Project Title: FPGA System for Computer-aided Diagnosis of Cardiac Murmurs

Author: Kai Lu

Abstract: A computer-aided diagnosis (CAD) algorithm was designed and implemented on an Altera Cyclone II FPGA to detect cardiac murmurs from recorded heart signals. Training and evaluation data sets were obtained from the online "Classifying Heart Sounds Challenge" sponsored by PASCAL. The detection algorithm calculates the Low Energy Rate (LER) from a recorded heart signal and performs a binary classification of the sample as either normal or murmur. The FPGA system interfaces with a commercial digital stethoscope to acquire real time data as well as a VGA-compatible monitor for visualization and metric reporting. In addition, basic I/O was developed for user input to the system. The result is a complete embedded system capable of acquiring and analyzing cardiac data in real time to detect heart murmurs.

Executive Summary

Cardiac murmurs are pathologic sounds that are produced by turbulent blood flow in the heart. Detailed diagnoses of pathologic murmurs often require echocardiogram procedures. Although the procedure is effective, it requires special equipment and trained technicians to capture the necessary images and measurements. On the other hand, heart murmurs can sometimes be detected by a physician using a standard stethoscope during auscultation. This procedure is commonly performed during routine check-ups. However, depending on the grade or severity of the murmur, the quality of the stethoscope, and the training and skill of the physician, it can be difficult for a physician to distinguish a murmur from a normal heartbeat. This design project aims to assist physicians in detecting heart murmurs by analyzing cardiac signals in real time during auscultation and reporting any detected abnormalities.

A computer-aided diagnosis (CAD) algorithm was designed and implemented on an Altera Cyclone II FPGA to detect cardiac murmurs from recorded heart signals. Training and evaluation data sets were obtained from the online “Classifying Heart Sounds Challenge” sponsored by PASCAL. The detection algorithm calculates the Low Energy Rate (LER) from a recorded heart signal and performs a binary classification of the sample as either normal or murmur using a trained support vector machine. The Low Energy Rate of a heart signal is defined as the fraction of the signal that is below its root mean square (RMS) value. The FPGA system interfaces with a commercial digital stethoscope to acquire real time data as well as a VGA-compatible monitor for visualization and metric reporting. In addition, basic I/O was developed for user input to the system. The result is a complete embedded system capable of acquiring and analyzing cardiac data in real time to detect heart murmurs.

Table of Contents

1. Introduction

1.1. Motivation

Murmurs are pathologic sounds that are produced by turbulent blood flow in the heart. Detailed diagnoses of pathologic murmurs often require echocardiogram procedures. Although the procedure is effective, it requires special equipment and trained technicians to capture the necessary images and measurements. On the other hand, heart murmurs can sometimes be detected by a physician using a standard stethoscope during auscultation. This procedure is commonly performed during routine check-ups. However, depending on the grade or severity of the murmur, the quality of the stethoscope, and the training and skill of the physician, it can be difficult for a physician to distinguish a murmur from a normal heartbeat. This design project aims to assist physicians in detecting heart murmurs by analyzing cardiac signals in real time during auscultation and reporting any detected abnormalities.

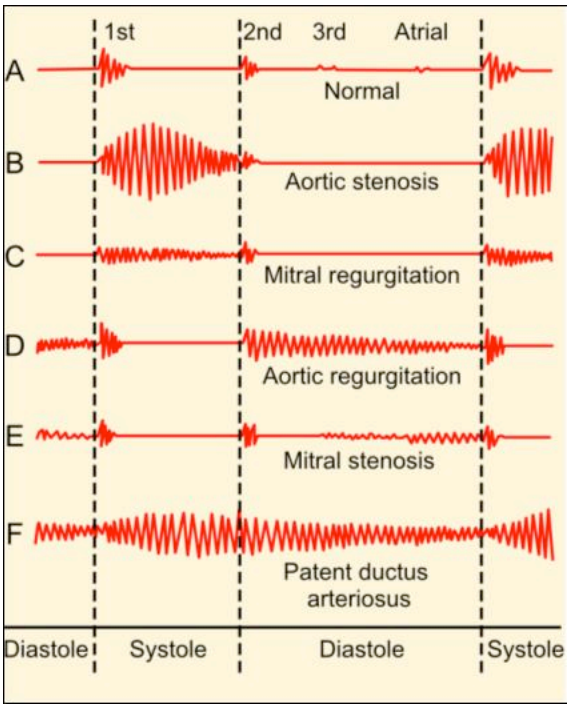

Figure 1. Ideal acoustic models for normal and abnormal heart sounds [1]

1.2. Background

The task of designing a cardiac murmur detection algorithm has been previously explored by several researchers in various academic groups. In general, the task can be described as a pattern recognition problem using 1-dimensional medical data. Pattern recognition typically involves two key steps: (1) Feature extraction and (2) Classification. In feature extraction, one or more discriminative metrics are calculated using the input data. These metrics are then used in classification to assign a specific class label to the input data. For the problem of detecting cardiac murmurs, the classification is binary – assigning either a normal or murmur label to the analyzed data.

The features commonly evaluated in literature belong to three distinct families: (1) time-varying and time-frequency features, (2) perceptual features, and (3) fractal features [2]. Time-varying and time-frequency features are based on different analyses of the signal's frequency spectrum and include operations such as the Short-time Fourier Transform and Wavelet Transform. Perceptual features are calculated based on the observation that a human's perception of sound frequencies follows a logarithmic instead of a linear curve. The most popular perceptual feature is the Mel-Frequency Cepstral Coefficients of the signal. Lastly, fractal features involve the estimation of complexity and/or chaos of a signal using measures such as the Hurst Exponent and the Correlation Dimension.

The classification scheme can be selected independently from the features calculated from the data. In general, classifiers can be grouped as either parametric or non-parametric. Parametric models are often used when a specific mathematical form can be estimated from the dataset. Examples of parametric classifiers include Support Vector Machines and Maximum Likelihood Estimation. Non-parametric models are typically used when the relationship between the input data and their respective classes cannot be simply represented or are not well understood. Examples of non-parametric models include K-Nearest Neighbor, Neural Networks, and Hidden Markov Models.

1.3. Challenges and Constraints

Although various features and classification approaches have been successfully developed and tested, the performance of the algorithms depends highly on the specific training and evaluation data sets. Pattern recognition in medical diagnostics often suffers from a lack of data, particularly in comparison to the problems being solved in non-medical fields such as voice recognition in audio recordings or face detection in images. The lack of data is partly due to patient confidentiality but is also caused by the limited number of available ground truth datasets. Trained physicians and technicians are often needed to generate accurate ground truth annotations. In comparison, almost any individual is capable of labeling faces in image or the words that are being spoken in a recording.

In addition to robustly handling small data sets, there are several hardware and software constraints related to the implementation of the detection algorithm on an FPGA system. The memory constraint is the most significant challenge. The Altera DE2 development board includes 8 MB SDRAM, 512 KB SRAM, and 4 MB Flash memory. The designed algorithm and user interface must be memory-efficient to fit within these limits. In addition, computational efficiency is also an important factor. Many features require several matrix operations to be computed on the input data. These must be computed relatively quickly to reduce the analysis time on the system.

2. System Design

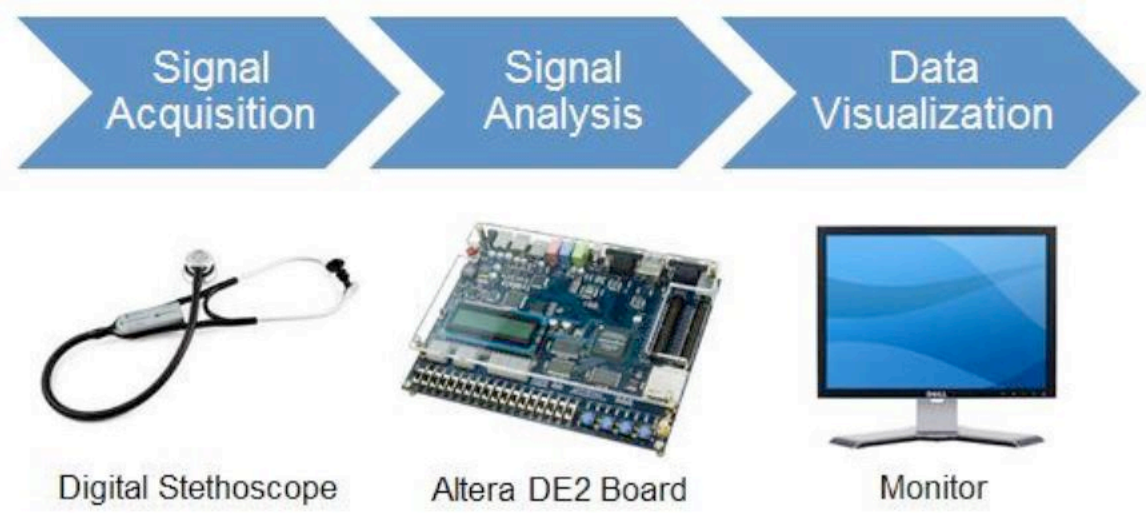

Figure 2. System Overview

The system hardware is composed of three core components: (1) A commercial digital stethoscope [3] for signal acquisition, (2) the Altera DE2 development board for signal analysis and user interface processing, and (3) A standard VGA-compatible monitor for data visualization. The system design was divided into two phases: (1) Development of the murmur detection algorithm and (2) Implementation on the FPGA.

2.1. Murmur Detection

2.1.1. Datasets

Training and evaluation data sets were obtained from the online "Classifying Heart Sounds Challenge" sponsored by PASCAL [4]. The challenge provides two anonymous and publicly available datasets of acoustic heart signals: (1) Dataset A – generated from the commercially available iStethoscope Pro iPhone app and (2) Dataset B – generated from a digital stethoscope DigiScope being tested in clinical trials. Because of the poor and inconsistent quality of Dataset A, only Dataset B was used for this project.

Dataset B contains 4 ground truth classifications of heart signal data and 1 unlabeled evaluation set. The classifications are as follows: normal, murmur, extra systole, and artifact (recordings with excessive noise or other background sounds). Only the normal and murmur ground truth sets were used for training and evaluation. Any extra systole (extra heart sound) or noisy signal would be ideally classified as a murmur by the detection algorithm. From the normal and murmur training sets and the single evaluation set, noisy recordings were manually removed, resulting in a final dataset count of 149 training recordings (83 normal, 66 murmur) and 117 evaluation recordings.

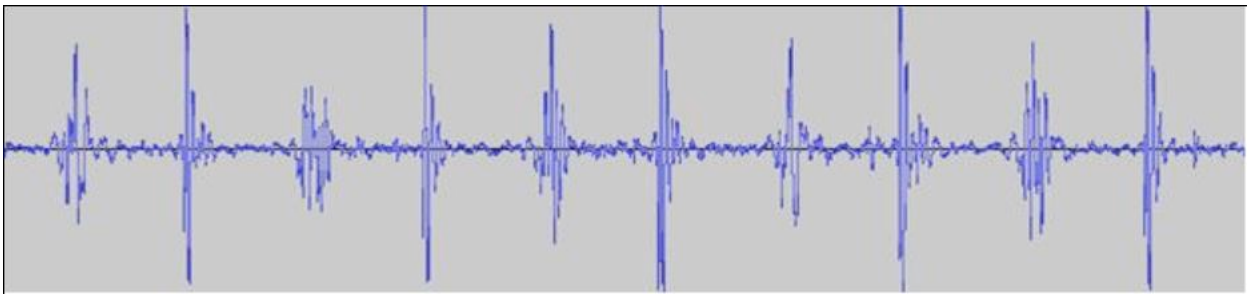

Figure 3. Example Waveform from Normal Dataset (170_1307970562729_C.wav)

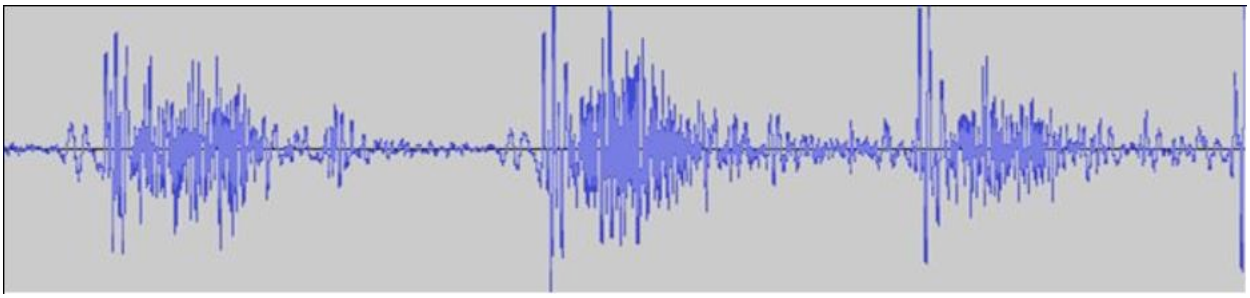

Figure 4. Example Waveform from Murmur Dataset (248_1309201683806_C.wav)

2.1.2. Feature Extraction

The feature extraction step of the detection algorithm calculates the Low Energy Rate [5], a single scalar value to be used as the discriminating parameter during classification. The Low Energy Rate of a heart signal is defined as the fraction of the signal that is below its root mean square (RMS) value. The RMS value for a signal x is:

$$X_{RMS}=\sqrt{\frac{1}{n}(x_1^2+x_2^2+\cdots+x_n^2)}$$

Figure 5. RMS Equation

where x_i is the i_{th} sample of signal x (1 indexed). The Low Energy Rate for a signal is between 0 and 1.

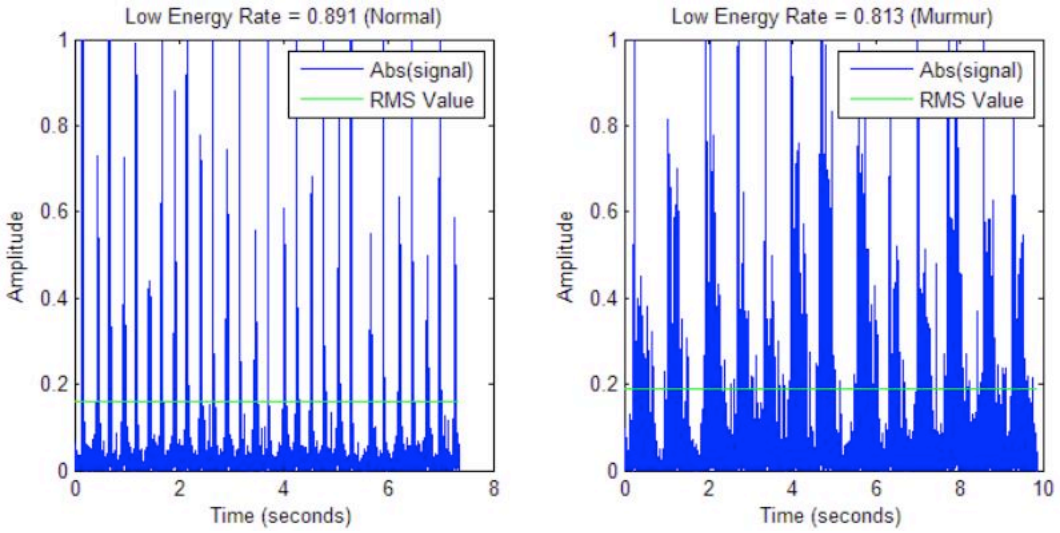

Figure 6. Low Energy Rate Visualization from Example Waveforms

Figure 6 shows a visualization of the Low Energy Rate for a normal (left) and murmur (right) waveform. The absolute value of each signal is used to compare against the RMS value when calculating the fraction of data that is below the RMS. The intuition behind this measure is that murmur heart sounds are frequently more turbulent and contain higher amplitude events than normal, which results in smaller Low Energy Rates. Calculations of the Low Energy Rate on both the training and evaluation datasets were implemented using MATLAB.

2.1.3. Classification

After the Low Energy Rates were calculated for each waveform in the normal and murmur ground truth training sets, a support vector machine was trained in MATLAB

using the data. The function *'svmtrain'* outputs an SVM model that describes an optimized decision plane (or decision line for this specific project) through support vectors and bias/scaling data. Using the training data, the optimal decision boundary was computed to be a Low Energy Rate of 0.8633. For classification of the evaluation set, signal Low Energy Rates > 0.8633 are classified as normal and Low Energy Rates ≤ 0.8633 are classified as murmurs.

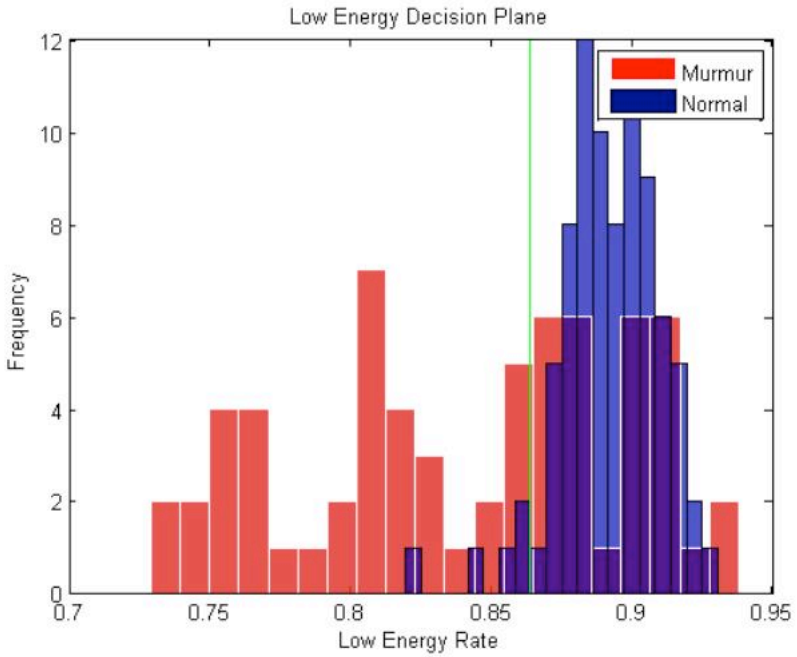

Figure 7. SVM Decision Plane for Training Data

Figure 7 shows a histogram of the Low Energy Rates calculated from the training datasets. The green vertical line indicates the decision plane output by the SVM. As the data shows, the feature space is not completely separable and a degree of classification error will occur on both the training and evaluation sets. The specific performance of this algorithm is described in the next section.

2.1.4. Performance

Three metrics were used to evaluate the performance of the algorithm: (1) Accuracy, (2) Sensitivity, and (3) Specificity. These measures are defined as follows:

$$\text{Accurary} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Sensentivity = \frac{TP}{TP + FN}$$

$$Specificity = \frac{TN}{TN + FP}$$

Figure 8. Evaluation Metrics

where TP/FP are the number of true/false positives and TN/FN are the number of true/false negatives. A positive identification refers to a murmur classification, while a negative refers to a normal classification. The accuracy describes the overall accuracy of the system, measuring how many classifications the algorithm correctly labeled out of the total number of classifications assigned. Sensitivity is a measure of how well the system detects and classifies murmurs. Similarly, specificity is a measure of how well the system detects and classifies normal signals.

There is always a trade-off between the sensitivity and specificity of a system. For example, if the decision plane was set to 1. All analyzed signals would be categorized as murmurs, which results in a perfect sensitivity of 1 but a poor specificity of 0. On the other hand, if the decision plane was set to 0. All signals would be classified as normal, which results in a poor sensitivity of 0 and a perfect specificity of 1. This trade-off for a specific system is typically summarized as a receiver operating characteristic (ROC) curve.

Figure 9 shows the ROC curve for the implementation of the murmur detection algorithm and its evaluation using the evaluation dataset. The y-axis shows the true positive rate (sensitivity), while the x-axis shows the false positive rate (1 – specificity). The points along the curve were obtained by incrementally adjusting the decision plane away from the optimal boundary output by the SVM. Each point on the plot represents a sensitivity/specificity performance combination at which the system can operate.

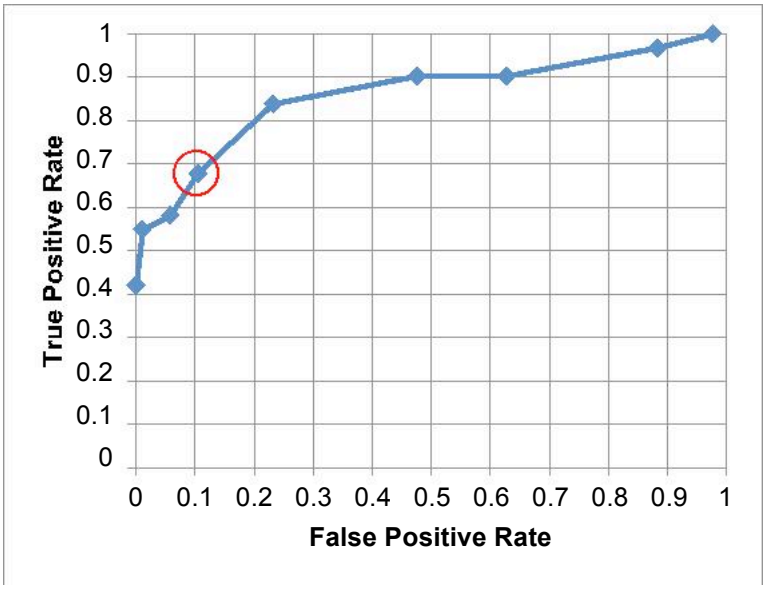

Figure 9. System ROC Curve

The point circled in red represents the performance achieved using the SVM decision boundary. The performance metrics at this operating point are: Accuracy: 83.76%, Sensitivity: 67.74%, and Specificity: 89.53%. Although this point was calculated to be optimal by the SVM, diagnostic systems are typically tuned to operate at a higher sensitivity to better detect diseases or abnormalities. Typically, physicians and patients would rather err on the side of caution and have the system output more false positives rather than false negatives.

2.1.5. Alternative Designs

Several feature vectors and their combinations were tested in the process of identifying a suitable metric for implementation. These features include: spectrogram volume, spectrogram shape, fractal dimension, signal entropy, signal brightness, MFCC values, dynamic time warp matching, and others. None of these features or any tested combinations of them resulted in better system performance than the Low Energy Rate parameter. Adaptive boosting (AdaBoost) was also applied to each of these weak classifiers using support vector machines to try to improve the system performance. Although AdaBoost can arbitrarily minimize the error bound on the training dataset, the results in the evaluation dataset did not improve.

2.2. FPGA Implementation

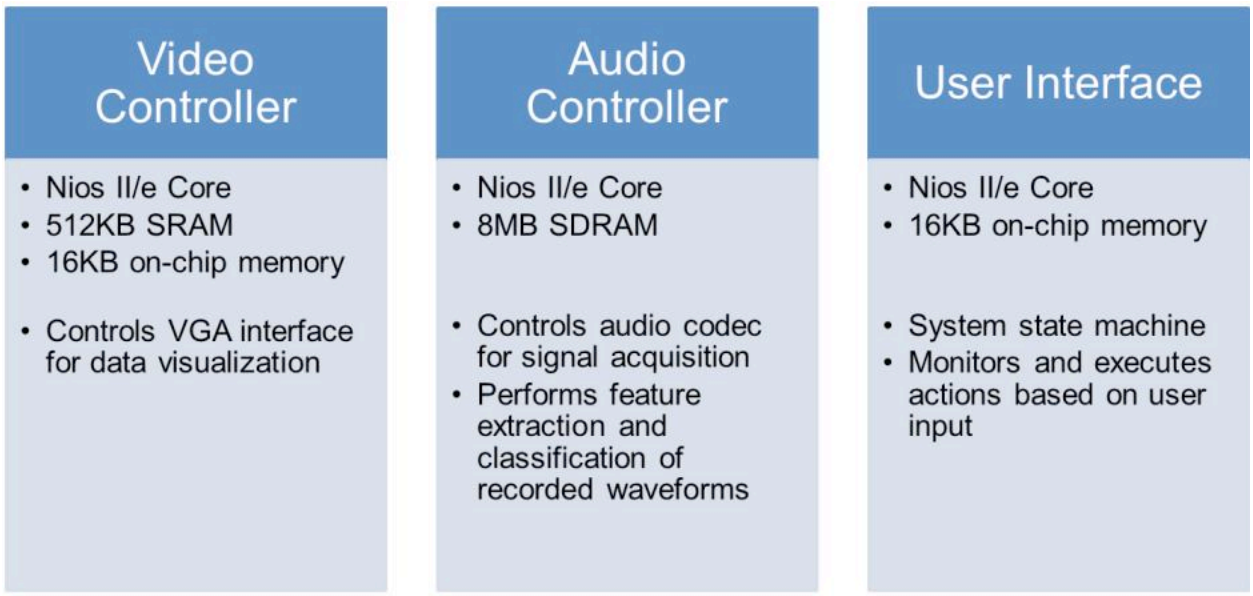

Figure 10. FPGA Overview

The core FPGA design is based on implementations of three Nios II/e economy cores, which execute compiled C programs that utilize the Nios II hardware abstraction layer (HAL) API. The HAL provides a simple device driver interface to control external components on the Altera DE2 development board. Using Nios II cores and HAL allows for rapid development of robust and complex system logic with a potential trade-off in system performance and memory utilization. The final design synthesis uses 7,263/33,216 (22%) total logic elements, 3812 registers, and 389,248/483,840 (80%) total memory bits.

2.2.1. Video Controller

The Nios II core in the video controller is configured to use the University Program (UP) – Video IP Suite, which offers basic functionality for driving output to a VGA interface. The controller supports simultaneous character and pixel display by using an alpha blender to combine two video streams into a single output to the hardware VGA controller. ASCII characters (8x8 pixels large) are supported in monochrome format and can only appear white on a transparent background. Pixel data

is represented in RGB but can only be output to a 320 x 240 resolution buffer, reducing the standard VGA resolution in ½ in both dimensions. Character data is stored in on-chip memory referencing ASCII bitmaps. Pixel data is stored in a buffer located in the 512KB external SRAM.

The video controller updates the display on start-up and subsequently only when data is received from the audio or user interface controllers. When waveform data is received from the audio controller, the pixel buffer is updated to draw a line on the on-screen plot from the last recorded data point to the new point. The visual plot has a resolution of 250 x 128 pixels. Once the plot is full, the video controller logic clears the screen to prepare for the arrival of new data. The video controller is also responsible for displaying information about the user controls, system mode, mode descriptions, analysis results, and system status. Figure 11 shows an image of the controller's VGA output.

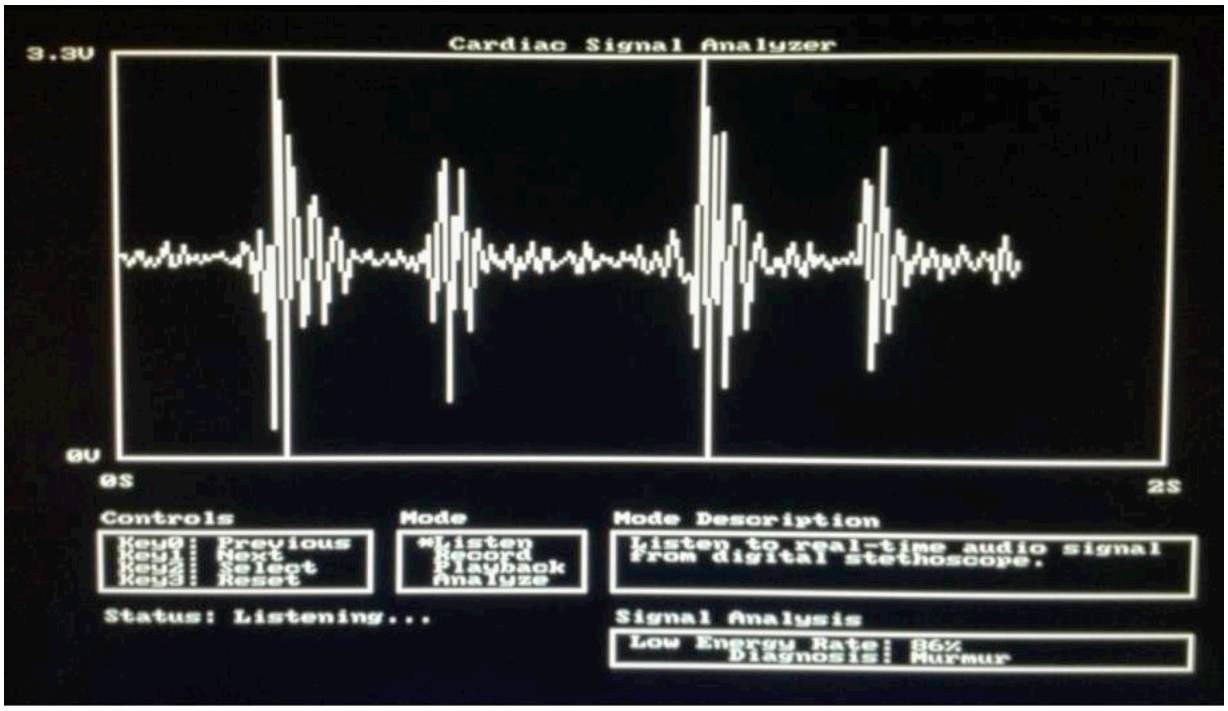

Figure 11. Screen Shot of Video Controller Output

2.2.2. Audio Controller

The audio controller acquires waveform data by sampling the analog signal from the digital stethoscope using the Wolfson WM8731 audio codec. The data is stored on the external 8MB SDRAM memory. The codec is configured to output 16 bit data at a sampling rate of 32 kHz using the UP – Audio device. However, Altera recommends implementing the device on a Nios II standard or fast processor in order to ensure that HAL driver functions will be able to read from the FIFO buffer at an appropriate rate. These processors are too large to implement in the same FPGA design as the video and user interface controllers. Therefore, the audio controller logic was developed to sample the FIFO buffer at roughly 8 kHz. Although the lower sampling frequency has adverse effects on voice recordings which range from 20 Hz – 20 kHz, internal body sounds are much lower in frequency, with major components that are much less than 1 kHz. Thus, the integrity of the acquired audio signal will not affect signal analysis.

In addition to signal acquisition, the audio controller is responsible for performing feature extraction and classification for the murmur detection algorithm. For computational efficiency, no floating point operations are used when calculating the Low Energy Rate. Once the feature is calculated and classified, the results are sent to the video controller for visualization. Data is also sent to the video controller when performing signal acquisition. Because the pixel buffer has limited resolution, the acquired signal is sub-sampled for visual representation.

2.2.3. User Interface

The user interface controller contains the state machine of the system and monitors pushbutton activity to respond to user actions. It also controls other basic I/O on the development board, such as the green and red LEDs and switches. The user interacts with the system by using the four pushbuttons KEY0 – KEY3 to change and select the mode of operation. Figure 12 provides a summary of each system mode.

Mode	Description
Listen	Listen to real-time audio signal from digital stethoscope.
Record	Record audio signal and store in memory for future

	playback or analysis.
Playback	Playback the last recorded signal.
Analyze	Analyze the last recorded signal. The Low Energy Rate and murmur diagnosis are reported.

Figure 12. System Modes

The user interface is also responsible for signaling the other controllers when a state change or pushbutton event occurs.

2.2.4. Inter-processor Communication

Data and signals must be passed and shared between one controller and another. This inter-processor communication requires synchronization to safely handle the concurrency. Each controller uses Altera hardware mutex devices for multiprocessor coordination. These mutexes are locked before a core accesses shared resources and unlocked after it is finished operating on them. Because the amount of space required for the shared resources is minimal, they are implemented as Parallel I/O ports for convenient access. If larger segments of memory are needed, the resources could be implemented using on-chip RAM memory instead.

3. Summary and Conclusions

An FPGA-based platform was developed for computer-aided diagnosis of cardiac murmurs. Acoustic heart signals are captured and analyzed in real-time with visualization on a VGA-compatible monitor for reporting. A new algorithm for murmur detection was tested and evaluated in MATLAB and implemented on an Altera Cyclone II FPGA using a multi-core Nios II system. The resulting system is capable of aiding diagnosis by detecting murmurs with 83.76% accuracy.

The murmur detection algorithm that performed the best during preliminary testing was a relatively simple feature, the Low Energy Rate. Calculations of the Low Energy Rate do not require any complex matrix operations unlike the FFT operator. The system was initially proposed to be implemented on an FPGA in order to take advantage of hardware parallelization and efficiency for more complicated feature calculations. However, the results of this project indicate that the extra processing power is unnecessary due to the simplicity of the low energy feature. Instead, a more compact microcontroller-based system can be developed instead for lower energy utilization and a smaller device footprint, as long as VGA support is not necessary.

4. References

[1] "Heart Murmur." *Wikipedia*. Wikimedia Foundation, 12 Sept. 2012. Web.

[2] Delgado-Trejos E et. Al. Digital auscultation analysis for heart murmur detection. Annals of Biomedical Engineering 2009;37:337-53.

[3] Thinklabs ds32a+ Electronic Stethoscope. <http://www.thinklabsmedical.com/>.

[4] Bentley, P. and Nordehn, G. and Coimbra, M. and Mannor, S. “The PASCAL Classifying Heart Sounds Challenge 2011”. <http://www.peterjbentley.com/heartchallenge/index.html>.

[5] G. Tzanetakis and P. Cook "Musical Genre Classification of Audio Signals", *IEEE Transactions on Speech and Audio Processing* , 10(5), July 2002.

5. Resources

- Nios II Software Handbook: http://www.altera.com/literature/hb/nios2/n2sw_nii5v2.pdf
- Altera University Program IP Cores: http://www.altera.com/education/univ/materials/comp_org/ip-cores/unv-ip-cores.html
- Support Vector Machines: http://www.mathworks.com/help/bioinfo/ug/support-vector-machines-svm.html
- ECE 5760 Course Website: http://people.ece.cornell.edu/land/courses/ece5760/