QRS Detection Algorithm

Daniel Rodas Bautista
115121569

February 26, 2016

1 Introduction

The aim for this project is to analyse and implement a heart rate monitor on an FPGA board. The most important part for monitoring a heart rate is to correctly detect the QRS complex in the ECG heart signal [4]. This first report focuses on the software implementation of an algorithm for the detection of the QRS complex.

Two algorithms were analysed and implemented for reasons that will be explained throughout this report. Initially I worked with the famous Pan Tompkins algorithm since it is the foundation for many other algorithms. Implementing it on software is quite straightforward but since the final goal of this project is to implement it on hardware I ended deciding that it was not fit for this.

Both the algorithms and the final fixed point implementation were tested using the MIT-DIB database [1] available on the physionet website [2]. One minute of samples was used in each but keep in mind that the plots in this report show only 15 seconds of samples for the sake for clarity.

2 Pan Tompkins Algorithm

2.1 Description of the algorithm

The algorithm is divided into two main parts [5]. First there is a preprocessing stage followed by the decision stage. The preprocessing stage is made up of several linear filters and one non-linear filter. In principle the final signal of this stage corresponds to the energy of the initial signal. First there is a band-path filter to get rid of the noise. This is followed by a differentiator which accentuates the characteristic slope of the QRS complex. This signal is then squared (this is the non-linear filter) after which a moving window average, i.e. an integrator, is applied.

The actual detection stage uses a double threshold technique which is applied both to the output of the integrator as well as the output of the differentiator. A QRS complex is said to be detected only when it has been detected in both signals. Moreover the algorithm keeps track of the last eight RR intervals, i.e.

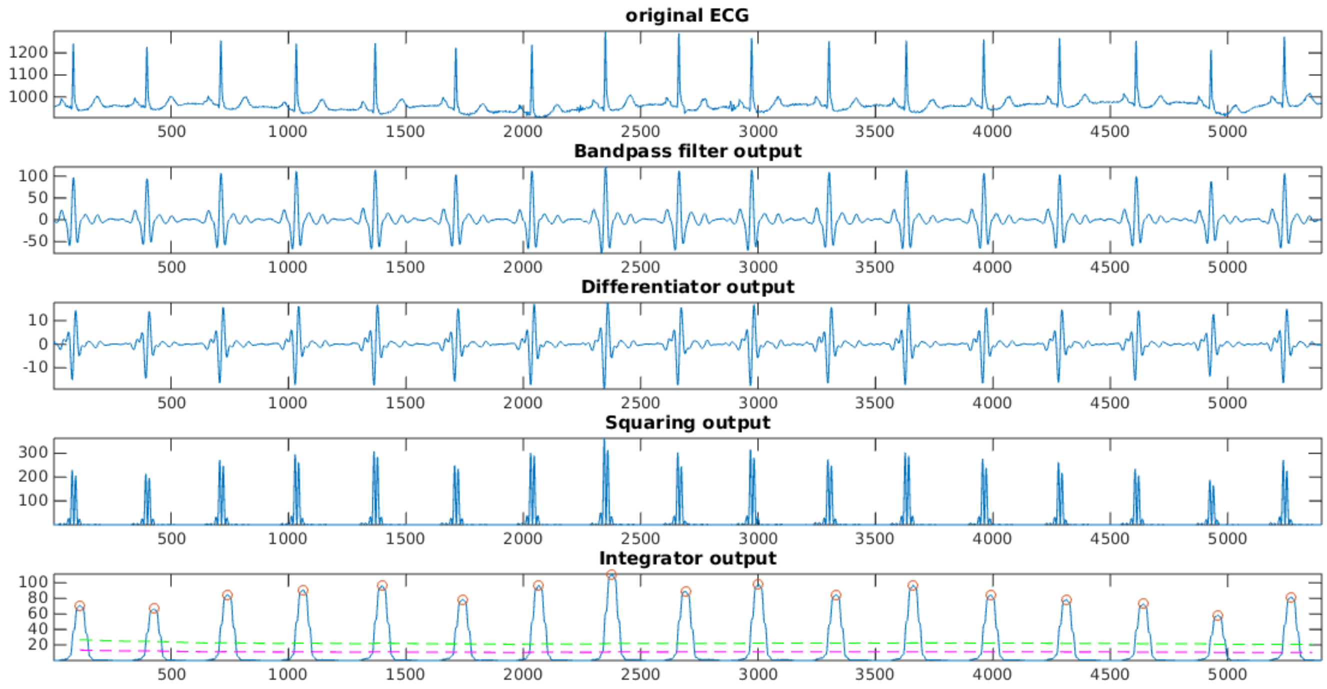

Figure 1: Signal at the various steps of the Pan Tompkins Algorithm

the interval between two QRS peaks. Actually two different RR intervals are memorized but for simplicity in this report I will only mention one. There is a main threshold and if the signal is higher than it and it lays in a valid RR interval then this peak is detected as a valid QRS peak. In the other hand if no valid peak is found in a valid interval then a search-back method is performed by using the second threshold (which is obviously lower than the first one).

2.2 Implementation

The implementation found in [6] was the starting basis for the implementation I developed. I decided to simplify the implementation and use only one type of band-path IIR filter for filtering the noise. The detection stage was completely re-done with a more simple approach.

The Pan Tompkins algorithm is quite straightforward to implement in software but it uses IIR filters, a non-trivial detection mechanism and this results in complex hardware. For this reason I decided to go against using this particular algorithm for my project.

3 Novel Real-Time Low Complexity Algorithm

3.1 Description of the algorithm

The algorithm is proposed in [3]. This is a much simpler algorithm in terms of computational power. The preprocessing stage uses less and simpler filter. The processing or detection stage is implemented with a three state finite state machine (FSM).

Preprocessing is made first by a differentiator, followed by an integrator, finishing with a square function. The differentiator is implemented using a simpler difference equation than the one used in the previous algorithm thus making it much easier to implement in hardware. The integrator uses a window equal to eight (with a sampling frequency of 360 Sa/s) which is a power of two and thus the division (in the average equation) can be easily implemented by a shifter in hardware. Not only does this algorithm uses less filters but they are simpler to implement.

Detection is based on adaptive thresholding. There is only one threshold in this case and it is only applied to the final signal from the preprocessing stage but it adapts to the signal. As mentioned before the detector is a FSM with three states. There is a counter running and it's value is the decisive value in two of the states. During the first state the FSM looks for the highest value in the signal which is then saved as a peak, it goes to the next state when the value of the counter is higher than the minimum feasible RR interval (200ms) plus the standard duration of the QRS complex (60ms). Then the counter is set to a value equal to the current signal index minus the index of the previously found peak. The second state simply keeps the counter going until the value of the counter is higher than the minimum RR interval. This is where the threshold is set to the average value of all the previous peaks. In the final state the FSM looks for a value of the signal higher than the threshold, in the meantime the threshold is decreased exponentially.

3.2 Implementation

The novel algorithm was implemented on matlab keeping true to it's real time nature by analyzing one sample after the other. The initial implementation was written in floating point arithmetics and testes with the MIT-DIB database. After validating the results, the script was converted into fixed-point arithmetics manually by with the help of the fixed point toolbox of Matlab. This toolbox automatically decides the best word-length and the length for the fraction part for each variable.

4 Future Work

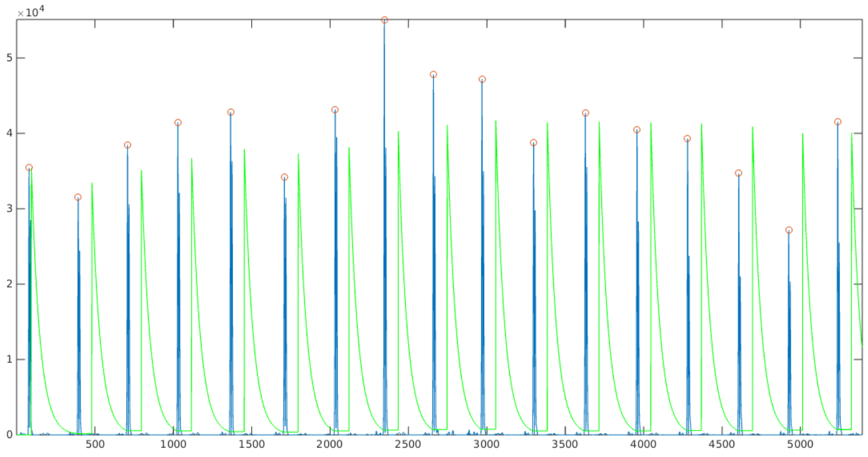

Figure 2: Close up of the final output signal with the adaptive threshold

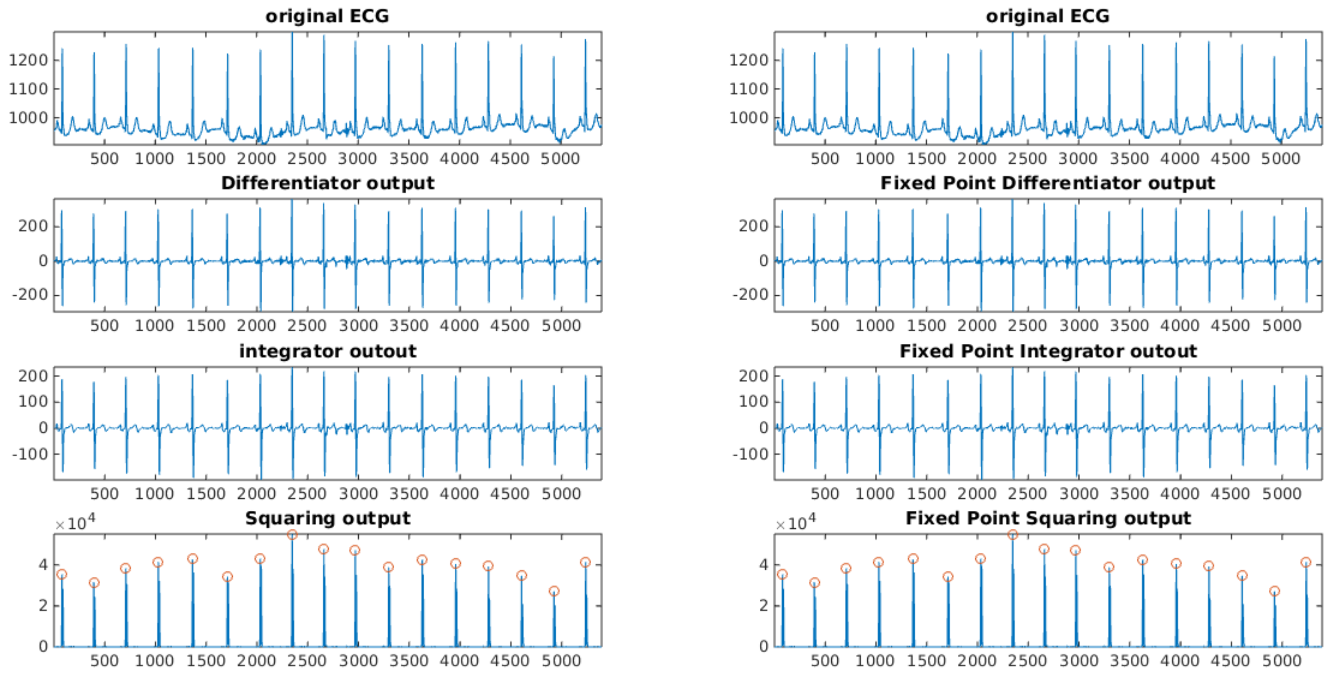

Figure 3: Comparison of floating point vs. optimal fixed point

Signal/Variable	Signed/Unsigned	Wordlength	Fraction length
Input Signal	Unsigned	11	0
Differential Output	Signed	12	0
Integrator Output	Signed	15	3
Squaring Output	Unsigned	24	0
count	Unsigned	7	0
R peak values	unsigned	24	0
R peak position values	unsigned	15	0

Table 1: Optimal Word and Fraction lengths for important variables

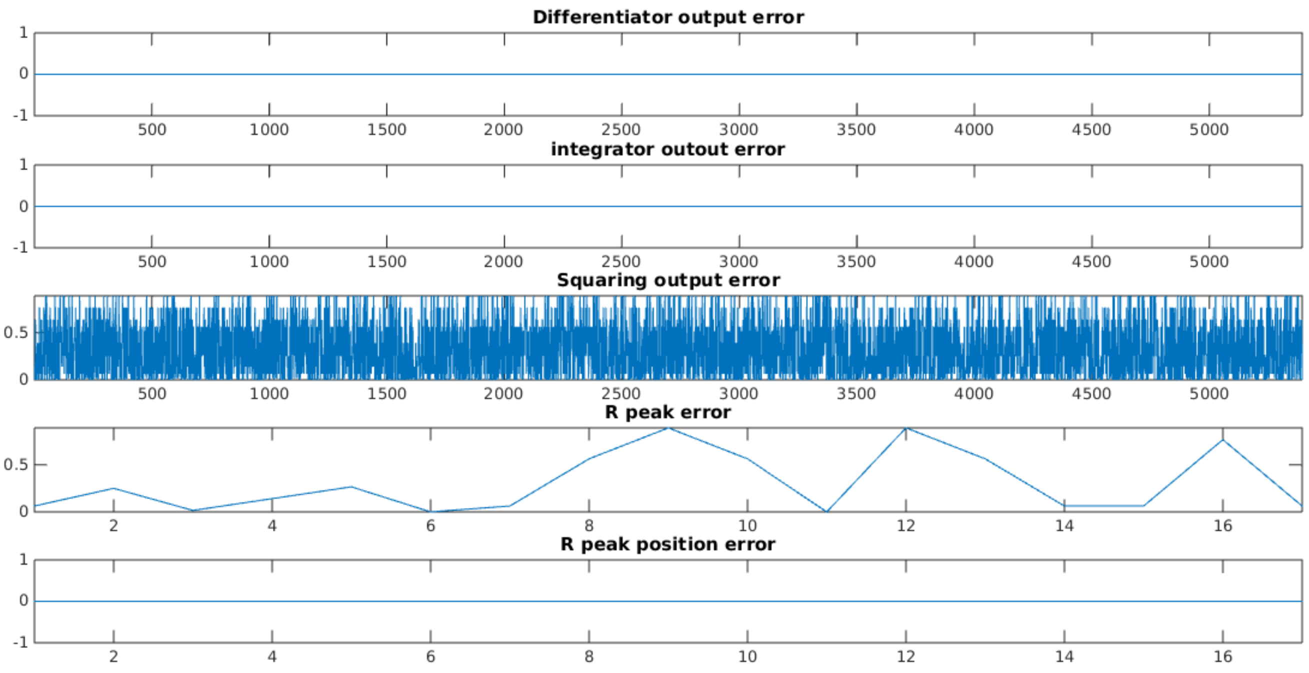

Figure 4: Absolute Errors of the optimal fixed point Algorithm

Signal/Variable	Signed/Unsigned	Wordlength	Fraction length
Input Signal	Unsigned	11	0
Differential Output	Signed	12	0
Integrator Output	Signed	12	0
Squaring Output	Unsigned	24	0
count	Unsigned	7	0
R peak values	unsigned	24	0
R peak position values	unsigned	15	0

Table 2: Used Word and Fraction lengths for important variables

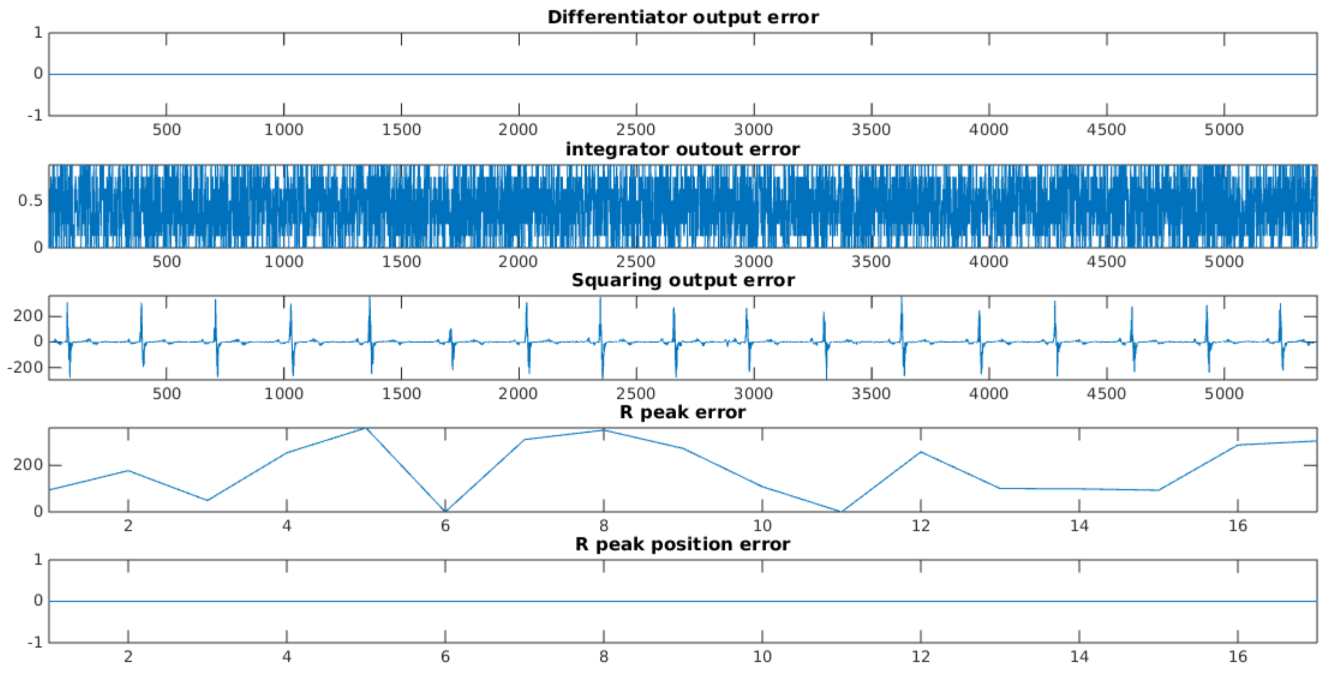

Figure 5: Absolute Errors of the used fixed point Algorithm

References

[1] Moody GB and Mark RG. The impact of the mit-bih arrhythmia database. *IEEE Eng in Med and Biol*, 20(3):45–50, May-June 2001.

[2] AL Goldberger, LAN Amaral, L Glass, JM Hausdorff, PCh Ivanov, RG Mark, JE Mietus, GB Moody, C-K Peng, and HE Stanley. Physiobank, physiotoolkit, and physionet: Components of a new research resource for complex physiologic signals, June 2000.

[3] Raquel Gutirrez-Rivas, Jess Garca, William P. Marnane, and Alvaro Hernndez. Novel real-time low-complexity qrs complex detector based on adaptive thresholding. *IEEE Sensors Journal*, 15(10):6036–6043, October 2015.

[4] Bert-Uwe Kohler and Carsten Hennig. The principles of software qrs detection. *IEEE Engineering in Medicine and Biology*, 2002.

[5] Jiapu Pan and Willis J. Tompkins. A real-time qrs detection algorithm. *IEEE Transactions on Biomedical Engineering*, BME-32(3):230–236, March 1985.

[6] Hooman Sedghamiz. Complete pan tompkins implementation ecg qrs detector, March 2014.