Dynamic Thresholding based Real Time ECG Signal Processing and Monitoring System using FPGA

Md. Anwarul Abedin*, **Jahid Hasan, Akash Modak and Shubro Biswas**

Department of Electrical and Electronic Engineering, Dhaka University of Engineering & Technology, Gazipur, Bangladesh

ABSTRACT

This paper presents the design and implementation of a real-time ECG signal processing and monitoring system using dynamic thresholding on an FPGA platform. The proposed system focuses on efficient and accurate detection of the QRS complex, a key feature in the ECG signal, while minimizing computational overhead. The motivation for this research stems from the growing need for reliable, high-speed medical monitoring systems capable of handling noisy ECG signals in real-time. The study demonstrates that the dynamic thresholding method, combined with FIR filters for noise reduction, offers a significant improvement in QRS detection accuracy compared to traditional constant threshold techniques. The implementation of the algorithm on FPGA hardware ensures faster processing, real-time performance, and low power consumption. The system adapts the threshold based on the previous R-peaks and RR intervals, allowing it to accurately detect heartbeats across different ECG signals. The proposed system is built using VHDL by Xilinx ISE14.6 package. Simulation of the algorithm is performed using MATLAB Version 8.3.0.532 (R2021a) System Generator. Experimental validation is conducted using multiple ECG signals from MIT-BIH databases. The results show that the proposed system achieves an error rate of 0.54%, sensitivity of 99.46% and F1 score of 99.73% outperforming similar systems in both detection accuracy and processing speed. The FPGA implementation effectively meets the requirements for real-time medical applications, ensuring high reliability in detecting arrhythmias while maintaining low computational complexity.

1. INTRODUCTION

The heart is just the size of a large fist but it is the most vital organ of human body. Its proper functioning is critical to maintaining human life; without it, life becomes unstable. Cardiovascular diseases have long been, and continue to be, the leading cause of death in the modern world. Sudden cardiac death accounts for about 50 percent of these fatalities, with the majority being caused by acute ventricular arrhythmias. These arrhythmias can sometimes occur sporadically, making them difficult to detect through routine check-ups and monitoring, further complicating the identification of underlying heart conditions [1].

Electrocardiograph (ECG) signal is a compulsive recording of the heart on a graph with help of various electrodes placed at specific points on the body. This recording helps assess the heart's functionality. By analyzing the ECG waveform, valuable insights into the patient's heart condition can be obtained. Among the key features of the ECG is the Q wave, R wave and S wave (QRS), which provides significant information related to arrhythmias. To make the most of this data, the ECG signal can be preprocessed, and advanced signal processing techniques can be employed to extract critical features, such as detecting the QRS complex, for better diagnosis and monitoring of heart health [2].

In most cases, these processes are carried out using software applications, which offer several advantages. Software is often easy to work with and can achieve high accuracy because it operates in a virtual environment where precise computations are possible. However, one of the main drawbacks of software-based QRS complex detection is the inability to provide real-time results. Typically, the ECG signal is collected from the patient, stored, and then processed offline. This method can be slow, as software applications tend to execute multiple tasks sequentially, and they lack the ability to handle tasks in parallel, which is crucial for faster ECG signal processing introduced in [3].

In contrast, signal processing using hardware presents significant advantages, particularly in terms of performance and efficiency. Hardware implementations,

*Corresponding author's email: abedin@duet.ac.bd such as Field Programmable Gate Array (FPGA) or Application-Specific Integrated Circuit (ASIC), enable real-time processing with minimal delays, as they can perform parallel signal processing, drastically improving execution speed compared to the linear nature of software. Moreover, hardware systems are generally more power-efficient, as they are custom-built to perform only the necessary function, which eliminates unnecessary computations and overhead. They also tend to be more reliable because, once deployed, hardware is less prone to errors like software bugs or system crashes [4].

FPGA is considered one of the best hardware options for signal processing due to its nature as a dedicated hardware platform. It is increasingly being utilized for various computationally intensive tasks, particularly in the fields of Digital Signal Processing (DSP) and communications. Advances in technology have led to modern FPGAs containing a large number of Configurable Logic Blocks (CLBs), making them more practical for a wider range of applications. The high Nonrecurring Engineering (NRE) costs and extended development times associated with ASICs have also made FPGAs a more appealing choice for application-specific DSP tasks. In the context of ECG signal processing, FPGAs offer a faster, more efficient and reliable solution, particularly in scenarios that demand high throughput and real-time performance [4].

In [5] the authors have proposed an efficient method for detecting the QRS complex using dynamic thresholding, designed to minimize computational overhead. The method begins by employing a series of window-based Finite Impulse Response (FIR) filters to remove baseline wander noise and high-frequency interference from the ECG signal. Once the signal is filtered, it is passed to the QRS complex detection stage. At this stage, the initial threshold for detecting the QRS complex is determined using the baseline and Root Mean Square (RMS) values of the ECG signal over the first three seconds. A dynamic thresholding process is then employed to continuously update the threshold after each *R-peak* detection. This adaptive process adjusts the threshold based on the amplitude of the previously detected *R-peak* and the RR intervals, ensuring that the detection remains accurate even as the ECG signal varies over time. The *R-peak* is detected using the updated threshold value, and subsequently, the Q-wave and S-wave are identified by searching for local minima and maxima in relation to the detected *R-peak*. After detecting the QRS complex, the heart rate is calculated based on the detected beats, and this information is passed to the decision-making stage, where the arrhythmia condition is evaluated. The proposed method has been validated by applying it to several ECG signals from various databases, demonstrating its effectiveness in detecting arrhythmias with minimal computational demands.

In [6], the authors have introduced implementation of LPF, HPF and FIR filters using FPGA. The work highlights the advantages of using FPGA for real-time digital signal processing, including high-speed computation, parallelism, and low latency. These features are essential for applications that require efficient, fast processing of signals, such as ECG signal processing in medical monitoring systems. In relation to our thesis, this paper demonstrates the practical implementation of FIR filters for noise reduction, which is a key element in ECG signal preprocessing. The filtering methods discussed are crucial for cleaning ECG signals, removing baseline wander (low-frequency noise) and high-frequency interference steps that align closely with our work. Additionally, the paper underscores the flexibility of FPGA platforms for handling digital signal processing, supporting our decision to use FPGA for real-time ECG processing.

The implementation of real-time ECG signal processing using FPGA, specifically targeting noise reduction and beat detection has been presented in [4]. The work outlines the design of filters to eliminate noise, such as baseline wander and high-frequency interference, followed by the use of algorithms to detect ECG beats. The FPGA platform's parallel processing capabilities allow for efficient and high-speed ECG signal processing, making it well-suited for real-time medical applications.

The effectiveness of FPGA for real-time ECG processing, emphasizing the need to reduce computational complexity for faster results has been proposed in [7]. The use of FPGA for parallel processing in beat detection is directly applicable to the present work in improving the performance of dynamic thresholding algorithms. However, the work does not delve into dynamic thresholding techniques or adaptive algorithms for *QRS*-complex detection, which are central to our proposed method. While it provides a solid foundation for FPGA-based ECG processing, it lacks the adaptive capability that our dynamic thresholding method introduces, thus showcasing a gap that our research aims to fill in improving accuracy and real-time performance in ECG beat detection.

In [8], an efficient algorithm for real-time *QRS* complex detection in ECG signals has been presented, with a focus on low computational complexity and adaptability to resource-constrained platforms. The combination of a pre-processing stage (which reduces noise) with a dynamic thresholding mechanism (adjusted by a finite-state machine) ensures robust detection of R peaks even in noisy environments, with reported sensitivities and positive predictivities exceeding 99%. The proposed methodology of this paper presented in Fig. 1, begins with real-time ECG data being passed through a preprocessing stage, where the signal is filtered to eliminate noise and artifacts. Once the signal is filtered, it is sent to the *QRS* complex detection unit for further analysis. This unit accurately identifies and detects the R peaks, which are critical components of the ECG waveform. The detection process involves sophisticated algorithms that ensure precise identification of the *QRS* complexes, even in noisy environments. This system provides real-time feedback, allowing for continuous monitoring of heart activity, and is particularly effective for identifying cardiac abnormalities. The combination of preprocessing and detection stages ensures both accuracy and reliability in detecting *R-peaks*.

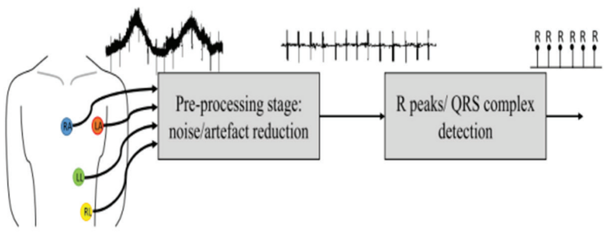
Pre-processing stage: noise/artefact reduction
R peaks/ QRS complex detection
R R R R R R

Fig. 1: Typical Structure of a QRS Complex Detection Algorithm

The study fills a crucial gap in ECG signal processing by addressing the challenge of real-time detection on devices with limited computational resources, making it suitable for portable medical applications. The outcomes show a 50% reduction in processing time and a 75% decrease in resource usage compared to the well-known Pan and Tompkins algorithm.

However, hardware implementations do not explore here such as FPGA, which could further enhance performance. This gap is relevant to our work, as we focus on FPGA based dynamic thresholding for ECG processing, which would offer real-time processing advantages that complement the algorithm's low complexity and high accuracy.

There are numerous methods for detecting the QRS complex, with thresholding being one of the more effective approaches. However, a key limitation of traditional thresholding is its inability to consistently detect peaks across varying ECG signals, as it relies on a constant threshold value. This approach fails to account for the fact that the magnitude of different ECG signals can vary significantly. Moreover, the conventional thresholding method is unsuitable for real-time signal processing since it requires an initial calculation of the signal's average to establish the threshold. To address these issues, we propose a dynamic thresholding method, which adapts the threshold value in real time, ensuring more accurate peak detection and making the system more suitable for real-time ECG signal processing.

This study aims to implement a real-time ECG signal processing and monitoring system based on dynamic thresholding using FPGA technology. The primary objective of the dynamic thresholding technique is to provide an effective and adaptive method for detecting heartbeats across various types of ECG signals. By utilizing dedicated hardware, the design is intended to ensure faster and more efficient signal processing. Once the ECG beats are detected, the system will perform an analysis to assess the heart rate and identify potential arrhythmias, enabling real-time monitoring of the patient's cardiac condition.

2. IMPLEMENTATION OF ECG SIGNAL PROCESSING SYSTEM

The complete methodology of QRS-complex detection is represented by a process flow diagram shown in Fig. 2. Although the model is designed to work with real-time signals, for simplicity, we have used a sample signal in this instance. Our proposed technique is subdivided into four stages. Since the signal may contain some noise, we first preprocess it before moving to the actual analysis. Once preprocessed, the signal is passed through the primary detection process to identify the QRS complexes. Based on the QRS complexes, arrhythmia is calculated and subsequently evaluated using a decision-making device to determine its classification.

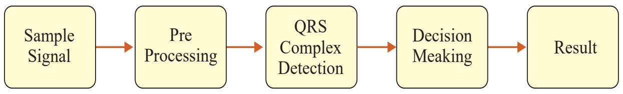

Fig. 2: Process Flow Diagram of the Proposed System

2.1 Sample Signal

For data collection, we utilize the widely recognized Physionet Massachusetts Institute of Technology- Beth Israel Hospital (MIT-BIH) Arrhythmia Database, a crucial resource in ECG signal research renowned for its precise annotations and wide-ranging diversity. This freely accessible database plays a pivotal role in facilitating research and educational initiatives globally, offering access to various types of ECG records from several patients. Each record includes two signals spanning two hours, sampled at 1000 samples per second, with a resolution of 12 bits within a 20 mV interval. Since all the 12 signals from various electrodes are stored in a single ".mat" file, we have used the MATLAB coding to separate the individual signals and separated individuals signal has been shown in Fig. 3.

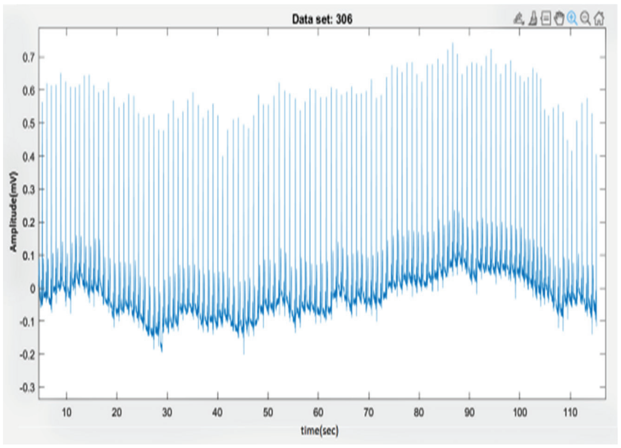

Fig. 3: Sample ECG Signal

2.2 Preprocessing

The sample ECG signal contains various types of noise, such as baseline wander, power line interference, and high-frequency noise. To ensure accurate signal analysis, the ECG signal must undergo a preprocessing stage to adjust certain parameters and remove noise. Initially, the signal is passed through a high-pass filter to eliminate baseline wander and low-frequency noise. Following this, it is processed through a low-pass filter to reduce power line interference and remove high-frequency noise, making the signal suitable for further analysis.

2.3 Peak Detection

The detection of the QRS complex is the most vital part of the ECG signal processing scheme. There are several methods for QRS complex detection, but the proposed method stands out by offering a dynamic threshold value. This allows the detection of peaks even in noisy signals, as the threshold is continuously updated every second, ensuring more accurate and reliable detection in varying signal conditions. Additionally, the signal is transmitted to a real-time monitor, allowing continuous observation and analysis of the signal. The inner functional diagram of the arrhythmia detection is shown in Fig. 4.

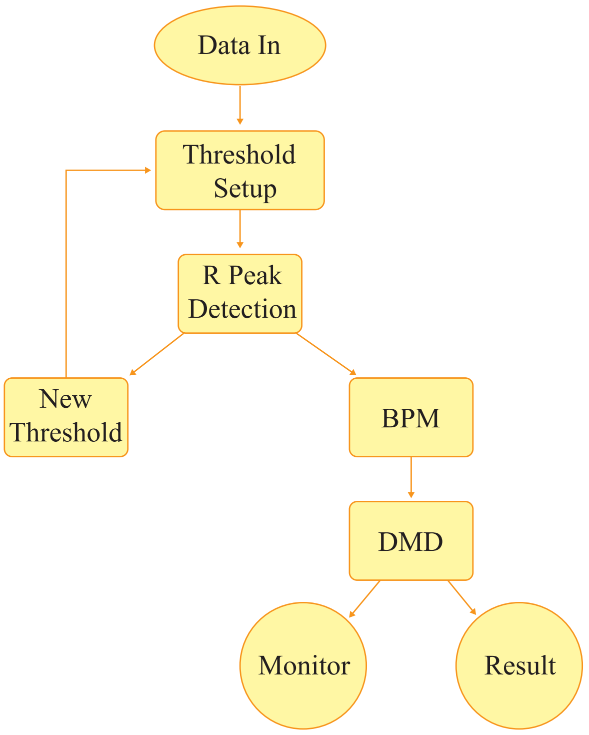

Fig. 4: Inner Functional Diagram of the Arrhythmia Detection Algorithm

2.3.1 Input Signal

The enhanced ECG signal, after passing through the filter, is stored in a Xilinx state array variable. This array is essential for comparing the previous values of the ECG signal with the new values, allowing for the accurate detection of peaks.

2.3.2 Dynamic Threshold

The threshold is a reference value used to detect the R-peak in an ECG signal. If the threshold is set as a constant, peak detection may be inaccurate, as ECG signals vary in magnitude across individuals. Additionally, the magnitude of a noisy ECG signal can fluctuate over time. In such scenarios, a constant threshold method may fail to accurately detect all peaks. This highlights the necessity of a dynamic threshold method, which adapts to changing signal conditions, ensuring more reliable and accurate peak detection. Initially, a constant value is set as the threshold. A counter is then started to measure a one-second interval, during which the maximum value of the ECG signal is determined. If this maximum value is greater than 0.1, the threshold is updated to half of the maximum value detected within that second. This condition is necessary because, in some cases, no R-peak may occur within a one-second interval. Without this safeguard, the algorithm might mistakenly select the highest value from background noise, which typically hovers near zero, as the R-peak. This would result in an incorrect threshold value.

$$TH(n) = \frac{R_{peak}(n-1)}{2} + \frac{R_{peak}(n-1)}{4} \quad \text{………….. (1)}$$

Eqn. (1) outlines the threshold calculation. In the n-th iteration, the threshold is set as half of the R-peak value from the previous iteration. This adaptive approach ensures that if the *R-peak* level decreases; the corresponding threshold value will also decrease, allowing the algorithm to adjust dynamically to changes in signal amplitude. This process ensures that the threshold is dynamically adjusted and is updated continuously at one second intervals for more precise peak detection.

2.3.3 *R-peak* Detection

The *R-peak* in the QRS complex for each beat is an important aspect to be identified. The *R-peak* acts as a reference point for the identification of other aspects of the signal. Thus, to detect the *R-peak* accurately is an important part of this algorithm. As previously explained, the algorithm designed here serves as a foundation for applications that require real-time detection of *R-peak* positions. Additionally, it is optimized for implementation on platforms with limited resources, ensuring efficient processing and performance even in constrained hardware environments. In this real-time signal processing system, it's impossible to store all the data at once for processing. Instead, the algorithm works with individual data points sequentially. The key idea of the proposed algorithm is that, we know, the value next to the peak value in any signal will be lower than the peak itself. While ECG signals have many fluctuations, this condition alone does not make any sense. But this becomes meaningful when combined with two other conditions described in (2). Together, these three conditions form an effective method for detecting the *R-peak*, which is then used to detect other peaks.

If(*data*(n) > *TH* && *data*(n) >*data*(n - 1) && *Flagtimer* = 1) *Then R-peak* = *data*(n) ....(2)

The first condition ensures that the current data point *data*(*n*) is greater than the threshold, eliminating the chance of falsely detecting other peaks as *R-peak*. If this condition holds, the algorithm checks whether the current data point *data*(n) is higher than the previous one *data*(n - 1). If both the conditions are true, which ensures that the data is a peak value laying above the threshold. So, the data is obviously denoting an *R-peak* as shown in Fig. 5(a). The problem arises with the with the falling edge of the *QRS* complex of an ECG signal. The previous both conditions are true for this edge. So, it will detect all the points on the falling edge as *R-peak*. Here comes the necessity of the third condition. We used a timer that counts 300 ms after an R-peak is detected which holds the algorithm for further detection of peak during the time interval. Initially, the timer flag is set to 1(High). When a peak is detected, the timer set the timer flag to 0(Low) and it starts then counting again. After 300 ms interval, it set the timer flag to 1 (High) again as described in eqn. (2). The 300 ms interval is chosen because the falling edge duration of *QRS* complex is typically less than that, ensuring no additional R-peak appears within that period. The process continues and the detected R-peaks will immediately be passed to the output.

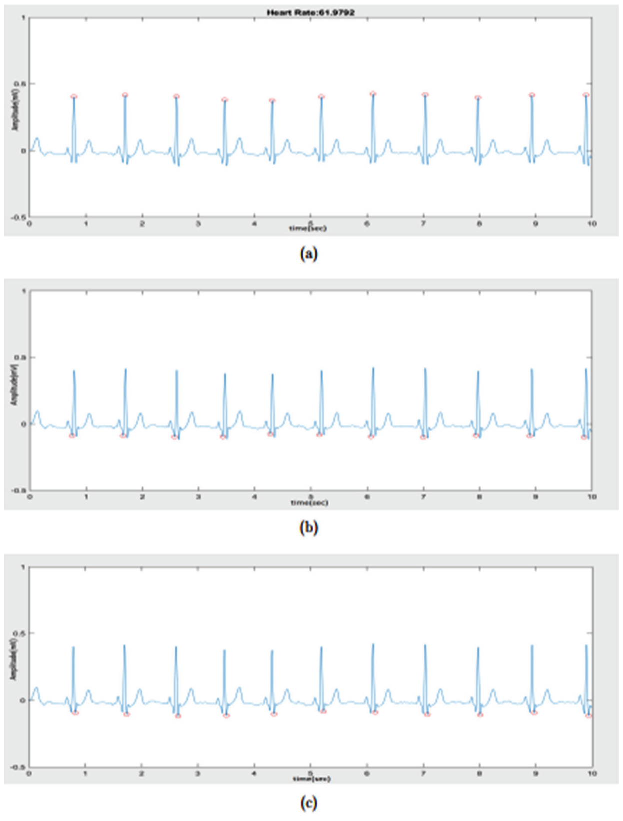

Fig. 5: Detected QRS complex in MATLAB (a) *R-peak* (b) *Q-peak* and (c) *S-peak*

2.3.4 QRS Complex Detection

The *R-peak* detected in the previous stage will be used as reference to find the remaining *Q* wave and *S* wave. The *R-peak* location plays a very important role in detecting the *QRS* complex as these tow peaks remain near the *R-peak*, we can easily detect these waves finding the minimum value near the *R-peak*. The *Q* wave represents the minimum value that occurs just before each *R-peak* within a specific time interval. We identified the *Q* wave by locating the minimum value within a 300 ms interval preceding each *R-peak*. Similarly, the *S* wave is the minimum value that follows each *R-peak* as shown in Fig. 5(b). We determined the *S* wave by finding the minimum value in the 300 ms interval following each *R-peak* as illustrated in Fig. 5(c). Once detected, the *Q* and *S* waves were stored in a file for further analysis. By combining the corresponding *Q* wave, *R* wave and *S* wave, we identified the *QRS* complex, which is then used for heart rate detection in the next stage.

2.3.5 Determination of Heart Rate

There are several ways of calculating the heart rate (HR). We used the following technique to calculate Beats Per Minute (BPM) of an ECG signal. we first identify the number of heartbeats detected within a specific time frame. The ECG records electrical activity of the heart over time and each *R-peak* corresponds to a single heartbeat. The formula we used for calculating the heart rate is:

$$HR = \frac{No_{beat} * 60}{Time}(bpm) \qquad \text{…………..} (3)$$
$$Time = \frac{No_{sample}}{f_s}(sec)$$

Here, No_{beat} refers to the number of *QRS* complex detected within the given time (in seconds), and "time" refers to the duration over which the beats are measured. The result is then multiplied by 60 to convert it from beats per second to beats per minute. This method provides an accurate estimation of heart rate based on the detected *QRS* complexes in the ECG signal.

2.3.6 Decision Making Device

Decision making device (DMD) is a crucial component. The primary function of the DMD is to assess the HR derived from the previous signal processing stage and determine the corresponding heart condition. We have used tow relational operator blocks to design this DMD. The DMD evaluates the HR by comparing it against predefined thresholds specifically, 60 BPM and 100 BPM. If the HR exceeds 100 BPM, the DMD classifies the condition as tachycardia, indicating an abnormally fast heart rate. Conversely, if the HR is below 60 BPM, it identifies the condition as bradycardia, signifying an unusually slow HR mentioned in [9]. For HR values within the range of 60 to 100 BPM, the DMD categorizes the heart condition as normal described in eqn. (4). This simple yet effective classification method enables real-time monitoring of cardiac health, providing essential information for detecting and responding to abnormal heart rhythms in a timely manner.

$$\begin{cases} \text{if} & HR > 100 & \text{Result} = \text{"Tachycardia"} \\ \text{else if} & HR < 60 & \text{Result} = \text{"Bradycardia"} \\ \text{else} & & \text{Normal} \end{cases} \qquad \text{……..} (4)$$

2.3.7 Monitor

The monitoring section of the system plays a vital role in this real-time ECG signal observation and interpretation. This part of the design continuously monitors the ECG signal, displaying crucial information such as the HR and the heart's condition. Once the heart rate is determined, it is sent to the DMD, which classifies the heart condition into one of three categories: tachycardia, bradycardia, or normal, based on predefined thresholds. Both the HR and the classified heart condition are displayed on the monitoring system interface, providing immediate feedback. This real-time display allows for continuous tracking of cardiac health, enabling quick diagnosis and response to abnormal heart conditions. The integration of dynamic thresholding ensures that the system adapts to signal variations, ensuring accurate monitoring and timely detection of any peak.

3. FPGA IMPLEMENTATION

In the FPGA implementation phase, we began by executing the proposed algorithm and simulating the design in MATLAB Simulink to verify its functionality before moving on to hardware implementation. The simulation results, which will be included in this section, provided a crucial preliminary validation of the design. After successfully simulating the design in Simulink, we transitioned to the Xilinx System Generator for hardware implementation.

3.1 Simulation Using MATLAB Simulink

Figure 6 show the simulation of the proposed ECG signal processing system was conducted using MATLAB Simulink to validate the algorithm's functionality before hardware implementation. Simulink provides a graphical environment to model and simulate dynamic systems, making it ideal for testing complex signal processing designs. By using Simulink, we were able to simulate various stages of the ECG processing system, including signal acquisition, filtering, and HR detection. The results from these simulations will be presented in the following sections, demonstrating the system's performance and accuracy prior to hardware integration.

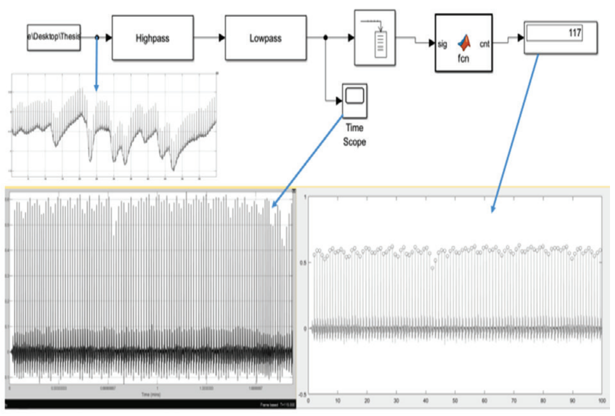

Fig. 6: Simulation Model of the Proposed System with Output Wave Shapes

3.2 Implementation Using System Generator Blocks

It is important to note that standard Simulink blocks are not directly supported by hardware, meaning they are more suited for algorithm testing and simulation rather than real-world execution on FPGA devices. In contrast, Xilinx System Generator blocks are specifically designed for FPGA-based systems, allowing for direct implementation on hardware. System Generator is a powerful tool that integrates seamlessly with Simulink, enabling FPGA code generation and ensuring the design is optimized for real-time hardware execution. By using System Generator blocks, the design is effectively translated into hardware-level operations, providing a path from simulation to physical implementation. The results from the System Generator simulations will also be included to illustrate the successful transition from software to hardware. This methodology ensures the reliability and accuracy of the FPGA-based real-time ECG signal processing and monitoring system.

Figure 7 illustrates the overall architecture of the proposed system, implemented using System Generator blocks. The filters are designed using the FDA Tool in MATLAB Simulink, with the filter coefficients saved in a MATLAB file and subsequently uploaded into the filters. The results for each segment are presented in the Fig. 8.

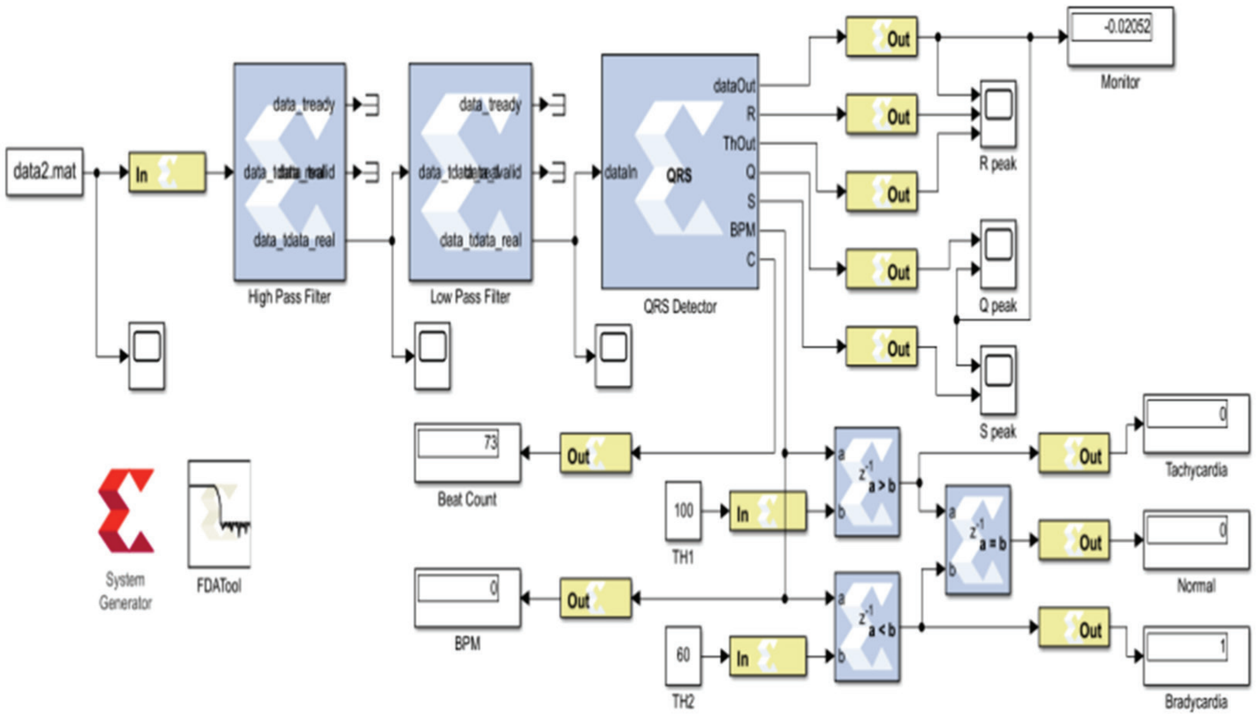

Fig. 7: Architecture of the *QRS* Complex Detection and Monitoring System Using System Generator Blocks

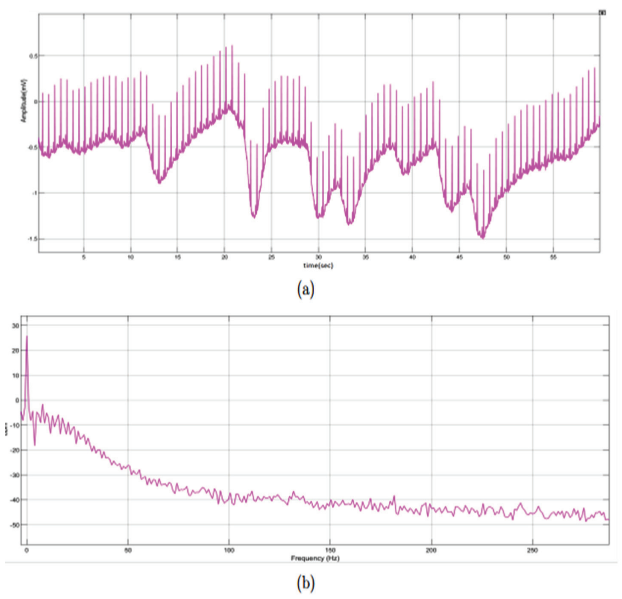

Fig. 8: Raw Ecg Signal for System Generator (a) Signal in Time Domain and (b) Signal in Frequency Domain

4. RESULTS AND DISCUSSION

The proposed method was applied to ECG signals from 10 different subjects from the MIT-BIH database to evaluate its performance. After detecting the heartbeats, a comparative analysis was conducted between the actual beats and the beats detected by the system. The results, summarized in Table I, include the percentage error between the actual and detected beats, highlighting the accuracy of the system.

Table I: Performance Evaluation of Proposed Methodology of Beat Detection

Data set	Actual Beats	Detected Beats	% error
101	122	119	2.46
113	123	123	0.00
118	176	176	0.00
122	162	161	0.61
124	180	180	0.00
136	163	162	0.61
145	169	169	0.00
162	116	116	0.00
174	152	150	1.31
187	107	106	0.93
Total	1470	1462	0.54

The results of our proposed ECG signal processing system demonstrate its high accuracy and reliability in real-time beat detection. As shown in Table I, the overall error rate of the system is just 0.54%, highlighting its efficiency in detecting heartbeats. Out of a total of 1,470 beats, only 8 beats were undetected, which is a low error margin compared to similar systems. From this result we have calculated the sensitivity and F1 score and the values are found 99.46% and 99.73% respectively. Compared to ref [5] though the sensitivity is little low but the F1 score was found better. This showcases the effectiveness of the dynamic thresholding algorithm implemented in FPGA, which adapts varying ECG signal amplitudes and noise levels, ensuring accurate detection of the *QRS* complex. Furthermore, Table II evaluates the performance of the DMD, which uses the detected beats to analyze the heart rate and assess potential arrhythmia conditions.

Table II: Performance Evaluation of the DMD

Data set	Actual BPM	Detected BPM	DMD Output
096	110.0	109.2	Tachycardia
113	64.00	64.00	Normal
162	58.56	58.00	Bradycardia
174	79.10	78.12	Normal
181	105.3	104.5	Tachycardia
187	55.00	55.00	Bradycardia
193	89.82	89.34	Normal
207	44.64	43.00	Bradycardia
399	108.00	107.3	Tachycardia
412	103.00	102.0	Tachycardia

Although there are minor discrepancies in the detected BPM compared to the actual BPM values, the overall output of the DMD is 100% accurate in terms of identifying arrhythmias. This demonstrates that, despite small errors in BPM detection, the system is highly reliable in diagnosing cardiac conditions. The integration of real-time signal processing with FPGA and the dynamic thresholding method has significantly improved the system's performance, making it a robust solution for medical applications. Additionally, the real-time monitoring system continuously updates the BPM on the display every minute, providing ongoing feedback on the heart's condition. The arrhythmia status, as determined by the DMD, is also displayed, offering immediate identification of tachycardia, bradycardia, or normal heart rhythms. Alongside these results, the enhanced real-time ECG signal is monitored on the system interface, allowing for continuous observation and analysis of the cardiac activity. This comprehensive monitoring and analysis showcase the system's effectiveness in real-time heart rate detection and condition monitoring.

5. CONCLUSION

The paper focuses on designing and implementing an ECG signal processing and monitoring system that sacrifice memory utilization for improved performance, accuracy, speed, and real time operation. It also explores various strategies in signal processing system design, including window filtering, adaptive method and thresholding method. The dynamic thresholding technique updates the threshold value continuously after each *R-peak* detection. The advantage of this method over traditional ECG signal processing systems is a real time processing system with dynamic detection technique. The proposed system can be applied for different types of signals and also it can detect the peaks form noisy signals. The system is suitable for applications where different types of ECG signal are needed to be processed and high speed is required. The work also focuses on designing and analyzing a high-performance system using FPGA hardware. The implemented system detected 1462 beats among 1470 beats from several ECG signals. The overall percentage of error of the system was 0.54%, sensitivity of 99.46% and the F1 score was 99.73%. The FDA tool is used to design the filters to preprocess the raw signal and the system generator block sets are used for hardware implementation. The analysis includes beat detection error and evaluation of DMD to classify the signal condition. The findings of this work provide that the system cannot detect a peak if the magnitude difference between that peak and the previous peak is greater than the threshold value. Additionally, the system may miss a peak at the very beginning when the threshold has not yet been established.

REFERENCES

[1] J. F. Pascual, P. J. Marchite, J. R. Silva and N. R. Gándara, "Arrhythmic syncope: From diagnosis to management," World Journal of Cardiology, Vol. 15, Issue 4, pp. 119-141, 2023.

[2] M. F. Safdar, R. M. Nowakand P. Palka, "Pre-Processing techniques and artificial intelligence algorithms for electrocardiogram (ECG) signals analysis: A comprehensive review," Computers in Biology and Medicine, Vol. 170, 107908, 2024.

[3] R. Popa, "ECG signal filtering in FPGA," 6th International Symposium on Electrical and Electronics Engineering (ISEEE), pp. 1–6, 2019.

[4] M. B. Altman, W. Wan, A. S. Hosseini, S. A. Nowdeh and M. Alizadeh, "Machine learning algorithms for FPGA Implementation in biomedical engineering applications: A review," Heliyon, Vol. 10, Issue 4, e26652, 2024.

[5] Z. Habibi, K. Karimizadeh, A. Nosratpour and I. Alipourfard, "Enhanced QRS detection and ECG compression using adaptive thresholding: A real-time approach for improved monitoring and diagnosis," Computers and Electrical Engineering, Vol. 119, Part A, 109528, 2024.

[6] E. S. Kolawole, W. H. Ali, P. Cofie, J. Fuller, C. Tolliver and P. Obiomon, "Design and implementation of low-pass, high-pass and band-pass finite impulse response (FIR) filters using FPGA," Circuits and Systems, Vol. 6, pp. 30-48, 2015.

[7] J. Rahul, M. Sora and L. D. Sharma, "Dynamic thresholding based efficient *QRS* complex detection with low computational overhead," Biomedical Signal Processing and Control, Vol. 67, p. 102519, 2021.

[8] V. K. Sinha and S. K. Kar, "An efficient real-time ECG QRS-complex identification by A-CLT and digital fractional order differentiation," Biomedical Signal Processing and Control, Vol. 92, 106055, 2024.

[9] Y. Chou, J. Gu, J. Liu, Y. Gu and J. Lin, "Bradycardia and tachycardia detection using a synthesis-by-analysis modeling approach of pulsatile signal," IEEE Access, Vol. 7, pp. 131256–131269, 2019.