Quantized 1D-CNN for a Low-power PDM-to-PCM Conversion in TinyML KWS Applications

Paola Vitolo*, Gian Domenico Licciardo*, Anna Chiara Amendola*, Luigi Di Benedetto*,
Rosalba Liguori*, Alfredo Rubino*, and Danilo Pau**
Email: {pvitolo, gdlicciardo,ldibenedetto,rliguori,arubino}@unisa.it a.amendola28@studenti.unisa.it danilo.pau@st.com
* *Department of Industrial Engineering, University of Salerno*, Fisciano (SA), Italy
** *System Research and Applications, STMicroelectronics*, Agrate Brianza (MI), Italy

***Abstract*—This paper proposes a novel low-power HW accelerator for audio PDM-to-PCM conversion based on artificial neural network. The system processes samples from a digital MEMS microphone and converts them in PCM format by using a 1-Dimensional Convolutional Neural Network (1D-CNN). The model has been quantized to reduce the computational complexity while preserving its Signal-to-Noise Ratio (SNR) and the HW accelerator has been designed to minimize the physical resources. The SNR achieved is 41.56 dB while the prototyping of the design on a Xilinx Artix-7 FPGA shows a dynamic power consumption of 1 mW and a utilization of 606 LUTs and 410 FFs. These results enable the proposed system to be the first step of a tiny low-power end-to-end neural network-based Keyword Spotting (KWS) system.**

Keywords— PDM-to-PCM conversion, neural network, keywork spotting, FPGA, low power

I. Introduction

With the rapid spread of smartphones, digital assistants, tablets and other smart devices, the use of voice has become a common method of interacting with technology. The report in [1] shows that the smart speaker market was worth approximately USD 7.1 billion in 2020 with an expected growth rate of around 17% over the period 2020-2025. Voice user interface is based on Automatic Speech Recognition (ASR), which has increasingly used Artificial Intelligence (AI) and, in particular, Deep Learning (DL) over the past decade [2]. However, DL requires a high computational effort and memory access operations, while battery-powered smart devices have stringent constrains in terms of power consumption and area. Therefore, DL-based speech recognition systems are usually performed by using cloud resources, which, however, introduce issues related to service availability and bandwidth. In this regard, KWS, which exploits edge computing, is a possible solution. It is a tiny always-on system, enough energy-saving to be deployed to edge devices, devoted to the detection of some wake words, which then activates the much more energy-hungry function blocks in the cloud to accomplish speech recognition. Digital Micro-Electrical-Mechanical System (MEMS) microphones are best suited in smart applications due to their extremely low cost, noise robustness, and compactness compared to analog alternatives, which instead require an external analogue to digital converters (ADCs) and amplifiers, resulting in greater bulk and cost [3]. The output signals from digital MEMS microphones are encoded with Pulse Density Modulation (PDM) by using a sigma-delta oversampling ADC, which consists of a one-bit quantizer in the frequency range of GHz. PDM signals must be converted to an easier to manipulate Pulse Code Modulation (PCM) encoding to interface with traditional audio processing systems. PCM uses sampling frequencies in the kHz range with a bit depth ranging from 8 to 32 bits. Therefore, PDM-to-PCM conversion requires downsampling and alias-rejection filtering operations, which are challenging due to high values of the decimation factor. Computationally efficient decimation filters are based on Cascaded-Integrator-Comb (CIC) filters since they do not require multipliers and memory for the filter coefficients [4]. However, CIC filters show a poor cut-off and need to be compensated with more complex Finite Impulse Response (FIR) filters to suppress aliasing [4], [5]. Traditional FIR filter designs are based on windowing, optimization methods, and approximation via truncation of impulse response [6]. Emerging alternative designs use DL methods, exploiting the capabilities of Neural Networks (NNs) to be universal approximators of even complex and non-linear functional relationships [7]-[9]. In [7] the authors propose a FIR filter design based on a single-layer NN trained with the aim of minimizing the magnitude response. NN-based filter response is improved in [8] by initializing weight values and inserting additional factors into the error function to make the priority of enhancing passband, transition, or stopband performance flexible. In [9] a generative adversarial network (GAN) is suggested to design various FIR filters (e.g., low pass, band pass, high pass filter) with any cut-off frequency using the ideal time-domain filter function as the input to the generator of the GAN. However, existing NN-based approaches do not investigate the design of *decimation* filters, which is essential in PDM-to-PCM conversion as they determine the quality of the signals passed from MEMSs to audio processing systems, and hence are fundamentals for realizing compact tinyML KWS systems.

This work proposes a novel PDM-to-PCM HW converter based on a tiny 1D-CNN, which, for the first time in the literature, includes a decimation filter, and exploits a quantization scheme to achieve a good trade-off between the number of physical resources and the SNR. The converter has been devised to enable the system to be joined with a low-power DL-based TinyML KWS application, e.g. [10], realizing an end-to-end KWS system that takes as input the MEMS microphone output and outputs the probability that a given command is present.

Although quantized, the proposed system achieves a SNR of 41.56 dB while it shows a dynamic power consumption of 1 mW and a utilization of 606 LUTs and 410 FFs when implemented on a Xilinx Artix-7 FPGA.

II. The Proposed Design

As shown in Fig. 1, the proposed PDM-to-PCM converter consists of a 1D-CNN model, which has been chosen because it has fewer network parameters and, consequently, requires less memory than fully connected (FC) layers. Furthermore, convolutional layers are well suited for implementation through an iterative architecture, resulting in a smaller

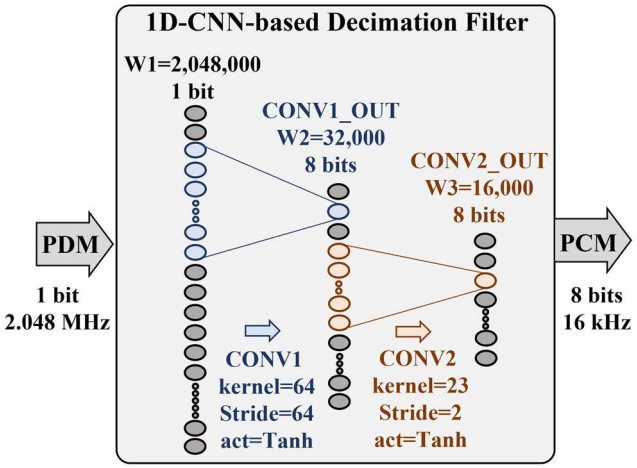

Fig. 1. Schema of the proposed 1D-CNN based decimation filtering system.

occupied area with an acceptable increase in latency [11]-[17]. Moreover, the CNN stride can be used for decimation.

We have considered as input a PDM signal with a sampling rate of 2.048 MHz, which is a usual output sampling rate of digital MEMS microphones [18], and as output a PCM signal with a sampling rate of 16 kHz and a bit depth of 8 bits. This output is suitable as an input for a TinyML KWS system that can be integrated into a MicroController Unit (MCU). An example of these systems is the quantized system for audio wake words available, already trained, in [10]. It is composed of a Mel-Frequency Cepstral Coefficient (MFCC) feature extraction block and a Separable Depthwise 2D CNN. As reported in [10], this system achieves an accuracy of 92% over twelve classes.

A. Model

As shown in Fig. 1, the 1D-CNN input window is composed of W1 = 2,048,000 samples, corresponding to 1 second. Each sample is encoded with 1 bit, consistently with PDM. The model consists of two convolutional layers (CONV), with 1 channel, same padding, and tanh function (1) as activation function. The kernel sizes are 64 and 23 for CONV1 and CONV2, respectively. The stride of CONV1 has been set to 64 while the stride of CONV2 is 2. Therefore, CONV1 performs a decimation by a factor of 64 with a consequent output shape of 2,048,000/64 = 32,000, while CONV2 decimates by 2 and its output shape is 32,000 / 2 = 16,000, with an overall decimation factor of 64×2 = 128. The output of CONV1 and CONV2 are encoded with 8 bits.

$$y = tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \quad (1)$$

B. Dataset

A custom dataset has been used for training and evaluating the proposed model. Since the target application is KWS, the dataset has been created using as labels an extract of PCM values from Google Speech Commands Dataset (GSCD) [19] and as features the corresponding PDM values. The GSCD consists of 105,829 utterances of 35 words. Each utterance duration is 1 s (or less) and each sampling data is encoded as 16-bit PCM value at 16 kHz rate. In this work, we have considered 11 of 12 classes selected in [10]. As can be seen in Table I, they are composed of 10 command words and 1 unknown class that contains words not belonging to the above 10 classes. The features corresponding to the PCM utterances have been obtained through the Delta Sigma Toolbox [20] in Matlab, setting an order of sigma delta ADC of 4 and an OverSampling Ratio (OSR) of 128.

TABLE I. NUMBER OF RECORDINGS OF EACH WORD OF THE DATASET CREATED

Word	Down	Up	Left	Right	Yes	No	Go	Stop	Off	On	Unk
Number Of Utterances	40	40	40	40	40	40	40	40	40	40	40

C. Training and Evaluation

To evaluate the proposed system, a traditional CIC-based decimation filter has been designed in Matlab following the filtering chain presented in [21]. The resulting block diagram of the system and the relative magnitude response are shown in Fig. 2.

A custom loss function, Fast-Fourier-Transform Mean Absolute Error (FFT-MAE), has been created with the aim of approximating the magnitude response of the desired decimation filter presented in Fig. 2. The function returns the mean absolute error between the FFT of model outputs and the FFT of the corresponding labels.

The proposed 1D-CNN-based filter has been modeled and trained using TensorFlow (TF) [22] framework.

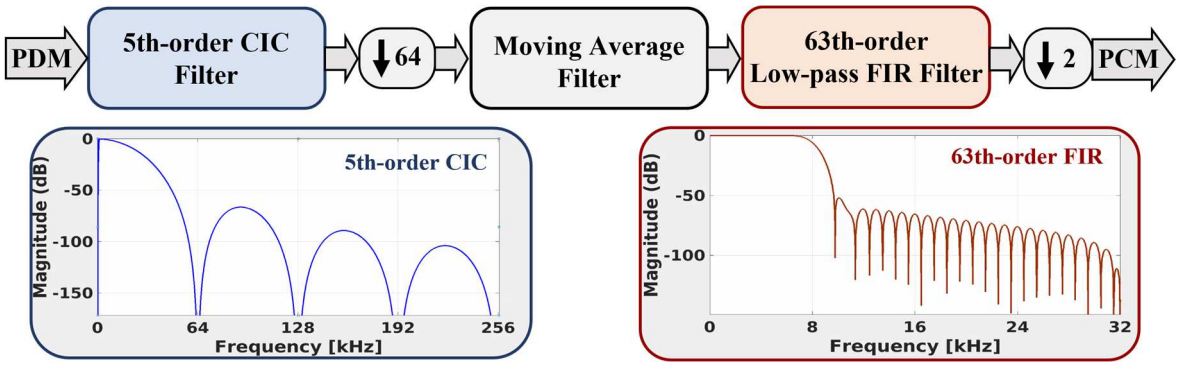

Fig. 2. Block diagram of traditional filtering chain with an input sampling rate of 2.048 MHz and an output sampling rate of 16 kHz for PDM signals generated by a fourth-order sigma-delta ADC.

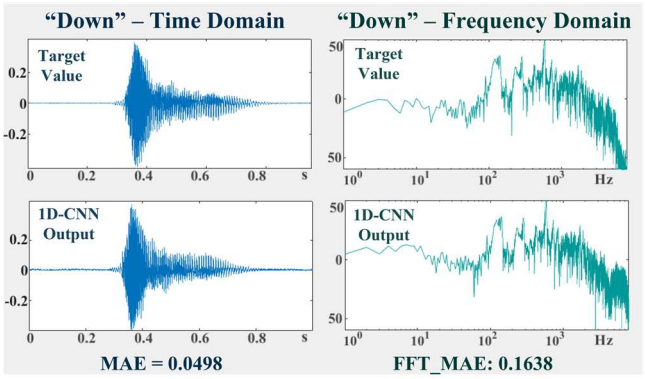

Fig. 3. Time Domain (left) and frequency domain (right) representations of an audio signal "Down". The target value is represented at the top while the output of the proposed model is shown at the bottom.

Subsequently, the weights, biases and activations of the TF model have been quantized to 8 bits using QKeras [23] framework, and the quantized model has been fine tuned. The custom dataset has been divided into training (80%), validation (10%), and test (10%) datasets. The number of epochs has been set to 150. To evaluate the model, FFT-MAE and MAE have been calculated on the test dataset, achieving 0.19 and 0.054, respectively. The SNR achieved at a frequency of 1 kHz is 41.56 dB, which is about 13% lower than the theoretical maximum SNR with a bit depth of 8 bits. These results represent a good trade-off between the number of employed physical resources and the accuracy of the output signals for KWS applications. Indeed, the proposed filter has been used as input block of the tinyML KWS system, already trained, available in [10]. In particular, the MFCCs have been calculated from the PCM outputs of our system and they have been sent to the KWS model, achieving an accuracy of 89% using our dataset.

Fig. 3 shows an output example of the proposed model, represented in the time and frequency domain, and the resulting MAE and FFT_MAE, respectively. Table II reports the memory for storing the parameters and the number of operators required by the proposed model and by the traditional CIC-based filter of Fig. 2. Although the number of adders required by our proposal is slightly greater than the traditional filter, the multipliers and memory are an order of magnitude lower, resulting in lower computational complexity and resources. Consequently, our system is more suitable for HW implementation in contexts with limited resources, such as in KWS applications at edge devices.

TABLE II. MEMORY FOR PARAMETERS AND OPERATORS REQUIRED BY THE PROPOSED SYSTEM AND CIC-BASED FILTER

	ADDs per window	MULTs per window	Parameters [Bytes]
CIC-based Filter	4,064,000	2,016,000	252
Proposed System	4,544,000	368,000	89

III. HARDWARE ARCHITECTURE

The HW architecture of the proposed system is schematized in Fig. 4a. It is composed of two main blocks: a Control Unit (CU), consisting of a Finite State Machine

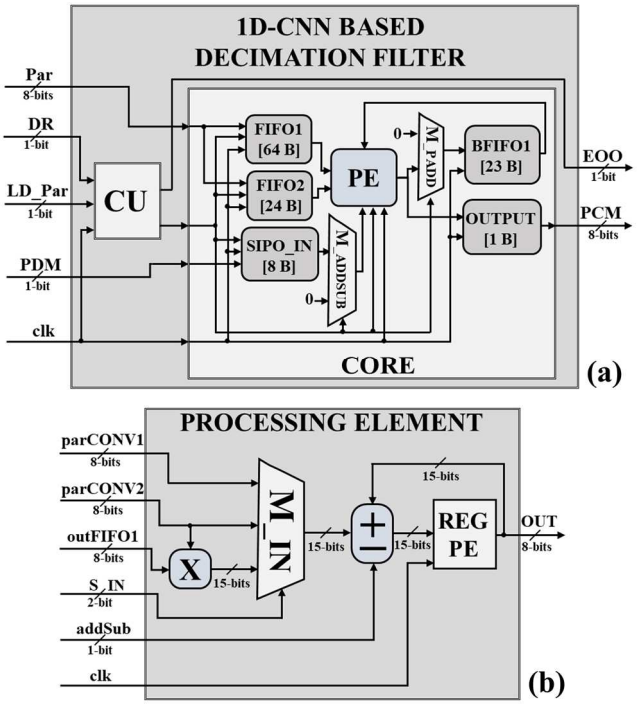

Fig. 4. Block diagram of: (a) the HW design of the proposed 1D-CNN based decimation filter; (b) the processing element.

(FSM) which generates the control signals for managing the flow of data, and a Core, which recursively implements all the layers of the network. As shown in Fig. 4a, the Core consists of a Processing Element (PE) which performs all the necessary operations, a glue logic to properly route the signals, and memory elements to store the NN parameters (FIFO1 and FIFO2) and partial results (BFIFO1), and to buffer the incoming data (SIPO_IN). SIPO_IN, BFIFO1, FIFO1 and FIFO2 store 8, 23, 65 and 24 bytes, respectively. During startup, the network parameters (weights and biases) of CONV1 and CONV2 must be loaded into the corresponding FIFOs. Subsequently, the CU configures the FIFOs as circular buffers for the rest of the time. BFIFO1 is set as shifter register when it must be written while it is a circular buffer when it

TABLE III. FPGA RESULTS AND COMPARISONS

		CIC-based Filter	Proposed System
Design Specification	Input Freq. [MHz]	2.048	2.048
	Output Freq. [kHz]	16	16
	OSR	128	128
Clk Freq. [MHz]		83.33	83.33
LUTs		744	606
FFs		812	410
DSPs		1	0
Dyn. Power [mW]		7	1

must be read. As shown in Fig. 4b, the PE consists of a multiplier, an adder, and a register to store the output. The arithmetic coding is 15-bit fixed point (4.11) to account for the increase in bit depth during the convolutions. The system can take an input sample each 27 clock cycles and provides an output after 91 or 28 clock cycles, depending on whether padding is required.

IV. FPGA Implementation Results

The proposed HW design has been implemented on a Xilinx Artix-7 (xc7a35tfgg484-1) FPGA by using Xilinx Vivado Design Suite. To evaluate our design, we have implemented the traditional filter design of Fig. 2 on the same FPGA using the Xilinx LogiCORE IP CIC compiler core [24] and the Xilinx LogiCORE IP FIR compiler [25], with the aim of comparing the two designs. The clock frequency has been set at 83.33 MHz, which ensures a real-time processing. As reported in Table III, although the number of LUTs occupied by our proposal is 606, slightly less than the alternative, the number of FFs results about 43% lower than the traditional design while not using DSPs. In addition, the dynamic power consumption of the proposed filtering system is 1 mW, which is 7 times lower than its counterpart. Consequently, our custom design overcomes the traditional filter design by requiring fewer physical resources. These results enable the system to be combined with a low-power DL-based TinyML KWS application, creating an end-to-end KWS system that can be deployed to the edge.

V. Conclusions

This paper proposes a new *decimation* filter for audio PDM-to-PCM conversion by using a 1D-CNN. A custom loss function has been realized with the aim of minimizing the magnitude response and a custom dataset based on GSCD has been created. The proposed NN-based filter has been modeled with TensorFlow and 8-bit quantized with QKeras. The proposed filtering has been accelerated with a custom HW design, which exploits an iterative architecture to reduce the required physical resources. The FPGA implementation results overcome the traditional filtering design in terms of number of mapped physical resources and prove that the proposed system can be the first step towards a CNN-based end-to-end KWS deployable to the edge.

References

[1] "Smart Speaker Market with COVID-19 Impact Analysis by IVA (Alexa, Google Assistant, Siri, DuerOS, Ali Genie), Component (Hardware (Speaker Driver, Connectivity IC, Processor, Audio IC, Memory, Power IC, Microphone) and Software), Application, and Region - Global Forecast to 2025." [Online]. Available: https://www.researchandmarkets.com/reports/5116500/smart-speaker-market-with-covid-19-impact.

[2] A. B. Nassif, I. Shahin, I. Attili, M. Azzeh, and K. Shaalan, "Speech Recognition Using Deep Neural Networks: A Systematic Review," IEEE Access, vol. 7, pp. 19143–19165, 2019, doi: 10.1109/ACCESS.2019.2896880.

[3] E. Zwyssig, M. Lincoln, and S. Renals, "A digital microphone array for distant speech recognition," in ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings, pp. 5106–5109, 2010, doi: 10.1109/ICASSP.2010.5495040.

[4] B. P. Stosic, "Improved Classes of CIC Filter Functions: Design and Analysis of the Quantized-Coefficient Errors," 2021 56th Int. Sci. Conf. Information, Commun. Energy Syst. Technol., pp. 65–68, 2021, doi: 10.1109/ICEST52640.2021.9483471.

[5] E. B. Hogenauer, "An Economical Class of Digital Filters for Decimation and Interpolation," IEEE Trans. Acoust., vol. 29, no. 2, pp. 155–162, 1981, doi: 10.1109/TASSP.1981.1163535.

[6] J. Proakis and D. Manolakis, Digital Signal Processing: Principles, Algorithms, and Applications, 4th ed. Prentice Hall, 2007.

[7] K. Pachori and A. Mishra, "Design of FIR digital filters using ADALINE neural network," Proc. - 4th Int. Conf. Comput. Intell. Commun. Networks, CICN 2012, no. 3, pp. 800–803, 2012, doi: 10.1109/CICN.2012.93.

[8] D. A. Alwahab, D. R. Zaghar, and S. Laki, "FIR Filter Design Based Neural Network," 2018 11th Int. Symp. Commun. Syst. Networks Digit. Signal Process. CSNDSP 2018, no. July, pp. 1–4, 2018, doi: 10.1109/CSNDSP.2018.8471878.

[9] M.-S. Koh, "Learnable Linear Phase FIR Filter Designs Using a Generative Adversarial Network," pp. 1–8, 2021, doi: 10.1109/icspcs53099.2021.9660300.

[10] MLCommons, "Pre-Trained Audio Wakeword Models." https://github.com/mlcommons/tiny/tree/v0.5/v0.5/training/keyword_spotting/trained_models.

[11] A. De Vita et al., "A Partially Binarized Hybrid Neural Network System for Low-Power and Resource Constrained Human Activity Recognition," in IEEE Trans. Circuits and Syst. I: Reg. Papers, vol. 67, no. 11, pp. 3893-3904, Nov. 2020.

[12] A. De Vita et al., "Low-Power HWAccelerator for AI Edge-Computing in Human Activity Recognition Systems," in Proc. 2020 IEEE Int. Conf. Artif. Intell. Circuits Syst. (AICAS), pp. 291–295, 2020.

[13] A. De Vita et al., "Low Power Tiny Binary Neural Network with improved accuracy in Human Recognition Systems," 2020 23rd Euromicro Conference on DSD, pp. 309-315, 2020.

[14] G. D. Licciardo, C. Cappetta, L. Di Benedetto, A. Rubino , R. Liguori, "Multiplier-Less Stream Processor for 2D Filtering in Visual Search Applications," in IEEE Trans. Circuits Syst. Video Technol., vol. 28, no. 1, pp. 267–272, Jan. 2018.

[15] G. D. Licciardo, C. Cappetta, L. Di Benedetto, M. Vigliar, "Weighted Partitioning for Fast Multiplierless Multiple-Constant Convolution Circuit," in IEEE Trans. Circuits Syst. II Express Briefs, vol. 64, no. 1, pp. 66–70, Jan. 2017.

[16] G. D. Licciardo and M. Costagliola, "An H.264 Encoder for Real Time Video Processing Designed for SPEAr Customizable System-on-Chip Family," 2007 IEEE International Conference on Signal Processing and Communications, Dubai, pp. 824-827, 2007.

[17] P. Vitolo, G. D. Licciardo, L. di Benedetto, R. Liguori, A. Rubino and D. Pau, "Low-Power Anomaly Detection and Classification System based on a Partially Binarized Autoencoder for In-Sensor Computing," 2021 28th IEEE International Conference on Electronics, Circuits, and Systems (ICECS), pp. 1-5, 2021.

[18] STMicroelectronics, "MEMS audio sensor omnidirectional digital microphone for industrial applications." IMP34DT05 - Rev 4 - June 2021.

[19] P. Warden, "Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition," 2018, arXiv: 1804.03209. [Online]. Available: http://arxiv.org/abs/1804.03209.

[20] R. Schreier, "Delta Sigma Toolbox," MATLAB Cent. File Exch., 2022, [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/19-delta-sigma-toolbox.

[21] B. Da Silva, L. Segers, A. Braeken, K. Steenhaut, and A. Touhafi, "Design exploration and performance strategies towards power-efficient FPGA-Based architectures for sound source localization," J. Sensors, vol. 2019, 2019, doi: 10.1155/2019/5761235.

[22] "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, [Online]. Available: https://www.tensorflow.org/.

[23] "QKeras," [Online]. Available: https://github.com/google/qkeras.

[24] Xilinx, "Xilinx LogiCORE IP CIC Compiler." DS845 June 22, 2011.

[25] Xilinx, "Xilinx LogiCORE IP FIR Compiler." PG149 Jan 21, 2021.