

Voice Activity Detector of Wake-Up-Word Speech Recognition System Design on FPGA

Veton Z. Kėpuska, Mohamed M. Eljhani, Brian H. Hight

Electrical & Computer Engineering Department
Florida Institute of Technology, Melbourne, FL 32901, USA

Abstract

A typical speech recognition system is push-to-talk operated that requires activation. However for those who use hands-busy applications, movement may be restricted or impossible. One alternative is to use Speech-Only Interface. The proposed method that is called Wake-Up-Word Speech Recognition (WUW-SR) that utilizes speech only interface. A WUW-SR system would allow the user to activate systems (Cell phone, Computer, etc.) with only speech commands instead of manual activation. The trend in WUW-SR hardware design is towards implementing a complete system on a single chip intended for various applications. This paper presents an experimental FPGA design and implementation of a novel architecture of a real time feature extraction processor that includes: Voice Activity Detector (VAD), and features extraction, MFCC, LPC, and ENH_MFCC. In the WUW-SR system, the recognizer front-end with VAD is located at the terminal which is typically connected over a data network(e.g., server)for remote back-end recognition. VAD is responsible for segmenting the signal into speech-like and non-speech-like segments. For any given frame VAD reports one of two possible states: VAD_ON or VAD_OFF. The back-end is then responsible to score the features that are being segmented during VAD_ON stage. The most important characteristic of the presented design is that it should guarantee virtually 100% correct rejection for non-WUW (out of vocabulary words - OOV) while maintaining correct acceptance rate of 99.9% or higher (in vocabulary words - INV). This requirement sets apart WUW-SR from other speech recognition tasks because no existing system can guarantee 100% reliability by any measure.

Keywords:Speech Recognition System (SR); Wake-Up-Word (WUW) Speech Recognition; Front-end (FE); Voice Activity Detector (VAD); Feature Extraction; Mel-frequency Cepstral Coefficients (MFCC); Linear Predictive Coding (LPC); Enhanced Mel-frequency Cepstral Coefficients(ENH_MFCC); Field Programmable Gate Arrays (FPGA).

I. Introduction

The Voice Activity Detector is responsible for segmenting the signal into speech and non-speech segments. For any given frame, VAD reports one of two possible states: VAD_ON or VAD_OFF. Word recognition in the Back-end stage begins when the VAD is in VAD_ON state, and ends when the VAD switches to VAD_OFF state. VAD works in two phases: in the first phase, a classifier decides whether a single input frame is speech-like or non-speech-like; in the second phase, the number of speech-frames and non-speech-frames over a period of time is analyzed and certain rules are applied to report the final decision (e.g., VAD_ON or VAD_OFF).

The VAD is responsible for finding sections of speech by segmenting them from the rest of the audio stream. The back-end then will identify whether or not the segmented utterance is a WUW. In the "Front-end of Wake-Up-Word Speech Recognition System Design on FPGA" [1], we showed the generation of three sets of spectrograms. In the "Wake-Up-Word Feature Extraction on FPGA" [2], we presented an efficient hardware architecture and

implementation of Front-end of WUW-SR on FPGA. This Front-end is responsible for generating three sets of features:

1. Mel-frequency Cepstral Coefficients (MFCC),
2. Linear Predictive Coding (LPC), and
3. Enhanced Mel-frequency Cepstral Coefficients (ENH-MFCC).

A great deal of work has been done to address the problem of recognizing speech-like segments by designing an efficient hardware front-end with built-in VAD in FPGA. The board that has been used is based on Altera DSP system, acting as a processor that is responsible for extracting three different sets of features from the input audio signal.

The feature extraction of speech is an important issue in the Front-end. There are two types of acoustic measurements of the speech signal. One is the parametric modeling approach, which is developed to match closely the resonant structure of the human vocal tract that produces the corresponding speech sound. It is mainly derived from Linear Predictive analysis, such as LPC-based Cepstrum (LPC). The other approach, MFCCs, is the

nonparametric modeling method that originates from the human auditory perception system. The MFCCs are derived in this form [3].

In recent studies of speech recognition system, the MFCC parameters perform better than others in the recognition accuracy [4, 5]. This presented paper shows VAD based feature extraction solution. This solution is optimized for implementation in FPGA structures. The system is designed to be implemented as a System-on- Programmable-Chip (SOPC). This design not only has a relatively low resource usage, but also maintains a reasonably high level of performance (last long if plugged-in battery usage).

The remainder of this paper is organized as follows: Section II, describes the Wake-Up-Word

speech recognition and its architecture. Section III, describes the embedded front-end of WUW-SR design procedure and architecture. Section IV, describes the Mel-frequency Cepstral Coefficient Feature design. Section V, describes the Linear Predictive Coding Feature design. Section VI, describes the Enhanced Mel-frequency Cepstral Coefficient Feature design. Section VII, describes the Voice Activity Detector Design and Implementation. Section VIII, describes results and comparisons of three spectrograms and features from FPGA hardware implementation and C++ front-end algorithm. These are followed by conclusions in Section IX.

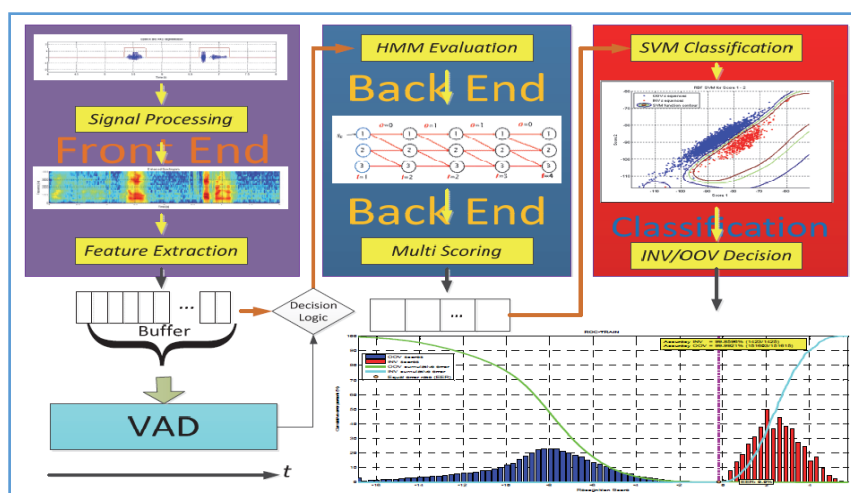


Figure 1: Overall WUW-SR architecture

II. WUW-SR Overall System Architecture

As shown in Fig. 1, the WUW-SR can be broken down into three components as explained in [6]. The front-end system process takes an input waveform (“Operator” audio signal) and outputs a sequence of parameters; that is MFCCs, LPCs, and ENH-MFCCs features as described in [2]. The Voice Activity Detector system is responsible for segmenting the signal into speech and non-speech segments. For any given frame, VAD reports one of two possible states: VAD_ON or VAD_OFF.

Word recognition in the Back-end stage begins when the VAD enters VAD_ON state, and ends when the VAD switches to VAD_OFF state. Whereas the back-end process takes this sequence, e.g., VAD_ON state, and outputs the recognized command. From the figure above, the signal processing module accepts raw audio samples and produces spectral representations of short time signals. The feature-extraction module generates features from this spectral representation, which are decoded with the corresponding hidden Markov’s models (HMMs). The individual feature

scores are classified using support vector machines (SVMs) into INV, or OOV: in-, out-of-vocabulary speech [6].

III. Embedded Front-end with Built-in VAD of WUW-SR Architecture

As shown in Fig. 2, the new Front-end design is divided into twenty-eight modules (five-stages). The first seven pink-colored modules represents the pre-processing stage and used as basic modules to provide windowed speech signal for the other stages. The six yellow-colored modules represent the LPC stage and used to generate 13-LPC features. The five brown-colored modules represents the MFCC stage and used to generate 13 MFCCs features. The five green-colored modules represents the ENH-MFCC stage and used to generate 13 ENH-MFCC features. The five red-colored modules represents the VAD stage that is responsible for finding segments of spoken speech.

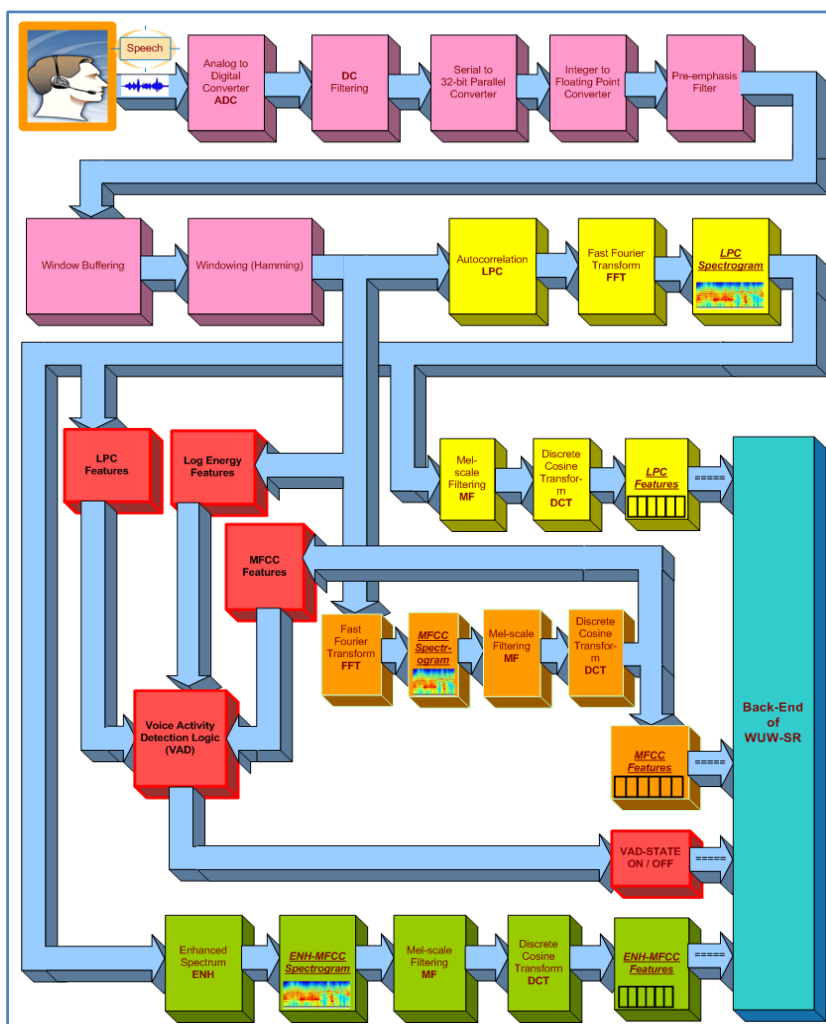


Figure 2: Front-end with Built-in VAD of WUW-SR Block Diagram

Stage A: Pre-Processing

1. Analog to Digital Converter ADC.
2. DC Filtering.
3. Serial to 32-bit parallel converter.
4. Integer to floating-point converter.
5. Pre-emphasis filtering.
6. Window advance buffering.
7. Hamming window.

Stage B: Linear Predictive Coding Coefficients

1. Autocorrelation Linear Predictive Coding.
2. Fast Fourier Transform (FFT).
3. LPC Spectrogram.
4. Mel-scale Filtering.
5. Discrete Cosine Transform (DCT).

Stage C: Mel-Frequency Cepstral Coefficients

1. Fast Fourier Transform (FFT).
2. MFCC Spectrogram.
3. Mel-scale Filtering.
4. Discrete Cosine Transform (DCT).

Stage D: Enhanced Mel-Frequency Cepstral Coefficients

1. Enhanced Spectrum.
2. Enhanced MFCC Spectrogram.

3. Mel-scale Filtering.
4. Discrete Cosine Transform (DCT).

Stage E: Voice Activity Detector

The five blue-colored modules represents the Voice Activity Detector stage. The VAD is responsible for finding utterances spoken in the correct context and segmenting them from the rest of the audio stream, then the system will identify whether or not the segmented utterance is a WUW.

1. LPC Spectrogram Features.
2. Log Energy Features.
3. MFCC Feature.
4. Voice Activity Detection Logic.

IV. Mel-scale Frequency Cepstral Coefficients (MFCC) Design and Implementation

The feature extraction involves identifying the *formants* in the speech, which represents the frequency locations of energy concentrations in the speaker's vocal tract. There are many different approaches used: Mel-scale Frequency Cepstral Coefficients (MFCC), Linear Predictive Coding (LPC), Linear Prediction Cepstral Coefficients (LPCC), Reflection Coefficients (RCs). Among these,

MFCC has been found to be more robust in the presence of background noise compared to other algorithms [7]. Also, it offers the best trade-offs between performance and size (memory) requirements. The primary reason for effectiveness of MFCC is that, it models the non-linear auditory response of the human ear which resolves frequencies on a log scale [8].

Intensive efforts have been carried out to achieve a high performance front-end. Converting a speech waveform into a form suitable for processing by the decoder requires several stages as shown in Fig. 2:

1. **Filtration:** The waveform is sent through a low pass filter, typically 4 kHz to 8 kHz. As is evidenced by the bandwidth of the telephone system being around 4 kHz; this is sufficient for comprehension and used a minimum bandwidth required for telephony transmittal.
2. **Analog-to-Digital Conversion:** The process of digitizing and quantizing an analog speech waveform begin with this stage. Recall that the first step in processing speech is to convert the analog representations (first air pressure, and then analog electric signals from a microphone), into a digital signal.
3. **Sampling rate:** The resulting waveform is sampled. Sampling rate theory requires a sampling (Nyquist) rate of double the maximum frequency (so 8 to 16 kHz as appropriate). The sampling rate of 8 kHz was used in our front-end. (We used CODEC Chip to perform first, second, and third stages).
4. **Serial to Parallel Converter:** This model gets serial digital signal from CODEC and converts it to 32-bit.
5. **Integer to floating-point converter:** This module converts 32-bit, signed integer data to single-precision (32-bit) floating-point values. The input data is routed through the int_2_float Megafunction core named ALTFP_CONVERT.
6. **Pre-emphasis:** the digitalized speech signal $s(n)$ is put through a low-order (LPF) to spectrally flatten the signal and to make it less susceptible to finite precision effects later in the signal processing. The filter is represented by:

$$y[n] = x[n] - \alpha x[n-1]$$

$$\text{Output} = \text{Input} - (\text{PRE_EMPH_FACTOR} * \text{Previous_input}).$$

Where we have chosen the value of PRE_EMPH_FACTOR(α) as 0.975.
7. **Window Buffering:** A 32-bit, 256 deep dual-port RAM (DPRAM) stores 200 input samples. A state machine handles moving audio data into the RAM, and pulling data out of the RAM (40 samples) to be multiplied by the Hamming coefficients, which are stored in a ROM memory.
8. **Windowing:** The Hamming window function smooth the input audio data with a Hamming curve prior to the FFT function. This stage slices the input signal into discrete time segments. This is done by using window N milliseconds, typically 25 ms wide (200 samples). A Hamming window size of 25ms which consists of 200 samples at 8 KHz sampling frequency and 5 ms frame shift (40 samples) is picked for our front-end windowing.
9. **Fast Fourier Transform:** In order to map the sound data from the time domain to the frequency domain, the Altera IP Megafunction FFT module is used. The module is configured so as to produce a 200-point FFT. This function is capable of taking a streaming data input in natural order, and it can also output the transformed data in natural order, with maximum latency of 200 clock cycles once all the data (200 data samples) has been received.
10. **Spectrogram:** This module takes the complex data generated by the FFT and performs the function:

$$20 * \log_{10}(\text{fft_real}^2 + \text{fft_imag}^2).$$

We designed spectrogram to show how the spectral density of a signal varies with time. We used spectrogram module to identify phonetic sounds. Digitally sampled data, in the time domain, are broken up into chunks, which usually overlap, and Fourier transformed to calculate the magnitude of the frequency spectrum for each chunk. Each chunk then corresponds to a vertical line in the image; a measurement of magnitude versus frequency for a specific moment in time. The spectrums or time plots are then "laid side by side" to form the image surface.
11. **Mel-scale filtering:** While the resulting spectrum of the FFT contains information in each frequency in linear scale, human hearing is less sensitive at frequencies above 1000 Hz. This concept also has a direct effect on performance of automatic speech recognition systems; therefore, the spectrum is warped using a logarithmic Mel scale. In order to create this effect on the FFT spectrum, a bank of filters is constructed with filters distributed equally below 1000 Hz and spaced logarithmically above 1000 Hz.
12. **Discrete Cosine Transform:** DCT is a Fourier-related transform similar to the discrete Fourier

transform (DFT), but using only real numbers. DCTs are equivalent to DFTs of roughly twice the length, operating on real data with even symmetry (since the Fourier transform of a real and even function is real and even). A DCT computes a sequence of data points in terms of summation of cosine functions oscillating at various frequencies. The idea of performing DCT on Mel Scale is motivated by extraction of the speech frequency domain characteristics. DCT module reduces the speech signal's redundant information, and reaches the aim of regulating the speech signal into feature coefficients with minimal dimensions.

V. Autocorrelation Linear Predictive Coding (LPC) Design and Implementation

As shown in Fig. 2, an additional module named Autocorrelation Linear Productive Coding (LPC) used to extract the speech as LPC features. The basic idea of LPC is to approximate the current speech sample as a linear combination of past samples as shown in the following equation:

$$x[n] = \sum_{k=1}^p a_k x[n - k] + e[n]$$

$x[n - k]$: Previous speech samples
 p : Order of the model
 a_k : Prediction coefficient
 $e[n]$: Prediction error

This module gets windowed data from the window module for representing the spectral envelope of a digital signal of speech in compressed form, using the information of a linear predictive model. We used this method to encode good quality speech and provide an estimate of speech parameters.

The goal of this method is to calculate prediction coefficients a_k for each frame. The order of LPC, which is the number of coefficients p , determines how closely the prediction coefficients can approximate the original spectrum. As the order increases, the accuracy of LPC also increases. This means the distortion will decrease. The main advantage of LPC is usually attributed to the all-pole characteristics of vowel spectra. Also, the ear is also more sensitive to spectral poles than zeros [9]. In comparison to non-parametric spectral modeling techniques such as filter banks, LPC is more powerful in compressing the spectral information into few filter coefficients [10].

VI. Enhanced Mel-scale Frequency Cepstral Coefficients (ENH-MFCC) Design and Implementation

The spectrum enhancement module is used to generate ENH-MFCC set of features. We have implemented this module as shown in the Fig. 2, to perform an enhancement algorithm on the LPC spectrum signal. The ENH-MFCC features have a higher dynamic range than regular MFCC features, so these new features will help the back-end in improving the recognition quality and accuracy.

The algorithm uses only the single-sided spectrum, so the state machine starts the calculations when 128 data points have been written into the input RAM.

VII. Voice Activity Detector (VAD) Design and Implementation

There are various kinds of signals that may contain actual speech. In our Front-end system it is imperative to monitor and detect any speech signal. The algorithm is designed to most efficiently delineate between speech vs non-speech. It will consider speech segments that trigger VAD_ON for more than 15 frames otherwise it will not trigger.

To achieve speech / non-speech detection, as shown in fig. 3, we designed and implemented VAD models. The VAD continuously monitors the system and calculates for every frame if the signal is a speech signal or a background noise. The VAD uses three inputs: Log Energy, LPC Features, and MFCC Features; to decide whether VAD is ON or VAD is OFF.

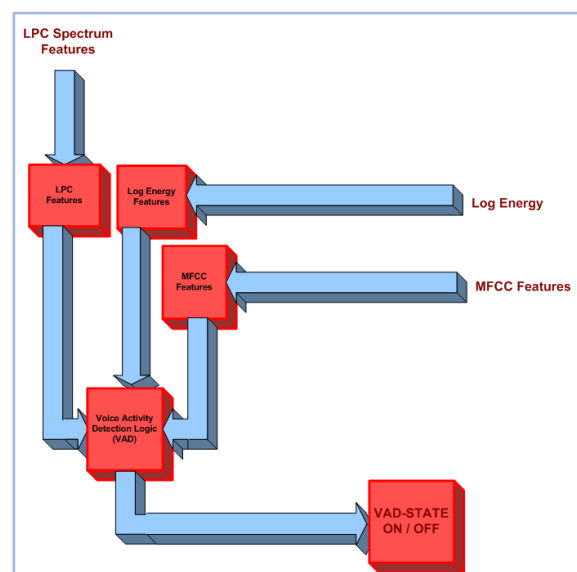


Figure 3: Voice Activity Detector Architecture

Those features are finely tuned to detect any voice in the speech frame. Each feature has an independent detection method. Depending on the calculations done by the features, the overall VAD

logic reacts by turning the VAD output ON/OFF (Speech or Noise Background). Each feature calculates the presence of speech by calculating Variance and Mean deviation. Each feature has its own flag to be set. Each feature is finely tuned to meet certain criteria before its feature flag is set.

The final decision is based on all these flags/states; if at least two out of the three are true, the overall logic of VAD triggers and the VAD turns ON. Fig. 4 shows “Onward” waveform as input audio data superimposed with its VAD segmentation, its MFCC

spectrogram, LPC spectrogram, and ENH-MFCC. The VAD, as is implemented, uses three features to provide backend with the final decision if the frame contains speech. Those three features are: Energy, MFCC and LPC [6].

The VAD logic depends on at least of two out of all three features to trigger in order for VAD to be ON. Otherwise VAD is not triggered (OFF).

In the following sections each individual feature is described:

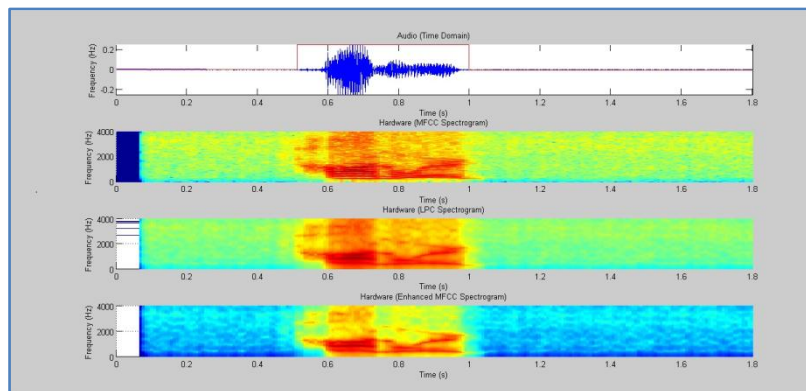


Figure 4: Speech signal “Onward” with VAD segmentation, MFCC, LPC, and ENH-MFCC spectrograms

1. Log Energy Feature

The Energy feature is computed by squaring the samples of the frames. It is used to track the change in mean frame energy. In addition it is used to detect if the frame contains speech signal or not. If there is a drastic change in frame energy the Energy flag ($\log_2_frame_energy$) is set. This feature function receives the frame energy parameter, e.g., $\log_2_frame_energy$, from the Hamming Window module. It uses this parameter to calculate the overall frame energy mean and compare it to the present frame energy. The mean frame energy, $mean_log_2_fm_en$, is dependent on a variable lambda, λ_{LTE} , which is calculated differently in the first few frames.

The mean is ignored during the VAD ON stage: $mean_log_2_fm_en_VAD_ON$. If the difference between the mean frame energy and the current frame energy is greater than certain

threshold, $SNR_THRESHOLD_VAD$, Energy flag is set.

2. Mel-frequency Cepstral Coefficient Feature

The Mel-frequency Cepstral Coefficients are calculated for each frame and they are included into the feature vector comprising MFCC component. When the mean value ($mean_vad_mfcc_feature$) of all the frames features is over the current frame’s

MFCC feature value ($local_aver_mfcc_fea_val$), the flag is set ($VAD_MFCC_State_Flag$). The mean is calculated differently depending on $VAD_MFCC_State_Flag$ and depending on the frame counter. It is dynamically adapted depending on actual values so it can detect the signal accurately. The mean is lowered if the current frame feature value is too low compared to the mean calculated up to that point. The mean is ignored during the stage where VAD is turned ON and is stored into a different variable ($mean_vad_mfcc_feature_VAD_ON$).

3. Linear Predictive Coding Feature

Linear Predictive Coding is computed from Spectrum of the signal. This function uses a different from of feature computation. It uses the variance measurement of the signal to detect if a speech is present. When the current variance (var_spec) is much larger than the average variance ($aver_var_spec$) computed from the previous frames, it indicates that there is speech. The variance flag ($VAD_SPEC_State_Flag$) is set when there is a sudden change in variance when compared to previous frames and is turned off if the change is minimal.

VIII. Results and Comparisons

After observing the VAD system on several test utterances we concluded that its hardware

implementation performs very well as compared to C++ implementation. In addition, the analysis included testing of the performance of the hardware by comparing its three sets of spectrograms and features (MFCC, LPC, and ENH-MFCC) with the C++ software implementation of the WUW's front-end.

The VAD described in this paper has been modeled and integrated with the Front-End as a processor in Verilog HDL and implemented in low cost, high speed, and power efficient (Cyclone III EP3C120F780C7) FPGA on DSP development kit.

The embedded front-end with built-in VAD used C++ implementation as a baseline. In addition the floating-point MATLAB implementation of the front-end of WUW was done. Each module was tested to ensure correct operation.

The sample utterance containing the WUW "Operator" was sampled with 8 KHz sampling rate. This utterances was included in the experiments presented in this paper as input audio data for testing the WUW's embedded Front-end. In addition, three sets of spectrograms and features produced by the front-end were compared with the baseline C++ implementation. The results depict:

As shown in Fig. 5, and 6, below, the VAD, MFCC, LPC, and ENH-MFCC spectrograms generated from the hardware front-end, and C++ baseline front-end are practically identical.

In Fig. 7, and 8, depict the generated VAD state, and MFCC, LPC, and ENH-MFCC features from hardware front-end, and baseline C++ are practically identical.

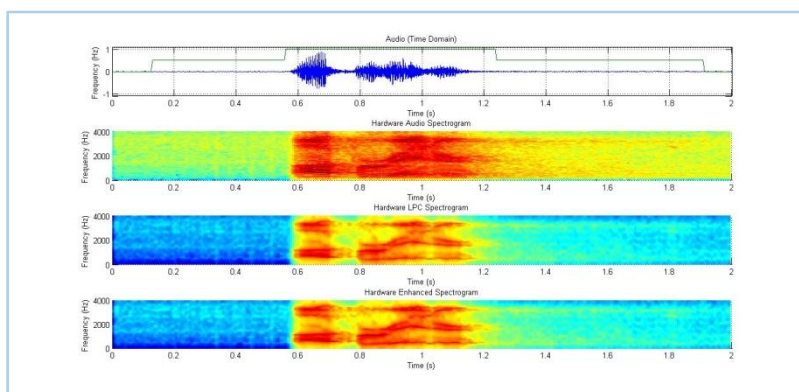


Figure 5: Hardware Front-end with VAD, MFCC, LPC, and Enhanced MFCC Spectrograms for "Operator". The VAD has three states indicating INITIAL; zero - valued, VAD_OFF; one-valued, and VAD_ON state; two – valued curve above.

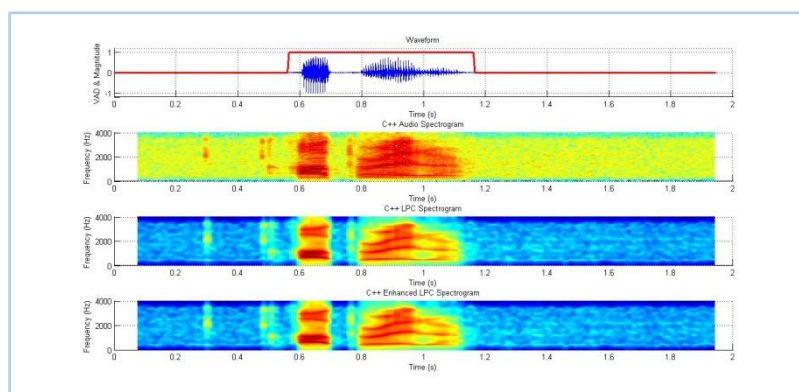


Figure 6: C++ Front-end with VAD, MFCC, LPC, and Enhanced MFCC Spectrograms for "Operator" Audio Data

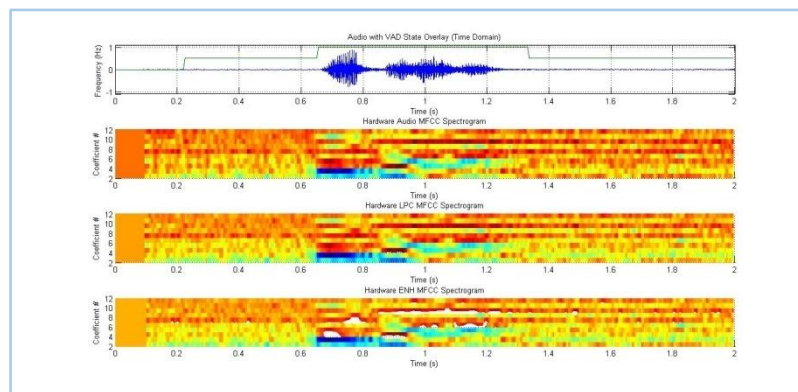


Figure 7: Hardware Front-end with VAD, MFCC, LPC, and Enhanced MFCC Histograms for "Operator" Audio Data. The VAD has three states indicating INITIAL; zero - valued, VAD_OFF; one-valued, and VAD_ON state; two – valued curve above.

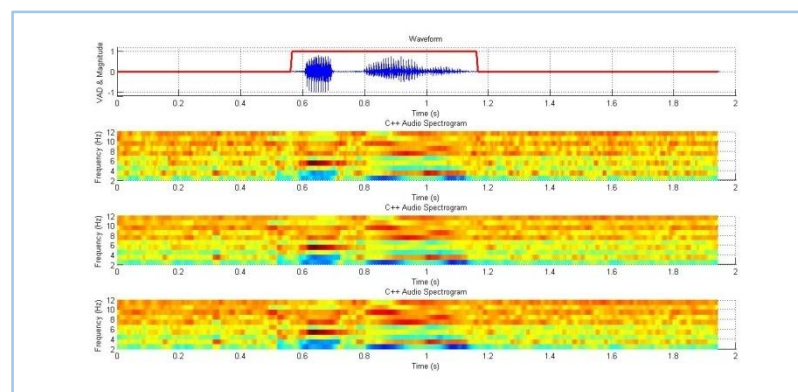


Figure 8: C++ Front-end with VAD, MFCC, LPC, and Enhanced MFCC Histograms for "Operator" Audio Data

IX. Conclusions and Applications

In this study, an efficient embedded front-end of WUW-SR hardware implementation with built-in VAD in FPGA is described. This front-end is responsible for generating three sets of features: MFCC, LPC, and ENH-MFCC. These features are needed to be decoded with corresponding HMMs in the back-end stage of the WUW Speech Recognizer. The computational complexity and memory requirement of three features and VAD algorithms is analyzed in detail. The proposed front-end is the first embedded hardware system specifically designed for WUW-SR feature extraction based on three different sets of features. To demonstrate its effectiveness, the proposed design has been implemented in cyclone III FPGA hardware [11]. The custom DSP board developed is a power efficient and flexible in design.

References

- [1] Kėpuska VZ, Eljhani MM, Hight BH (2013) Front-end of Wake-Up-Word Speech Recognition System Design on FPGA. *J TelecommunSyst Manage* 2: 108. doi:10.4172/2167-0919.1000108.
- [2] Kėpuska, V.Z., Eljhani, M.M. and Hight, B.H. (2014) Wake-Up-Word Feature Extraction on FPGA. *World Journal of Engineering and Technology*, 2, 1-12. <http://dx.doi.org/10.4236/wjet.2014.21001>.
- [3] O. Burak, Tuzun, M. Demirekler and K. Bora, "Comparison of Parametric and Non-Parametric Representations of Speech for Recognition," Proceedings of MELECON. Mediterranean Electrotechnical Conference, 1994, pp. 65-68.
- [4] J.P. Openshaw, Z.P. Sun, J.S. Mason, A comparison of composite features under degraded speech in speaker recognition, Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, Vol. 2, Minneapolis, USA, 1993, pp. 371-374.
- [5] R. Vergin, D. O'Shaughnessy, V. Gupta, Compensated mel frequency cepstrum coefficients, Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, Minneapolis, USA, 1996, pp. 323-326.
- [6] V. Z. Kėpuska and T. B. Klein, A novel Wake-Up-Word speech recognition system, Wake-Up-Word recognition task, technology and evaluation, *Nonlin. Anal.: Theory Meth. Appl.*

- 71, pp. e2772–e2789, 2009. doi:10.1016/j.na.2009.06.089.
- [7] S. Davis and P. Mermelstein, “Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences,” *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 28, no. 4, pp. 357–366, 1980.
- [8] H. Combrinck and E. Botha, “On the mel-scaled cepstrum,” department of Electrical and Electronic Engineering, University of Pretoria.
- [9] M. R. Schroeder, “Linear prediction, extremely entropy and prior Information in speech signal analysis and synthesis,” *Speech Communication*, vol. 1, no. 1, pp. 9–20, May 1982.
- [10] K. K. Paliwal and W. B. Kleijn, *Speech Synthesis and Coding*, chapter Quantization of LPC parameters, pp. 433–466, Elsevier Science Publ., Amsterdam, the Netherlands, 1995.
- [11] <http://my.fit.edu/~vkepuska/web1/>