Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 9138578, 14 pages
https://doi.org/10.1155/2018/9138578

Research Article

Embedded System Based on an ARM Microcontroller to Analyze Heart Rate Variability in Real Time Using Wavelets

Victor H. Rodriguez,[1] **Carlos Medrano**,[1,2] **and Inmaculada Plaza**[1,2]

[1]*EduQTech, E.U. Politecnica, Universidad de Zaragoza, c/Atarazana 2, 44003 Teruel, Spain*
[2]*IIS Aragón, Universidad de Zaragoza, Zaragoza, Spain*

Correspondence should be addressed to Victor H. Rodriguez; victor_rod_ont@hotmail.com

Received 15 June 2018; Revised 18 September 2018; Accepted 26 September 2018; Published 16 October 2018

Guest Editor: Guangjie Han

The analyses of electrocardiogram (ECG) and heart rate variability (HRV) are of primordial interest for cardiovascular diseases. The algorithm used for the detection of the QRS complex is the basis for HRV analysis and HRV quality will depend strongly on it. The aim of this paper is to implement HRV analysis in real time on an ARM microcontroller (MCU). Thus, there is no need to send raw data to a cloud server for real time HRV monitoring and, consequently, the communication requirements and the power consumption of the local sensor node would be far lower. The system would facilitate the integration into edge computing, for instance, in small local networks, such as hospitals. A QRS detector based on wavelets is proposed, which is able to autonomously select the coefficients the QRS complex will be detected with. To validate it, the MITBIH and NSRDB databases were used. This detector was implemented in real time using an MCU. Subsequently HRV analysis was implemented in the time, frequency, and nonlinear domains. When evaluating the QRS detector with the MITBIH database, 99.61% positive prediction (PP), 99.3% sensitivity (SE), and a prediction error rate (DER) of 1.12% were obtained. For the NSRDB database the results were a PP of 99.95%, an SE of 99.98%, and a DER of 0.0006%. The execution of the QRS detector in the MCU took 52 milliseconds. On the other hand, the time required to calculate the HRV depends on the data size, but it took only a few seconds to analyze several thousands of interbeat intervals. The results obtained for the detector were superior to 99%, so it is expected that the HRV is reliable. It has also been shown that the detection of QRS complex can be done in real time using advanced processing techniques such as wavelets.

1. Introduction

The ECG signal is a defined waveform representation that shows the phases through which the heart passes. The signal represents the polarization and depolarization of the atrium and the ventricle (see Figure 1) [1]. With the ECG, doctors can detect heart disease across the heart rate variability (HRV). Bearing in mind that cardiovascular diseases generate 30% of global deaths, the analysis of the ECG is considered a topic of great interest for researchers [2]. The heart waveform is also often referred to as the QRS complex and is the basis for most of the algorithms used for ECG analysis.

Basically, the detection of the QRS complex consists in detecting the R peaks of the ECG signal that is the peak between the Q and S waves (see Figure 1). The variability of time between R peaks is the basis for HRV analysis. In the literature, the interval between R peaks is usually referred to as interbeat interval (*IBI*) [3], normal to normal (*NN*) [4], or RR interval (*RR*) [5]. From now on, any of them will be used interchangeably. The analysis of the HRV is a noninvasive method that allows analyzing the activity of the autonomic nervous system (ANS). Likewise, it has been found that HRV alterations are linked to cardiovascular diseases [6–8] or that meditation can alter HRV patterns [9]. There are several methods to measure the HRV, but the most common belong to three categories: time domain, frequency domain, and nonlinear. In the time domain, statistical and geometric measures are included [5, 6, 10].

The main problem of analyzing the ECG signal is the noise present due to its susceptibility to interferences such as power line, RF interferences, and muscle artifacts, complicating the detection of the QRS complex (see Figure 1). This is why in recent years different types of algorithms have been developed for the elimination of noise and the detection

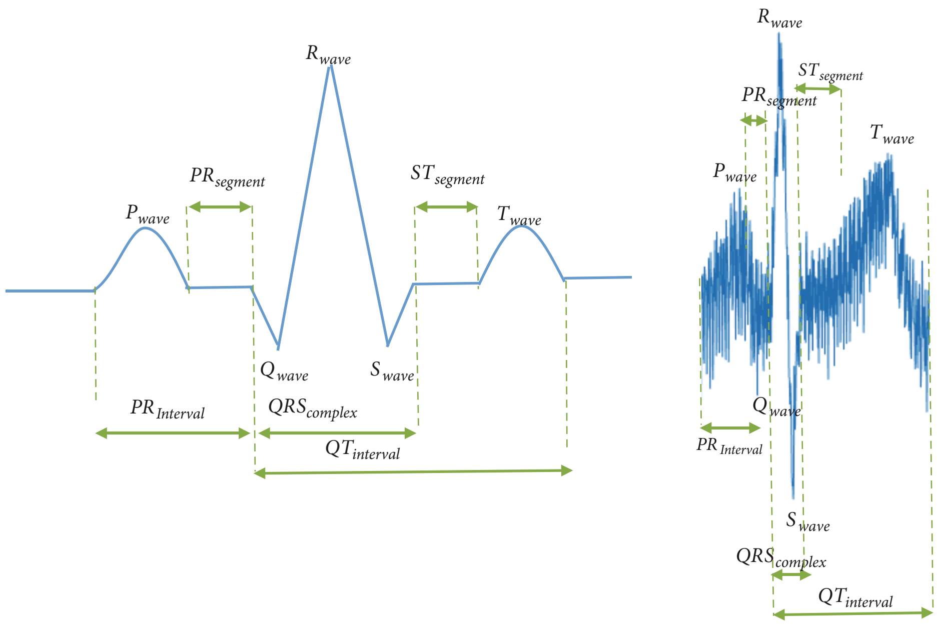

Figure 1: ECG waves, intervals, and segments. Left: theoretical wave; right: real wave.

of QRS complex. Pan and Tompkin were pioneer in this topic. Their algorithm consisted in using a digital band pass filter and a dynamic threshold [11]. Subsequently, algorithms with more advanced techniques for the detection of the QRS complex emerged, such as the use of wavelets [5, 12], adaptive filters [13], Differential Threshold [14], Level-Crossing Sampling [15], Hidden Markov Models [16], S-transform [17], and many more. The use of wavelets has allowed the detection of R peaks even in different scenarios like varying QRS morphologies and high grade of noise. It has obtained the best results. The adaptive filters use methodologies based on the leaky-LMS (LLMS) algorithm of LMS family. The differential threshold algorithms outstand for their low computational requirements. The level-crossing sampling was tested with a hardware implementation, leading to an ECG-monitoring system with a low energy consumption, noise cancelation and low-drawn input current leads. The application of HMM and S-transform are in a more experimental phase.

Recent studies have shown that HRV analysis in the frequency domain reveals the activity of the sympathetic nervous system (SNS) and the parasympathetic nervous system (PNS), where the high frequencies band (HF: 0.15-0.40 Hz) corresponds mainly to the activity of the PNS, and the low frequency band (LF: 0.04-0.15 Hz) corresponds to the activity of the SNS [5]. It has also been found that the reduction of HRV and the increase in the LF/HF ratio are associated with several cardiovascular diseases [5, 6].

In recent years the eHealth and mHealth services have grown and they are expected to continue growing in order to offer more efficient services to patients [18] thanks to the growth of the Internet of Things (IoT) and the improvement of portable devices in the area of health. On the other hand, as smart devices are increasingly involved in people's lives (for example, fall detection systems, monitors of physical activity, vital signs, or sleep quality), they require wide bandwidths and lower latencies. The use of cloud computing is not recommended in applications that require very low latencies between the data sources and the processing unit [19]. Some specific examples try to define strategies to overcome the associated problems. For instance, Gonzalez-Landero et al. [20] made an intelligent tracking system of the heart rate which predicts the hours in which it is high. Then, the heart rate is measured with high frequency (every minute) at certain moments and low frequency (every 10 minutes) at other moments. This saves energy in communication. However, this cannot be generalized to any kind of measurements since the requirements for sampling are higher or changing the rate of the communications is not an option. For HRV, the sampling frequency must be very high (500 Hz recommended) and the possible reduction of communication implies a computation in a local node, which is the proposed approach of this paper.

To solve the problems of cloud computing, a new processing technique has emerged, edge computing. In contrast to cloud computing, in edge computing the data generated by the device is processed in the network edge instead of being transmitted to a centralized cloud for processing, resulting in very low latencies and lower bandwidth requirements [19, 21]. The characteristics of edge computing make this technique the most suitable for many eHealth and mHealth applications in which sending raw data would not be feasible. Health applications are one of the typical areas of edge computing [22]. In [23] real-time signal processing algorithms are proposed to be implemented in a local node,

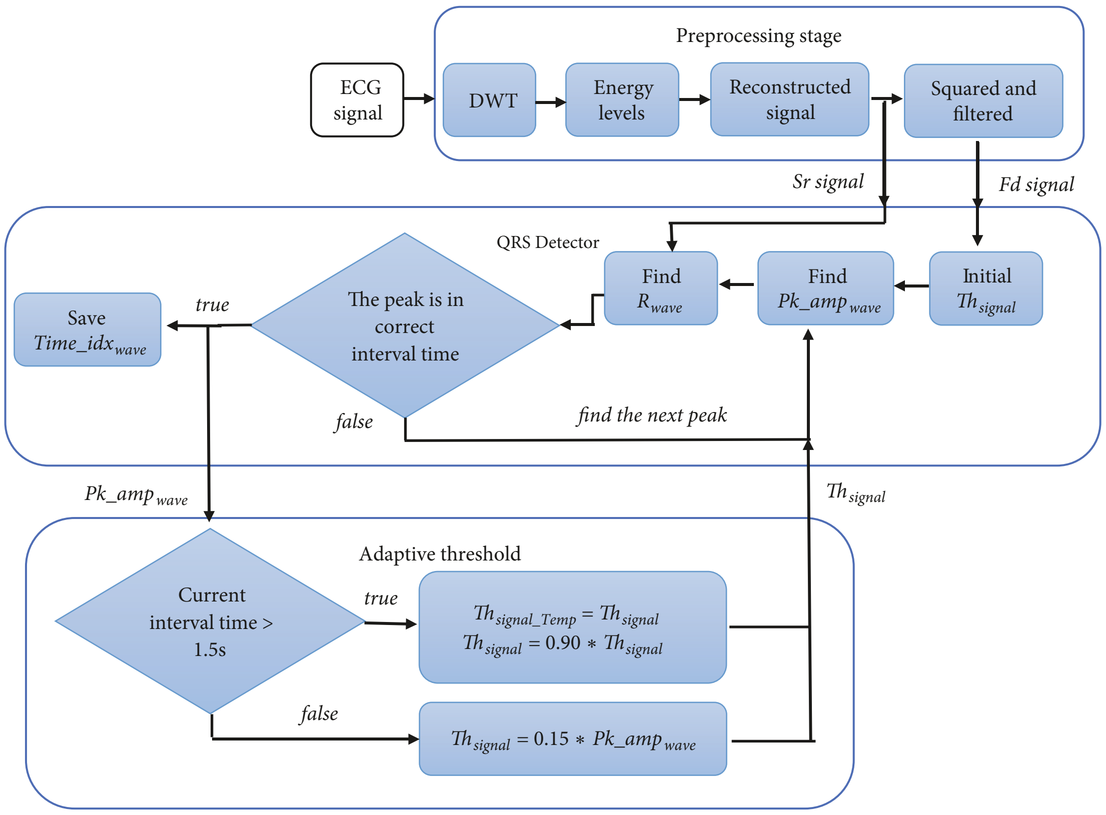

Figure 2: Flow chart of the complete QRS detector.

closer to the sensing environment. They are responsible for all the real-time processing of health-related collected data to enable a set of personalized services. The proposed scenario includes applications for gas leak detection, fall detection, and pulse and oxygen abnormal level detection. Sometimes the computation in local nodes requires the search for algorithms effective yet simple enough to be run in low end processors. In [24] a wireless acoustic sensor for ambient assisted living is proposed in keeping with the philosophy of edge computing. The proposed sensor is able to record audio samples at least to 10 kHz sampling frequency. It is capable of doing audio signal processing without compromising the sample rate and the energy consumption.

The aim of this paper is to propose a portable system capable of doing a real-time analysis of the HRV using an ARM microcontroller. The solution adopted is efficient in terms of energy by avoiding communication of raw data. Within this aim, we have developed and improved a QRS complex detector using wavelets. This detector has the capability of selecting autonomously the coefficients to detect the R peaks. The implementation on the MCU required the optimization and improvement of the functions for HRV analysis. The proposed device is designed to be used in a portable way in small local networks, such as hospitals, where the advantages offered by edge computing can show up, especially in topics related to privacy, in addition to a real-time analysis of patients. In this way the quality of the mHealth services could be increased. The system could also be used in applications for remote HRV monitoring like in [25] or [26].

The QRS detector proposed in this work is an extension of the paper sent to the International Conference On Biomedical Engineering and Applications (ICBEA) [27]. The differences with respect to [27] are the following: (i) in the current paper the detector is analyzed in more detail (block diagram of the detector, use more variable symbols so that the text can be followed easily, use more images and the full set of conditions to find an R peak); (ii) In addition, the HRV is also measured in the current work; (iii) the optimization of the algorithms for a lower RAM consumption is described, giving the possibility of creating applications in embedded systems with limited resources and achieving real-time capability.

2. Materials and Methods

2.1. Complex QRS Detector Algorithm. The proposed algorithm consisted of three stages. The first (preprocessing) was responsible for filtering and adjusting the signal for the detector. In the second stage the detector itself was implemented, which decided whether the found peak was an R peak or not. And finally, in the third stage an adaptive threshold was built, updating its level with the last peak found. Figure 2 shows a block diagram of the complex QRS detector algorithm.

(a) Preprocessing Stage. The first step in this stage was to apply the Discrete Wavelet Transform (DWT). This tool is based on

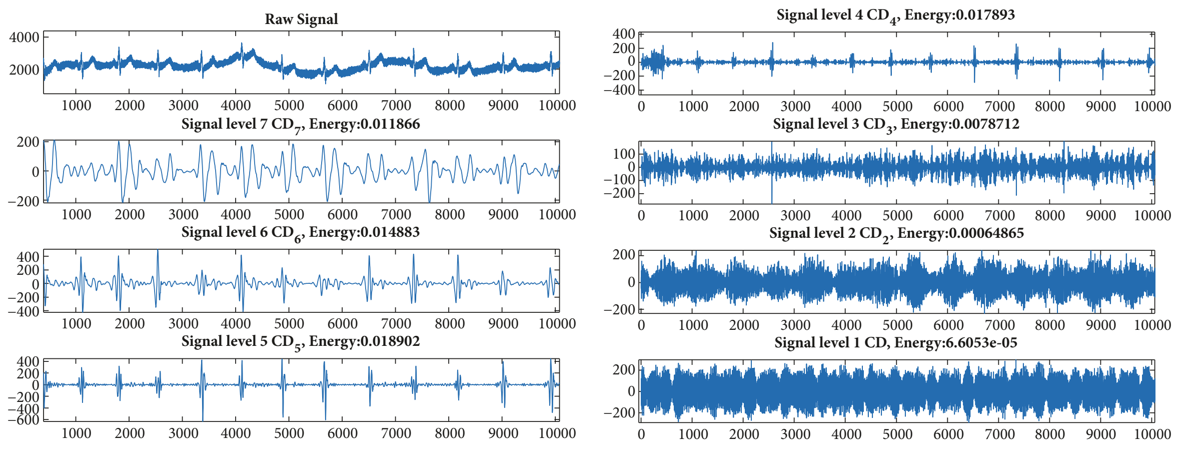

Figure 3: Decomposition of the signal into wavelets and their energy levels in percentage.

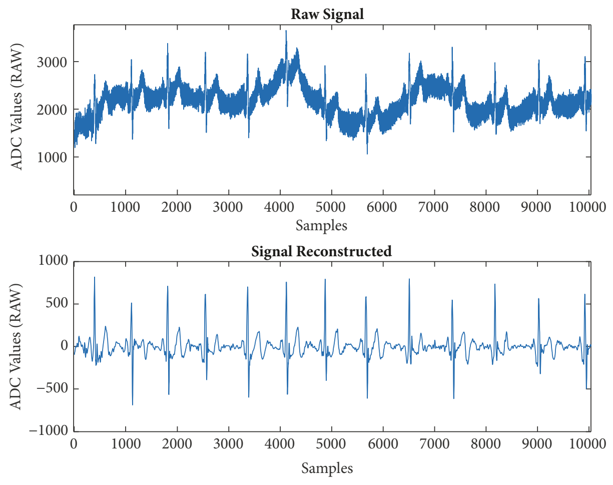

Figure 4: Signal reconstructed with 4 levels of detail coefficients (5, 4, 6, and 7).

the decomposition of a signal in subbands by means of the use of a pair of digital filters (low pass and high pass filters). The outputs of the low pass filter are named approximation coefficients (CA_n), while the outputs of the high pass filter are named detail coefficients (CD_n), where n represents the level of the subband. This process of decomposition through filtering is repeated n times. In each iteration the signal is subsampled by a factor of 2. In practice, the DWT is implemented with the Mallat pyramid algorithm [28]. Some studies have shown that the use of a four order Daubechies wavelet is one of the most effective when processing ECG signals [29]. Afterwards, the energy percentage of each level was calculated (see (1)-(3) and Figure 3), then the four ones with more energy were selected to reconstruct the signal (Figure 4). In this way it was assured that the levels with more information of the ECG signal were selected because noise or some interferences such as those of the electrical network or the artifacts are usually found at low energy levels (high frequencies, generally between CD_1 and CD_2) (see Figure 3). Therefore, using energy levels to discriminate the noise of QRS complex was a good option. Finally we proceeded to remove the offset of the signal by leaving out the approximation coefficients (CA_7). In the present study a db4 mother wavelet with 7 levels of decomposition was selected. The number of levels was selected because the data were processed in buffers of 1024 and thus the number of iterations allowed was 7.

$$E_{TCD} = \sum_{j}^{N} \left(\sum_{i}^{l} \left| CD_{ij} \right|^2 + \sum_{i}^{l} \left| CA_{ij} \right|^2 \right) \tag{1}$$

$$ECD_j = \sum_{i=1}^{l} \left| CD_{ij} \right|^2 \quad j = 1, 2, \dots, N \tag{2}$$

$$PCD_j = \frac{ECD_j}{E_{TCD}} * 100 \quad j = 1, 2, \dots, N \tag{3}$$

where E_{TCD} is total energy of all CD, ECD_j is energy on each CD, PCD_j is percent of energy on each CD, j is number of decomposition levels, i is number of coefficient on each CD, l is length of CD, and N is maximum number of decomposition levels.

The next step was to reconstruct the signal (Sr) using only the coefficients of the 4 details with the highest energy (see Figure 4). Then, the first difference was applied (4) and later it was squared to emphasize the R peaks (5). Finally, by means of an average filter, the signal was smoothed using a window of 0.2 s ($windowsSize$ samples) (see (6)). The size of the average filter is an important factor. If the $windowsSize$ is too wide, the filter will merge the QRS and T complexes. If it is too narrow, the QRS complex will produce several peaks and it can cause difficulties in its detection. Generally, the size should be approximately as wide as the QRS complex [11].

$$Sd[n] = Sr[n] - Sr[n-1] \tag{4}$$

$$Sq[n] = (Sd[n])^2 \tag{5}$$

$$Fd[n] = \frac{1}{windowsSize} * (Sq[n] + Sq[n-1] + \cdots + Sq[n - (windowsSize - 1)]) \tag{6}$$

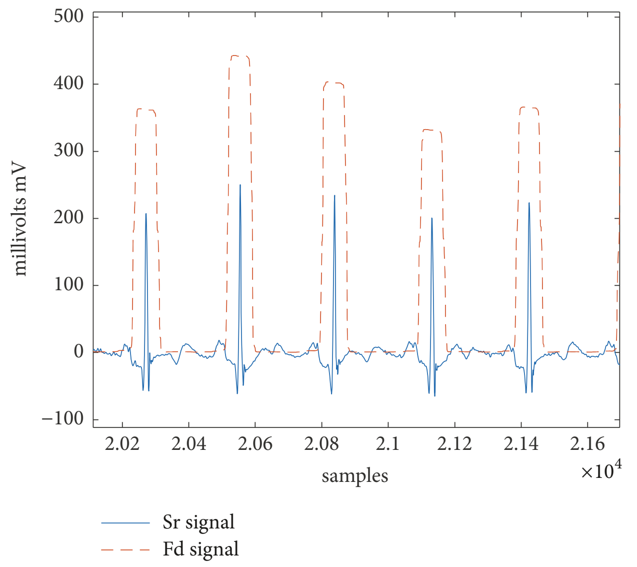

FIGURE 5: Input signals to QRS detector (signal 100 from MITBIH database).

(b) QRS Detector Stage. The stage of the detector required two inputs for its operation, which are the reconstructed signal Sr (containing the ECG signal) and the Fd signal that serves as a basis to find the possible location of the QRS complex (see Figure 5). The signal Fd had to start from the half of $windowsSize$ plus one (see (7)) so that both signals (Sr and Fd) coincided in the location of the peaks. This is because the average filter introduced a delay in the signal.

$$Fd = Fd\left[\left(\frac{windowsSize}{2}\right) + 1\ to\ end\right] \tag{7}$$

The steps followed by the detector are as follows:

(i) To calculate the initial threshold using the Fd signal. The threshold (Th_{signal}) was set to 15% of the maximum peak that was located in the first 0.2 s of the Fd signal.

(ii) To find a peak (Pk_amp_{wave}) that exceeded Th_{signal} in the Fd signal. When this occurred the index (PK_{index}) of the Fd signal was stored.

(iii) The next step was to find an R_{wave} in the signal Sr using PK_{index}. Then, a window of 0.4 s was selected around PK_{index}. The window size was set 0.4 s because this detector is limited to a range of 40 to 150 BPM, where 0.4 s equals 150 BPM. Thus, the maximum peak was searched using (8). The peak obtained in this way was a candidate for an R_{wave}. To determine if the peak found was a true R_{wave}, two criteria were followed:

(a) For the first 10 peaks, the time interval between the current peak and the previous peak had to be between 0.4 s and 1.5 s.

(b) For the rest of peaks (more than 10), an average of the intervals of all previous peaks was taken (mPk_{time}), and if the current interval was greater than 60% of mPk_{time}, without being greater than 1.5 s, the current peak was considered an R_{wave}. With this we avoided confusing the R_{wave} with the P_{wave} or the T_{wave}, sometimes they tend to have similar amplitudes.

$$R_{wave} = \max\_peak\left(Sr\left(Pk_{index} - 0.2s\ to\ Pk_{index} + 0.2s\right)\right) \tag{8}$$

(iv) Finally, the index (R_idx_{wave}) in which the R_{wave} was found was multiplied by the sampling period. In this way the time in which it had occurred was obtained ($Time\_idx_{wave}$) and later stored. The amplitude of the peak of the signal Fd (Pk_amp_{wave}) will be used in the next stage (adaptive threshold).

(c) Adaptive Threshold. In this stage the Th_{signal} changed its value according to the following two conditions:

(i) If an R_{wave} peak was confirmed, the amplitude Pk_amp_{wave} was used to update the Th_{signal} using (9).

(ii) If after a time greater than 1.5 s (corresponding to 40 BPM) starting from the last R_idx_{wave}, no R_{wave} was found, the Th_{signal} was stored as Th_{signal_Temp}, and then it was reduced by 10%. Then, the search from the last R_idx_{wave} was restarted. This modification of Th_{signal} could be repeated up to 3 times. If no R_{wave} was found afterwards, the Th_{signal} retrieved its value from Th_{signal_Temp} and the algorithm continued with the search for more peaks without returning to the previous index. With this process, we could detect R_{wave} waves that had a smaller amplitude, which might not exceed the Th_{signal} because the previous one had a very large amplitude, see Figure 6.

$$Th_{signal} = 0.15 * Pk\_amp_{wave} \tag{9}$$

2.2. HRV Analysis. Table 1 shows the parameters obtained in the different categories of HRV analysis methods. The following subsections explain each category but we insist on the nontrivial ones or the details required to reproduce our results.

2.2.1. Preprocessing for HRV Analysis. This section is devoted to the set of operations required to eliminate ectopic beats. They are known to give erroneous measures in the HRV analysis if they are not eliminated. This preprocessing is performed before the calculation of any of the parameters shown in Table 1. We used the method proposed by [30], following these steps:

Step 1. The linear trend of the IBI vector was removed using (10). For that purpose, the line that best fits IBI was calculated by least squares.

$$IBI_{lineal\_detrend} = IBI - \left(X * C_1 + C_2\right) \tag{10}$$

where X = is a slope and *C_1 and C_2 are the coefficients after resolving the system by Least-Squares.*

Table 1: Parameters of the analysis HRV.

Domain	Description	Symbol	Units∗	Biological mean
Time	Mean between IBI interval	IBI_{mean}	ms	
	Mean heart rate	HR_{mean}	BPM	
	Standard deviation of NN intervals	SDNN	ms	Estimate of overall HRV
	Measure the mean of standard deviation of intervals RR in windows of 5 minutes	iSDNN	ms	
	Measure the standard deviation of the mean of the intervals RR in windows of 5 minutes	SDANN	ms	
	Root mean squared of successive differences	RMSSD	ms	Short term component
	Percentage of intervals that differs more than 50 ms	pNN50	%	
	Standard deviation of the heart rate	HR_{std}	BPM	
	Maximum time between IBI interval	Max	ms	
	Minimum time between IBI interval	Min	ms	
Frequency	Low frequency power	LF	ms^2	Sympathetic nervous System activity
	High frequency power	HF	ms^2	parasympathetic nervous System activity
	Ratio between low and high frequency	LF/HF		Sympathovagal balance
	Low frequency normalized	LFn	n.u	
	High frequency normalized	HFn	n.u	
Non linear	Poincare	SD1, SD2 SD_{ratio}	ms	Term variability
Geometric (time domain)	Triangular index	RRTrin		Estimate of overall HRV
	Triangular Interpolation index	TINN		

∗ BPM(beats per minute), ms (milliseconds), ms^2(milliseconds squared), n.u. (normalized units), %(percent).

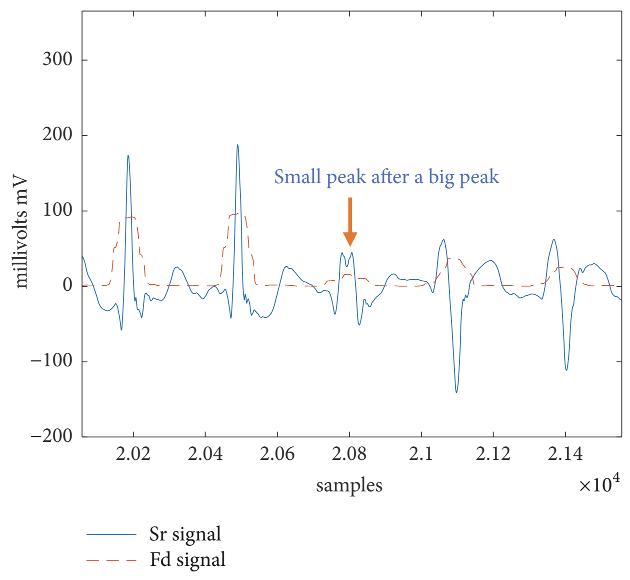

Figure 6: Portion of the signal 104 of the MITBIH database showing large differences in amplitude.

Step 2. The standard deviation (STD_{IBI}) and the mean (Mnd_{IBI}) of $IBI_{lineal\_detrend}$ were calculated.

Step 3. A threshold ($trsh_{sd}$) was set in order to find the ectopic beats. In this case it was equal to three times the STD_{IBI}:

$$trsh_{sd} = 3 * STD_{IBI} \tag{11}$$

Step 4. Finally, to find an ectopic beat, the mean Mnd_{IBI} was subtracted from the absolute value of the $IBI[x]_{lineal\_detrend}$. Ectopic beats were those beats in which the result was higher than $trsh_{sd}$; in those cases the sample $IBI[x]$ was substituted by the mean of the five preceding samples and the five following samples of it.

2.2.2. Analysis of HRV in the Time Domain. After the ectopic beats were removed, the vector IBI was transformed from seconds to milliseconds ($mIBI$) and the statistical parameters shown in Table 1 were calculated, which correspond to widely known statistical measurements (means, standard deviations).

2.2.3. Analysis of HRV in the Frequency Domain. The analysis in the frequency domain had three stages: (a) preprocessing: in this stage the signal went through a series of filters to eliminate its offset; (b) interpolation: the signal without offset was interpolated at 4 Hz; (c) spectral analysis: the spectral density (PSD) was calculated using the Welch method.

Preprocessing

Step 1. It began by smoothing the *IBI* vector. Some authors have shown that the smoothness prior filter is usually very effective in bioelectric signals like the ECG [37, 38]. It is even used in commercial software such as Kubios for HRV analysis [39]. However, its algorithm requires many resources in RAM memory to be implemented in an MCU. Therefore, in this paper we opted to do the filtering using wavelets. The filtering with the wavelets was done by eliminating the coefficients of the details. In this case, a wavelet *db*5 with four decomposition levels was used, in which after having eliminated all the *CD* coefficients, the signal was reconstructed (IBI_{wave}). Finally using (12) the signal was smoothed. The purpose of this filtering process was to remove any disturbance in low frequency that affects RR intervals.

$$IBI_{smooth}[x] = (IBI[x] - IBI_{wave}[x]) \tag{12}$$

where $x = 0, 1, \dots N$ and $N = length(IBI)$.

Step 2. Calculate the temporary vector (t), which will be used to perform the interpolation. The temporary vector is the cumulative sum of the *IBI* vector minus the first value of itself (13).

$$t = \text{cumulative}_{\text{sum}}(IBI) - IBI[0] \tag{13}$$

Interpolation. The vector IBI_{smooth} was passed to ms ($mIBI_{smooth}$). Then, the interpolation was carried out using the cubic spline algorithm at 4 Hz (14). Once $mIBI_{smooth}$ was interpolated, the average was subtracted (15).

$$mIBI_{interpolate} = spline(t, mIBI_{smooth}) \tag{14}$$

$$mIBI_{toPSD} = mIBI_{interpolate} - mean(mIBI_{interpolate}) \tag{15}$$

Calculation of the Spectral Density. To calculate the PSD with the Welch method, the algorithm implemented used the fast Fourier transform (FFT) with a window of 256 points ($Data_{window}$) and overlapping of 128 points. Before applying the FFT to each window, the data were smoothed by multiplying them by a Hamming function ($Hamming_{window}$) of the same width as $Data_{window}$. In this way we avoided abrupt discontinuities at the beginning and end of each window.

Obtaining the Parameters in the Frequency Domain. After computing the PSD (PSD_{signal}) of the $mIBI_{toPSD}$ signal, the area under the curve was calculated for each band of frequencies associated with the HRV: the *VLF* band (0 to 0.04 Hz), the *LF* band (0.04 to 0.15 Hz), and the *HF* band (0.15 to 0.4 Hz). Once the areas for each band were calculated, the *LF*/*HF* ratio was calculated by dividing the total area of *LF* by the total area of *HF* (16). The normalized *LF* and *HF* were calculated with (17) and (18).

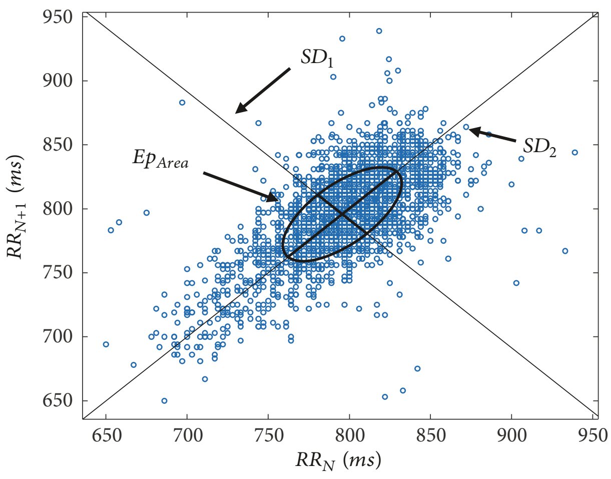

FIGURE 7: Representation of the Poincaré analysis.

$$LF/HF = \frac{LF}{HF} \tag{16}$$

$$LF_{normalized} = \frac{LF}{LF + HF} \tag{17}$$

$$HF_{normalized} = \frac{HF}{LF + HF} \tag{18}$$

2.2.4. Poincaré Analysis. The Poincaré analysis is a graphical method that evaluates the dynamics of the HRV from the current (RR_N) and the next (RR_{N+1}) RR intervals:

$$RR_N = (mIBI[0], mIBI[1], \dots, mIBI[N-1]) \tag{19}$$

$$RR_{N+1} = (mIBI[1], mIBI[2], \dots, mIBI[N]) \tag{20}$$

where $N = length(mIBI)$.

To perform a quantitative analysis and evaluate the HRV, an ellipse was fit to the data (see Figure 7). The width of the ellipse is known as the standard deviation 1 (SD_1) and the length as the standard deviation 2 (SD_2). From SD_1 and SD_2 the area of the ellipse was calculated (Ep_{Area}) [6]. It is said that SD_1 represents the HRV in short times (short term) and is correlated with the SNA, while SD_2 represent long periods (long term) and is correlated with SNS [40]. Due to the correlation between the standard deviation of the RR interval difference ($dIBI$) (see (21)) and SD_1, SD_1 was calculated using (22)-(24). From the *SDNN* parameter and SD_{dIBI} (see (23)) SD_2 was calculated with (25), the area Ep_{Area} with (26), and finally the ratio SD_{ratio} with (27) [41].

The Poincaré plot represents the healthy case by a large ellipse area and small for critical diseases. To perform the analysis of Poincaré periods between 5 to 20 minutes are recommended [6]. On the other hand [42, 43] observed that a lowest SD_{ratio} value is present in healthy subjects. In the same way a small value in SD_1 for diseased subjects indicates weakening of parasympathetic regulation by health disorder.

As SD_2 decreases, the SNS activity increases. Figure 7 shows an arrhythmia case from the signal 215 (MITBIH), in which it can observed that the most of the samples are focused in the center of the plot and represent a small ellipse area.

$$dIBI[x] = mIBI[x+1] - mIBI[x] \tag{21}$$

$$dIBI_{mean} = \frac{\sum_{x=0}^{N}(dIBI[x])}{N} \tag{22}$$

$$\mathrm{SD}_{dIBI} = \sqrt{\frac{\sum_{x=0}^{N}\left|dIBI[x] - dIBI_{mean}\right|^2}{N}} \tag{23}$$

$$SD_1 = \sqrt{0.5 * \mathrm{SD}_{dIBI}{}^2} \tag{24}$$

$$SD_2 = \sqrt{(2 * SDNN^2) - (0.5 * \mathrm{SD}_{dIBI}{}^2)} \tag{25}$$

$$Ep_{Area} = \pi * SD_1 * SD_2 \tag{26}$$

where $x = 0, 1, \ldots N$ and $N = length(\mathrm{d}IBI)$.

$$SD_{ratio} = \frac{SD_1}{SD_2} \tag{27}$$

2.2.5. Triangular Geometric Analysis. Triangular geometric analysis is usually considered as part of the analysis in the time domain. However, it deserves further explanation since it is more complex than the rest. This analysis is done by calculating the histogram of the vector *mIBI*, through which two parameters can be calculated, the triangular index (RR_{Trin}) and the index of triangular interpolation of the base of the histogram (*TINN*). The RR_{Trin} is equal to the total number of RR intervals divided by the maximum value of the histogram (Val_{max}) (28). This index gives an overall estimate of the HRV. The width of the bins used in this analysis was set to 7.8125 ms. The *TINN* index gives a value on the distribution of the density of all the RR intervals as the base of a triangle and it is usually calculated by means of a least squares estimate. Thus $Area_{tinn}$ (see (29)) was minimized and the limiting points N and M were found. Then, *TINN* was calculated with (30); see Figure 8 [44].

$$RR_{Trin} = \frac{N}{Val_{max}} \tag{28}$$

$$Area_{tinn} = \int_0^{\infty} (D(t) - q(t))^2\, dt \tag{29}$$

where D *is histogram output,* q *defines the base of triangle that fits histogram and its cero for* $t \le N; t \ge M$*, and* $q(X) = Y$.

$$TINN = M - N \tag{30}$$

where M and N are the limits of $Area_{tinn}$.

2.3. Parameters and Databases Used to Evaluate the Performance of QRS Detector. In order to compare the performance of the QRS detector with those found in the literature, we used the sensitivity parameter (*SE*) (see (31)), the positive predictive value (*PP*) (see (32)), and the percentage of the prediction error rate (*DER*) (see (33)). *DER* was also used to evaluate the accuracy of the algorithm.

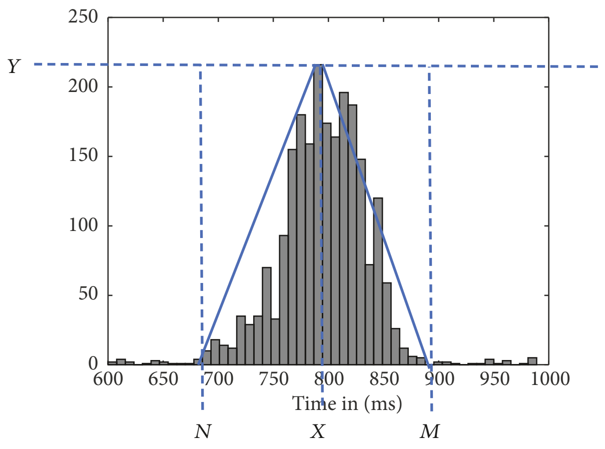

FIGURE 8: Points of the histogram used to calculate the indexes of the triangular geometric analysis.

$$SE = \frac{TP}{TP + FN} * 100 \tag{31}$$

$$PP = \frac{TP}{TP + FP} * 100 \tag{32}$$

$$DER = \frac{FP + FN}{TP} * 100 \tag{33}$$

where TP are the true positives in detecting the R_{wave}, FN are the R_{wave} peaks that have not been detected, and FP are the peaks that have been mistakenly detected as R_{wave}.

In order to validate and compare our proposed algorithm for QRS complex detection, the MIT-BIT arrhythmia database (MITBIH) and the normal sinus rhythm database (NSRDB), both available online, were used [45]. Then, the results obtained were compared with other algorithms to see their effectiveness. The MITBIH database consists of 48 recordings. Each recording has two signals extracted from half hour of a 24-hour recording, which have been sampled at 360 Hz and belong to 47 patients in total. The NSRDB database contains 18 signals with a duration of 130 minutes sampled at 128 Hz and belong to healthy adults aged 20 to 50 years.

2.4. Implementation on an MCU. The MCU used for the implementation of the QRS detector and the HRV analysis was an STM32F407ZET6 MCU. Two features can be highlighted for the present study: it has a set of instructions for digital signal processing (DSP) and a floating point processing unit (FPU), which make it an MCU capable of performing advanced digital processing calculations. Likewise, it has a flash program memory of 512 Kbyte, 192 Kbyte of SRAM memory and a working frequency of 168 MHz.

The DSP libraries provided by CMSIS (Cortex Microcontroller Software Interface Standard) were used for the signal

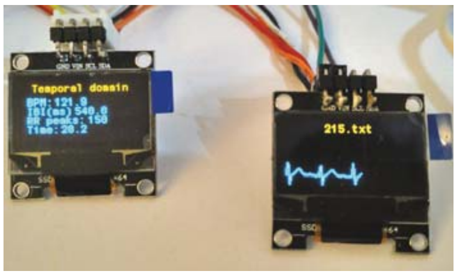

Figure 9: Analysis of the HRV in real time.

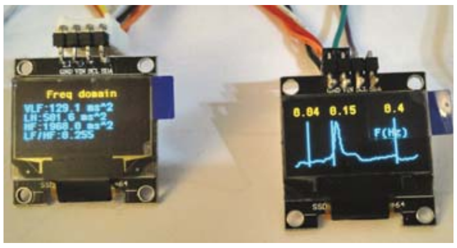

Figure 10: Analysis of the HRV in frequency domain.

processing and the open source wavelib library [46] was used for the wavelet implementation. The wavelib library had to be modified and optimized in terms of RAM resources to be used in an MCU.

The MCU accessed the MITBIH and NSRDB databases through a microSD card memory, in which each recording was stored in a separate text data file. They were processed in real time in blocks of 1024 data. The block size was chosen for two reasons: the first one because this number of samples is enough to apply the DWT with 7 levels of decomposition; and the second because the amount of RAM required is constant (25 kB) and did not compromise the rest of calculations. It is worth explaining that for an application in real life, the MCU could acquire data in real time from two external sources, selecting only one at a time. The first one would involve a serial port through the HC-05 Bluetooth module at a baud rate of 230400. The second one could use the AD8232 module using the digital analog converter (ADC) with a sampling frequency of 500 Hz. In both cases interrupts could be used, being activated whenever there is a new data available. Thus even while the signal is being processed, the MCU can continue to fill new buffers for further processing. To display the data locally, two LCD screens SSD1306 of 128x64 pixels were incorporated using the I2C protocol. An example of the information displayed on LCD are show in the Figures 9 and 10. Figure 9 shows the data displayed in real time, while Figure 10 shows the analysis in the frequency domain for the signal 215 (MITBIH database).

To reduce RAM consumption when processing the signal, the following programming techniques were applied:

(i) Avoiding the creations of arrays or matrices within functions: it is better to use loops and perform the corresponding calculations in each iteration.

```
typedef  union{
     unsigned uint8_t AllFlags ;
     struct{
       unsigned flag_0:1;
       unsigned flag_1:1;
       unsigned flag_2:1;
       unsigned flag_3:1;
       unsigned flag_4:1;
       unsigned flag_5:1;
       unsigned flag_6:1;
       unsigned flag_7:1;
        };
}DeviceStatusFlags ;
DeviceStatusFlags Flags;
Used Bytes in RAM: 1

       uint8_t flag_0;
       uint8_t flag_1;
       uint8_t flag_2;
       uint8_t flag_3;
       uint8_t flag_4;
       uint8_t flag_5;
       uint8_t flag_6;
       uint8_t flag_7;
Used Bytes in RAM: 8
```

Code 1: Efficient use of RAM memory using structures and unions.

(ii) If possible, using the input variables as output variables in functions, so that new variables are not needed.

(iii) Making use of structures and unions to pack data and take control of system flags: in this way the amount of RAM required is fully utilized. For example, in Code 1 you can see that less RAM is required using structures and unions for the same number of flags.

For example, in Code 2 it is assumed that the input variable y has a size of 2000 data. In Matlab 64047 bytes is required to perform the same calculation (remove linear trend), since it does so by means of arrays, while the algorithm of Code 2 occupied only 48 bytes in the MCU (12 $float$ variables and an *int* variable that require 4 bytes each one) since the result was stored in the same input variable.

The algorithms optimized following the ideas shown in Code 2 were the average filter, the peak detection and the threshold updating in the QRS detector, the calculation of *iSDNN* and *SDANN*, the application of DWT (see (10)), the linear trend removal (see (12)), and the calculation of the PSD (see (15)).

3. Results

The results of Table 2 correspond to the performance of the QRS detector when evaluated with the MITBIH database. DB is the number of detected beats and TB is the total number of beats in each ECG signal. It can be observed that 109086 pulses of a total of 109494 have been detected: 108713

```
void arm_lineal_detrend_f32(float32_t *y, int size){
   float32_t a,b,c,d,e,f,x1,x2,C1,C2,determinant;
   int i;
    a=0;b=0;c=0;d=0;e=0;f=0;
    x2=1;
    determinant=0;
    for (i=0; i<size; i++){
            x1=((float32_t)i+1)/(float32_t)size;     // create a slope
            a=a+(x1 * x1);
            b=b+x1;
            c=c+x1;
            d=d+x2;
            e=e+(x1 * y[i]);
            f=f+(x2 * y[i]);
   }
   determinant = a * d - b * c;
   if ( isnan(determinant) || isinf(determinant)){
         return ;
   }else{
           C1 = (e * d - b * f)/determinant;        // resolve the System
           C2 = (a * f - e * c)/determinant;        // to find the coefficients C1 and C2
           for (i=0; i<size; i++){
                    x1=((float32_t)i+1)/(float32_t)size;
                    y[i]=y[i]-(x1 * C1+C2);          // remove lineal trend
           }
   }
}
```

Code 2: C-code to eliminate the linear trend.

Table 2: Parameters used to evaluate the QRS detector.

Record No.	TB	DB	TP	FP	FN	SE%	PP%	DER%
Total	109494	109086	108713	373	781	99.30	99.61	1.12

were true positives, 781 were false negative, and 373 were false positives. In addition, the QRS detector showed a *SE* of 99.30% with a positive prediction of 99.61% and an error rate in the detection of pulses of 1.12%

Table 3 shows the results obtained between the different algorithms found in the literature and ours, using the MIT-BIH database. The results we have obtained are a little worse. However, the difference between the algorithm with better *SE* [34] and ours is only 0.57%. In *PP*, the difference with respect to [35] is 0.30% and in *DER*, the difference with respect to [36] is only 0.84%.

Table 4 shows the comparison between the results obtained by our algorithm and those obtained by [32] using the *NSRDB* database. In this comparison [32] has better results than us in *SE* and *PP* by 0.01% while *DER* is only 0.0002% better.

The time required by the MCU for the HRV analysis will vary according to the size of the IBI vector. Table 5 shows the execution times that were required to analyze the signal 215 of the MITBIH database. This particular file was chosen because it contains more QRS complex than the rest of the signals (3358 found by our algorithm).

On the other hand, the execution times for the QRS detector will always be the same (47 ms for detection and 5 ms for saving in a microSD the temporary location of the R_{wave} found) because the length of the input vector has a constant size (1024 samples).

The time required to fill the 1024 data buffer is $1024 * (1/fs)$, where fs is the sampling frequency. In the case of the MITBIH fs is 360 Hz, so it would take 2.84 s to fill it. In the case of the NSRDB fs is 128 Hz, so that it would take 8 s. In both cases the detection of the QRS complex can be done in real time since it only takes 52 ms. The detector may operate in real time up to a maximum sampling frequency of 19.5 kHz. However, some studies show that 500 Hz is a very reliable frequency for HRV analysis [47]. Our system can easily achieve real time for this recommended frequency.

Although the times required for the analysis of the HRV vary depending on the size of *IBI*, in Table 5 it can be seen that the smoothing of the signal using wavelets and the analysis in the frequency domain required more time for its execution. On the other side, the Poincaré analysis was the fastest. The time required for the complete HRV analysis of signal 215 (MITBIH database) was 4.401 s.

Table 3: Comparison of the results obtained using different algorithm in the MITBIH database.

	DB	TP	FP	FN	SE%	PP%	DER%
J.P Martínez et al. [31]	109428	109208	153	220	99.80	99.86	0.34
J. Pan & Tompkins [11]	109809	109532	507	277	99.75	99.54	0.71
Gutiérrez-Rivas et al. [32]	-	109447	289	502	99.54	99.73	-
Chouakri et al. [33]	109488	108043	3068	1446	98.68	97.24	4.12
Santanu et al. [34]	109666	109351	315	144	99.87	99.69	0.42
Moody & Mark [35]	109428	107567	94	1861	98.30	99.91	1.79
Merah et al.[36]	109494	109316	126	178	99.84	99.88	0.28
Proposed algorithm	109086	108713	373	781	99.30	99.61	1.12

Table 4: Comparison of the results using the NSRDB database.

Algorithm	TB	TP	FP	FN	SE%	PP%	DER%
Gutiérrez-Rivas et al. [32]	192389	192325	64	17	99.96	99.99	0.0004
Proposed algorithm	192389	192297	92	25	99.95	99.98	0.0006

Refreshing the LCD screens took 50 ms and it was done while the buffer was filling, so that the update times of the screens had no influence on the calculations of the HRV.

The execution time on a regular PC with the processing performed in Matlab is also shown in Table 5 for comparison. It is lower than in the MCU implementation because of the powerful processor. The idea is not to be competitive with a PC server in this aspect, but to avoid transmitting data to a server in the cloud, which would be a great burden for the network. In addition, this would imply a higher power consumption for the sensor node. For instance, the proposed system has these power consumption contributions: 4.2 mA for the MCU in sleep mode, 4.9 mA while processing a signal from a micro SD card, 7.5 mA while processing signals from AD8232 or 16.9 mA while processing signals from Bluetooth (HC-05 module). Thus it is clear that avoiding communication is of great interest.

4. Conclusions and Future Work

The results obtained with the QRS detector that we have proposed based on wavelets and with automatic selection of the coefficients of the details have been higher than 99% in *SE* and *PP* using the MITBIH database. They are worse than previous studies but the difference does not exceed 0.9% in any of the parameters used in that comparison (*SE*, *PP*, *DER*). On the other hand, using the NSRDB database the results have been better ($SE = 99.95\%, PP = 99.98\%$, and $DER = 0.0006\%$). In comparison with [32] SE and PP have been only 0.01% lower.

The QRS detection algorithm was followed by an HRV analysis. In both stages, several algorithms had to be optimized for the implementation on an MCU. Thanks to the efficient use of RAM, it has been possible to develop the whole HRV analysis as a standalone application embedded on an MCU ARM.

With the execution times shown in Table 5, it has been shown that the QRS detector is capable of running in real time for the most common frequencies used in ECGs. Likewise, given that some parameters such as IBI_{mean}, $SDNN$, $RMSSD$, and HR_{mean} require very little time for their execution and that they are in the time domain, they could be displayed by sending them via Bluetooth or Wifi to a mobile application. In this way, an HRV analysis would be done in real time. In addition, the energy consumption is low, so it can facilitate its integration on portable devices.

The time required for the complete analysis of the HRV will be variable (depending on the amount of data), but it will be in the order of seconds for practical purposes, making it a viable application.

Our system could also help in applications like the one shown in [25], in which HRV and location are measured to evaluate wellness and recommend a place to live accordingly. All the RR intervals are sent to the user's mobile phone, and from it to a server in which a time domain parameter of HRV is calculated. The system could be improved if the wearable device measured itself HRV, avoiding the battery consuming process of the communication with the mobile phone. Besides, our system can calculate all the HRV parameters and could improve the determination of the user's wellness. Another application of our system could be the integration into the system to monitor asthma disease presented in [26], in which the heart rate is also a key component.

This system developed on an ARM MCU could be complemented by developing a mobile application that displays the results of the HRV analysis in a more adequate way to the user, showing comparisons or statistics with respect to previous analyses. In this way the user would get better control over his/her HRV.

As future work, several lines are devised. In the first one the QRS detector will be improved to obtain better results than those seen in Tables 3 and 4. The second line will be devoted to the optimization of the proposed system code to integrate it in the most optimal way into an *IoT* network. It can be tested using Wifi, ZigBee, or Bluetooth networks for short ranges. For instance, it could receive information from a network edge and then send back to a server only the HRV results to be displayed. This will help to improve

Table 5: Time required for HRV analysis of the signal 215 (MITBIH database).

Main function	Sub functions (ms)	Execution time (ms) in MCU	Execution time (ms) in Intel i7 (MatLab)
Detector QRS		47	39
	DWT perform (37)		
	First difference (<1)		
	Squared signal (<3)		
	Filter average (<3)		
	Find peaks and update threshold (<3)		
Save data to SD		5	-
Preprocessing	Remove Artifacts	3	1
Time Domain HRV		30	9
	IBI_{mean} (<1)		
	HR_{mean} (1)		
	$SDNN$ (<1)		
	$iSDNN$ (12)		
	$SDANN$ (11)		
	$RMSSD$ (<1)		
	$pNN50, NN50$ (1)		
	HR_{std} (1)		
	Max (<1)		
	Min (<1)		
Geometric triangular HRV		89	2
	RR_{Trin} (17)		
	$TINN$ (62)		
Poincare HRV		1	1
	SD_1, SD_2 and Ep_{Area} (1)		
Refresh LCD		50	-
Frequency Domain HRV		4176	163
	IBI_{smooth} (2084)		
	t (<1)		
	$mIBI_{interpolate}$ (1331)		
	PSD_{signal} (302)		
	$VLF, LF, HF, LF/HF, LF_{normalized}$ and $HF_{normalized}$ (458)		

mHealth and eHealth services using systems embedded in microcontrollers. On the other hand, we will work to make the proposed device as small as possible, similar to smart band. In this way, it will be able to track heart rate variability during the daily activity.

Data Availability

The data supporting this study are from previously reported studies and datasets, which have been cited or are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors would like to thank the “Fondo Social Europeo”, the “Diputación General de Aragón” (reference group EduQTech: T49_17R). Victor H. Rodriguez acknowledges a grant from “CONACYT-Gobierno del Estado de Durango, México 330795/386043”.

References

[1] G. Vega-Martínez, C. Alvarado-Serrano, and L. Leija-Salas, "Wavelet packet based algorithm for QRS region detection and R/S wave identification," in *Proceedings of the 12th International Conference on Electrical Engineering, Computing Science and Automatic Control, CCE 2015*, Mexico, October 2015.

[2] R. Saktheeswari and K. Adalarasu, "Survey on signal processing techniques for diagnoising cardiovascular diseases," in *Proceedings of the 2017 4th International Conference on Innovations in Information, Embedded and Communication Systems*, pp. 1–4, Coimbatore, March 2017.

[3] H. J. Baek and J. Shin, "Effect of Missing Inter-Beat Interval Data on Heart Rate Variability Analysis Using Wrist-Worn Wearables," *Journal of Medical Systems*, vol. 41, no. 10, pp. 1–9, 2017.

[4] F. Shaffer, R. McCraty, and C. L. Zerr, "A healthy heart is not a metronome: an integrative review of the heart's anatomy and heart rate variability," *Frontiers in Psychology*, vol. 5, 2014.

[5] E. K. Kerut, K. W. Swan, F. To, T. D. Giles, and P. J. Kadowitz, "Respiratory sigh associated transient autonomic changes detected with a continuous wavelet method of heart rate variability analysis," *Biomedical Signal Processing and Control*, vol. 38, pp. 143–147, 2017.

[6] M. B. Tayel and E. I. AlSaba, "Poincaré plot for heart rate variability," *International Journal of Biomedical and Biological Engineering*, vol. 9, pp. 708–711, 2015.

[7] C. Gentili, S. Messerotti Benvenuti, D. Palomba, A. Greco, E. P. Scilingo, and G. Valenza, "Assessing mood symptoms through heartbeat dynamics: An HRV study on cardiosurgical patients," *Journal of Psychiatric Research*, vol. 95, pp. 179–188, 2017.

[8] Y. Kubota, L. Y. Chen, E. A. Whitsel, and A. R. Folsom, "Heart rate variability and lifetime risk of cardiovascular disease: the Atherosclerosis Risk in Communities Study," *Annals of Epidemiology*, vol. 27, no. 10, pp. 619–625, 2017.

[9] I. García-Magariño and I. Plaza, "ABS-MindHeart: An agent based simulator of the influence of mindfulness programs on heart rate variability," *Journal of Computational Science*, vol. 19, pp. 11–20, 2017.

[10] Butta. Singh and Singh. Manjit, "ECG artifacts and poincaré plot based heart rate variability," *International Journal of Latest Trends in Engineering and Technology*, vol. 5, pp. 304–309, 2015.

[11] J. Pan and W. J. Tompkins, "A Real-Time QRS Detection Algorithm," *IEEE Transactions on Biomedical Engineering*, vol. BME-32, no. 3, pp. 230–236, 1985.

[12] M. Rakshit and S. Das, "An efficient wavelet-based automated R-peaks detection method using Hilbert transform," *Biocybernetics and Biomedical Engineering*, vol. 37, no. 3, pp. 566–577, 2017.

[13] S. Jain, M. K. Ahirwal, A. Kumar, V. Bajaj, and G. K. Singh, "QRS detection using adaptive filters: A comparative study," *ISA Transactions*, vol. 66, pp. 362–375, 2017.

[14] D. Lai, F. Zhang, and C. Wang, "A real-time QRS complex detection algorithm based on differential threshold method," in *Proceedings of the 2015 IEEE International Conference on Digital Signal Processing (DSP)*, pp. 129–133, Singapore, 2015.

[15] N. Ravanshad and H. Rezaee-Dehsorkh, "An event-based ECG-monitoring and QRS-detection system based on level-crossing sampling," in *Proceedings of the 2017 Iranian Conference on Electrical Engineering (ICEE)*, pp. 302–307, Tehran, Iran, 2017.

[16] M. Belkadi and A. Daamouche, "An improved QRS detection method using Hidden Markov Models," in *Proceedings of the 2017 6th International Conference on Systems and Control (ICSC)*, pp. 81–84, Batna, Algeria, 2017.

[17] Z. Zidelmal, A. Amirou, D. Ould-Abdeslam, A. Moukadem, and A. Dieterlen, "QRS detection using S-Transform and Shannon energy," *Computer Methods and Programs in Biomedicine*, vol. 116, no. 1, pp. 1–9, 2014.

[18] F. Firouzi, A. M. Rahmani, K. Mankodiya et al., "Internet-of-Things and big data for smarter healthcare: From device to architecture, applications and analytics," *Future Generation Computer Systems*, vol. 78, pp. 583–586, 2018.

[19] Y. Ai, M. Peng, and K. Zhang, "Edge computing technologies for Internet of Things: a primer," *Digital Communications and Networks*, vol. 4, no. 2, pp. 77–86, 2018.

[20] F. González-Landero, I. García-Magariño, R. Lacuesta, and J. Lloret, "Green Communication for Tracking Heart Rate with Smartbands , Sensors," in *Sensors*, vol. 18, p. 2652, Basel, Switzerland, 2018.

[21] P. Kochovski and V. Stankovski, "Supporting smart construction with dependable edge computing infrastructures and applications," *Automation in Construction*, vol. 85, pp. 182–192, 2018.

[22] A. Ahmed and E. Ahmed, "A survey on mobile edge computing," in *Proceedings of the 10th International Conference on Intelligent Systems and Control (ISCO)*, IEEE, Coimbatore, India, 2016.

[23] R. Craciunescu, A. Mihovska, M. Mihaylov, S. Kyriazakos, R. Prasad, and S. Halunga, "Implementation of Fog computing for reliable E-health applications," in *Proceedings of the 2015 49th Asilomar Conference on Signals, Systems and Computers*, pp. 459–463, IEEE, Pacific Grove, CA, USA, 2015.

[24] M. Quintana-Suárez, D. Sánchez-Rodríguez, I. Alonso-González, and J. Alonso-Hernández, "A low cost wireless acoustic sensor for ambient assisted living systems," *Applied Sciences*, vol. 7, no. 9, p. 877, 2017.

[25] R. Lacuesta, L. Garcia, I. Garcia-Magarino, and J. Lloret, "System to Recommend the Best Place to Live Based on Wellness State of the User Employing the Heart Rate Variability," *IEEE Access*, vol. 5, pp. 10594–10604, 2017.

[26] S. Sendra, L. Parra, J. Lloret, and J. Tomás, "Smart system for children's chronic illness monitoring," *Information Fusion*, vol. 40, pp. 76–86, 2018.

[27] V. H. Rodriguez, C. Medrano, and I. Plaza, "A Real-Time QRS Complex Detector Based on Discrete Wavelet Transform and Adaptive Threshold as Standalone Application on ARM Microcontrollers," in *Proceedings of the 2018 International Conference on Biomedical Engineering and Applications (ICBEA)*, pp. 1–6, IEEE, Madeira, Portugal, 2018.

[28] G. Strang, "Wavelets and dilation equations: a brief introduction," *SIAM Review*, vol. 31, no. 4, pp. 614–627, 1989.

[29] A. A. Fedotov, A. S. Akulova, and S. A. Akulov, "Applicability of multiresolution wavelet analysis for QRS-waves detection," in *Proceedings of the 2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 3793–3796, IEEE, Orlando, FL, USA, 2016.

[30] A. E. Aubert, D. Ramaekers, F. Beckers et al., "The analysis of heart rate variability in unrestrained rats. Validation of method and results," *Computer Methods and Programs in Biomedicine*, vol. 60, no. 3, pp. 197–213, 1999.

[31] J. P. Martínez, R. Almeida, S. Olmos, A. P. Rocha, and P. Laguna, "A wavelet-based ECG delineator: evaluation on standard databases," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 4, pp. 570–581, 2004.

[32] R. Gutiérrez-Rivas, J. J. García, W. P. Marnane, and Á. Hernández, "Novel Real-Time Low-Complexity QRS Complex Detector Based on Adaptive Thresholding," *IEEE Sensors Journal*, vol. 15, no. 10, pp. 6036–6043, 2015.

[33] S. A. Chouakri, F. Bereksi-Reguig, and A. Taleb-Ahmed, "QRS complex detection based on multi wavelet packet decomposition," *Applied Mathematics and Computation*, vol. 217, no. 23, pp. 9508–9525, 2011.

[34] S. Sahoo, B. Kanungo, S. Behera, and S. Sabut, "Multiresolution wavelet transform based feature extraction and ECG classification to detect cardiac abnormalities," *Measurement*, vol. 108, pp. 55–66, 2017.

[35] G. B. Moody and R. G. Mark, "Development and evaluation of a 2-lead ECG analysis program," in *Computers in Cardiology*, pp. 39–44, IEEE Computer Society Press, Los Alamitos, CA, USA, 1982.

[36] M. Merah, T. A. Abdelmalik, and B. H. Larbi, "R-peaks detection based on stationary wavelet transform," *Computer Methods and Programs in Biomedicine*, vol. 121, no. 3, pp. 149–160, 2015.

[37] F. Zhang, S. Chen, H. Zhang, X. Zhang, and G. Li, "Bioelectric signal detrending using smoothness prior approach," *Medical Engineering & Physics*, vol. 36, no. 8, pp. 1007–1013, 2014.

[38] R. Sameni, "Online filtering using piecewise smoothness priors: Application to normal and abnormal electrocardiogram denoising," *Signal Processing*, vol. 133, pp. 52–63, 2017.

[39] M. P. Tarvainen, J.-P. Niskanen, J. A. Lipponen, P. O. Ranta-aho, and P. A. Karjalainen, "Kubios HRV—heart rate variability analysis software," *Computer Methods and Programs in Biomedicine*, vol. 113, no. 1, pp. 210–220, 2014.

[40] Hugo. Cerda-Kohler and Carlos. Henríquez-Olguín, "Variabilidad del ritmo cardiaco y ejercicio físico," *Revista Horizonte Ciencias de la Actividad Física*, vol. 5, pp. 140–158, 2014.

[41] M. P. Tulppo, T. H. Makikallio, T. E. Takala, T. Seppanen, and H. V. Huikuri, "Quantitative beat-to-beat analysis of heart rate dynamics during exercise," *American Journal of Physiology-Heart and Circulatory Physiology*, vol. 271, no. 1, pp. H244–H252, 1996.

[42] A. K. Golińska, "Poincare plots in analysis of selected biomedical signals," *Studies in Logic, Grammar and Rhetoric*, vol. 35, no. 48, pp. 117–127, 2013.

[43] Kumar. Sobhendu Ghatak and Subhra. Aditya, "Poincare parameters and principal component analysis of Heart rate variability of subjects with health disorder," *Medical Physics*, pp. 1–12, 2018.

[44] Task Force of the European Society of Cardiology the North American Society of Pacing Electrophysiology, "Heart rate variability standards of measurement, physiological interpretation, and clinical use," *Circulation*, vol. 93, no. 5, pp. 1043–1065, 1996.

[45] Physionet, "PhysioBank ATM," https://physionet.org/cgi-bin/atm/ATM,.

[46] Rafat. Hussain, "Wavelib," https://github.com/rafat/wavelib/wiki,.

[47] M. Baumert, M. Schmidt, S. Zaunseder, and A. Porta, "Effects of ECG sampling rate on QT interval variability measurement," *Biomedical Signal Processing and Control*, vol. 25, pp. 159–164, 2016.