaller numbers after addition and this preserves the bit width of the module.

$$mean = \frac{x_1 + x_2 + \cdots + x_N}{N} \qquad (Eq.\ 4.3)$$

$$mean = \frac{x_1}{N} + \frac{x_2}{N} + \cdots + \frac{x_N}{N} \quad (Eq.\ 4.4)$$

Every element has 22 bits width which corresponds to 1 bit for sign, 11 bits for integer and 10 bits for decimal numbers. The input of detrend is from 10-bit ADC which produces integers only. Additional bit is added in our module, Since 11 bits for integer. 10 bits for decimal is justified in chapter 3 which gives small error and does not produce huge hardware.

Two versions are designed for detrend module; an initial version followed by more optimized final one. Both are discussed below:

Initial version:

This version has large hardware, however, produces all outputs in only one clock cycle. It takes all inputs at once, divide all inputs, then sum these divided inputs to form mean signal. The mean is subtracted from the input signals as shown in diagram **Error! Reference source not found.**. The method produces high throughput output. However, it uses blocks of division and subtraction as the number of data elements which consumes hardware and power consumption. It can be shown that the throughput is not as important as hardware in our system. The calculations can be done over multiple cycles and the design is still real time design which respects its deadline. This is the idea of the final design.

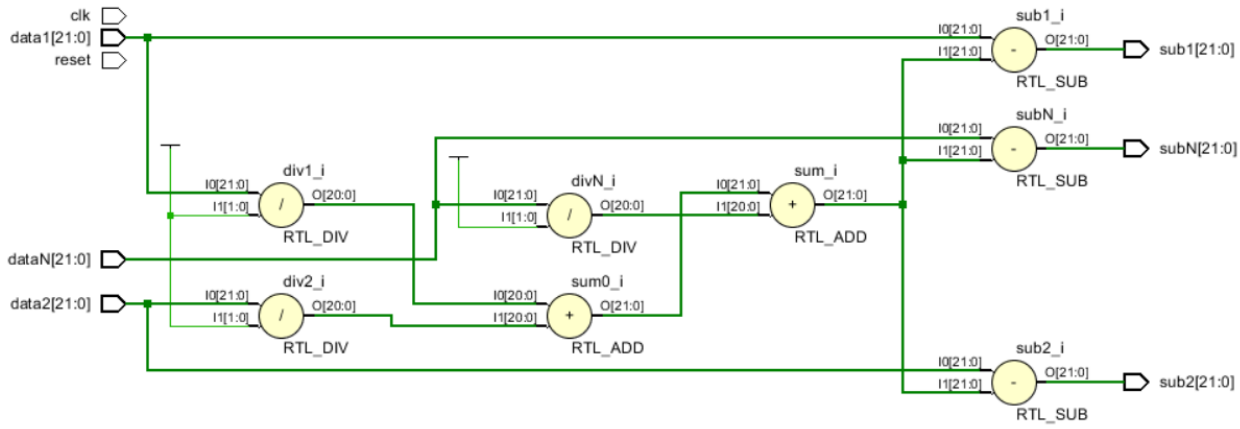


Figure 4- 1: A descriptive diagram for the hardware (not the entire hardware)

Final design:

The new design uses resources sharing instead of using multiple hardware components. Instead of using N elements of dividers, one element is used, and the design iterates on this single hardware for the calculations. This design sacrifices the throughput as the output now takes multiple clock cycles to be ready. Also, it incorporates signals that communicate validity of outputs between the submodules. Figure 4- 2 shows the idea of the new design.

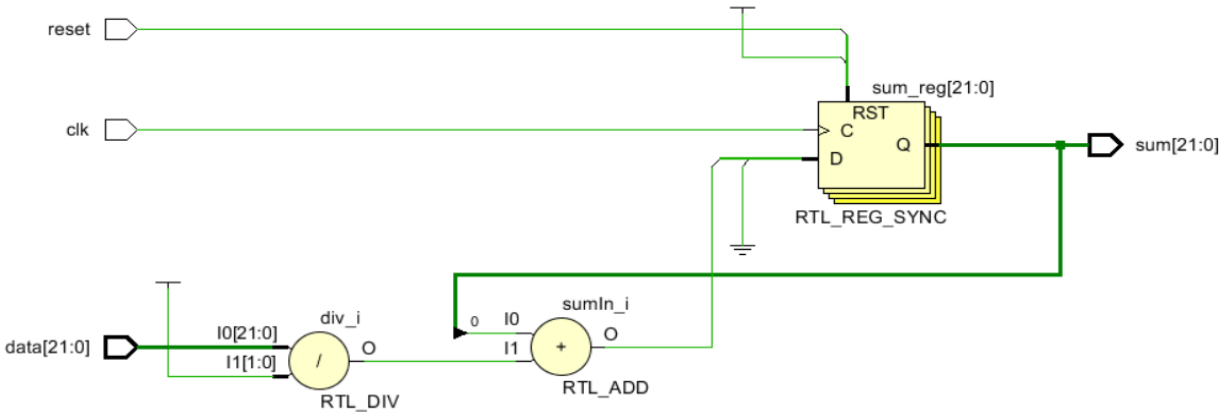


Figure 4- 2 The idea of iterating on single element

This resource sharing design consumed much less hardware as shown in Figure 4- 3. The first and final design is synthesized on Xilinx Zynq Ultrascale+ 104 Evaluation Board. Also, each has 360 data element which corresponds to one second of data in ECG signal.

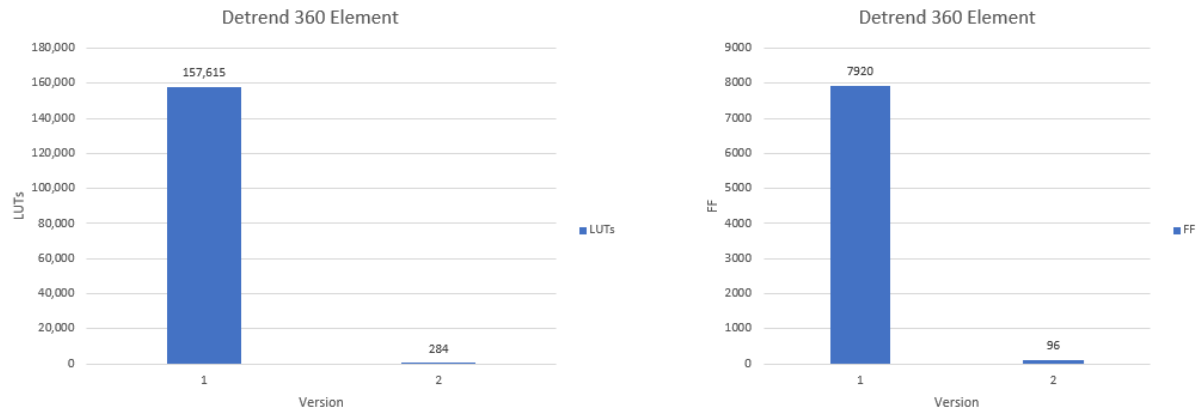


Figure 4- 3 LUTs and FF synthesized for every version

The final design is implemented into two submodules. Each has its counter and control signals for proper operation. As shown in the **Error! Reference source not found.**, the mean submodule begins operation when receives a data valid signal. It iterates until it produces a valid mean to the Subtractor submodule. When the Subtractor receives mean valid and data valid, it subtracts the mean from the data, then write this data in the Memory with correct address which depends on the number of the input element. When a new window is received the module starts the operation from the beginning and disables writing at the memory until a valid output is ready to be written again.

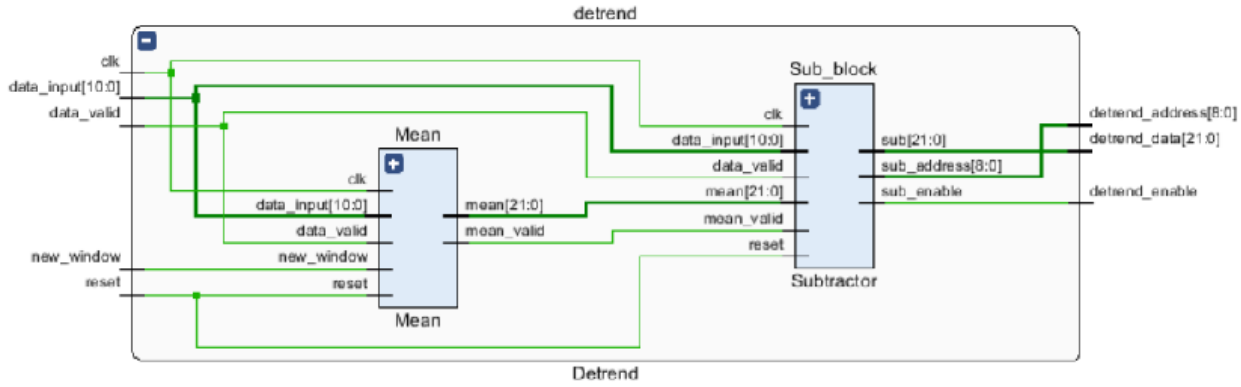


Figure 4- 4 Submodules of Final Detrend Design

AR Hardware Implementation

As explained in chapter 2; Eq. 2-19, Eq. 2-21, equation 2-22 and equation 2-23 are used to implement AR model using Burg method. In this hardware implementation, the inputs are loaded from the incorporated memory. These inputs are used in forward prediction error (called Efp in hardware) and backward prediction error (called Ebp in hardware). Through multiplication and division operations as shown in equation 2-19, the autocorrelation coefficient is calculated (called k in hardware). Then this coefficient is used to produce new Efp, Ebp and AR coefficients which in the next cycle uses these new Efp and Ebp to produce new K as in the lattice structure shown in chapter 2.

AR(4) is used in our hardware implementation which produces four AR coefficient at the end of the four cycles. Also, 22 bit for every data element is used like detrend implementation for the same reason explained above. The module can be divided into two submodules; first one calculates autocorrelation coefficient (K) and the second one updates Efp, Ebp and AR coefficients. Three versions are designed for AR model. In each version, the main goal is to minimize hardware resources and lower power consumption. A brief explanation about the difference between three versions are discussed below.

Initial Version:

In this version, a series implementation is used. Stage0 is followed by stage1 which is followed by stage2 etc. So, the output is produced in one clock with a huge combinational path and hardware resources to implement basic idea of lattice structure shown below in Figure 4- 5.

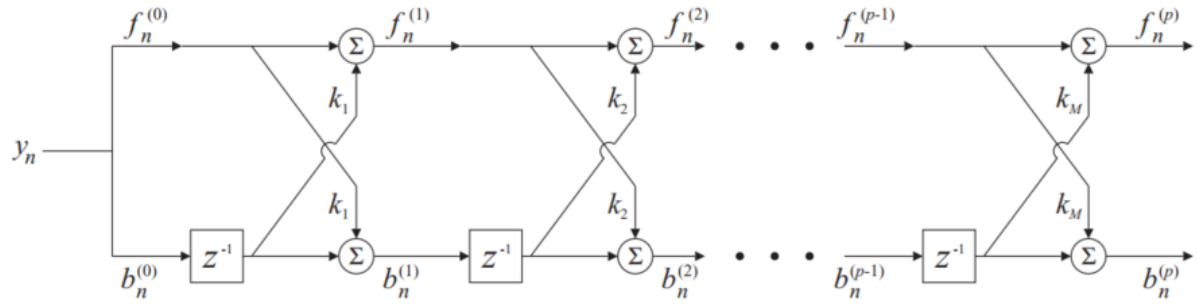


Figure 4- 5 Lattice structure for forward and backward prediction error [ref r25]

Second Version:

As can be seen from Figure 4- 5, the same hardware is repeated order of AR times to implement the algorithm. Instead of cascading these stages like in version one, one stage is implemented. Thus, counters and control signals are used to iterate multiple times depending on the order of AR (In our case four). This is reduced the hardware nearly four times which is a significant reduction in hardware and power consumption by sacrificing throughput. However, lower combinational path results in lower delay and higher performance.

Third Version:

Final version is built upon the second version. The second version is splitted into two submodules. The first module is optimized for smaller hardware by using resource sharing and multiple cycles to produce valid output. The second module is optimized for throughput and produces its output in one single cycle.

The below Figure 4- 6 shows how hardware resources are reduced by synthesizing a 64 data element of each version of the three designs on Xilinx Zynq Ultrascale+ FPGA.

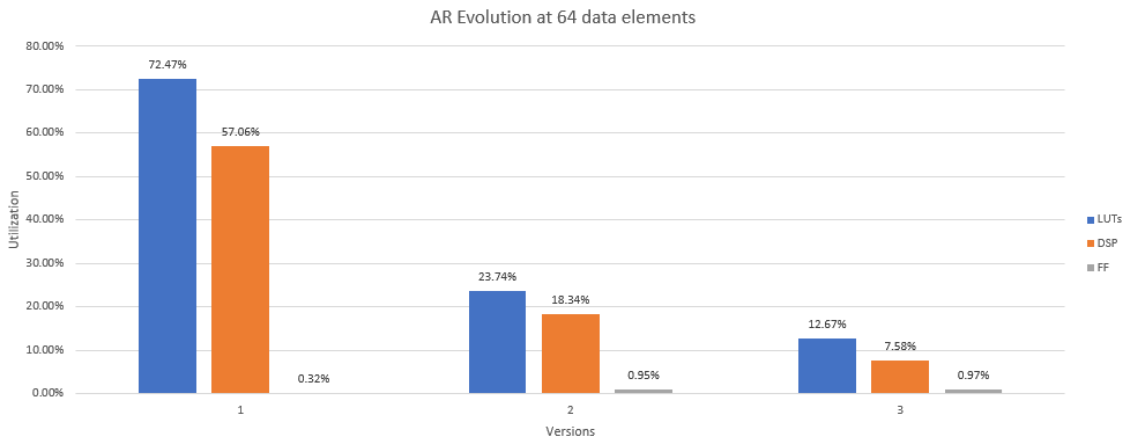


Figure 4- 6 AR reduction in hardware resources at 64 data elements

The synthesis results in the above graph contains memory besides AR algorithm which adds overhead in the results. However, this overhead is same for the three versions as the exact memory is being synthesized. This memory is important in AR system to be able to interface with the input without exceeding IO requirements of the board. The figure shows important aspect in our system design. The first two versions cannot implement a full second of ECG which requires 360 data element AR model on the zynq FPGA.

Final AR design description:

As shown in Figure 4- 7 and similar to detrend module, the submodules communicate by valid signals. When the memory data is valid, the Ep Submodule passes these data to K Submodule with a valid signal. K Submodule produces autocorrelation coefficient in multiple cycles equal to the number of data elements. Then, give it to Ep Submodule to update its coefficient. This loop continues until the final output is valid. Clock, reset and new window is omitted from the schematic for cleaner preview of the system. However, they are inputs for every submodule.

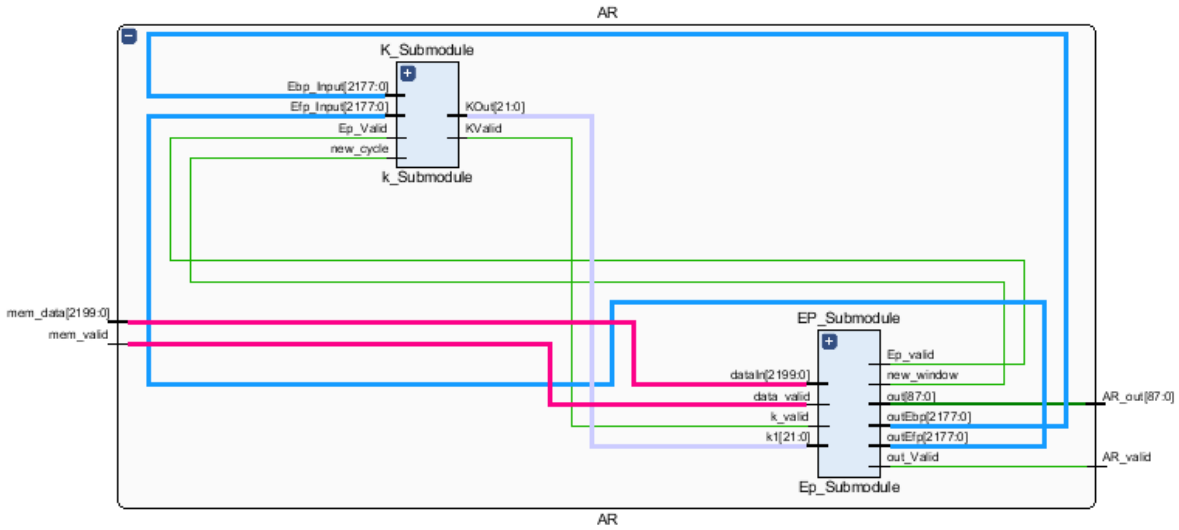


Figure 4- 7 AR Submodules

Final AR system Results:

All results of this subsection are produced from implementation on Xilinx Zynq UltraScale+ ZCU 104 Evaluation Board.

360 Element AR standalone System:

This system implements complete second of ECG signal with detrend, memory and AR modules as shown in Figure 4- 8. The system consumes 143,429 LUTs and 723 DSPs as shown in Figure 4- 9. This system is implemented on clock

frequency 10 KHz and consumes nearly 1 mW of dynamic power (excluding I/O Power) as shown in Figure 4- 10.

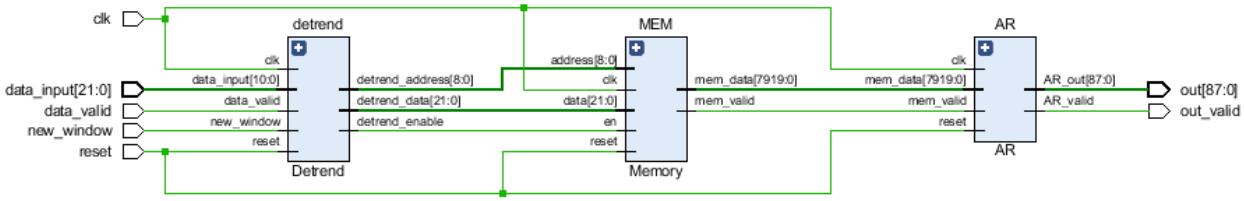


Figure 4- 8 Schematic of one second (360 element) standalone AR system

Resource	Utilization	Available	Utilization %
LUT	143429	230400	62.25
FF	24359	460800	5.29
DSP	723	1728	41.84
IO	104	360	28.89
BUFG	3	544	0.55

Figure 4- 9 Implementation resources for the system

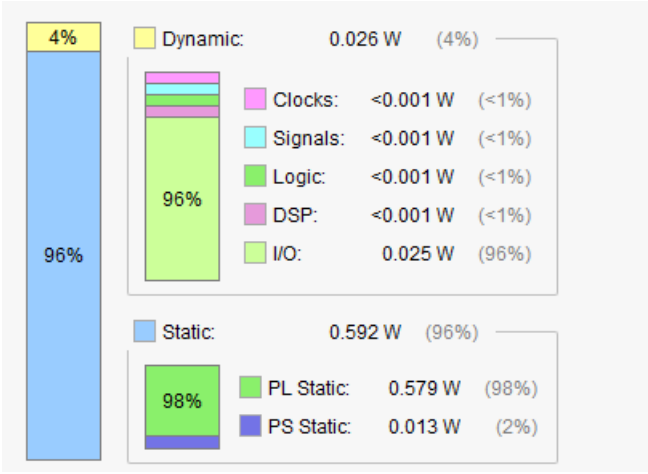


Figure 4- 10 Power report summary of the system

AR System that will be combined with Entropy module:

It is found in chapter 3 that it is better to resample the AR to 100 elements instead of 360 which gives higher F1_Scores and lower hardware resources. The system is implemented without detrend module as shown in Figure 4- 11. AR Sampling module is a very simple module just to adjust connectivity between the memory and AR resampled system. The hardware resources and power consumption figures are shown in Figure 4- 12, Figure 4- 13 and Figure 4- 14. The power consumption is less than one mW for 100 KHz frequency which more than enough to respect the deadline of the system and equals to 48 mW for 10 MHz which is implemented for comparing reasons only to show how the system behaves and consumes power at high frequencies. Figure 4- 15 shows how dynamic power excluding (I/Os) changes with higher frequencies.

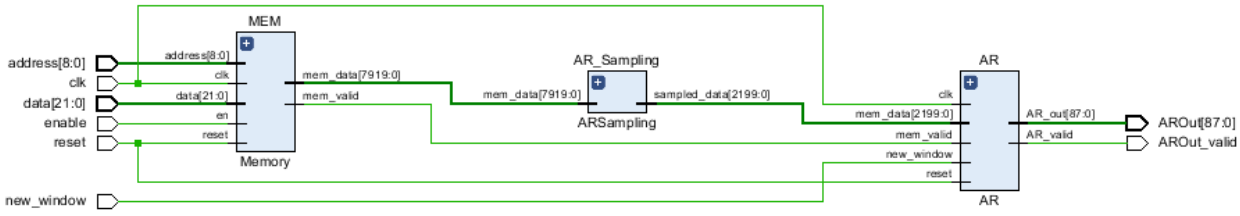


Figure 4- 11 Resampled AR system down to 100 from 360 elements

Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	42548	230400	18.47
FF	6879	460800	1.49
DSP	203	1728	11.75
IO	124	360	34.44
BUFG	3	544	0.55

Figure 4- 12 Utilization of hardware resources of AR resampled hardware

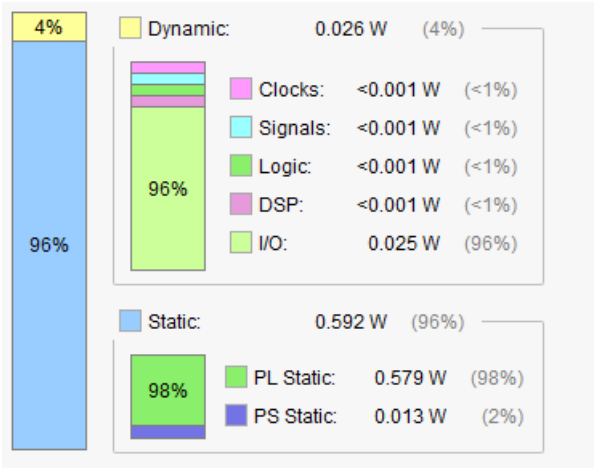


Figure 4- 13 Power consumption for 100 KHz clock frequency

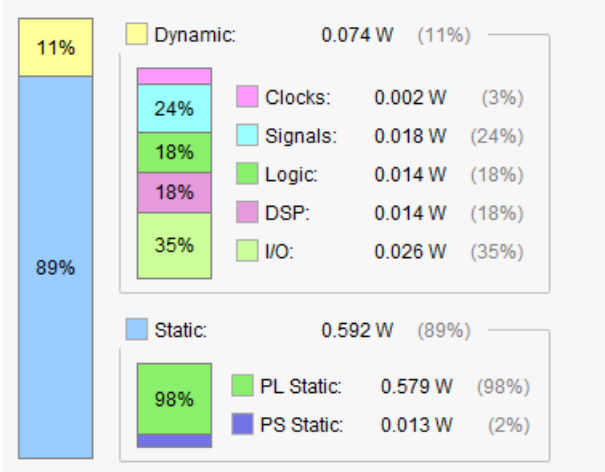


Figure 4- 14 Power consumption for 10 MHz clock frequency

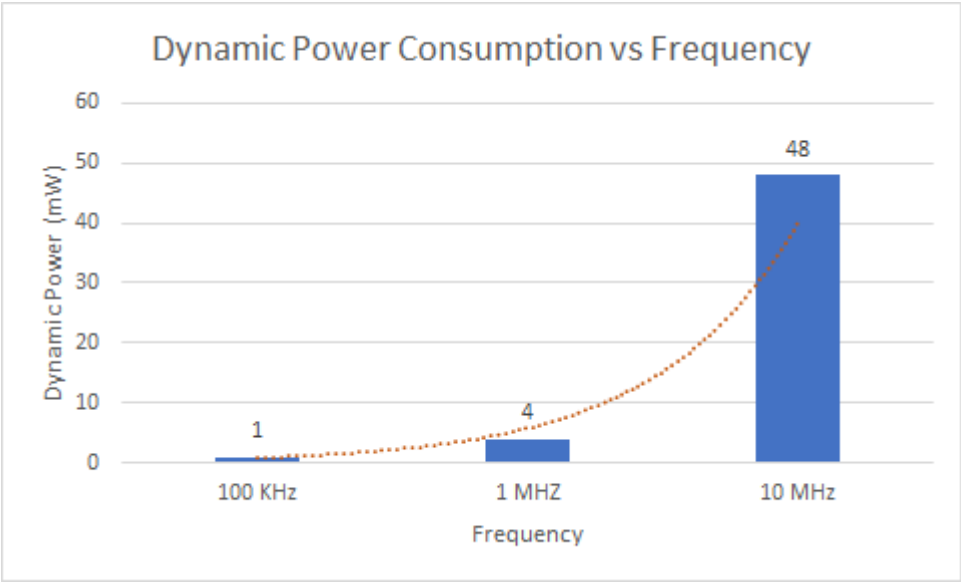


Figure 4- 15 Dynamic power Vs the Frequency

Section 2: Wavelet Packet Entropy by Wavelet Packet Decomposition and Entropy

Initial Implementation

The algorithm in the following flowchart is the one to be implemented to hardware architecture.

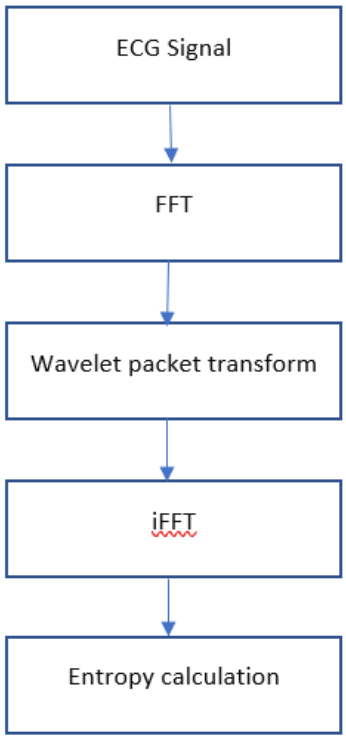
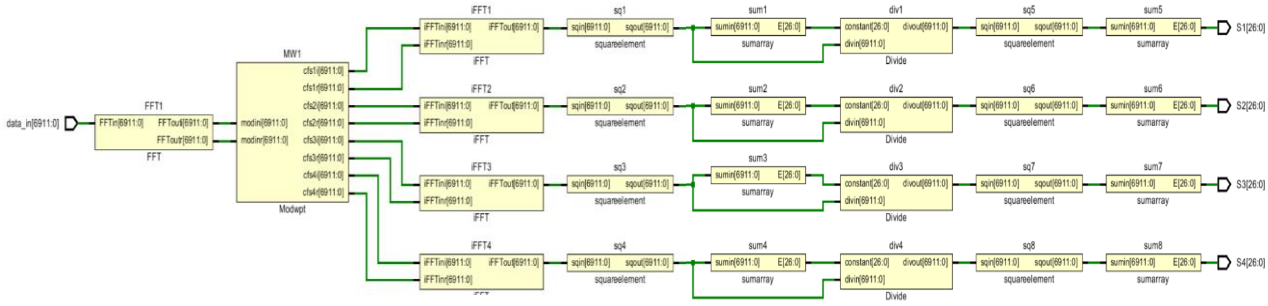


Figure 4- 16: Modwpt and entropy algorithm

The first hardware is implemented to translate the algorithm got from the software to a functional architecture that would be optimized later to get a first sight on the architecture of the algorithm at hardware implementation. The algorithm implemented is that for getting the four specified features combined giving highest accuracy determined from the simulations. The architecture was implemented exactly like the software simulations to give the output directly with combinational hardware only, consisting of one of both *FFT* and *Modwpt* modules followed by four *inverse FFT* modules and *four dividers* and *8 array-element-squaring* -instead of 4 array-element squaring and 4 array-element-log and multiplication for the architecture to be easier in optimization and for the architecture to have uniformity-, and 8 *summation* modules for arrays of length

256 elements. Following is flowchart and architecture of the algorithm using only combinational hardware with no flipflops and no clocks, except for *Modwt* which would be discusse



d later. This architecture is shown in Figure 4- 17.

Figure 4- 17: Pre-optimization architecture

Optimizing techniques with Final top module

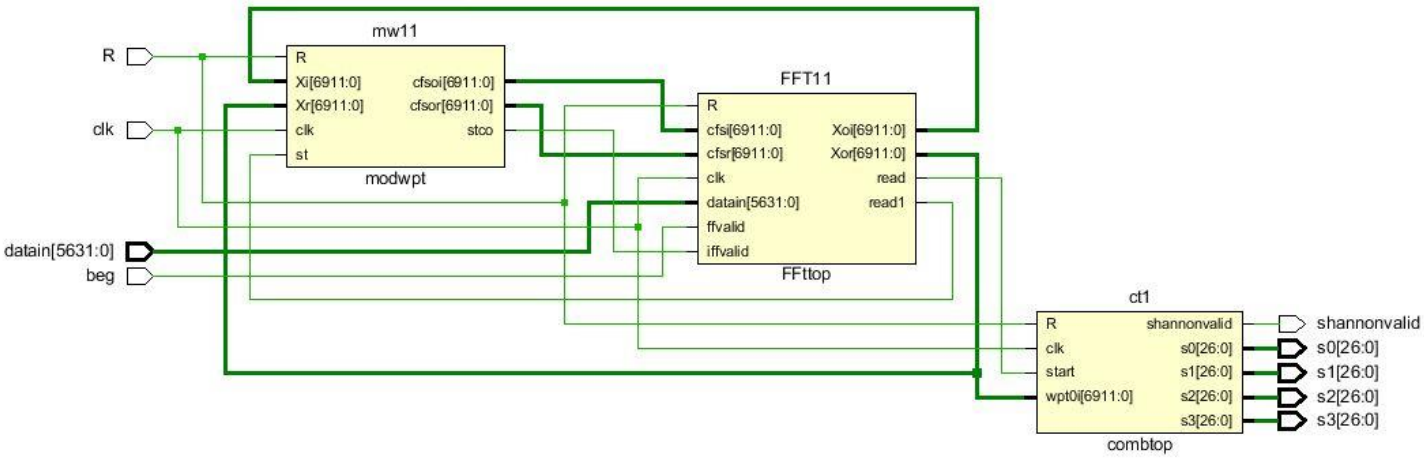


Figure 4- 18: Optimized architecture

The initial implementation was not possible to be implemented due its very large resources needed, which was not feasible to be implemented on FPGA and its utilization would pass 1000% of available resources on ZynQ board.

So, it is be optimized over many stages ending up with the following hardware architecture in Figure 4- 18.

This hardware used the fact of storing the data and use a single module rather than repeating it and loop the inputs to this module and store the output in another register to be delivered to the next module using hand-shake protocol through

signals. The final used modules are only three modules, *FFT* "which is used for both fast Fourier transform and inverse fast Fourier transform by the same hardware", *Modwpt* "for the decomposition tree generation algorithm of the output of the *FFT* and give the needed arrays serially again to the *FFT* module but this time use it as an *inverse FFT*" and final module that gets the data after inverse Fourier transform to operate the final arithmetic operations to give the desired features.

The flow of the data is as follows:

- Data entry from *detrend* module and after *resampling* to *FFT*.
- Output from *FFT* with both imaginary and real components enters to *modwpt* module which operates on them and then serially gives 4 arrays that goes back to *FFT* that operates in *iFFT* mode.
- The 4 output arrays from *iFFT* are then stored and serially given to *combtop* module that performs the entropy calculation.

The modules:

FFT

The following shown 5 modules at Figure 4- 20 and Figure 4- 19 from the initial implementation are more or less the same hardware but repeated 5 times to get all desired functionalities of both Fast Fourier transform and the inverse Fast Fourier transform "implanted 4 times" resulting in 5 repeated hardware of this module.

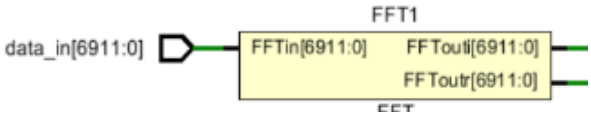


Figure 4- 19: FFT

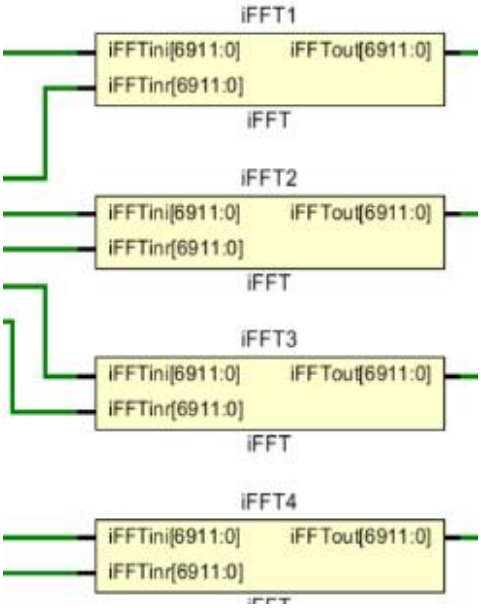


Figure 4- 20: iFFT

This module is optimized on two steps; the first one is to transform the combinational module of the FFT to a sequential module.

In this scope, the implementation of FFT depends on radix-2 DIT Cooley-Tukey algorithm which in turn is based on divide and conquer approach. As mentioned before, DFT for input dataset of length (N) is divided to two smaller DFTs (even and odd sets) each with size $(\frac{N}{2})$. Below Figure 4- 21 [1] shows divide and conquer approach.

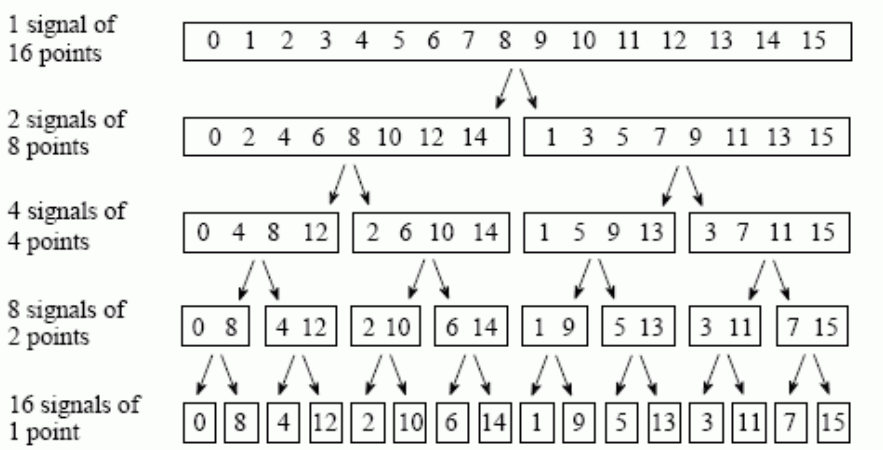


Figure 4- 21: FFT decomposition radix-2 DIT

After dividing dataset, multiplications with twiddle factors (W_N^n) are carried out on outputs of odd DFT. After that, addition and subtraction calculations are performed and result to final output dataset. FFT of size (N) needs $\frac{N}{2}\log_2 N$ complex multiplications and $N\log_2 N$ complex additions [2]. For example, for 128 point FFT,

$$Number\ of\ complex\ multiplications = 448$$

$$Number\ of\ complex\ additions = 896$$

Which are huge hardware. To solve this problem, FFT is implemented based on the resource sharing concept. This means that, in our case, one multiplier is used ($\frac{N}{2}\log_2 N$), this reduces hardware efficiently but increases time. In this project, the time is not critical but hardware and power.

FFT design features:

- 128-pint radix-2 DIT, could be either extend to larger N-points or smaller N-points.
- Forward and inverse with the same hardware.
- Resource sharing to save hardware.
- Data width is 27 bits for high accuracy.
- Implemented on FPGA Zynq UltraScale+ ZCU104 Evaluation Board (xczu7ev-ffvc1156-2-e) at different frequencies.
- Structure could be implemented using ASIC flow.

Interface

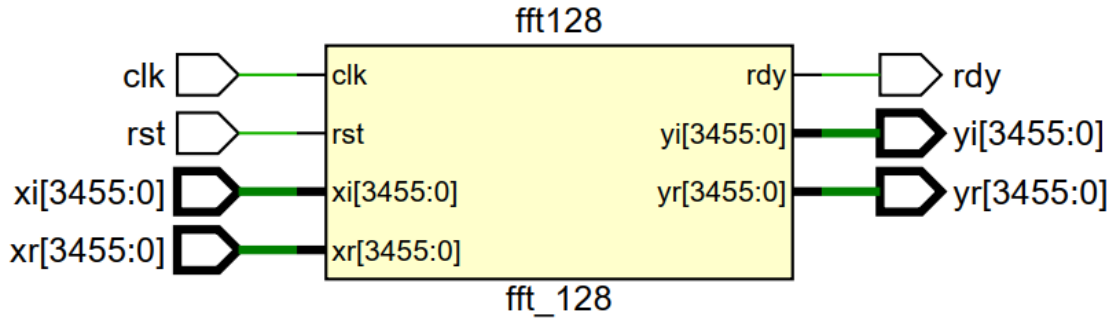


Figure 4- 22: 128-FFT core module

Table 4- 1: Signal description

Signal	**Type**	**description**
clk	input	Global clock
rst	Input	Global reset
Xr [3455:0]	Input	Real input data (27x128)
Xi [3455:0]	input	Imaginary input data (27x128)
Yr [3455:0]	output	Real output data (27x128)
Yi [3455:0]	Output	Imaginary output data (27x128)
Rdy	Output	Ready signal, outputs is ready

Architecture

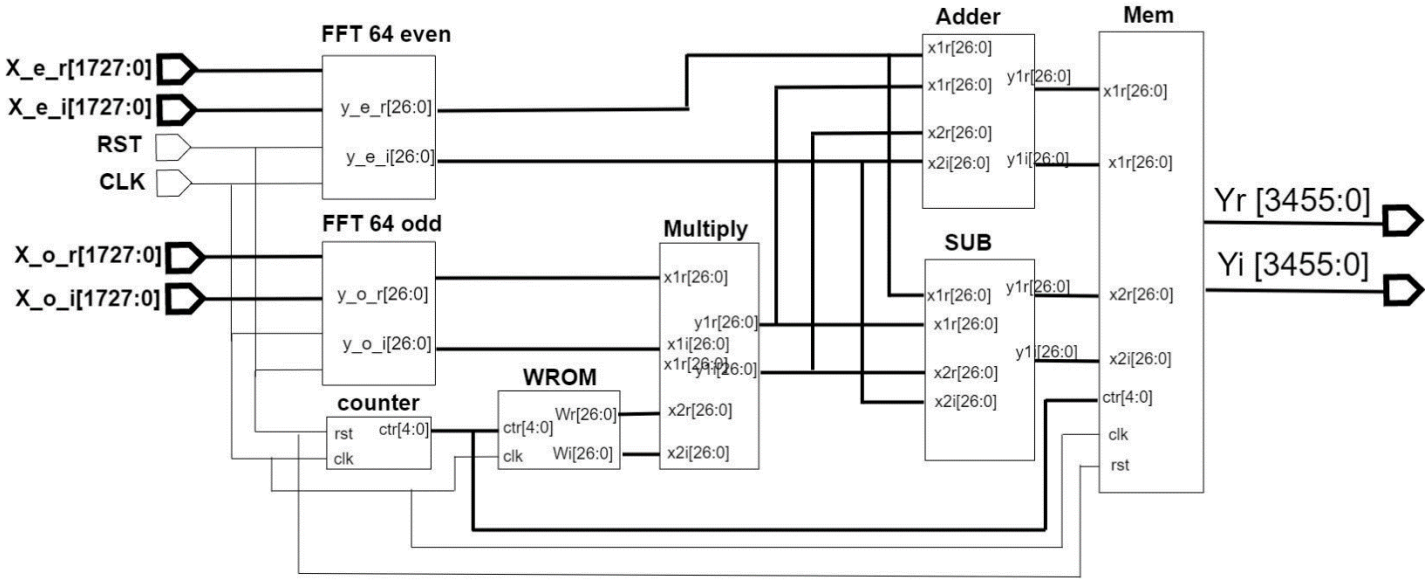


Figure 4- 23: FFT architecture

Components:

Counter

Counter module takes clock and reset as inputs. From its name, counter output (ctr) is input to WROM module and mem module.

WROM

WROM takes clock signal and counter and the output is twiddle factor. Twiddle factors are provided serially 27-bit by 27-bit to multiply module.

Multiply

Performs complex multiplication on output of odd 64 FFT with twiddle factor. The multiplier carries one multiplication per clock. The data enter the multiply module serially 27-bit by 27-bit.

Adder

Adder module takes the output of multiply module and output of even 64 FFT module and perform complex addition on them serially 27-bit by 27-bit.

Sub

Sub module takes the output of multiply module and output of even 64 FFT module and perform complex subtraction on them serially 27-bit by 27-bit.

Mem module

Memory module takes clock signal, reset and counter as input signals. It also takes outputs from adder and sub modules as inputs to store them serially with aid of counter signal. When all outputs are stored, then ready (rdy) signal is high and FFT is ready to deliver its outputs to the next module.

64-FFT

64-FFT block is the same as 128-FFT that are explained above. 64-FFT is decomposed to two 32-FFTs and each 32-FFT is decomposed to two 16-FFT and the pattern is repeated until sixty four 2-FFTs are reached. This approach is called divide and conquer.

The second step is to select between 5 inputs to the module “once for FFT and then 4 times serially for iFFT” to enter the module depending on which step of the data flow the algorithm is, if the step is the FFT the data entered is first divided by 1024 by shifting and transformed from 22 bits to 27 bits to avoid overflow results from multiplication, then the data is entered at the real input for *FFT* and the imaginary is set to zero, through the *FFTinpmem* module, and the when the output is ready from *FFT* it is passed through *FFToutmem* module to be the input for the next module. Then the *FFTinpmem* is now ready for the *iFFT* mode waiting for the ready signal from *Modwpt* module. When this signal is set to one the output from *Modwpt* is entered serially on 4 steps and stored at *FFTinpmem* and given array by array to the *FFT* module but with swapped real and imaginary arrays as discussed earlier.

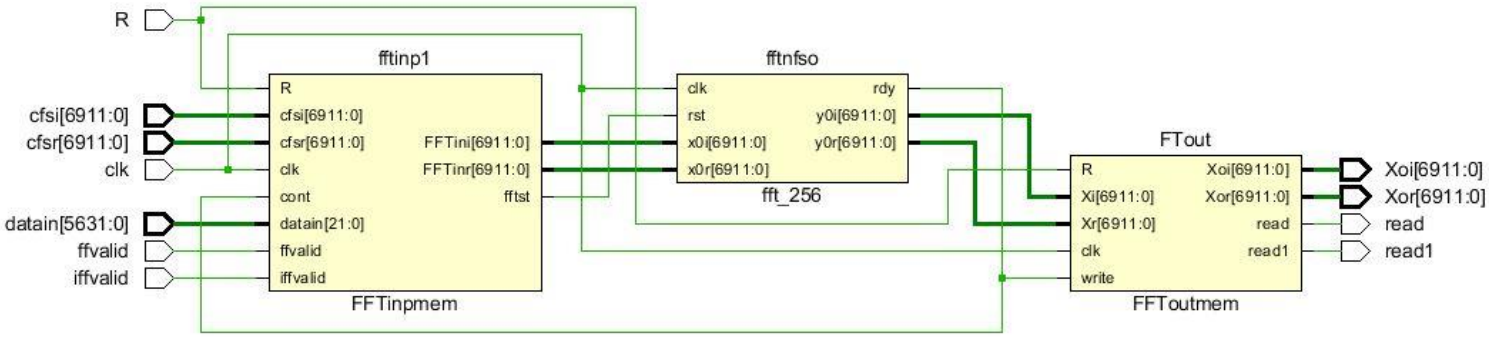


Figure 4- 24: Architecture of FFT

Then the output of the imaginary is the only taken outputs from the *FFT* is stored at *FFToutmem*, and after the four arrays are stored, they given serially to the final module for final arithmetic operations. The architecture of the *FFTtop* module is shown in Figure 4- 24.

Modwpt

This module is initially one module at top architecture, so it was optimized internally. The module consists of two sub-modules, *cfsmemory* and *modwptdec*. The first module is used to store and give data to the decompose the input array at *modwptdec* module.

The process of decomposition consists of giving an array from *cfsmemory* and decompose it at *modwptdec* then store the two resulting arrays in the memory. The decomposing process is done “as discussed earlier” by multiply each element from the array by two constants related to the algorithm order and decomposing depth and gets two arrays named *vhat* and *what* to be stored. This is repeated 15 times to finally have 30 outputs each one consists of one real and one imaginary to have a total of 60 arrays. Only the last 16 correspond to the 16 features that would be extracted.

This was optimized first by the optimizing the decomposition module, the constants to be multiplied is dependent on the step of the decomposing the system is currently at by a variable named UF. So, initially there were multipliers for each condition of the 4 conditions for multiplication. The optimization is done by

using a mux first, depending on UF to determine the constants for multiplication, furthermore the multiplication is done on four steps “once for real part and once for imaginary part of both arrays generated from decomposition”. So, the number of multipliers is approximately reduced to 1/16 of the initial implementation “1/4 as there are 4 UF conditions, and 1/4 for repeating the process and generate the two output arrays on 4 steps”. Second the *cfsmemory*, We are not in need to have the full 60 arrays stored because we will not use the whole 16 features, the features to be extracted are number *5,6,7,13* which means that last 3 real and imaginary arrays can be neglected and not implemented. Moreover, once memory content of an array is sent to the decomposition module this register can be overwritten for further optimization in flipflops. For first 2 memory locations they are filled by using the FFT output as direct input for decomposing. The following tables summarize the initial sequence of memory locations storage at initial implementation and the optimized one.

Step number	Input array 1 (decompose out 1)	Input array 2 (decompose out 2)	Output array (decompose in)
1	-	-	FFTout
2	0	1	0
3	3	2	1
4	4	5	2
5	7	6	3
7	8	9	4
8	11	10	5
9	12	13	6
10	15	14	7
11	17	16	8
12	**18**	**19**	9
13	21	**20**	10

14	22	23	11
15	25	24	12
16	26	**27**	13
17	29	28	-

Table 4 - 2: Initial sequence of cfsmemory entry

Table 4 - 2 shows the storing sequence before any optimization with needed features in bold.

Step number	Input array 1 (decompose out 1)	Input array 2 (decompose out 2)	Output array (decompose in)
1	-	-	FFTout
2	-	1	decompose out 1 from previous " "step
3	3	2	1
4	4	5	2
5	2	1	3
7	6	7	4
8	4	3	5
9	8	-	6
10	**13**	**12**	7
11	**14**	-	8
12	-	**16**	-

Table 4- 3: Optimized sequence of cfsmemory entry

Table 4- 3 shows the new sequence of data entry used to optimize in number flipflops and keep generating the needed 4 arrays generated “by generate the previous needed intermediate arrays”.

The final architecture consisting of two modules the memory and the decomposing module are shown in Figure 4- 25: Modwpt architecture.

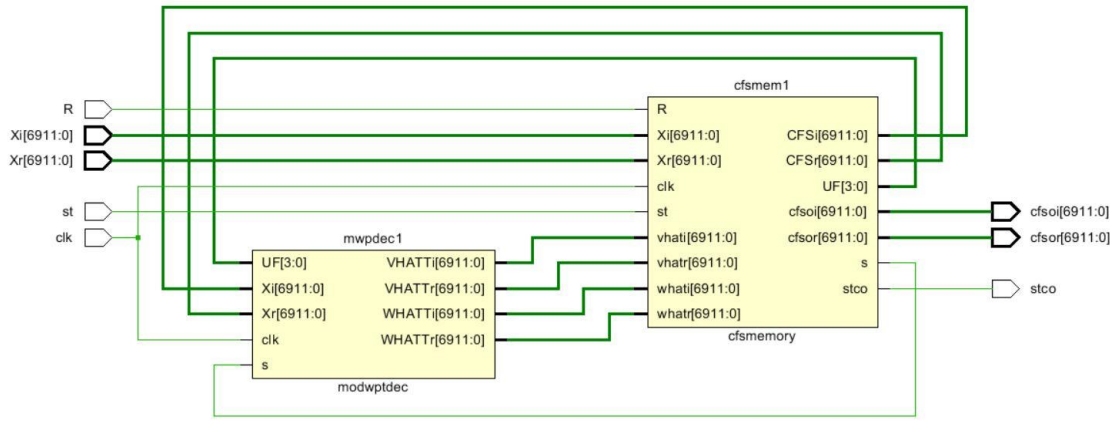


Figure 4- 25: Modwpt architecture

Combtop

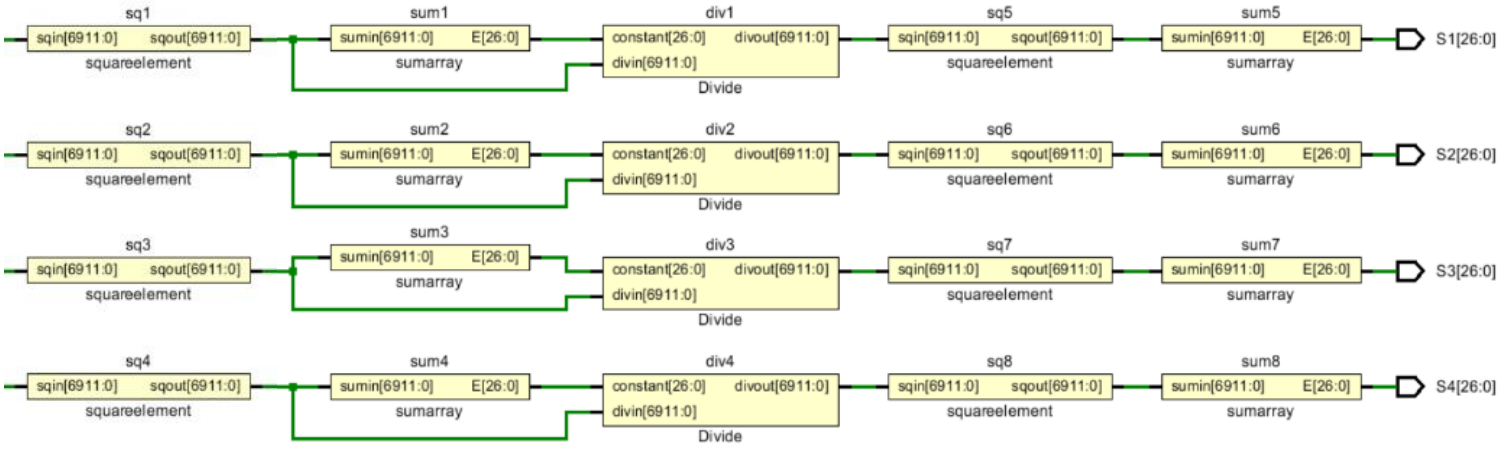


Figure 4- 26: Pre-optimization architecture of entropy calculation

The initial implementation of this part of the algorithm is shown at Figure 4- 26. This implementation has 2 modules each one is repeated twice, *squareelement* and *sumarray*. Consequently, it is obvious that one implementation only of each module would be used twice at this part of the algorithm. Furthermore, the number of dividers should be limited as each divider gets huge utilization of available LUTs on the FPGA. So *divmemory* module is added to store the initial array to be divided, and serially insert partition of the array to be divided and over-write again at *divmemory*, this process is optimized to finish after 16 clock

cycles. Finally, as the data is inserted serially from the *iFFT*, so, it needs to be stored first in a memory then serially operate on each array the first squaring, summation and division then restore it in the memory, then redo the squaring and summation to get the output features, as discussed earlier the squaring is used again for more optimization in the hardware with verification from software simulations that accuracy is not reduced. The optimized version schematic is shown in Figure 4- 27.

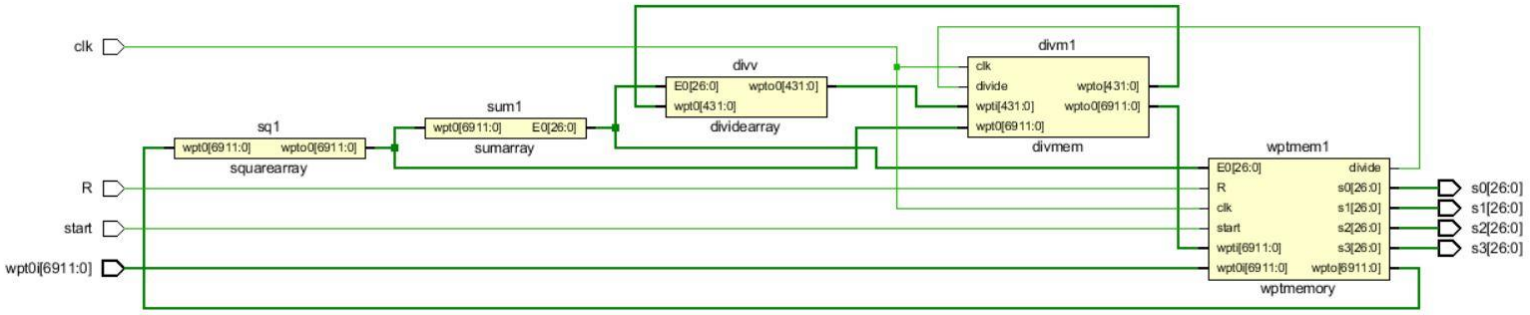


Figure 4- 27: Optimized architecture of entropy calculation

So, the data flow of the algorithm is as follows, after storing the 4 arrays serially they are given array by array to the *squareelement* to square each element of the array, and in order to reduce the error of decimal points truncation and as the input is set to be fraction the output of the *squareelement* is multiplied by 256 by shifting, then this array is delivered to both *sumarray* and *divmem* to get the summation of the array and then divide each element of the squared array by the summation, after the division is over the resulting array is stored at *wptmemry* by overwriting and the next array is set for squaring, summation and division. When the 4 arrays are done the looping starts again from first array that is overwritten and goes again for squaring and summation and the summation is then delivered to the output through *wptmemory* module with a signal of shannonvalid set to one when the 4 features are set and ready.

Utilization and power estimation of different implementations on FPGA kit.

The design of Shannon algorithm is implemented on ***Zynq UltraScale+ ZCU104 Evaluation Board (xczu7ev-ffvc1156-2-e)***. Full Shannon Entropy system is implemented for 32-point "non-optimized FFT", 64-point "optimized FFT" and 128-point "optimized FFT" -as we were restricted to "2 power" point numbers for FFT implementation- and they all are compared in terms of resource utilization and estimated power consumption, before using it in the full system to be implemented.

The following figure shows the comparison between resource utilization between the three implemented Shannon algorithms with different points number. The following 3 figures,

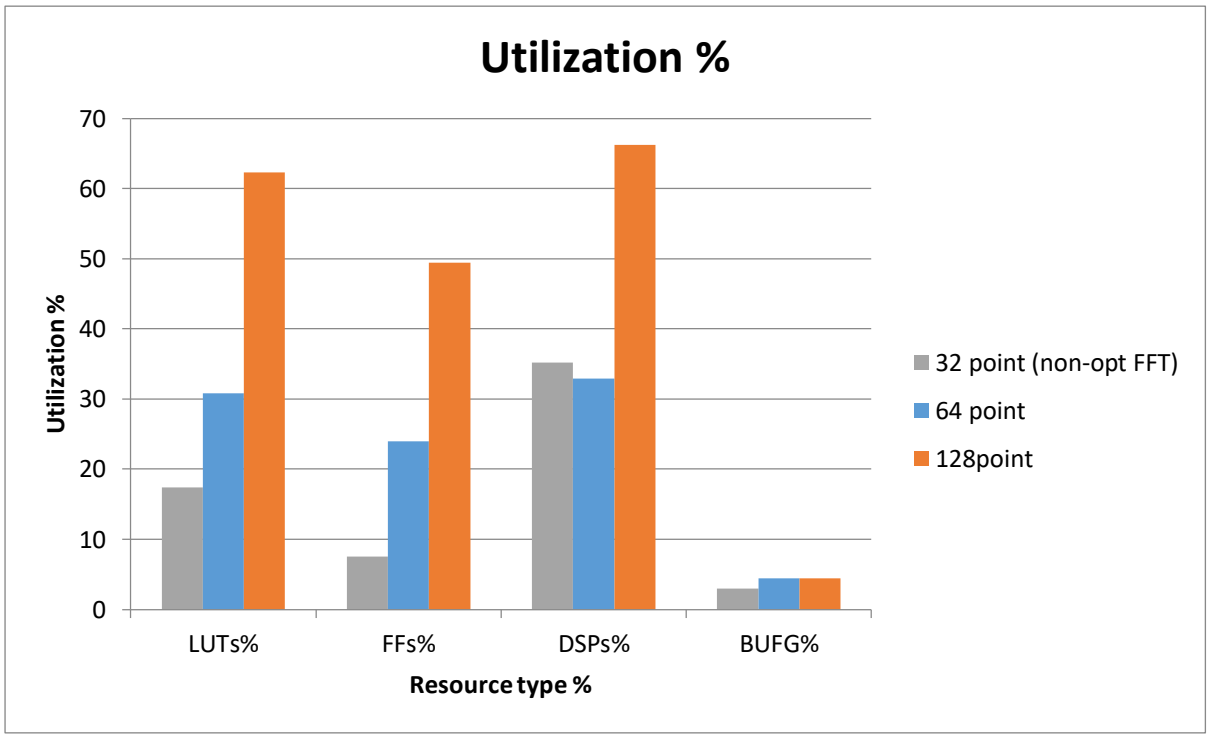


Figure 4- 28, Figure 4- 29 and

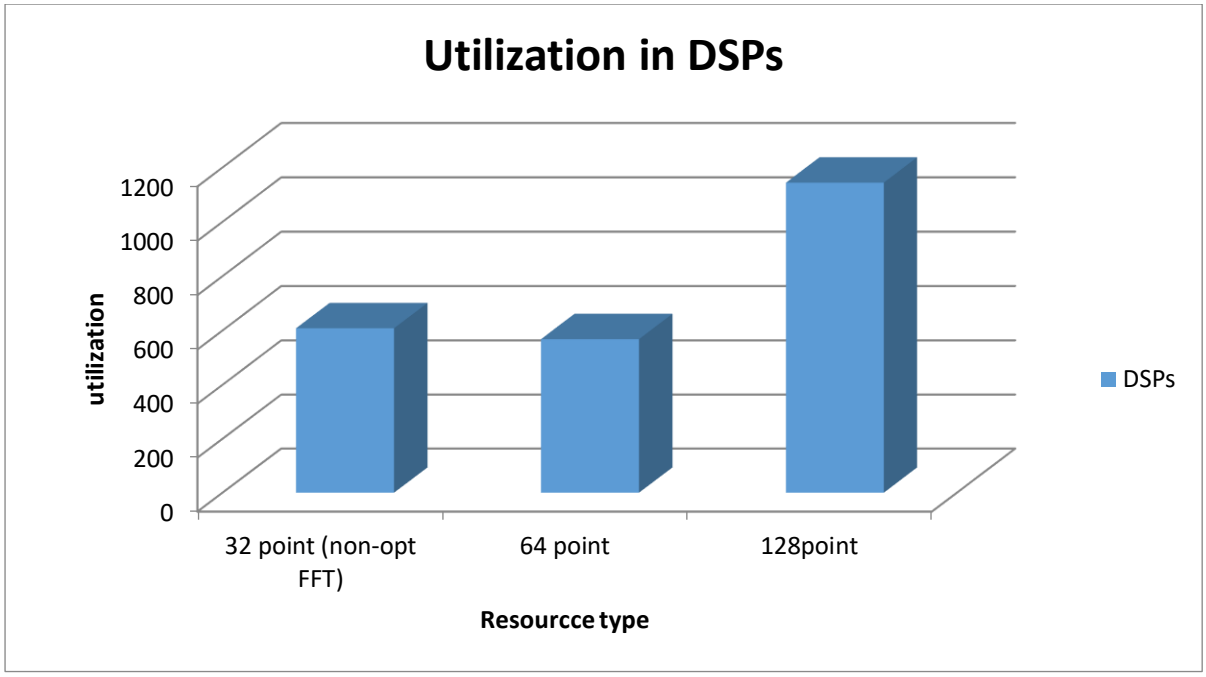


Figure 4- 30 are showing the difference in resource utilization between the three implementations.

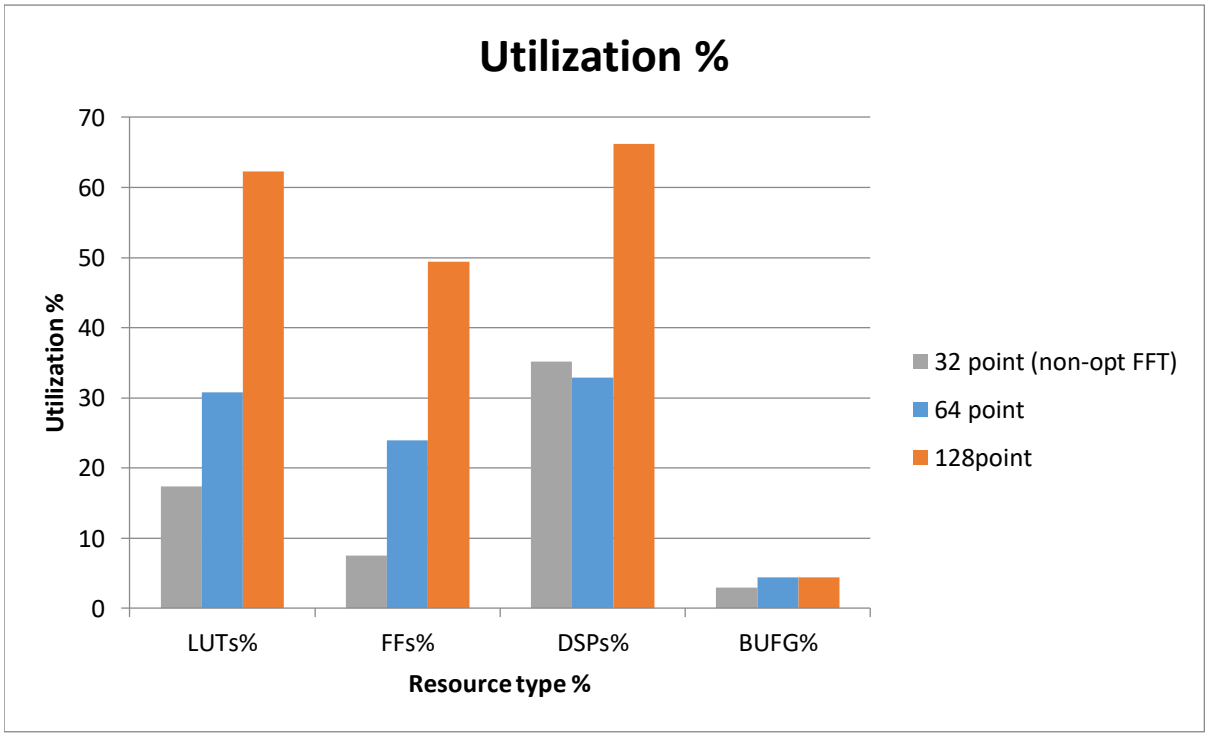


Figure 4- 28: Resource utilization % comparison between 3 different point numbers

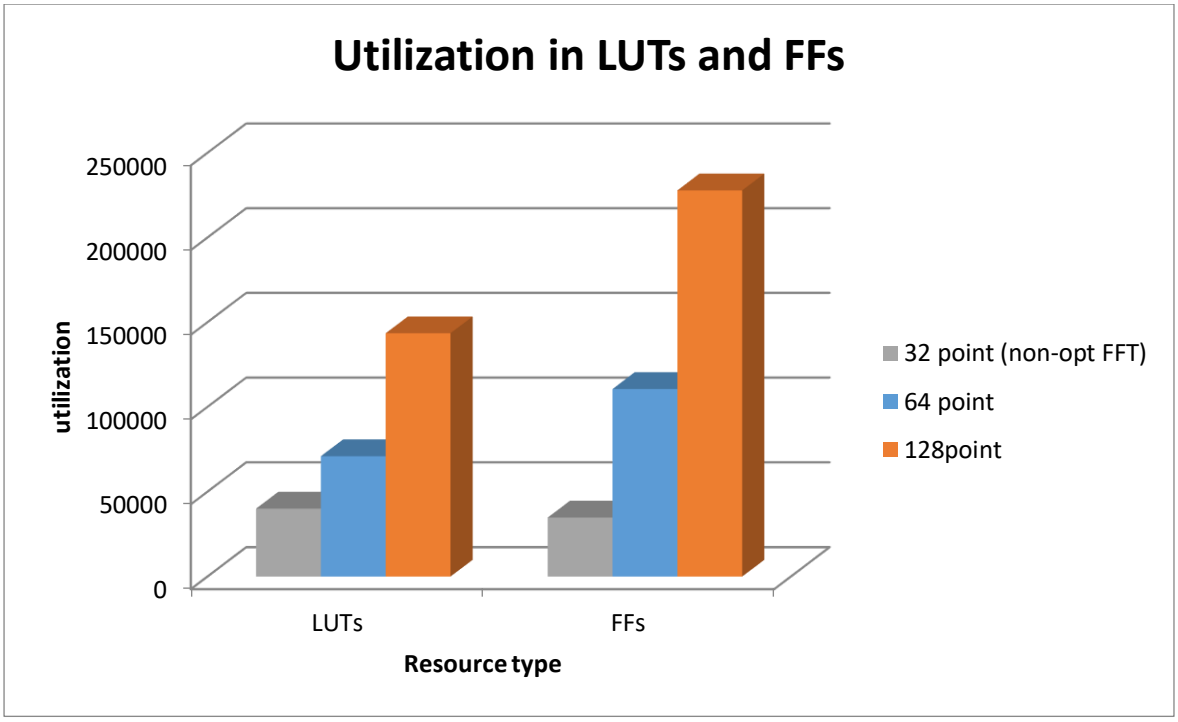


Figure 4- 29: LUTs and FFs comparison between 3 different point number

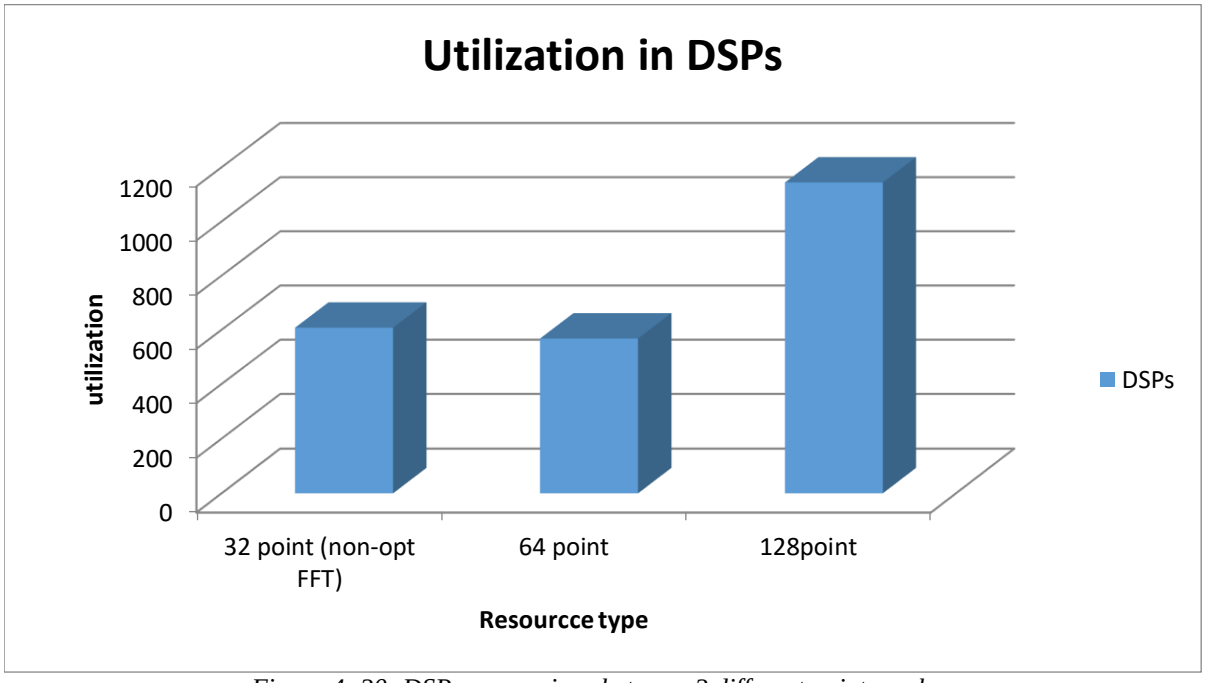


Figure 4- 30: DSPs comparison between 3 different point number

It is shown that as we are going up with number of points the resource utilization is increased linearly with number of points. Except for DSPs as the FFT used in Shannon 32 point was not optimized and has much resources used, which shows the impact of FFT optimization.

The power is estimated of the 3 designs and compared together at three different frequencies,100K, 1M and 10M; the results are summarized in Figure 4- 31, Figure 4- 32, and Figure 4- 33.

Note: if no power is present in the graph then it is estimated to be less than 0.001W.

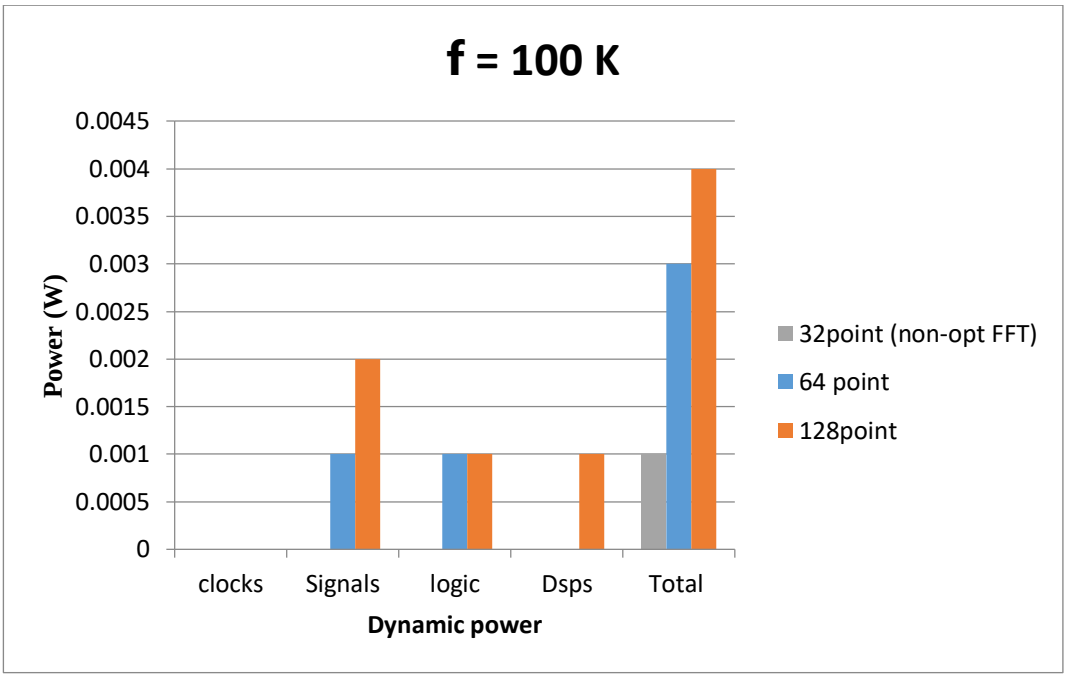


Figure 4- 31: Power comparison at 100k Hz

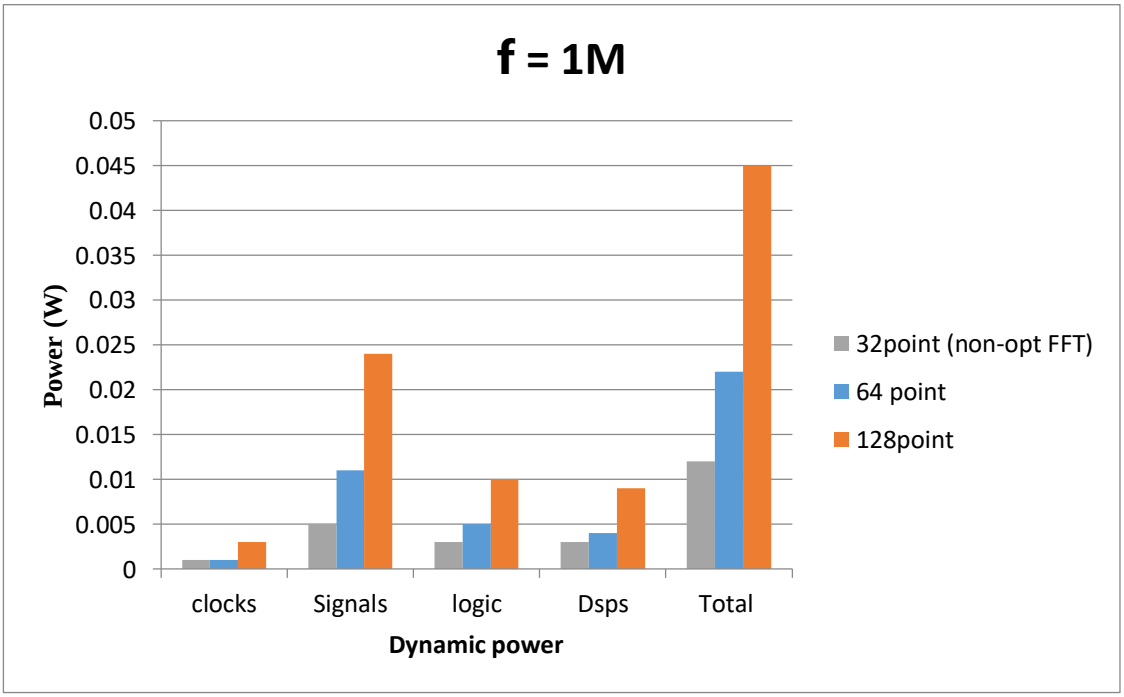


Figure 4- 32: Power comparison at 10M Hz

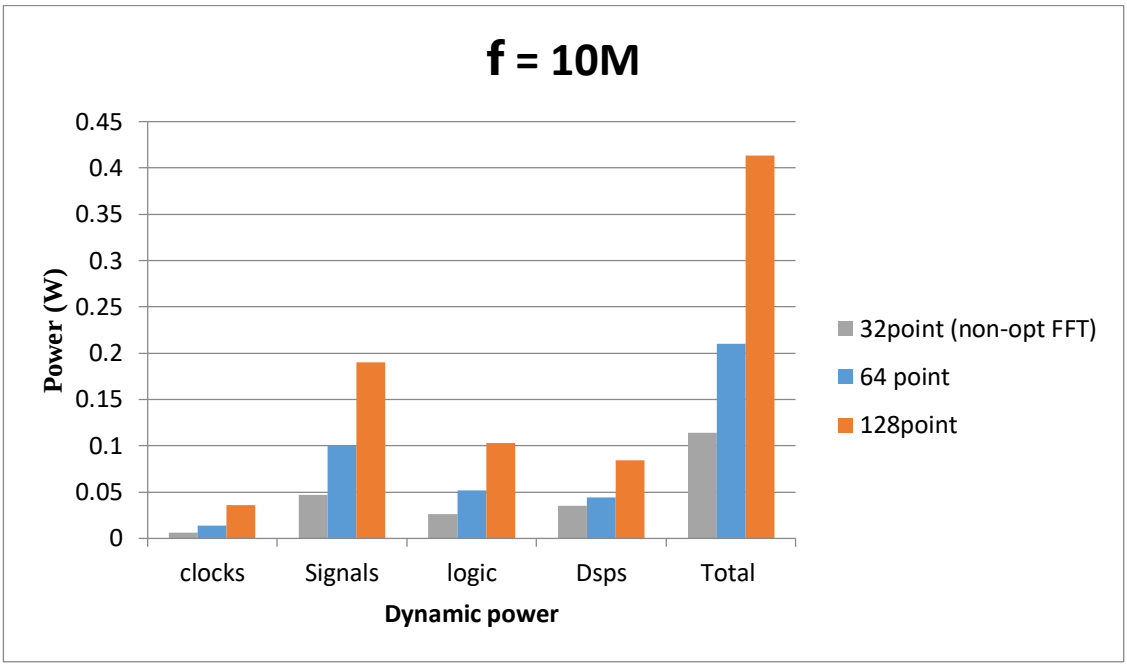


Figure 4- 33: Power comparison at 1M Hz

Final System

All modules are integrated into one single system. Three final systems are designed. First two systems are implemented on Vivado Software on Xilinx Zynq Ultrascale+ ZCU 104 Evaluation board. The last system is implemented on real Virtex 7 Board in ONE-Lab in Cairo University using a Vivado software.

First two systems:

Both systems have same architecture showed in Figure 4- 34 . The system operates on one second 360 element for detrend and memory modules. Then these 360 data elements are resampled for AR module and Entropy module. The only difference between the two systems is the number of resampled elements in Entropy module.

In the fig, data paths are represented by blue color, control signals are represented by green color. Clk, reset, new window is omitted for cleaner schematic.

All included module has been explained above except Shannon Sampling module and Feature Valid module. SE sampling module behaves like AR sampling and

assigns correct sampled data elements to Shannon module. However, it provides additional functionality as SE module needs special handling of valid signal from the memory. Valid signal is a level signal that continues to be one as long as the output of the memory is valid. SE module needs this valid signal as pulse which returns to zero after one clock cycle. SE Sampling module provide this proper handling for this control signal. Feature Valid module handles the outputs of AR and SE modules since the two outputs are not valid in the same time as AR has different throughput than SE. Feature Valid module captures the output of each module when valid and does not output them until both are valid.

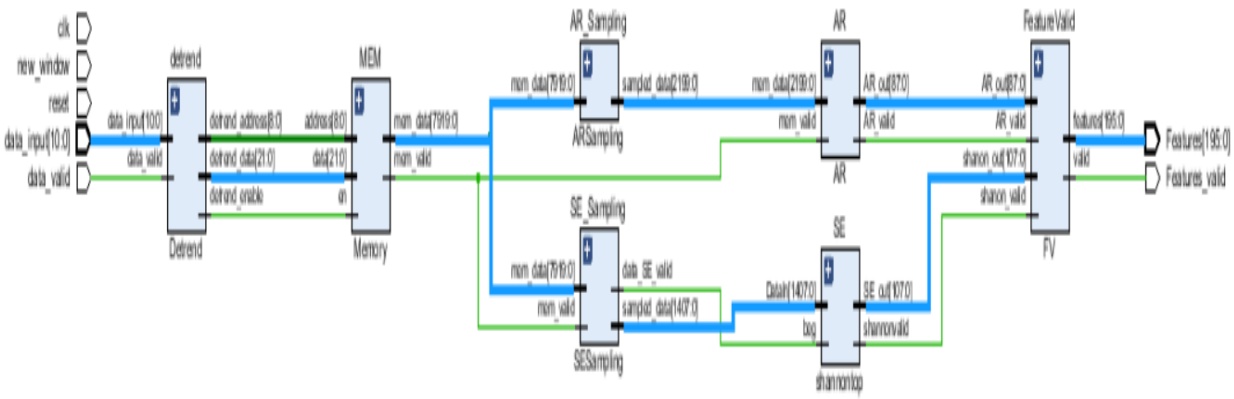


Figure 4- 34 Schematic of full system on Zynq Ultrascale+

As pointed out in the previous chapter, 128-SE module gives better accuracy than 64-SE module. We implemented the two systems to judge between the tradeoff between hardware resources and accuracy. Table 4- 4 hardware resources and accuracy of both systems. Table 4- 4 shows hardware resources and accuracy of both systems. It is clear that 128-SE consumes significant resources and gives a higher accuracy by 1%. A figure of merit is calculated to judge is it worth to implement 128-SE instead of 64-SE. The FOM is chosen such that it gives more weight to the accuracy (error is cubed) and gives less weight to Flip-flops as they have low utilization with respect of LUTs and DSPs. Both designs have nearly same dynamic power. So, it is not included in the FOM. It is clear that system 1 is better

as it has lower FOM than system 2 with nearly three times less. Thus, we have chosen this design to implement it on real FPGA.

	System 1 (SE: 64)	System 2 (SE: 128)
Resampled elements (AR)	100 data elements	100 data elements
Resampled elements (SE)	64 data elements	128 data elements
Bits for single element (AR)	22 bits	22 bits
Bits for single element (SE)	27 bits	27 bits
LUTs	112,775	187,230
DSPs	771	1347
FFs	117,404	234,712
Dynamic Power for 10 KHz	< 1mW	< 1mW
Accuracy (Mean F1 score)	91.8%	92.9%
FOM ($LUTs \cdot DSPs \cdot \sqrt{FFs} \cdot (1 - Accuracy)^3$)	16.43	43.7

Table 4- 4 hardware resources and accuracy of both systems.

Real System implementation:

As indicated above, we chose System 1 to be implemented on Virtex-7 VC709 Evaluation Platform in ONE-Lab at Cairo University. The system needed modification to be able to give it inputs and monitors the outputs. The system has huge Input/Output pins which cannot be assigned and controlled through the board. We implemented five different blocks in the system:

- Test generator module: we implemented a module that saved inside it some test vector and this module outputs the correct input to the system depending on the cycle and reset signal.
- A Buffer: that converts the system differential clock into single ended clock to our module.
- Clock divider module: to lower the frequency of the system clock.

- Virtual Input/output Intellectual Property (VIO IP): This IP gives us control over the assigned inputs and outputs which were reset, new window, data input valid, Features output and Feature valid signal.
- Integrated Logic Analyzer Intellectual Property (ILA IP): This IP produces a waveform of the assigned signals which were the output signals from AR and SE modules and the full system outputs in order to monitor them when output valid is triggered.

Figure shows the hardware resources of the implemented system with power consumption. Hardware resources are nearly the same as relatively small logic is added to the system. However, dynamic power has been increased significantly. This is due to high clock frequency of the system.

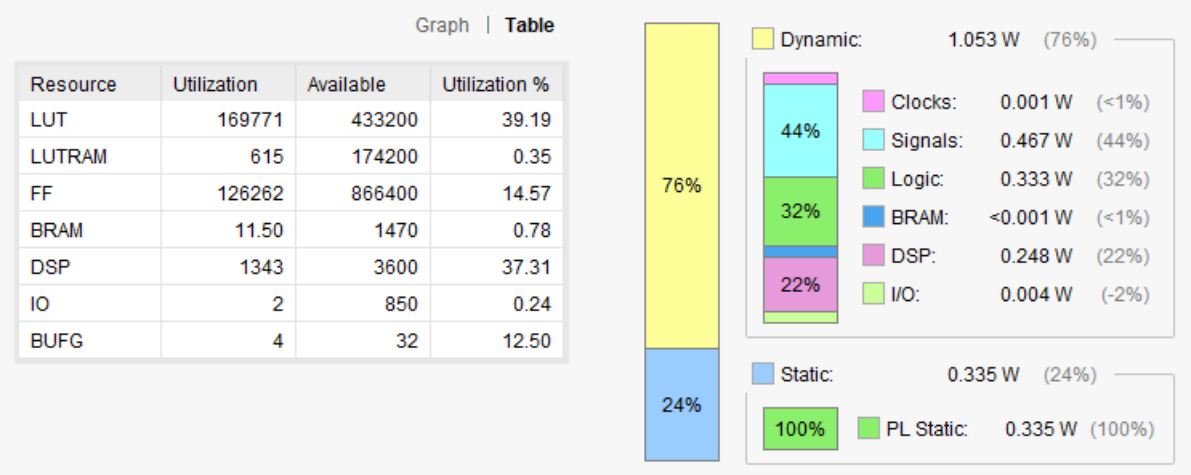


Figure 4- 35 Hardware resources and power consumption of real implemented system

References

[1] S. Smith, The scientist and engineer's guide to digital signal processing. [San Diego, Calif.]: California Technical Pub., 2002, p. Ch12.

[2] Peebles P Z. Probablity, Random Variables, and Random Signal Principles, 2nd ed. McGrawHill. NY. 1987.

Chapter 5: Artificial Database Generation Based on Modified COMSOL Multiphysics Model

Introduction

Following the context of the previous chapters, it is obvious that the main aim of this project is to design and build a complete hardware-based Arrhythmia detection system from a single lead, real-time ECG signal. For selecting the feature extraction algorithm that yields the highest accuracy, MIT-BIH Arrhythmia database [1] available on PhysioNet platform [2] has been used for training and testing.

MIT-BIH Arrhythmia database is composed of 48 half-hour records of two-channel ECG leads, from which 46 records have the record from the modified limb lead II (MLII). The records are studied by the Boston's Beth Israel Hospital (BIH) Arrhythmia Laboratory such that the first 23 records, 100 series, are selected randomly from Arrhythmia patients' recorded tapes and the rest 25 records, 200 series, are chosen to cover clinically important Arrhythmia conditions that are uncommon [3]. The database provides an annotation for each beat which is used in training the SVM classifier. Despite the fact that this is the most commonly cited database in literature [4], it has a major problem as most of the records are biased either for normal beats or abnormal beats which is not suitable for the adopted personalized training strategy [5] and leads to unreliable results and accuracies [6]. One more limitation is the duration of the records, the thirty minutes of each record is distributed on training and testing with percentages of 70% and 30% respectively. This short duration of training set leads to poor-insufficient training for some records [7].

Hence, this part of the project attempts building the first artificial ECG database based on a physical model. The main objective of this chapter is to

build an unlimited and biased-controlled ECG database which can be used in machine learning and artificial intelligence (AI) applications. To the best of our knowledge, this is the first reported attempt for generating an artificial database.

The artificial ECG is generated through modifying the cardiac electrical activity bidomain finite element model proposed in [8]. This is a simplified two dimensional model for human heart implemented on COMSOL Multiphysics® [9] which incorporates the modified FitzHugh–Nagumo (FHN) differential equations describing electrical activity of the heart [10].

Modeling of the Cardiac Electrical Activity

The simplified 2D implementation of the modified FitzHugh–Nagumo (FHN) model in COMSOL Multiphysics V5.3 [8] incorporates solving the coupled differential equation in the heart, which is composed mainly of atria, ventricles, septum, bundles, Purkinje fibers, and heart chamber cavities, and simulating the propagation of electrical signals through the torso and lungs as indicated in Figure 5- 1. We have modified the geometry proposed in [8] through defining a boundary

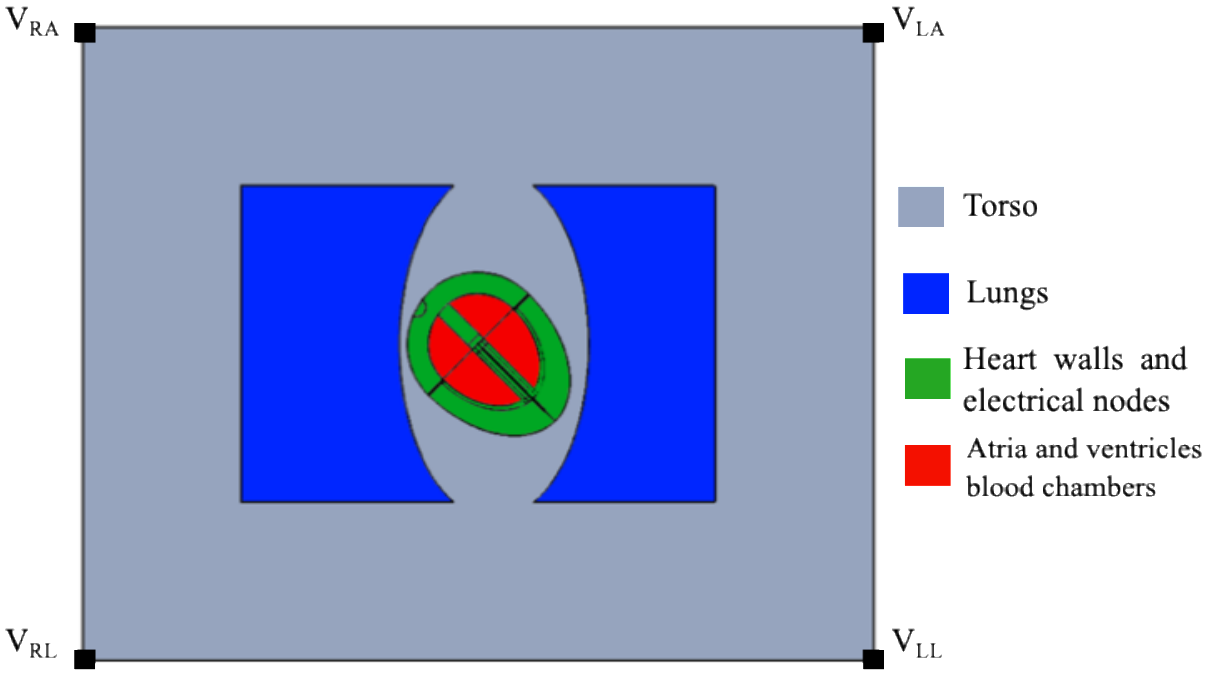


Figure 5- 1: The geometry of the 2D model used in the simulation and position of electrodes used to generate lead II ECG signal.

between both right and left ventricles to enable studying medical conditions as bundle branch block (BBB) [11], the model geometry is shown in Figure 5- 2.

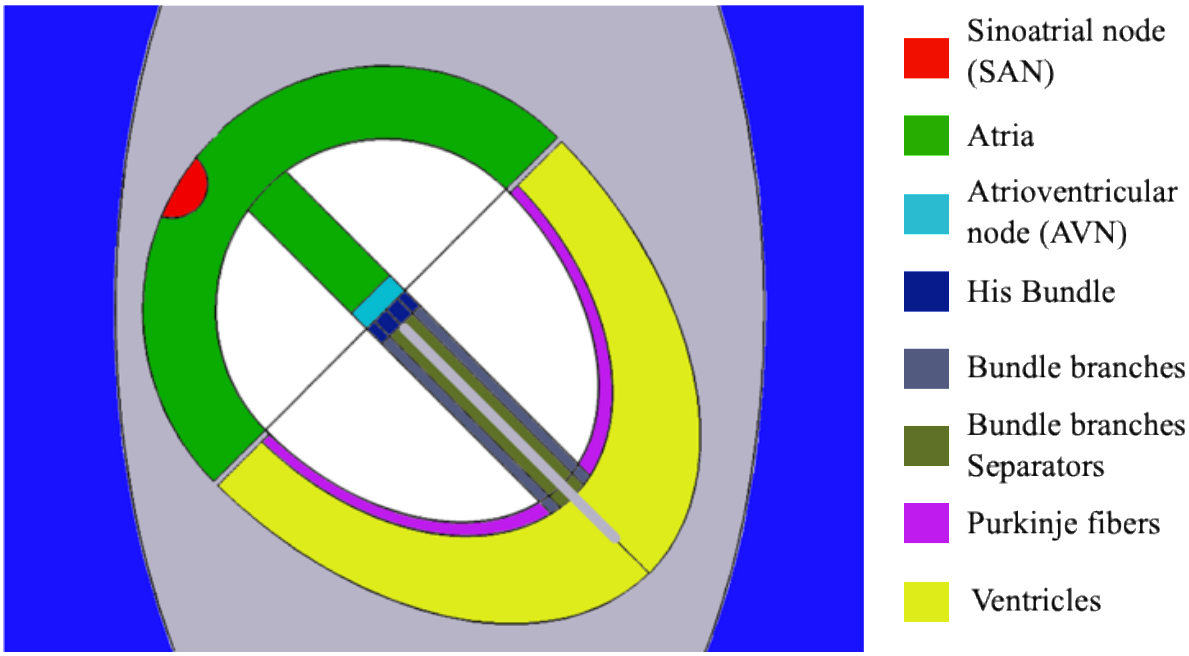


Figure 5- 2 Heart simplified 2D structure simulated.

The modified FHN equations [10] describe the electrical activities and their propagation across heart wall tissues. This modified version of the FHN model has the advantage of simple control of the refractory period and action potential duration with a single numerical parameter (e). The electrical activity of any tissue on the heart walls (myocardial tissue) is defined by its intracellular potential (V_i), extracellular potential (V_e), and a periodic excitation variable (u). Hence, the equations governing the potential development in each region in hearts' walls are defined according to [8]:

$$\frac{\partial V_e}{\partial t} - \frac{\partial V_i}{\partial t} + \nabla(-\sigma_e \nabla V_e) = i_{ion} \quad (1)$$

$$\frac{\partial V_i}{\partial t} - \frac{\partial V_e}{\partial t} + \nabla(-\sigma_i \nabla V_i) = -i_{ion} \quad (2)$$

$$\frac{\partial u}{\partial t} = ke\left[\frac{(V_m - B)}{A} - du - b\right] \quad (3)$$

$$i_{ion} = kc_1(V_m - B)\left[a - \frac{(V_m - B)}{A}\right]\left[1 - \frac{(V_m - B)}{A}\right] + kc_2u \quad (4)$$

$$i_{ion} = kc_1(V_m - B)\left[a - \frac{(V_m - B)}{A}\right]\left[1 - \frac{(V_m - B)}{A}\right] + kc_2u(V_m - B) \quad (5)$$

where a, b, c_1, c_2, d, e, k, A, B are mathematical parameters, σ_i *and* σ_e are the intracellular and extracellular conductivities of myocardial tissues, $V_m = V_i - V_e$, and i_{ion} is the ionic current within cell membranes and is defined within the sinoatrial node (SAN) according to Eqn. (4) and within all other regions according to Eqn. (5). The values of these parameters differ from one region to another and the objective of this study is to tailor these parameters, based on physical behavioral, in order to generate the ECG of the medical conditions presented in the MIT-BIH arrhythmia database.

In all other domains rather than the heart walls, the extracellular voltage (V) is defined in the lungs, torso and blood chambers according to:

$$\nabla(-\sigma_b \nabla V) = 0 \quad (6)$$

Where σ_b is the electrical conductivity of the domain at which the equation is solved. At boundaries of the myocardial walls tissues, the voltage (V) is set to the extracellular voltage within the heart (V_e). The torso external boundaries are electrically insulated; hence outward current equals to zero.

Flux boundary condition is used to define the inward and outward flux across all boundaries. The interior boundaries, heart boundaries in contact with blood chambers and the torso, have a zero outward flux for the intercellular voltage (V_i); thus, $\mathbf{n}\cdot\boldsymbol{\Gamma} = 0$ where n is the outward vector normal to the boundary and Γ is the outward flux which is defined according to $\boldsymbol{\Gamma} = -\boldsymbol{\sigma_i}\ \frac{\partial \boldsymbol{V_i}}{\partial \boldsymbol{n}}$. For the extracellular voltage (V_e), the flux is continuous at these boundaries; thus, the outward current density (J) equates the inward flux which can be written as $\mathbf{n}\cdot\mathbf{J} = -\boldsymbol{\sigma_e}\ \frac{\partial \boldsymbol{V_e}}{\partial \boldsymbol{n}}$. This

model is already implemented on COMSOL Multiphysics [8]; however, it has been tailored according to our modified geometry.

This model is capable of generating the ECG signal of the signal-averaged ECG (SAECG) and the standard 12-lead system through monitoring the potential at different points in the model; however, we have focused our analysis on the Einthoven lead (II) which is calculated from the points indicated in Figure 5- 1 according to:

$$V_{II} = V_{LF} - V_{RA} \;\&\; V_{RL} = 0 \;\; (grounded) \tag{7}$$

However, the 2D model is capable only of generating an in-plane projection of the ECG signal. The 3D model can be used to study the topological effects and out-of-plane ECG signal [12,13]; however, it is beyond the aim of this project and adds unnecessary computational burden.

Methodology

Table 5.1 indicates a summary of the medical conditions available in the database. As indicated, the most repeated beats are for normal healthy condition, left bundle branch block (LBBB), and right bundle branch block (RBBB). For verifying the proposed hypothesis, the model is used to generate ECG signals that are correlated to these three conditions. The normal beats start with the depolarization of the sinoatrial node (SAN) which spreads in the atrial muscles (Corresponds to P-wave), then the electrical signal is delayed in the atrioventricular node (AVN) which later triggers the depolarization of the ventricles (Corresponds to QRS-complex) while the atria are being repolarized, eventually the ventricles are repolarized (Corresponds to T-wave) as illustrated in Figure 5- 3 . Normally, the electrical signal reaches both ventricles simultaneously as the left and right pathways, bundle branches and Purkinje fibers, have the same electrical and ionic conductivities. In left bundle branch block, the left pathway is blocked and the

right ventricle depolarizes firstly and then the depolarization signal transfers from the left ventricle to the right ventricle through conduction of heart muscles. In the same way, the right ventricle induces the repolarization of the left ventricle. Similarly, right bundle branch block involves a delayed depolarization and repolarization of the right ventricle that is induced by the left ventricle due to blockage of the right bundle or/and right Purkinje fibers [11].

Table 5- 1 SUMMARY OF BEATS REPORTED IN MIT-BIH DATBASE IN DESCENDING ORDER[1,2]

Beat Condition	Number of beats
Normal beat	75052
Left bundle branch block beat	8075
Right bundle branch block beat	7259
Premature ventricular contraction	7130
Paced beat	7028
Atrial premature beat	2546
Fusion of paced and normal beat	982
Fusion of ventricular and normal beat	803
Ventricular flutter wave	472
Nodal (junctional) escape beat	229
Non-conducted P-wave (blocked APB)	193
Aberrated atrial premature beat	150
Ventricular escape beat	106
Nodal (junctional) premature beat	83
Unclassifiable beat	33
Atrial escape beat	16
Supraventricular premature beat	2

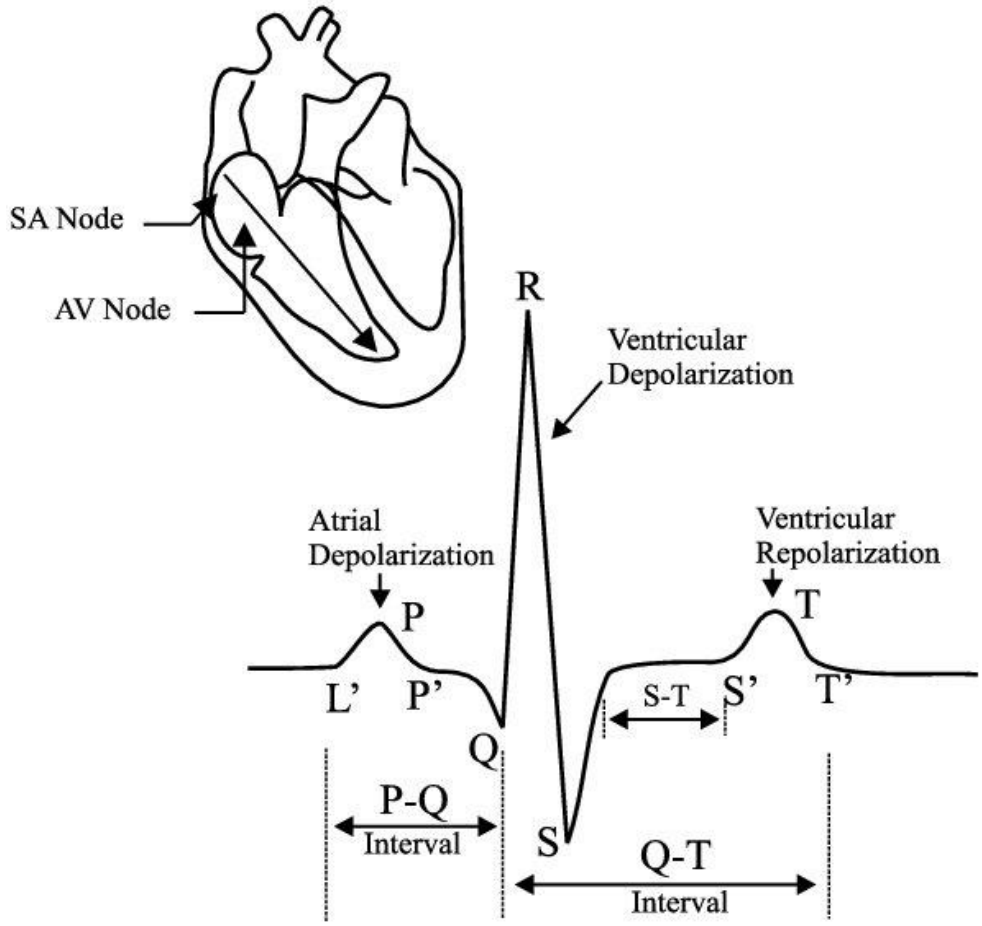


Figure 5- 3 The ECG curve and the physical phenomenon corresponding to its peaks [11].

The capability of generating artificial ECG database is verified through attempting generating ECG similar to that available in the database. Based on Table 5- 1, We aimed to generate ECG records that are similar to the available records of patients 212 and 111 [1]. Patient 212 is a 32 years female suffering from the right bundle branch block (RBBB). Its record is composed of 1825 RBBB beats (66.5%) and 923 normal beats (33.5%), it is less biased compared to the other records suffering from RBBB. Patient 111 is a 47 female suffering from the left bundle branch block (LBBB). Its record contains 2123 LBBB beats (99.9%); however, it does not include any normal beats which hinders the usage of this record as a training set. Actually, the MIT-BIH arrhythmia database does not include any record that contains both normal beats and LBBB beats; hence, these LBBB records are not suitable for the training and classification strategy we used. This highlights and emphasizes the significance of this part as we could

easily generate ECG records with both normal and LBBB beats which can be used as a training set.

The model parameters used in [8] are used as a starting and minor modifications are done to generate ECG that fits both records of patients 212 and 111. The variations in database records can be interpreted by the difference in age, heart size, placement position of electrodes, the physical status of the patient ... etc. The Branch bundle blockage condition is induced through reducing the ionic current (i_{ion}), through reducing the parameter k, and the extracellular conductivity (σ_e) in the corresponding bundle, the region between the bundle and the ventricles separator, and the Purkinje fiber on the blocked side. Moreover, the model is used to generate the transmembrane potential distribution in the heart the shows the depolarization and repolarization sequence as indicated in the results section, it also verifies the reduction of the ionic current and extracellular conductivity to cause the bundle branch block (BBB) condition.

For measuring the similarities between the generated ECG and the MIT-BIH records 212 and 111, the autocorrelation coefficient for a single beat is calculated, the beat is chosen randomly from the record. The autocorrelation coefficient evaluates the linear dependence of the two signals independent of their phase difference [14]. Additionally, the cross-correlation coefficient for 5 successive beats randomly chosen from the record and 5 beats generated from the model is calculated. The cross-correlation coefficient considers the possibility of having a phase lag between the 2 signals [14], it is a measure of the correlation between the ECG signal from the database and a lagged version of the generated ECG.
For better visualization, the time-domain features of the ECG signal ,for both the database and generated signal, are extracted using the Multilevel Teager Energy Operator (MTEO) algorithm [15,16] and their distribution is plotted as shown in the results and discussion section. However, the database signal is first subjected

to a baseline wandering filter to remove the noise modulating the ECG signal as shown in Figure 5- 4[17,18].

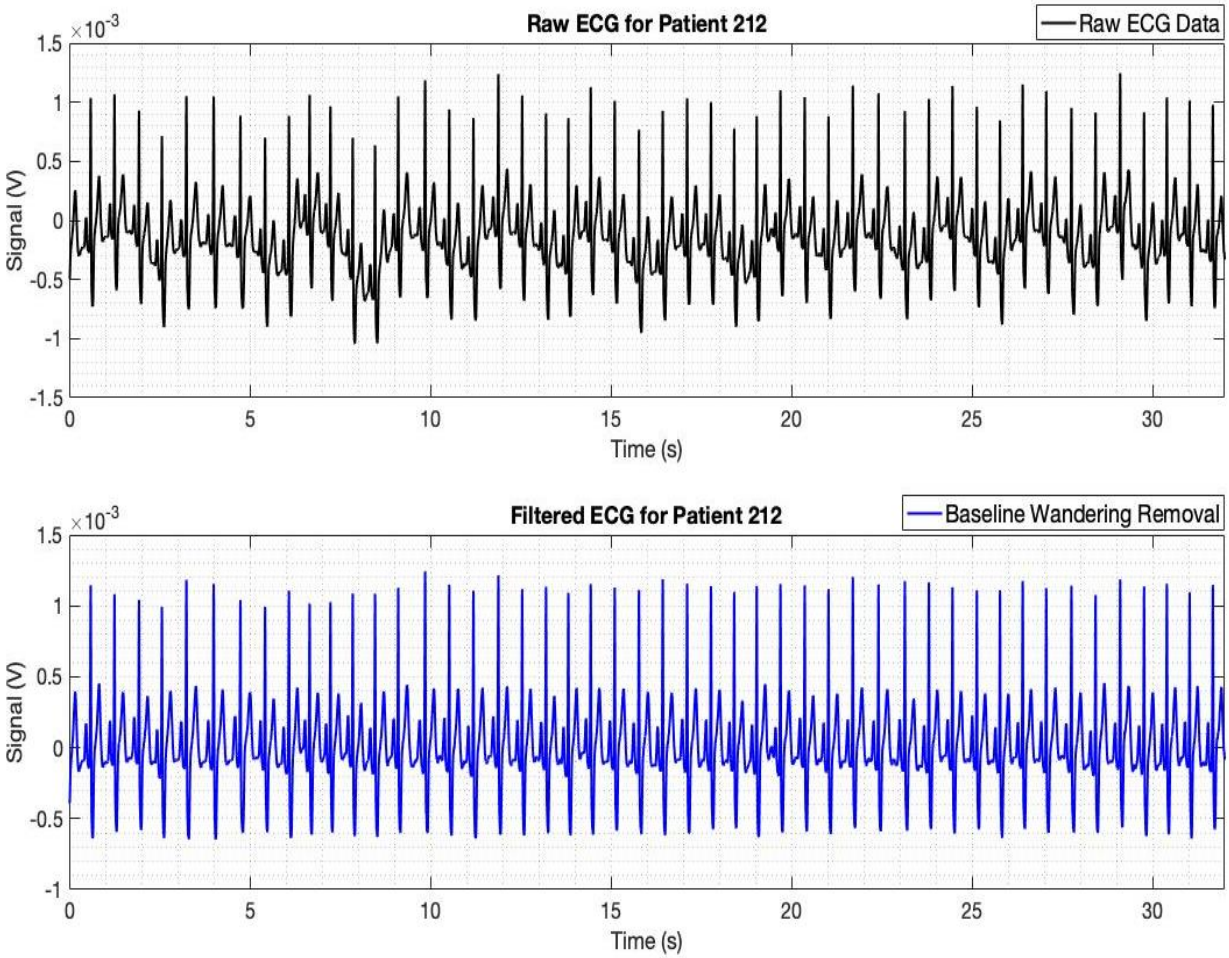


Figure 5- 4 The impact of the Multilevel Teager Energy Operator for baseline wandering removal, black curve indicates the raw ECG data from the database for patient 212 and the blue curve indicates the same sampled data after baseline wandering removal.

Results and Discussion

As discussed earlier, the correlation between the generated ECG signals and MIT-BIH database, records 212and 111, is evaluated and tabulated in Table 5.2. We have set our satisfactory criterion as a 0.7 or higher correlation coefficient between the two compared signals. The single beat correlation coefficient is a measure of the similarities between the ECG beats generated from COMSOL and the patient's ECG data, it focuses on the peak values, PQRST peaks, rather than the intervals between these peaks. On the other hand, the five beats cross-correlation is a measure of the similarities between the two records as a whole, it focuses on both the peak values, the intervals between these peaks and the general beat rate. Hence, the values calculated for the cross-correlation coefficients are lower than that of the autocorrelation coefficient due to considering the error in intervals between these ECG peaks. Figure 5- 5 shows a single beat randomly selected from the database and COMSOL generated ECG signal for the same condition, these signals are used for calculating the single beat autocorrelation coefficient. Figure 5- 6, Figure 5- 7 and Figure 5- 8 show five beats selected from the database and the correlated COMSOL generated ECG signal for the same condition. As indicated in Figure 5- 6, Figure 5- 7, Figure 5- 8 and, *Table 5- 2* the

Table 5- 2 CORRELATION COEEFICIENTS BETWEEN MIT_BIH DATABASE RECORDS (212, 111) AND COMSOL GENERATED ECG

Condition:	Normal (212)	RBBB (212)	LBBB (111)
Single beat autocorrelation	0.8824	0.8724	0.8507
Five beats cross-correlation	0.7152	0.7992	0.7531

produced ECG signal from COMSOL is highly correlated to that of the database for the three conditions simulated.

Moreover, the model is used to generate ECG records that contain both normal and abnormal beats through varying the ionic current parameter and the extracellular conductivity as discussed in the previous section.

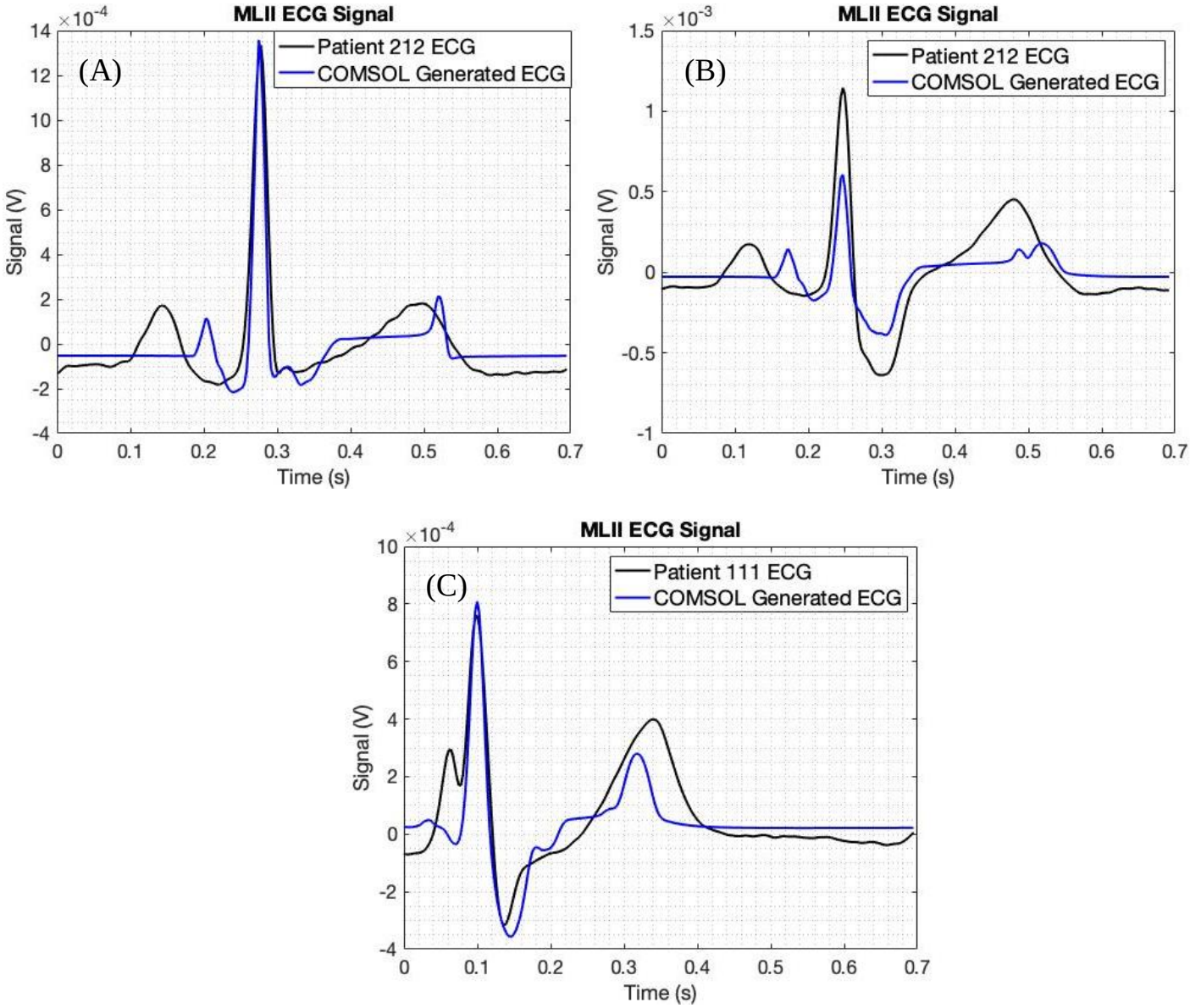


Figure 5- 5 MIT-BIH database, records 212 and 111, ECG signal and COMSOL generated ECG signal. The signals plotted are used for calculating the autocorrelation coefficient where (A) shows patient 212 normal beat, (B) shows patient 212 RBBB beat, and (C) shows patients 111 LBBB beat.

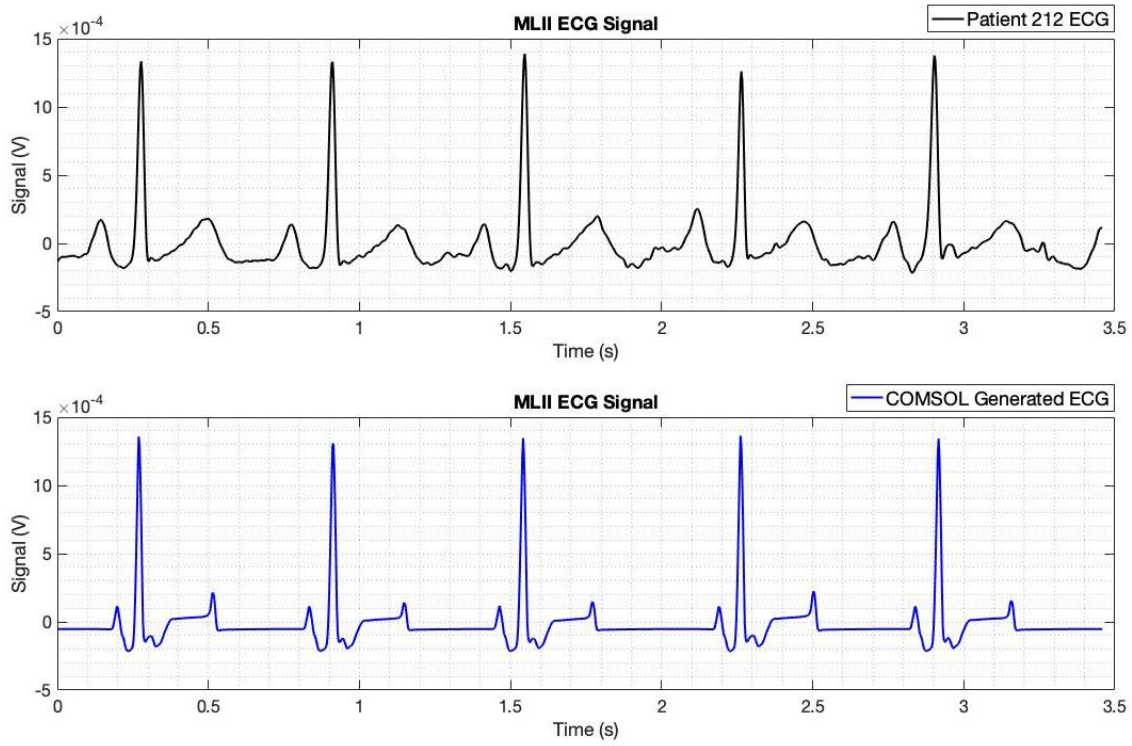


Figure 5- 6 Normal beats sequence from record 212 and corresponding produced ECG signal from COMSOL.

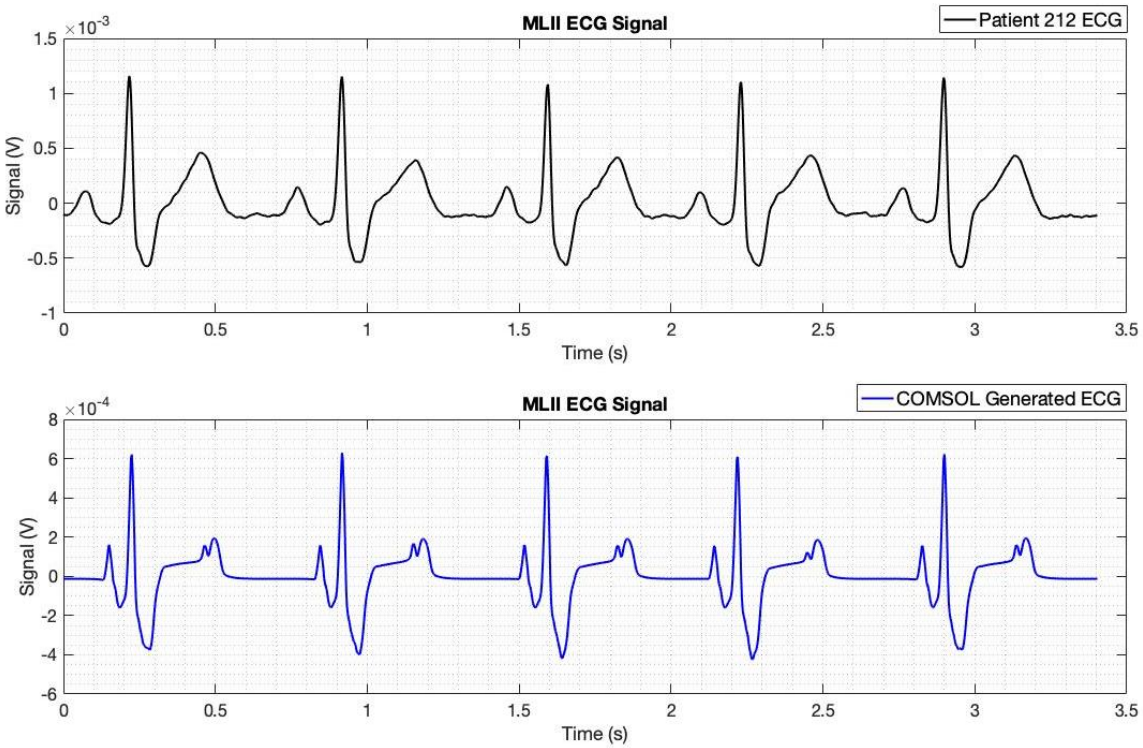


Figure 5- 7 RBBB beats sequence from record 212 and corresponding produced ECG signal from COMSOL.

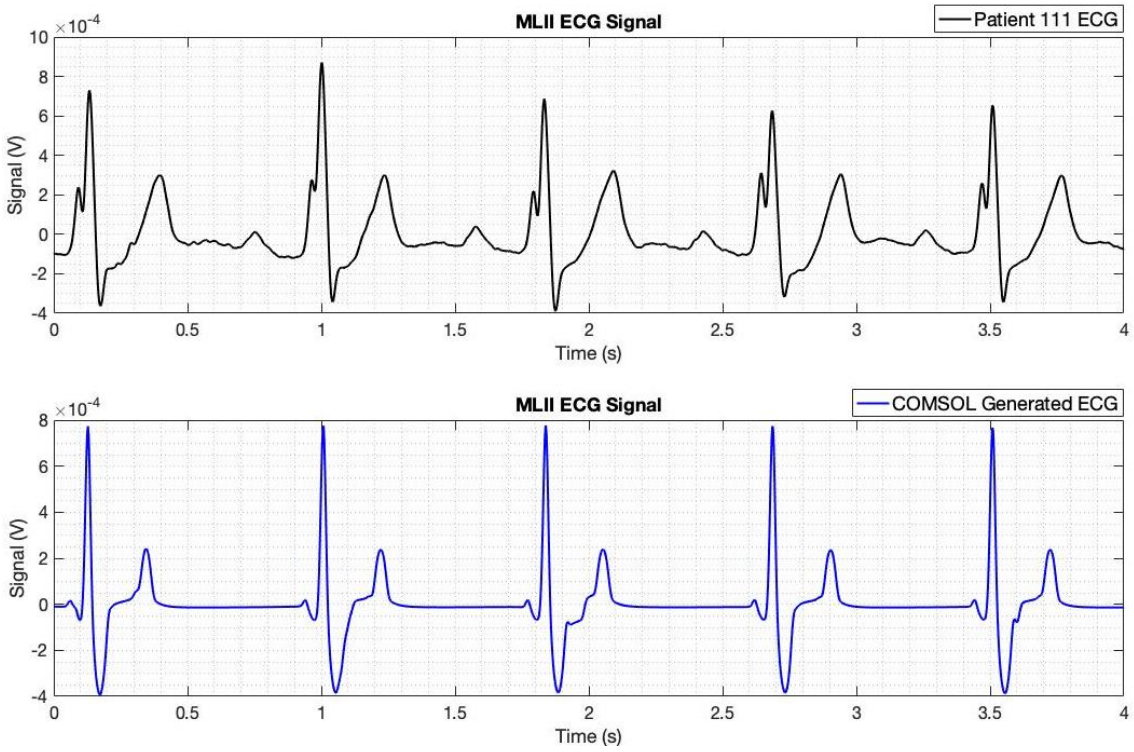


Figure 5- 8 LBBB beats sequence from record 111 and corresponding produced ECG signal from COMSOL.

One merit of using COMSOL model is visualizing the transmembrane potential and the activation sequence in the cardiac domain. The transmembrane potential is another way of verifying the condition that yields these ECG signals. Fig 5.9 shows the transmembrane potential and how it forms the ECG signal for a normal beat, both sides of the atria and ventricles are depolarized and repolarized simultaneously. Fig 5.10 shows the transmembrane potential for a RBBB beat. Firstly, the left ventricle is being depolarized and then it induces the depolarization of the right ventricle through the heart ventricles walls. Similarly, the left ventricle is repolarized and then it induces the repolarization of the right ventricle; this is the typical cardiac electrical activity of a RBBB patient [11]. Analogously, the transmembrane potential for a LBBB beat is shown in Fig 5.11, the depolarization and repolarization start in the right ventricle, then the electrical signal propagates to the left ventricle. Fig 5.9, 5.10 are based on the geometry used in generating the ECG signal that is correlated to the record of patient 212, and Fig 5.11 has different geometry as it is used in generating the ECG that is correlated to the record of patient 111. The difference in geometry and parameters

is related to the difference in age, physical status, severity of the condition and many other factors.

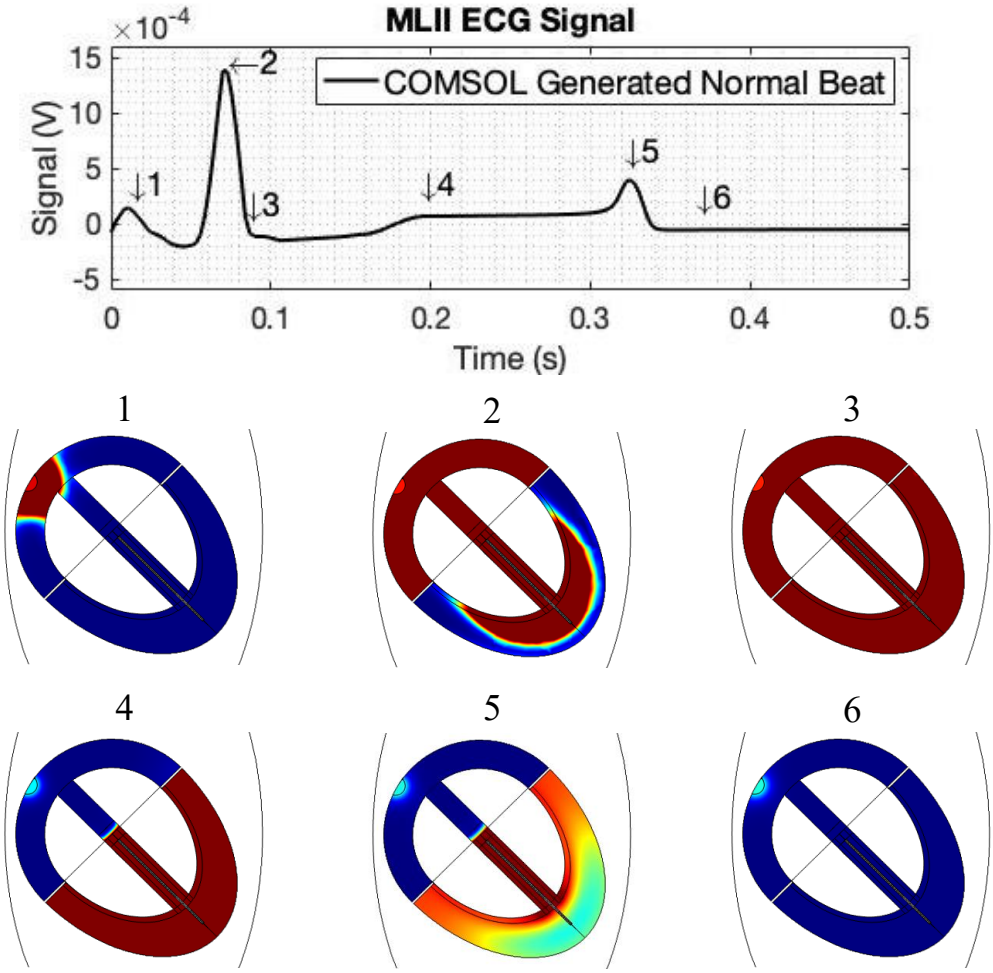


Figure 5- 9 Transmembrane potential and corresponding points on the ECG for a normal beat.

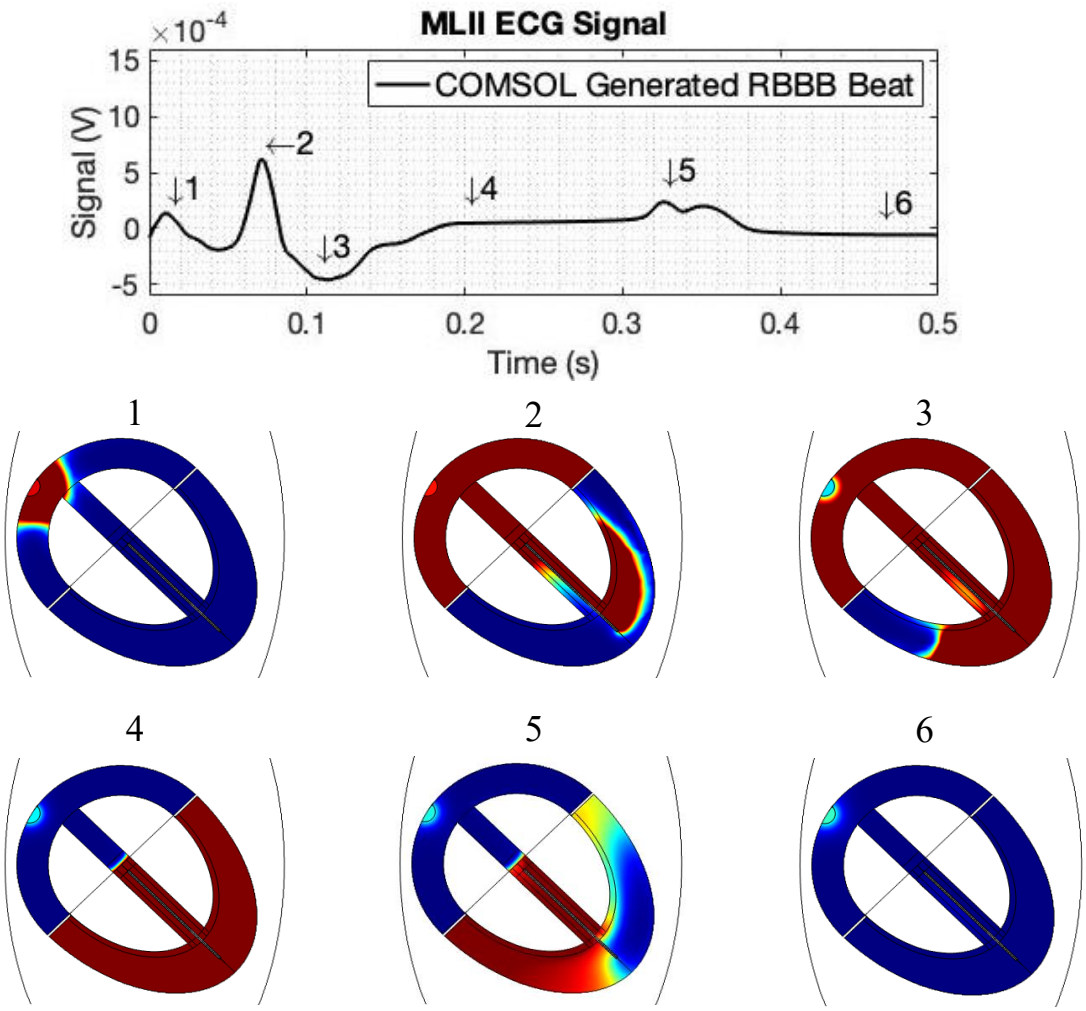


Figure 5- 10 Transmembrane potential and corresponding points on the ECG for a RBBB beat.

As discussed earlier, the left bundle branch block (LBBB) has the second highest presence in the MIT-BIH database in terms of the number of beats; however, the database does not include any record that contains both LBBB and normal beats. Thus, all records containing LBBB beats cannot be used for training or testing the SVM classifier. Nevertheless, using the same geometry and model used to produce the ECG signal that has been correlated to the record 111, it is possible to predict the patient's normal beat morphology as shown in Figure 5- 12. Based on this, a record that contains both normal and LBBB beats can be generated from the model which complements the database and enables training and testing LBBB detection algorithms.

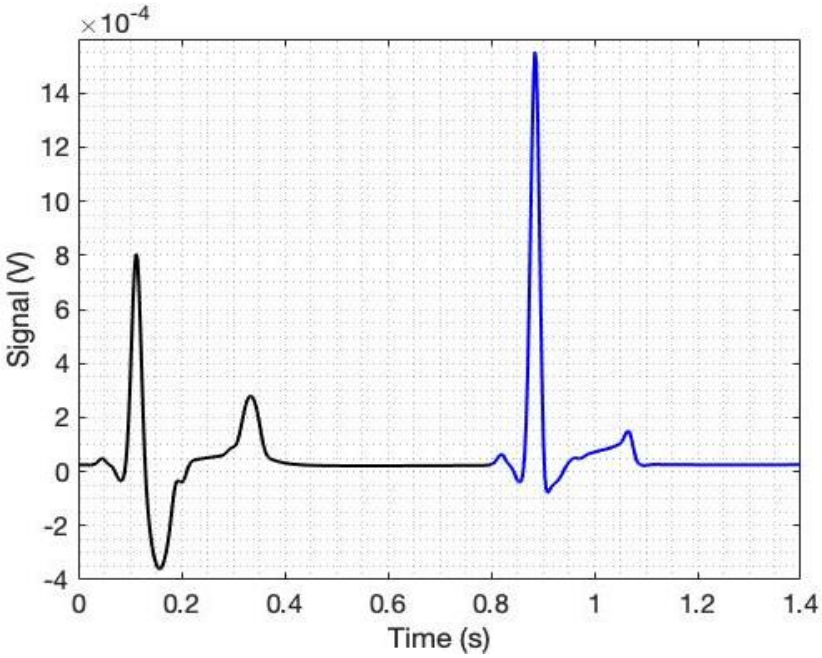


Figure 5- 11 ECG signal that contains both LBBB (black) and normal (blue) beats generated from COMSOL. The LBBB beat is correlated to the record 111 from the database, the normal beat is a prediction for the normal beat morphology based on the same model parameters and geometry.

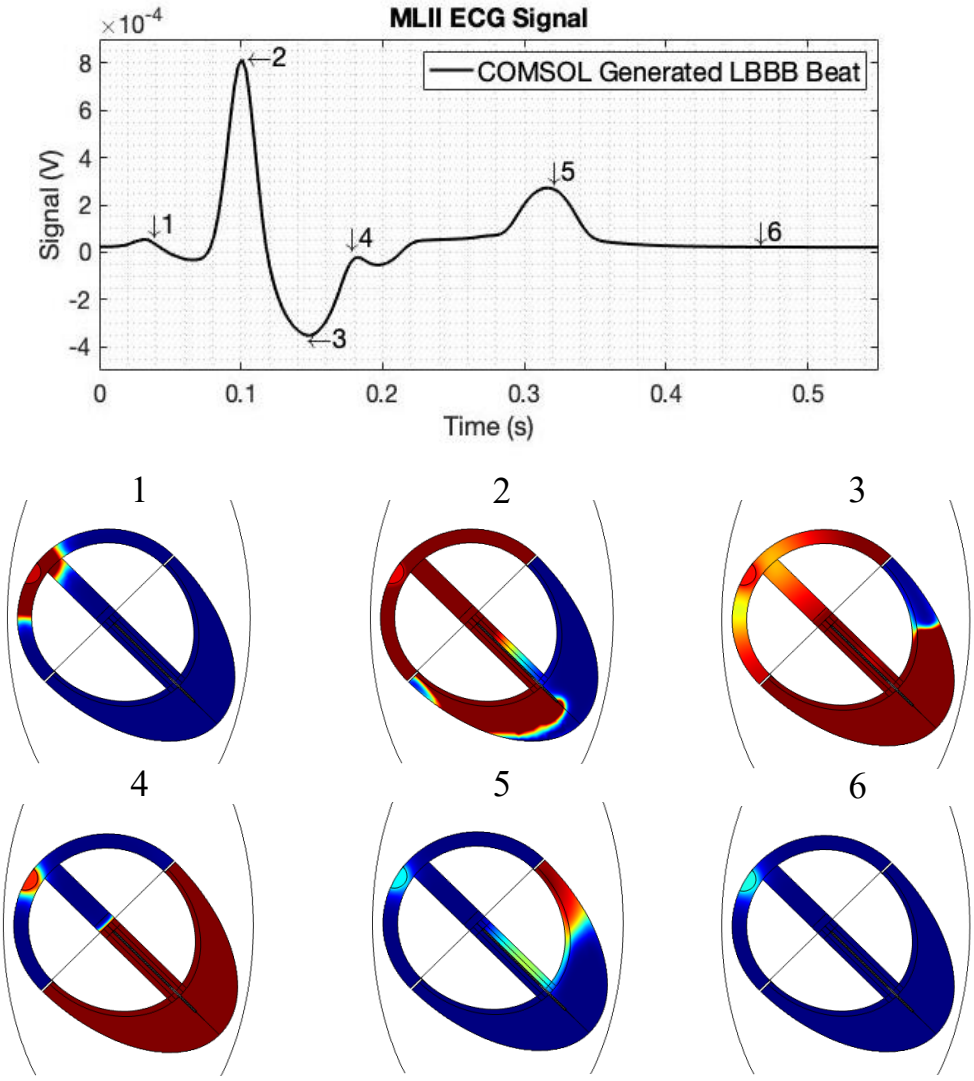


Figure 5- 12 Transmembrane potential and corresponding points on the ECG for a LBBB beat.

Successfully, long records that include normal beats and either RBBB or LBBB beats are generated from COMSOL as shown in Figure 5- 13. The Multilevel Teager Energy Operator (MTEO) algorithm [15,16] is used to extract the QRS complex to visualize the distribution of these time-domain features for the database record 212 and a single minute generated from COMSOL as presented in Figure 5- 13. Notably, the S and R peaks' distribution is separable which reveals that both features are correlated to the right bundle branch block (RBBB) condition. The COMSOL generated signal has a separable S and R peaks' distribution which agrees with the correlation coefficient values and support our hypothesis that COMSOL is capable of generating a limit-less and unbiased ECG

database. However, graphs indicate that both signals' distributions are centered around different values; this could be enhanced further through fine-tuning of the parameters used in the simulation.

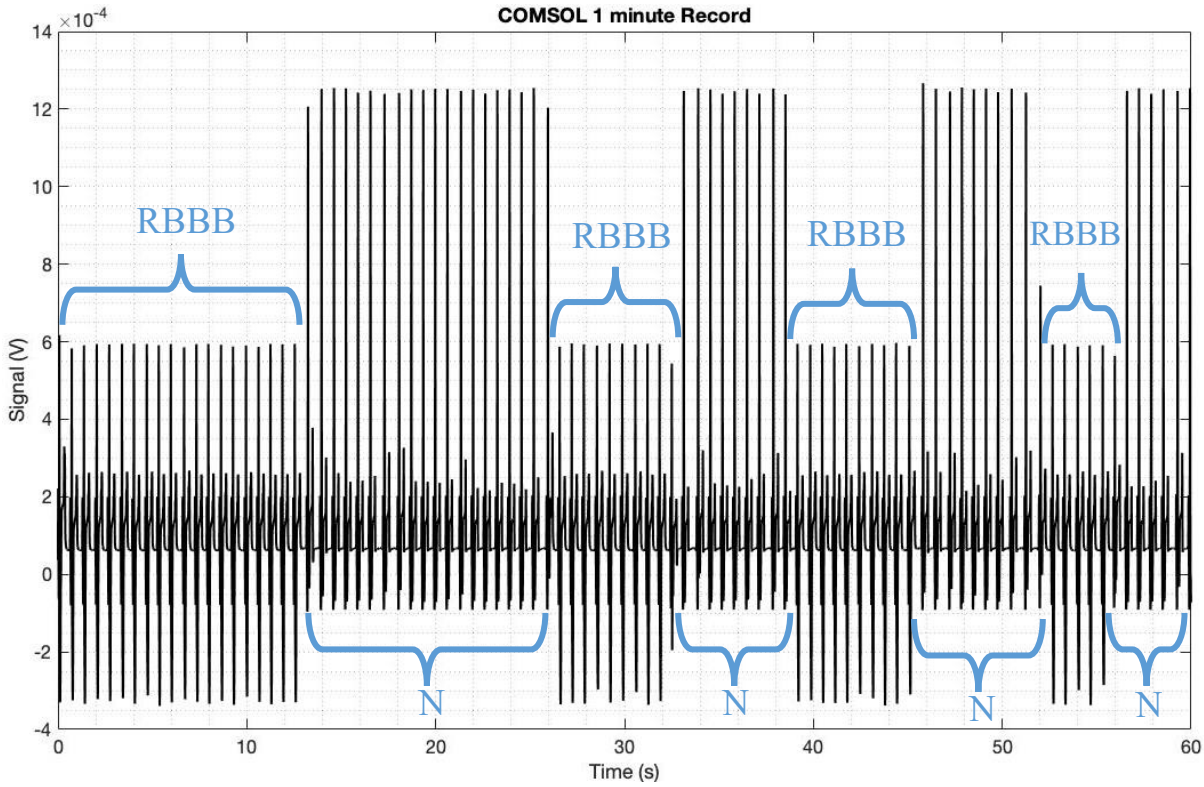


Figure 5- 13 The single minute record generated from COMSOL model with an un-biased distribution of normal (N) and RBBB beats.

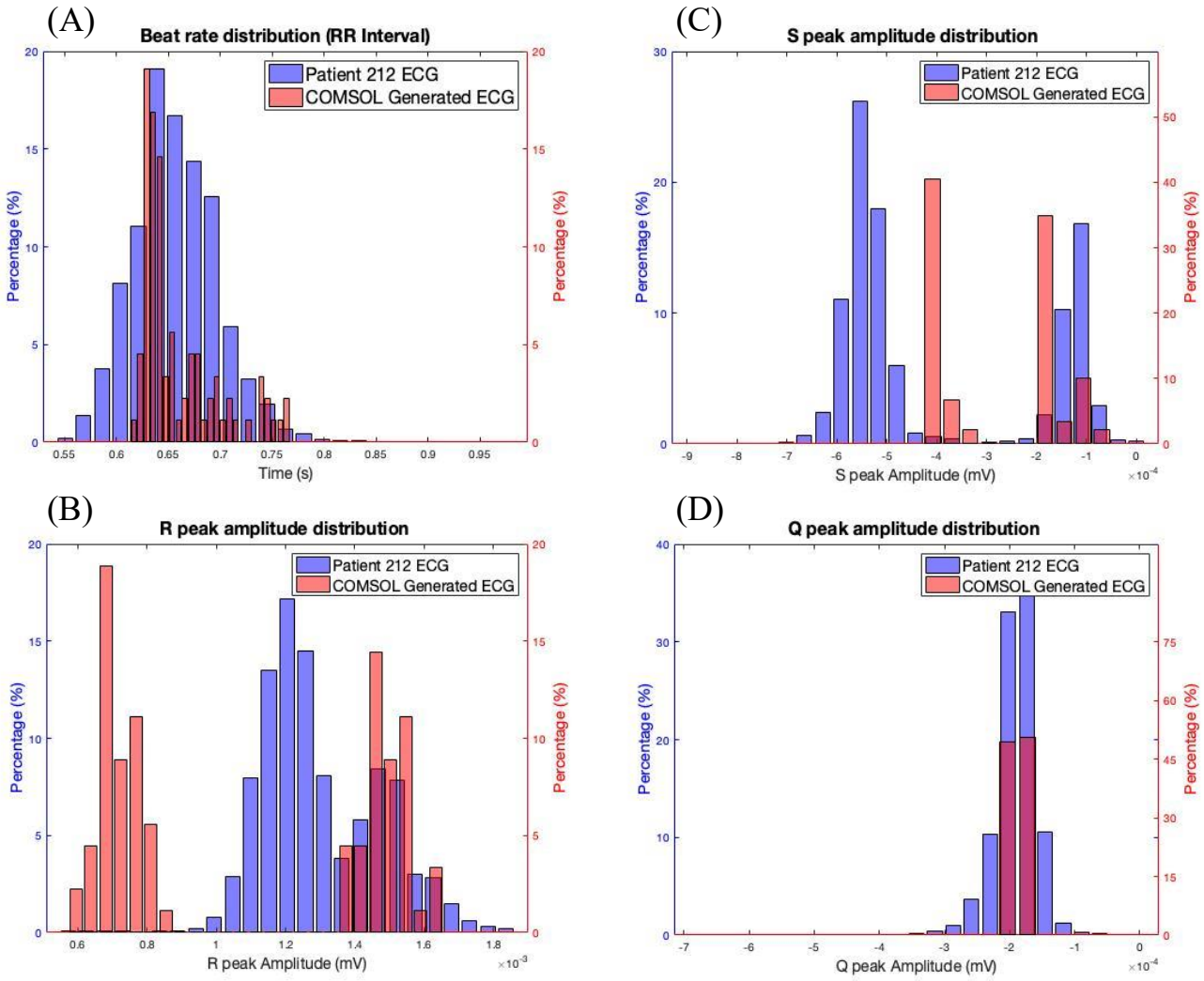


Figure 5- 14 QRS complex features distribution for the 30 minutes record 212 in the database (Blue) and the one-minute record generated from COMSOL (Red) where (A) shows the RR interval distribution, (B) shows the R peak value distribution, (C) shows the S peak value distribution, and (D) shows the Q peak value distribution.

Moreover, these long records are used for training the SVM classifier, the autoregressive (AR) and Shannon entropy (SE) feature extraction algorithms [19] are used for extracting 8 features, based on a one-second window, that are then passed to the linear (first order) SVM for training and testing with 70% and 30% segments respectively. Results are tabulated in *Table 5- 3* in comparison to results obtained from record 212 and 111. The accuracy reported is the mean of the F1 score without cross-validation of the model. For simplicity, we shall refer to the generated COMSOL records as 111_C and 212_C such that 111_C record includes normal and LBBB beats, and the 212_C record includes normal and RBBB beats. The records are 1 minute, and 10 minutes long, we have not

increased the duration further due to the increase in the size of files generated as tabulated in *Table 5- 4*. Simulations are performed with a time step of 0.0028s, a sampling frequency of 360 to match the database, on an Intel Core i7-7800x processor workstation with 128 GB of memory, and 6 cores with 12 logical processors. Additionally, white noise is added to the generated COMSOL signal with a signal-to-noise ratio (SNR) of 10 dB as shown in Figure 5- 15 to study noise's impact on detection accuracy. These preliminary results confirm the hypothesis and shed light on the significance of this work.

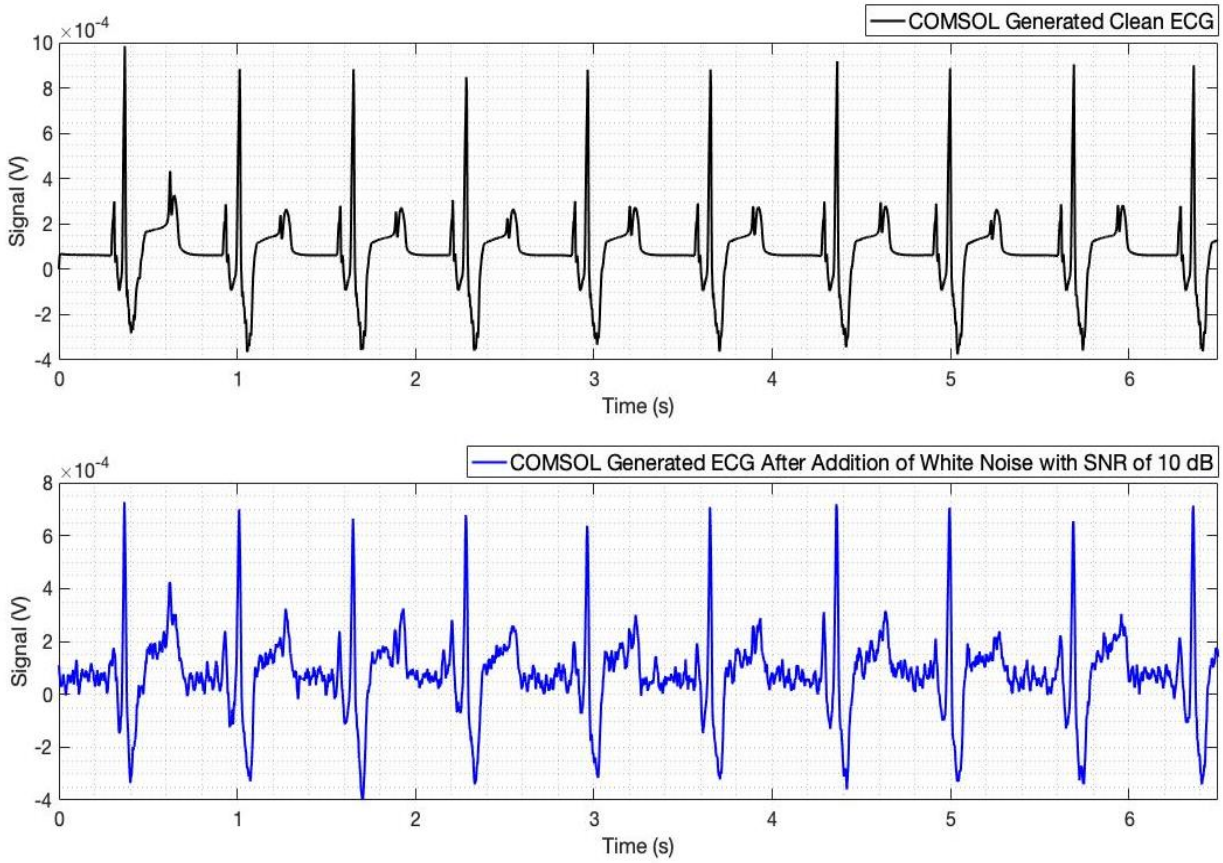


Figure 5- 15 Black curve shows a short interval of record 212_C, and the blue curve shows effect of adding randomized-white noise with signal-to-noise ratio of 10 dB.

Table 5- 3 TRAINING CONDITIONS AND TESTING RESULTS.

	Record Bias: Normal / Abnormal	**Accuracy (%)**
212 (30 minutes)	33.5% / 66.5%	**91.11**
212_C (1 minute)	48.3% / 51.7%	**70.58**
212_C (10 minutes)	49.6% / 50.4%	**97.75**
212_C (10 minutes) with white noise (SNR 10 dB)	49.6% / 50.4%	**97.75**
111 (30 minutes)	0% / 99.99%	**N.A**
111_C (1 minute)	50.6% / 49.4%	**88.19**
111_C (10 minutes)	50.3% / 49.7%	**98.31**
111_C (10 minutes) with white noise (SNR 10 dB)	50.3% / 49.7%	**99.44**

Table 5- 4 COMPUTITIONAL DETAILS.

	212_C (10 minutes)	111_C (10 minutes)
Number of Triangular Elements.	5103	4461
Computational time.	15 hrs, 37 min	11 hrs, 3 min
Results file size.	70.5 GB	62.5 GB

References

[1] G. Moody and R. Mark, "The impact of the MIT-BIH Arrhythmia Database", IEEE Engineering in Medicine and Biology Magazine, vol. 20, no. 3, pp. 45-50, 2001. Available: 10.1109/51.932724.

[2] G. Moody, R. Mark and A. Goldberger, "PhysioNet: a Web-based resource for the study of physiologic signals", IEEE Engineering in Medicine and Biology Magazine, vol. 20, no. 3, pp. 70-75, 2001. Available: 10.1109/51.932728.

[3] M. Llamedo and J. P. Martínez, "QRS detectors performance comparison in public databases," Computing in Cardiology 2014, Cambridge, MA, 2014, pp. 357-360.

[4] N. Flores, R. L. Avitia, M. A. Reyna, and C. García, "Readily available ECG databases," *Journal of Electrocardiology*, vol. 51, no. 6, pp. 1095–1097, 2018.

[5] E. Luz and D. Menotti, "How the choice of samples for building arrhythmia classifiers impact their performances", 2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 2011. Available: 10.1109/iembs.2011.6091236.

[6] S. Kotsiantis, D. Kanellopoulos and P. Pintelas, "Handling imbalanced datasets: A review," GESTS International Transactions on Computer Science and Engineering, 2006, vol. 30(1), pp. 25-36.

[7] J. Nalepa and M. Kawulok, "Selecting training sets for support vector machines: a review," Artificial Intelligence Review, 2018.

[8] S. Sovilj, R. Magjarević, A. Abed, N. Lovell and S. Dokos, "Simplified 2D Bidomain Model of Whole Heart Electrical Activity and ECG Generation", Measurement Science Review, vol. 14, no. 3, pp. 136-143, 2014. Available: 10.2478/msr-2014-0018.

[9] COMSOL Multiphysics® v. 5.3. www.comsol.com. COMSOL AB, Stockholm, Sweden.

[10] J. Li, S. Inada, H. Dobrzynski, H. Zhang and M. Boyett, "A modified FitzHugh-Nagumo model that allows control of action potential duration and refractory period," 2009 36th Annual Computers in Cardiology Conference (CinC), Park City, UT, 2009, pp. 65-68.

[11] K. Bilchick, S. Kamath, J. DiMarco and G. Stukenborg, "Bundle-Branch Block Morphology and Other Predictors of Outcome After Cardiac Resynchronization Therapy in Medicare Patients", *Circulation*, vol. 122, no. 20, pp. 2022-2030, 2010. Available: 10.1161/circulationaha.110.956011.

[12] S. Sovilj, R. Magjarević, N. H. Lovell, and S. Dokos, "A Simplified 3D Model of Whole Heart Electrical Activity and 12-Lead ECG Generation," *Computational and Mathematical Methods in Medicine*, vol. 2013, pp. 1–10, 2013.

[13] S. Sovilj, V. Čeperič, and R. Magjarevič, "3D Cardiac Electrical Activity Model," *Automatika*, vol. 57, no. 2, pp. 540–548, 2016.

[14] MATLAB and Data Analaysis Toolbox Release 2018a, The MathWorks, Inc., Natick, Massachusetts, United States.

[15] H. Sedghamiz and D. Santonocito, "Unsupervised detection and classification of motor unit action potentials in intramuscular electromyography signals," *2015 E-Health and Bioengineering Conference (EHB)*, 2015.

[16] H. Sedghamiz, "BioSigKit: A Matlab Toolbox and Interface for Analysis of BioSignals," *Journal of Open Source Software*, vol. 3, no. 30, p. 671, 2018.

[17] P. Dechazal, M. Odwyer, and R. Reilly, "Automatic Classification of Heartbeats Using ECG Morphology and Heartbeat Interval Features," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 7, pp. 1196–1206, 2004.

[18] L. Sörnmo and P. Laguna, "ECG Signal Processing," *Bioelectrical Signal Processing in Cardiac and Neurological Applications*, pp. 453–566, 2005.

[19] MATLAB and Signal Processing Toolbox Release 2018a, The MathWorks, Inc., Natick, Massachusetts, United States.

Conclusion and future work

MIT BIH Arrithmya database was used through the project for training and testing. A supervised machine learning model was used for classification which is Support Vector Machine (SVM) which is a robust classification Algorithm. The feature extraction algorithms used are Autoregressive model, and Shanon Entropy for Maximal Overlab Discrete Wavelet Packet Transform of the data. A low power Hardware implementation of these complex feature extraction algorithms was done in order to do hardware acceleration that would be useful for wearable technologies and IoT applications in the near future. A real-time FPGA implementation was done on Zynq Ultrascale+ FPGA with dynamic power less than 1mW under operating frequency of 10 KHz.

For future work, the system is to be implemented using ASIC flow to be implemented with the System on Chip (SoC).

Furthermore, more optimizations are to be completed in order to have reduction in resource utilization, which may help to add more feature detection algorithms that would help to increase abnormality detection accuracy. Moreover, disease detection will not stop only for arrythmia, it shall be extended to have more diseases in case of available data set for such diseases, as our work and feature extraction is generic and may map to more than just arrythmia. Finally, the generated data set from COMSOL is to be used as an artificial training data set to train the SVM instead of the real extracted dataset from MIT. This will provide us with very huge dataset for training which will boom the efficiency and accuracy of the system. The system is also to be extended from detection to be able to predict the disease before happening especially for heart attacks, which is a life crucial problem and may save many peoples' lives.

Additionally, this thesis records the first attempt to build an artificial ECG database generated from a COMSOL Multiphysics physical model. This attempt aimed to overcome the existing limitations of the MIT-BIH arrhythmia database;

hence, the aimed database is to have a controllable bias in its records and an unlimited records' lengths. The methodology and hypothesis are verified through correlating the artificial records generated from COMSOL with the database records, which yields high correlation coefficients (> 0.7). Moreover, the proposed methodology complements the existing database and generates a record that contains both normal beats and left bundle branch block (LBBB) beats; hence, this record can be used in training and testing. The generated records are subjected to the same feature extraction algorithms and the support vector machine (SVM) classifier, the accuracies are higher but comparable with accuracies calculated for database records.

The outlook of this part is to generate more conditions as premature ventricular contraction (PVC) and premature atrial contraction (PAC) to complement this artificial database as it should include all arrhythmia types and conditions. Additionally, longer records are to be generated with increasing the beat-to-beat randomization to improve the database's capability as a training set.