2020 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)

# Low-Power Low-Cost Audio Front-End for Keyword Spotting

Daljit Josh*, John-Anthony Elenis, Heman Muresan, Petros Spachos and Stefano Gregori*
*School of Engineering, University of Guelph*
Guelph, Ontario, N1G 2W1, Canada
*Email: {djosh, sgregori}@uoguelph.ca

***Abstract*—This paper presents a low power audio front end for keyword spotting. A multi-stage approach is used to reduce the power consumption of the system by only using different stages when they are required. A working prototype was created and tested to verify its functionality. The effectiveness of the multi-stage approach is shown by comparing the power consumption of the system in its idle state to the systems active state. The prototype has a power consumption of 4.1 mW in the idle state that can be reduced below 3 mW with a keyword detection accuracy of 87%.**

***Index Terms*—Voice activity detection, wake on feature, speech recognition, neural network**

## I. Introduction

Speech recognition is incorporated in a growing number of products, such as intelligent virtual assistants, that perform tasks based on verbal commands. These systems rely on edge or cloud computing to distribute the computational tasks.

A device generally listens for a phrase before the keyword detection is performed. The requirements of mobile and wearable applications introduce the importance of power consumption. Therefore, adding a dedicated low-power microcontroller (MCU) for handling the keyword spotting and for waking the device upon detection is beneficial. Because virtual assistants spend the majority of the time waiting for a keyword, further power savings can be achieved if the waiting state only needed to perform sound detection instead of keyword recognition.

In this paper, a three-stage approach is implemented. The first stage performs level detection. If the sound level is above a specified threshold, then the second stage is activated. In this stage, pitch identification is used to verify whether a human voice is present. In this case, the third stage performs the keyword recognition using a neural network (NN). This multistage system reduces the time for power-hungry tasks, and in turn decreases the overall power consumption.

The following section briefly reviews some existing solutions. Section III describes the development of a prototype and covers the design of the three stages in depth. Finally, the results of the experimentation are discussed in Section IV and a summary of the paper is given in Section V.

## II. Literature Review

Voice activity detection (VAD) and speech recognition have been researched extensively. Performing VAD at an application level (desktop computers) is a common task. Dedicated VAD devices are always-on, therefore their power consumption is minimized while sacrificing as little accuracy as possible. However, obtaining the desired performance is increasingly challenging as the microprocessor power is scaled down. Viable solutions are based on multiple stages, where each stage increases in power consumption and computational complexity. In a previous implementation of a voice activity detector with two stages [1], the first stage performs an analog feature extraction and, if satisfied, a digital classification is done using an event driven analog-to-digital conversion (ADC). Additionally, an ADC can be used for feature extraction while keeping a detection accuracy of 90% [2]. Other solutions also rely on multistage schemes for minimizing power consumption and ASIC implementations to improve the efficiency further [3], [4].

The main approaches to perform pitch detection are: time domain analysis, frequency domain analysis, and time-frequency domain analysis. The time-domain approaches include the short-time average magnitude difference function (AMDF) and the autocorrelation function (ACF). The calculation complexity is similar for both cases with AMDF requiring more summations and ACF using more multiply-accumulate operations [5]. ACF will quantify the waveform to a time delayed version of itself. When the time delay is equal to the desired harmonics, a relationship between the two is observed [6]. Normally, the calculation for ACF is computationally intensive. However, the autocorrelation function of a signal can be calculated by taking the inverse Fourier transform of the Fourier transform multiplied by its conjugate as stated by the Wiener-Khinchin theorem [7]. This implementation is effective when the MCU of the second stage contains a fast-Fourier-transform hardware accelerator.

Machine learning is an appropriate method for implementing speech detection due to the complexity of human speech. Different neural networks are used depending on the limitations of the implementation, such as processing power or speed. Speech detecting neural networks take the features of a signal as an input and produce probabilities as an output [8]. For keyword spotting, the neural network will have two possible outputs. One representing the correct keyword being detected and the other for an incorrect keyword. Different types of neural networks require different amount of physical resources and have an effect on the energy consumption [9].

The aim of this work is to take a new approach to this problem by combining the features of previous models and explore an implementation with discrete components.

## III. System Design

### A. Level Detection

The purpose of the first stage is to perform threshold detection. This is done by filtering out ambient noise and creating an approximate digital signal from the analog audio. The microphone used was a SparkFun Sound Detector (SEN-14262). This microphone was chosen due to its simplicity in implementation. It contains a few critical features such as an envelope follower, a gate signal and a Schmitt trigger, which reduce the overall complexity of the first stage.

The audio signal from the microphone is amplified before being fed to the envelope follower. The amplitude of the voltage waveform proportional to the power of the audio signal is used to trigger the second stage using the Schmitt trigger.

Lastly, a low-pass filter rejects higher unwanted frequencies. The cutoff frequency chosen was 3 kHz due to the frequency range of the human voice. After testing the implementation for both an active and a passive filter, the passive filter was chosen due to the reduced power consumption with the results on the audio output not affecting the functionality.

The effect of the audio filter is shown in Fig. 1 using Audacity. Audacity is an open-source software used for audio manipulation which can plot spectrograms of audio signals. The spectrograms in Fig. 1 are from an audio sample of someone saying the command “light on.” The frequency is along the y axis and the time along the x axis. Red and purple areas signify a large magnitude, while blue sections represent smaller magnitude. The unfiltered spectrogram has much higher magnitudes in the higher frequencies of the spectrogram. Additionally, the signal is shown being attenuated significantly past 3 kHz, which demonstrates the effectiveness of the passive filter in this application.

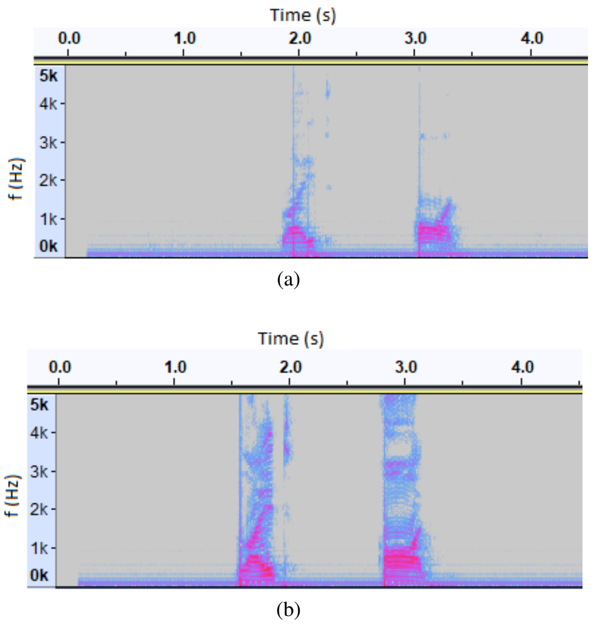

Fig. 1. Spectrogram of the phrase “light on” captured from Audacity. (a) Filtered output. (b) Unfiltered output.

### B. Pitch Detection

The second stage is responsible for verifying if a human voice is present in the audio signal. This is achieved by performing pitch detection on the signal and then comparing the pitch with the known frequency range of the human voice. The pitch detection is performed by taking the autocorrelation of a block of data and then locating the peaks in the resulting data. In this application, a block size of 128 samples and a sampling rate of 4 kHz were used. Pitch detection includes the following steps,

1: Apply a window function to the data (a Hanning window was used)
2: Zero pad the signal to twice the length
3: Perform a discrete Fourier transform (DFT) on the data
4: Multiply the data by the conjugate of itself
5: Perform the inverse discrete Fourier transform (IDFT)
6: Observe the first half of the resulting data (the second half of the data is mirrored)

In order to make this stage feasible, a development board with an onboard hardware accelerator is required to assist in the complex computations. The MSP430FR5994 from Texas Instruments fits all of the requirements. The MSP-EXP430FR5994 is the corresponding development board. The algorithm implemented on this board can be made stricter by limiting the peak detection frequency range to a smaller range to help reduce false activations of the third stage.

### C. Keyword Detection

The most power intensive stage is the final one, which is the keyword detection. The audio is sampled at 16 kHz and the Mel frequency cepstral coefficients (MFCC) are calculated to extract frequencies as input features to a neural network. It is important to select a neural network that is able to reach 90% accuracy specification while minimizing the processor resources. From previous literature and in our own experiments, we found that a relatively small depthwise separable neural network (DS-CNN) is ideal to use [8]. The structure of the network consisted of one convolutional input layer followed by four depthwise separable convolutional layers of decreasing size and a fully connected hidden layer containing 64 neurons. The output from the fully connected layer is piped through a SoftMax function which represents the keyword as a set of probabilities. Fig. 2 illustrates the structure of the neural network. The DS-CNN offers above 90% accuracy with minimal computational costs due to CNN’s ability to map spatial and temporal relations between input frequencies.

The neural network first had to be trained on a workstation before being implemented on the MCU. Google’s TensorFlow GPU API was used for training along with the preprocessing scripts provided in their audio recognition example. The dataset consisted of 105,000 one-second audio clips with a 80%, 10%, 10% split for the training, validation and testing

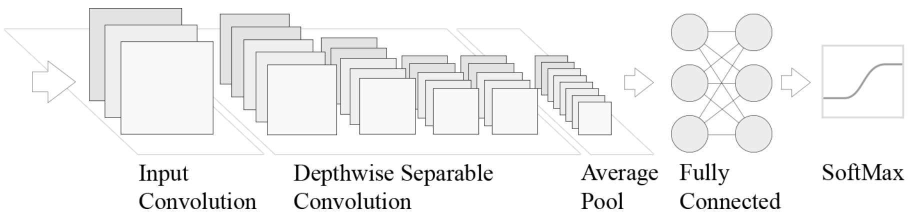

Fig. 2. The structure of DS-CNN algorithm.

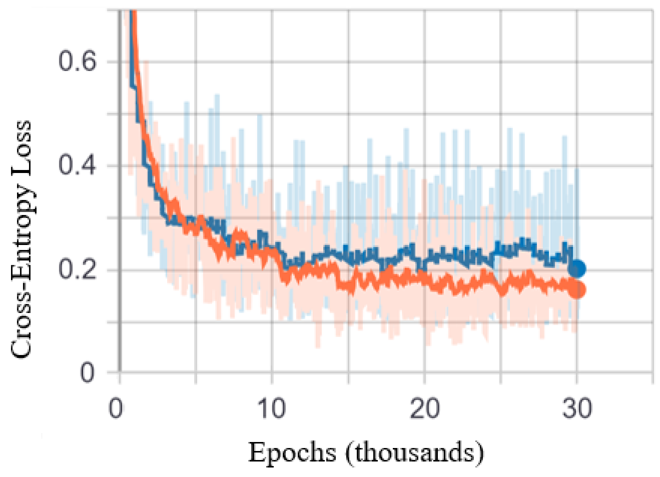

Fig. 3. The cross-entropy of the cost function for the DC-CNN trained over 30,000 epochs. Training in orange and validation in blue.

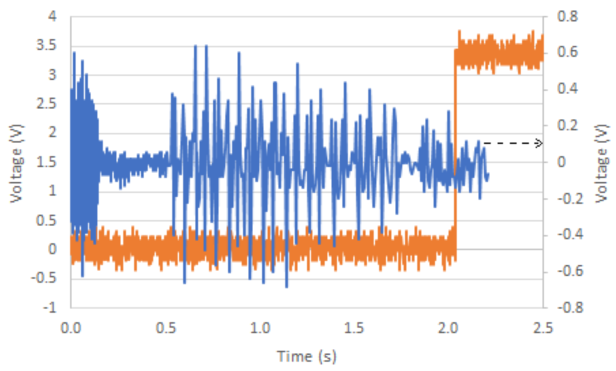

Fig. 4. K64F output transition after the command “light on” is detected. Audio signal in blue and system output in orange.

sets. The network completed its training epochs for 2.5 hours before converging using a workstation running a I7-6700K CPU, a GTX-1080 GPU and 16 GB of memory. A single training epoche is a full cycle through the entire training dataset. TensorBoard was used to provide a visualization for the cross-entropy error of the neural network (NN) for both the training and validation sets. It provides a way of calculating how incorrect the average prediction is. Inspection of the networks cross-entropy error in Fig. 3 shows the network converging to an accuracy of 94% at approximately 15,000 epochs. This maximum accuracy is obtained over the validation set when it overfits. Overfitting yeilds a high accuracy because the model is simply memorizing the training data. However, this performance will not generalize and work with real word data. A python script was used to select a checkpoint to reduce the effect of overtraining. More information regarding the neural network structure is available in [8]. After training, the network is exported to the MCU by quantizing the floating-point weights to integers for fixed point use. The same structure is then implemented using the optimized facilities for DSP, and NN calculations are provided by ARM in the CMSIS 5 library. The MCU for this stage chosen was the MK64FN1M0VLL12 and the FRDM-K64F is the corresponding development board. This board was chosen due to the low power MCU and the large on board memory which is required for the NN.

The algorithm created for this stage has a strict recording window of two seconds upon wakeup. There are 32,000 samples within one recording window which the program divides into two frames. The width of the sliding window is half the size of a frame. In this system the window shifts by 1,700 samples after every calculation. After a frame has been checked the NN returns its confidence interval on the data. If the word “light” is detected the program continues to look through the remaining frame for another comand, otherwise it returns to sleep until it is woken up again.

## IV. Results and Assessment

The functionality of the system was tested with live data, specifically two commands “light on” and “light off.” When told to turn the light on, the output would become high turning on a light bulb by means of a relay and remain on until the system is reset or told to turn off. Random phrases were also said to ensure proper functionality of the NN. Fig. 4 shows the last second of recording and the output transitioning from low to high after it successfully detects a command.

A total of nine participants were asked to command the system for ten samples each to assess the accuracy of the neural network with typical office noise in the background. The users were asked to wake up the device and activate it by saying the phrase “light on.” If it was successfully detected, they would then be asked to re-wake the system and say “light off.” One on/off cycle is considered to be one sample. The sample is considered a failure if either one of the commands is not recognized. The collection of data began after the users familiarized themselves with the system for one minute.

TABLE I
POWER CONSUMPTION

| Stage | Idle Power (mW) | Active Power (mW) |
|---|---|---|
| 1 | 2.45 | 4.60 |
| 2 | 0.607 | 67.20 |
| 3 | 0.107 | 81.62 |
| Total | 4.11 | 153.42 |

There were two testing sessions held with different participants of varying background. After aggregating the results, the neural network was accurate 87% of the time. This accuracy is close to our expected 90% from the training of the neural network. The main source of error in these trials came from the two words being said to closely together resulting in the speech wave looking like one long word instead of two distinct ones.

Minimizing power consumption is very important in this application as it is essential to the longevity of the system life before batteries must be replaced. Both microcontrollers (FRDM-K64F and MSP-EXP430FR5994) were placed into very low power modes to minimize the power consumption. A digital multimeter was used to measure the current through each individual stage.

The power draw for an individual stage can be calculated by taking the measured current and multiplying it by the respective operating voltage. The microphone has an operating voltage of 3.5 V and draws 700 $\mu$A while idle. The active power for the microphone was measured at 1.3 mA and taken in a worst case scenario where it is always picking up audio. The pitch detection process draws 177.3 $\mu$A while idle and 1.92 mA when active. The keyword detection process draws 625 $\mu$A while idle (VLPS mode) and 23 mA when active. The operating voltage for this stage ranges from 1.71 V (idle) to 3.6 V (active). The total power consumption for the system while idling is 4.11 mW as shown in Table I. Idle power for a stage is defined as the system on standby, while active power is when the stage is awake performing its respective function.

The power consumption of the K64F can be further reduced by enabling very low leakage stop (VLLS) mode. In this mode most of the modules are disabled and must be re-enabled on wakeup. With more optimization both microcontrollers could enter VLLS mode thus ideally reducing the power consumption below 3 mW. Another thing that can be considered to minimize the power consumption would be to undersample the audio. Undersampling the signal will reduce the power consumption of the ADC at the cost of accuracy. Another method that can be used is to raise the threshold for voice detection. Raising the threshold reduces the amount of times the system switches from the idle to active state thus reducing power consumption. As the threshold rises, it makes the use of the system more inconvenient which is less desirable in VAD applications.

## V. CONCLUSION

This paper presented a low-power audio multistage system for keyword spotting using microcontrollers. The functionality of the system was determined through physical experimention. The overall power consumption of the system was calculated to be 4.11 mW when in the idle state. The ideal power consumption of our proposed system is in the 3-5 mW range when in the idle state and 250 mW when in an active state after optimization. By reducing the time of power intensive operations, this multistage approach provides a means to significantly reduce total power consumption. It was found that power savings can be further improved by raising the threshold during audio sampling and reducing the systems sensitivity to voices.

## REFERENCES

[1] M. Yang, C. Yeh, Y. Zhou, J. P. Cerqueira, A. A. Lazar, and M. Seok, "Design of an always-on deep neural network-based 1-$\mu$W voice activity detector aided with a customized software model for analog feature extraction," *IEEE J. Solid-State Circuits*, vol. 54, no. 6, pp. 1764–1777, Jun. 2019.

[2] S. Lecoq, J. L. Bellego, A. Gonzalez, B. Larras, and A. Frappé, "Low-complexity feature extraction unit for "wake-on-feature" speech processing," in *Proc. IEEE Int. Conf. Electron. Circuits Syst. (ICECS)*, Dec. 2018, pp. 677–680.

[3] S. Lauwereins, W. Meert, J. Gemmeke, and M. Verhelst, "Ultra-low-power voice-activity-detector through context- and resource-cost-aware feature selection in decision trees," in *Proc. IEEE Int. Workshop Mach. Learning Signal Process. (MLSP)*, Sep. 2014, pp. 1–6.

[4] M. Price, J. Glass, and A. P. Chandrakasan, "A low-power speech recognizer and voice activity detector using deep neural networks," *IEEE J. Solid-State Circuits*, vol. 53, no. 1, pp. 66–75, Jan. 2018.

[5] S. Kumar, "Performance evaluation of novel AMDF-based pitch detection scheme," *ETRI J.*, vol. 38, no. 3, pp. 425–434, 2016. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.4218/etrij.16.0115.0926

[6] S. Hamzenejadi, S. A. Y. Hosseini Goki, and M. Ghazvini, "Extraction of speech pitch and formant frequencies using discrete wavelet transform," in *Proc. Iranian Joint Congr. Fuzzy Intell. Syst. (CFIS)*, Jan. 2019, pp. 1–5.

[7] L. Cohen, "The generalization of the Wiener-Khinchin theorem," in *Proc. IEEE Int. Conf. Acoustics Speech Signal Process. (iCASSP)*, vol. 3, May 1998, pp. 1577–1580.

[8] Y. Zhang, N. Suda, L. Lai, and V. Chandra, "Hello edge: Keyword spotting on microcontrollers," *CoRR*, vol. abs/1711.07128, 2017. [Online]. Available: http://arxiv.org/abs/1711.07128

[9] R. Tang, W. Wang, Z. Tu, and J. Lin, "An experimental analysis of the power consumption of convolutional neural networks for keyword spotting," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Apr. 2018, pp. 5479–5483.