THÈSE PRÉSENTÉE
POUR OBTENIR LE GRADE DE

DOCTEUR
En
« Électronique, Microélectronique, Systèmes intégrés »

À l'Université de Lille
École Doctorale Des Sciences de l'Ingénierie et des Systèmes
(ENGSYS-632)

Event-driven continuous-time signal processing in 28nm FD-SOI CMOS technology for ultra-low power keyword spotting application

Traitement du signal en temps continu piloté par les événements dans la technologie 28nm FD-SOI CMOS pour les applications de détection de mots clés à très faible consommation

Présentée et soutenue publiquement par :

Soufiane Mourrane

Le 11 Décembre 2023

Jury:

Mme Marian VERHELST	Professeure	Katholieke Universiteit Leuven	Rapportrice
M. Joachim RODRIGUES	Professeur associé	Lund University	Rapporteur
M. Christoph STUDER	Professeur associé	ETH Zurich	Examinateur
M. Pascal VIVET	Directeur scientifique	CEA-LIST	Examinateur
M. Laurent CLAVIER	Professeur: Président de Jury	Université de Lille	Examinateur
M. Andreas KAISER	Directeur de recherche	Université de Lille/Junia	Examinateur
M. Sylvain CLERC	Ingénieur	STMicroelectronics	Invité
M. Antoine FRAPPÉ	Professeur associé	Université de Lille/Junia	Directeur de thèse
Mme Andreia CATHELIN	Directrice de recherche	STMicroelectronics	Co-Directrice de thèse
M. Benoit LARRAS	Professeur associé	Université de Lille/Junia	Co-Encadrant de thèse

"Life is and will ever remain an equation incapable of solution, but it contains certain known factors"

Nikola Tesla

"The voice is the signature of the soul"

Alfred Tomatis

Acknowledgements

First of all, I want to thank my supervisors Dr. Andreia Cathelin, Dr. Antoine Frappé and Dr. Benoit Larras for giving me the opportunity to work on this exciting subject and for their continuous support, high quality advising and commitment.

I want, also, to specially thank M. Sylvain CLERC for his help, support and technical assistance.

Besides, I want to thank all the PhD friends that I met during my journey at STMicroelectronics, Crolles. Especially: A. Oudrhiri, M. Bouchoucha, R. Dumont, G. Tochou, D. Flores, S. Sadlo, P. Cathelin, F. Paillardet, J. Poupon, V. Colrat, A. Asprilla, S. Abdelbar, E. Taly, D. Pino, V. Cincon, M. Tmimi, A. Leravallec, D. Gaidioz, C. Arricastres, C. Bonnafoux, JB Casanova.

For my new team at STMicroelectronics, I want to thank you for giving me the opportunity to join such joyful environment and work on cutting-edge products.

I want to thank Florence Alberti for her excellent assistance and support delivered during all the PhD years.

I want also to thank all the colleagues from Junia, J. Duforest, M.Chêne, B.Stefanelli, A. Gautier.

To my friends, thank you for being here when I was in need and thank you for guiding me when I felt lost.

Finally, here comes the persons to whom I dedicate this thesis: My family. Without you, none of this would never happened. You assisted me since the first day of my life, I never felt alone even in the loneliest days of my life. I always felt protected and assisted wherever I went. Thank you for being in my side and thank you for assisting me in my choices and thank you for being comprehensive, helpful and encouraging. Even by using all the thanking words in all the languages, I can never thank you enough.

Abstract

Nowadays, the cloud connects a network of several sensors collecting multiple key environmental information such as sound, temperature and humidity... These collected data is further processed through multiple sensors' interfaces and communicated to the human for easy understanding of the surrounding environment. These interfaces are continuously analyzing the sensors' measured data without considering the relevance of collected information. This blind processing generates massive amount of unimportant data and requires the interfaces to be always on. This will principally impact the power efficiency and degrade the battery lifetime.

The sound represents one the key environmental signals allowing the communication between the human and machine. Near-microphone processing units analyze the detected sound to translate its transmitted information. This information can take multiple forms such as speaker identification, keyword spotting or only voice activity detection. The near-microphone processing complexity will scale with the complexity of the extracted information. In our case, we are interested by keyword spotting. This task allows the detection of single spoken keywords in different background environments.

In this context, this dissertation presents a keyword spotting unit acting as a wake-up unit analyzing the relevance of input speech signals. The work, presented in this thesis, describes the development of an ultra-low power event-driven keyword spotting unit. This unit is composed of an event-driven continuous-time feature extraction unit to generate spectral-based spectrograms and a convolutional neural network to generate classification results. The architecture of the keyword spotting is modeled using *Matlab*® to validate its operation. Then, The feature extraction unit is implemented in STMicroelectronics 28nm FD-SOI CMOS technology to prove its hardware implementation, extract the performances and compare it to the different state of the art keyword spotting systems.

Résumé

De nos jours, le cloud connecte un réseau de plusieurs capteurs collectant plusieurs informations environnementales clés telles que le son, la température et l'humidité... Ces données collectées sont ensuite traitées via les interfaces de plusieurs capteurs et communiquées à l'humain pour faciliter la compréhension de l'environnement. Ces interfaces analysent en permanence les données mesurées par les capteurs sans considérer la pertinence des informations collectées. Ce traitement aveugle génère une quantité massive de données sans importance et nécessite que les interfaces soient toujours actives. Cela aura principalement un impact sur l'efficacité énergétique et dégradera la durée de vie de la batterie.

Le son représente l'un des signaux environnementaux clés permettant la communication entre l'humain et la machine. Les unités de traitement proches du microphone analysent le son détecté pour traduire les informations transmises. Ces informations peuvent prendre plusieurs formes telles que l'identification du locuteur, la détection de mots clés ou uniquement la détection d'activité vocale. La complexité du traitement proche du microphone évoluera avec la complexité des informations extraites. Dans notre cas, nous nous intéressons au repérage de mots clés. Cette tâche permet la détection de mots-clés à voix unique dans différents environnements d'arrière-plan.

Dans ce contexte, cette thèse présente une unité de détection de mots clés agissant comme une unité de réveil analysant la pertinence des signaux vocaux d'entrée. Le travail présenté dans cette thèse décrit le développement d'une unité de détection de mots-clés événementielle à très faible consommation. Cette unité est composée d'une unité d'extraction de caractéristiques en temps continu pilotée par événements pour générer des spectrogrammes basés sur le spectre et d'un réseau neuronal convolutif pour générer des résultats de classification. L'architecture du repérage de mots-clés est modélisée sous MATLAB pour valider son fonctionnement. Ensuite, l'unité d'extraction de fonctionnalités est implémentée dans la technologie CMOS FD-SOI 28 nm de STMicroelectronics pour prouver sa mise en œuvre matérielle, extraire les performances et la comparer aux différents systèmes de détection de mots clés de pointe.

Contents

List of Figures **xv**

List of Tables **xxi**

1 Introduction **1**
1.1 Context 1
1.2 Thesis objective 3
1.3 Main contributions 3
1.4 List of publications 4
1.5 Manuscript structure 5

Bibliography **7**

2 State of the art **9**
2.1 Introduction 9
2.2 Voice activity detection and keyword spotting systems 10
2.2.1 Voice activity detection systems 10
2.2.2 Keyword spotting systems 13
2.2.3 Synchronous systems 13
2.2.4 Event-driven systems 16
2.3 Target system 18
2.4 Target performance & Comparison with state of the art 18
2.5 Conclusion 21

Bibliography **22**

3 System description and simulation methods **25**
3.1 Introduction 25
3.2 Target specifications 27

3.3 Abstract description of the proposed system 28
3.3.1 Spectral-based feature extractor 28
3.3.2 Continuous-time digital signal processing 29
3.4 Event-driven continuous-time spectral-based feature extractor: 34
3.4.1 Simulation environment: Oversampling to simulate continuous-time systems. 34
3.4.2 Level-crossing ADC parameters. 36
3.4.3 Model and parameters of FIR filters. 38
3.4.4 Energy extraction process 44
3.5 Classification accuracy 48
3.5.1 Convolutional neural network 50
3.5.2 Recognition accuracy 50
3.5.3 Impact of increasing the granular delay on the recognition accuracy 52
3.6 Conclusion 55

Bibliography **56**

4 Concept validation using FPGA **59**
4.1 Introduction 59
4.2 Differences between *Matlab*® and RTL model 60
4.3 Clock-gated delays based keyword spotting architecture 61
4.3.1 FIR filter 61
4.3.2 Multi-frame energy accumulator 64
4.3.3 Storage and Truncation 66
4.4 FPGA hardware implementation 71
4.4.1 Synthesis constraints 72
4.4.2 Hardware resources 73
4.5 Test and measurement results 74
4.5.1 Test setup 74
4.5.2 Test results 75
4.6 Synthesis in the 65nm CMOS process 78
4.6.1 System performance 79
4.7 Conclusion 82

Bibliography **84**

5 28nm FD-SOI CMOS event-driven mixed-signal spectral-based feature extractor design and physical implementation 85
5.1 Introduction . . . 85
5.2 28nm FD-SOI technology choice . . . 86
5.3 Proposed keyword spotting system . . . 89
5.4 Delay line . . . 91
5.4.1 Delay creation principle . . . 91
5.4.2 Delaying techniques . . . 91
5.4.3 Proposed delay element . . . 93
5.4.4 Granular delay schematic . . . 94
5.4.5 Granular delay characterization . . . 96
5.4.6 Cascoded biasing block . . . 100
5.4.7 Delay tap architecture . . . 101
5.4.8 Delay line architecture . . . 103
5.5 Physical implementation details . . . 104
5.5.1 Full custom delays design . . . 104
5.5.2 Semi-custom digital flow . . . 107
5.6 Post-layout simulation results . . . 116
5.6.1 Analog delays simulation . . . 116
5.6.2 Back-annotate simulations . . . 117
5.7 Chip finishing . . . 119
5.8 Conclusion . . . 120

Bibliography 122

6 28nm FD-SOI System-on-Chip measurements 125
6.1 Introduction . . . 125
6.2 Fabricated System-on-Chip "Capybara" . . . 125
6.3 Packaging . . . 126
6.4 Test plan . . . 127
6.4.1 Socket choice . . . 127
6.4.2 Test setup . . . 127
6.4.3 Materials used for test . . . 128
6.5 Tests and measurement results . . . 129
6.5.1 Chip-alive test . . . 129
6.5.2 SPI configuration test . . . 129
6.5.3 Synchronization signals test . . . 131

6.5.4 Delay line test . . . 132
6.5.5 Test on audio files . . . 135
6.5.6 Recognition accuracy . . . 138
6.6 Power analysis . . . 139
6.6.1 Delay line . . . 140
6.6.2 Digital processing bands . . . 140
6.6.3 Power repartition . . . 142
6.7 Comparison with state of the art . . . 143
6.8 Conclusion . . . 144

Bibliography **147**

7 Conclusion & Perspectives **149**
7.1 Conclusion . . . 149
7.2 Future research perspectives . . . 150

APPENDIX A Chip assembly : Pad-ring **152**

APPENDIX B Measurement configurations. **157**
B.1 Chip alive test . . . 157
B.2 Pins configuration for the other tests . . . 158

List of Figures

1.1 Annual size of the global datasphere [1], [2]. 1
1.2 Estimation of the data center energy consumption considering the Moore's law limitation [3]. 2
1.3 Pre-processing wake-up unit to perform selective processing. 3

2.1 mixed-signal spectral-based VAD system architecture [1]. 10
2.2 Acoustic sensing system architecture (top), and operation principle of time-interleaved mixer-based frequency scanning (bottom) [2]. 11
2.3 Architecture of the overall VAD system [3] 12
2.4 Serialized DFT-based feature extractor [3] 12
2.5 Diagram of the power-proportional keyword spotting system [4]. 14
2.6 MFCC circuit composed of pre-emphasis, framing, windowing, FFT, MEL, log2 operation, and dct [5]. 15
2.7 DSCNN topolgy [5]. 15
2.8 System architecture of the keyword spotting and speaker identification circuit [7]. 16
2.9 Architecture and signal flow of Sound Detection and Feature Extraction [7]. 16
2.10 Event-driven general purpose wake-up chip [8]. 17
2.11 Abstract diagram of the target keyword spotting system. 18

3.1 Generic architecture of keyword spotting systems 25
3.2 Classical pipeline for extracting log-Mel spectral and Mel-frequency cepstral speech features using the fast Fourier transform (FFT) [2]. 26
3.3 Abstract model of the proposed system. 29
3.4 Different signal processing domains [11]. 29
3.5 Continuous-time digital signal processing chain. 30
3.6 Level-crossing ADC and its operation principle. 31
3.7 Impact of CT-DSP on the aliasing problem [14]. 32

3.8 Architecture of the FIR filter. 33
3.9 Continuous-time event-driven feature extraction unit. 34
3.10 Impact of the sampling and oversampling on the quantization error compared to continuous-time amplitude sampling [16]. 35
3.11 Distribution of the resolution at which level-crossing ADC generates more events than a 16kHz conventional DT ADC, for 500 audio samples. 37
3.12 Speech spectrum and characteristics of the human cochlea. 38
3.13 Magnitude response of the proposed bank of 12 digital CT FIR filters, with a stop-band attenuation of 20dB. 41
3.14 Magnitude response of the proposed bank of 8 digital CT FIR filters. The stop-band attenuation of -5dB soothes the design constraints to minimize the number of taps in each filter. 43
3.15 Energy extraction flow for different signal processing systems 46
3.16 Spectrograms generated using DT and CT methods and their difference. . . 47
3.17 Architecture of the approximate squarer and the iterative version [23]. . . . 48
3.18 Histograms presenting the resolution needed to decode the maximum energy value for different audio samples. 49
3.19 Impact of the energy resolution on the recognition accuracy in the 10 keywords case. 49
3.20 Architecture of the used convolutional neural network. 50
3.21 Confusion matrices for 10 keywords recognition including the background using the two filtering models. 51
3.22 The impact of the ADC resolution on the recognition accuracy for different numbers of keywords. 52
3.23 Delay line architecture and the operation of granular delays. 53
3.24 Minimum time between two successive pulses at the *REQ* signal, generated for multiple audio samples for four different keywords. 54
3.25 Delay line architecture and the operation of granular delays. 54

4.1 RTL vs *Matlab*® model. 60
4.2 Architecture of the keyword spotting system. 61
4.3 FIR filter architecture. 62
4.4 MAC architectures. (a) MAC architecture with matched delay for asynchronous continuous-time processing. (b) Proposed MAC architecture with one cycle delay processing. 62
4.5 Operation flow for the two MAC architectures. 63
4.6 Delay line architecture and operation. 64

4.7 Framing signals. 65
4.8 Architecture of the energy accumulator. 65
4.9 Energy multiplexing flow. 66
4.10 Architecture of the self truncation memory. 67
4.11 pseudo clock used to control the writing process. 68
4.12 Energy writing process in the register-matrix memory. 68
4.13 Read flag generation process. 69
4.14 Read trigger synchronization on the read flag. 70
4.15 Reading synchronization process of the memories in the 8 processing bands. 71
4.16 RTL Architecture block diagram. 72
4.17 *Intel*® DE2-115 Cyclone IV FPGA validation board from terasic. 73
4.18 GPIOs mapping into the Cyclone IV FPGA platform. 75
4.19 Test setup. 77
4.20 Normalized spectrograms generated using *Matlab*® and FPGA implementations for "One" and "Two" keywords, respectively ((a), (b) ; (d), (e)). Spectrograms presenting the calculated difference ((c), (f)). 77
4.21 Average relative error in energy values for 10 keywords from each number. 78
4.22 Power consumption repartition in the three different PVT scenarios. 80
4.23 Area breakdown of the proposed feature extraction unit. 81

5.1 FD-SOI family devices. (a) LVT using Forward Body Bias (FBB) to decrease V_{th}, and b) RVT using Reverse Body Bias (RBB) to increase V_{th}. 86
5.2 Threshold voltage (V_{th}) variation for RVT and LVT transistors, N- and P-MOS in 28 nm FD-SOI technology, for the respective applied body-bias voltage (VB) [1]. 87
5.3 Multiple channel lengths (poly biasing) support in 28nm FD-SOI standard cells [2]. 88
5.4 Architecture of the feature extraction unit. 90
5.5 (a) Thyristor-delay concept [8]. (b) Example of a thyristor delay element [8]. 91
5.6 Architecture overview of the delay element. 94
5.7 Operation of the delay element. 95
5.8 Granular delay schematic. 96
5.9 (a) Muller gate symbol. (b) Muller gate architecture. (c) Muller gate truth table.[9] 96
5.10 (a) Evolution of the delay value vs the biasing control current. (b) Variation of the delay value as a function of the frequency of the pulses in the REQ signal for a delay element configured at 1.26μs and biased at 1V. 98

5.11 Montecarlo simulation (1000 samples): delay variation due to process and mismatch variations 98
5.12 (a) Static power repartition. (b) Evolution of the total power as a function of the period of pulses at the REQ signals, for different supply voltages. 100
5.13 Cascode current mirror architecture and sizing of its different devices. . . . 101
5.14 (a) Control voltage evolution in time for different control current values. (b) Current mismatch of the biasing stage. (c) I-V characteristic of the biasing stage. 102
5.15 Serpentine architecture of the delay tap. 103
5.16 Complete architecture of the 16 delay line with biasing current mirror and output buffers. 103
5.17 Layout structure of the granular delay cell. 105
5.18 Layout of the delay tap element. 106
5.19 Delays line final layout structure. 107
5.20 Digital design flow. 108
5.21 Combined REQ signal constraint condition. 110
5.22 Post synthesis leakage power repartition. 113
5.23 Digital Place & Route flow. 114
5.24 Floorplan. 115
5.25 Normalized spectrograms generated using *Matlab*® and the system physical implementations for "Zero" and "Nine" keywords, respectively ((a), (b) ; (d), (e)). Spectrograms presenting the calculated difference ((c), (f)). 118
5.26 Power consumption of the digital processing bands in the three set scenarios. 118
5.27 Power reparation after back-annotate simulations. 119
5.28 Decoupling capacitance value for the different supply voltages. 120
5.29 Chip final layout. 121

6.1 Chip micrograph. 126
6.3 Setup used to test the feature extractor SoC. 128
6.4 Used measurement devices. 129
6.5 Setup to test the coefficient bank programming. 130
6.6 Coefficient bank programmed alternatively by "000000" and "111111" values. 130
6.7 framing signals. 131
6.8 Memory reading process in all the digital processing bands. 132
6.9 Combined REQ signal from the 1st processing band. 133
6.10 Combined REQ signal from the 8th processing band. 133

6.11 Delay tap value variation as a function of the biasing current in measurement and post-layout simulation. 134
6.12 Measured delay variability over 10 SoC samples. 135
6.13 Measured power consumption of the delay line as a function of the frequency of pulses at the REQ signal input. 136
6.14 Examples of some audio files. 136
6.15 Measured confusion matrix for 11 audio classes (10 keywords + background). 138
6.16 Measured recognition accuracy Vs Number of keywords. 139
6.17 Measured power consumption variation for the delay line as a function of the supply voltage. Total power concerns keywords analysis. Static power is leakage. 140
6.18 Measured power consumption variation for the digital processing bands as a function of the supply voltage. This variation is measured when the body bias voltages are equal to 1.8V and -1.8V, for V_BB_P_CORE and V_BB_N_CORE, respectively. 141
6.19 Power consumption variation for the digital processing bands on a logarithmic scale. 141
6.20 Measured power consumption variation as a function of the body bias voltage, for a supply voltage set to 0.61V. 142
6.21 (a) Measured power breakdown. (b) Measured power for the three functional modes. 143

A.1 Pad ring. 156

B.1 I-V curve for the delays. 158
B.2 I-V curve for the digital processing bands. 158

List of Tables

3.1 Characteristic key frequency values for the 12-bandpass FIR filter bank. . . 41
3.2 Characteristic key frequency values for the 8-bandpass FIR filter bank. . . . 44

4.1 Clock signals constraints. 73
4.2 Total number and repartition of hardware logic elements. 74
4.3 Type and function of each IO in the feature extraction hardware design. . . 76
4.4 Power consumption values in the three PVT scenarios for a speech rate of 60 words/minute. 79
4.5 Comparison with state-of-the-art keyword spotting systems. 82

5.1 Leakage reduction achieved by poly-biasing in 28nm FD-SOI technology [2]. 88
5.2 Qualitative comparison of the different delay creation techniques [3]. 93
5.3 Sizing of all the transistors in the discharge control part. 97
5.4 leakage evolution with the supply voltage. 99
5.5 Energy per event evolution with the supply voltage. 100
5.6 Constraints of the different synchronizing signals. 111
5.7 Post synthesis leakage power estimation for different scenarios. 112
5.8 Variability of the delay taps values. 117
5.9 Power repartition inside of the delay line 117

6.1 Normalized spectrograms generated using *Matlab*® and measured SoC for "Zero", "Five" and "Seven" keywords and the 10X magnified difference. . . 137
6.2 Comparison with state-of-the-art feature extraction systems targeting keyword spotting. 145

A.1 Different Inputs/Outputs and their function 152

B.1 Configuration of supply pins for the chip-alive test 157
B.2 Configuration of pins for the SPI configuration test 159
B.3 Configuration of pins to test framing signals. 159

B.4 Configuration of pins to test the delay line. 159

Chapter 1

Introduction

1.1 Context

Nowadays, people and machines are connected through multiple platforms either to communicate, analyze, or transfer several forms of data. This enhanced use of connected networks generates massive amounts of data, that will be stored or processed by the dedicated cloud or data center infrastructures. According to the studies presented in [1] and [2], the yearly data deluge has increased exponentially. Fig. 1.1 shows the evolution of the generated data amount from 2010 up to 2025. Consequently, the global datasphere is expected to achieve the limit of 175 Zetabytes (1 Zetabyte = 10^{21} byte) by 2025.

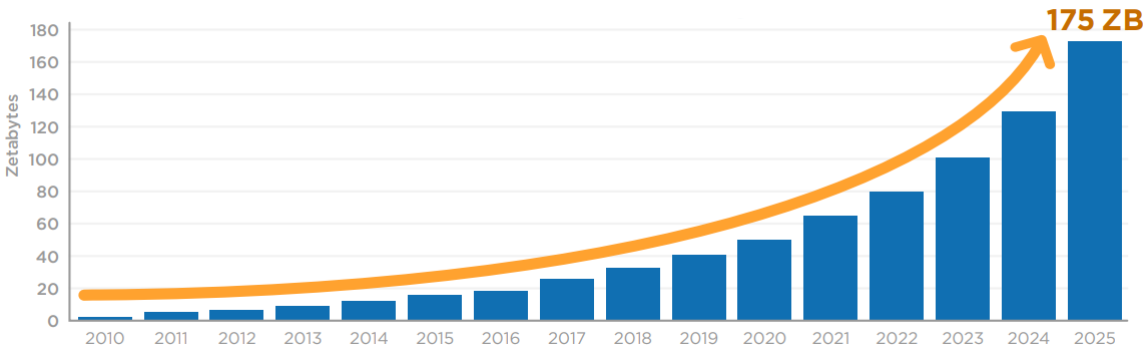


Figure 1.1 Annual size of the global datasphere [1], [2].

Dealing with this data deluge requires increasing the number of data centers and their storage and processing capacities. Hence, the electrical power dedicated to these data centers will increase. The study presented in [3] describes the yearly evolution of the power consumption of data centers based on the impact of Moore's law limitation. This study is

shown in Fig. 1.2. Based on this study, by 2030 the power consumption of the data centers is expected to reach approximately 1000 TeraWatts.

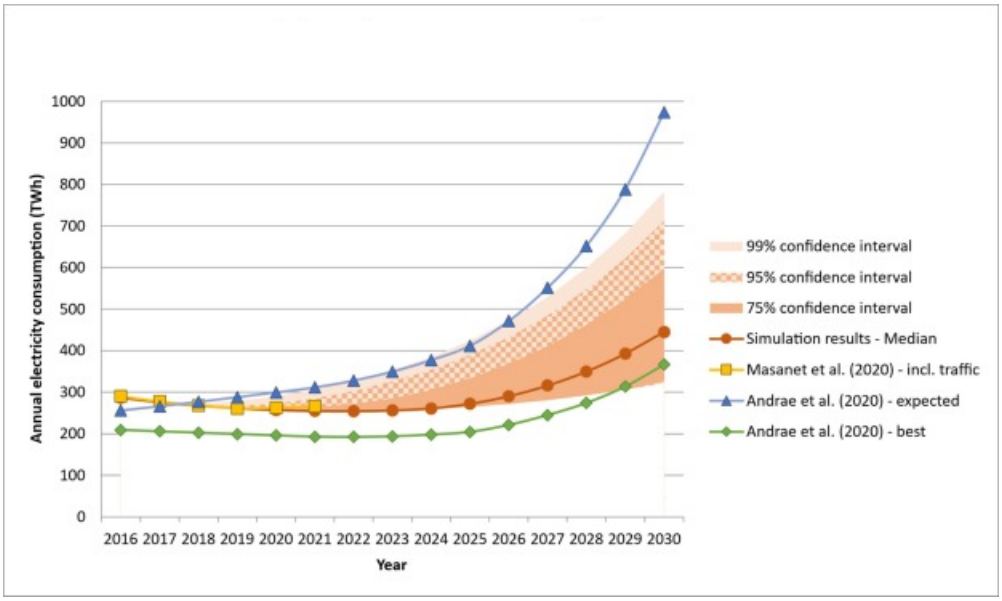


Figure 1.2 Estimation of the data center energy consumption considering the Moore's law limitation [3].

This increased power consumption requires more power production from the power delivery stations. In the case of non-green power production, this means an increased pollution level. Hence, this will impact the target sustainability level.

Due to Moore's law limitation, we can no longer reduce the power consumption by reducing the technology nodes. Hence, we need to change the adopted processing methods to analyze and generate only relevant information-holding data and ignore all the rest. By doing so, the number of required data centers will be drastically reduced.

In the context of the sensors, the cloud connects multiple Internet-of-Things (IoT) devices that interact with the outer environment through the sensor's interface to allow communication with the human. Moreover, dealing with real-world signals requires power-hungry always-on systems that continuously process the incoming signal. However, this continuous processing is maintained even when no significant information is present in the input signal. Thus, we fall into the blind processing case, where we produce a useless data deluge resulting in lower power efficiency and battery lifetime. To address this limit, the processing units in sensor nodes are bound to become intelligent to be able to adopt selective processing of only information-holding signals.

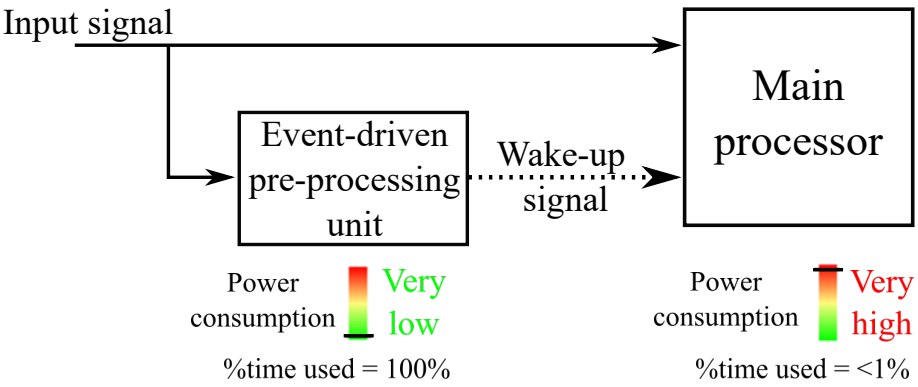


Figure 1.3 Pre-processing wake-up unit to perform selective processing.

As shown in Fig. 1.3, selective processing can be done by integrating an event-driven smart pre-processing unit that triggers the high-performance units only with the arrival of relevant information at the input. So, the system will be composed of two main blocks, the low-power event-driven pre-processing unit analyzing only relevant input signals to generate or not the wake-up signal triggering the high-power, high-performance units to further process the input signal. Consequently, the power consumption will be dominated by the event-driven wake-up unit since it is the one continuously working even if it has low power compared to the next blocks that are used only at a small fraction of time.

In our context, the interest falls into the development of embedded near-sensor processing units for applications such as sound detection and keyword spotting.

1.2 Thesis objective

This thesis has two main objectives. The first one is to explore and design a new architecture of a keyword spotting system, acting as a wake-up unit. The second objective is the low-power, low-complexity hardware implementation of the explored KWS system, using the 28nm FD-SOI technology. This technology is characterized by multiple features, such as body bias, to achieve the best power efficiency compared to state-of-the-art.

1.3 Main contributions

- ***Target specifications:*** Study of state-of-the-art keyword spotting systems to explore their architecture and collect information about the best achieved performance. The objective is to be better than state-of-the-art. Hence, we set principally our target

specifications to recognize 10 keywords with an accuracy higher than 90% and a power consumption of less than 500nW. These performances will gate the further steps concerning the design of the target KWS system.

- ***Keyword spotting architecture exploration, modeling, and validation:*** New developed event-driven keyword spotting system architecture composed of a spectral-based continuous-time feature extractor and convolutional neural network. The feature extractor is composed principally of 8 processing channels to construct the spectrograms, displaying the spectral energy distribution. Each channel is composed of a continuous-time Finite Impulse Response (FIR) filter and an energy extractor. This system was modeled using *Matlab*® to validate the choice of the building blocks and extract the different key parameters, such as bandwidth, filter's order, out-of-band attenuation... These modeling steps set the necessary basis to perform the hardware implementation. The feature extractor was translated from *Matlab*® syntax to a hardware-oriented RTL code. Then, the hardware feasibility of the proposed keyword spotting system was validated through an FPGA implementation of the RTL-designed feature extractor.

- ***28nm FD-SOI physical implementation with digital design flows customization:*** The physical implementation of the proposed feature extractor using CAD tools in the STMicroelectronics 28nm FD-SOI CMOS technology. This was done in two different environments. A full custom analog environment to implement the delay cells used in the feature extractor architecture. The second environment is the digital one. This latter environment is used to implement all the necessary logic operations (Filtering, energy extraction, storage...). The blocking point in this step is that the digital design tools are only used for synchronous systems. Since the proposed feature extraction unit is clockless, the synthesis and place & route flows were customized to be adapted to asynchronous logic and automatize the gate netlist and layout generation. Finally, the physically-designed feature extractor was electrically tested to validate its operation.

1.4 List of publications

The achieved results during this thesis have been evaluated through multiple publications and a patent :

- **S. Mourrane, B. Larras, A. Cathelin and A. Frappé, "Event-Driven Continuous-Time Feature Extraction for Ultra Low-Power Audio Keyword Spotting," 2021 IEEE 3rd International Conference on Artificial Intelligence Circuits and Systems**

(AICAS), 2021: This publication presents the results of the *Matlab*® modeled keyword spotting system.

- **S. Mourrane, B. Larras, S. Clerc, A. Cathelin and A. Frappé, "Low-Power Event-Driven Spectrogram Extractor for Multiple Keyword Spotting: A proof of concept," 2023 21st IEEE Interregional NEWCAS Conference (NEWCAS), Edinburgh, United Kingdom, 2023:** This publication details the FPGA Implementation and presents the simulation results.

- **S. Mourrane, B. Larras, S. Clerc, A. Cathelin and A. Frappé, "A 291nW Real-Time Event-Driven Spectrogram Extraction unit in 28nm FD-SOI CMOS for Keyword Spotting Application," ESSCIRC 2023- IEEE 49th European Solid State Circuits Conference (ESSCIRC), Lisbon, Portugal, 2023:** This publication describes the 28nm FD-SOI fabricated feature extraction chip and the measurement results.

- **S. Mourrane, B. Larras, A. Cathelin and A. Frappé, "Event-driven energy extractor" patent filed on 21-06-2023:** This patent details the energy extraction process adopted in the feature extraction unit.

Another publication is prepared for a more detailed version of the description of the feature extraction chip. This publication will target the Journal of Solid State Circuits (JSSC).

1.5 Manuscript structure

This dissertation is structured in five principal chapters as follows:

Chapter 2 is studying the different state of the art voice activity detection and keyword spotting systems. Based on this study, the specifications of the target keyword spotting system are set to be competitive or better than state of the art.

Chapter 3 presents the *Matlab*® modeling step followed to construct the proposed spectral-based keyword spotting system. The simulation results of this model as well as the different extracted design parameters are also presented.

Chapter 4 describes the RTL design, using VHDL language, of the proposed continuous-time clock-gated feature extraction unit. This chapter also presents the FPGA implementation of the designed system to validate its operation through the analysis of the recognition results. Moreover, the feature extraction unit is synthesized in 65nm CMOS technology, the same as the FPGA board, to validate the low-power nature of the proposed architecture.

Chapter 5 presents the hardware implementation of the asynchronous continuous-time feature unit using the 28nm FD-SOI technology. This chapter will detail the circuit design, simulation results, and the physical implementation of the different building blocks.

Chapter 6 describes the test setup and the electrical measurement results collected to validate the operation of the fabricated feature extraction unit. During this chapter, the performance of our system is extracted and compared to the latest state of the art keyword spotting systems.

This dissertation is concluded by the presentation of some future research perspectives to improve and continue the development of the proposed keyword spotting system.

Bibliography

[1] Reinsel, D., John Gantz, and J. Rydning. "IDC's "Data Age 2025" whitepaper." (2018).

[2] Rydning, D.R.J.G.J., Reinsel, J. and Gantz, J., 2018. The digitization of the world from edge to core. Framingham: International Data Corporation, 16, pp.1-28.

[3] Koot, Martijn & Wijnhoven, Fons. (2021). Usage impact on data center electricity needs: A system dynamic forecasting model. Applied Energy. 291. 116798. 10.1016/j.apenergy.2021.116798.

Chapter 2

State of the art

2.1 Introduction

Speech recognition is one of the most common integrated features in Internet-of-Things (IoT) devices. This task requires multiple complex hardware resources, like in data centers, to be able to perform the needed processing in a short time. However, this high-complexity processing will be accompanied by a high power consumption, which means low power efficiency and battery lifetime.

The speech processing can be done in multiple steps, with an increasing complexity when passing from one step to the next. These steps can be regrouped into three main functions. The first one is Voice Activity Detection (VAD), to detect only the presence of acoustic audio activity. The second one is Keyword Spotting (KWS), to detect single audio-spoken keywords. The last one is speech identification to recognize complete spoken speech including keywords and phrases. The processing complexity increases proportionally with the complexity of the task to perform.

To avoid power-hungry cloud processing, the research pushes towards the development of on-hardware embedded speech identification, especially the less complex tasks such as voice activity detection and keyword spotting. The advantage of embedding these systems on chip is to improve the power efficiency.

This chapter will present the relevant on-chip hardware design sound detection and keyword spotting systems. In more detail, the implementation of these systems will be described down to the system level to extract hardware opportunities leading to ultra-low-power audio feature extraction.

This chapter will discuss the state-of-the-art systems published until 2020, which corresponds to the beginning of this thesis work.

2.2 Voice activity detection and keyword spotting systems

As mentioned before, the complexity of the hardware scales proportionally with the task to perform. For this reason, we decided to target less complex activities to have a possible low-power on-chip hardware implementation. Hence, the focus, in this part, will be on the VAD and KWS systems.

2.2.1 Voice activity detection systems

Voice activity detection systems are developed to detect the presence of audio or acoustic activity in a background environment. These systems can be used as general wake-up units, for other processing elements, in multiple applications such as surveillance, maintenance, and home automation...

Several systems have been presented in the state of the art working on different on-chip architectures performing voice activity detection. Since we are targeting ultra-low-power performance, we will be concentrating on the works with the same power profile.

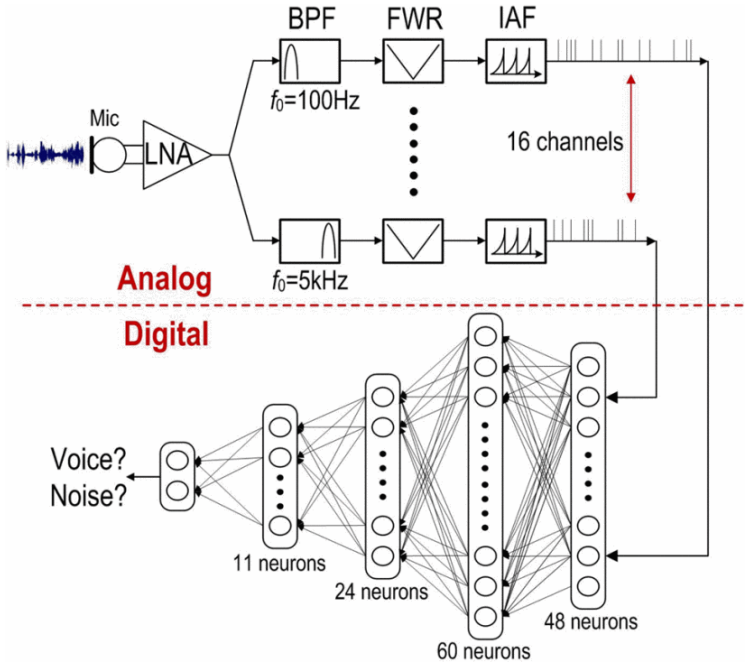


Figure 2.1 mixed-signal spectral-based VAD system architecture [1].

To start, Fig. 2.1 [1] presents an architecture of a system developed for the VAD objective. This mixed-signal architecture receives the audio signal from the microphone and then amplifies it using a low-noise amplifier (LNA). After that, the amplified signal is passed

through 16 parallel processing channels. Each channel contains a bandpass filter (BPF), a full-wave rectifier (FWR), and an integrate-and-fire (IAF) event encoder. The central frequencies of the BPFs are geometrically scaled from about 100Hz to 5kHz [1]. The IAF in each channel produces asynchronous events whose rate is roughly proportional to the signal energy of the respective band[1]. This system has demonstrated voice activity detection with an accuracy equal to 85% at maximum, while consuming 1μW. This system presents a competitive low-power VAD solution. However, this system presents multiple limitations starting with the non-programmability to be adapted for other applications and the non-scalability to technology nodes due to the use of analog filters. Moreover, this solution presents a high-complexity 16-band filtering feature extractor limited to a frequency range of 100Hz-5kHz used for a less complex VAD application.

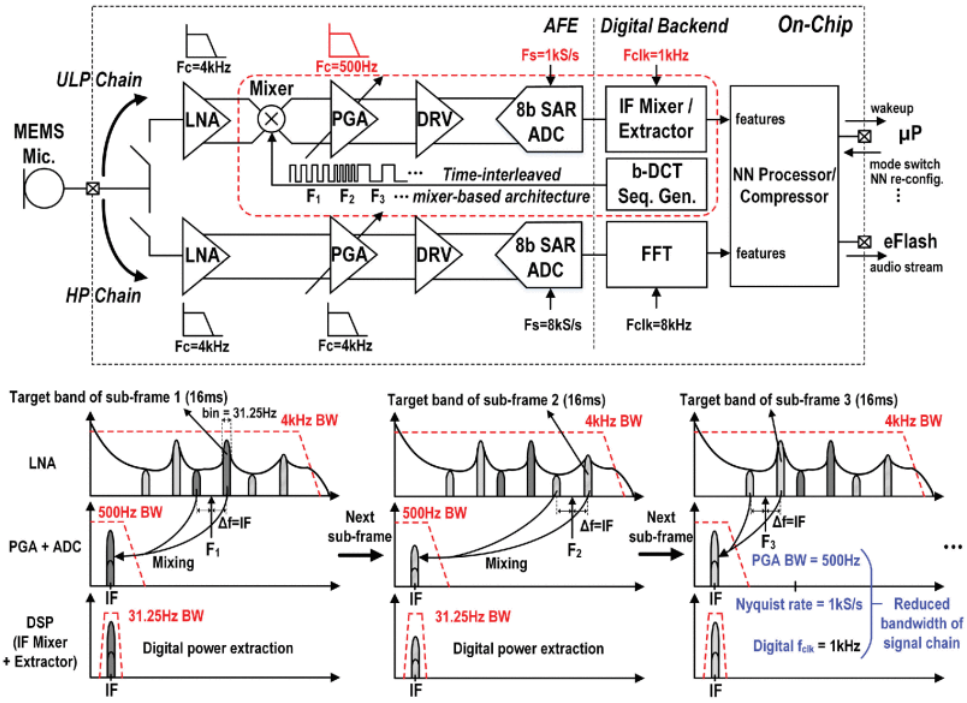


Figure 2.2 Acoustic sensing system architecture (top), and operation principle of time-interleaved mixer-based frequency scanning (bottom) [2].

Another low-power voice and acoustic activity detection system is displayed in Fig. 2.2(top), where the system is composed of two main working chains a high-performance (HP) chain and an ultra-low power (ULP) one. The operation principle of this system is shown in Fig. 2.2(bottom). The low-power chain is always on and continuously looking to analyze the presence of any sound activity. Then, the HP chain, which is power-gated by the previous chain, will be responsible for the further analysis of the signal to generate the needed features.

These features are then input into a neural network engine responsible for the generation of the wake-up signal. The ULP chain is a time-interleaved mixer-based frequency architecture. It performs the extraction of the input signal power after amplifying it and mixing it each time to a variable frequency lower than 500Hz [2]. This system has demonstrated sound and silent acoustic signal detection with a maximum accuracy of 91.5% while consuming 142nW for the ULP chain. This system presents a low-power programmable VAD solution to detect non-stationary audio events. However, this system adopts an always-on processing, which is not efficient during long silence periods. Moreover, it has a high latency value (512ms). This will impact the reaction time if this system is used as a wake-up unit.

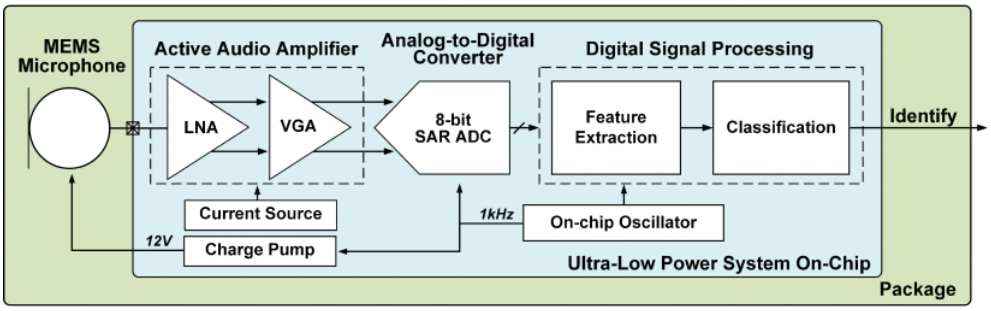


Figure 2.3 Architecture of the overall VAD system [3]

An additional VAD system is presented in [3]. The architecture of this system is depicted in Fig. 2.3. From an abstract perspective, it consists of an analog front-end, an 8-bit ADC, and a digital processing back-end divided into a feature extraction unit and a classifier.

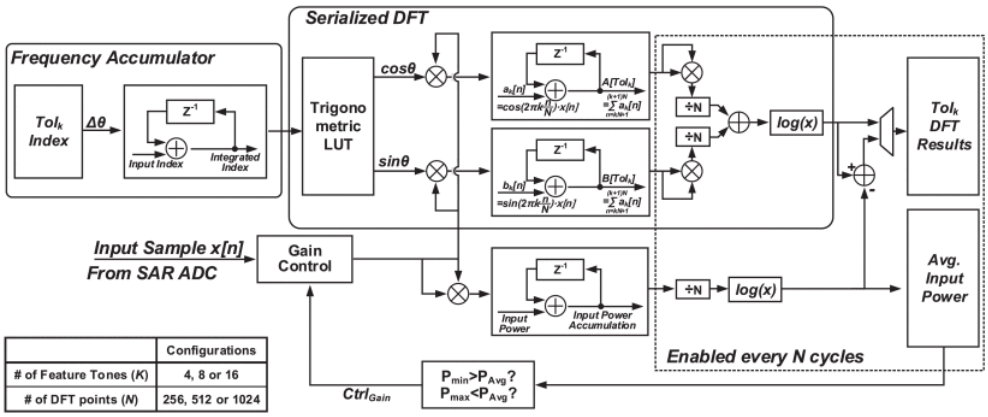


Figure 2.4 Serialized DFT-based feature extractor [3]

The feature extractor architecture is shown in Fig. 2.4. It is a spectral-based feature extractor performing a 512-point Discrete Fourier Transform (DFT) on discrete tones of

interest instead of the entire spectrum [3]. This DFT is performed using serialized computation, where the system monitors only one tone (frequency band) at a time and then switches to the next one [3]. The extracted DFT outputs are then injected into a support vector machine (SVM) based feature extractor to generate the identification results. This solution has demonstrated voice activity detection with an accuracy higher than 95% when detecting specific sounds such as the sound of a generator, car, or truck. This accuracy performance was achieved with a power budget of 12nW. This solution represents the lowest consuming state-of-the-art VAD system. However, it is limited to the detection of only stationary events and it lacks programmability for other use cases.

2.2.2 Keyword spotting systems

2.2.3 Synchronous systems

One of the first systems developed for speech identification is depicted in Fig. 2.5. It is composed of a feature extraction part performing the spectral division of the audio signal into 16 mel frequency bands covering a total bandwidth of 75Hz-5kHz. The spectral division is performed using Gm-C bandpass analog filters, then the average signal value is extracted in each band [4]. Furthermore, the features are injected into a decision tree to perform the classification. This system proposes a hierarchical power proportional operation, where the power consumption increases with the complexity of the processing needed to extract the key information from the input signal. This architecture has demonstrated speech recognition with an accuracy that can go up to 89% keeping the power consumption lower than 6uW in the worst case [4]. The use of Gm-C filters led to an architecture unable to scale with technologies and to ensure a configurable and programmable architecture to be adaptable for other audio applications

A more recent keyword spotting system is presented in [5]. It is composed of a spectral-based Mel Frequency Cepstrum Coefficient (MFCC) feature extractor and a depthwise-separable convolutional neural network (DSCNN). The feature extractor is depicted in Fig. 2.6, where the audio signal undergoes pre-emphasis, framing, windowing, Fast Fourier Transformation (FFT), MEL filter, log2 operation, and a Discrete Cosine Transform (DCT) [5]. In more detail, the test audio clips, taken from the Google Speech Command Dataset (GSCD) [6], are divided in two and sent to the feature extractor. Then, these divided samples undergo a 16-bit 256-point FFT at 8KHz sampling rate and frames of 32ms with a 16ms step [5]. The special point in this architecture is the use of a serial FFT instead of a parallel one to reduce the required memory by a factor of 8 size and power consumption by a factor of 11.

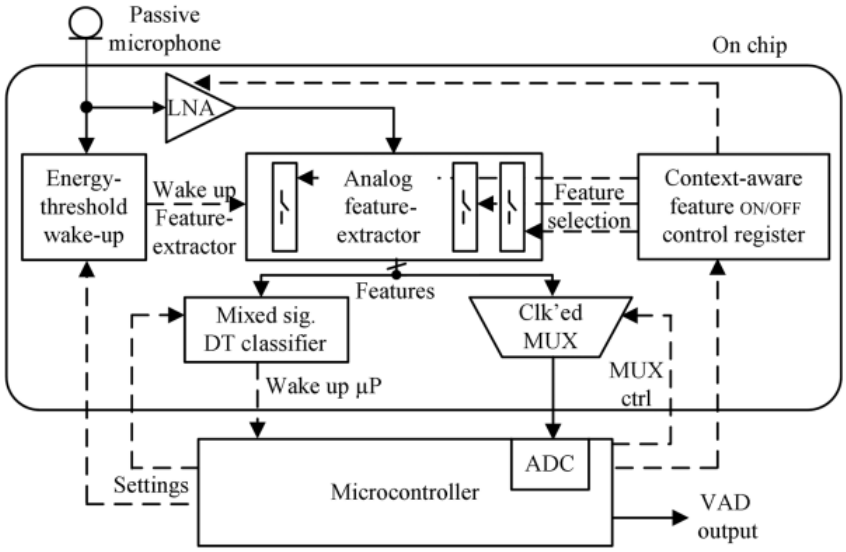


Figure 2.5 Diagram of the power-proportional keyword spotting system [4].

The 16-bit FFT output is then input into a simplified 2MAC Mel filters. Then, the filtered output is injected into a look-up table to perform the log2 operation.

The extracted 8-bit MFCC features are then input into a DSCNN to generate the classification results. The DSCNN, shown in Fig. 2.7, decomposes the standard 3D convolution to a 2D convolution, reducing data storage by 7X and computation by 7X as compared to a classic CNN [5]. This architecture has demonstrated the recognition of at most 2 keywords with a maximum accuracy of 94.8% while consuming 510nW. This system resolves the scalability problem thanks to the digital-friendly design environment. However, it presents a high-complexity solution limited to only the recognition of two keywords.

Another system is presented in [7], targeting not only keyword spotting but also speaker identification. This last task is more complex, which means that the system's complexity will scale with it. This system, shown in Fig. 2.8, is composed of an Analog Front End (AFE), a sound detector, a feature extractor, a Long Short Term Memory (LSTM) accelerator used as a classifier for KWS, and a Gaussian Mixture Model (GMM) accelerator used for speaker verification task. For KWS, the input audio signal is encoded using the AFE, then the sound detector, shown in Fig. 2.9, computes the average framed input amplitude and compares it

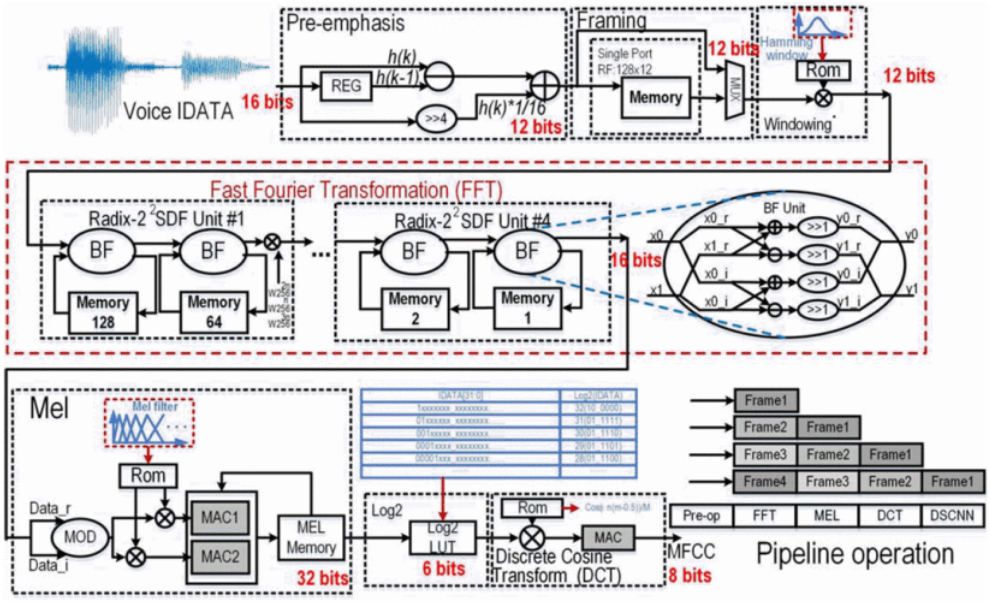


Figure 2.6 MFCC circuit composed of pre-emphasis, framing, windowing, FFT, MEL, log2 operation, and dct [5].

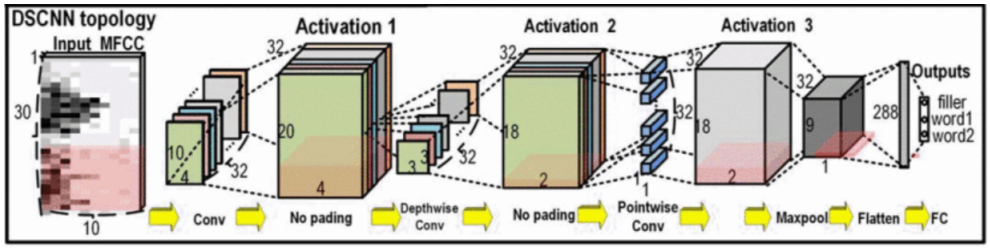


Figure 2.7 DSCNN topolgy [5].

continuously to a predefined threshold. Upon wake-up, the feature extractor processes the encoded audio signal to generate the corresponding MFCC features.

The used feature extractor is depicted in Fig. 2.9. It operates on frames of 32ms with 16ms step size and it computes 20 MFCC's, Δ's, $\Delta\Delta$'s at programmable center frequencies [7]. In each frame, a 1024pt real-DFT and 32pt DCT are operated. After the DFT, the mel-shaped bandpass filtering is performed [7]. The DFT and DCT depth, number of mel filters and features are all configurable to maintain a programmable architecture. This work presents a programmable and scalable architecture, which has demonstrated the recognition of 12 audio classes (10 keywords + silence + unknown) with an accuracy equal to 90.87% and a power budget of 16.11μW.

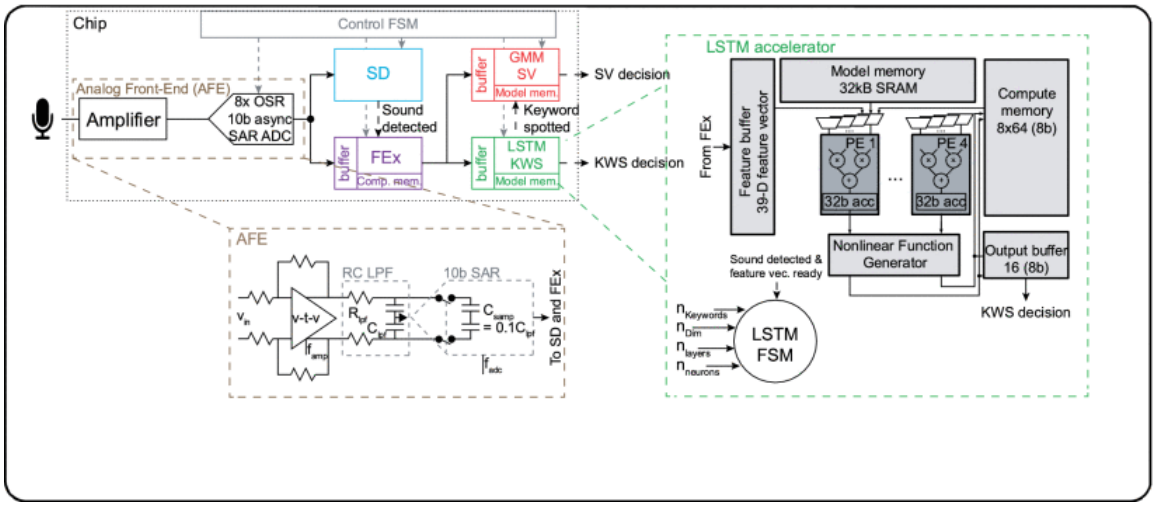


Figure 2.8 System architecture of the keyword spotting and speaker identification circuit [7].

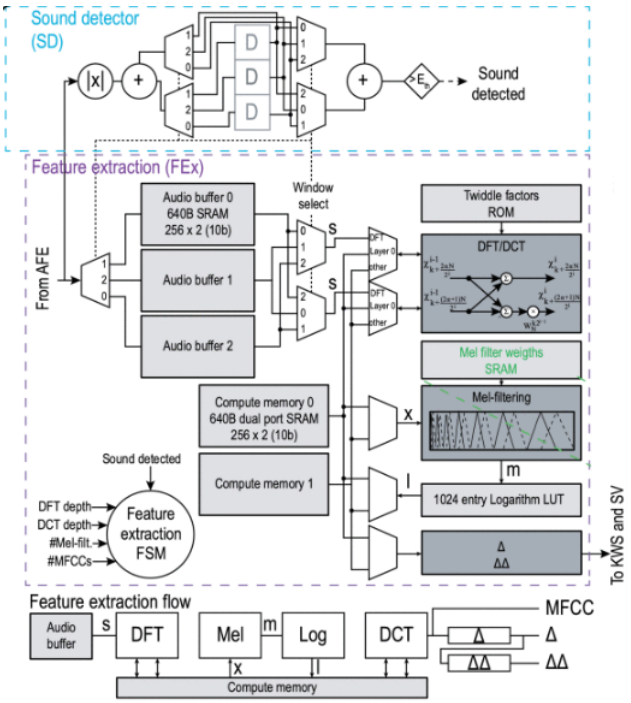


Figure 2.9 Architecture and signal flow of Sound Detection and Feature Extraction [7].

2.2.4 Event-driven systems

The systems presented in the previous section are completely synchronous, which means that there is a clock signal continuously running and synchronizing the different logic

operations. This approach has the advantage of perfect synchronization due to this clock signal. However, this clock stays running even if there is no significant information at the input to process. This means that these systems continue consuming power while treating non-relevant data. This induces a degradation of the power efficiency.

To resolve the previous issue, there is another processing approach. Event-driven processing consists of analyzing only input signals holding relevant information. This technique offers the opportunity to have a power consumption that scales with signal activity. The event-driven processing technique is well suited for audio signals because they contain sparsely distributed information separated by silence periods. Hence, the power efficiency will be improved thanks to consuming only leakage during silence periods.

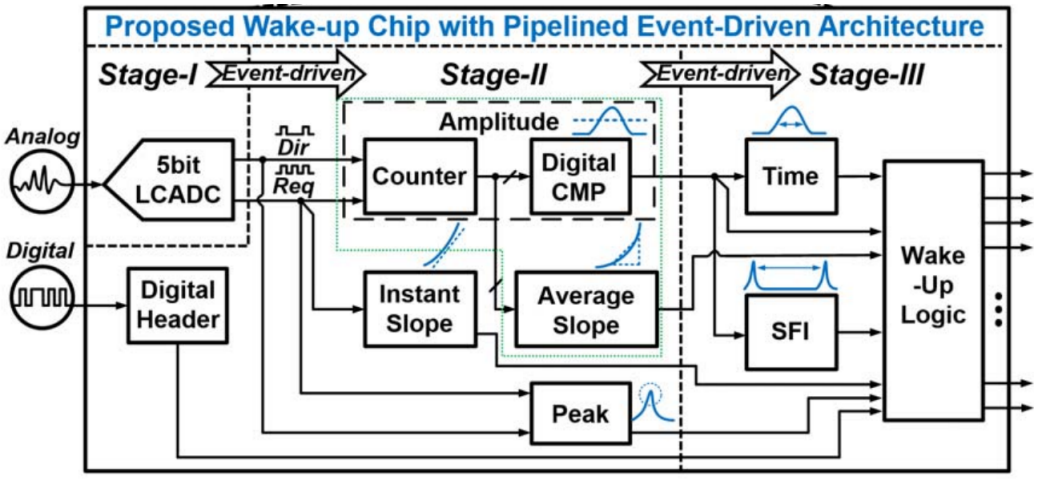


Figure 2.10 Event-driven general purpose wake-up chip [8].

An event-driven software-defined system is depicted in Fig. 2.10. It was developed for a general purpose and was tested on keyword envelope detection [8]. This system is divided into 3 event-driven processing stages, where each stage is triggered by the previous one. This architecture is a mixed-signal structure based on analog processing of the input signal through a level-crossing ADC that is then connected to a multi-function detector to extract different amplitude and time domain parameters (Instant slope, peak amplitude ...). Then, these extracted parameters are injected into a wake-up logic unit, which is responsible for generating or not the wake-up signal. This system has demonstrated its operation on the applications mentioned before by maintaining a low power profile, by consuming an event-rate dependent power of 75nW and an Idle power equal to 57nW. This system presents the lowest power consumption to other KWS systems, however, it is limited to the detection of the keyword envelope. Therefore, this system acts only as a wake-up for other keyword spotting systems.

2.3 Target system

The study of the different keyword spotting state-of-the-art systems led to the creation of an abstract idea about the target KWS system. Since we are dealing with audio signals with sparsely distributed information, the event-driven processing seems to be adapted to these signals and it offers a real opportunity to improve the power efficiency.

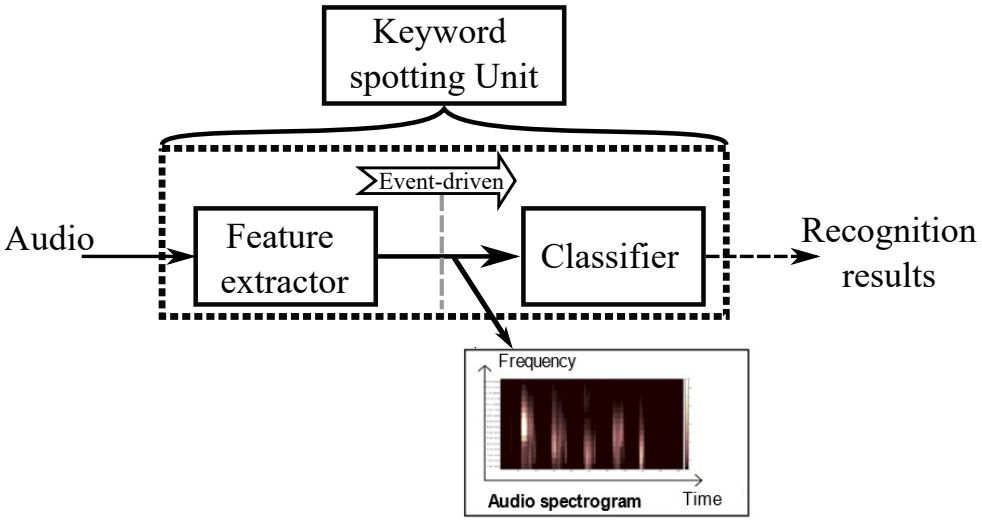


Figure 2.11 Abstract diagram of the target keyword spotting system.

An abstract architecture of the target keyword spotting system is shown in Fig 2.11. The audio signals are firstly analyzed using an event-driven feature extractor, performing the spectral division of the audio spectral content into multiple frequency bands to construct a spectrogram. These spectrograms display the energy distribution in time and frequency scales. They represent a unique signature differentiating one audio class from another. These spectrograms are then injected into a classifier, responsible for mapping them into some predefined classes to generate recognition results.

2.4 Target performance & Comparison with state of the art

The presented state-of-the-art systems demonstrated different performances either performing keyword spotting or only voice activity detection. The performance results of all the studied systems and the target system are summarized in Table 2.1. Each one of these systems is composed of a feature extractor unit and a neural network classifier to generate recognition results. Starting with the feature extractor, most of them are extracting spectral parameters such as MFCCs, power... The difference between these feature extractor units is

their implementation. They can either be implemented in an analog way using analog filters, which is the case for the systems in [1] and [4]. Other implementation methods are also shown in Table 2.1, where we have FFT and DFT-based feature extractor units. The implementation type will impact principally the programmability and scalability of the designed system. This is the reason pushing us towards targeting a spectral-based programmable and scalable feature extractor to be adapted for different types of applications.

These presented systems are mainly compared using two principal metrics: power consumption and recognition accuracy. Starting with the power, the first remark to deduce is that VAD systems are usually less consuming than KWS ones. This is expected because the power consumption scales with the complexity of the task to perform. In terms of application, we will target keyword spotting, as it is one of the trending applications in the state of the art. Hence, our system will be compared principally to the same type of system. In terms of recognition performance, we can see that the best results are achieved by the work in [7]. This system recognizes 12 audio classes (10 keywords + silence + unknown) with an accuracy of 90.87% and a power consumption of 18.3μW. Other systems show degraded performances like the one presented in [5], which targets only the recognition of at most 2 keywords with an accuracy of 94.6%. This was achieved while consuming 510nW. This system is less consuming than the one in [7], but the task complexity is way simpler in the system presented [5]. To be competitive with these results, we decided to target the recognition of 10 keywords with an accuracy higher than 90%, while having an activity-dependent power consumption of less than 500nW in the worst scenario.

To sum up, the challenge in KWS systems is to achieve the best trade-off between power consumption and recognition accuracy. Hence, we will be developing our systems to have the best performances compared to the state-of-the-art systems.

Table 2.1 Comparative table for the performances of the state-of-the-art KWS and VAD systems.

	Target performance	**ISSCC'20 [8]**	**VLSI'19 [7]**	**ISSCC'20 [5]**	**JSSC'16 [4]**	**JSSC'18 [3]**	**ISSCC'19 [2]**	**ISSCC'18 [1]**
Application	Keyword spotting	General purpose	Keyword spotting	Keyword spotting	Speech identification	Voice activity detection	Voice activity detection	Voice activity detection
Feature Extractor type	Spectral-based Feature extractor	Multi-function detector	MFCC	Serial-FFT MFCC	Analog Gm-C filters	Serial DFT	FFT-based power extractor	Analog bandpass filters
Technology	28nm FD-SOI	180nm	65nm	28nm	90nm	180nm	180nm	180nm
Supply voltage (V)	<0.7	0.6	0.6	0.41	—	0.6/1.2	0.6	0.6
Area (mm^2)	<1	2.54	2.56	0.23	2	3.44	17.55	2.52
Scalability/Programmability	Yes/Yes	Yes/Yes	No/Yes	Yes/Yes	No/No	No/No	No/Yes	No/No
Event-Driven	Yes	Yes	No	No	No	No	No	No
Idle Power (µW)	<0.5	0.075	—	—	—	—	—	—
Total Power (µW)	<0.5	0.057	18.3	0.51	6	0.0122	0.142	1
Dataset	GSCD	N.A	GSCD	GSCD	Babble	—	Babble	AURORA4
#Classes(keywords)	11(10)	N.A	12(10)	1-2	—	—	—	—
Accuracy (%)	>90	N.A	90.87	98-94.6	89	>95 (car detection)	91.5	85
Classifier	CNN	N.A	LSTM	DSCNN	Decision tree	SVM	Processor-based NN	Binarized neural network

2.5 Conclusion

Keyword spotting presents one of the most trending applications targeted by hardware integration. Multiple state-of-the-art systems were presented performing this task. The study of such systems led us to explore the opportunity offered by event-driven processing to improve power efficiency when dealing with audio signals containing sparsely distributed information. This deduction led us to develop a first idea about the abstract architecture of the target keyword spotting system. It will be composed of an event-driven feature extractor unit transforming audio signals into spectrograms, representing a signature to differentiate between audio classes. Then, these audio signals are fed into a classifier to generate recognition results.

The comparison with the different state-of-the-art systems helped us set the target performance to be competitive or better than the already existing systems. In more detail, the objective of the target system will be the recognition of 10 keywords with an accuracy higher than 90% and an activity-dependent power consumption lower than 500nW in the worst case.

Using the set specifications as a final target, in the next chapter we will start developing at first a software model to construct the different key building blocks of the target system, study its feasibility, and set the needed design parameters to start the hardware implementation.

Bibliography

[1] M. Yang, C. -H. Yeh, Y. Zhou, J. P. Cerqueira, A. A. Lazar and M. Seok, "A 1μW voice activity detector using analog feature extraction and digital deep neural network," 2018 IEEE International Solid - State Circuits Conference - (ISSCC), San Francisco, CA, USA, 2018, pp. 346-348, doi: 10.1109/ISSCC.2018.8310326.

[2] M. Cho et al., "17.2 A 142nW Voice and Acoustic Activity Detection Chip for mm-Scale Sensor Nodes Using Time-Interleaved Mixer-Based Frequency Scanning," 2019 IEEE International Solid- State Circuits Conference - (ISSCC), San Francisco, CA, USA, 2019, pp. 278-280, doi: 10.1109/ISSCC.2019.8662540.

[3] S. Jeong et al., "Always-On 12-nW Acoustic Sensing and Object Recognition Microsystem for Unattended Ground Sensor Nodes," in IEEE Journal of Solid-State Circuits, vol. 53, no. 1, pp. 261-274, Jan. 2018, doi: 10.1109/JSSC.2017.2728787.

[4] K. M. H. Badami, S. Lauwereins, W. Meert and M. Verhelst, "A 90 nm CMOS, 6μW Power-Proportional Acoustic Sensing Frontend for Voice Activity Detection," in IEEE Journal of Solid-State Circuits, vol. 51, no. 1, pp. 291-302, Jan. 2016, doi: 10.1109/JSSC.2015.2487276.

[5] W. Shan et al., "14.1 A 510nW 0.41V Low-Memory Low-Computation Keyword-Spotting Chip Using Serial FFT-Based MFCC and Binarized Depthwise Separable Convolutional Neural Network in 28nm CMOS," 2020 IEEE International Solid- State Circuits Conference - (ISSCC), San Francisco, CA, USA, 2020, pp. 230-232, doi: 10.1109/ISSCC19947.2020.9063000.

[6] P.Warden, "Speech commands: A public dataset for single-word speech recognition." Dataset available from http://download.tensorflow.org/data/speech_commands_v0.01.tar.gz,2017.

[7] J. S. P. Giraldo, S. Lauwereins, K. Badami, H. Van Hamme and M. Verhelst, "18μW SoC for near-microphone Keyword Spotting and Speaker Verification," 2019 Sym-

posium on VLSI Circuits, Kyoto, Japan, 2019, pp. C52-C53, doi: 10.23919/VLSIC.2019.8777994.

[8] Z. Wang et al., "20.2 A 57nW Software-Defined Always-On Wake-Up Chip for IoT Devices with Asynchronous Pipelined Event-Driven Architecture and Time-Shielding Level-Crossing ADC," 2020 IEEE International Solid- State Circuits Conference - (ISSCC), San Francisco, CA, USA, 2020, pp. 314-316, doi: 10.1109/ISSCC19947.2020.9062952.

Chapter 3

System description and simulation methods

3.1 Introduction

As seen in the state of the art, keyword spotting systems are mainly composed of three building blocks. The first block is the microphone responsible for the acquisition of the acoustic audio signal. Then, this signal is transmitted to the feature extraction unit responsible for extracting the key parameters characterizing the audio signals. After that, the extracted parameters are fed into a neural network performing the inference on the given features to compare them with the features used for training and finally generate the accuracy results.

Inside the feature extraction unit, the audio signal undergoes multiple operations to extract a unique signature allowing the categorization of different audio classes. Multiple features can be extracted to classify the audio signals. In Fig. 3.1, we have selected two of the principal features that are widely used in the state of the art.

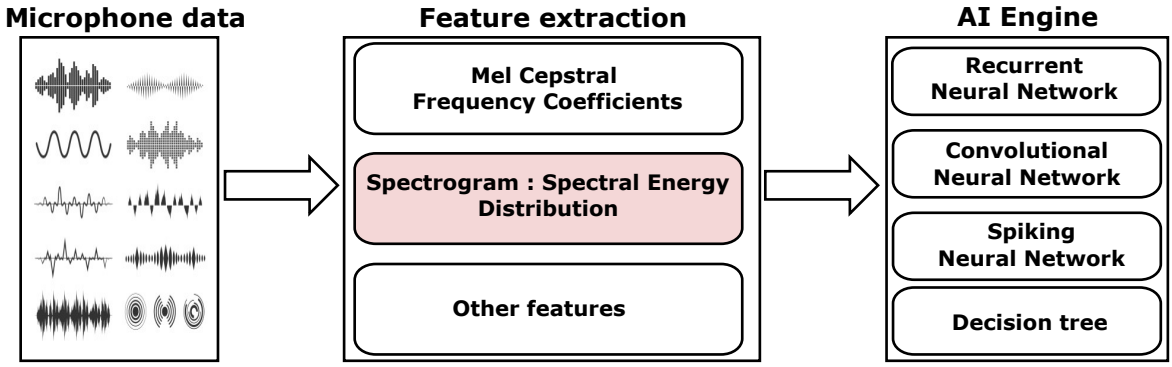


Figure 3.1 Generic architecture of keyword spotting systems

The Mel Frequency Cepstral Coefficient (MFCC) [1], [2], [3] are doubtlessly the most used features in keyword spotting. The extraction of these coefficients is done through a digital processing chain consisting of framing, applying the FFT, filtering through a log-Mel scale, and finally applying the DCT transform.

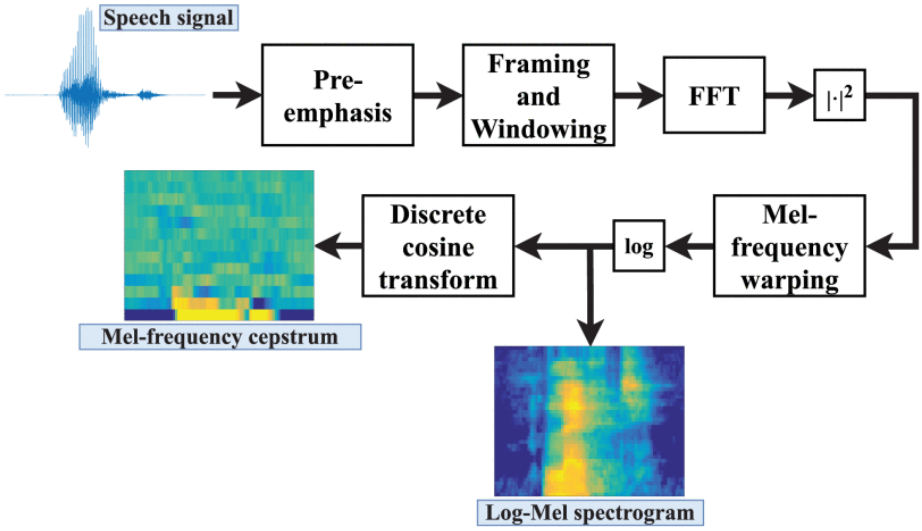


Figure 3.2 Classical pipeline for extracting log-Mel spectral and Mel-frequency cepstral speech features using the fast Fourier transform (FFT) [2].

As mentioned before and described in Fig. 3.2, the MFCC extraction process requires multiple power-hungry operations such as the FFT. Moreover, the works based on the MFCC computation usually use a filter bank composed of 20 to 128 channels like in [4], [5], and [6]. From these two statements, we can conclude that MFCC features cannot be well adapted to low-power applications without further optimizations to limit hardware complexity.

Other features characterizing the audio can be the prosodic features [7], [8]. The prosodic features can take the form of the energy, pitch, or duration... The energy is the prosodic parameter mostly used for audio identification. In more details, the continuous spectral analysis of the audio signals through time can be used as a tool to detect the presence of speech or sound when we remark an increase in the intensity of the spectral content. This technique was used in multiple systems but was implemented in different ways. Typically, the system in [9] proposes a feature extraction unit based on a set of 16 bandpass Gm-C filters and an averaging circuit. Another system is presented in [10], which performs keyword spotting while extracting the energy in different frequency bands. This latter system performs the energy extraction using a set of 8 discrete-time FIR filters and a classic energy extractor performing the following formula: $\sum |x|^2$. This system is targeting the recognition of 10 keywords while maintaining a good trade-off between system complexity and recognition accuracy.

3.2 Target specifications

The objective of our system is the recognition of 10 keywords while maintaining a low hardware complexity and low power consumption. The system presented in [10] has demonstrated the ability to recognize up to 12 different audio classes (10 keywords + unknown + noise) by extracting spectral spectrograms that are injected into a 6 filtering steps convolutional neural network. In the same theme, we have decided to choose the energy distribution as the key parameter to extract.

Generally speaking, the computation of the energy distribution follows a process of four steps: filtering to isolate the spectral content in each frequency band, power computation, framing, and finally energy extraction. Then, the generated per-band energies are combined to form a matrix to be transferred in parallel to a convolutional neural network to generate the classification results.

To sum up, we are targeting a keyword spotting system with a spectral-based feature extractor linked to a CNN. To be competitive or better than the different state-of-the-art keyword spotting systems shown in the previous chapter, we have set as a final goal the recognition of 10 keywords including the background noise with an accuracy higher than 90%. Besides, the challenging point will be to design this system with the lowest possible hardware complexity, which means the lowest possible power consumption. For this reason, multiple optimizations will be done in the aim of degrading the filtering stage by modifying its key parameters like the bandwidth and the out-of-band attenuation to decrease the system complexity while meeting the target specifications. Besides the filtering stages, all the needed logic operations will be optimized for the objective of better power efficiency. With this said, we are expecting a power consumption budget of less than 1μWs to be competitive or better than the state-of-the-art.

Most of the work will be focused on the optimization of the feature extractor with the objective of decreasing the complexity and power consumption. However, for the classification, we are using a state-of-the-art convolutional neural network adopted for low-power systems. The thesis work covers the design of the feature extractor, but the CNN was used only for the validation step to test if the extracted features are sufficiently relevant to achieve the specified recognition accuracy.

The main target of this work is to optimize the design of the feature extraction unit for a low hardware complexity and better power efficiency. The feature extractor is combined with a state-of-the-art convolutional neural network, adopted for low-power systems, to test if the extracted features are sufficiently relevant to achieve the specified recognition accuracy.

3.3 Abstract description of the proposed system

The first fundamental step is the development of a software model of the complete keyword spotting system to be able to specify the needed design parameters that will be used when passing to the hardware implementation. The model was built step by step using *Matlab*®, and the main objective of this modeling step was to have a global view of the system and to test it on real-time audio signals to have a preliminary prediction of the classification accuracy results. Moreover, the model allows the study of the variability of the system performance as a function of the different key parameters, and through this study, we were able to specify a value range of the latter parameters to have a better trade-off between accuracy and power consumption.

3.3.1 Spectral-based feature extractor

The proposed system is based on spectral processing of the audio signals to construct the corresponding spectrograms. These spectrograms can be either constructed in a digital way or an analog way depending on the constraints fixed by the system. So, the system is built in a digital-friendly environment to provide a high scalability degree to technology nodes and also because the CNN is only compatible with digital bit-coded signals. Moreover, the system in [10] has proven a competitive recognition accuracy using a digital-based system.

Fig. 3.3 shows an abstract idea about the architecture of the system to design. This system is responsible for generating the needed spectrograms through a processing chain starting with the audio quantization using the convenient analog-to-digital converter, then the digitized signal will enter a filter bank to spectrally decompose the audio signals over multiple frequency bands. After that, the filtered signals enter a block responsible for computing the instantaneous power and performing the accumulation to calculate the per-band energies. Then, these energy values are regrouped together to form the spectrogram. This spectrogram is input to a convolutional neural network to recognize the needed keywords. The classification results will inspect the precision and relevance of the extracted features in categorizing the different audio classes.

Another important point to specify is the method to perform the logic operations needed to construct the different functions shown in Fig. 3.3. In general, digital signal processing can be done either in a clocked way to make all the logic operations synchronized on a global clock signal and this is what defines the discrete-time DSP. The other method is the continuous-time DSP which is based on asynchronous processing allowing the use of continuous digital states instead of discrete samples. In this method, the output of the logic blocks varies only with the change of the input signal, otherwise, it stays the same. In more

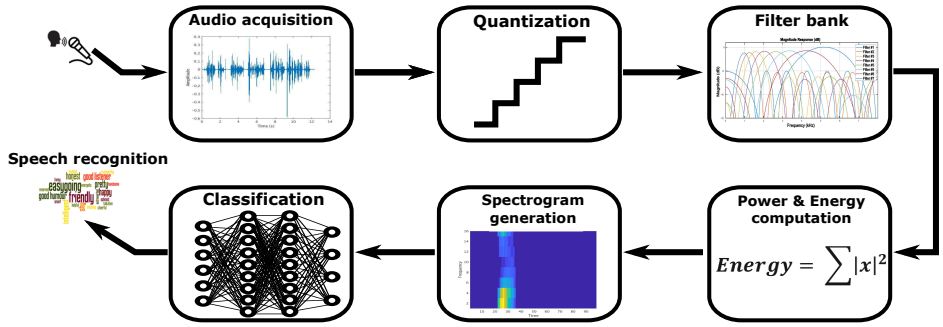


Figure 3.3 Abstract model of the proposed system.

details, the operation of the continuous-time logic will be completely gated by the input signal activity. Each of these two mentioned methods can be efficient in a certain scenario. For example, DT-DSP can be efficient when we are treating signals with a high data rate. However, continuous-time DSP systems become more efficient for the processing of signals containing long periods of silence and sparsely distributed information.
Continuous-time digital processing will be described in more details in the next section to show its different characteristics and further highlight the advantages in comparison to the conventional DT-DSP.

3.3.2 Continuous-time digital signal processing

3.3.2.1 Context

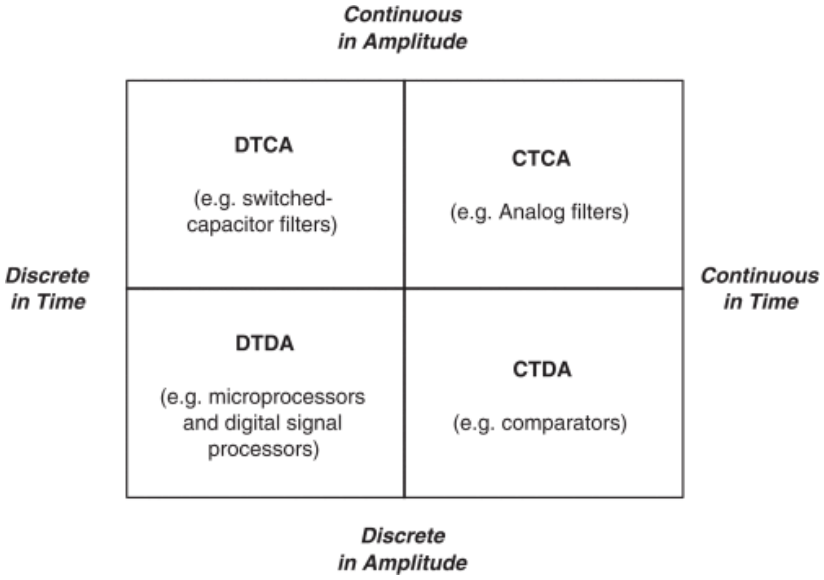


Figure 3.4 Different signal processing domains [11].

Signal processing can be done in multiple methods depending on the nature of the signal. The four principal processing methods are described in Fig. 3.4.

- **Discrete-time Discrete Amplitude (DTDA)**: This method profits from the digitization of signals to get rid of the analog limitations such as noise, matching... This method requires a sampling on both time and amplitude bases. Thus, it adds the imperfections linked to the sampling typically the aliasing problem.
- **Continuous-time Continuous Amplitude (CTCA)**: This method processes signals of an analog nature with a total absence of any sort of sampling. This technique suffers from imperfections originating from the analog environment such as matching and noise...
- **Discrete-time Continuous-Amplitude (DTCA)**: This method can be used in systems such as switched capacitors. It suffers from the combination of the limitations existing in the two previous processing techniques.
- **Continuous-time Discrete amplitude (CTDA)**: The last possible combination is to discretize only the amplitude without time. This method avoids the aliasing problem generated from time discretization such as aliasing and reduces the quantization error impact. Moreover, this technique is completely dependent on the input signal activity quantified by the amplitude, thus the power consumption will be majorly gated by the input activity. As a result, this kind of systems will be suitable for signals with sparse information distribution and long silence periods, such as audio signals, medical signals...

3.3.2.2 CT digital signal processing chain

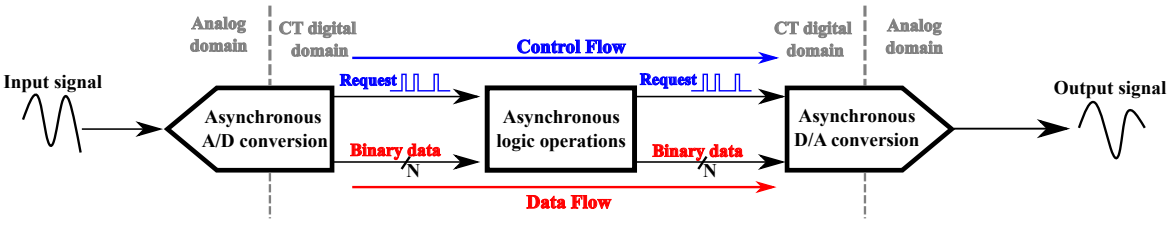


Figure 3.5 Continuous-time digital signal processing chain.

In general, the CT-DSP systems are designed in the form of a hierarchical processing chain. This chain, illustrated in Fig. 3.5, can be decomposed into three main building blocks. In such systems, the analog input signal is at first decoded using an asynchronous

ADC, representing the limit between analog domain and CT digital one, to generate the request signal that serves as a control signal to sequence the different further clockless logic operations performed on the binary data. Finally, the processed digital signal is reconstructed using an asynchronous DAC, representing the boundary with the analog domain. The CT processing is completely event-driven by the request signal and does not represent any synchronizing relation to a defined clock domain.

3.3.2.3 Asynchronous A/D conversion: Level-crossing ADC

The level-crossing ADC (LC-ADC) is the block responsible for decoding asynchronously the input signal into a pair of bits (Req, Dir) indicating the crossing of some predefined threshold levels i.e., quantization levels, and the crossing direction, respectively. In that context, the events are defined as the crossings, thus the number of generated events directly depends on the number of threshold levels i.e., ADC resolution. The LC-ADC is an asynchronous delta encoder composed of two analog comparators continuously comparing the input signal to a predefined set of high and low amplitude thresholds, to generate the Req and Dir signals, representing the request and data respectively. Then, the generated pair of bits is injected into a digital unit that refreshes the values of the low and high thresholds defining the comparison window. This type of ADCs is called floating window LC-ADC [12], [13]. The operation principle of the LC-ADC is shown in Fig. 3.6, where a sample is taken whenever the input signal fulfills the following conditions: $input \geq V_H$ or $input \leq V_L$. The thresholds are updated by $\pm$LSB, depending on the direction of the crossing. The implementation and operation shown in Fig. 3.6, are a specific example for our application case, but multiple other implementations are possible.

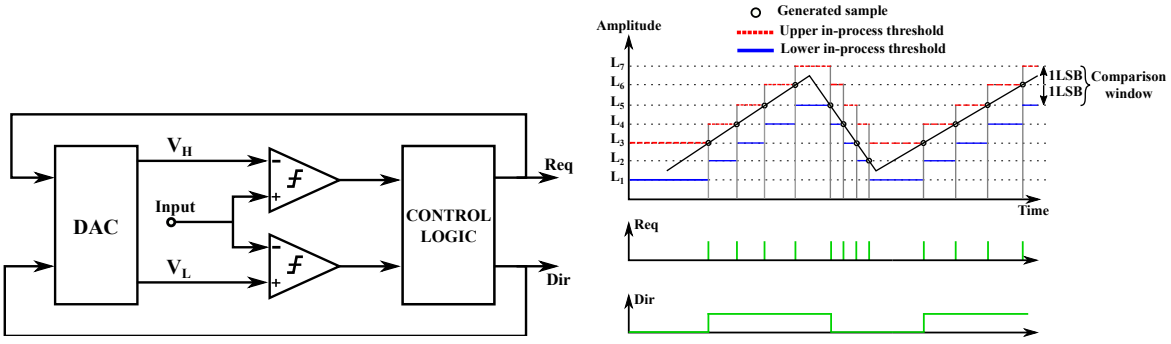


Figure 3.6 Level-crossing ADC and its operation principle.

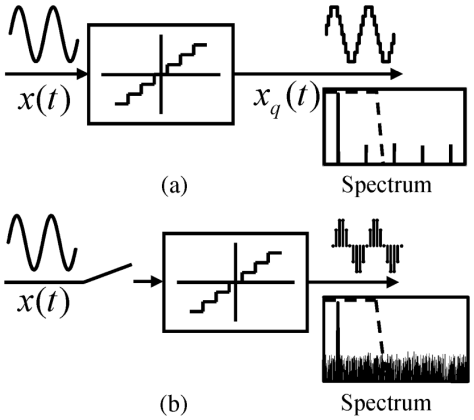


Figure 3.7 Impact of CT-DSP on the aliasing problem [14].

3.3.2.4 Comparison between discrete-time ADC and continuous-time ADC

In conventional discrete-time digital signal processing, we generate uniform samples synchronized by the sampling clock, which is usually set to be at least twice as fast as the fastest component in the input signal. This is defined by the Nyquist theorem. Besides sampling, the signal is also quantized as the amplitude is also coded on several levels defined by the resolution of the ADC ($Number\ of\ quantization\ levels = 2^N/A_{Max}$, with N the ADC resolution, and A_{Max} is the maximal peak-to-peak amplitude). In such systems, if the time sampling is not fast enough, it will introduce the aliasing problem, as depicted in Fig. 3.7 (b). However, a high-frequency clock induces a higher power consumption.

In CT DSP, the input signal goes only through amplitude quantization. The generated samples do not contain any time information, thus the aliasing problem automatically disappears, so the spectrum appears to be cleaner than the DT-generated one, as shown in Fig. 3.7 (a). Besides providing an alias-free environment, the CT DSP offers also an adaptive power consumption feature. In more details, the CT processing deals only with the variations in amplitude, which means that if the input signal activity is low, the system will generate fewer samples and the power consumption will be lower. Hence, the power consumption of the CT DSP systems will be completely gated by the signal activity. To sum up, the CT DSP is more suited for event-driven keyword spotting applications. However, the challenge in this case will be to adapt the design for a clockless environment.

3.3.2.5 Asynchronous logic: FIR filters

The asynchronous CT DSP offers a limited choice of possible logic operations that can be performed on the encoded signal since it does not contain any time information. However, filtering is one of the most used processing operations, that can be applied with a completely

clockless behavior. More specifically, the filters that are concerned in this case are digital Finite Impulse Response (FIR) filters. This choice is based on the fact that these filters were the first ones to be possibly implemented in an asynchronous environment. This is demonstrated in [11] and [15].

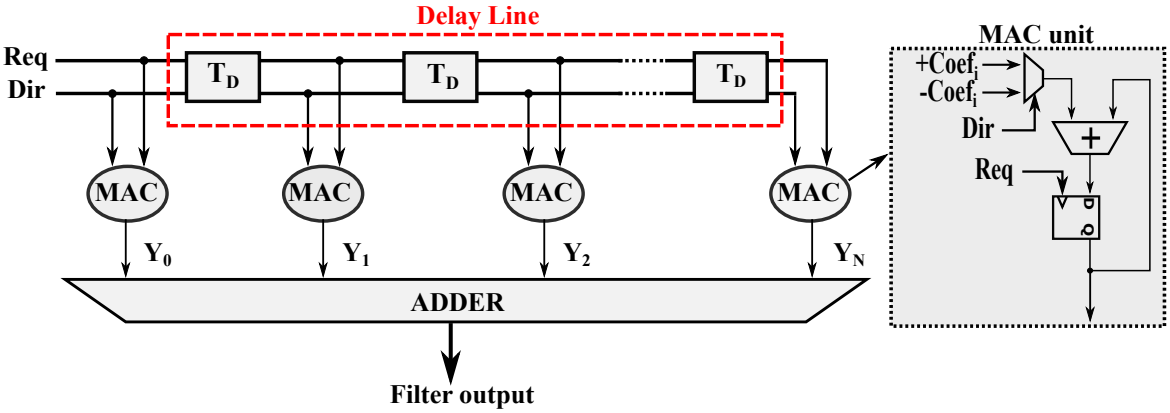


Figure 3.8 Architecture of the FIR filter.

The architecture of these CT FIR filters was introduced in [11] and was detailed in [15]. Fig. 3.8 shows the architecture of these filters. The tokens (Req, Dir) are successively delayed using a series of analog delay taps (T_D). These delays are completely clockless and are based on a current-controlled capacitance discharge. More specifically, each delay tap is formed by a succession of granular delays. These delays represent the smallest timestep between two successive events (an event here is a pulse at the Req signal). The maximum granular delay value depends principally on the maximum input frequency f_{input} and LC-ADC quantization resolution N, through the following formula: $T_{GRAN} = \frac{1}{2^N \pi f_{input}}$.
Afterwards, the delayed tokens are injected into their corresponding multiplication and accumulation (MAC) units. The MAC unit architecture, depicted in Fig. 3.8, shows that the multiplication is replaced by an addition+accumulation. Precisely, the output of each MAC (Y_i) can be described using the following equation:

$$\begin{cases} Y_i = Y_i + Coef_i, & if\ Req_i = 1\ and\ Dir_i = 1 \\ Y_i = Y_i - Coef_i, & if\ Req_i = 1\ and\ Dir_i = 0 \end{cases} \tag{3.1}$$

Finally, these MAC outputs are summed together to form the filter output.
One of the biggest advantages of utilizing this MAC architecture is to avoid using multipliers, which can be very complex to implement at a hardware level. Another point is that the operation of these FIR filters is completely event-driven and independent of any external clock.

As a summary, continuous-time digital signal processing systems provide an opportunity to have selective processing to improve power efficiency. These systems are triggered to start the processing whenever a relevant event is present at the input, thus the power consumption will be strongly dependent on the input signal activity. However, in conventional synchronous systems, The input data is sampled at each rising or falling edge of the main clock. This means that the processing is always on without taking into account the relevancy of the input signal. Consequently, for signals with long silence periods and sparsely distributed information, the consumption of CT-DSP systems is drastically reduced compared to conventional sampled systems.

3.4 Event-driven continuous-time spectral-based feature extractor:

To benefit from the different advantageous points of CT-DSP systems, the feature extraction unit will be based on continuous-time processing to extract the spectrograms. In the audio context, the architecture of the feature extractor is depicted in Fig. 3.9.

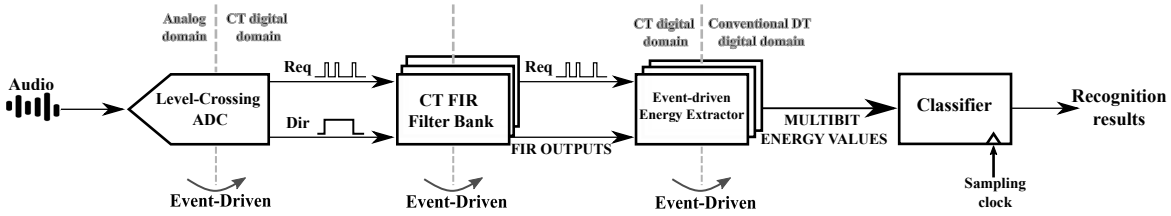


Figure 3.9 Continuous-time event-driven feature extraction unit.

The feature extraction unit is composed of three hierarchical event-driven processing stages. The first stage is the level-crossing ADC responsible for the asynchronous decoding of the audio input into Req and Dir tokens. Then, a CT digital FIR filter bank receives the pair of bits (Req, Dir) to perform the decomposition of the audio spectral content through the frequency bands of interest. The filters outputs are then propagated along with the Req signal towards the next block that performs energy extraction. Then, the energy values are received and sampled by the classifier to generate the recognition results.

3.4.1 Simulation environment: Oversampling to simulate continuous-time systems.

After elaborating a first idea about the top view of the feature extraction unit targeting Keyword spotting application, the following step will be to create a software model to validate

its operation and extract the preliminary performances. For this reason, a *Matlab*® model will be built step by step until the completion of the system. As much know, modeling software, such as *Matlab*®, can deal only with discretized binary data. Thus, the biggest challenge is to simulate continuous-time processing.

As a solution, we decided to oversample the input audio samples with higher frequency to collect sufficient data without missing any important information. The audio test samples, taken from the GSCD database, are already sampled to a frequency of 16kHz.Thus, by using the Nyquist theorem, the fastest component in these audio signals will not exceed 8kHz. By applying the theoretical formula of the granular delay on this frequency, this delay will vary between 7.8125μs and 122.07ns, for an ADC resolution ranging from 4 to 10 bits. So, to cover this latter bandwidth, it was decided to resample the audio signal with an oversampling ratio of 1000, which means a frequency of 16MHz and a time step of 62.5ns. This oversampling frequency will stay sufficient to capture the totality of the events in the Req signal without any distortion, as long as the ADC resolution does not exceed 10 bits. The ADC resolution will be set in the following sections and will help us to adjust the oversampling ratio if needed.

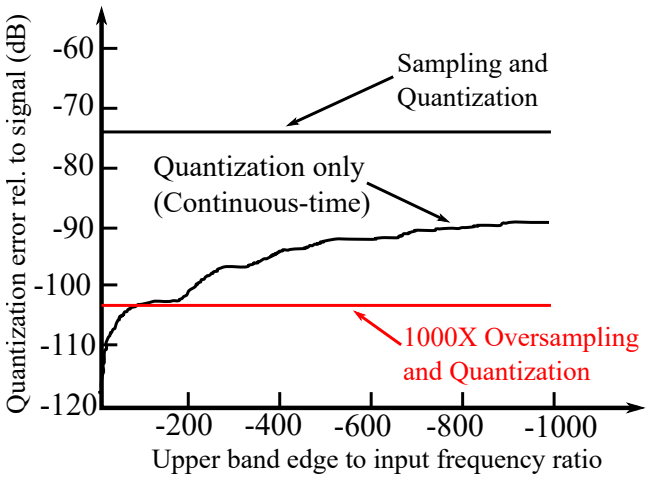


Figure 3.10 Impact of the sampling and oversampling on the quantization error compared to continuous-time amplitude sampling [16].

Moreover, the oversampling will help to get rid of the aliasing problem. Fig. 3.10 describes the variation of the quantization error as a function of the ratio between the upper band edge and input frequency for different scenarios. This sketched variation concerns a 12-bit resolution case. The sampling at the Nyquist rate introduces a higher quantization error compared to the quantization-only scenario [16]. However, when applying a 1000X oversampling, we can see that this error is reduced to a lower level than the continuous-time scenario, except for a frequency range that will not be of interest in our work on audio

signals. To sum up, the aliasing problem will be diminished with oversampling. Hence, the continuous-time processing can be approximated and simulated through a 1000-factor oversampling.

3.4.2 Level-crossing ADC parameters.

3.4.2.1 Size of the comparison window.

One of the critical drawbacks of the level-crossing ADC is the high sensitivity to noise. With the presence of very noisy signals at the input, the LC ADC starts to generate multiple false events that do not contain any important information helping to process or reconstruct the input signal. These generated events induce a high false alarm rate, which increases automatically the power consumption.

The classic floating window LC-ADC, presented in [12], has a comparison window of one LSB. But, if the noise exceeds this level, the ADC will suffer from high false event generation. So, a solution was described in [17] that considers increasing the comparison window to reduce noise sensitivity. For audio signals from the GSCD database, this window was set to 2 LSB which was sufficiently high to get rid of the noise effect without any loss of information and without distorting the signal to be reconstructed.

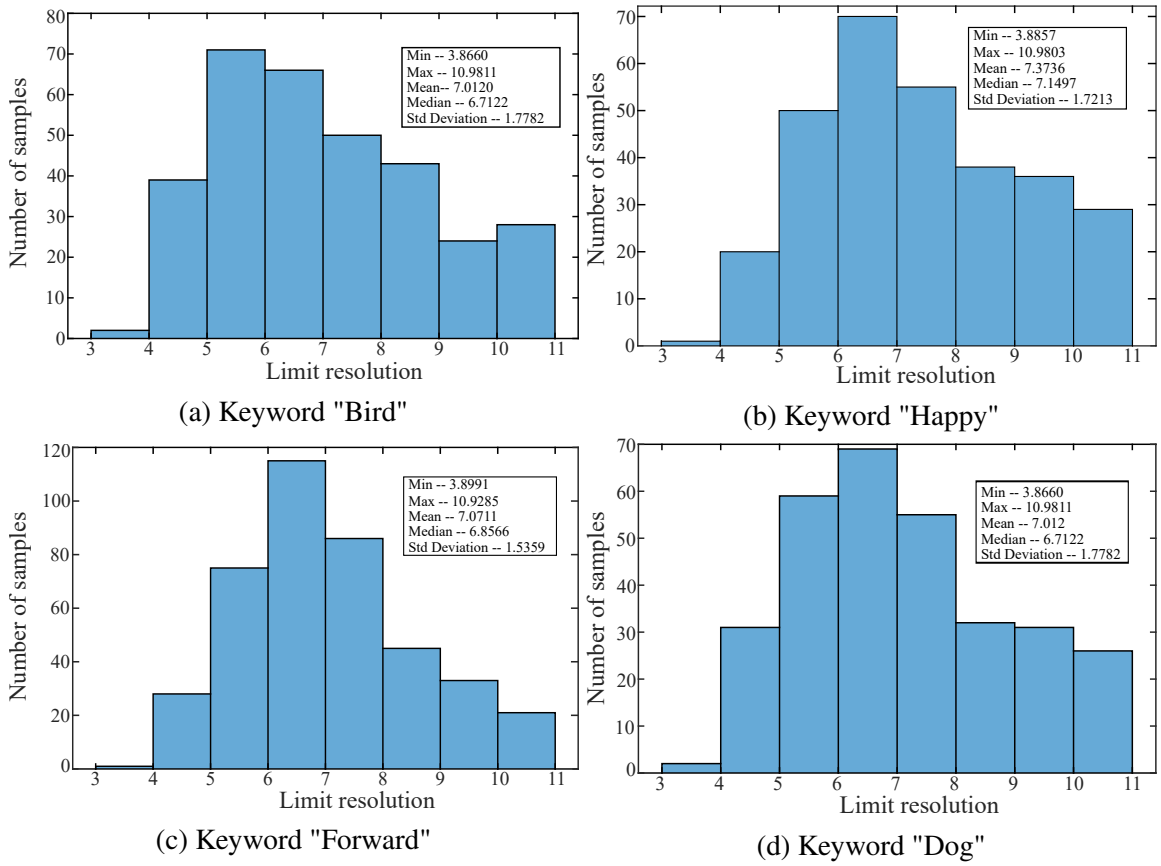
3.4.2.2 Quantization resolution.

To specify the optimal LC-ADC resolution, a comparison study is performed to find the highest resolution possible at which the LC-ADC stays advantageous compared to the conventional discrete-time ADC. This study is based on calculating the number of generated events (pulses in the Req signal) as a function of the resolution. For a sinusoidal signal, the number of events generated by each type of ADCs during one period is defined by the following equations:

$$\begin{aligned} N_{events}\ \ in\ \ LC-ADC &= \frac{V_{MAX}}{2^{Nbits}} \\ N_{events}\ \ in\ \ DT-ADC &= \frac{1}{F_{sampling}} \end{aligned} \tag{3.2}$$

With V_{MAX} the maximum amplitude value, N_{bits} the resolution, and $F_{sampling}$ the sampling frequency. The number of events generated by the DT-ADC is constant, while in the LC-ADC it varies with the resolution and amplitude.

The histograms, depicted in Fig. 3.11, shows the result of the latter comparison study. Each histogram shows the distribution of the limit resolution over 500 audio samples for



(a) Keyword "Bird"

(b) Keyword "Happy"

(c) Keyword "Forward"

(d) Keyword "Dog"

Figure 3.11 Distribution of the resolution at which level-crossing ADC generates more events than a 16kHz conventional DT ADC, for 500 audio samples.

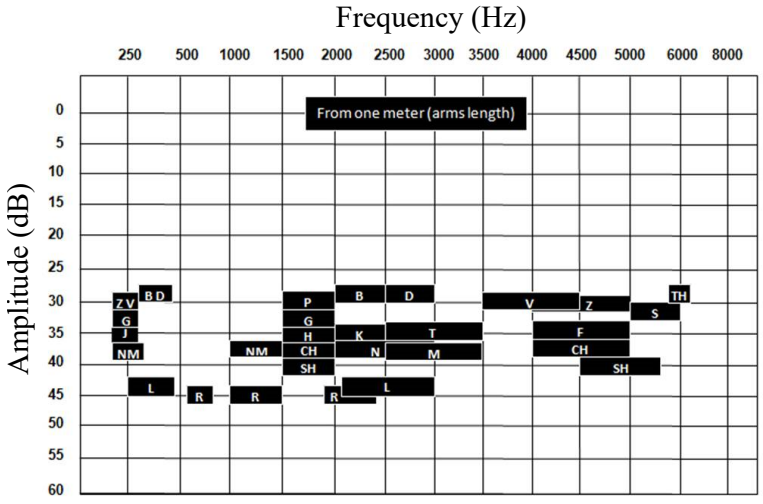
a specified keyword. This limit resolution represents the resolution value beyond which the level-crossing ADC generates more events than the conventional discrete-time ADC, sampled at the Nyquist rate (16kHz, because the maximum frequency of audio samples from the GSCD database is 8kHz). For this study, we have randomly chosen four keywords "Bird", "Happy", "Forward" and "Dog", with a different number of syllables. For all the keywords, the distribution of the limit resolution follows a nearly Gaussian shape. As shown in the different histograms in Fig. 3.11, the mean limit resolution for each keyword is around 7bits, which means that an LC-ADC with a resolution lower than 7 bits will, on average, generate fewer events than the DT ADC.

To make sure that level-crossing ADCs are beneficial compared to DT ADCs, a 6-bit LC-ADC is then supposed to be a good tradeoff between the number of events and the total accuracy of the system. This will be proven in Section 3.5.2 of this chapter. The resolution can be decreased further to reduce the activity but at the expense of reduced system-level accuracy.

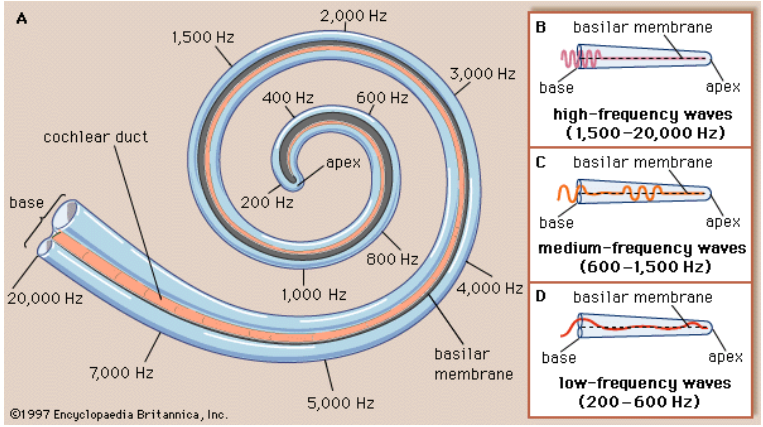
3.4.3 Model and parameters of FIR filters.

3.4.3.1 Conversational speech spectral bandwidth and its distribution.

The human audio bandwidth covers a frequency range going from 20Hz to 20kHz [18]. However, conversational speech spectral content can be found in a low-frequency range. Fig. 3.12a depicts the speech audiogram for consonants spoken from a one-meter distance. This spectrum shows that these vocal letters have an equivalent spectral signature contained in a frequency bandwidth ranging from 200Hz up to 7000Hz. Hence, for keyword spotting application the spectral analysis of audio samples will be focused on this frequency range.



(a) Speech audiogram for spoken consonants [19].



(b) Frequency distribution inside of the human cochlea [18].

Figure 3.12 Speech spectrum and characteristics of the human cochlea.

The speech analysis over the specified frequency range cannot be done once at a time. However, this speech bandwidth should be divided into multiple frequency bands to be able

to apply filtering operations and extract the needed parameters from each band independently. One of the main challenges is to choose the distribution methods of these frequency bands. For this reason, the work in [19] presents the frequency distribution inside the human cochlea. The result of this study is shown in Fig. 3.12b. In this figure, we can see qualitatively that the spiral morphology of the cochlea leads to a logarithmic frequency distribution inside of it. Moreover, this qualitative remark is complemented by a mathematical analysis done by Professor Donald D. Greenwood in 1990 [20]. This study has found a correlation between the frequency value and the fractional distance inside the cochlea, described through the following equation:

$$F_{sound} = C(10^{ax} - K) \tag{3.3}$$

F_{sound} is the sound frequency at the specified x position inside of the cochlea, C is a constant specified by the sound frequency and the considered upper-frequency limit, a is a variable specified by the slope of the straight part of the human cochlea, x is the fractional distance, calculated from the apex, inside the spiral shape of the cochlea and K is a constant depending on the considered lower frequency limit.

The Greenwood equation has built a logarithmic relation between the audio frequency and the position in the human cochlea. Thus, to emulate the cochlea operation, we should adopt a speech analysis over multiple frequency bands distributed on a logarithmic scale.

3.4.3.2 12-band filtering model.

As a first approach, speech analysis is performed over a bandwidth ranging from 200Hz to 7kHz. Following the conclusion of the previous section, this frequency range will be divided into multiple logarithmically distributed bands. The number of bands represents a key parameter that will specify the selectivity, order, and complexity of the filtering stages. In more details, the bigger the number of frequency channels, the more selective are the filters in each band, so the more precise the speech analysis.

Multiple spectral-based keyword spotting systems adopt a filtering model based on the spectral decomposition through at least 16 frequency bands, like the works presented in [21] and [22]. To be competitive with these solutions and reduce the system complexity, the number of bands will be reduced to 12 while showing that 12 band-filtering is sufficient to maintain a decent performance in terms of recognition accuracy. This choice should be validated by studying the filtering complexity and its hardware feasibility. As a first validation step, the filtering will be modeled by *Matlab*® using the "Filter Designer" tool, to extract the different parameters of the dedicated filters.

Each frequency band will correspond to a bandpass FIR filter. To simulate the filtering behavior in all the bands, the first step is to calculate the key frequencies for the bandpass filters, such as the central and border frequencies for each filter. In more details, the central frequencies will be calculated by performing the logarithmic division of the specified bandwidth into 12 bands. However, the border frequencies will be set depending on the filter selectivity. This selectivity is decided by the quality factor Q, which is defined by the following equation:

$$Q = \frac{F_{Central}}{F_{Stopband2} - F_{Stopband1}} \tag{3.4}$$

where $F_{Central}$ is the central frequency of the filter, $F_{Stopband2}$ is the higher stopband frequency at $-3dB$ and $F_{Stopband1}$ is the lower stopband frequency at $-3dB$.
From the equation above, it can be concluded that a higher quality factor will result in a tighter frequency band, which means the filter will be more selective.

The work in [10] concerns a keyword spotting system with a spectral-based discrete-time feature extraction unit. This system has demonstrated the capacity to recognize 10 keywords with an accuracy higher than 90%. The feature extraction unit in this system is based on the spectral decomposition of the audio input signal into 8 logarithmically distributed bands using a set of bandpass FIR filters. In this case, these filters were characterized by a quality factor equal to 1 [10]. The recognition accuracy is impacted by the selectivity of filters i.e. quality factor. Then, this parameter should be optimized (equal or higher than 1) to maintain the recognition accuracy higher than 90% in the 10 keywords case.

Another parameter required to build these filters is the out-of-band attenuation, describing the capacity of the filter to reject any signal out of its band of interest. This parameter is set to 20dB but will be gradually adjusted after studying the trade-off between three main key factors: The filter selectivity, the hardware complexity, and the recognition accuracy.

After gathering all the previously specified key parameters, the definition of the needed filters is now complete. These parameters were fed into the *Matlab*® "Filter Designer" tool, which was able to construct these filters and sketch the magnitude response depicted in Fig. 3.13. Moreover, Table 3.1 presents all the key frequencies of each filter. In addition, the "Filter Designer" tool generates the coefficients corresponding to each bandpass FIR filter. The number of the coefficients determines the order of the filter. These orders are also shown in Table 3.1. The order of the filters increases with their selectivity, which results in a maximum order of 250 for the lowest frequency and most selective filter #1.

Using this filtering configuration, the proposed system has demonstrated the recognition of 10 keywords with an accuracy equal to 94.9% compared to an accuracy of 90.87%

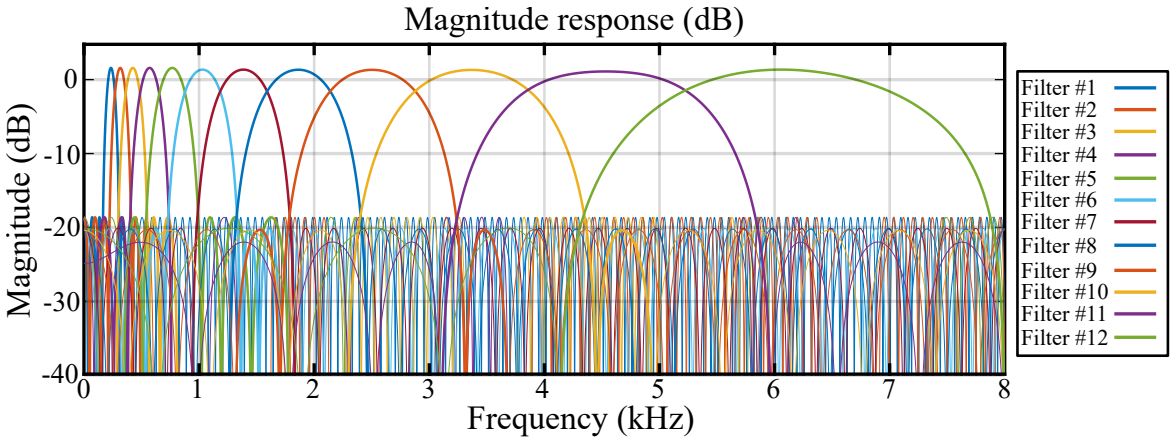


Figure 3.13 Magnitude response of the proposed bank of 12 digital CT FIR filters, with a stop-band attenuation of 20dB.

Table 3.1 Characteristic key frequency values for the 12-bandpass FIR filter bank.

Filter Number	$F_{Stopband1}$ (Hz)	$F_{Central}$ (Hz)	$F_{Stopband2}$ (Hz)	Filter order	Out-of-band attenuation
Filter #1	165.5	200	303.5	250	20dB
Filter #2	222.6	276.3	408.1	200	
Filter #3	299.3	381.7	548.9	140	
Filter #4	402.6	527.4	738.1	90	
Filter #5	541.4	728.6	992.6	75	
Filter #6	728.1	1006.7	1334.9	55	
Filter #7	979.2	1390.7	1795.2	40	
Filter #8	1316.8	1921.4	2414.4	31	
Filter #9	1771.1	2654.5	3246.9	21	
Filter #10	2381.7	3667.4	4366.6	16	
Filter #11	3203.1	5066.7	5872.4	12	
Filter #12	4307.6	7000	7897.5	11	

in the work presented in [22] and an accuracy of 93.8% in the system described in [10]. Unfortunately, the demonstrated accuracy performance requires a very high-order filter.

In fact, a high filter order will result in a very narrow passband bandwidth, which means a very selective filter. The high selectivity automatically means better filtering. However, this good filtering quality will be limited by the hardware complexity of these filters. In other words, the final objective of this thesis is not only to construct a software model of the keyword spotting system but also to design an equivalent low-power hardware implementation. Hence, having a filter order in the range of 250 will require 250 delay units and 250 MAC units with a dedicated adder, and this is impossible to implement in terms of hardware when the principal idea is to have a competitive low-power solution.

Therefore, one of the most challenging points in this work is to reduce the filters orders to the minimum possible, while keeping a good filtering quality allowing the extraction of relevant features able to differentiate between the different audio classes with a recognition accuracy higher than 90% in the 10 keywords case.

3.4.3.3 Downscaling to 8-band filtering model.

As mentioned before, the final objective is to have possible hardware feasibility with a low power profile of the keyword spotting system. For this reason, reducing the order of the filters is a mandatory requirement to scale down the hardware complexity.

The filter order is defined by its selectivity, which itself depends on the filter bandwidth and out-of-band attenuation. Hence, these latter parameters should be optimized without considerably affecting the quality of the filtering. The first point to optimize is the number of bands dividing the conversational speech bandwidth. The increased number of frequency bands will tighten the passband of each filter, which means filters will become more selective resulting in a high order. Thus, we decided to decrease the number of bands from 12 to 8, based on the work in [10] that showed the ability to recognize 10 keywords with an accuracy higher than 90% with only using an 8-band filter bank. In our case, the reduction to 8 frequency bands degraded the recognition accuracy from 94.9% down to 94%. This impact on accuracy is compensated by a decrease in the maximum filter order from 250 to 160. This is a significant reduction but still not sufficient enough to validate a possible low-complex hardware implementation. However, a further decrease in the number of bands increases the risk of degrading the filtering quality.

So, instead of reducing more and more the number of bands, we decided to optimize another key parameter, which is the total speech bandwidth. As seen in the Table 3.1, the low-frequency filters are the ones characterized with the biggest orders. Thus, if we increase the lower frequency limit of the speech bandwidth, the filters' orders should be decreased.

But before doing so, we need to check if we still covering an important speech spectral content. The previously specified speech bandwidth was specified from the data in Fig. 3.12a, but we can see in the same figure that most of the spectral content is concentrated between 1000Hz and 6000Hz. Based on this deduction, we decided to narrow the speech bandwidth of interest to an interval going from 840Hz up to 6250Hz. This compressed speech bandwidth improved the tradeoff between the recognition accuracy, which stayed higher than 90% in the 10 keywords case, and the maximum filter order that decreased down to 108. This is a notable improvement, but still not sufficient to stop the optimization of the filter bank parameters. The last parameter to optimize is the out-of-band attenuation. Up to now, this parameter was set to 20dB. This parameter defines the capacity of the filter to reject signals out of its passband. Hence, a higher out-of-band attenuation value means that the filter will be more selective, which results in a higher filter order. This deduction leads to the importance of reducing the out-of-band attenuation. This latter parameter was reduced step by step, with analyzing the impact on the filtering quality and recognition accuracy. After multiple reductions, we achieved a limit value of 5dB. Below this value, the filtering is completely degraded. Setting all the other parameters to their specified values and the out-of-band attenuation to 5dB, the system achieved the recognition of 10 keywords with an accuracy of 90.4%, while reducing the maximum filter order to only 16.

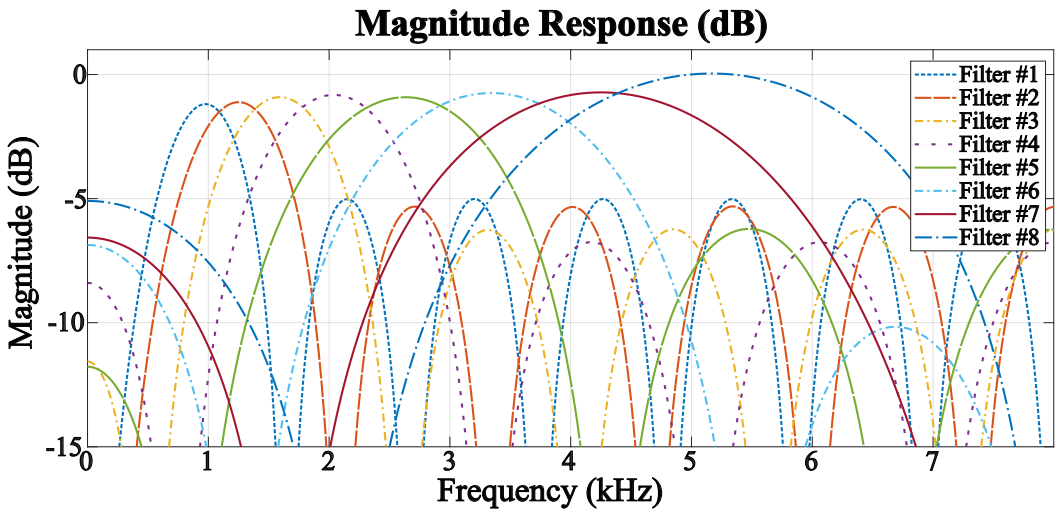


Figure 3.14 Magnitude response of the proposed bank of 8 digital CT FIR filters. The stop-band attenuation of -5dB soothes the design constraints to minimize the number of taps in each filter.

To sum up, the multiple optimizations, done using *Matlab*®, led to the down-scaling of the filter bank from 12 to 8 filters with relaxed hardware parameters. The magnitude response of the newly specified filter band is shown in Fig. 3.14. This filter bank covers a frequency

bandwidth ranging from 840Hz to 6250Hz and is characterized by an out-of-band attenuation equal to 5dB. The key frequencies and orders of each filter are specified in the Table 3.2. As a final result, the filter bank can be designed with a maximum order of 16, which translates in terms of hardware by the use of a maximum of 16 delay taps to design all the filters. This is the best tradeoff achieved between hardware complexity, filtering quality, and recognition accuracy.

Table 3.2 Characteristic key frequency values for the 8-bandpass FIR filter bank.

Filter Number	$F_{Stopband1}$ (Hz)	$F_{Central}$ (Hz)	$F_{Stopband2}$ (Hz)	Filter order	Out-of-band attenuation
Filter #1	573.8	840	1345.6	16	5dB
Filter #2	737.5	1079.5	1729.3	13	
Filter #3	947.8	1387.3	2222.4	11	
Filter #4	1218	1782.9	2856.2	9	
Filter #5	1565.4	2291.3	3670.6	7	
Filter #6	2011.7	2944.6	4717.2	6	
Filter #7	2585.3	3784.3	6062.2	5	
Filter #8	3322.5	4863.3	7790.8	4	

3.4.3.4 Coefficients binary coding

After setting the different filtering parameters, we were able to extract all the needed hardware specifications required for the on-chip design. The coefficients of each filter represent one of these specifications. Using *Matlab*® modeling, we extracted the exact values for the different coefficients. After multiple optimizations, the coefficients were finally truncated on 6 signed bits. This truncation led to a degradation in the precision of the values of the coefficients but without an important impact on the filtering operation and recognition accuracy that was maintained higher than 90% in the 10 keywords case.

3.4.4 Energy extraction process

As seen previously in Fig. 3.9, the audio signal undergoes an asynchronous decoding using the LC-ADC, and then the decoded signal is fed into an 8-bandpass CT FIR filter bank. The following step is to calculate the energy. More specifically, the energy is extracted in each frequency band to visualize the spectral content distribution across the speech bandwidth.

Usually, the energy of a signal is classically calculated through two steps. The first one is to calculate the instantaneous power values. The second step is to accumulate these

power values multiplied by the sampling clock period. Considering a signal X, the energy is mathematically expressed using the following formula:

$$Energy = T_{Sampling} \sum_{k=1}^{N} X(t_k)^2 \tag{3.5}$$

This energy calculation method is applied in conventional discrete-time digital signal processing systems. Nevertheless, this method cannot be adopted in the continuous-time case due to the absence of a time reference. This limitation in energy computing is considered also one of the main challenges faced during the implementation of our system.

In continuous-time digital signal processing, the logic operation is completely asynchronous and driven by the presence of events in the Req signal. These events are randomly distributed and depend only on the crossing of the predefined amplitude quantization level. Considering the random times of the occurrence of events and the absence of a fast clock to capture the events, it is impossible to quantify the time between successive events. Without this time value, the energy cannot be computed.

In our system, we are interested in calculating the energy of the different filter outputs. As described in Fig. 3.8, the filter output is the sum of the MACs outputs $(Y_0, Y_1, ..., Y_N)$. In more details, the output of a filter with order k will vary whenever there is a change in the output of the corresponding MAC units $(Y_0, ..., Y_k)$. Moreover, it is illustrated in the structure of the MAC depicted in Fig. 3.8, that Y_i varies by $\pm coef_i$ whenever there is an event present at Req_i, with Req_i being the Req signal delayed by the delay chain up to the ith delay. Each event at Req_i generates a change in Y_i, which itself generates a variation in the value of the filter output. Thus, the filter output is modified whenever there is a pulse at Req_0 up to Req_k. To track the filter output variations, we need to record all the events in all the corresponding delayed Req signals. More specifically, the filter output will vary according to the presence of an event in the signal concatenating all the corresponding delayed Req signals. This signal is defined as follows:

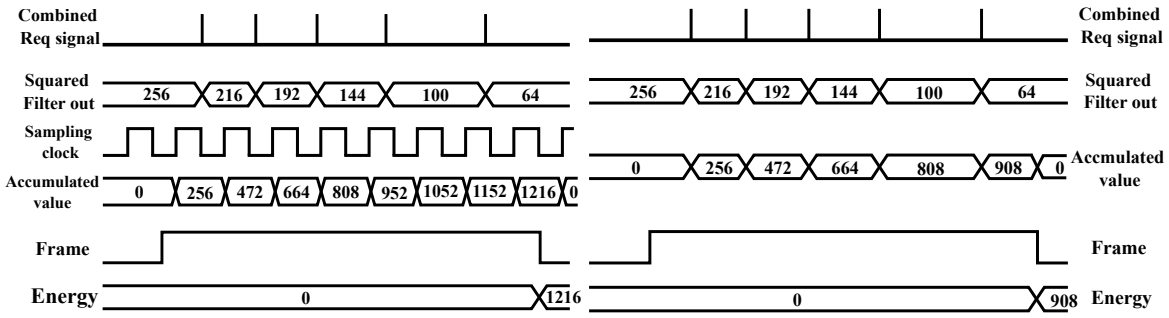
$$Combined\,Req\,pulses = (Req_0\,or\,Req_1\,or...or\,Req_k) \tag{3.6}$$

with k is the filter order. In other words, this signal creates a non-uniformly distributed time base to track the filter output and synchronize the energy calculation, which resolves the previous limitation.

The audio signals are usually divided into frames. This framing helps extract the audio key parameters in specified intervals to have a detailed analysis. In such wise, we will have an energy value in each frame per each frequency band. For audio analysis, these frames are

set to a duration of 25ms with 10ms overlap. Considering these values, the one-second audio samples taken from the GSCD database will be divided into 100 frames.

The chronograms depicted in Fig. 3.15 describe the energy calculation processes for continuous-time and discrete-time DSP systems. In Fig. 3.15a, the energy accumulation is synchronized on the sampling clock. In more details, the squared filter output varies only with the arrival of a pulse at the combined Req signal. But, these squared values are successively accumulated at each clock rising edge, while the frame signal is at the high state. Finally, at the frame falling edge, we collect the framed energy value, and the accumulator is reset to 0. In the left part, Fig. 3.15b shows the energy extraction process in the continuous-time case. After generating the filter output power, non-uniform samples are successively accumulated whenever there is an event at the Combined Req signal. The accumulation also continues as long as the frame signal is at the high level. Then, the energy value is collected and the accumulator is reset to 0 when the frame signal passes to a low level.

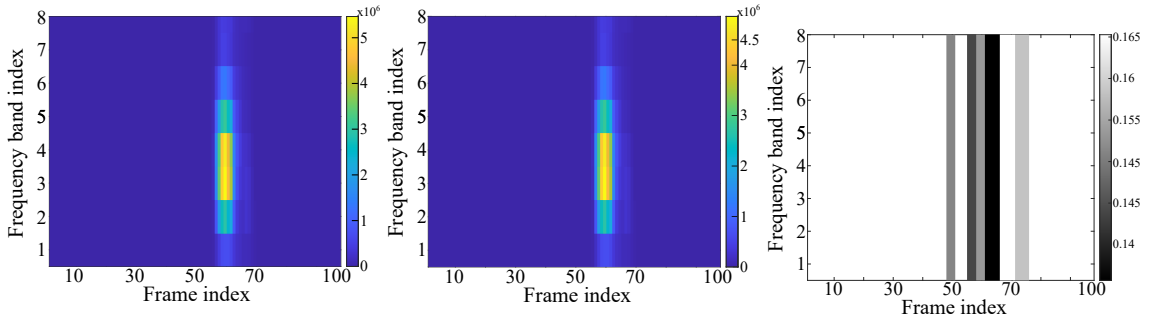


(a) Energy calculation in DT DSP. (b) proposed energy calculation in CT DSP.

Figure 3.15 Energy extraction flow for different signal processing systems

The proposed energy extraction method is completely event-driven and based on the accumulation of non-uniform samples without a regular time reference. However, this method introduces an error compared to the classical sampled method. This error originated from considering that the time between non-uniform samples stays the same. Thus, the arbitrary variation of the arrival time of events will not be taken into account in the estimated energy value. This error in the collected energy value is visualized in Fig. 3.15. This error can be critical if we are interested in calculating the exact energy value. But for keyword spotting, we are interested in displaying the energy distribution in each frequency band. So, even if the energy values are not mathematically correct, the spectral energy distribution will be rather similar. Fig. 3.16 shows spectrograms corresponding to the keyword "One", produced using the DT sampled and CT methods. On a visual scale, the two spectrograms match. In addition, the visual comparison is complemented with a quantitative analysis to calculate the difference between these two methods. This error is depicted in Fig. 3.16c. This difference

was calculated for multiple keywords, and on average it does not exceed 4.2%. Moreover, the impact of this error on the recognition accuracy in the 10 keywords case does not exceed 1.3%, while keeping an accuracy higher than 90%.



(a) Discrete-time spectrogram for "One"

(b) Continuous-time spectrogram for "One"

(c) Difference spectrogram

Figure 3.16 Spectrograms generated using DT and CT methods and their difference.

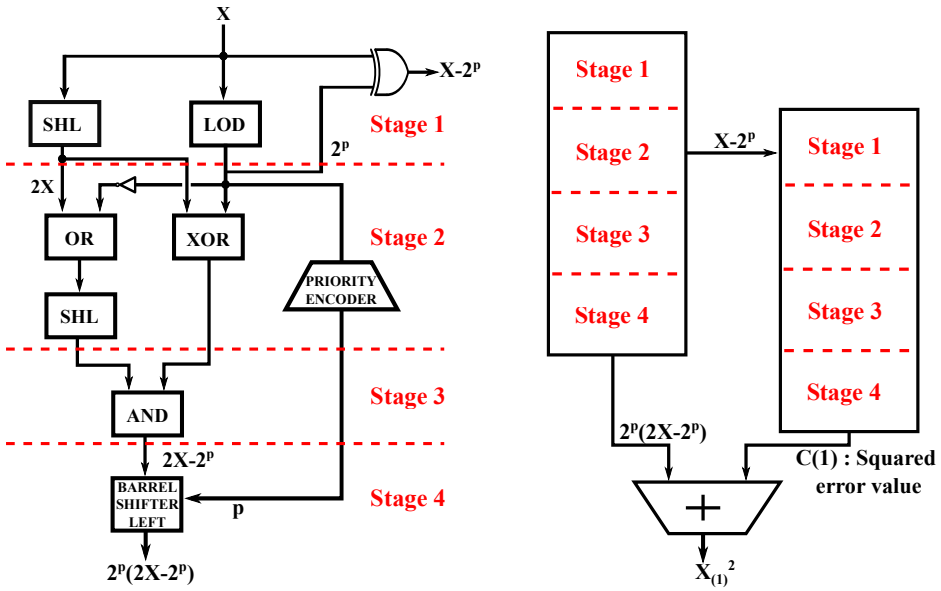
3.4.4.1 Approximate squarer

The first step to calculate the energy is to compute the instantaneous power. The classic solution for this operation is to use a multiplier. However, the architecture of the multipliers is complex and not suitable for low-power implementation. Thus, for the reason of hardware simplicity and low power consumption, the squared value of X can be approximated by the following quantity: $\tilde{X}^2 = 2^p(2X - 2^p)$ [23], with p the position of the leading one in X. The hardware implementation of the latter formula is shown in Fig. 3.17 (a). Instead of using a complex multiplier, the squaring function is done using some elementary logic units. This squaring function has demonstrated the estimation of the perfect squared value with an average relative error of 9.25%.

For the aim of increasing the squaring precision, we decided to adopt the 2-stage iterative version described in Fig. 3.17 (b). In this version, the previous value $\tilde{X}^2$ is added to the squaring error of the first stage $C(1)$ to improve the squaring precision. This iterative version has demonstrated approximating the squared value with a relative error of 1.37%. Furthermore, the error on the squaring operation can affect the absolute value of the computed power but will not change the relative spectral distribution of the energy.

3.4.4.2 Energy resolution

The energy must be binary coded on a specified number of bits to be compatible with the hardware implementation. For this reason, a quantitative study is performed on multiple



(a) Basic approximate squarer (b) Iterative approximate squarer

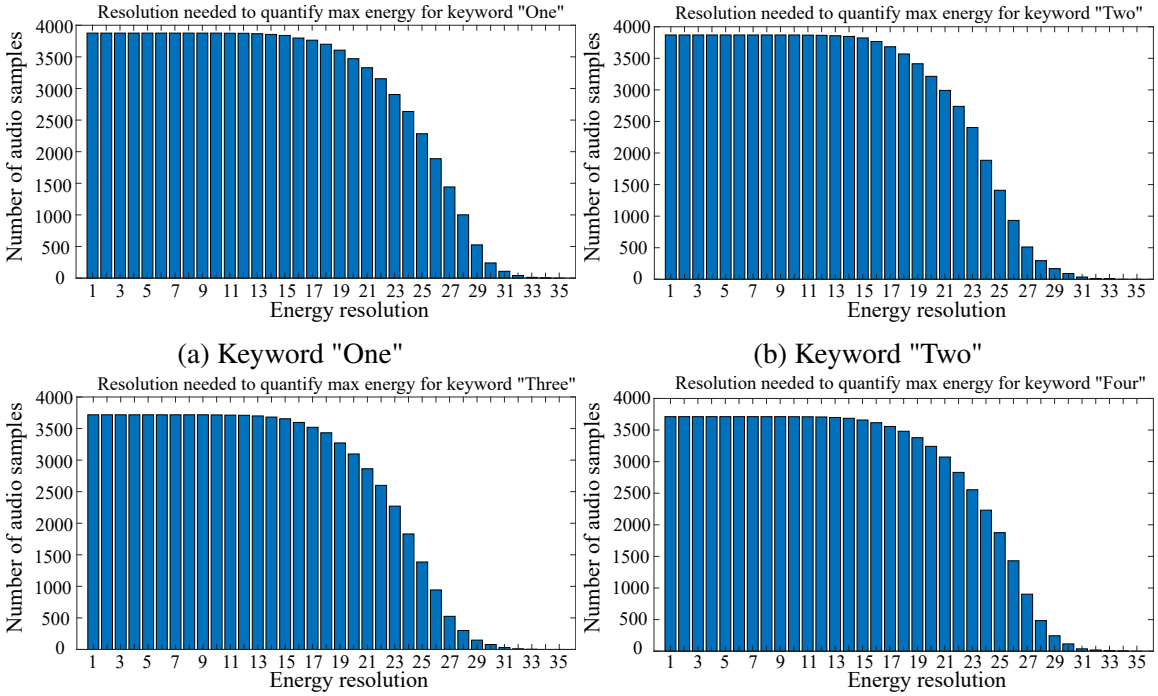
Figure 3.17 Architecture of the approximate squarer and the iterative version [23].

audio samples to extract the resolution equivalent to the maximum energy value for each audio sample. This analysis was performed on 4 keywords (One, Two, Three, and Four), and the result is illustrated by the histograms depicted in Fig. 3.18. Each bar in these histograms indicates the number of audio samples whose maximum energy value exceeds the value 2^N, with N the resolution of the specified bar. It can be seen from these histograms that for all keywords the maximum energy can be coded on 33 bits. This study was performed on more keywords with always the same conclusion.

However, not all the 33 bits can be significant. Thus, the energy values can be truncated on a lower resolution starting from the leading one position. But before that, we need to evaluate the impact of this truncation on the recognition accuracy. For this reason, Fig. 3.19 shows the variation of the recognition accuracy, in the 10 keywords case, as a function of the number of truncation bits. The accuracy stays higher than 90% as long as the energy resolution is higher than 6 bits. Thus, a 6-bit truncation is sufficient to maintain a good tradeoff between the recognition accuracy and hardware complexity.

3.5 Classification accuracy

The spectrograms, representing the distribution of the spectral content through the specified speech bandwidth, are fed into an AI classifier. The role of this classifier is to perform a mapping between the input audio classes and some predefined classes used for training. The



(a) Keyword "One"

(b) Keyword "Two"

(c) Keyword "Three"

(d) Keyword "Four"

Figure 3.18 Histograms presenting the resolution needed to decode the maximum energy value for different audio samples.

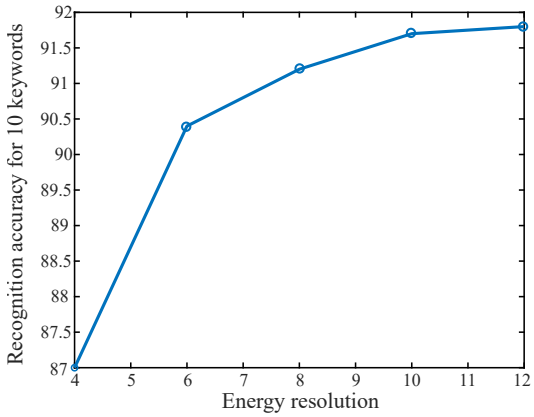


Figure 3.19 Impact of the energy resolution on the recognition accuracy in the 10 keywords case.

capacity of the features to differentiate between audio classes will be judged by the classifier recognition accuracy.

3.5.1 Convolutional neural network

For keyword spotting, we used a classifier with different filtering steps performing convolution operations between arriving data and stored weights. In more detail, the classifier is a convolutional neural network (CNN) with multiple successive filtering steps. The used CNN is similar to the classifier in [10]. The architecture of this CNN is depicted in Fig. 3.20. It consists of six convolutional layers, each layer is composed of k parallel 3x3 kernel layers, with k equal to 16 and 32 for the first two convolutional layers, respectively, and equal to 64 for the last 4 layers. Moreover, a max-pooling layer is inserted between each two consecutive stages. Finally, a fully connected layer, with a size matching the number of target keywords, is inserted after the last filtering step. Another reason that pushed towards choosing this CNN architecture is that they are compatible with low-power embedded implementation, based on the work presented in [24].

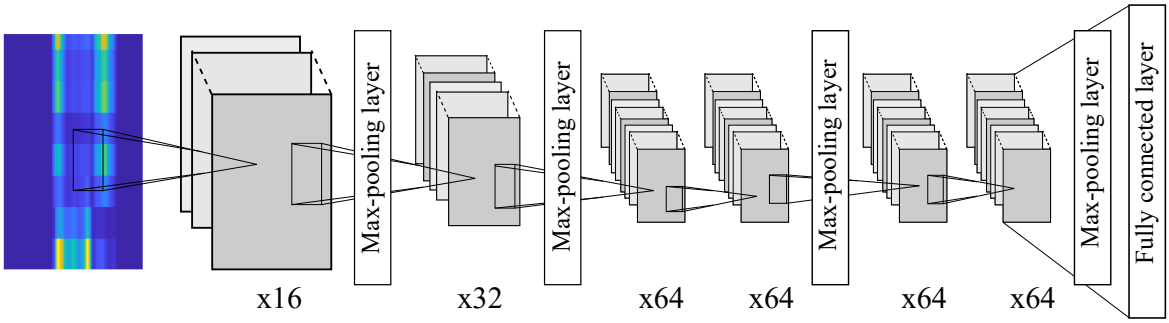
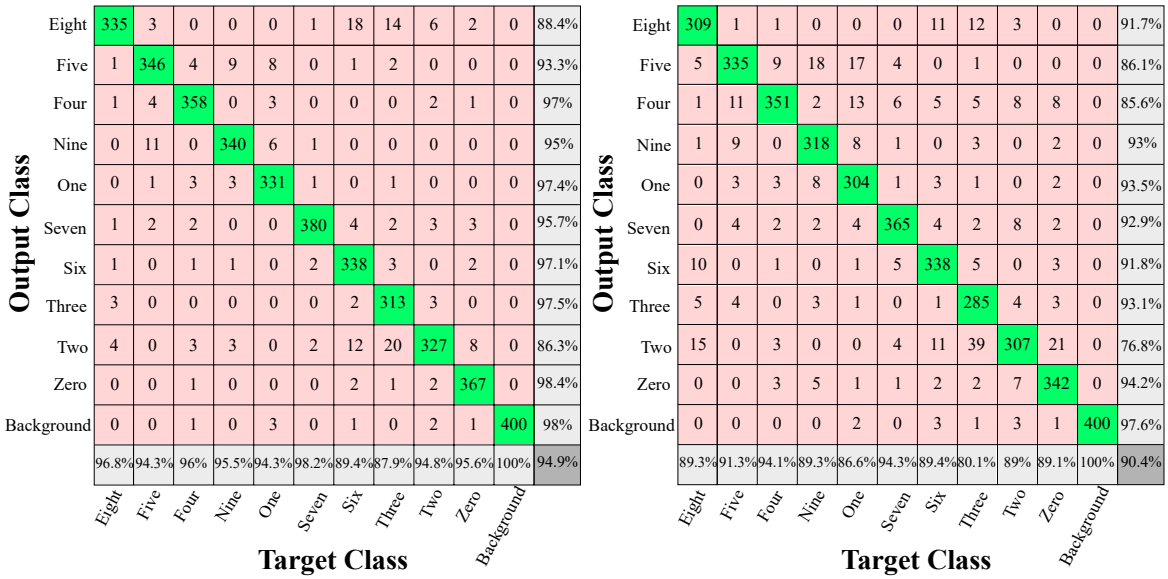


Figure 3.20 Architecture of the used convolutional neural network.

3.5.2 Recognition accuracy

To generate recognition results, the CNN, depicted in Fig. 3.20, will be continuously comparing the input audio classes with some predefined classes. In more details, the CNN operates in two different phases. The first one is the training phase. During this phase, the CNN model receives a large amount of data to learn all the characteristics of the data to be analyzed. This helps the CNN to develop the ability to make predictions later. The second operation phase is the inference. During this phase, the neural network will be processing new data that was not among the training dataset. The CNN will be analyzing this new data, based on its previous learning. Then, the model will generate a prediction result.

For test matters, the training and inference was performed on the GSCD dataset that contains multiple audio samples for each keyword, with a different pronunciation each time. For the 10 keywords case, we decided to use the numbers from 0 to 9 as test samples. The total number of audio samples including the background is 34,700, with 31,100 samples used



(a) Accuracy for the model with 12 filters.

(b) Accuracy for the model with 8 filters.

Figure 3.21 Confusion matrices for 10 keywords recognition including the background using the two filtering models.

for training and 3,600 for inference. At the end of the inference phase, we collect a confusion matrix representing the recognition accuracy of each keyword among others and the overall average accuracy. Fig. 3.21 shows the extracted confusion matrices in the 10 keywords case for the two previously discussed filtering models. Fig. 3.21a shows the recognition performance achieved using the 12-band filtering model. Using this filter bank, the system demonstrates the recognition of 10 keywords including the background with an average accuracy of 94.9%. This relatively high accuracy is faced with an impossibility of hardware implementation due to the high order of filters, which was equal to 250 at maximum. On the other hand, using the down-scaled 8-band filter bank, the system achieved a recognition accuracy of 90.4%, as shown in Fig. 3.21b. Between the two filtering models, there is a loss of 4.5% in the recognition accuracy. However, this accuracy degradation is compensated by a reduction of the filters' maximum order, which was equal to 16 in the maximum case. This guarantees the low-complexity and low-power hardware feasibility.

To summarize the accuracy performance, Fig. 3.22 describes the variation of the recognition accuracy with the number of keywords for multiple LC-ADC resolutions. The system becomes naturally more accurate with a lower number of target keywords, with an accuracy higher than 99% for the example of one keyword. Additionally, the variation of the accuracy in, Fig. 3.22, is represented for an ADC resolution ranging from 5 to 8 bits. Considering the 10 keywords case, the accuracy is higher than 90% for all the resolutions except the

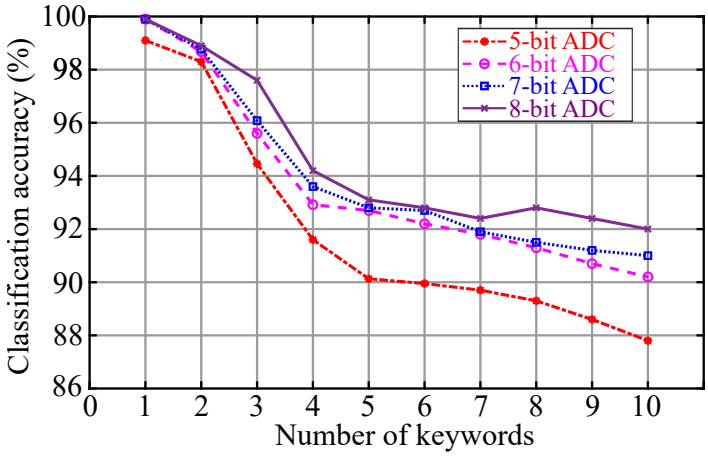


Figure 3.22 The impact of the ADC resolution on the recognition accuracy for different numbers of keywords.

5-bit LC-ADC. In our case, keeping the recognition of 10 keywords higher than 90% is considered as the gauge to decide on the different hardware parameters. For this reason, the previous choice of a 6-bit ADC is validated, as this is the lowest resolution used to achieve the required recognition accuracy.

3.5.3 Impact of increasing the granular delay on the recognition accuracy

Now that the development of the model is complete, we started looking for new tracks to reduce the hardware resources needed to have a low-power on-chip implementation. As seen before, the architecture of CT FIR filters contains a principal delay line. This delay line is composed of a succession of delay taps. Each delay tap is equal to $62.5\mu s$ to be compatible with the 16kHz sampling frequency of the audio samples in the GSCD database. Moreover, inside each delay taps we find a succession of granular delays T_{GRAN}, as depicted in Fig. 3.23. This granular delay is defined in section 3.2.5. Based on this definition, the granular delay is set to be equal to $621.7ns$. Considering this value, each delay tap will contain approximately 50 granular delays. Consequently, reducing the number of granular delays creates an opportunity to reduce the hardware complexity and optimize power efficiency. However, it is not that simple to achieve this reduction. For instance, reducing the number of granular delays means increasing the granular delay value. However, it is shown in Fig. 3.23 that these granular delays operate in a way that for two pulses spaced by less than the granular delay, the second one will be ignored and not propagated through the delay chain. Hence, increasing this granular delay can distort the signal through the loss of several

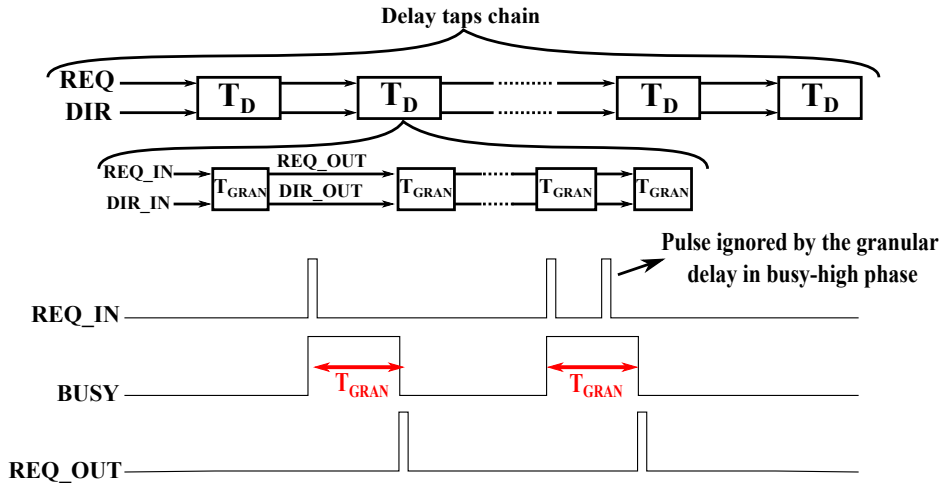
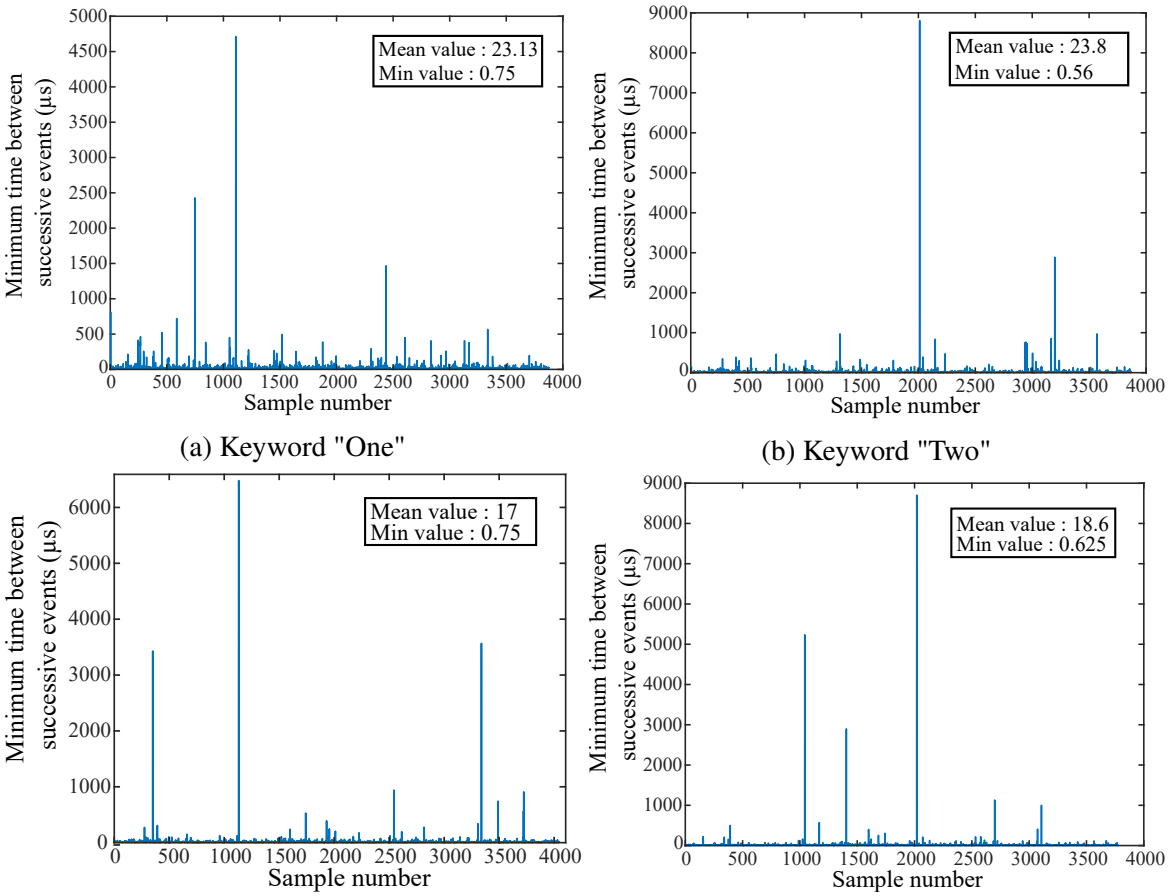


Figure 3.23 Delay line architecture and the operation of granular delays.

events. To evaluate the possibility of increasing the granular delay, we performed an analysis giving information about the minimal time between two successive events at the *Req* signal, generated from multiple audio samples for different keywords. The result of this analysis is illustrated in Fig. 3.24. The analysis was applied to all the target keywords, but here we display the result of only four keywords. From this result, we can see that the minimal time between two successive events is at minimum equal to $560ns$ with a mean value ranging from 17 to $23.8\mu s$. Thus, the mean value of this minimum time is way bigger than the theoretical granular delay. So, an increase in the granular delay can be conceivable.

Indeed, the increase in the granular delay can optimize the hardware implementation of the proposed system. However, the impact of this increase should not degrade the accuracy performance. For this reason, we have performed a study to analyze the trade-off between the granular delay and recognition accuracy. Fig. 3.25 shows the result of this latter study, where we can see that the accuracy decreases by increasing the granular delay. Thus, we have found an optimal functioning point where the granular delay is set to $1.25\mu s$ while maintaining an accuracy of 90.4% in the 10 keywords recognition case.



(a) Keyword "One"

(b) Keyword "Two"

(c) Keyword "Five"

(d) Keyword "Eight"

Figure 3.24 Minimum time between two successive pulses at the *REQ* signal, generated for multiple audio samples for four different keywords.

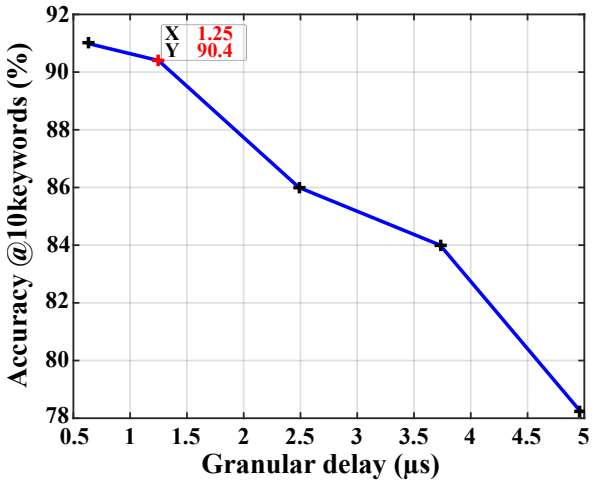


Figure 3.25 Delay line architecture and the operation of granular delays.

3.6 Conclusion

The proposed keyword spotting system targets the recognition of multiple spoken words. These systems are usually composed of a feature extraction unit to create a single signature allowing the differentiation between different audio classes. Then, the extracted features are injected into a classifier to generate the recognition results. The extracted features may be of multiple types such as MFCC coefficients, spectrograms, or other features. In our case, we extract the spectrograms, representing the spectral energy distribution.

To study the relevance of these spectrograms, a model of the keyword spotting system was created using *Matlab*®. This model has led to the definition of the main building blocks of the proposed system. Hence, the system is divided into hierarchical blocks, starting with a 6-bit LC-ADC to asynchronously decode the audio input into a pair of tokens, then these tokens are injected into a continuous-time FIR filter bank to perform the spectral decomposition of the audio content into 8 logarithmically distributed frequency bands, covering a bandwidth ranging from 840Hz up to 6250Hz. The filtering step was extensively optimized to adapt it for feasible hardware implementation. These optimizations simplified the architecture of the filters that were designed with a maximum order of 16. After the filtering step, the energy extraction process is performed in an event-driven way. Then, the per-band energies are truncated on 6 bits and concatenated to form the needed spectrograms. Finally, these spectrograms were fed into a 6-layer convolutional neural network to generate the recognition accuracy results. The parameters of each block were optimized independently to respect a common objective of getting the best tradeoff between hardware complexity and recognition accuracy. As a result, the system achieved the recognition of 10 keywords with an accuracy of 90.4%.

Finally, the modeling work presented in this chapter, was a mandatory step to specify all the needed design parameters to prepare for the hardware implementation. Moreover, since the system architecture is mainly digital, the simulation results of the model will represent the basis to start the design using hardware description languages. More precisely, the following step in this thesis will be to validate the hardware feasibility through the implementation and the simulation of the system on an FPGA validation platform.

Bibliography

[1] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," in IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 28, no. 4, pp. 357-366, August 1980, doi: 10.1109/TASSP.1980.1163420.

[2] I. López-Espejo, Z. -H. Tan, J. H. L. Hansen, and J. Jensen, "Deep Spoken Keyword Spotting: An Overview," in IEEE Access, vol. 10, pp. 4169-4199, 2022, doi: 10.1109/ACCESS.2021.3139508.

[3] I. López-Espejo, Z. -H. Tan and J. Jensen, "Exploring Filterbank Learning for Keyword Spotting," 2020 28th European Signal Processing Conference (EUSIPCO), Amsterdam, Netherlands, 2021, pp. 331-335, doi: 10.23919/Eusipco47968.2020.9287772.

[4] R. Prabhavalkar, R. Alvarez, C. Parada, P. Nakkiran, and T. N. Sainath, “Automatic gain control and multi-style training for robust small-footprint keyword spotting with deep neural networks,” in Proc. IEEE Int. Conf. Acoust., Speech Signal Process., Brisbane, MD, Australia, May 2015, pp. 4704–4708.

[5] R. Alvarez and H.-J. Park, “End-to-end streaming keyword spotting,” in Proc. IEEE Int. Conf. Acoustics, Speech Signal Process., Brighton, U.K., May 2019, pp. 6336–6340.

[6] A. Coucke, M. Chlieh, T. Gisselbrecht, D. Leroy, M. Poumeyrol, and T. Lavril, “Efficient keyword spotting using dilated convolutions and gating,” in Proc. IEEE Int. Conf. Acoust., Speech Signal Process., Brighton, U.K., May 2019, pp. 6351–6355.

[7] E. Shriberg, L. Ferrer, S. Kajarekar, A. Venkataraman, A. Stolcke, Modeling prosodic feature sequences for speaker recognition, Speech Communication, Volume 46, Issues 3–4, 2005, Pages 455-472,ISSN 0167-6393.

[8] L. Ferrer, N. Scheffer and E. Shriberg, "A comparison of approaches for modeling prosodic features in speaker recognition," 2010 IEEE International Conference on

Acoustics, Speech and Signal Processing, Dallas, TX, USA, 2010, pp. 4414-4417, doi: 10.1109/ICASSP.2010.5495632.

[9] K. M. H. Badami et al., ”A 90 nm CMOS, 6 µW Power-Proportional Acoustic Sensing Frontend for Voice Activity Detection,” in IEEE Journal of Solid-State Circuits, vol. 51, no. 1, pp. 291-302, Jan. 2016.

[10] S. Lecoq et al.,” Low-complexity feature extraction unit for “Wake-on Feature” speech processing,” 2018 25th IEEE ICECS, Bordeaux, 2018, pp. 677-680.

[11] Y. W. Li, K. L. Shepard, and Y. P. Tsividis, "Continuous-time digital signal processors," 11th IEEE International Symposium on Asynchronous Circuits and Systems, New York, NY, USA, 2005, pp. 138-143, doi: 10.1109/ASYNC.2005.15.

[12] B. Schell and Y. Tsividis, ”A Continuous-Time ADC/DSP/DAC System With No Clock and With Activity-Dependent Power Dissipation,” in IEEE Journal of Solid-State Circuits, vol. 43, no. 11, pp. 2472-2481, Nov. 2008.

[13] T. Marisa et al., ”Pseudo Asynchronous Level Crossing adc for ecg Signal Acquisition,” in IEEE Transactions on Biomedical Circuits and Systems, vol. 11, no. 2, pp. 267-278, April 2017.

[14] Y. Tsividis, "Event-Driven Data Acquisition and Digital Signal Processing—A Tutorial," in IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 57, no. 8, pp. 577-581, Aug. 2010, doi: 10.1109/TCSII.2010.2056012.

[15] Y. Tsividis, "Continuous-time digital signal processing," Electronics Letters 39(21), 1551 (2003).

[16] Y. Tsividis, "Digital signal processing in continuous time: a possibility for avoiding aliasing and reducing quantization error," 2004 IEEE International Conference on Acoustics, Speech, and Signal Processing, Montreal, QC, Canada, 2004, pp. ii-589, doi: 10.1109/ICASSP.2004.1326326.

[17] T. Marisa et al., "Pseudo Asynchronous Level Crossing adc for ecg Signal Acquisition," in IEEE Transactions on Biomedical Circuits and Systems, vol. 11, no. 2, pp. 267-278, April 2017, doi: 10.1109/TBCAS.2016.2619858.

[18] Joseph E.Hawkins.”Human ear” Encyclopædia Britannica. October 25, 2018. https://www.Britannica.com/science/ear.

[19] K. L. Anderson, “Speech Spectrum Audiogram for Consonants. Based on the work of Ross, M., Brackett, D., Maxon, A.B (1991). Assessment and management of mainstreamed hearing-impaired children: Principles and practices.” 2012

[20] Donald D. Greenwood (1990). "A cochlear frequency-position function for several species – 29 years later". Journal of the Acoustical Society of America. 87 (6): 2592–2605. Bibcode:1990ASAJ...87.2592G. doi:10.1121/1.399052. PMID 2373794.

[21] K. M. H. Badami et al., "A 90 nm CMOS, 6 μW Power-Proportional Acoustic Sensing Frontend for Voice Activity Detection," in IEEE Journal of Solid-State Circuits, vol. 51, no. 1, pp. 291-302, Jan. 2016.

[22] J. S. P. Giraldo, S. Lauwereins, K. Badami and M. Verhelst, "Vocell: A 65-nm Speech-Triggered Wake-Up SoC for 10-μW Keyword Spotting and Speaker Verification," in IEEE Journal of Solid-State Circuits, vol. 55, no. 4, pp. 868-878, April 2020, doi: 10.1109/JSSC.2020.2968800.

[23] V. Risojević, A. Avramović, Z. Babić and P. Bulić, "A simple pipelined squaring circuit for DSP," 2011 IEEE 29th International Conference on Computer Design (ICCD), Amherst, MA, USA, 2011, pp. 162-167, doi: 10.1109/ICCD.2011.6081392.

[24] S. Zheng et al., "An Ultra-Low Power Binarized Convolutional Neural Network-Based Speech Recognition Processor With On-Chip Self-Learning," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 66, no. 12, pp. 4648-4661, Dec. 2019, doi: 10.1109/TCSI.2019.2942092.

Chapter 4

Concept validation using FPGA

4.1 Introduction

The modeling step detailed, in the chapter before, allowed the extraction of the different key parameters needed to start the hardware implementation. However, before beginning the design in an advanced CMOS technology environment, we should verify the system compatibility with hardware design tools. Up to now, the system was ideally modeled using *Matlab*®, without taking into account any hardware imperfections such as noise mismatch... Hence, to fall into the non-ideal hardware world, the system will be designed using a Hardware Description Language (HDL) to construct an abstract Register-Transfer Level (RTL) model.

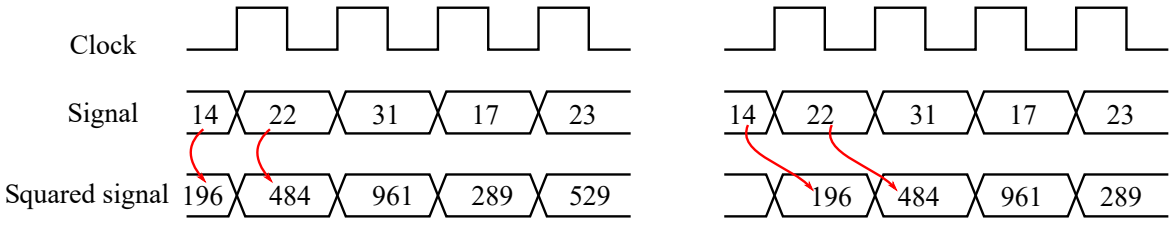
The RTL design will model the data flow from the input to the output passing by all the necessary registers. Besides, the control flow with all the synchronization signals will also be modeled in this step. Moreover, the RTL model will give a high-level view of the circuit implementation using the different logic basic operations. Using this preliminary implementation, we can conclude on the quantity of needed hardware resources.

Even though the RTL model describes a preliminary circuit-level architecture, it is still an ideal model that does not take into account some advanced non-idealities such as timing problems. Thus, the RTL model is not sufficient to completely validate the hardware implementation. For this reason, another validation step is needed in the roadmap of the thesis. It consists of synthesizing the designed RTL model on a Field Programmable Gate Arrays (FPGA) platform. This validation step permits the mapping of the RTL model on a set of logic gates and registers to form the corresponding circuit architecture. By doing so, the system operation will be demonstrated in the real world by considering all the possible imperfections.

This chapter will describe in more detail the RTL design of the keyword spotting system as well as the hardware implementation on an *Intel*® FPGA platform. In addition, the system will be tested on the GSCD dataset to extract the performance results.

4.2 Differences between *Matlab*® and RTL model

The software model developed using *Matlab*® was instantaneous, which means that the logic operations applied on a certain parameter consider the actual value of the parameter. However, in the RTL model, the specified logic instructions are not executed in the same way. In more detail, for synchronous systems, the applied logic instructions on a certain parameter will consider its last value set before the arrival of the synchronizing event, which is in most cases a rising or falling edge of the synchronization signal. An example of a synchronized squared signal for the two models is shown in Fig. 4.1. It can be seen that the result of the squaring function is delayed by one clock cycle in the RTL model compared to the instantaneous processing in the *Matlab*® model.



(a) Processing in the Matlab model (b) Processing in the RTL model

Figure 4.1 RTL vs *Matlab*® model.

Furthermore, the processing in the *Matlab*® model can be done in parallel, which means that we can generate multiple result values simultaneously even if in reality they are successive in time. However, this is impossible to have in the RTL model. Hence, if it is needed to perform a logic operation on multiple signal values delayed in time, then these signal values should all be stored until finishing the application of the required logic operation. In more detail, the need for storage is not required for *Matlab*® because every value is automatically stored in the allocated memory used by the software. But, in the RTL case, since it is a hardware description language, we need to specify and define every use of the storage unit if needed.

4.3 Clock-gated delays based keyword spotting architecture

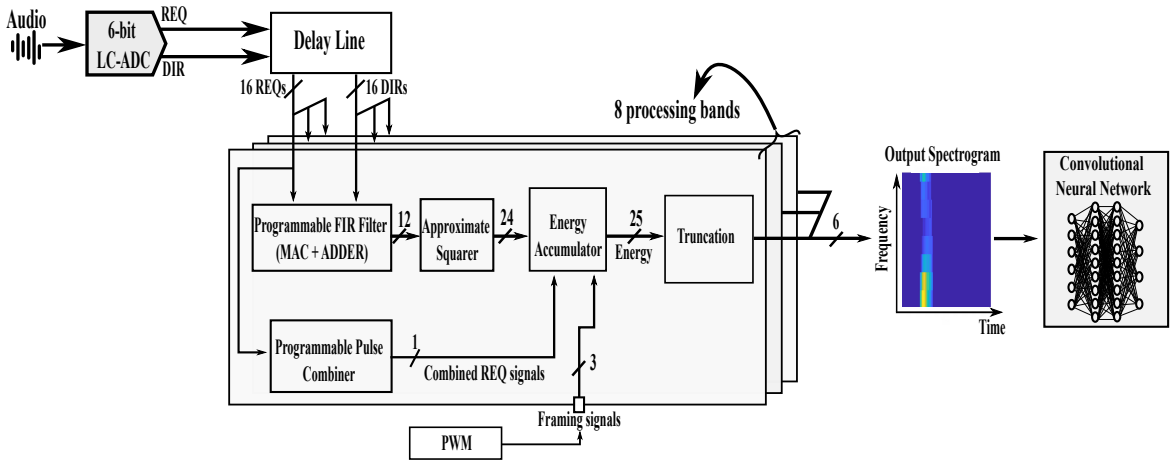


Figure 4.2 Architecture of the keyword spotting system.

As a reminder, Fig. 4.2 depicts the architecture of the proposed keyword spotting system, where the audio signals are clocklessly quantized using a 6-bit LC-ADC to generate the (REQ, DIR) tokens indicating the occurrence of the event and its direction. Afterwards, the tokens are injected into a 16-tap delay line, then all the delayed REQ and DIR signals are simultaneously fed into 8 parallel processing chains. In each chain, the signal undergoes at the first stage a filtering operation in continuous time through a 16-coefficient digital FIR filter to keep only the spectral content in the band specified by the coefficient values. The next processing step is to compute the per-band energy. This is done following two steps. Firstly, an approximate squarer computes the instantaneous power value of the filter's output each time a pulse is generated by the programmable REQ combiner, which represents the combination of all the delayed REQ signals. Then, an energy accumulator accumulates the successive squared values to generate an energy value during each frame. The energy values are coded on 25 bits, but the relevant information can be coded on a lower resolution. Thus, the energy values are truncated on only 6 bits after finding the global maximum value across the whole spectrogram. Finally, this spectrogram (matrix of truncated energy values) is transmitted to the CNN for generating the classification results.

4.3.1 FIR filter

As mentioned before, the FIR filter bank contains 8 bandpass filters designed to cover a frequency range starting from 840Hz up to 6.25kHz with an out-of-band attenuation of

5dB. The architecture of these filters is depicted in Fig. 4.3. It is composed of a delay line responsible for transmitting the delayed tokens (REQ, DIR) to the set of MAC blocks. Then, the outputs of the different MACs are added to generate the filter output. As mentioned in the chapter before, the filters are designed with an order ranging from 4 to 16 at maximum, for keyword spotting application. But, to maintain a generic architecture of these filters to be re-adapted to other applications, we decided to design all 8 filters with 16 MAC units and activate only the MACs designated by the corresponding filter order.

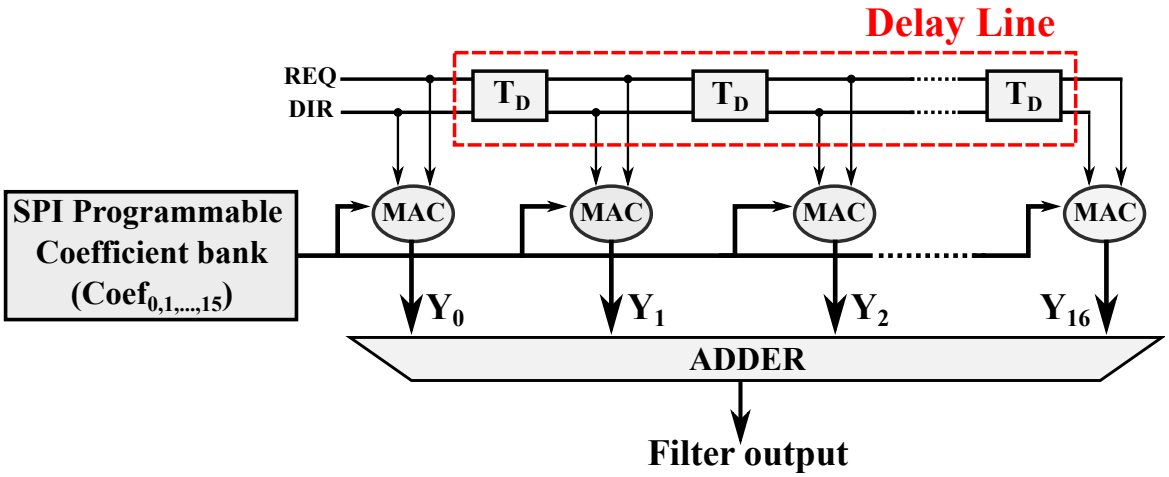
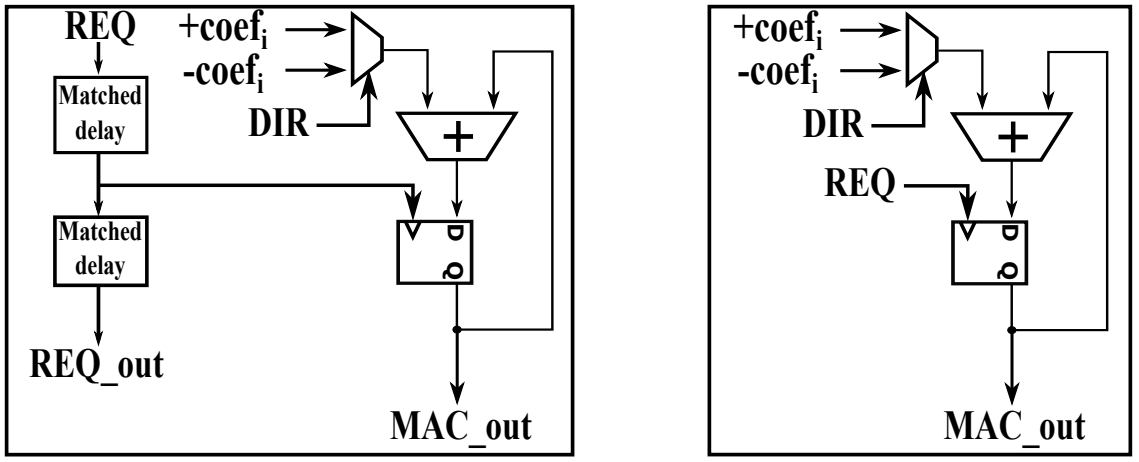


Figure 4.3 FIR filter architecture.

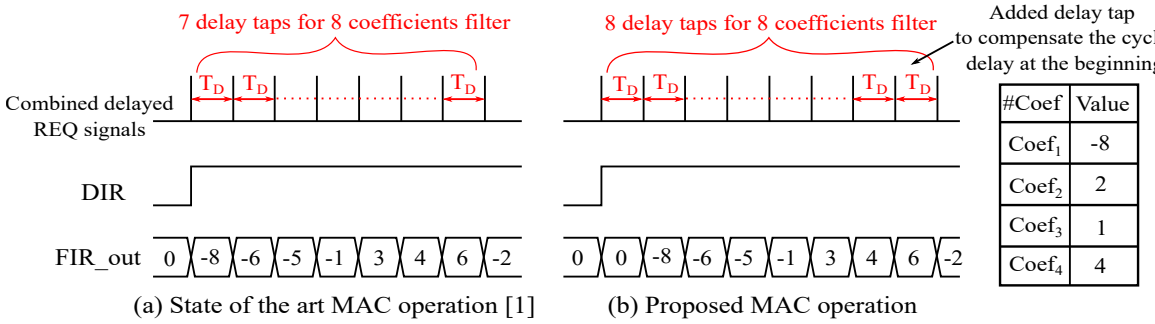
The MAC architecture in the CT FIR filter replaces the multiplication by an "addition + accumulation" process. Indeed, the output of the MAC is specified by the equation 3.1.



(a) MAC architecture in CT DSP systems [1] (b) Proposed MAC architecture

Figure 4.4 MAC architectures. (a) MAC architecture with matched delay for asynchronous continuous-time processing. (b) Proposed MAC architecture with one cycle delay processing.

The MAC architecture used for CT DSP systems [1] is shown in Fig. 4.4 (a). The specific point about this architecture is that the REQ signal, acting as a clock, is propagated through a series of matched delays to have an asynchronous pseudo-instantaneous operation. In more details, these delays match the propagation delay of the adder and the D flip-flop (DFF), so that the REQ signal rising or falling edges arrive at the same time or later than the arrival of the data at the input of the adder or the DFF. In such a way, the processing will be done on the actual value not the value of the cycle before. However, the complexity of this architecture is that it should designed in a full custom way due to the non-advancement of the asynchronous semi-custom synthesis flows. In the counterpart, we propose a MAC architecture, depicted in Fig. 4.4 (b), with the same operation except for the use of matched delays. This architecture will allow the use of the classic semi-custom design flows with significant modifications that will be discussed later. The FIR filter output is generated as the sum of the output signal of all the MAC units. The flow generating this FIR output is depicted in Fig. 4.5. It can be seen that when using the CT DSP MAC [1], the response of the MAC is instantaneous, and at each REQ event, we collect the actual MAC value, and then the FIR filter is updated instantaneously. In the other case, the response of the MAC is delayed until the arrival of the next REQ event. Thus, the FIR output is always delayed by one cycle and we cannot collect the real instantaneous value. This is the reason why we added an extra delay cell at the end to compensate this cycle delay and collect the actual FIR output value. In Fig. 4.5, we considered an example of a classic FIR filter with 8 coefficients and 7 delay taps. However, in our case, we used 8 delay taps for the same filter to compensate the starting cycle delay. This justifies the choice of using 16 delay taps in our filters for keyword spotting application, knowing that we have only 16 coefficients at maximum.



(a) State of the art MAC operation [1] (b) Proposed MAC operation

Figure 4.5 Operation flow for the two MAC architectures.

4.3.1.1 Delay line

The delay line is formed by a sequence of 16 delay taps of 62.5μs to cover the 8kHz signal bandwidth needed to process speech. Fig. 4.6 shows that each delay tap contains a sequence of granular delays T_{GRAN}, defined in Chapter 3. The operation of granular delays was shown in Chapter 2 and is reminded in Fig. 4.6.

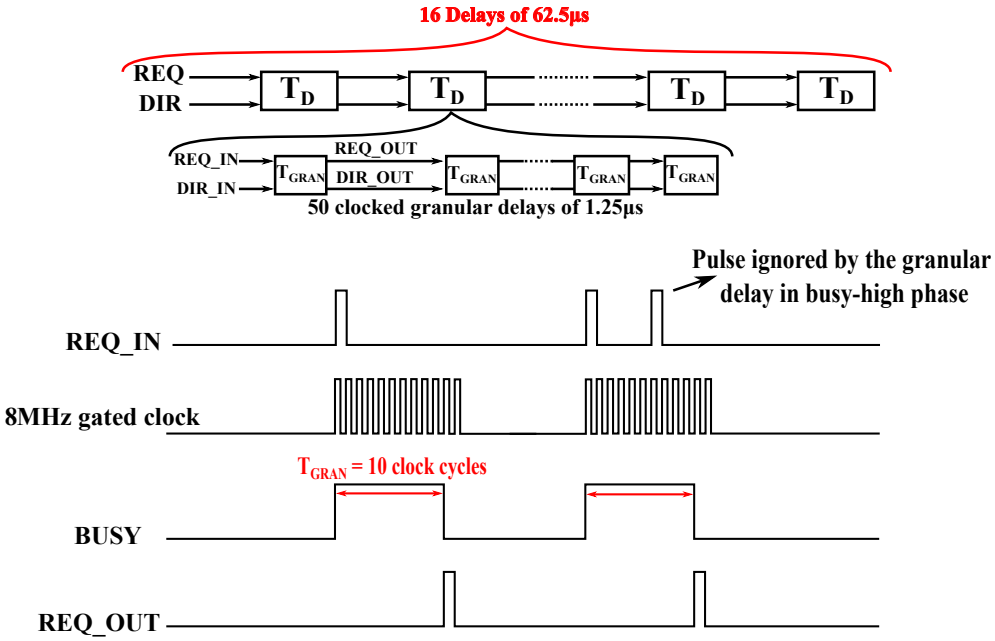


Figure 4.6 Delay line architecture and operation.

To maintain a completely digital architecture, we decided to design these granular delays as clock-gated counters to keep an event-driven functioning. The operation flow of these granular is illustrated in Fig. 4.6. The arrival of a pulse at the REQ_IN triggers an 8-MHZ system clock, equivalent to an 8-MHz oversampling rate of the input audio signals. Then, the granular counter enters a BUSY state and starts counting 10 clock cycles to match a granular delay of 1.25μs. After that, a pulse is generated at the REQ_OUT signal.

4.3.2 Multi-frame energy accumulator

After the filtering, the following step is to calculate the energy in each frequency band. As mentioned in the previous chapter, the energy is calculated during multiple frames until covering the speech duration. More specifically, in standard speech processing the energy is calculated during successive frames of 25ms, with 10ms overlap each time [2].

The challenge here remains in the method to generate these frames. In the case of non-overlap, only one periodic framing signal can be used to control the energy computation. But,

since we are concerned by 10ms overlap each time, this solution is not sufficient. To cover this overlap, we need more than one framing signal. The number of these signals will be decided by the ratio between the frame and the overlap durations, rounded to the maximum, i.e. three in our case. Each one is a 30ns periodic signal with a duty cycle of 83.33% to generate the 25ms frames. Then, the 3 framing signals are delayed by 10ms each. These signals are depicted in Fig. 4.7.

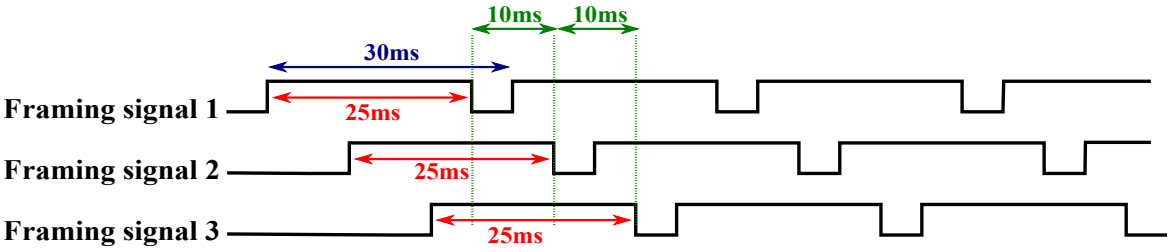


Figure 4.7 Framing signals.

These framing signals can be either externally injected or internally generated. In order to maintain a generic architecture, we opted for the second option. In more details, these signals will be generated using an integrated Pulse Width Modulator (PWM). The PWM will receive at its input a 1kHz clock signal and output the three specified framing signals. Since the framing signals will be the same for all the processing bands, there will be only one PWM module shared between all 8 processing bands.

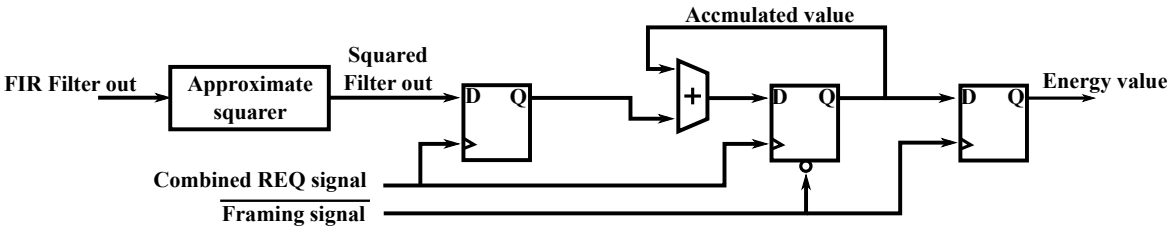


Figure 4.8 Architecture of the energy accumulator.

After generating the framing signals, the circuit architecture of the energy accumulator, whose flow was illustrated in Fig. 3.15, is depicted in Fig. 4.8. During each frame, the accumulated value is updated with a squared filter output value, generated using the approximate squarer described in the previous chapter, each time we have a pulse at the Combined REQ signal. Then, at the end of the frame, the accumulator is reset to 0 and the corresponding energy value is collected. Moreover, due to the overlap, we cannot multiplex the framing signals and use only one energy accumulator. Hence, each processing band will be equipped with a multi-frame energy accumulator composed of three energy accumulators shown in Fig. 4.8. Moreover, these three energy values should be concatenated together to

form one global energy quantity. This is done through a multiplexing process, shown in Fig. 4.9. Besides generating the framing signals, we create a frame index that indicates the actual last recent frame rose to a high-level state. By doing so, we collect the last energy value controlled by the framing signal corresponding to the frame index value.

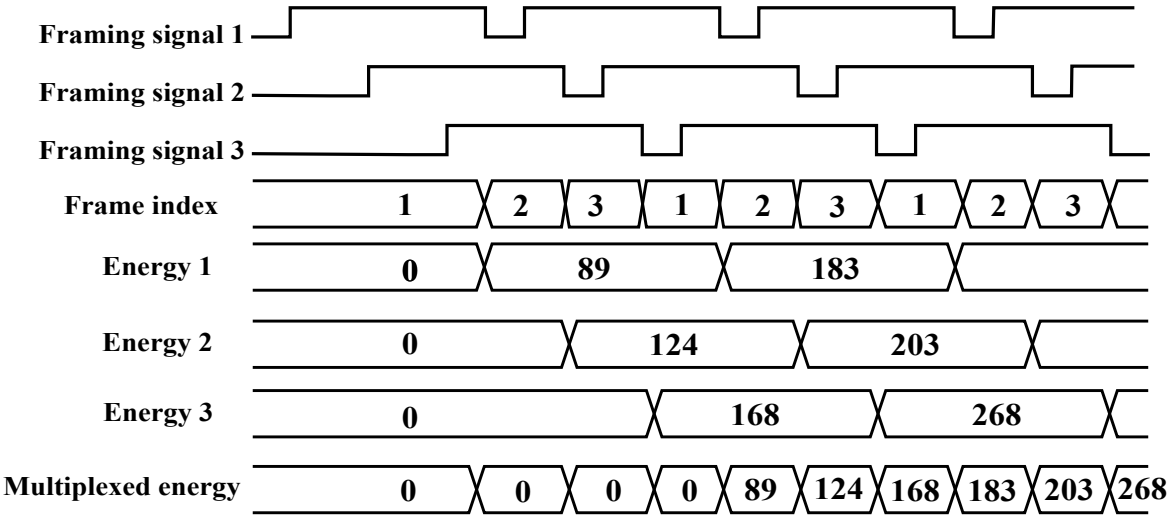


Figure 4.9 Energy multiplexing flow.

4.3.3 Storage and Truncation

The last step before outputting the final spectrogram is truncation. As described before, the energy is coded on 25 bits, but as seen in the previous chapter the audio dynamic range of interest can be coded on only 6 bits. The truncation will not be automatically done starting from the same bit position each time, but for each keyword, we need to identify the global maximum energy value across the whole spectrogram. Then, all the energy values will be truncated starting from the leading one position in the identified maximum. In each processing band, the energy values are calculated successively and at the end of each frame, the latest energy value is lost and get replaced by the new one. Thus, if we need to identify the global maximum value across multiple successive energy values, we need to store them. Hence, the complete spectrogram needs to be stored until finding the global maximum and getting truncated, then gets updated with new energy values for a new truncation process.

The audio samples, used for the test, are taken from the GSCD database [3] and contain one second of speech, with different pronunciations of the keywords in different backgrounds. considering the audio sample duration of 1s and the frame duration of 25ms with 10ms overlap, the audio sample will be covered by a total of 100 frames. This is the number of successive energy values to be stored for each keyword.

4.3.3.1 Register-matrix based self-truncation memory

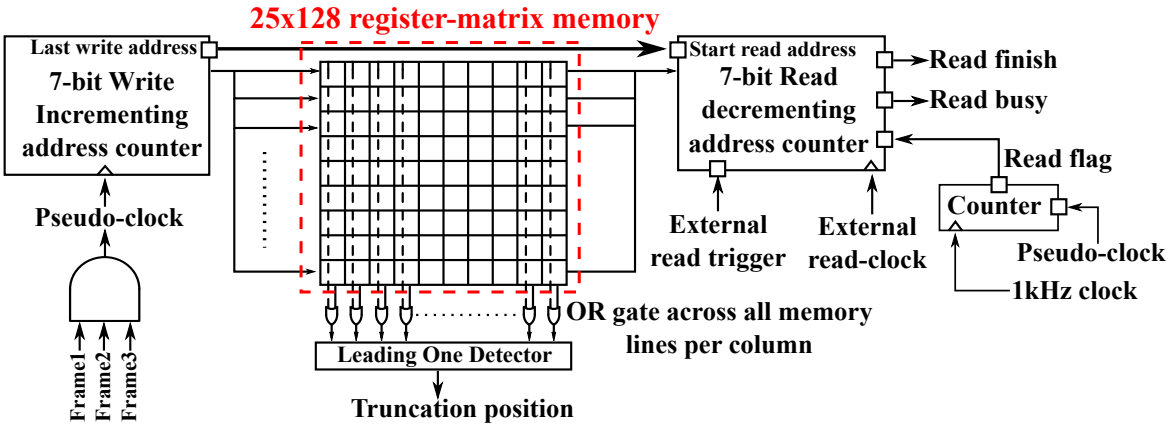


Figure 4.10 Architecture of the self truncation memory.

The block responsible for storing and truncating the energy values is displayed in Fig. 4.10, and will be called from now on a self-truncation memory. This block is composed of a register matrix able to store up to 128 energy values of 25bits, to store the full spectrogram corresponding to the 1s speech keywords. The size of the memory was not cut at 100 for the reason of maintaining a generic architecture. For the truncation part, an OR gate network is integrated into the register matrix to have in-memory computing for better power efficiency. In more details, an OR operation is performed across all memory lines for each column to search for a 1 value. Then, the result of this operation will be a 25-bit word indicating all the positions in the whole memory containing a 1 value. In addition, this block is integrated into each processing band. Hence, to perform a truncation across the whole spectrogram, the 25-bit output of the OR network in all 8 processing bands will be passed through another OR network, and then the result will be fed into a global Leading One Detector (LOD) responsible for returning the highest position of a 1 value. This position presents the leading one position across the whole spectrogram, which will indicate the truncation starting position.

4.3.3.2 Reading and writing processes

After setting the design specifications of the memory, we need to define how the writing and reading processes are managed. The writing process should be synchronized with the energy values generation rate. Since the energies are generated every 10ms, then the writing process needs to have the same period. The signal synchronizing the writing can either be brought from an external source or created internally. To maintain a programmable architecture, this signal will be internally generated. Fig. 4.10 shows the 7-bit write counter

used to increment the 128 writing addresses. This counter is clocked by a pseudo-clock signal that is created as a combination of the three framing signals. This pseudo-clock is shown in Fig. 4.11. It is generated in such a way that every time we have a rising or falling edge in a framing signal, we have the same edge happening at the pseudo-clock, except for the first frame. By doing so, we end up generating a pseudo-clock signal with a period of 10ms, starting from the second rising edge.

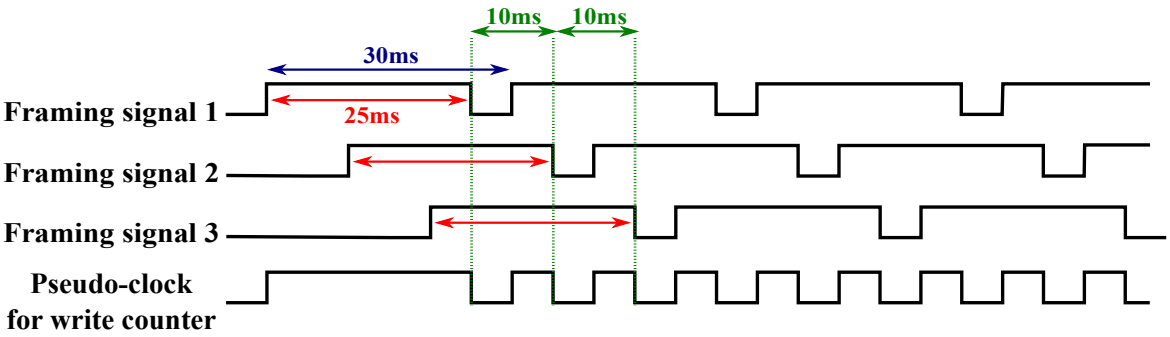


Figure 4.11 pseudo clock used to control the writing process.

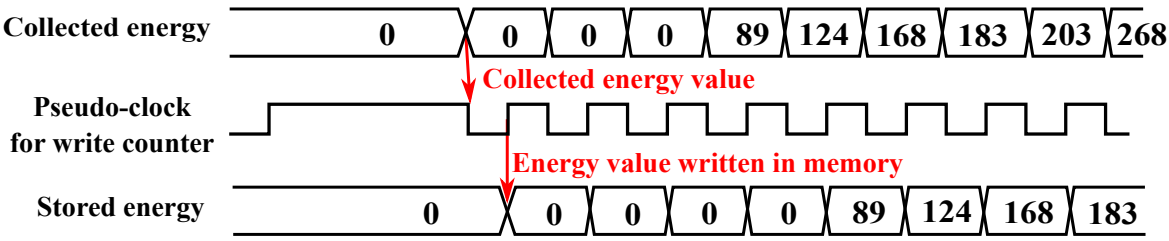


Figure 4.12 Energy writing process in the register-matrix memory.

Besides, it is shown in Fig. 4.12 that the energy values are collected at the falling edge of the pseudo-clock, but the writing in the register-matrix is done at the rising edge. This is done to avoid setup problems because the data is going to be ready at least one half cycle before the arrival of the rising edge.

The reading process is less straightforward because it should be managed for the 8 processing bands at the same time, compared to the writing that is handled independently in each processing band. More specifically, when the reading process is triggered, it will concern the complete spectrogram, not only one band. Due to this fact, we need to set carefully the reading clock frequency. As mentioned before, the writing is done with a frequency equal to: $F_{Writing} = \frac{1}{T_{Writing}} = \frac{1}{10ms} = 100Hz$. Then, we need to read the full spectrogram before the writing of a new energy value, to avoid compromising the spectrogram integrity. This means

reading successively the 128 memory lines in all the processing bands during less than one writing cycle. Hence, reading frequency should respect the following condition:

$$F_{Reading} > 8 \times 128 \times F_{Writing} => F_{Reading} > 102.4kHz \tag{4.1}$$

Another important aspect when reading the memory is to describe the multiple signals synchronizing this process. The reading will be tracked by a control signal called the read flag. This flag presents the acceptable period of time for read operation without disturbing write operation. It is generated by a counter in such a way that it is long enough to read all the register matrices in all eight bands. The read flag generation flow is depicted in Fig. 4.13. During each writing pseudo clock cycle, the counter is reset to 0 and starts incrementing, then the read flag rises to a high state when the counter value is between 1 and 8, then the read flag falls to low state. As mentioned before, a new energy value is written each time we have a rising edge of the pseudo clock. This is the reason for waiting for two cycles before raising the read flag to avoid any hold problems concerning the writing in memory. The reading process needs to start and finish while the read flag is at a high level, to ensure that the full spectrogram has been read before the writing of a new energy value.

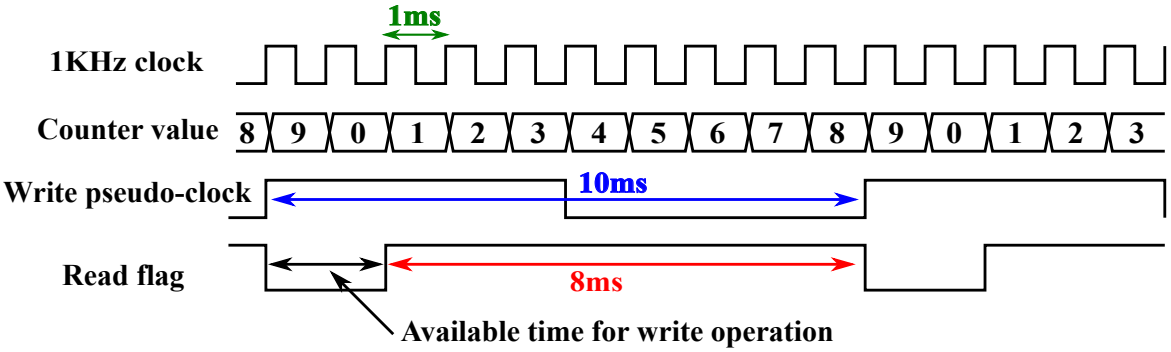


Figure 4.13 Read flag generation process.

As seen in Fig. 4.10, the reading counter receives an external read trigger signal. This signal is responsible for setting off the reading process. The arrival of a pulse in this signal will trigger the counter to start decrementing the reading addresses. However, this trigger should be synchronized with the following rising edge of the read flag to not influence the write operation. This synchronization process is illustrated in Fig. 4.14. In addition, after the synchronization of the read trigger, it is shown in Fig. 4.10 that the reading counter retrieves the last writing address from the writing counter. After that, the reading counter starts decrementing the memory addresses from the last writing address, to read the requested number of stored energy values from the last to the least recent one. Moreover, since the

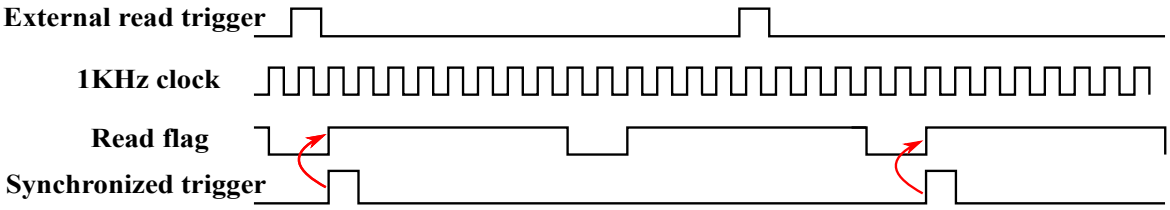


Figure 4.14 Read trigger synchronization on the read flag.

reading of all the memory has to be done during the read flag high state, then the frequency of the reading should be higher because the high-state phase last only 8ms. Hence, the new condition on the reading frequency is :

$$F_{Reading} > 8 \times 128 \times \frac{1}{T_{Read\,flag}} => F_{Reading} > 128kHz \tag{4.2}$$

with $T_{Read\,flag}$ the high state duration of the read flag signal.

The last point to discuss is the method to manage the read trigger synchronization in all the processing bands. For test matters, it was decided to read the register matrices successively from one processing band to the next one. The reading synchronization process is shown in Fig. 4.15. The external read trigger is input only to the first processing band, then we start reading the memory content in this band, and after finishing a read finish pulse is generated. This read finish pulse acts as a read trigger for the next processing band. This process continues until finishing the reading in all the processing bands. Moreover, we added 8 read busy signals, each of them rising to a high state when the corresponding processing band memory is being read. This method helps us track the reading process and identify which processing band content is being read at each time.

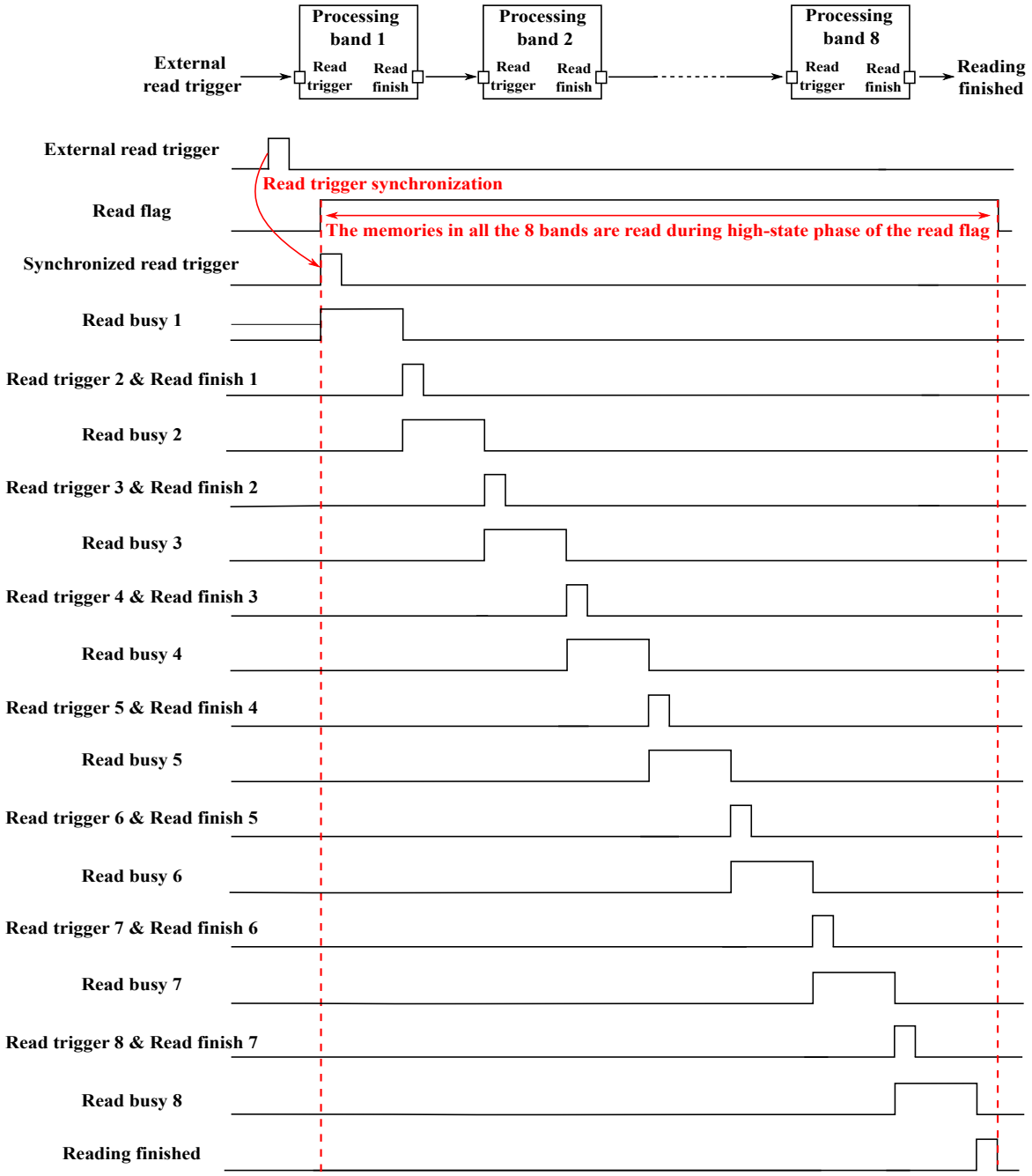


Figure 4.15 Reading synchronization process of the memories in the 8 processing bands.

4.4 FPGA hardware implementation

Now, after completing the RTL description, the system will be decomposed into multiple sub-blocks as shown in Fig. 4.16. The RTL model allows us to simulate the hardware described system using CAD tools such as *Cadence*® Xcelium logic simulator. However,

these simulations are still ideal and do not take into account multiple hardware imperfections like mismatch, timing setup, and hold problems... Hence, we cannot consider the RTL simulations as the last validation point before starting ASIC hardware implementation in an advanced CMOS technology. The RTL system will be thus mapped into an FPGA validation platform to ensure the hardware operation of our system and extract the first performance of the hardware-designed system.

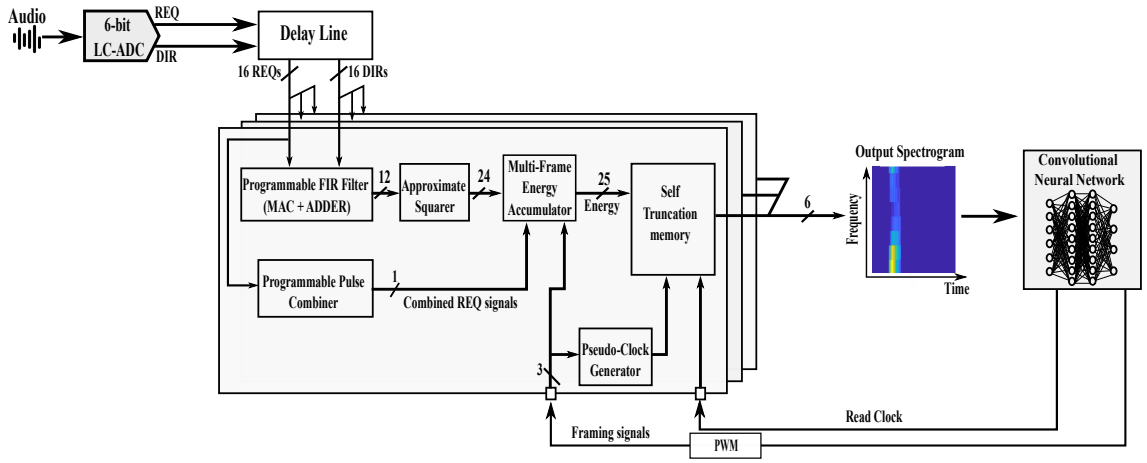


Figure 4.16 RTL Architecture block diagram.

The RTL described system does not concern the complete system but only the feature extraction part, without including the LC-ADC. Thus, for test matters, this hardware-described feature extractor will be combined with a behavioral model of the LC-ADC to generate the REQ and DIR input tokens and a software-defined convolutional neural network to generate the classification results. Technically speaking, the feature extractor will be programmed into an *Intel*® DE2-115 Cyclone IV FPGA board, depicted in Fig. 4.17.

4.4.1 Synthesis constraints

The advantage of FPGA programming is the ability to have generic synthesis constraints that will be adapted to the circuit mapped into it. For this reason, specifying synthesis constraints when mapping the design on FPGA is not critical and can be bypassed. However, for more controllability, we decided to add some constraints, especially on the clock signals, so that the FPGA synthesis software can define the critical paths depending on the clock frequencies. The different clock signals with their functions and frequencies are depicted in Table 4.1. The rest of the signals will not be constrained and we will let the *Intel*® *Quartus*® automatic compiler and router set and apply the default constraints.

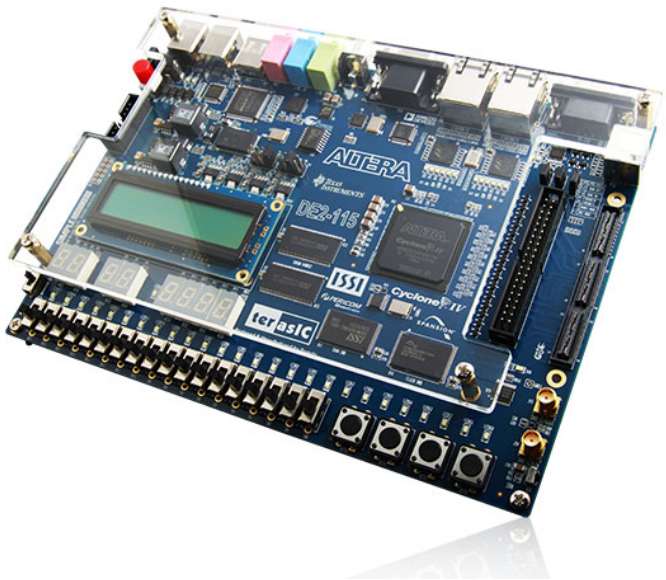


Figure 4.17 *Intel*® DE2-115 Cyclone IV FPGA validation board from terasic.

Table 4.1 Clock signals constraints.

Clock name	Function	Frequency
Delay clk	Synchronize all REQ and Combined REQ signals	8MHz
Frame clk	Base clock to generate framing signals	1kHz
Read clk	Clock to read memory matrices	130kHz

4.4.2 Hardware resources

The feature extraction unit is mapped into the FPGA platform. Then, to have an idea about the hardware complexity of our circuit, we decided to generate a preliminary report counting the total number of logic gates. This report, shown in Table 4.2, shows that our circuit is represented by a total of 174042 logic elements with a repartition of 80.4% combinatory cells and 19.6% registers. The high number of registers is due to the use of register-matrix memories (3.24 kbit/band) in each processing band.

Table 4.2 Total number and repartition of hardware logic elements.

FPGA Ressources	
Family	Cyclone IV
#Logic elements	174042 (100%)
#Combinatory cells	139907 (80.4%)
#Registers	34135 (19.6%)

4.5 Test and measurement results

After programming the FPGA platform with our RTL design, the next step is to test the operation of the system. To do so, the implemented feature extractor will be tested on the audio samples from the Google speech command dataset, to evaluate its capacity to generate spectrograms and test their relevancy through the recognition accuracy results.

4.5.1 Test setup

For test matters, the FPGA-implemented feature extractor is combined with multiple test devices to control the different inputs, and then visualize and analyze the outputs. The DE2-115 *Intel*® Cyclone IV FPGA has 36 General Purpose Inputs Outputs (GPIO). The RTL design IOs will be mapped to these GPIOs using the *Quartus*® software. After programming the design into the FPGA, the physical connections to the design will be done through the GPIOs header shown in Fig. 4.18. Our design contains only 25 preliminary IOs to analyze its operation. The RTL design IOs are connected to the GPIOs header following the order depicted in Fig. 4.18. Besides the previously described clock signals, the meaning and function of each IO are described in Table 4.3. Finally, the design GPIOs need to be externally connected either to inject an input or visualize an output. For this reason, the setup used to test the feature extraction unit is schematized in Fig. 4.19. It is composed of an arbitrary wave generator (AWG) that acts as an interface between *Matlab*® LC-ADC behavioral model and the hardware implementation. The role of the AWG is to transmit the generated tokens REQ and DIR using the LC-ADC model, to the corresponding input pins of the designed feature extraction circuit. Moreover, a frequency generator is used to deliver all the needed clock signals.

For configuration matters, an SPI interface is then used to program the coefficients, the number of lines to be read in the memories in each processing band, and read the complete

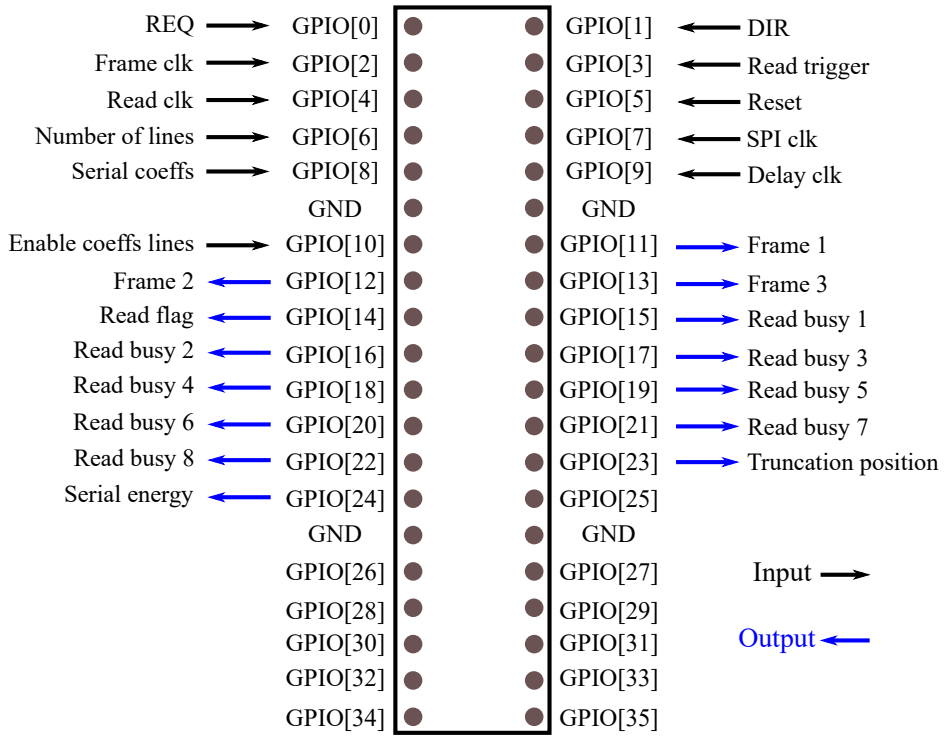


Figure 4.18 GPIOs mapping into the Cyclone IV FPGA platform.

spectrogram. In addition, a logic analyzer will be used to visualize the different outputs and inputs to be able to debug in case of the appearance of an abnormality in the feature extractor operation.

4.5.2 Test results

4.5.2.1 Spectrograms generation

Using the test setup previously described, the FPGA-implemented feature extractor will be tested on audio samples from the GSCD database. The test will concern the ability of the designed system to construct the audio spectrograms, that will be injected into a software-defined CNN to test their relevance to differentiate between the different audio classes through studying the recognition accuracy results. Moreover, the *Matlab*® model will be considered as an ideal reference. Hence, the spectrograms generated using the FPGA implementation will be compared to the ones issued from the *Matlab*® simulations. Fig. 4.20 depicts an example of spectrograms generated using the *Matlab*® model and the hardware implementation, for keywords "One" and "Two". The visual inspection of the spectrograms, shown in Fig. 4.20 ((a), (b) ; (d), (e)), leads to the deduction that the two implementations

Table 4.3 Type and function of each IO in the feature extraction hardware design.

IO	Type	Width	Function
REQ	Input	1 bit	Request signal generated using LC-ADC
DIR	Input	1 bit	Direction signal generated using LC-ADC
Read trigger	Input	1 bit	Signal to trigger the memory reading process
Reset	Input	1 bit	Global inverted reset
Number of lines	Input	7 bits	Number of memory lines to be read
SPI clk	Input	1 bit	Clock to serialize the SPI interface to program the coefficients, number of lines and to serialize the energy values.
Serial coeffs	Input	1 bit	Signal of all the coefficients in series to program the coefficient bank
Enable coeffs lines	Input	1 bit	Enable signal to start programming the coefficient bank and setting the number of lines
Frame k	Output	1 bit	K^{th} framing signal
Read flag	Output	1 bit	Signal synchronizing the reading in all the processing bands
Read busy k	Output	1 bit	Signal at high level during memory reading in the kth processing band
Truncation position	Output	5 bits	Truncation position output bitwidth
Serial energy	Output	1 bit	Serialized energy values stored in the memory

generate perfectly similar spectrograms. However, the visual comparison is not sufficient to conclude about the spectrograms' similarity. For this reason, the visual inspection is complemented with a quantitative analysis to calculate the relative error between these spectrograms. To emphasize the differences, the spectrograms (c) and (f) use a 10 times finer scale. The relative error in the case of these two keyword examples is equal to 0.78% and 0.93%, for "One" and "Two" respectively. In addition, to have a more robust study, this simulation was performed on 100 randomly chosen keywords with 10 keywords from each number (zero to nine), taken from the GSCD database. The result of this study is sketched in the graph 4.21. The FPGA implementation demonstrated the capacity to generate similar spectrograms to the *Matlab*® reference model with an average relative error of 0.88%.

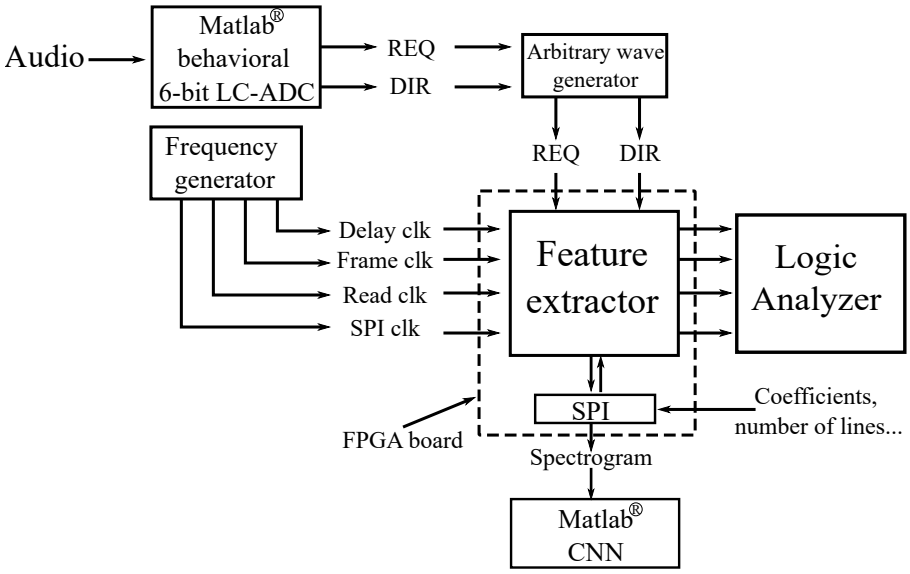


Figure 4.19 Test setup.

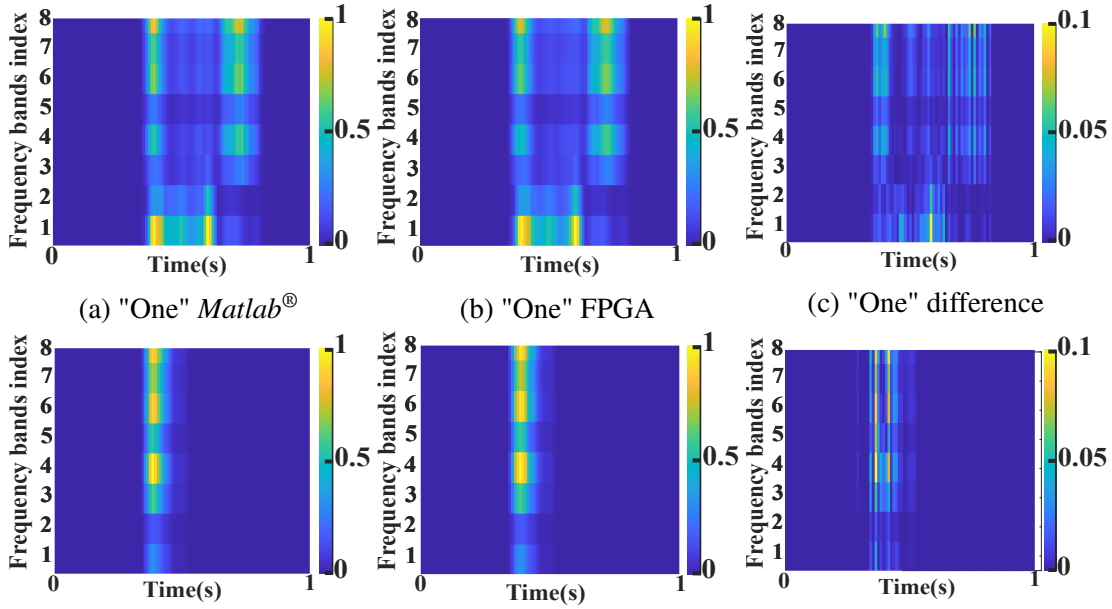


Figure 4.20 Normalized spectrograms generated using *Matlab*® and FPGA implementations for "One" and "Two" keywords, respectively ((a), (b) ; (d), (e)). Spectrograms presenting the calculated difference ((c), (f)).

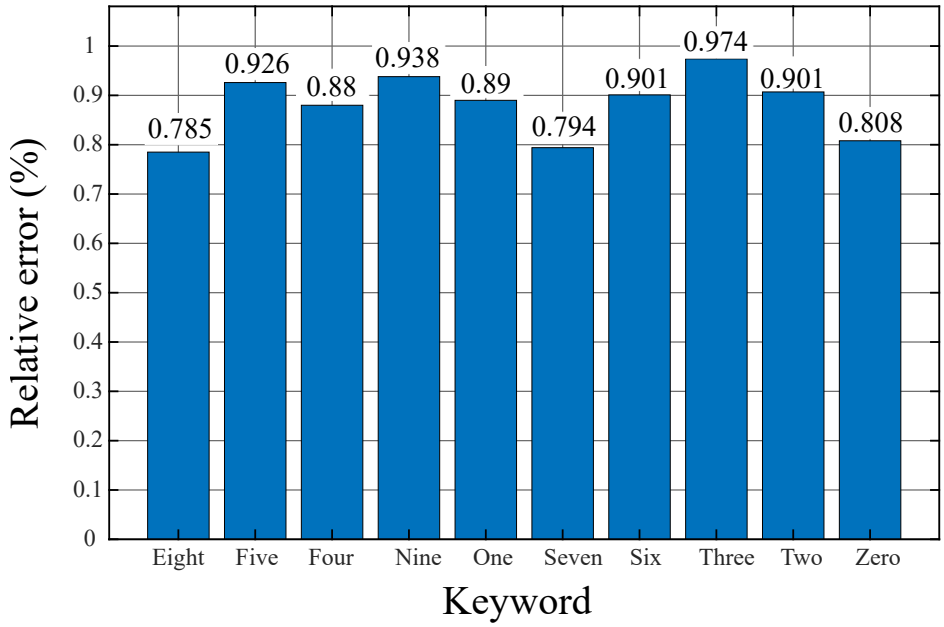


Figure 4.21 Average relative error in energy values for 10 keywords from each number.

4.5.2.2 Classification results

To evaluate the system recognition accuracy, a thorough classification test was performed on 11 audio classes (10 keywords + background noise) from the GSCD database. The keywords considered for the test are the numbers from "zero" to "nine". The altered spectrograms generated using the FPGA implementation have been input into the *Matlab*® CNN model for learning and inference steps. As a result, the FPGA-implemented system demonstrated the recognition of the specified audio classes with an accuracy of 90.4%. Hence, the recognition accuracy is not impacted by the slight relative error induced when passing from the *Matlab*® to the FPGA hardware implementation.

4.6 Synthesis in the 65nm CMOS process

The FPGA-implemented system has demonstrated the ability to construct the audio spectrograms like the *Matlab*® model, with a slight degradation evaluated with an average relative error in energy values of 0.88%. However, up to now, we do not have any information about the power performance of the mapped design. The power consumption measurement in the FPGA case will contain not only the power of the mapped design but also the whole board consumption with the different active blocks beside the Cyclone IV device. Hence, the

challenge here is to de-embed the power consumption of our design from the total power. This is a very complex task to perform that was not resolved up to now.

An alternative solution for the previous challenge is to simulate the power consumption. To do so, the system should be synthesized in a CMOS environment to have a gate design description. For this reason, the feature extractor was synthesized in 65nm CMOS technology (the same technology as the FPGA Cyclone IV device on the board) to be able to calculate the power consumption in the same possible CMOS environment as the programmed design.

4.6.1 System performance

4.6.1.1 Power analysis

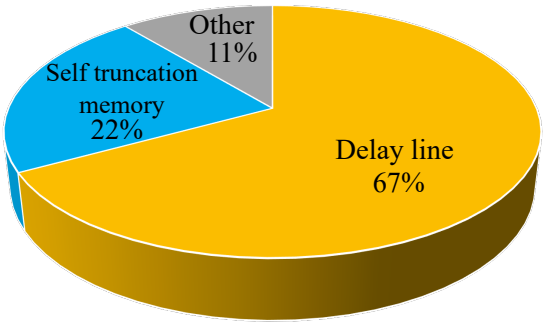
The RTL design was synthesized in the 65nm CMOS technology using high-threshold standard cells to optimize the power efficiency. The power consumption is estimated in 3 different Process, Voltage, and Temperature (PVT) scenarios to visualize the best and worst-case estimations.
The feature extraction unit is completely event-driven and clock-gated with the REQ signal. Then, the power consumption will depend on the application scenario. Since the GSCD database contains keyword samples of 1 second, we decided to study the worst-case scenario of 1 keyword/second which leads to a 60 keywords/minute speech rate.

Table 4.4 Power consumption values in the three PVT scenarios for a speech rate of 60 words/minute.

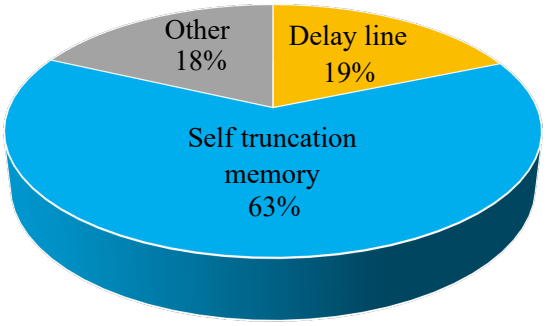
Scenario	(TT, 1.1V, 25°C)	(FF, 1.3V, 125°C)	(SS, 0.9V, -40°C)
Leakage	1.66µW	686.9µW	0.07µW
Total power	9.24µW	695.7µW	4.34µW

For this speech rate, the power consumption estimation results are shown in Table 4.4. The PVT scenarios are chosen to give an idea about the total variation range of the power consumption. In Table 4.4, the power consumption is minimally estimated at 4.34μW when measured at -40°C for the Slow/Slow corner and a supply voltage of 0.9V. In the maximal case, the power consumption is equal to 695.7μW when measured at 125°C for the Fast/Fast corner and a supply voltage of 1.3V. These previous scenarios define the best and worst power consumption cases. But, for the typical case, the power consumption is estimated at 9.24μW when measured at 25°C for the Typical/Typical corner and a supply voltage of 1.1V.

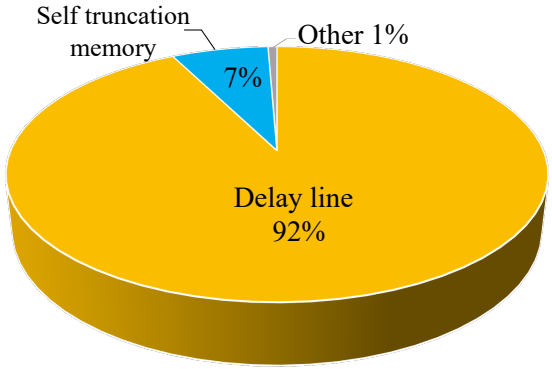
This typical case will be the one considered when comparing our system's performance to other state-of-the-art keyword spotting systems.



(a) Scenario : (TT, 25°C, 1.1V).



(b) Scenario : (FF, 125°C, 1.3V).



(c) Scenario : (SS, -40°C, 0.9V).

Figure 4.22 Power consumption repartition in the three different PVT scenarios.

Furthermore, to analyze which blocks are dominating the power consumption, a power repartition is performed and shown in Fig. 4.22 for each PVT scenario. As seen in this figure, the power repartition is different for each scenario. This difference is due to the domination or not of the leakage power compared to the active one. For example, in the (FF, 125°C, 1.3V) scenario, the leakage power is dominating as shown in Table 4.4. Hence, the power consumption will be dominated by the self-truncation memory since it is the block with the most standard cells with a total of 25600 registers. This is confirmed by the power repartition in Fig. 4.22b, where the self-truncation memory is dominating with a percentage equal to 63%, then the delay line comes second with a percentage equal to 19%. However, in the (SS, -40°C, 0.9V) case, Table 4.4 shows that the power consumption is mainly dominated by active power. Then, the power consumption in this scenario will be dominated by the delay line, because it is the block operating with the highest frequency (8MHz). As a validation argument, the power repartition in this case, depicted in Fig. 4.22c, shows that the delay line part is equal to 92% of the total power, the self truncation memory comes in second place

with 7% of the total power. Finally, in the typical case, the leakage power is equal to 18% of the total power. Hence, the power consumption in this case will stay dominated by the delay line, but with only 67% of the total power.
The conclusion from this power analysis is that despite the PVT scenario, the power consumption is majorly dominated by the delay line and the self-truncation memory, with the biggest part for the delay line in the typical use case.

4.6.1.2 Area analysis

The feature extraction unit, synthesized in the 65nm bulk CMOS technology, occupies a total surface of 0.76 mm^2. This area distribution is depicted in Fig. Figure 4.23. The area is majorly dominated by the self-truncation memory since it is the block with the highest number of registers. This area domination is in line with the power domination in the scenarios when the leakage power is dominating. Besides, the rest of the area is divided between the delay line (22%), MAC units (6%), and the rest of the blocks (7%).

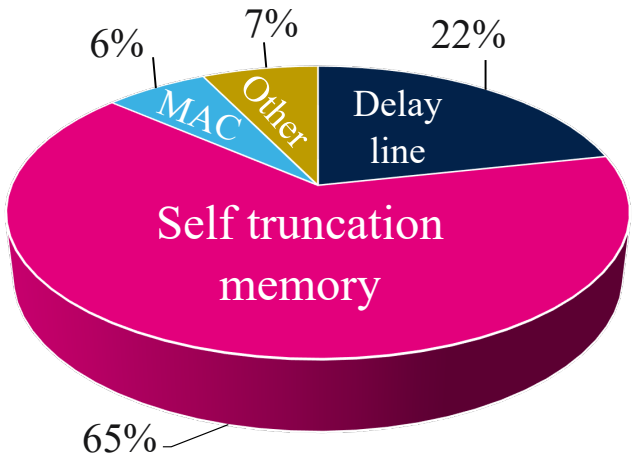


Figure 4.23 Area breakdown of the proposed feature extraction unit.

4.6.1.3 Preliminary comparison to state of the art

The demonstrated system performance will be compared to some of the most recent state-of-the-art keyword spotting systems to validate the competitiveness of our solution. Table 4.5 shows that our system performs the recognition of 10 keywords with an accuracy of 90.4% and a speech-rate dependent power budget of 9.24μW compared to an accuracy of 86.03% and a fixed power consumption of 9.3μW in the system in [4]. The system presented in [5] is showing a power consumption 11 times lower than ours, but we are targeting a higher speech rate of 60 keywords/min instead of 12 in their system. Thus, in terms of

energy efficiency, they are approximately 5 times better while targeting the recognition of a maximum of 1 keyword with an accuracy of 94% instead of 99.1% in our case. The system presented in [6] is 27 times better in terms of power efficiency but it is limited to the recognition of 1 and 2 keywords with an accuracy of 97.3% and 94.6%, respectively, compared to 99.1% and 98% in our system. To sum up, our system presents a competitive trade-off between power consumption, recognition accuracy, and the number of recognized keywords.

Table 4.5 Comparison with state-of-the-art keyword spotting systems.

	This work (Post-synthesis)	[4] Kim ISSCC'22	[6] Shan JSSCC'21	[5] Wang ISSCC'21
Feature Extractor type	8-Band FIR filters	Ring-Oscillator based BPF	Serial-FFT + 16bit ADC	Spike-based Slope detector
Technology	65nm	65nm	28nm	180nm
Area	0.76mm^2	2.03mm^2	0.55mm^2	13.86mm^2
Active Power	9.24µW[a] (Feature extractor)	9.3µW (Feature extractor)	340nW (Feature extractor)	378nW[b] (Complete system)
No activity Power	1.66µW	N/A	N/A	148nW
Energy per event	9.24µJ/event	—	—	1.89µJ/event
Maximum #keywords	10	10	2	1
Accuracy @1word	99.1	N/A	97.3	94
Accuracy @2word	98	N/A	94.6	—
Accuracy @10word	90.4	86.03	—	—

a : power measured for a speech rate of 60 keywords/minute
b : power measured for a speech rate of 12 keywords/minute

4.7 Conclusion

This chapter described the translation of the *Matlab*® software model into an RTL hardware-oriented model, using hardware description language (VHDL). This RTL model allows the construction of the hardware model of the feature extraction part of the keyword spotting system. However, the RTL model is still an ideal model that does not consider all circuit-oriented non-idealities such as timing violations, and mismatch... Hence, to have a complete hardware view of our system, we decided to program our RTL design into an *Intel*® Cyclone IV FPGA validation platform. By doing so, the system will be mapped on real hardware standard cells and will undergo electrical measurements to verify its operation. This is a primordial step to validate the feasibility of our system before starting the ASIC design.

The FPGA-implemented system demonstrated the ability to reconstruct the same spectrograms as the reference *Matlab*® model, with a slight average relative error equal to 0.88%. This error does not impact the recognition accuracy that stays equal to 90.4% in the 10 keywords recognition case. As a complementary analysis, the system was synthesized in 65nm bulk CMOS technology (the same technology as the Cyclone IV device on the FPGA board) to estimate the power consumption of the feature extractor. As a result, the system had an activity-dependent power consumption of 9.24μW in the typical conditions, for a 60 keywords/minute speech rate. These performances are at par or better than state-of-the-art keyword spotting systems.

At the issue of power analysis, we found out that the delay line and the self-truncation memory are the blocks dominating the power consumption, with the highest part for the delay line. Hence, these blocks need to be optimized for low-power implementation when designing the ASIC in a CMOS environment.

To conclude, this chapter demonstrated the hardware feasibility of the feature extraction unit with a competitive performance in terms of recognition accuracy and power consumption. Thus, after optimizing the critical blocks in terms of power consumption, the following step will be to consider the design of this system in a CMOS environment using the different CAD tools to fabricate an ASIC version of our system.

Bibliography

[1] Y. W. Li, K. L. Shepard and Y. P. Tsividis, "A continuous-time programmable digital FIR filter," Proceedings of the IEEE 2005 Custom Integrated Circuits Conference, 2005., San Jose, CA, USA, 2005, pp. 695-698, doi: 10.1109/CICC.2005.1568763.

[2] Prasad Bhanu et al., Speech, Audio, Image and Biomedical Signal Processing using Neural Networks. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008.

[3] P.Warden, "Speech commands: A public dataset for single-word speech recognition." *Dataset available from http://download.tensorflow.org/data/speech_commands_v0.01.tar.gz*,2017.

[4] K. Kim et al., ”A 23µW Solar-Powered Keyword-Spotting ASIC with Ring-Oscillator-Based Time-Domain Feature Extraction,” 2022 IEEE International Solid- State Circuits Conference (ISSCC), 2022, pp. 1-3, doi: 10.1109/ISSCC42614.2022.9731708.

[5] Z. Wang et al., ”A 148nW General-Purpose Event-Driven Intelligent Wake-Up Chip for AIoT Devices Using Asynchronous Spike-Based Feature Extractor and Convolutional Neural Network,” 2021 IEEE International Solid- State Circuits Conference (ISSCC), 2021, pp. 436-438, doi: 10.1109/ISSCC42613.2021.9365816.

[6] W. Shan et al., ”A 510-nW Wake-Up Keyword-Spotting Chip Using Serial-FFT-Based MFCC and Binarized Depthwise Separable CNN in 28-nm CMOS,” in IEEE Journal of Solid-State Circuits, vol. 56, no. 1, pp. 151-164, Jan. 2021

Chapter 5

28nm FD-SOI CMOS event-driven mixed-signal spectral-based feature extractor design and physical implementation

5.1 Introduction

The hardware feasibility of the proposed keyword spotting system has been validated through the implementation on an *Intel*® FPGA board. Besides, the system was synthesized on a 65nm bulk CMOS technology to perform a power analysis. As a result, the power was dominated by the delay line. Hence, this block needs to be improved for better power efficiency.

The obtained hardware validation results will be used to start designing a keyword spotting System on Chip (SoC) in an advanced CMOS technology node compatible with low-power systems. The physical implementation will concern only the feature extraction unit. The design strategy will be mainly focused on optimizing the power consumption.

This chapter will introduce the technology node used for the physical implementation, as well as the different steps, followed when going from a behavioral model up to a complete physical on-chip system. The challenges faced during the on-chip system design can be summarized in the design of a new delay line and the difficulty of adapting the semi-custom digital design flow to synthesize a clockless system.

5.2 28nm FD-SOI technology choice

The technology node is chosen based on two main criteria: Leakage and signoff frequency. Since we are dealing with low-frequency audio signals, the constraint on the signoff frequency is relaxed. Hence, the most important parameter becomes only the leakage.

The feature extraction unit was designed in STMicroelectronics 8 metal layers 28nm FD-SOI CMOS technology. The circuit implementation is constrained to operate at low supply voltages to improve power efficiency. In our case, we are talking about a supply voltage that can go as low as 0.6V. The FD-SOI technology offers more margin to decrease power consumption through the Body Bias feature. The challenge appearing in this case is to sophisticate the design of the different building blocks in the system while ensuring robustness to PVT transistors and operating properly at low supply voltages combined with the added body bias.

FD-SOI technology represents the perfect candidate to reduce the short channel effects (SCE) and to improve the channel control thanks to the ultra-thin Buried Oxide (BOX) layer. Moreover, the body bias feature offers an additional degree of freedom to improve either the circuit latency or power consumption. Fig. 5.1 shows the two different transistor families available in this technology: ***Regular-***V_{Th} (RVT) and ***Low-***V_{Th} (LVT). The threshold voltage of the two cell families varies depending on their physical structure.

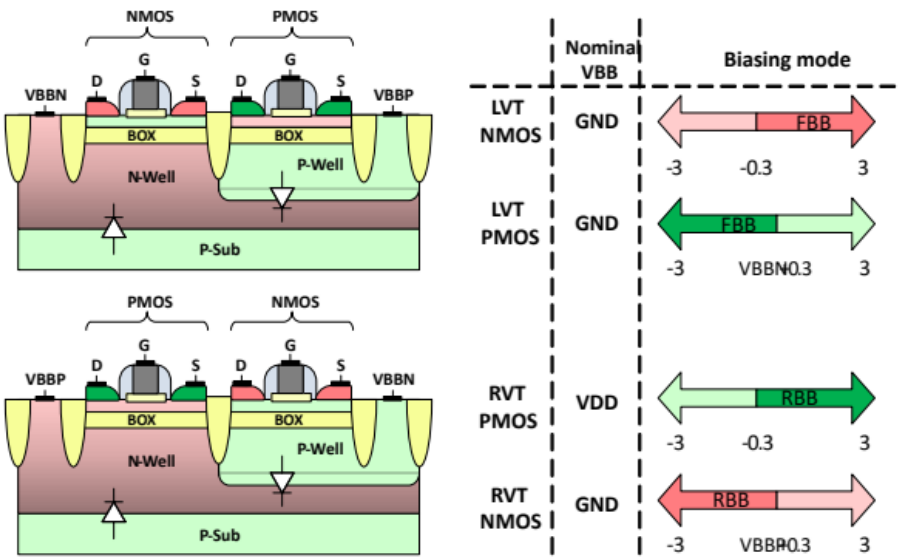


Figure 5.1 FD-SOI family devices. (a) LVT using Forward Body Bias (FBB) to decrease V_{th}, and b) RVT using Reverse Body Bias (RBB) to increase V_{th}.

Each cell family reacts differently to the body bias application. For the RVT family, the PMOS device is built over an N-well, which means it requires a negative voltage at its body back-gate. This configuration is called Reverse Body Bias (RBB). It induces an increase in the threshold voltage (V_{Th}), which means less leakage. On the other hand, the NMOS device in the LVT family is built over an N-well. This means that the required voltage applied at the body terminal will be positive. This configuration is known as Forward Body Bias (FBB). This configuration has the effect of decreasing the threshold voltage (V_{Th}), which means more leakage but high speed (low latency). The RBB and FBB effect on the threshold voltage for the two RVT and LVT transistors is shown in Fig. 5.2. As described in this graph, the RVT cells with RBB features are characterized by a higher threshold voltage (V_{th}) on average, which means they offer a better opportunity to improve power efficiency compared to LVT cells.

The keyword spotting system deals with low-frequency (<8kHz) audio signals. Moreover, considering the target of an ultra-low power implementation, the RVT cells will be more suited for this application. Hence, they will be used for most of the feature extraction unit design.

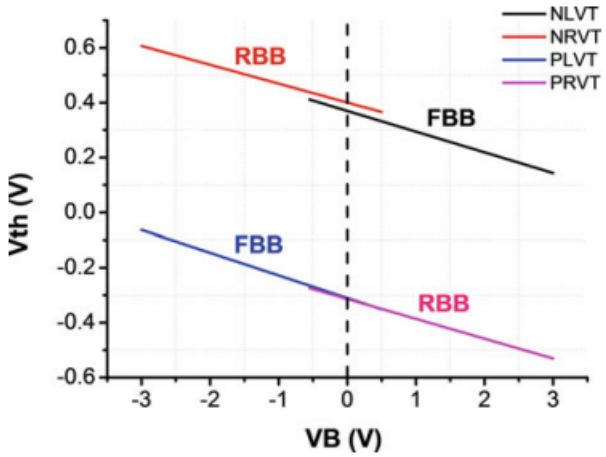


Figure 5.2 Threshold voltage (V_{th}) variation for RVT and LVT transistors, N- and P-MOS in 28 nm FD-SOI technology, for the respective applied body-bias voltage (VB) [1].

Additionally, the 28nm FD-SOI technology offers a wide range of digital standard cells, that are adapted for different kinds of applications. Globally, three types can be used either for low-power circuits, mainstream systems such as micro-controllers, or high-performance systems. For keyword spotting application, we are targeting an ultra-low power target, thus we will be using the first standard cells type.

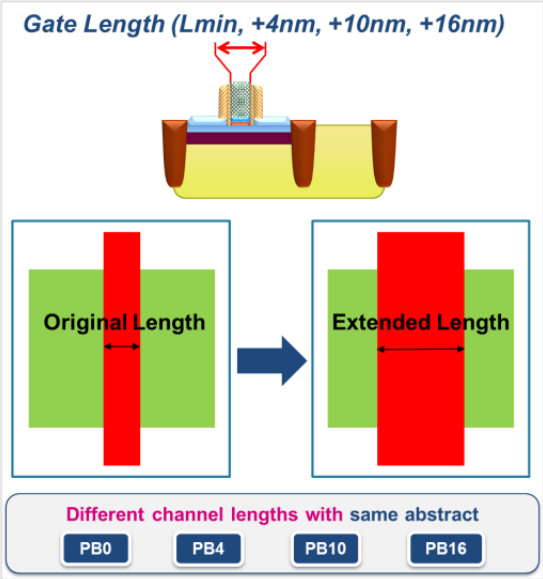


Figure 5.3 Multiple channel lengths (poly biasing) support in 28nm FD-SOI standard cells [2].

In the previous standard cell types, there are multiple transistor structures with different channel lengths. The basic channel length of the transistors in this technology is 28nm, but there are other options to increase the channel length depending on the target application. Fig. 5.3 shows that the original channel length can be extended either by 4, 14, or 16nm. The bigger the channel length, the lower the leakage power. Table 5.1 shows the gain in leakage power when increasing the channel length. Poly Biasing 16 (PB16) standard cells are characterized by a leakage power that is 30X lower than the standard PB0 cells. For keyword spotting application, we are dealing with low-frequency audio signals, which means that the standard cell speed is not a critical requirement compared to the low power performance. Hence, the feature extraction system will be mainly designed using PB16 digital standard cells.

Table 5.1 Leakage reduction achieved by poly-biasing in 28nm FD-SOI technology [2].

Gate Length (Lgate)	Poly-Bias Options	Leakage Reduction
L_{min} (+0 nm)	PB0	1
+4nm	PB4	x4
+10nm	PB10	x10
+16nm	PB16	x30

5.3 Proposed keyword spotting system

After detailing the different building blocks in the previous chapter, the top architecture of the proposed feature extraction unit is depicted in Fig. 5.4. As a reminder of the operation, The audio signal is decoded using a 6-bit LC-ADC to generate the REQ and DIR tokens. These tokens are then input into 16 successive delay taps. After that, all the intermediate delayed signals $REQ_{(0,..,15)}$ and $DIR_{(0,\ldots,15)}$ are simultaneously injected into the 8 parallel digital processing bands. In each band, (*REQ, DIR*) tokens trigger four processing stages. The first stage is the filtering done using a clockless CT FIR filter. The architecture of this filter consists of 16 MAC units, where the multiplication is replaced by a *Req*-gated "addition+accumulation" process. Then, an adder tree generates the filter output by adding the different MAC outputs. These filters are generic thanks to a coefficients bank which can be programmable through an external SPI interface. The second processing stage is the programmable pulse combiner, which concatenates the selected delayed *Req* signals to create the combined *Req* signal. The third processing step is dedicated to energy calculation. This is done through computing and accumulating power values. After squaring the filter's output, the result is fed into 3 parallel framed energy accumulators, where power values are successively accumulated during the frame-high period whenever there is a pulse at the combined *Req* signal to maintain an event-driven operation. The frames are internally generated using a pulse width modulator. For audio matters, the frame duration is set to 25ms with a 10ms overlap. Finally, the 25-bit energy values can either be streamed to construct a real-time continuous spectrogram or can be injected into the truncation memory responsible for storing and truncating the spectrogram after finding the maximum value across the 128 memory rows in all the bands ($>$1 second of recording).
The ASIC design will include all the blocks shown in Fig. 5.4 except the 6-bit LC-ADC, which emulated externally during the test procedure.

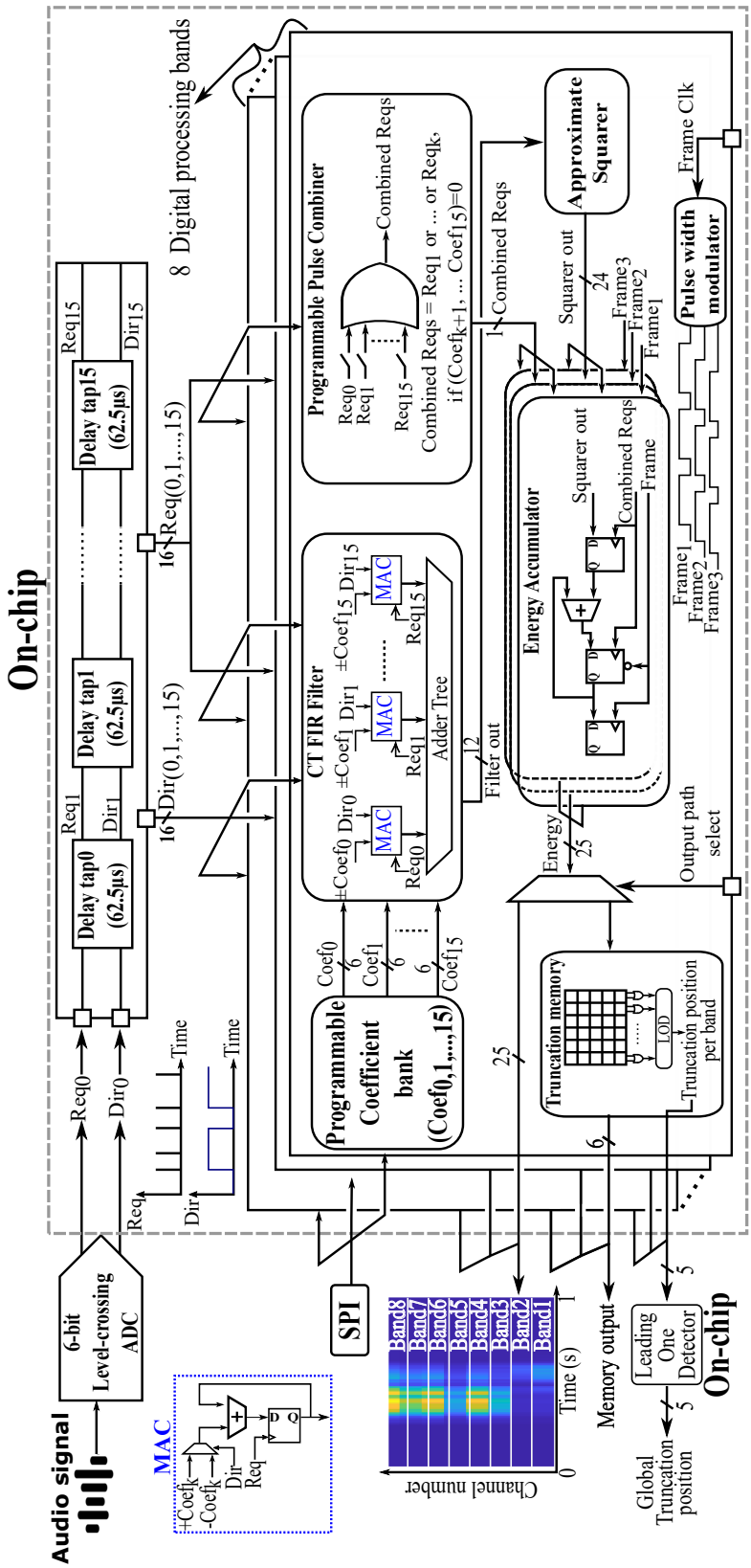


Figure 5.4 Architecture of the feature extraction unit.

5.4 Delay line

5.4.1 Delay creation principle

The delay line represents the critical block in terms of power consumption. Moreover, this block was designed using clocked counters in the FPGA hardware implementation. However, this is not compatible with the event-driven clockless nature of our system. Thus, this block needs to be re-designed to maintain a complete clockless operation and improve the power efficiency.

In a digital environment, a controlled delay is only created using counters. However, this is not an acceptable solution in our use case. Hence, the delay needs to be created in an analog way. The delay creation principle in the analog domain is based on capacitance discharge control. More specifically, the delay is defined by the following equation:

$$\tau = \frac{C.\Delta V}{I_{Control}} \tag{5.1}$$

with τ is the created delay that is proportional to the product of the capacitance C and the voltage headroom ΔV, but inversely proportional to the control current $I_{Control}$.

5.4.2 Delaying techniques

To create the previous delays, multiple analog techniques can be adopted. These techniques are detailed and discussed in the thesis work presented by I.Sourikopoulos in [3]. The delay creation techniques can be summarized in :

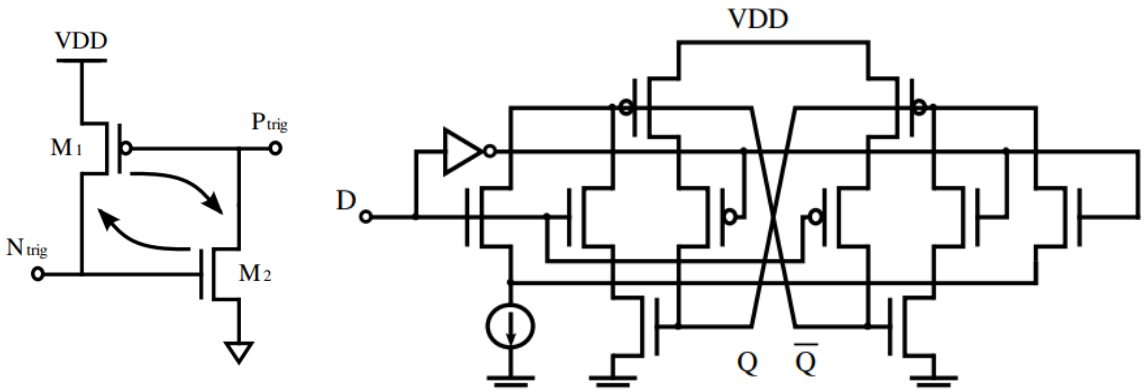


Figure 5.5 (a) Thyristor-delay concept [8]. (b) Example of a thyristor delay element [8].

- **Cascaded inverters**: This is the most simplistic technique used to create a delay that will be equal to the sum of the propagation delays of all the inverters in the delaying chain. The disadvantage of this technique is the high delay sensitivity to process, voltage, and temperature variations [4].

- **Capacitive shunting**: This technique creates a delay through the charge of a load capacitance. Thus, the delay will be directly dependent on the time constant. This technique, described in [5], presents the same inconvenience as the previous cascaded inverters, as well as having a power consumption proportional to the delay.

- **Semi-static topology**: This kind of topology is known for its advantage of reducing short circuit current generated during the low/high and high/low transitions, as shown in [6]. However, this advantage is countered by an augmentation of the static power consumption.

- **Current-starving**: This technique is done by adding an extra transistor in series with the PMOS and NMOS transistors of the inverter. In this type of architecture, the delay is controlled by the gate voltage of the starving transistors. Multiple architecture topologies, based on this technique, have been proposed in the state of the art [5], [7]. One of the main disadvantages of this technique is the disability to have very sharp state transitions, which will produce a higher short-circuit current.

- **Thyristor-based technique**: This technique was introduced in [8]. The thyristor operation principle is shown in Fig. 5.5a. It is described by the activation of a device when the threshold voltage level is crossed. For delaying, the thyristor event to be triggered will be the charge or discharge of a capacitance, when the voltage across its terminals crosses the set threshold level. An example of a thyristor-based delay element is shown in Fig. 5.5b. The delaying concept is discussed in more detail in [3]. This technique presents multiple advantages such as very low static power consumption and a reduced supply sensitivity [3], nevertheless it suffers from charge sharing.

Table 5.2 presents a qualitative comparison of the different delaying techniques. These techniques are compared based on four main parameters: static power, energy per event, supply, and temperature sensitivities. Based on this comparison, thyristor-based delays represent the best delaying solution, as it shows near-zero static consumption, a constant energy/toggle, and a very low supply and temperature sensitivity. Hence, the basic delaying elements to be used in the proposed feature extraction unit will be based on the thyristor concept.

Table 5.2 Qualitative comparison of the different delay creation techniques [3].

Type	Inverter chain	Capacitive shunt	Semi-static	Current-starved	Thryistor
Static power	no	no	yes	yes	no
Energy/toggle	$\propto$ to delay	$\propto$ to delay	$\propto$ to delay	$\propto$ to delay	constant
Supply sensitivity	high	medium	medium	medium	low
Temperature sensitivity	high	medium	medium	medium	very low

5.4.3 Proposed delay element

The delay line in the feature extraction unit will be composed of a series of analog thyristor-based elementary delay elements. These elementary delays are the granular delays. Taking into account the audio keyword spotting application and its event-driven clockless behavior, these delays should fulfill the following conditions:

- Ability to propagate REQ pulses and DIR data.
- Ability to delay pulses with a width of 62.5ns (period of the oversampling clock used in *Matlab*® simulations.).
- Tunable delay value in the limit of the application.
- Ability to communicate with adjacent delays through an asynchronous handshaking protocol.

An example of a thyristor-based delay element was proposed in [9]. This delay element was used as an elementary granular delay for CT FIR filters used for keyword spotting application [10]. This delay respects all the previously mentioned constraints. Thus, the granular delay design will be inspired by this work in our case.

This delay element, presented in Fig. 5.6, is composed of three main parts:

- **Discharge control :** This part is responsible for creating the delay through a capacitance discharge when the voltage between its terminal crosses a certain threshold voltage defined by the biasing voltage and the control current.
- **Handshaking :** This mechanism is responsible for ensuring safe communication between a delay and its adjacent neighbors. More specifically, a certain delay cannot

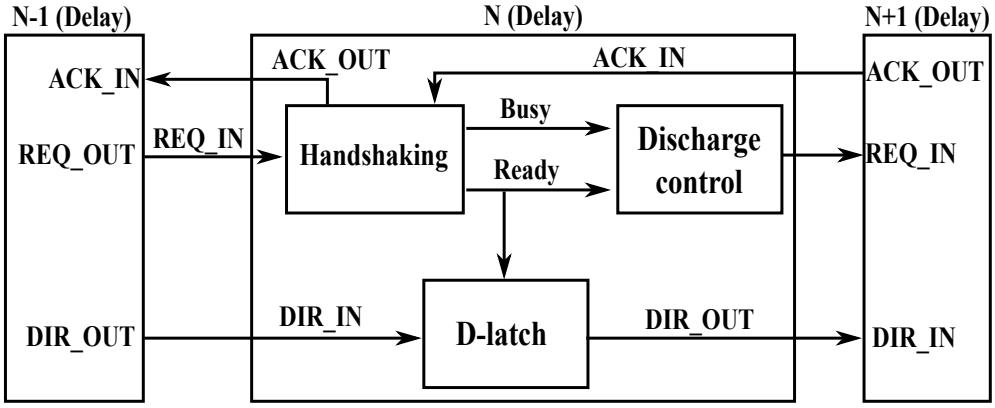


Figure 5.6 Architecture overview of the delay element.

forward a pulse to the next delay if it does not receive and acknowledge the signal from this next delay. In the same perspective, a delay transmits an acknowledge signal to the next delay when it finishes the delay to inform the next delay that the output pulse is ready.

- **D-latch :** This block is responsible for delaying the data, which is the DIR signal in this case.

The operation principle is described by the waveforms sketched in Fig.5.7. By default, the delay is always in the Ready state. However, when a pulse arrives in the `REQ_IN` signal and the `ACK_IN` signal is at a high state, the delay enters a Busy state and performs the delaying operation until outputting the pulse in the `REQ_OUT` signal and the corresponding `DIR_OUT` data. Then, the delay re-enters the Ready state and transmits the information of delaying completion to the previous delay unit through the `ACK_OUT` signal. This process is repeated for each each pulse at the `REQ_IN` signal until delaying all the needed information. Moreover, the handshaking protocol managed by the `ACK_IN` and `ACK_OUT` signals, ensures that the delaying operation will not get disturbed by the arrival of a pulse during the Busy state. If a pulse arrives during an ongoing delaying operation, this pulse will be completely ignored and lost.

5.4.4 Granular delay schematic

The MOS schematic of the granular delay, detailing the different previously mentioned building blocks, is depicted in Fig. 5.8. Starting with the handshaking block, it is composed of a Muller gate [9] and an SR-latch. The Muller gate more known as the C-element in asynchronous circuits design, is an asynchronous logic block that maintains its output

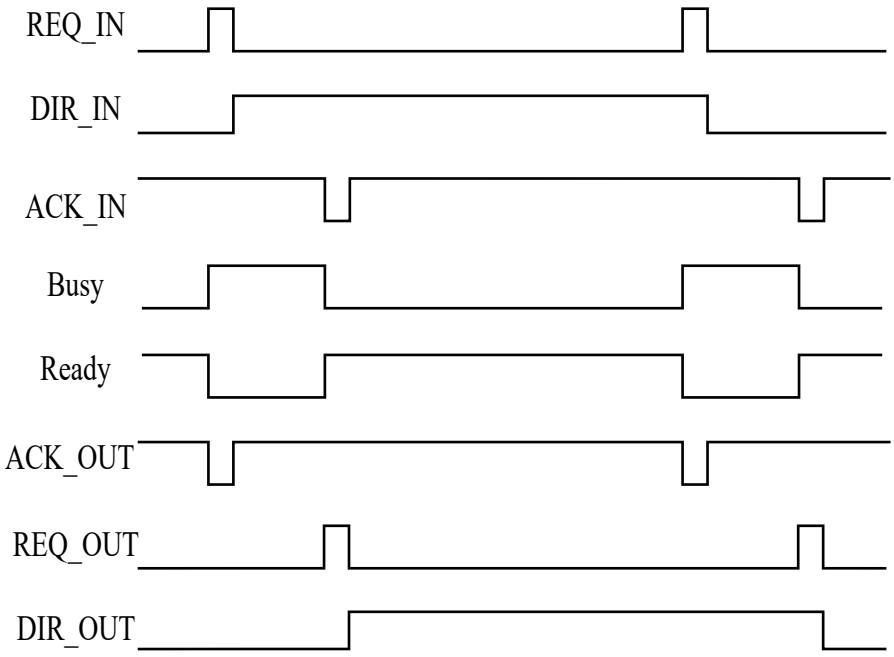


Figure 5.7 Operation of the delay element.

unchanged until both of its inputs settles to the same value. The conventional architecture and truth table of the Muller gate [9] are shown in Fig. 5.9. Besides these two sub-blocks in the handshaking part, we added an inverter responsible for setting the circuit to reset and starting states, depending on the Start signal value. In this case, the circuit is positively reset at the Start high state. The addition of this inverter allowed the proper glitch-free reset of the circuit without the appearance of a short circuit at the rising edge moment of the Start signal. The second principal bloc of this delay element is a NAND-based D-latch with the mission of memorizing the DIR data state and outputting the same data value after finishing the delaying operation.

The final part of the delay element is the discharge control. As shown in Fig. 5.8, this part is the one responsible for controlling the delay operation through the discharge of the highlighted capacitance. In more detail, this block is firstly biased through the $V_{Control}$ voltage, which induces the control current $I_{Control}$. Then, with the arrival of a pulse at the input of the delay element, the capacitor starts to charge slowly until lowering the voltage at the M9 PMOS gate below a certain threshold level. After that, the M9 transistor acts as a closed switch, which pushes the capacitor to discharge rapidly through the M12 NMOS transistor. The delaying operation is finished when the capacitor is completely discharged. Finally, the delay element goes back to its initial ready state waiting for new input pulses.

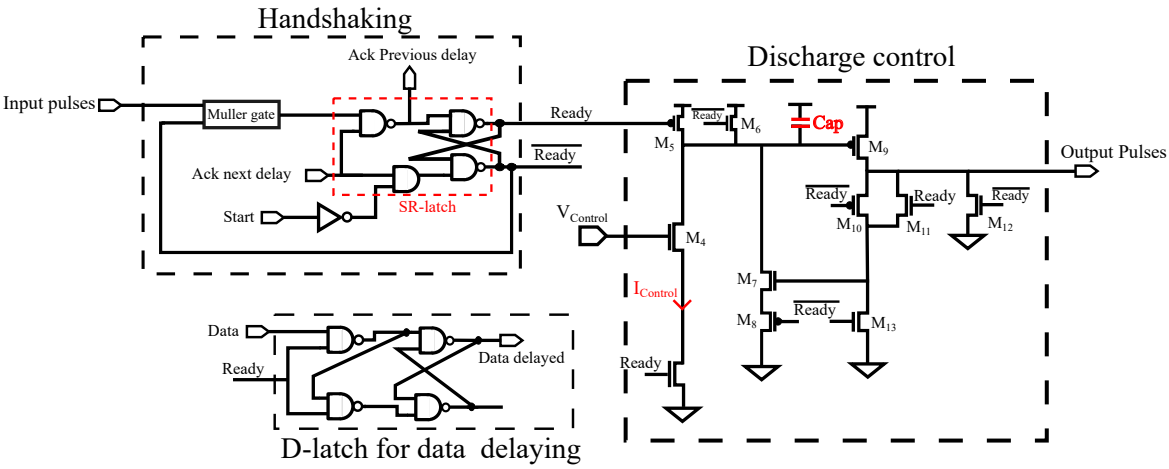


Figure 5.8 Granular delay schematic.

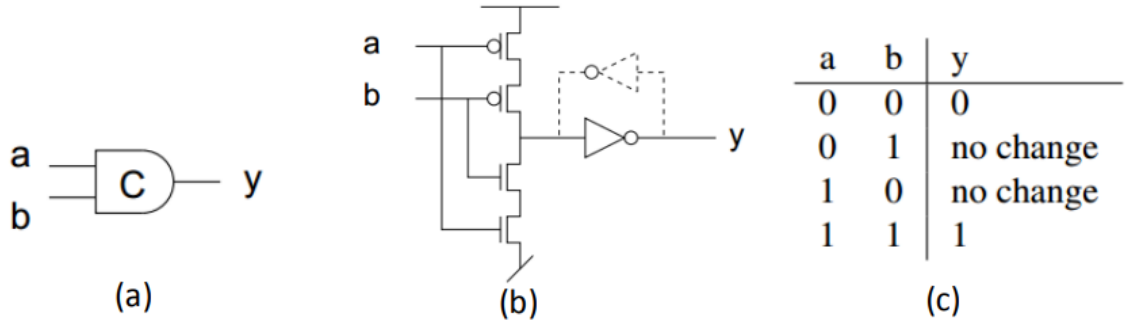


Figure 5.9 (a) Muller gate symbol. (b) Muller gate architecture. (c) Muller gate truth table.[9]

5.4.5 Granular delay characterization

5.4.5.1 Transistors sizing

The different PMOS and NMOS transistors in the discharge control part were sized iteratively until the optimum size for each transistor to have the optimal delay operation. The different sizes are depicted in Table 5.3. Another point, all the transistors were taken from the RVT 28nm FD-SOI technology library. Besides, the handshaking mechanism and D-latch were designed using a 28nm FD-SOI low-power RVT standard cell library with an increased gate thickness.

The back gate of all the transistors, used in the delay element design, was shorted to zero body bias.

Table 5.3 Sizing of all the transistors in the discharge control part.

Device	M3, M7, M10, M11	M4, M9	M5	M6	M8	M12	M13
W(nm)/L(nm)	300/40	300/100	200/100	200/80	400/50	160/100	160/400

5.4.5.2 Capacitance size

To size the key capacitance, it should respect the low power performance. The capacitance value should be chosen to minimize the consumed energy. The dynamic energy is expressed using the following equation:

$$Energy = \frac{Cap.\Delta V^2}{2} \tag{5.2}$$

The energy of the delay element is directly proportional to the capacitance value. Hence, to minimize the energy, we should minimize the capacitance value. For this reason, the capacitance value was set to 4.5fF, as this is the minimal Metal-Oxide-Metal (MOM) capacitance value available in design kit.

5.4.5.3 Delay performances

To completely characterize the designed thyristor-based granular delay, we performed multiple simulations to study the delay dependency on the control current and delay variability due to the process and mismatch variations. Moreover, a power analysis is done also to evaluate the delay element performance in terms of power consumption and extract the most consuming parts in the design to see if they can be more optimized to further improve the power efficiency.

Delay vs Control current: The delay variation as a function of the control current is described in the graph shown in Fig. 5.10a. As expected, the delay value increases exponentially when decreasing the control current. For keyword spotting application, the granular delay was set to 1.25μs. This delay value is generated by the delay element when biased by a current value of 1.93nA.

Input signal dependency: One of the main challenges faced by the different delay designs is to isolate the effect of the input signal frequency on the delay value. Normally, the thyristor-based delay is meant to resolve this problem. To ensure the input signal independence, a simulation varying the frequency of the pulses in the REQ signal is performed on the delay element. This simulation is done on a delay element configured at 1.26μs. The result of this simulation is shown in Fig. 5.10b. The maximum delay variation is about 5

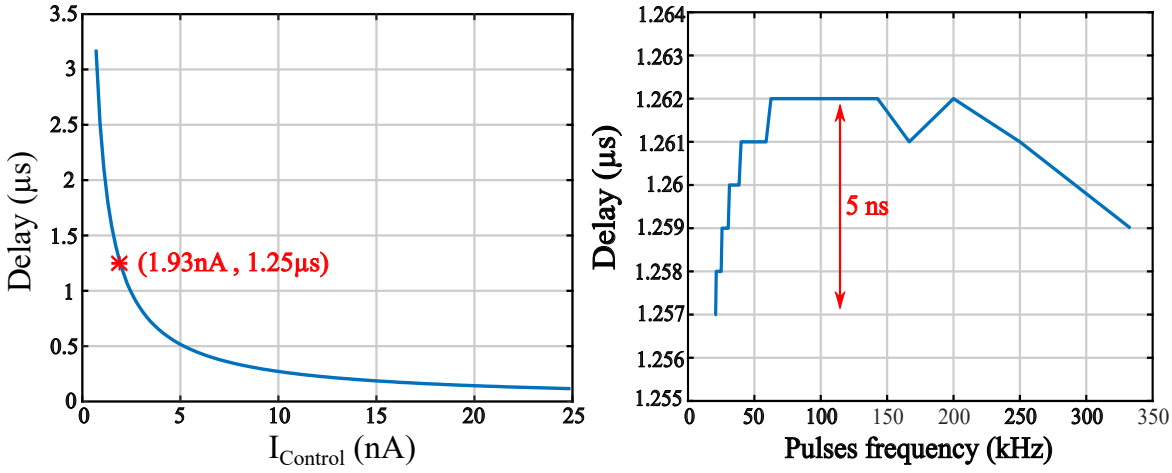


Figure 5.10 (a) Evolution of the delay value vs the biasing control current. (b) Variation of the delay value as a function of the frequency of the pulses in the REQ signal for a delay element configured at 1.26μs and biased at 1V.

ns when varying the frequency from 0 up to 350kHz. This represents only a 0.4% variation from the mean delay value.

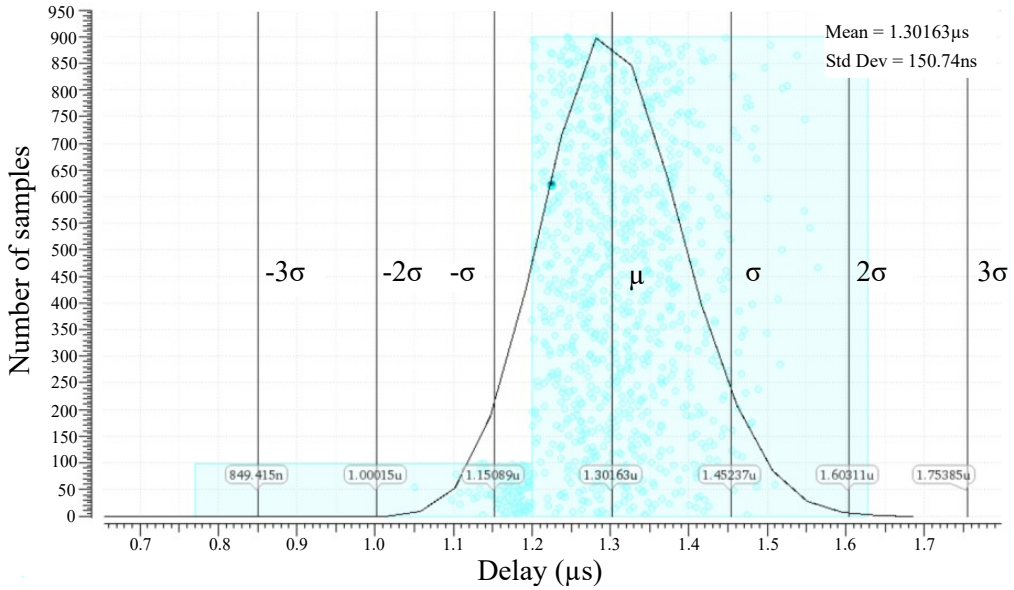


Figure 5.11 Montecarlo simulation (1000 samples): delay variation due to process and mismatch variations

Delay mismatch: In order to determine the respective contribution of PVT (global) and mismatch (local), Fig. 5.11 shows the result of the Montecarlo simulation describing the delay variation with respect to the mismatch and process variations. The delay variation follows a Gaussian distribution around the average value of 1.301μs, with a standard deviation of 150.7ns. Moreover, it is shown in Fig. 5.11 that the maximum delay variation is equal to 2σ(standard deviation). This is evaluated at 23% maximum variation due to the process and mismatch variations.

5.4.5.4 Power analysis

The objective of the power analysis is to evaluate the power consumption of the delay element, to perform the decomposition of the total power into dynamic and leakage power and to extract the power repartition.

The first step is to perform a DC static analysis to evaluate the leakage power. This analysis identified some very leaking transistors that increase drastically the power consumption. These transistors are the M5 and M12. Hence, these leaking transistors were replaced by thick oxide transistors offered by the 28nm FD-SOI technology, called EGVRVT transistors. This type of transistor is characterized by a threshold voltage of around 700mV. After this replacement, the delay element was re-tested and the conclusion was the reduction of leakage power without any significant impact on the delay functionality.

After replacing the leaking transistors, the power analysis was performed on the delay element to evaluate the static and dynamic power consumption. The supply voltage used for the delay element varies from 0.7V up to 1V. The delay element kept operating properly at the limit of a supply voltage of 0.7V. Thus, the power analysis will show the delay element power performances for the mentioned supply voltage range. The static power analysis

Table 5.4 leakage evolution with the supply voltage.

Supply voltage (V)	0.7	0.8	0.9	1
Leakage (pW)	171.2	224.2	312.5	482

results are depicted in Table 5.4. The delay cell demonstrated a sub-μW leakage power, precisely ranging from 171.2pW up to 482pW when varying the supply voltage from 0.7V to 1V. This leakage power repartition is shown in Fig 5.12a. The leakage is dominated by the handshaking and the D-latch that are taking 72% and 25% of the total power, respectively. Finally, the discharge control part is the less consuming block by only a percentage of 3%.

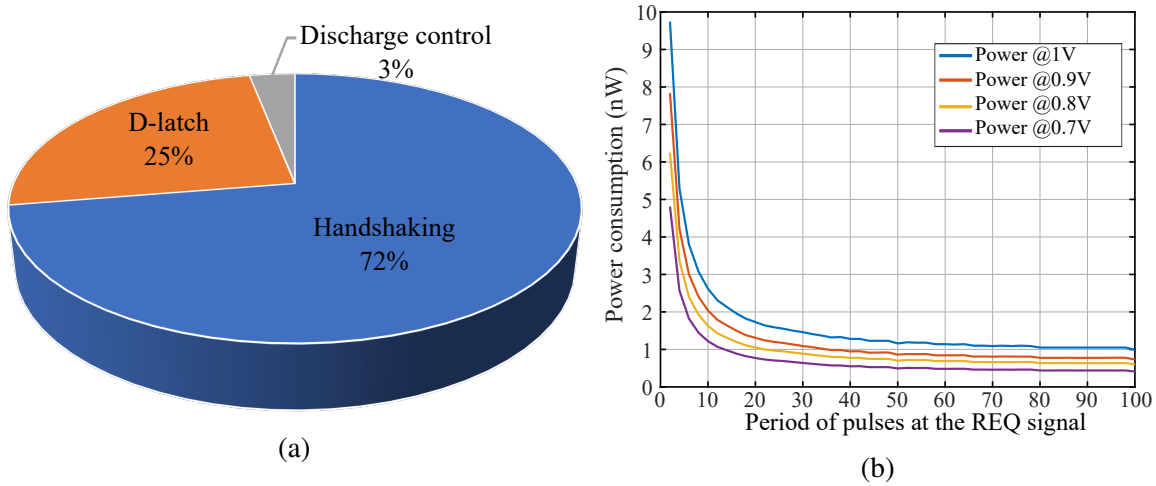


Figure 5.12 (a) Static power repartition. (b) Evolution of the total power as a function of the period of pulses at the REQ signals, for different supply voltages.

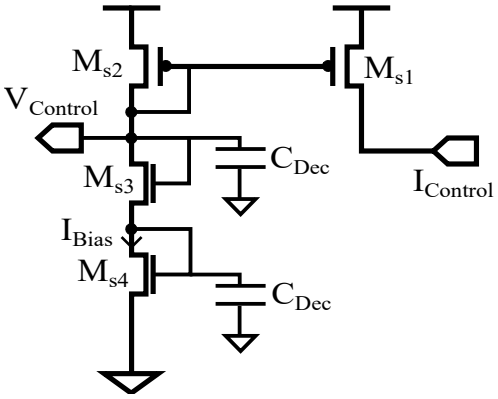
The static power analysis is complemented with a dynamic power one. The graph in Fig. 5.12b shows the evolution of the total power (dynamic + leakage) as a function of the pulses period in the REQ signal, for different supply voltages. The power increases when the period of the pulses decreases. Thus, the power is completely gated by the signal activity. The energy per event will indicate the dependence of the power on input signal activity. This energy is evaluated for different supply voltages. The values of the energy per event are shown in Table 5.5. As a result, the energy is evaluated at a minimum of 24.09fJ/event for a supply voltage of 0.7V.

Table 5.5 Energy per event evolution with the supply voltage.

Supply voltage (V)	0.7	0.8	0.9	1
Energy/event (fJ)	24.09	34	43.24	61.3

5.4.6 Cascoded biasing block

In the previous part, the characterization of the delay element was done using ideal current references. However, this solution is not physically implementable. For this reason, the delay element will need a biasing block to generate the $V_{Control}$ signal. The architecture of the biasing stage is shown in Fig. 5.13. It is composed of a 2 stage cascode current mirror, with adding a decoupling capacitance at the gate of each diode-connected transistor. The



Device	Size (W/L)
M_{s1}	400n/600n
M_{s2}	400n/600n
M_{s3}	300n/100n
M_{s4}	300n/500n

C_{Dec} = 21fF

Figure 5.13 Cascode current mirror architecture and sizing of its different devices.

cascode topology with added decoupling capacitance will offer the advantage of stabilizing the control voltage $V_{Control}$. This is demonstrated in Fig. 5.14a, where $V_{Control}$ is simulated in time for different control currents. For a specific current value, the control voltage stays quasi-constant in time.

Besides, the size of all the transistors is shown in the table shown in Fig. 5.13. Most of transistors have a width bigger than the length. The purpose behind this choice is to decrease the current mismatch. The graph in Fig. 5.14b shows the variation of the current mismatch between $I_{Control}$ and I_{Bias}, with I_{Bias} the copied current at the second branch as shown in Fig. 5.13. This biasing architecture has demonstrated low mismatch values equal at maximum to 49.5 pA.

The I-V characteristic of the cascode current mirror is depicted in Fig. 5.14c. The voltage increases logarithmically with the increasing input control current. To have a delay value of 1.25 μs, the delay element is biased with a current of 1.93nA, which corresponds to a voltage value of 585.39mV.

5.4.7 Delay tap architecture

After finalizing the architecture and characterization of the granular delay cell, the next step will be to implement the delay tap. As mentioned in chapter III, the delay tap is composed of 50 successive granular delays, which results in a delay tap value of 62.5 μs. One of the main challenges in this case is to find the optimal topology to create this delay chain. For this reason, the granular delays inside each delay tap are regrouped in the form of a serpentine chain, as shown in Fig. 5.15. The advantage of this architecture is to minimize the distance between the first and last delay to reduce the delay mismatch. In addition, the serpentine topology offers the best shape in terms of aspect ratio to minimize the occupied

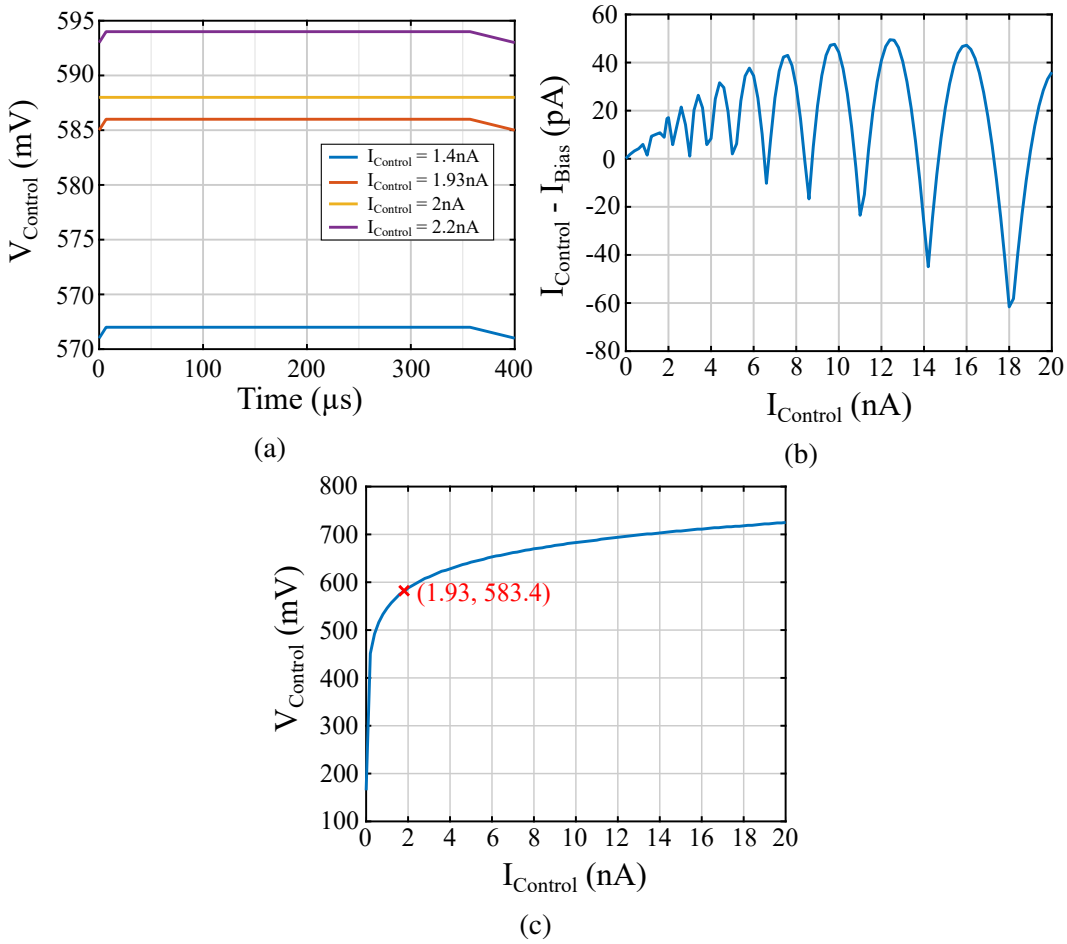


Figure 5.14 (a) Control voltage evolution in time for different control current values. (b) Current mismatch of the biasing stage. (c) I-V characteristic of the biasing stage.

area. However, this architecture suffers from the appearance of more parasitic capacitance, compared to a straight chain architecture. This means more power consumption. But, the straight topology suffers from a bigger delay mismatch due to far distance between delays and it is not a physically feasible solution, because it will be a very long and very narrow architecture, which is not implementable on chip. Hence, the serpentine architecture presents the best tradeoff between shape factor, mismatch, and power consumption.

Due to the adoption of a serpentine architecture and for matters of easy integration and optimized mismatch, the number of granular delays in each delay tap is reduced to 48. This is done because, in a FIR filter, where these delays are used, it is more suitable to have the input and the output of the delay tap on the same side since they are connected to the same logic. This results in a serpentine architecture, with 12 granular delays per line, as depicted in Fig. 5.15.

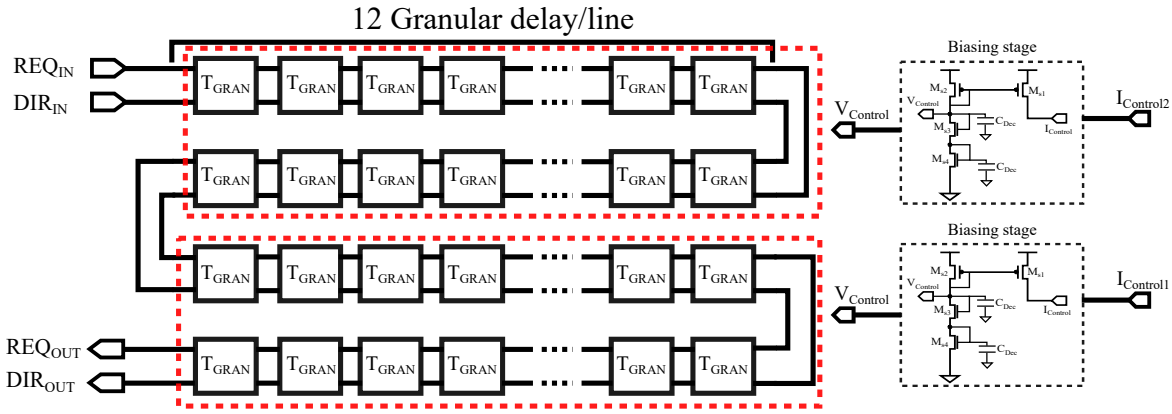


Figure 5.15 Serpentine architecture of the delay tap.

The reduction in the number of granular delays in each delay tap, means that the granular delay value needs to be increased from 1.25μs up to 1.302μs. This translates into reducing the biasing control current from 1.93nA down to 1.81nA.

Another point, each delay tap will be equipped with two biasing cascode current mirrors that will be sufficient to bias all the 48 granular delays. Each biasing block is common to 24 granular delay cells to correlate the control current variation effect on the corresponding delay cells.

5.4.8 Delay line architecture

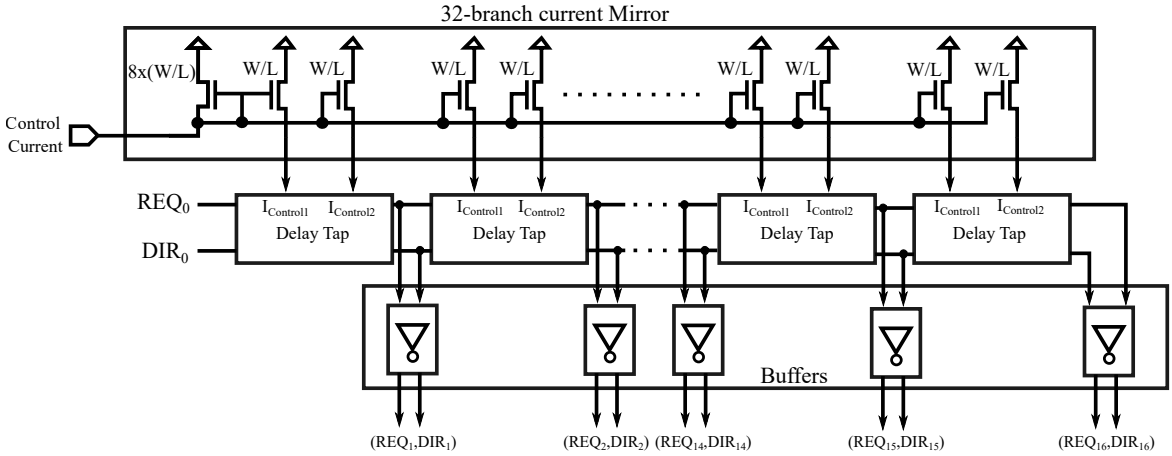


Figure 5.16 Complete architecture of the 16 delay line with biasing current mirror and output buffers.

The final delay line is composed of 16 successive delay taps. Each delay tap is biased using two cascoded current mirrors. These biasing stages need to receive the control current to deliver the cascode control voltage to the corresponding granular delay cells. These control currents cannot be brought externally because we will need 32 IO pads. This is a very demanding solution in terms of hardware resources. To avoid this solution, we implemented a 32-branch current mirror responsible for distributing the control current to all the delay taps. This current mirror divides the entry current by a ratio of 8. The reason for to increase in the entry transistor size is the difficulty of having a current generator able to deliver a stable low current value of 1.81nA. After verifying the availability of the test material in the lab, we found a current generator able to deliver low current at the limit of 10nA. Hence, we decided to have a current mirror with a ratio of 8 to have an entry current of 14.48nA. The architecture of the delay line and the 32-branch current mirror is shown in Fig. 5.16. In addition, the intermediate outputs of each delay tap are passed through a buffer to reinforce the signal. Moreover, since the output of each delay tap is transmitted to the digital processing part, the buffers will also serve to sharpen the rising and falling edges in the REQ and DIR signals to allow faster toggling.

5.5 Physical implementation details

Due to the analog architecture of the delay line, it will be physically implementable using an analog full custom flow. On the other side, the RTL-designed digital part will be fully synthesized using only standard cells and will be physically implemented using automatic semi-custom Place & Route (P&R) digital flow.

5.5.1 Full custom delays design

5.5.1.1 Granular delay layout

The granular delay layout is done in a fully custom way. The layout is done optimally to decrease the parasitic capacitance effects, power consumption, and area. Moreover, since the granular delay cell will be duplicated 768 times to form the delay line, it should be designed as a Lego basic cell for easy integration in a bigger layout. Fig.5.17 shows how the granular delay layout is arranged to optimize area. The Muller gate and handshaking are positioned vertically side by side. This is done because these two blocks are fully made of standard cells. The standard cells in 28nm FD-SOI technology have a standard layout with the same height and aligned poly-silicon rods, which means they can be easily assembled for a very compact layout. This can be seen at the top part of the granular delay layout. Moreover, the

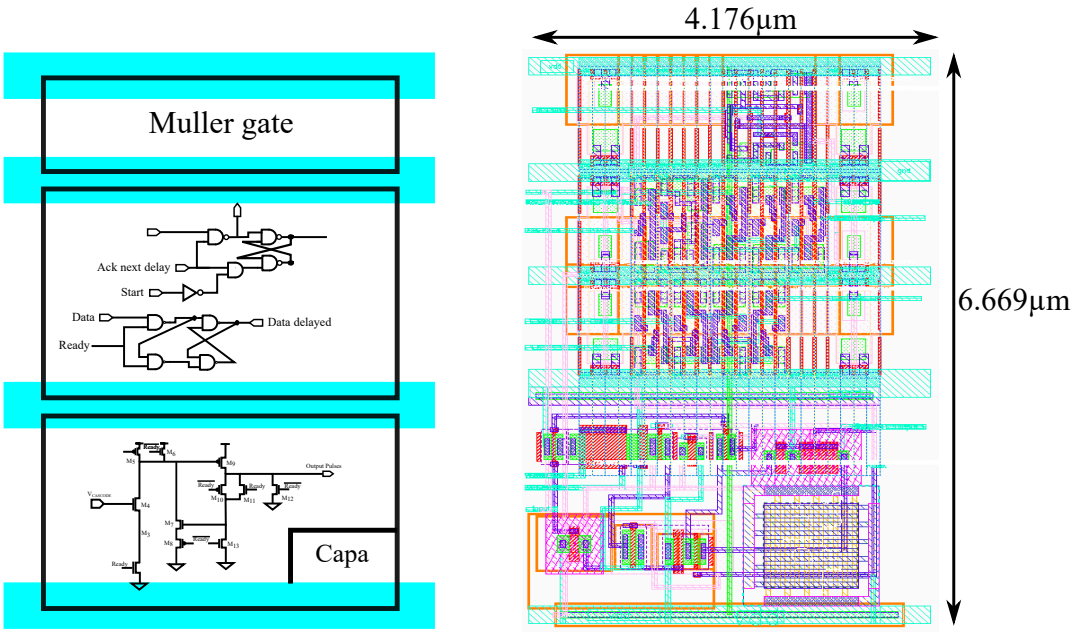


Figure 5.17 Layout structure of the granular delay cell.

discharge control part is positioned at the bottom of the layout. Taking advantage of the cells standardization, we ended up having a very compact layout of 6.669μm by 4.176μm, which results in an area of 28.84μm^2.

The digital layout convention respects a certain orientation of each metal level. The 28nm FD-SOI technology, used for design, contains eight metal levels. The digital convention specifies the use of pair metal levels in a horizontal way and impair ones in a vertical way. The proposed delay layout respects as maximum as possible this convention, as it will be connected to an automated placed and routed digital layout of the processing part. Moreover, the P&R digital flow reserves the second metal level (turquoise metal) for the power grid. This is the reason explaining the use of several horizontal metal 2 levels to alternate between supply and ground. By respecting the P&R recommended metal orientation, the connection between the layouts of the delay line and the digital part will be much easier.

5.5.1.2 Delay tap layout

The delay tap is composed of 48 granular delay cells arranged in a serpentine shape. The layout of the delay tap is depicted in Fig. 5.18. In this serpentine structure, each line of delays is horizontally flipped with respect to the previous one. This helped develop a fully symmetrical layout, either horizontally or vertically. This symmetry allowed the different delay cells to have the same surrounding metallic environment to decrease mismatch effects.

In addition, to not break this symmetry, the biasing stages are placed at the top of the delay cells. This placement will have a slight impact on the current mismatch but will economize the occupied area.

The delay tap respects also as maximum as possible the conventional digital metal orientation, so the power grid is fully conducted through the metal 2 horizontal bars. To sum up, the conventional metallic orientation and the very compact design of the granular delay cell, led to having a very compact delay tap layout with a height of 28.43μm and a width of 50.112μm, resulting in an area equal to 1424.68μm^2.

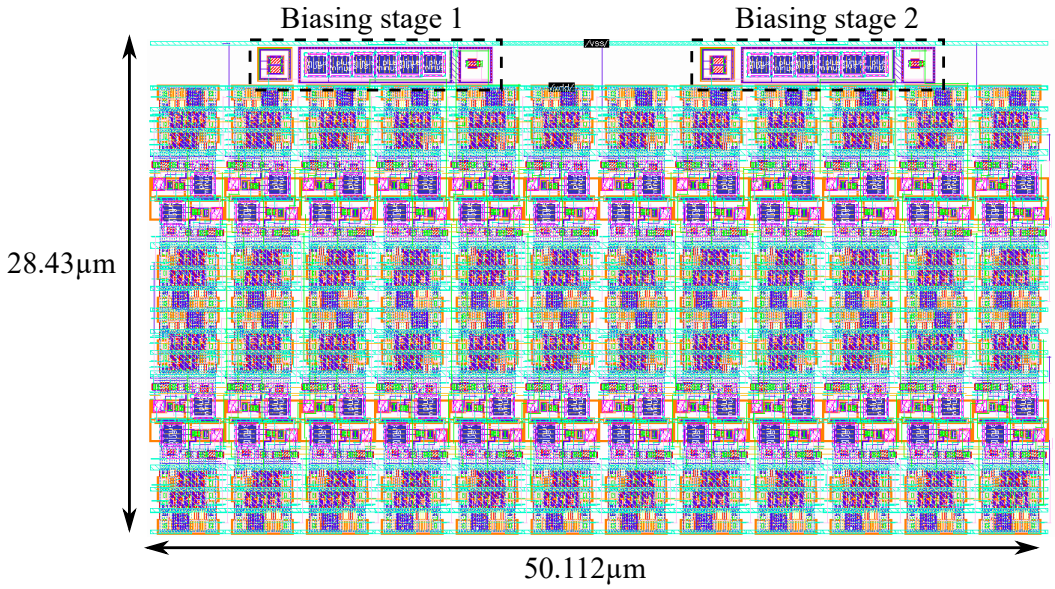


Figure 5.18 Layout of the delay tap element.

5.5.1.3 Delay line layout

The complete delay line contains a set of 16 delay taps, a 32-branch current mirror, and 32 buffers to drive the REQ and DIR signals. The delay line is assembling all the fully custom-designed blocks. The delay line layout structure is shown in Fig. 5.19, where the delay cells are regrouped vertically into two rows of eight delays. Each delay cell is facing another to end up having a symmetrical rectangular shape. The delay mismatch is reduced using this structure because each delay cell will have the same surrounding environment. Then, the 32-branch current mirror is put at the top of the delays. The currents delivered by this block are distributed following a network to be able to reduce the mismatch between current values. In addition, the layout respects as maximum as possible the digital metallic orientations. Thus, the delay line can be easily to the digital part. Finally, the delay line is

physically designed into a high aspect ratio rectangular shape of 231μm by 111μm, resulting in an occupied area of 25641μm^2.

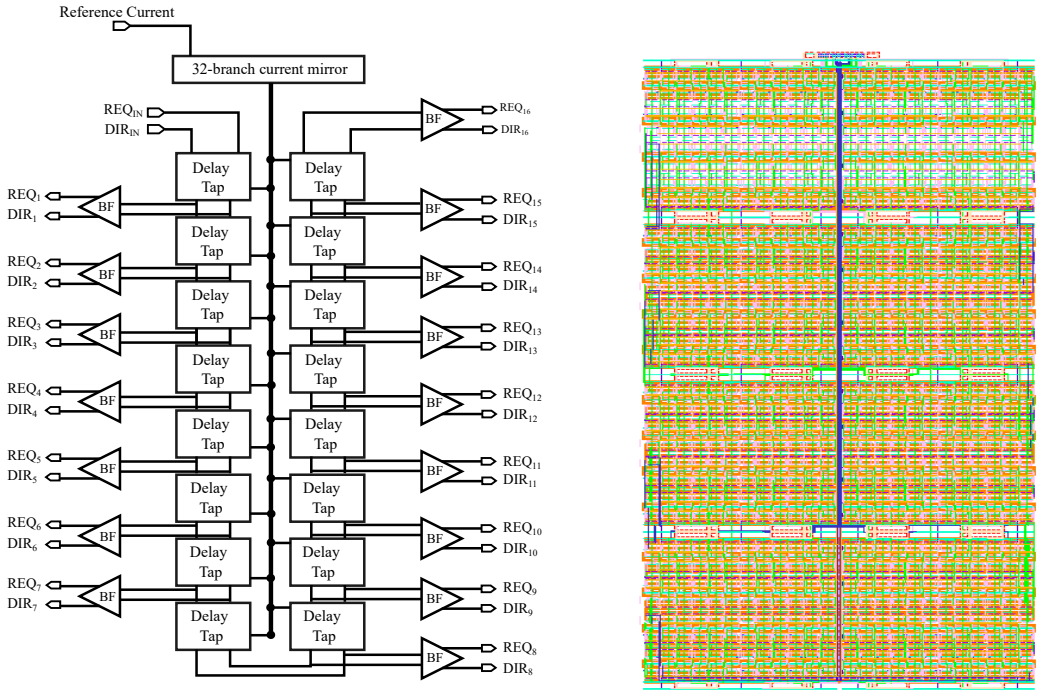


Figure 5.19 Delays line final layout structure.

5.5.2 Semi-custom digital flow

The RTL-designed digital processing part, described in the previous chapter, will be implemented using a semi-custom automatic digital flow. This flow, described in Fig. 5.20, is decomposed into multiple steps, with adding some optimization loops to optimize the design performance. The digital flow starts with RTL design, containing an algorithmic description of the design using hardware description languages (VHDL in our case). After the finalization of the RTL model, the following step will be to generate the gate equivalent netlist. But before, we should specify some constraints to help the synthesis software. These constraints principally concern timing such as the definition of clocks, minimum rise and fall time, load capacitance, and drivers... After the definition of the constraints, they are fed to the synthesis software to generate an equivalent gate netlist. However, based on the quality of synthesis results, the constraints will be optimized and we will enter a constraints-synthesis loop until we achieve the needed specifications. At the issue of this step, we will collect a final gate netlist. Then, we should verify if this netlist has the same behavior as the RTL

design. This is done through a formal verification. This step verifies if the gate netlist and RTL design have the same response for every possible combination at their inputs. If this verification is validated, then we pass to the next step, otherwise, we need to inspect the gate netlist and restart the synthesis flow. After the formality checks, the next steps will concern the physical implementation. The gate netlist is first mapped to the technology libraries

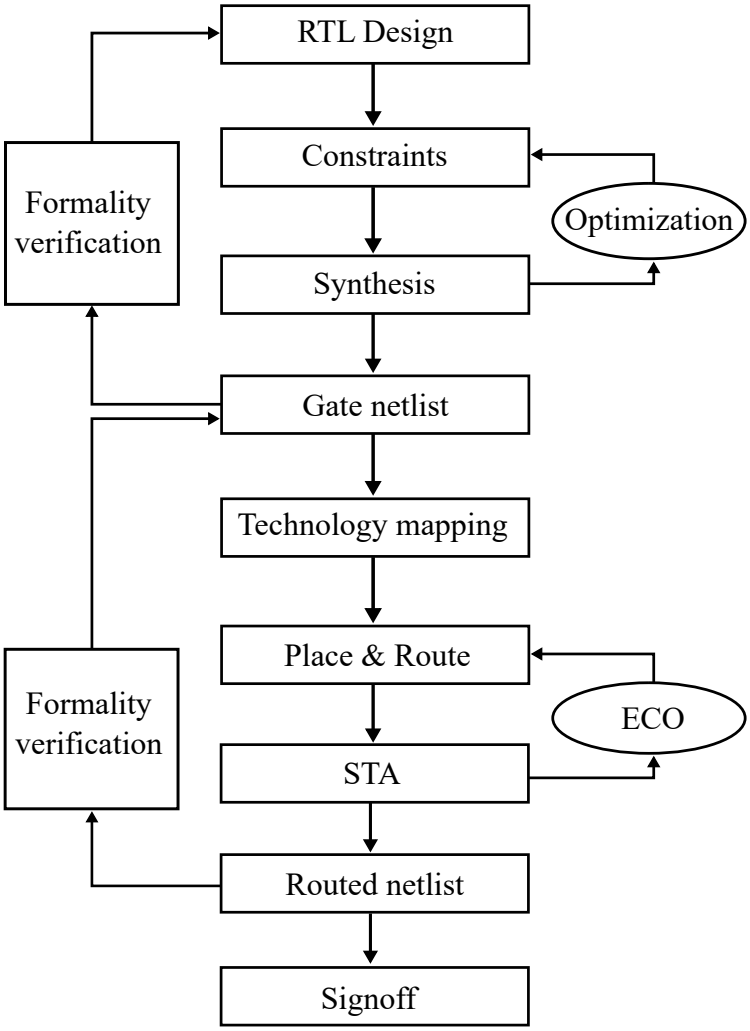


Figure 5.20 Digital design flow.

of the target CMOS technology. Then, the mapped netlist will undergo a Place and Route (P&R) flow to automatically generate the corresponding layout. The P&R flow is composed of multiple steps, that will be detailed after. At the end of the P&R step, we collect reports on static timing analysis to verify if the routed netlist respects the timing constraints imposed at the beginning. If the timing constraints are all respected, we pass to the following flow step. Otherwise, we perform some optimization loops called Engineering Change Order (ECO)

loops. In these loops, we first collect information on the places where the timing violations have occurred, then we try to do some microsurgery by replacing some devices with faster ones or generally with the optimal ones adapted for the types of violation. These ECO loops are performed repetitively until all the timing violations or at least the critical ones. At the end of the ECO optimizations, we collect the final routed netlist, which also undergoes a formality check to verify its functional equivalence with the gate netlist. Finally, the layout of this netlist is sent for fabrication.

5.5.2.1 Synthesis flow

The synthesis step concerns transforming the RTL-designed system into an equivalent electrical system described using the technology standard cells. This is done using *Synopsys*® Design Compiler [12] tool. As the input of this software, we provide the top RTL model and the constraints to be respected when synthesizing the design. These constraints detail, especially timing conditions, load, and drivers. If not provided, Design Compiler will synthesize the RTL model using default constraints, that are not specially compatible with the proposed design. So, the first step before starting the synthesis process is to set the constraints.

The synthesis flow proposed by *Synopsys*® tools is compatible with synchronous systems using regular clock signals. However, the proposed feature extraction system is meant to be clockless, as there is no regular clock globally synchronizing the system. Hence, we have two options in this case, the first one is to implement the system in a fully custom way, which is impossible to do considering the number of logic elements in our system (>150000 based on the FPGA simulations). The second option is to find a way to trick the synthesis software into synthesizing the system, despite its asynchronous nature. This last option seems to be the most logical one to choose.

As detailed in the feature extraction operation, the digital processing bands are triggered each time there is a pulse at the input REQ signal. So, the REQ signal is the one synchronizing the system. However, this signal is not periodic and has sparse random pulses. The digital processing parts are not directly dealing with the input REQ signal, but with the delayed $\text{REQ}_{1,2,...,16}$ signals, delivered by the delay line. Hence, for the synthesis process, we should constrain these signals.

The delayed signals $\text{REQ}_{1,2,...,16}$ are aperiodic and contains pulses randomly distributed in time. But, these pulses cannot be spaced by less than the granular delay (T_{GRAN}), which is equal to 1.302μs. Thus, for the synthesis target, these signals will be considered as clocks with a maximal frequency of $\frac{1}{T_{GRAN}}$ equal to 768.04kHz. In addition, these are not the only signals to constrain. There are the Combined REQ signals, in each band, that represent

the concatenation of the delayed REQ signals. The combined REQ signals are used to synchronize the energy accumulation. To constrain these signals, we should go back to see the oversampling clock to generate the REQ signals.

Fig. 5.21 shows an example of the delaying operation that identifies the minimum time between two pulses at the Combined REQ signal. Considering a delay tap with an input REQ_N and an output REQ_{N+1}. The REQ_N signal contains two pulses that are spaced by 62.5μs + 2x1/8MHz, with 8MHz being the frequency of the oversampling clock used to generate the REQ signal in the *Matlab*® model. Then, the output signal REQ_N is delayed by 62.5μs than REQ_N. In this situation, the first pulse of REQ_{N+1} and the second pulse of REQ_N will be spaced by a time equal to 1/8MHz. Hence, in the combined REQ signal, these two pulses create a pseudo-local clock with a frequency of 8MHz, and this is the maximal frequency to constrain this signal.

The other clock signals to constraint are the pseudo-clock to write in the memory, the read clock to read the content of the memory, the frame clock to create the framing signals, and the SPI interface clock.

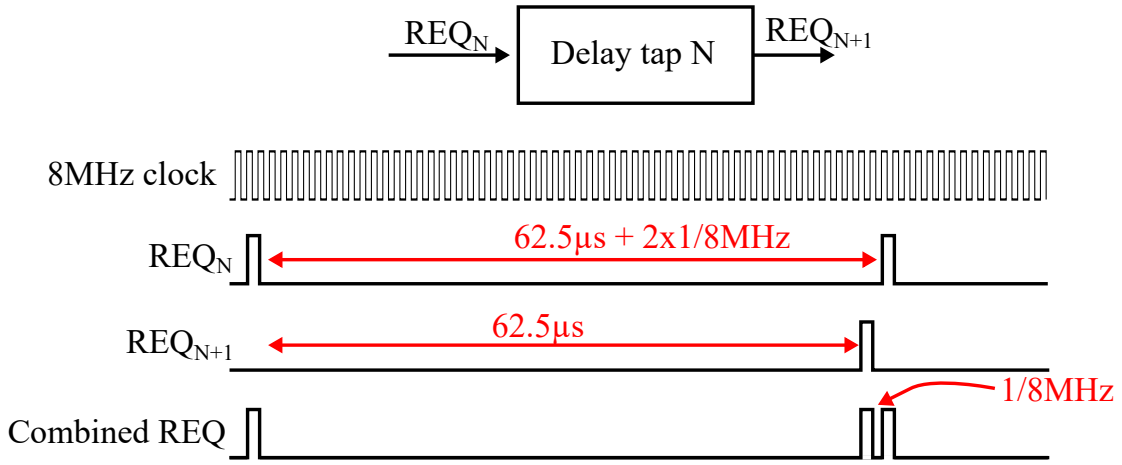


Figure 5.21 Combined REQ signal constraint condition.

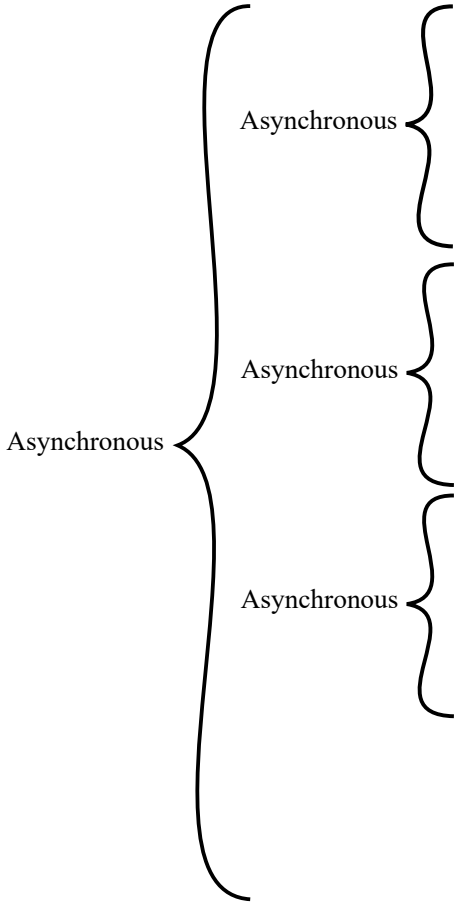
The different signal constraints are regrouped in Table 5.6. The signals $REQ_{1,2,...,16}$, Frame clock, Read clock, and SPI clock are declared as primary clock signals because they are externally injected into the digital processing part. The other signals are described as generated clocks because they are internally generated from the basis of the other primary clock signals. Another important point when setting the constraints is to declare the synchronization relation between the different clock signals. Starting with the delayed $REQ_{1,2,...,16}$, these signals are declared asynchronous between them, because they are not communicating

between each other. By declaring them asynchronous, we avoid any type of clock domain crossing problem.

In the same logic, the Combined REQ and Write clk signals are declared asynchronous. Finally, since we are synthesizing a clockless circuit, we ended up declaring all the constrained signals as asynchronous between them, which is the case in the system operation. setting these constraints helped synthesize the digital processing part without violating any timing condition.

Besides timing, other constraints were set by default, because they were not representing a critical danger for the synthesis process.

Table 5.6 Constraints of the different synchronizing signals.



Signal	Frequency	Nature
REQ_1	770 kHz	Primary clock
REQ_2	770 kHz	Primary clock
⋮	⋮	⋮
REQ_{16}	770 kHz	Primary clock
Combined REQ 1	8MHz	Generated clock
Combined REQ 2	8MHz	Generated clock
⋮	⋮	⋮
Combined REQ 8	8MHz	Generated clock
Write clk 1	100 Hz	Generated clock
Write clk 2	100 Hz	Generated clock
⋮	⋮	⋮
Write clk 8	100 Hz	Generated clock
Frame clock	1kHz	Primary clock
Read clock	128kHz	Primary clock
SPI clock	4MHz	Primary clock

5.5.2.2 Post synthesis results

At the issue of the synthesis, the generated gate netlist is simulated to verify its operation with the new gate imperfections, such as propagation delays. The operation of the system has been verified, through the generation of some spectrograms and comparing them to some *Matlab*® generated spectrograms. The system has demonstrated on the ability to construct the same spectrograms as the reference *Matlab*® model, with a slight difference evaluated with an average error less than 1%. The synthesized gate netlist was not fully simulated to see the recognition results, because it has passed the formality check. So, it will have the same behavioral operation as the RTL model, that was tested and demonstrated on the keyword recognition with an accuracy higher than 90% in the 10 keywords case.

An important result to be extracted at the issue of the synthesis is the first estimation of the power consumption. The proposed feature extraction system is triggered only with the arrival of an event at the input REQ signal, thus the total power will be dominated by the static one. Hence, the first power estimation to be presented here will contain only the leakage power.

Table 5.7 Post synthesis leakage power estimation for different scenarios.

Scenario	(SS, 0.9V, -40°C)	(TT, 0.8V, 25°C)	(FF, 1.15V, 125°C)
Leakage (µW)	0.62	4.19	1344

The leakage power is estimated for three different PVT scenarios : (SS, 0.9V, -40°C), (TT, 0.8V, 25°C), and (FF, 1.15V, 125°C). These scenarios are chosen to show the best and worst cases in terms of power consumption. The results of the power estimation are shown in Table 5.7. Leakage power goes from 620nW in the best case to 1.344mW in the worst case. However, in the typical case and at room temperature, the system presents a low leakage power of 4.19μW. Hence, the digital processing part presents a very low power profile. The leakage power reparation is shown in Fig. 5.22. As expected, the static power is mainly dominated by the self-truncation memory, which occupies 86% of the total leakage power. This is justified by the fact that the self-truncation memory is the biggest block in terms of standard cells, with over 25000 digital flip flops.

5.5.2.3 Place & Route flow

After simulating the synthesized gate netlist, and evaluating the quality of results, the next step will be the physical implementation. This is done through a Place and Route

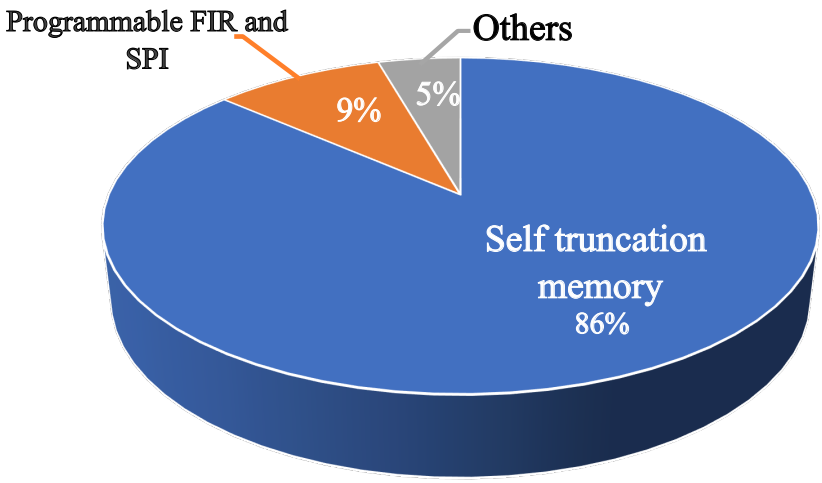


Figure 5.22 Post synthesis leakage power repartition.

(P&R) automatic digital flow. The P&R flow starts by setting the working environment that contains the different variables and files needed to execute the flow, as well as the tool used to perform the layout. In our case, we used *Cadence*® INNOVUS™ implementation system. INNOVUS™ is the most commonly used tool for the physical implementation of digital systems. The advantage of using this tool is the access to all the *Cadence*® support data.

Concerning the P&R flow, we used a digital flow developed internally in STMicroelectronics. This flow has been tested before on multiple systems and has demonstrated its ability to generate the layout of digital systems with optimized area and power consumption. The used P&R flow is depicted in Fig. 5.23. It is composed of 11 steps, most of them automatically done. Some of these steps can be executed several times, depending on the quality of results collected at the end of this step. The P&R flow starts by importing the synthesized gate netlist and the set constraints into the defined working environment. Then, the following step is to draw the floorplan that will border the design layout. This step is the only one that needs to be done manually.

The floorplan needs to be carefully set because it will definitively define the layout area. The chosen floorplan shape is depicted in Fig. 5.24. It has a rectangular shape of 810.4μm by 710μm, with two cut regions to be filled by analog macro blocks. The first cut region at the top center will host the layout of the delay line to easily perform the connection with the digital processing part. The second cut region at the top left is used to host some delay taps to test them individually. After filling the cut regions, the total layout of the digital and analog parts will be contained in a rectangular shape with an area equal to 0.575mm^2. Another point,

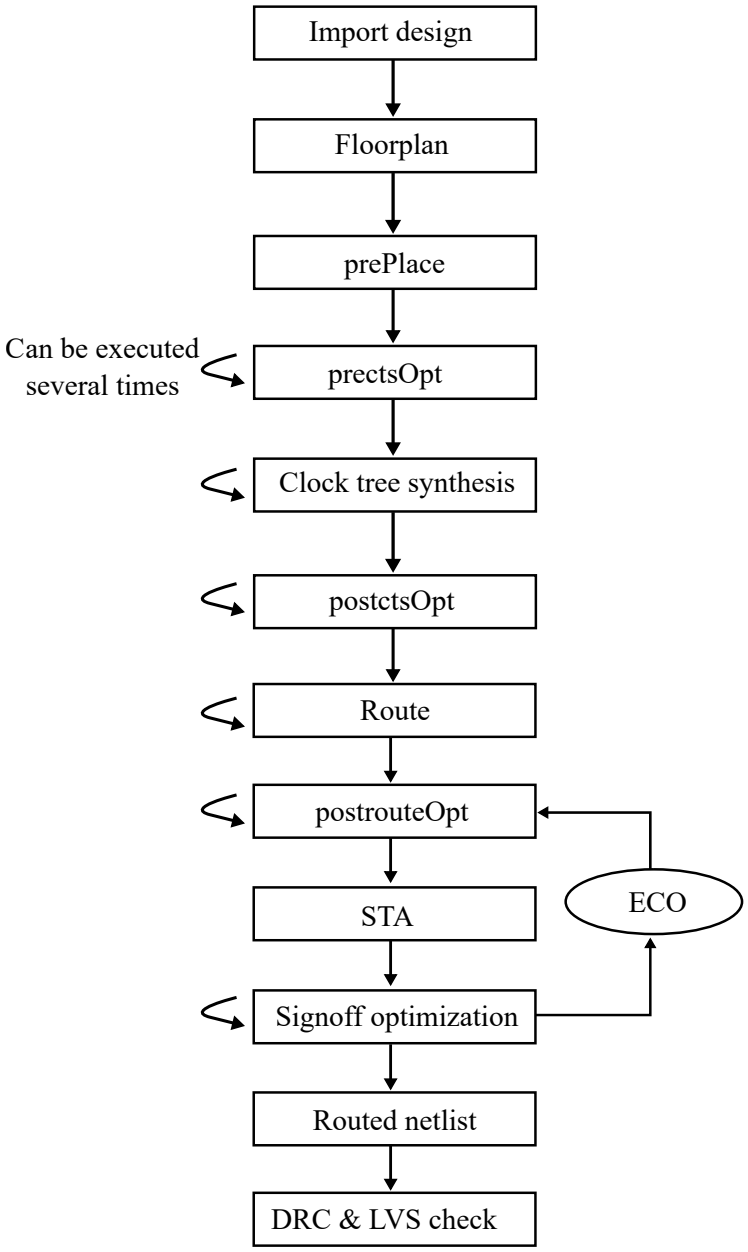


Figure 5.23 Digital Place & Route flow.

during the floorplan step, we pre-place the different input/output pins to be able to easily connect the layout to the pad ring. The pins' positions can be changed depending on the pad ring specifications.

After setting the floorplan, the rest of the flow steps are done automatically. The direct following step is to preplace the different standard cells and optimize their position. After we start the clock tree synthesis process, which is divided into three different steps. It starts

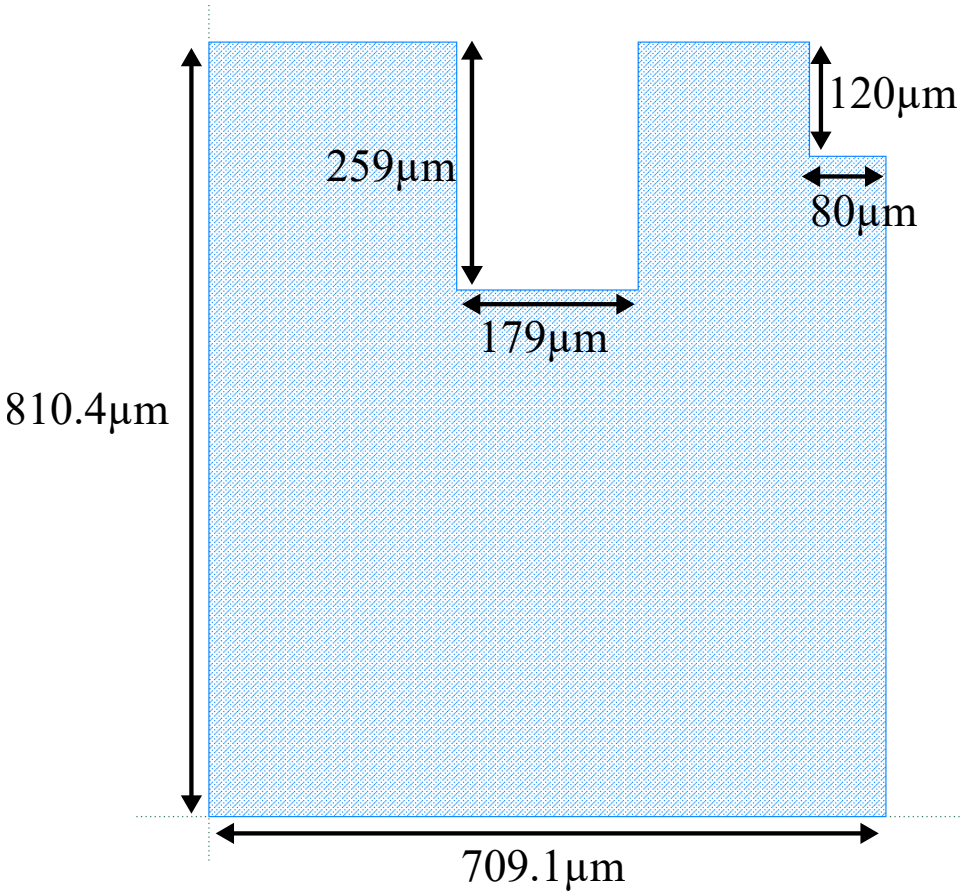


Figure 5.24 Floorplan.

with a pre-optimization to trace the different clock paths, then we begin the placement and synthesis of the different clock buffers defining the clock tree. The following step is the post-clock tree synthesis optimizations to optimize the critical paths and try to respect the set timing constraints. The three steps of the clock tree synthesis can be executed several times until achieving the best timing conditions. This was our case because, at the first synthesis, we ended up having several timing violations due to slow clock buffers. In the beginning, the clock buffers were taken from the PB16 RVT library. However, these cells are the slowest cells offered by the 28nm technology. So, to resolve the different timing violations, we decided to change the cells used for clock tree synthesis from PB16 RVT cells to PB10 RVT ones. After this change, all the critical timing violations were resolved. After finalizing the clock tree synthesis, the following step is to route the circuit. This step concerns building the complete layout, including all the interconnections and the power grid, using the available metal layers (8 metal levels in our case). Then, the generated layout undergoes a static timing analysis to re-check if all the timing conditions were respected after constructing the different

interconnections. If this is not the case, we optimize the routed netlist after suggesting new ECO changes to correct the targeted timing violations. The ECO loops are done iteratively until cleaning the design from any critical timing violation. Finally, we collect a cleanly routed netlist to be integrated into the final chip layout. The final routed layout undergoes two last checks for final validations. The Design Rule Check (DRC) to verify if the generated layout respects the different rules imposed by the technology design kit. The other test is the Layout Versus Schematic (LVS) check to verify if the layout and netlist are completely equivalent.

5.6 Post-layout simulation results

The necessary validation step, after finalizing the layout of the delay and processing part, is to perform post-layout simulations (PLS). These simulations will take into account the different imperfections generated by the layout such as parasitic capacitance, propagation delays, and interconnection resistances... The analog delays and the digital processing part are firstly simulated independently and then combined to test the complete feature extraction functionality.

5.6.1 Analog delays simulation

The delay taps are post-layout simulated for two reasons: the first one is to validate the operation of the delay chain and study the delay variability. The second reason is to perform a power analysis to evaluate the power consumption of this block.

Table 5.8 contains the information about the delay values of each tap inside of the delay line, when it is biased by a control current equal to 18.93nA. We can see that all the delays have nearly the same value, with a maximum mismatch of 1.048μs. This mismatch represents a 1.67% variation of the mean value. This low mismatch value is achieved thanks to the minimization of the variation at the level of the granular delay, which was equal to 23% (Montecarlo simulation). Since we have 48 granular delays per each delay tap, the delay tap mismatch will be less than $\frac{23\%}{sqrt(48)} = 4.7\%$. This theoretical value is in the same range as the simulated delay tap value.

In addition, the power consumption of the delay line was measured in the typical case and it resulted in a total of 279.7nW. The power repartition is shown in Table 5.9. The power is dominated by the delay taps with 58.5% of the total power, then the buffers and the current mirror are approximately sharing the rest of the power consumption. These values

Table 5.8 Variability of the delay taps values.

Delay	Delay 1	Delay 2	Delay 3	Delay 4	Delay 5	Delay 6	Delay 7	Delay 8
Value (µs)	62.478	62.848	62.492	62.497	62.44	62.46	62.429	62.5

Delay	Delay 9	Delay 10	Delay 11	Delay 12	Delay 13	Delay 14	Delay 15	Delay 16
Value (µs)	62.47	62.389	62.4	62.425	62.329	62.361	62.336	61.8

are simulated for a rate of 4000 pulse/s at the REQ signal. This is the average rate when decoding an audio signal from the GSCD database.

Table 5.9 Power repartition inside of the delay line

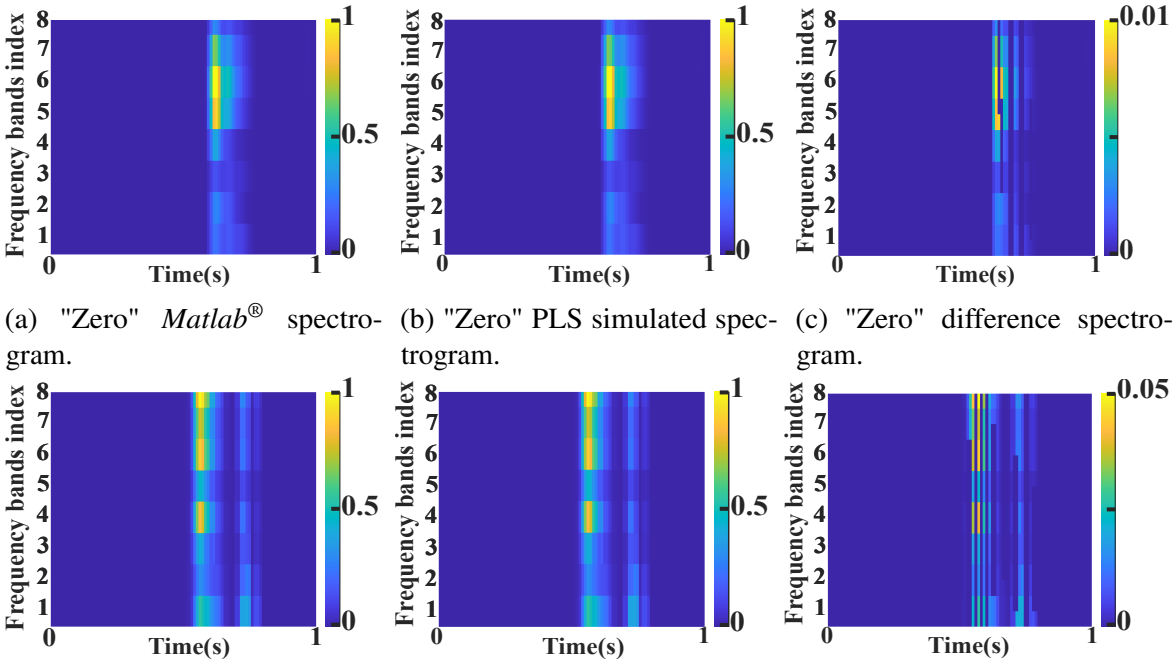
Block name	Delay taps	Current mirror	Buffers	Total
Power (nW)	163.7	78.07	85.63	279.7

The power consumption of these asynchronous analog delays is 22 times lower than the counter-based delay cells used in the FPGA implementation. Hence, the choice of using these delays, to improve the power efficiency, is validated.

5.6.2 Back-annotate simulations

The generated routed layout undergoes a post-layout parasitic parameters extraction to perform back-annotate simulations. These simulations contain information about the propagation delay, critical path, parasitic capacitance, and all the possible hardware imperfections. Through these simulations, we verify if the physically implemented system still performs the needed function, while taking into account all the physical non-idealities. These simulations are done in a completely digital environment using *Cadence*® Xcelium tool. Unfortunately, in this environment, we could not integrate the designed delay taps. Hence, the digital processing bands are tested with a behaviorally modeled delay line.

In a functional perspective, the digital processing part was tested on multiple keywords to test its capacity to generate spectrograms with the same precision as the *Matlab*® reference model. Fig. 5.25 shows an example of the spectrogram generation using the *Matlab*®



(a) "Zero" *Matlab*® spectrogram.
(b) "Zero" PLS simulated spectrogram.
(c) "Zero" difference spectrogram.
(d) "Nine" *Matlab*® spectrogram.
(e) "Nine" PLS simulated spectrogram.
(f) "Nine" difference spectrogram.

Figure 5.25 Normalized spectrograms generated using *Matlab*® and the system physical implementations for "Zero" and "Nine" keywords, respectively ((a), (b) ; (d), (e)). Spectrograms presenting the calculated difference ((c), (f)).

reference model and the physically implemented feature extractor. The spectrograms are generated for two keywords "Zero" and "Nine". Through these back-annotate simulations, the system has demonstrated the capacity to construct the spectrogram with the same precision as the reference model. To deepen the analysis, we sketched a third spectrogram emphasizing the difference. In this spectrogram, we visualize some discrepancies appearing on a 100X finer scale. These differences were quantitatively evaluated by an average relative error equal to 0.8%. The conclusion is that the routed netlist, taking into account all the imperfections, generated the same spectrograms as the reference model. Hence, the functional operation is validated.

Scenario	(SS, 0.8V, -40°C)	(TT, 0.8V, 25°C,)	(FF, 1.15V, 125°C)
Leakage	661nW	5.37µW	4.57mW
Total power	3µW	7.46µW	4.58mW

Figure 5.26 Power consumption of the digital processing bands in the three set scenarios.

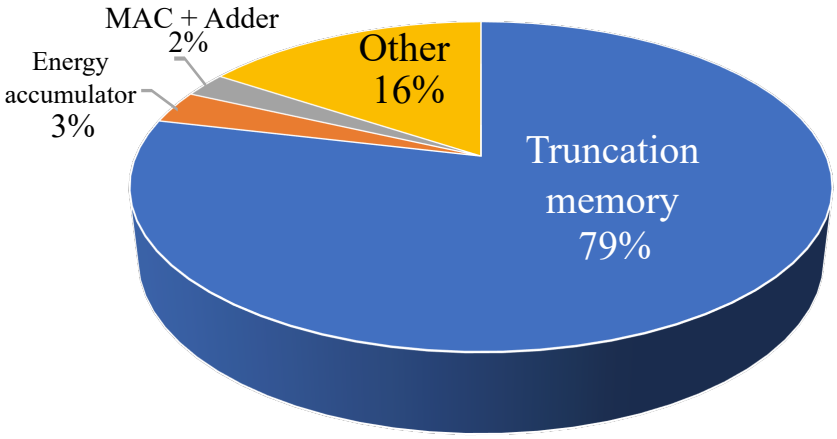


Figure 5.27 Power reparation after back-annotate simulations.

The second objective of these back-annotate simulations is to estimate the power consumption in different scenarios to study the best and worst cases. The power is estimated for three different scenarios. the first one concerns the slow slow corner and a temperature of -40°C and a supply voltage of 0.8V. The second scenario considers the typical typical corner, a temperature equal to 25°C and a supply voltage equal to 0.8V. The final scenario considers the fast fast corner, a temperature equal to 125°C, and a supply voltage equal to 1.15V. The power consumption values of the digital processing bands in the three mentioned scenarios are regrouped in Table 5.26. This power consumption is an average result of multiple simulations, with changing the input keyword for each simulation. The power values range from 3μW for the best case up to 4.58mw in the worst case, with a typical power consumption of 7.46μW. These values do not take into account the body bias effect, because the available technology RVT standard cell libraries did not contain any body bias scenarios.

The power consumption is mainly dominated by leakage power, except for the SS scenarios. This was expected because in this scenario the leakage reaches its minimal values and goes under the active power level.

Fig. 5.27 shows the power repartition between the different building blocks inside of the digital processing bands. As expected, the truncation memory is mainly dominating the power consumption with 79% of the total power.

5.7 Chip finishing

The core design and the pad ring was assembled to form the final chip. The deails of chip assembling is found in Appendix B.

To finalize the chip, the design core (delay line + digital processing bands) is connected to

the pad ring. Then, we add decoupling capacitance to attenuate the effect of supply noises. Table 5.28 shows the values of the decoupling capacitance values for all the supply voltages used in the design core.

Supply voltage	VDD DELAYS	VDD DELAYS_TEST	VDD_CORE	VBB_P_CORE	VBB_N_CORE
Decoupling capacitance (pF)	17.5	1.2	20.7	2.3	2.8

Figure 5.28 Decoupling capacitance value for the different supply voltages.

The following step is to add the sealring and resolve the density problems by regularizing the metals and polysilicon density values to respect the design rules. A seal ring structure is disclosed for protecting a core circuit region of an integrated circuit chip. The seal ring structure includes a metallization layer, having a bridge sublevel and a plug sublevel. An upper-level bridge is formed on the bridge sublevel at a predetermined location between a peripheral edge of the integrated circuit chip and the core circuit region. A lower-level bridge is formed on the plug sublevel in substantial alignment with the upper-level bridge, wherein the lower-level bridge has a width substantially the same as that of the upper-level bridge [14]. The sealring delimiting the chip disclosed surface is shown in Fig. 5.29.
The density correction is done through the tiling step. During tiling, the design has been meshed and each basic meshing block is filled with metal and polysilicon pieces until achieving the required density level. At the issue of this step, we collect a DRC clean design. The final chip design is depicted in Fig. 5.29. This design is extracted in the form of a binary Graphical Design System (GDS) file, and this is the file sent to the fabrication team.

5.8 Conclusion

This chapter has detailed all the steps followed to design an on-chip feature extractor system. The circuit design was done in two different environments. The first one is a full custom analog environment adopted to design the delay line, to design low-power asynchronous delay taps. These delay taps are composed of a chain of 48 thyristor-based granular delay cells. In these latter cells, the delay is created through a current-controlled discharge capacitance, and the synchronization between the delay units is ensured through a handshaking protocol. These delay units were designed using 28nm 8-metal levels FD-SOI MOS transistors and standard cells. Thanks to the used CMOS technology and the full-custom asynchronous implementation, the simulated delay line operates with a very low

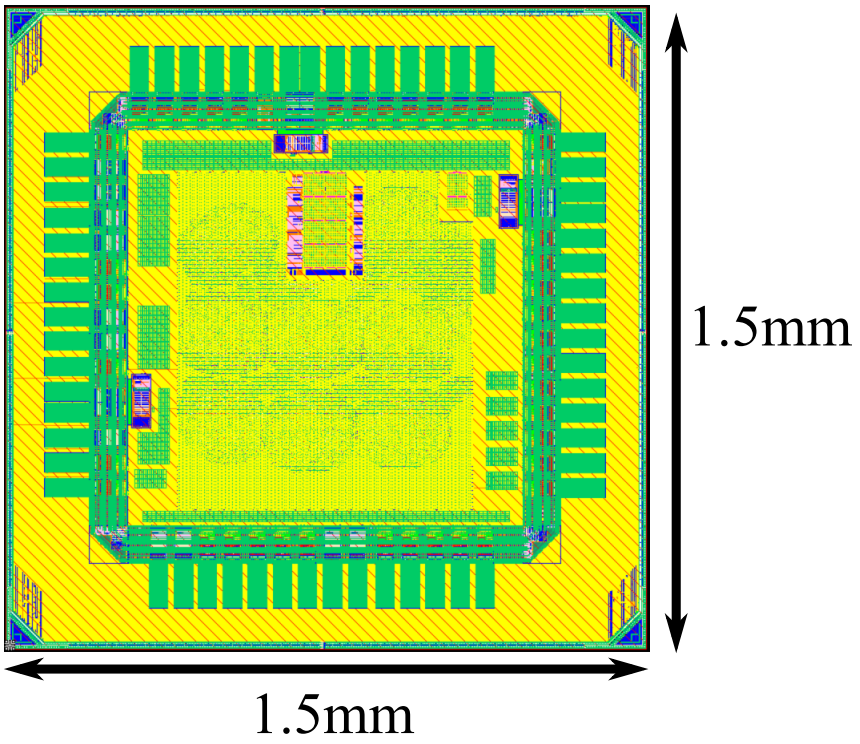


Figure 5.29 Chip final layout.

power consumption of 279.7nW, which is 22X lower than the synchronous delay line used for the FPGA implementation.

For the digital processing bands, the synthesis and Place and Route flows were customized to be adapted to the asynchronous behavior of our system. This customization led to an automatically generated physical implementation of the feature extraction system, designed using PB16 and PB10 standard cells. This system has been tested on the Google Speech Command dataset and has demonstrated the ability to construct the same spectrograms as the reference *Matlab*® model, with an average error of 0.8%. This was achieved while consuming around 7.5μW, without considering the body bias effect that will decrease more and more the power consumption.

The digital processing part and the delay line were connected to a custom-made IO pad ring to complete the chip design. Then, the chip was finalized, resulting in a total area of $2.25mm^2$.

Bibliography

[1] Sylvain Clerc, Thierry Di Gilio, and Andreia Cathelin, eds. The Fourth Terminal: Benefits of Body-Biasing Techniques for FDSOI Circuits and Systems. Integrated Circuits and Systems. Cham: Springer International Publishing, 2020. isbn: 978-3-030-39495-0 978-3-030-39496-7. doi: 10.1007/978-3-030-39496-7. url: http://link.springer.com/10.1007/978-3-030-39496-7.3.

[2] Venkatesh N Shivaram and Choudhury Bedanta. Best-in-Class Standard-Cell Libraries for High-Performance, Low-Power, and High-Density SoC Design in 28nm FD-SOI Technology, STMicroelectronics Technology Market, White Paper. Sept. 2015.

[3] I. Sourikopoulos, "Continuous-time digital processing techniques applied to channel equalization for low-power millimiter wave communications," Université de Lille, 2015.

[4] Rabaey, Jan M., Anantha P. Chandrakasan, and Borivoje Nikolić. Digital Integrated Circuits: A Design Perspective. 2. ed. Prentice Hall Electronics and VLSI Series. Upper Saddle River, NJ: Prentice Hall, 2003.

[5] Maymandi-Nejad, M., and M. Sachdev. "A Monotonic Digitally Controlled Delay Element." IEEE Journal of Solid-State Circuits 40, no. 11 (November 2005): 2212–19. doi:10.1109/JSSC.2005.857370.

[6] Jung, L.H., N. Shany, A. Emperle, T. Lehmann, P. Byrnes-Preston, N.H. Lovell, and G.J. Suaning. "Design of Safe Two-Wire Interface-Driven Chip-Scale Neurostimulator for Visual Prosthesis." IEEE Journal of Solid-State Circuits 48, no. 9 (September 2013): 2217–29. doi:10.1109/JSSC.2013.2264136.

[7] Maymandi-Nejad, Mohammad, and Manoj Sachdev. "A Digitally Programmable Delay Element: Design and Analysis." Very Large Scale Integration (VLSI) Systems, IEEE Transactions on 11, no. 5 (October 2003): 871–78. doi:10.1109/TVLSI.2003.810787.

[8] Gyudong Kim, Min-Kyu Kim, Byoung-Soo Chang and Wonchan Kim, "A low-voltage, low-power CMOS delay element," in IEEE Journal of Solid-State Circuits, vol. 31, no. 7, pp. 966-971, July 1996, doi: 10.1109/4.508210.

[9] A. González, A. Frappé, B. Larras, A. Kaiser and P. Cathelin, "A Wide Tuning Range Delay Element for Event-Driven Processing of Low-Frequency Signals in 28-nm FD-SOI CMOS," in IEEE Solid-State Circuits Letters, vol. 3, pp. 198-201, 2020, doi: 10.1109/LSSC.2020.3010877.

[10] Angel de Dios Gonzalez Santos. Ultra-low power continuous time digital signal processing circuits in 28nm FDSOI technology for audio applications. Micro and nanotechnologies/Microelectronics. Université de Lille, 2020. English. ⟨NNT : 2020LILUI047⟩. ⟨tel-03582251⟩

[11] J. Sparso and S. Furber, Principles of Asynchronous Circuit Design: A Systems Perspective. Springer, 2013.

[12] https://www.synopsys.com/content/dam/synopsys/implementation&signoff/datasheets/dc-graphical-ds.pdf

[13] https://www.cadence.com/en_US/home/tools/digital-design-and-signoff/soc-implementation-and-floorplanning/innovus-implementation-system.html

[14] Yao CH, Huang TC, Chi KS, Jeng CC, Liang MS, Wan WK, Hsia CC, inventors; Taiwan Semiconductor Manufacturing Co TSMC Ltd, assignee. Seal ring structure for integrated circuit chips. United States patent US 7,777,338. 2010 Aug 17.

Chapter 6

28nm FD-SOI System-on-Chip measurements

6.1 Introduction

The previous chapter have detailed and verified the feature extraction operation through simulations of the software model and the circuit implementation. This verification was done only using simulation tools. The obtained simulation results are sufficient to trust the circuit design and send it to fabrication, but not sufficient enough to confirm that the chip hardware will work properly. Hence, the fabricated chip needs to be electrically tested to verify the real operation and extract the real circuit performances in terms of accuracy and power consumption.

This chapter will present the test plan, with its different steps, to verify the electrical behavior of the fabricated chip. Moreover, the different measurement results will be presented to validate the hardware operation of the feature extraction system-on-chip. The measured on-chip circuit performances will then be compared to the latest state-of-the-art keyword spotting systems, to demonstrate the competitiveness of our solution and validate the design choices taken during the implementation of the feature extractor.

6.2 Fabricated System-on-Chip "Capybara"

The fabricated "Capybara" System-on-Chip (SoC) is depicted in Fig. 6.1. The border lines are apparent between the digital processing bands and the analog delay line. Inside the digital part, the processing bands are identified as eight zones with a strong concentration of

logic elements. The orientation of the chip is identified using the position of the "ST" logo and the "Capybara" chip name at the the bottom left of the chip.

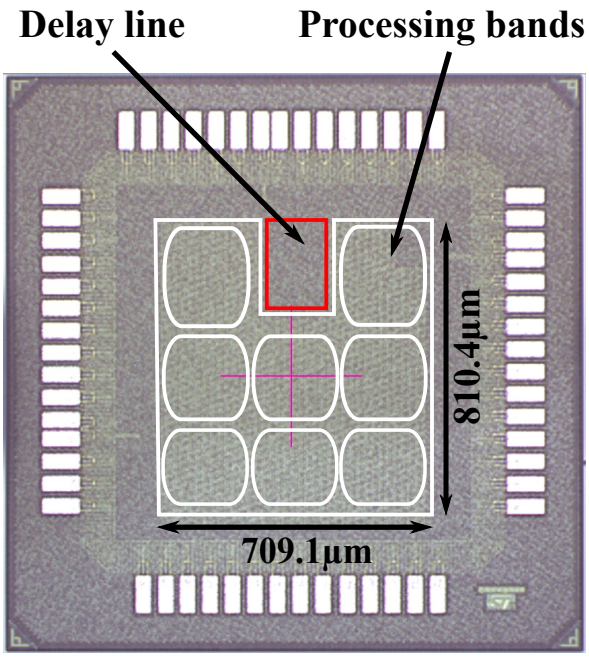


Figure 6.1 Chip micrograph.

6.3 Packaging

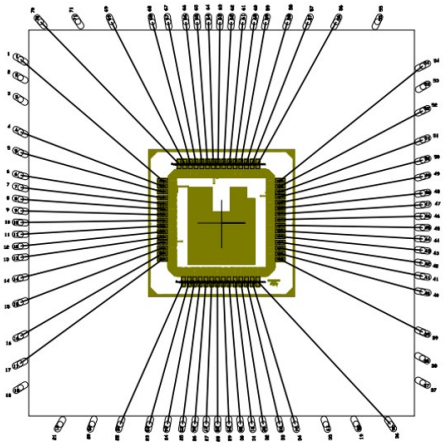
To prepare the test, the SoC needs to be packaged using a package compatible with the chip nature and the number of specified IOs in the designed pad ring. The feature extractor core design was encountered by a pad ring composed of 54 GPIOs. Hence, the package needs to have at least 54 pins.

The choice of the package depends not only on the number of pads but also on its robustness and compatibility with available test sockets. To avoid the dependence on the availability of a dedicated socket, it is preferable to use a package that can be tested either with or without a socket. The Land Grid Array (LGA) packages are molded array packages, which have external pins on the bottom of the package arranged in columns and rows [1]. The advantage of this kind of package is that it can be either connected through a socket or it can be directly soldered on the test PCB.

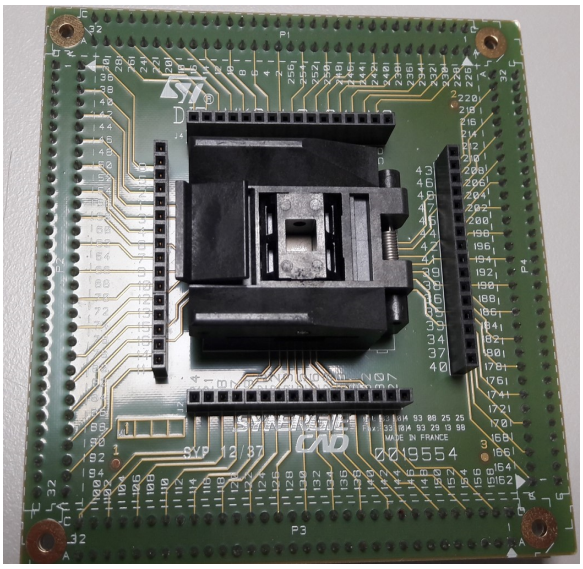
To test our circuit, we used a very very thin profile fine pitch LGA (WFLGA) package [1]. For our test case, we will specifically use a 9x9x0.8mm WFLGA package composed of

64 pins spaced with a pitch of 0.5mm. The die attach area inside this package has a 4.065 x4.056mm dimension, which is completely sufficient to host the 1.5x1.5mm fabricated chip.

The chip needs then to be wire-bonded to the specified package. Fig. 6.2a shows the bonding diagram, that will be respected to memorize the order of our IOs concerning the package pins.



(a) Wire bonding diagram.



(b) Socket board used for test.

6.4 Test plan

6.4.1 Socket choice

The used 64-pin WFLGA package is compatible with a QFN11T064 socket [2]. The test teams in STMicroelectronics have already dealt with these types of packages and developed a socket board, shown in Fig. 6.2b to make it reusable for every circuit having the same package. This socket board was borrowed from the test teams and will be used to test our circuit.

6.4.2 Test setup

The test setup is shown in Fig. 6.3. The test chip is connected to multiple supply voltages, one for the delays, one for the digital processing bands, and the other to supply the pad ring and generate the body bias. Besides, a control computer is used to generate the REQ and DIR signals, using a *Matlab*® coded 6-bit LC-ADC, and transmits them to the test chip through an

arbitrary waveform generator. Moreover, the same computer controls an FPGA to configure the different registers to program the coefficient bank and set the number of lines to be read. The FPGA is used as the master SPI interface in this case, but it is used as a slave SPI to read the energy values. Finally, a logic analyzer is used to visualize the different outputs and help debug the operation of the circuit.

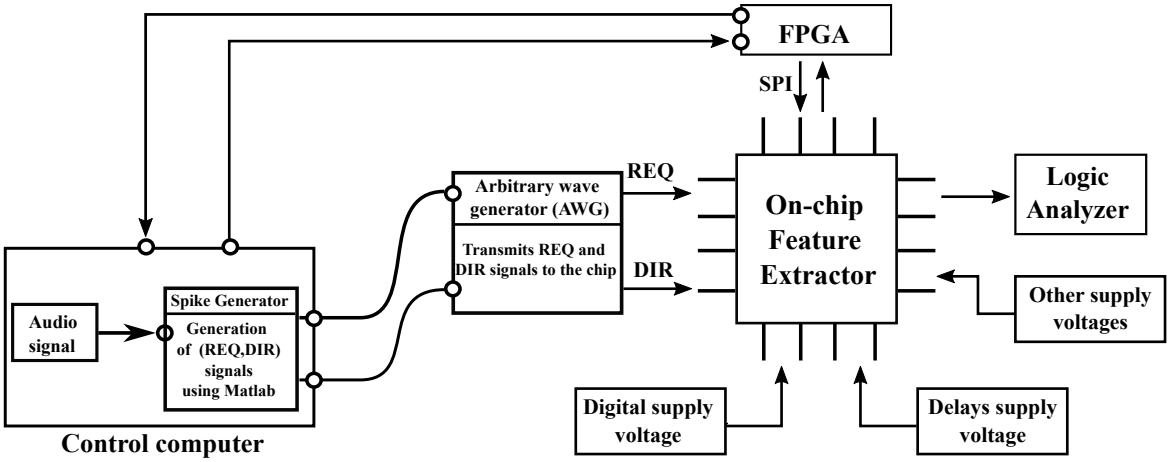


Figure 6.3 Setup used to test the feature extractor SoC.

6.4.3 Materials used for test

The different devices used to construct the previously described test setup are shown in Fig. 6.4.
These devices are listed as follows :

- **Keysight CX3322A device current waveform analyzer (a):** Used to analyze the output current and measure the power consumption.
- **RIGOL MSO7024 digital oscilloscope (b):** Logic analyzer to visualize the different outputs of the circuit.
- **RIGOL DP831 triple output programmable DC power supply (c):** Supply source to generate the supply voltages.
- **HEWLETT PACKARD E3631A triple output DC power supply (d):** Supply source to generate the supply voltages.
- **KEITHLEY 2450 sourceMeter (e):** Precise current source to generate the bias currents for the delays.

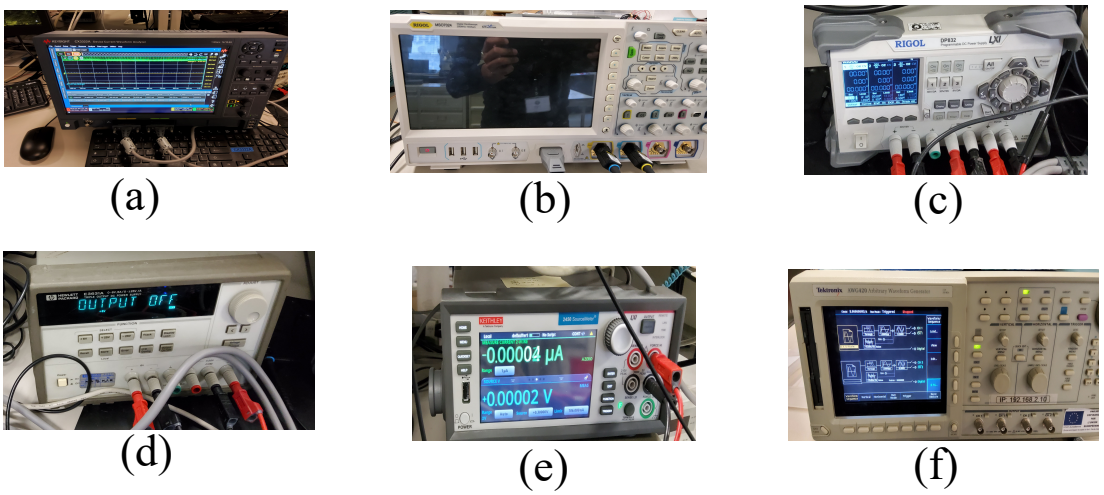


Figure 6.4 Used measurement devices.

- **Tektronix AWG420 arbitrary wave generator(f):** AWG used to generate the REQ and DIR signals.

Besides these devices, we used the *Intel*® DE2-115 Cyclone IV FPGA board presented in Chapter 3. These are the principal used test devices. Other devices were used temporarily, that are not mentioned here.

6.5 Tests and measurement results

For each test, the configuration of the used pins is presented in appendix B.

6.5.1 Chip-alive test

The chip was tested to check if it reacts properly to the application of supply voltages. The results of this test are shown in appendix B.

6.5.2 SPI configuration test

We need to verify if the chip can be configured correctly. The configuration, in our case, is done through an SPI interface responsible for programming the coefficient bank.

This test will verify if the coefficient bank is programmed correctly. In each processing band, we have a CT FIR filter composed of 16 MAC units, where each MAC unit has a coefficient as an input. Thanks to the symmetric FIR filter architecture, we need only

8 coefficients for the complete filter. Thus, the programmable coefficient bank, in each processing band, contains 8 programmable 6-bit coefficients.

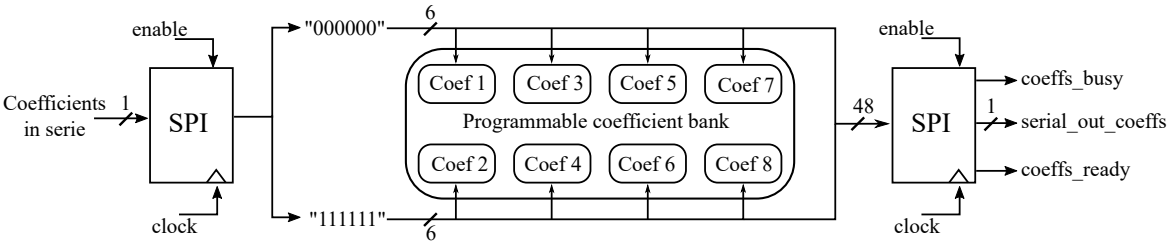


Figure 6.5 Setup to test the coefficient bank programming.

The setup, shown in Fig. 6.5, is used to test the coefficient bank. Using the FPGA-programmable SPI interface, the coefficients are programmed alternatively to "000000" and "111111", as described in Fig. 6.5. Then, we use another SPI interface to check if the coefficients were programmed correctly. The signals to check are serial_out_coeffs, coeffs_busy and coeffs_ready indicating the serialized coefficients, the programming duration and the end of the coefficient programming, respectively. For this test, we use the configuration detailed in Table B.2 shown in appendix B.

The result of this test is shown in Fig. 6.6. The signal serial_out_coeffs is serializing the programmable coefficient values during the high state phase of the coeffs_busy signal. As it is seen in Fig. 6.6, the serial_out_coeffs signal is toggling between 1 and 0 eight times proving that the coefficients are equal to "000000" and "111111", alternatively. At the end of each programming cycle, a pulse is generated in the coeffs_ready signal.

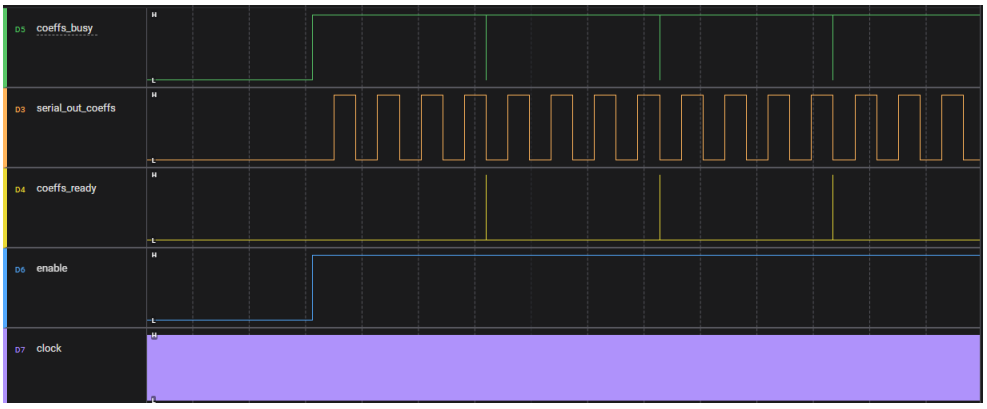


Figure 6.6 Coefficient bank programmed alternatively by "000000" and "111111" values.

The same test has been done to test the programming of the number of memory lines to be read.

6.5.3 Synchronization signals test

6.5.3.1 Framing signals

This test will verify if the pulse width modulator generates correctly the framing signals. For this test, the digital processing bands are activated and the framing clock (1kHz) is injected into the circuit. More specifically, the used pins configuration during this test is detailed in Table B.3 shown in appendix B.

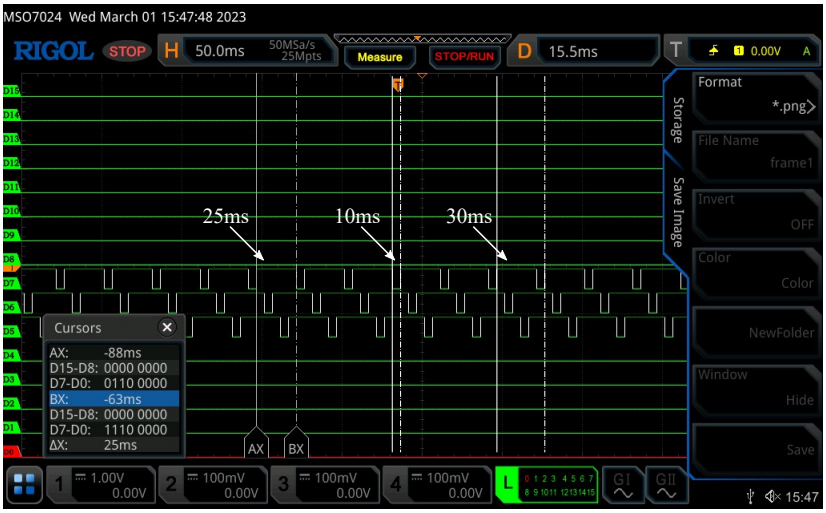


Figure 6.7 framing signals.

The framing signals are visualized using the logic analyzer in Fig. 6.7. These framing signals have a cycle of 30ms, a high state duration equal to 25ms, and a shift of 10ms from one to another. These results correspond exactly to the simulation ones. Hence, the framing signals generation is validated.

6.5.3.2 Memory reading synchronization signals

This test concerns the verification of the memory reading process in the 8 digital processing bands. Through this test, we will check if the different synchronization signals (read flag, read busy 1,...,8, synchronized read trigger) are correctly generated.

For this test, we keep the same pins configuration described in Table B.3 shown in appendix B, and we activate the read clock signal, which will have a frequency equal to 128kHz.

The reading process is visualized using the logic analyzer in Fig. 6.8. After the arrival of a pulse at the external read trigger signal, the memory reading does not start immediately,

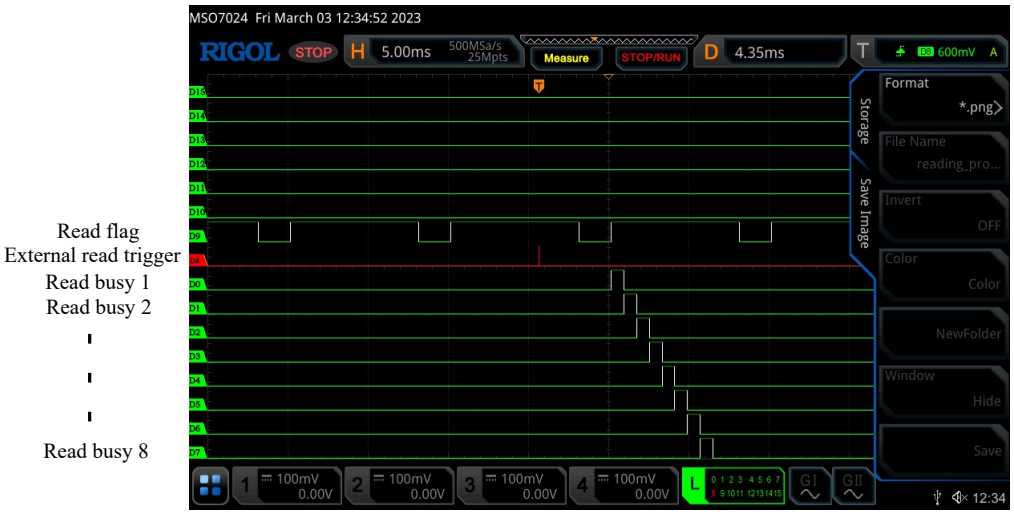


Figure 6.8 Memory reading process in all the digital processing bands.

but it waits until the following rising edge of the read flag signal. Then, the memories in the 8 processing bands are read chronologically starting from the first one to the eightieth one. This is illustrated by the read busy signals, that rise to a high state when the corresponding memory is being read. Moreover, we can see that all the memories are read during the high state period of the read flag. Hence, the memory reading process corresponds exactly to the simulated one.

6.5.4 Delay line test

6.5.4.1 Delay operation

The delay line is tested to verify if the pulses are properly delayed and if all the delay taps operate correctly. During this test, we will visualize the Combined REQ signal 1 and 8, which represent the concatenated delayed REQ signals at the output of each delay tap.

For this test, both the delay line and digital processing bands are activated. Then, we inject a pulse at the REQ signal and we show the Combined REQ signals using the logic analyzer. Moreover, for the other pins, we use the configuration shown in Table B.4 in appendix B.

The coefficient bank is programmed to the filtering configuration, which means that the filter in the first band has an order of 15, and the filter in the 8th band has an order equal to 3. This means that the combined REQ signal will be defined by the following equations.

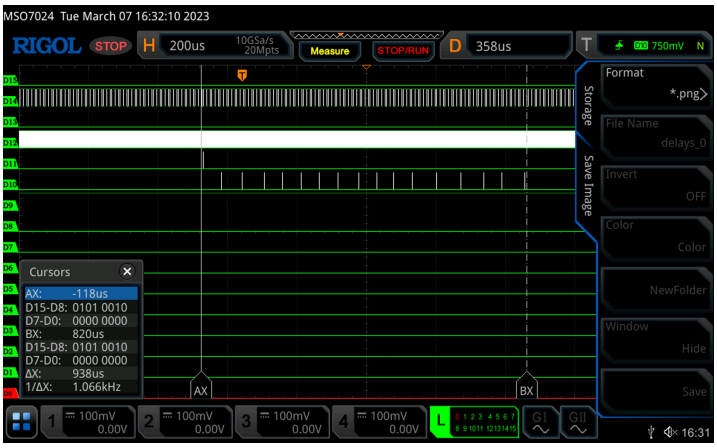


Figure 6.9 Combined REQ signal from the 1st processing band.

$$\begin{aligned} &\textit{Combined REQ signal } 1 = \textit{REQ}1 \textit{ or } \textit{REQ}2 \textit{ or } ... \textit{ or } \textit{REQ}15 \\ &\textit{Combined REQ signal } 8 = \textit{REQ}1 \textit{ or } \textit{REQ}2 \textit{ or } \textit{REQ}3 \textit{ or } \textit{REQ}4 \end{aligned} \tag{6.1}$$

The D11 and D10 signals in Fig. 6.9 correspond to the input REQ signal and the Combined REQ signal 1, respectively. As expected, The Combined REQ signal 1 contains 16 delayed pulses, which means that all the delay taps worked properly. Moreover, the complete delay duration through the whole delay line is equal to 938μs. This means that each delay tap is equal in average to 58.625μs. This value is lower than the expected 62.5μs. Thus, the polarization current value will be adjusted to set the delay taps to the needed value.

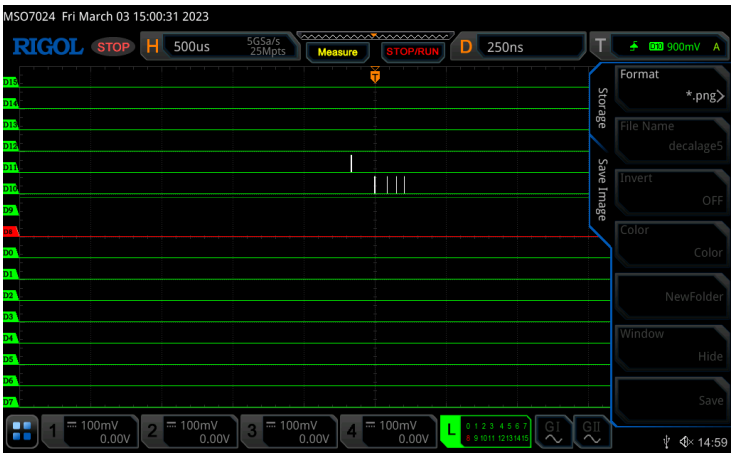
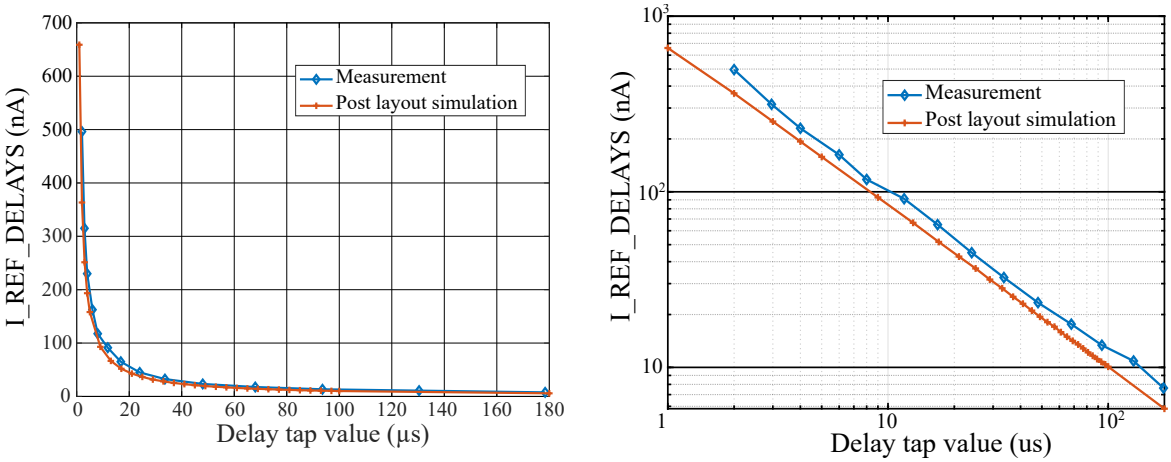


Figure 6.10 Combined REQ signal from the 8th processing band.

As a result of the same test, we can see the Combined REQ signal 8 in Fig. 6.10. This signal is containing 4 delayed pulses, proving that the FIR filter in the eightieth processing band is configured to have an order equal to 3.

This test does not only validate the operation of the delay line but also validates the correctness of the connection between the analog delay lines and the digital processing bands.

6.5.4.2 Delay dependence on biasing current



(a) Variation in linear scale.

(b) Variation in logarithmic scale.

Figure 6.11 Delay tap value variation as a function of the biasing current in measurement and post-layout simulation.

The objective of this test is to describe the dependence of the delay tap value on the biasing current. For this test, we kept the same pins configuration as the test before, except that the I_REF_DELAYS value is swept between 0 and 660nA.

As shown in Fig. 6.11, the delay is logarithmically dependent on the biasing current. On a linear scale, we can see that the measured variation and the simulated one match with a high precision. To emphasize the difference between measurement and the post-layout simulations, these variations are plotted on a logarithmic scale. The two curves are slightly shifted. This difference was evaluated to an average relative error value of 3.1%.

6.5.4.3 Delay variability

The histogram in Fig. 6.12 depicts the variability of the measured delay tap value among 150 delay taps in 10 different chips. The delays in each one of these chips is biased with a current equal 20nA. This current value corresponds to the mean delay value shown in Fig. 6.12. Moreover, we can see that the delay value varies with a standard deviation of 8.45μs.

This standard deviation represents a variation of about 12.9% from the mean value. This matches the variation evaluated using the Montecarlo simulation presented in the previous chapter. Hence, the fabrication process is very robust to deliver multiple chips with similar behavior.

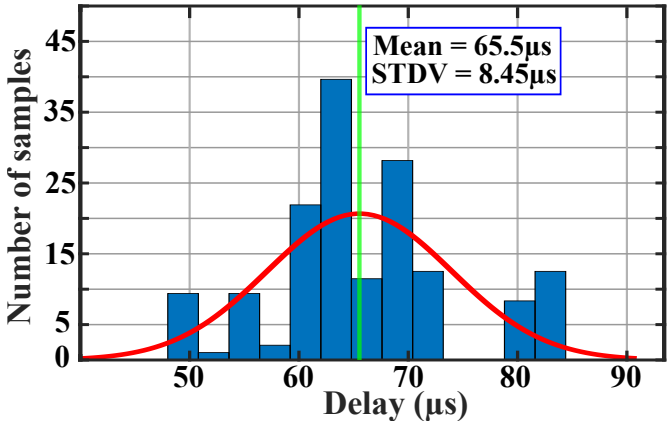


Figure 6.12 Measured delay variability over 10 SoC samples.

6.5.4.4 Delay ignorance of high-frequency events.

The objective of this test is to show that if the pulse at the REQ signal is spaced by less than the granular delay, the delay units will completely ignore them. For this test we kept the same pins configuration as before, except at the REQ input we increased the pulse frequency to see the delay response.

To show the capacity of the delay cells to ignore events, we decided to plot the evolution of the power consumption of the delay line, while increasing the frequency of pulses.

The result of this measurement is described by the graph shown in Fig 6.13. The power consumption of the delay line has a linear behavior at the beginning until arriving at the granular frequency ($1/T_{GRAN}$), where we can see that the power consumption then stagnates at the same value. This can be explained by the fact that increasing the frequency of events does not increase the operation frequency of the delay cells, because above the granular frequency limit, events with short period are ignored.

6.5.5 Test on audio files

The final functional feature to be tested on-chip is the ability to construct the audio spectrograms with the same precision as the reference Matlab® model. For this test, all the supply voltages, resets, and clocks are activated.

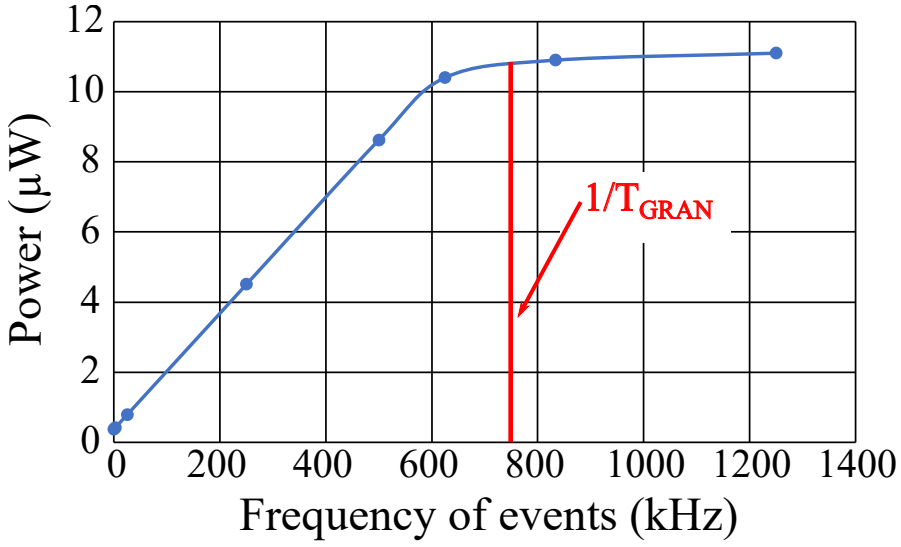


Figure 6.13 Measured power consumption of the delay line as a function of the frequency of pulses at the REQ signal input.

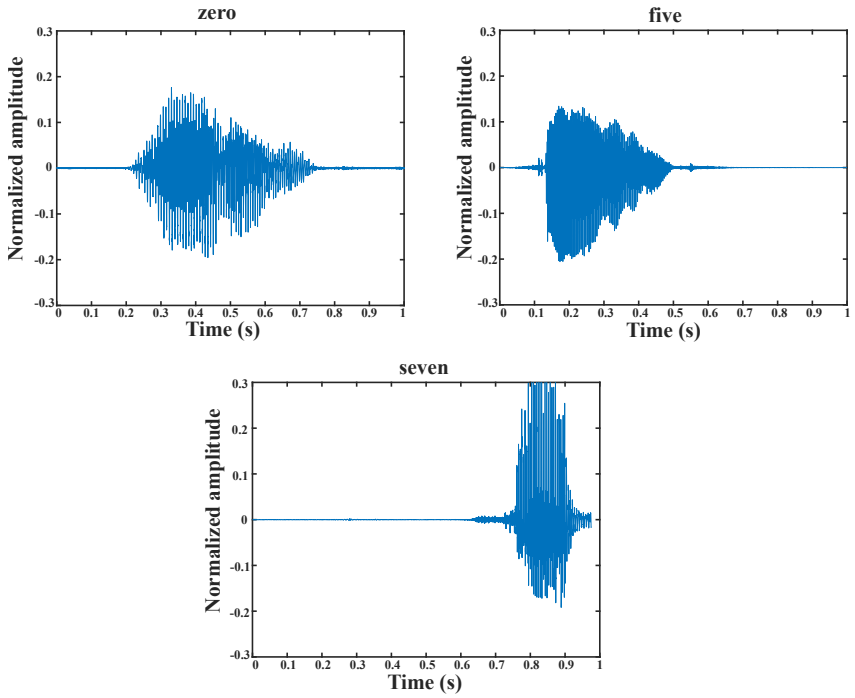
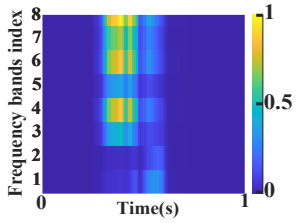
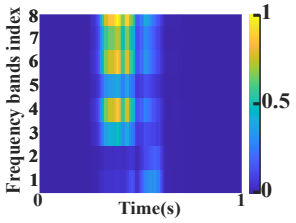
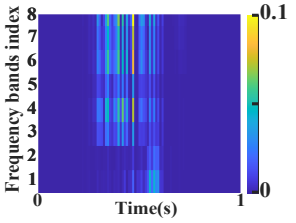
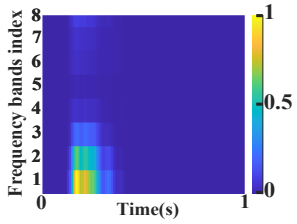
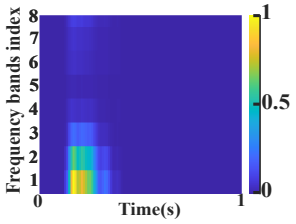
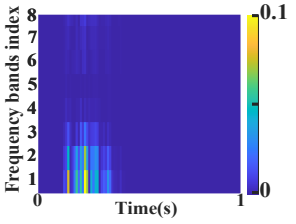
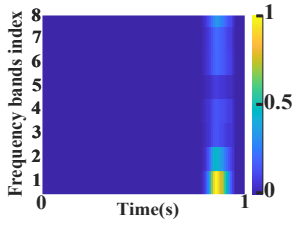
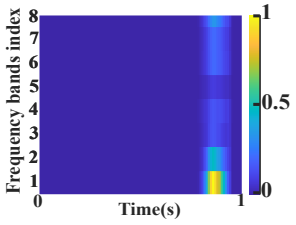
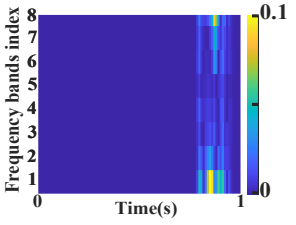


Figure 6.14 Examples of some audio files.

Audio samples, used for the test, are taken from the Google speech command dataset. Fig. 6.14 shows some audio samples for keywords "zero", "five" and "seven", that will be used specifically for this test. These audio samples have a duration of 1 second and a maximum

normalized amplitude equal to 0.3. These audio files are encoded using the behavioral 6-bit LC-ADC to generate the REQ and DIR signals, which are then input to the feature extractor using the corresponding pins.

Table 6.1 Normalized spectrograms generated using *Matlab*® and measured SoC for "Zero", "Five" and "Seven" keywords and the 10X magnified difference.

Keyword	Matlab generated spectrograms	SoC measured spectrograms	10X magnified difference
"Zero"			
"Five"			
"Seven"			

To analyze the result of this test, the same audio files are passed through the reference Matlab® model. Then, we will perform a comparison between the two generated spectrograms. The ideal and measured spectrograms are depicted in Table 6.1 for keywords "zero", "five" and "seven". From a visual inspection, these spectrograms are completely similar. To deepen the comparison, the third column of the previous table emphasize the differences on a 10X magnified scale. Effectively, on this finer scale, we start to visualize some discrepancies. Besides the visual comparison, we perform a quantitative analysis to calculate the relative error between the ideal and measured spectrograms. For the chosen example keywords, the

relative error is equal to 1.08%, 1.12%, and 0.53% for "zero", "five" and "seven", respectively. This difference between ideal and measured spectrograms is evaluated for 100 audio samples and results in an average relative error of 0.87%.

6.5.6 Recognition accuracy

The recognition accuracy is one of the key points to validate the operation of the feature extractor. Using the measured spectrograms as an input to the 6-layer CNN, the system will be tested on the recognition of different numbers of keywords, while including the background.

6.5.6.1 10 keywords case

The target of this keyword spotting system is to recognize 10 keywords plus the background noise. To do so, we chose the number from 0 to 9 as the test keywords. For this test, we used 31100 audio samples for the training phase, and the inference was done on 3600 samples. This operation was executed multiple times, and the average accuracy result is detailed in the confusion matrix depicted in Fig. 6.15. The system recognizes 11 audio classes (10 keywords + background) with an average recognition accuracy equal to 90.08%.

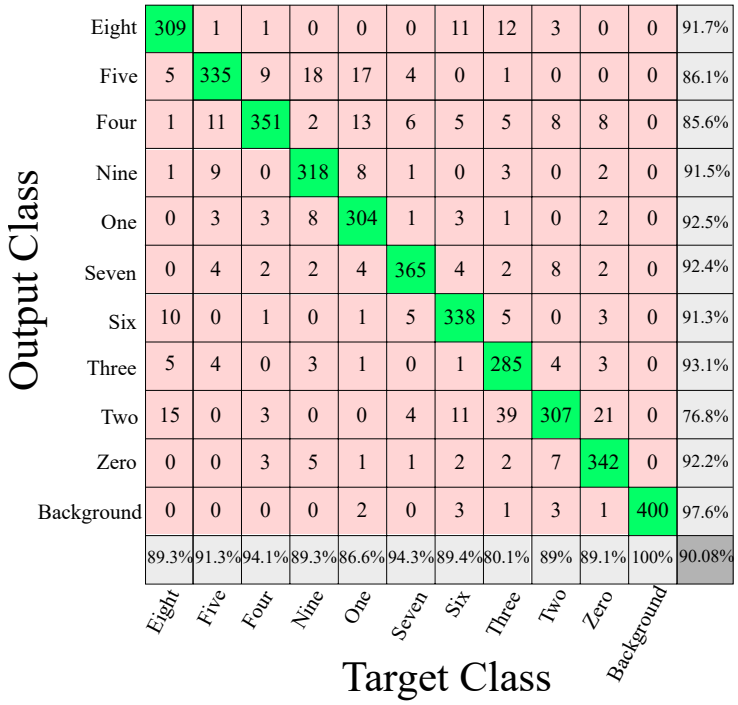


Figure 6.15 Measured confusion matrix for 11 audio classes (10 keywords + background).

6.5.6.2 Recognition accuracy variation with several keywords.

The graph shown in Fig 6.16 shows the evolution of the recognition accuracy as a function of the number of target keywords. The recognition is naturally decreasing with the increasing number of keywords. Specifically, the accuracy is going from 99.6% down to 90.08%, when increasing the number of keywords from 1 to 10.

Additionally, the impact of delay variability of the recognition accuracy is depicted in Fig. 6.16. This delay variability induces a slight reduction in the classification accuracy. Precisely, for 10 keywords, we observe a loss of 0.32% between the measured accuracy (90.08%), taking into account the variability impact, and the accuracy when considering an ideal delay (90.4%).

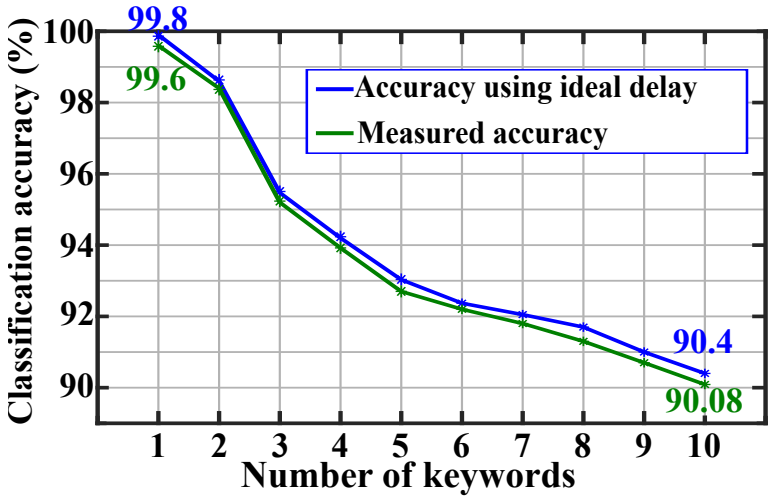


Figure 6.16 Measured recognition accuracy Vs Number of keywords.

6.6 Power analysis

The on-chip operation of the feature extraction system is validated. It has demonstrated a capacity to recognize up to 10 keywords with an accuracy higher than 90%. The further test concerns the evaluation of the necessary power budget to recognize these keywords.

This test will evaluate the power consumption of the delay line as well as the digital processing bands in two cases. The first case is when the system is analyzing an input signal with a rate of 1 keyword/second. The other case is when there is no audio activity (no keyword).

6.6.1 Delay line

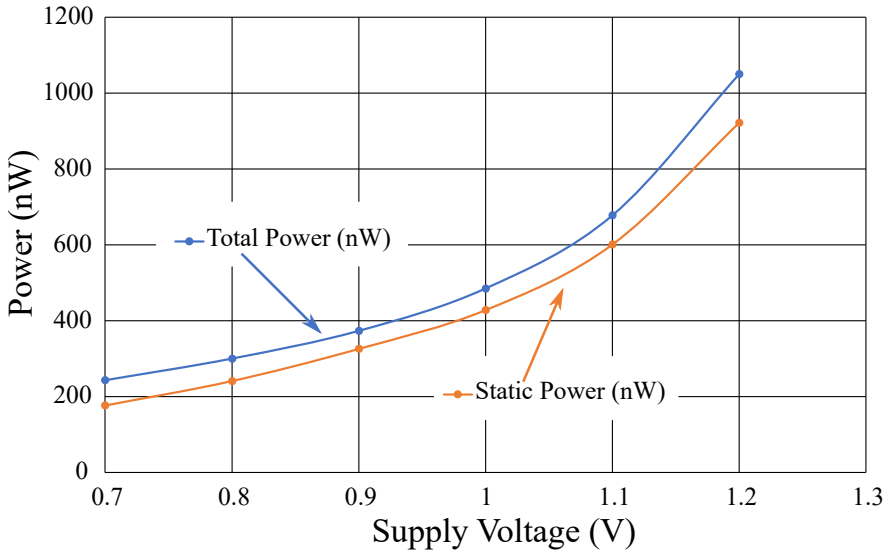


Figure 6.17 Measured power consumption variation for the delay line as a function of the supply voltage. Total power concerns keywords analysis. Static power is leakage.

Starting by the delay line, Fig. 6.17 shows the evolution of its power consumption in the two previously specified cases. For a supply voltage ranging from 0.7 to 1.2V, the delay line total power consumption increases from 245.6μW up to 1.05μW. Thanks to the event-driven operation, the total power consumption is mainly dominated by static power. At 0.7V, the static power is equal to 176.2nW, representing 71.7% of the total power.

6.6.2 Digital processing bands

For the digital processing bands, we set the body bias voltages to 1.8V and -1.8V, for V_BB_P_CORE and V_BB_N_CORE, respectively. Then, we study the evolution of the power consumption as a function of the supply voltage. this evolution is depicted in Fig. 6.18. The power ranges from 316.57nW up to 37.7μW when varying the supply voltage from 0.61V up to 1.2V. The two graphs in Fig. 6.18 representing the total and static power consumption are visually matching, which means that the static power is dominating.

To emphasize the difference between the two graphs, we plotted them on a logarithmic scale in Fig. 6.19. At 0.61V, the static power is equal to 275.484nW, which is equal to 87% of the total power. This static power domination validates the advantage of using event-driven processing.

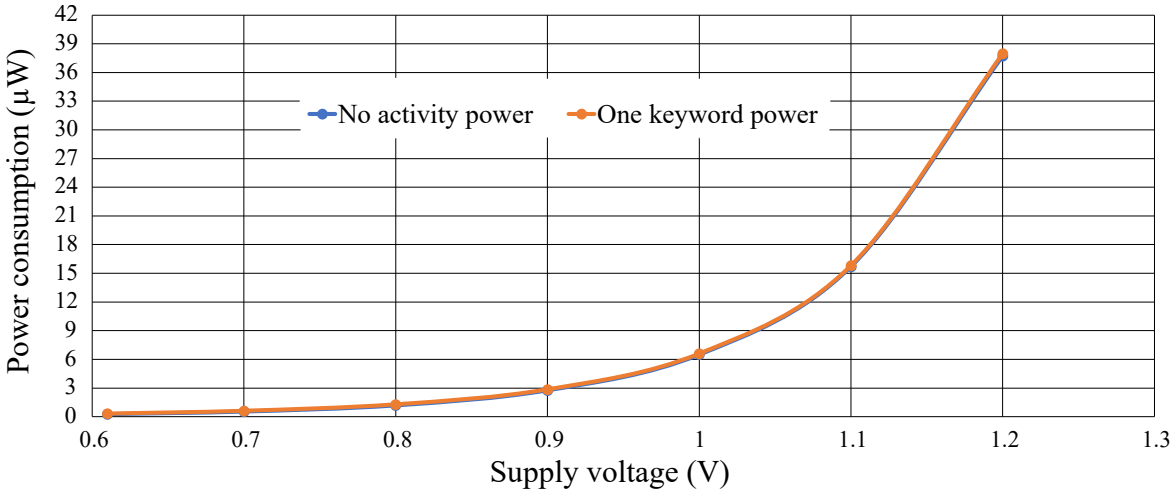


Figure 6.18 Measured power consumption variation for the digital processing bands as a function of the supply voltage. This variation is measured when the body bias voltages are equal to 1.8V and -1.8V, for V_BB_P_CORE and V_BB_N_CORE, respectively.

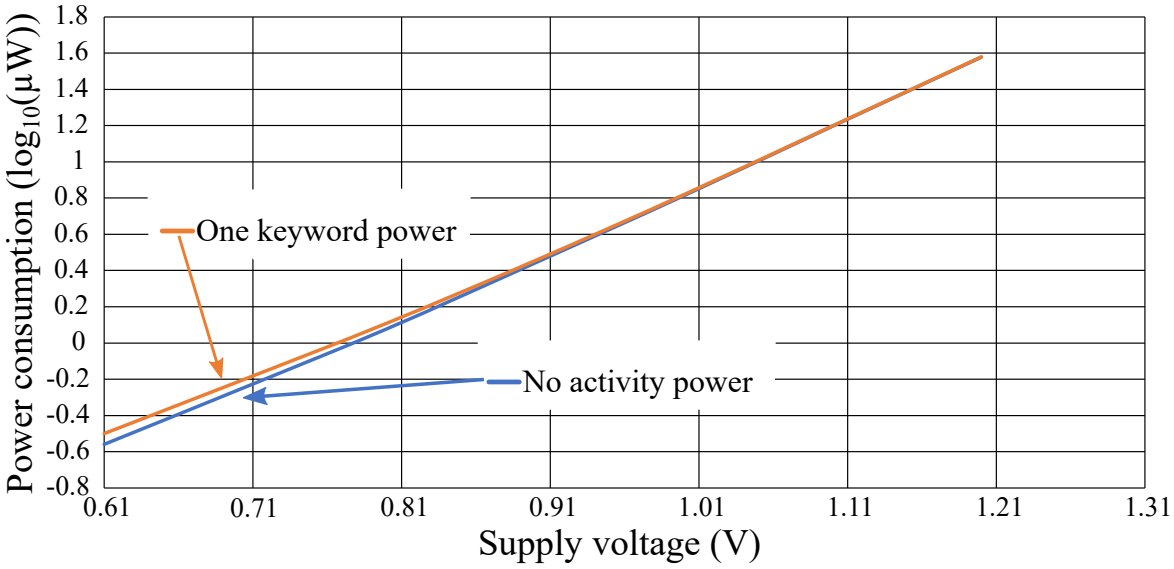


Figure 6.19 Power consumption variation for the digital processing bands on a logarithmic scale.

Up to now, the power consumption of the digital processing bands is measured when the two body bias voltages are equal to 1.8 and -1.8V. These body bias values are the limit voltages set by the technology. But, to study the body bias impact, we have studied the evolution of the power consumption when varying the body bias voltage. For this

measurement, the supply voltage of the digital processing bands (VDD_CORE) is equal to 0.61V.

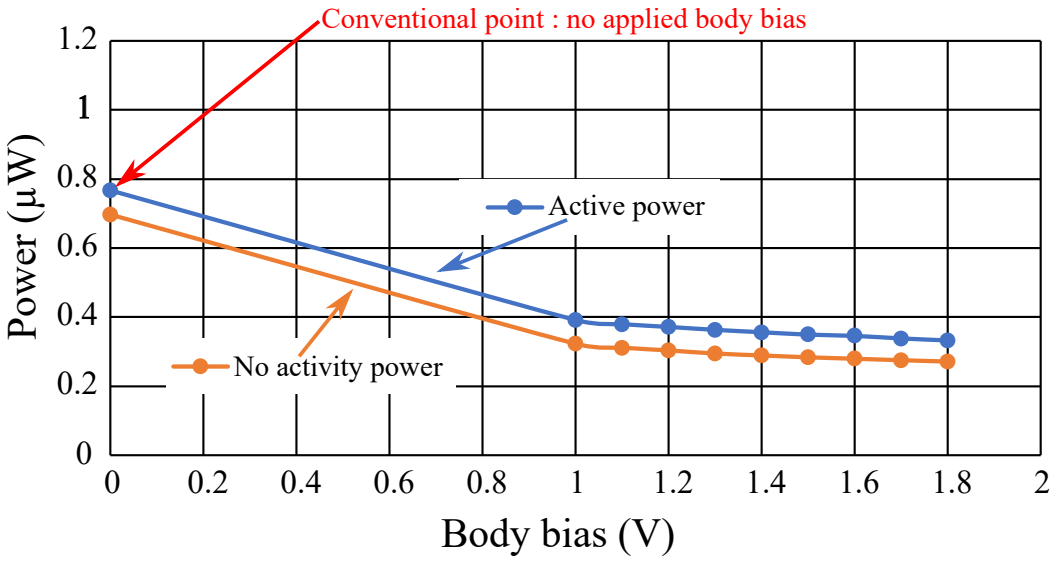


Figure 6.20 Measured power consumption variation as a function of the body bias voltage, for a supply voltage set to 0.61V.

Fig. 6.20 shows the power consumption variation as a function of the body bias voltage. This voltage, in this case, means the absolute value of V_BB_P_CORE, except the first point. This is the conventional point, where V_BB_P_CORE and V_BB_N_CORE are equal to 1 and 0, respectively. Thanks to the body bias voltage, we succeeded in decreasing the power consumption from 770.43nW down to 316.57nw. This represents a nearly 60% decrease. Hence, the body bias feature helped us to improve the power efficiency.

6.6.3 Power repartition

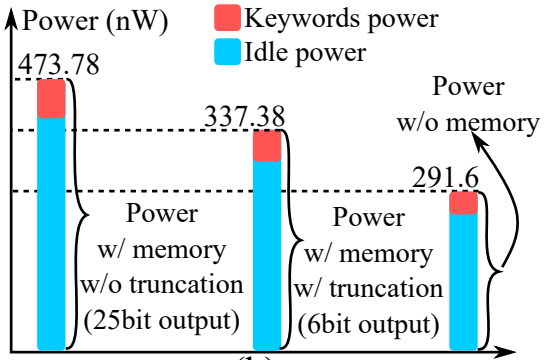
The power repartition is necessary to spotlight the blocks dominating the power consumption. To do so, the best method is to separate the supply pins of each sub-block inside of the system. Unfortunately, due to a tight schedule for tape-out, we were not able to add multiple power domains, especially in the digital routed part. So, to resolve this issue, we decided to combine the power consumption measurements and the post-layout simulated power repartition.

Besides, it was described in the previous chapter, that the feature extraction circuit can be used in three different functional modes. The first mode is the streaming mode, where the energy values are streamed in real time without going through the truncation memory. in this

mode, the memory is completely shut down. The second mode is the storage and truncation mode, where the energy values are stored and then truncated on 6 bits. In this mode, the non-used parts of the memory are switched off and we keep running only the part containing the truncated energy values. The last mode is storage without truncation mode. In this mode, the 25-bit energy values are stored, and then completely collected at the output. The reading clock speed in this mode is 4 times faster than in the "storage and truncation" mode because we need to read the 25-bit stored data instead of the 6-bit truncated data.

Block	Power (nW)	Percentage
Delay taps	189.51	40%
Truncation Memory	181.93	38.4%
Approx squarer	12.31	2.6%
CT FIR Filter	28.42	6%
Coefficient bank	28.42	6%
Energy accumulator	16.58	3.5%
Other	16.58	3.5%

(a)



(b)

Figure 6.21 (a) Measured power breakdown. (b) Measured power for the three functional modes.

For the first mode, the power repartition is detailed in the table depicted in Fig. 6.21a. The total power is equal to 473.78nW with a major domination of the memory (38.4%) and delays (40%). Fig. 6.21b shows the power repartition between idle and active states for the three possible functional modes. The power for the two modes using the on-chip memory shows interest in using the in-memory truncation that reduces the output data traffic, which results in a reduction of the power by 28.7%. Finally, the memory will not be included in the KWS case, resulting in an activity-dependent power consumption measured at 291.6nW, for a speech rate of 12 keywords/minute. This speech rate was chosen to compare our work to some state-of-the-art event-driven keyword spotting systems, that set the same rate for power consumption evaluation.

6.7 Comparison with state of the art

To evaluate the performance, we need to compare our system to the latest state-of-the-art feature extraction systems designed for keyword spotting applications.

Table 5.6 summarizes the performance of the proposed feature extraction system and other on-chip systems. In this table, we divided the state-of-the-art systems into two categories. The first one is the category specifying systems targeting recognition of 10 keywords at least. The second category contains systems with less than 10 keyword recognition. This categorization aims to highlight the fact that the more keywords we recognize, the more complex is the system.

Focusing on the 10 keywords category, we can see that our system presents a distinctive programmable event-driven feature extraction solution showing better performance in terms of accuracy, power consumption, and area compared to the solutions presented in [4], [5] and [6]. Moreover, the feature extractor presented in [7] shows better power and accuracy than our system, but uses a non-programmable and non-scalable high-complexity implementation, requiring 31 filters to cover a frequency range of 100Hz-4kHz, while in our case the complexity is considerably reduced since we need only 8 filters to cover a bandwidth of 840Hz-6.25kHz.

Concerning the other specified category, the systems presented in [8] and [9] are targeting the recognition of only 1 and two keywords, respectively. In terms of classification accuracy, the system in [8] recognizes 1 keyword, with an accuracy at best equal to 94%. Based on the graph in Fig. 6.16, our system is recognizing 1 keyword with an accuracy equal to 99.6%, which is higher by 5.6% than the system in [8]. The other system presented in [9] is limited to the recognition of 1 and 2 keywords with an accuracy of 97.3% and 94.6%, respectively, compared to 99.6% and 98.3% in our system. In terms power consumption, only the event-driven system described in [8] has a better performance. However, this system is limited to the recognition of only 1 keyword instead of 10 keywords in our case. This means that the target task in our case is more complex. Hence, the power consumption is scaled with the task complexity.

Globally, the performance of our proposed feature extraction system is at par or better than the latest state-of-the-art systems.

6.8 Conclusion

This chapter has detailed the different test steps followed to verify and validate the functioning of the on-chip implemented feature extractor. Based on the different results, the designed system has demonstrated the recognition of 11 audio classes (10 keywords + background) with an accuracy equal to 90.08%. This recognition performance was achieved, while having a speech rate dependent power consumption of 291nW, for a speech rate of 12 keywords/minute.

Table 6.2 Comparison with state-of-the-art feature extraction systems targeting keyword spotting.

	10 Keywords recognition					<10 Keywords recognition	
	This work	**JSSC'23 [7]**	**ISSCC'22 [4]**	**TCAS'21 [5]**	**VLSI'19 [6]**	**ISSCC'21 [8]**	**JSSC'21 [9]**
Feature Extractor type	Time-based CT FIR filters	Analog Filters	Ring-Oscillator based BPF	Switched-Capacitor filters	MFCC	Asynchronous spike-based IROC	Serial-FFT_based MFCC
Technology	28nm FD-SOI	65nm	65nm	130nm	65nm	180nm	28nm
Supply voltage (V)	0.7 for Delays 0.61 for Digital	0.4	0.5 for Analog 0.75 for Digital	1.2	0.6	0.6	0.41
Area (mm^2)	0.151 w/o Mem 0.57 w/ Mem	0.53	1.6	0.79	2.56 (Complete system)	13.86 (Complete system)	0.22 (Complete system)
Scalability[a]/Programmability	Yes/Yes	No/No	Yes/No	No/No	No/Yes	Yes/Yes	No/No
Event-Driven	Yes	No	No	No	No	Yes	No
Idle Power (nW)	270.7	—	—	—	—	35	—
Energy/keyword (nJ)	1.74	—	—	—	—	—	—
Total Power (nW)	291.6[bc]	80	9300	800	6282.9	378 (Complete system)	340
Frequency range (Hz)	840-6.25k	100-4k	111-10.4k	30-8k	—	—	—
Nb of channels	8	31	16	32	—	—	31
Feature rate (Hz)	100	100	62.5	100	—	—	62.5
#Classes(keywords)	11(10)	11(10)	12(10)	12(10)	10	1	1/2
Accuracy (%)	90.08	91.5	86.03	82.4-87.4[d]	90.87	85-94[e]	97.3/94.6
Classifier	Off-chip software CNN	Off-chip software CNN	On-chip RNN	Off-chip software RNN	LSTM & GMM	On-chip CNN	DSCNN

[a]Scalability : Technology nodes scalable [b]Total Power = Idle power +Speech rate x Energy/keyword
[c]Power calculated for a speech rate of 12 keywords/minute, without including the truncation memory
[d]Considering 5-10% degradation from excluding the off-chip logarithm [3] [e]Accuracy depends the target keywords

These performances have been compared to the ones of the latest state-of-the-art feature extraction systems. This comparison has concluded that our system presents the first scalable and programmable event-driven feature extraction unit targeting keyword spotting application for more than 10 audio classes.

Bibliography

[1] https://www.st.com/resource/en/application_note/an5886-guidelines-for-design-and-board-assembly-of-land-grid-array-packages-stmicroelectronics.pdf

[2] https://www.mouser.com/datasheet/2/448/QFN-05-100-pitch_01-845196.pdf

[3] https://www.keysight.com/us/en/assets/7018-05167/data-sheets/5992-1430.pdf

[4] K. Kim et al., "A 23-μW Keyword Spotting IC With Ring-Oscillator-Based Time-Domain Feature Extraction," in IEEE Journal of Solid-State Circuits, vol. 57, no. 11, pp. 3298-3311, Nov. 2022, doi: 10.1109/JSSC.2022.3195610.

[5] D. A. Villamizar, D. G. Muratore, J. B. Wieser and B. Murmann, "An 800 nW Switched-Capacitor Feature Extraction Filterbank for Sound Classification," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 68, no. 4, pp. 1578-1588, April 2021, doi: 10.1109/TCSI.2020.3047035.

[6] J. S. P. Giraldo, S. Lauwereins, K. Badami, H. Van Hamme and M. Verhelst, "18μW SoC for near-microphone Keyword Spotting and Speaker Verification," 2019 Symposium on VLSI Circuits, Kyoto, Japan, 2019, pp. C52-C53, doi: 10.23919/VLSIC.2019.8777994.

[7] S. Ray and P. R. Kinget, "Ultra-Low-Power and Compact-Area Analog Audio Feature Extraction Based on Time-Mode Analog Filterbank Interpolation and Time-Mode Analog Rectification," in IEEE Journal of Solid-State Circuits, vol. 58, no. 4, pp. 1025-1036, April 2023, doi: 10.1109/JSSC.2022.3227246.

[8] Z. Wang et al., "12.1 A 148nW General-Purpose Event-Driven Intelligent Wake-Up Chip for AIoT Devices Using Asynchronous Spike-Based Feature Extractor and Convolutional Neural Network," 2021 IEEE International Solid- State Circuits Conference (ISSCC), San Francisco, CA, USA, 2021, pp. 436-438, doi: 10.1109/ISSCC42613.2021.9365816.

[9] W. Shan et al., "A 510-nW Wake-Up Keyword-Spotting Chip Using Serial-FFT-Based MFCC and Binarized Depthwise Separable CNN in 28-nm CMOS," in IEEE Journal of Solid-State Circuits, vol. 56, no. 1, pp. 151-164, Jan. 2021, doi: 10.1109/JSSC.2020.3029097.

Chapter 7

Conclusion & Perspectives

7.1 Conclusion

The thesis work presented in this dissertation summarizes the development of a new event-driven continuous-time keyword spotting system. It is composed of an asynchronous spectral-based feature extraction unit translating the audio input signal into a spectrogram, displaying the spectral energy distribution through 8 frequency bands. The covered speech bandwidth is ranging from 840Hz up to 6.25kHz. The generated spectrograms are then input into a convolutional neural network to generate the classification results. The event-driven feature extractor is composed of 8 digital processing bands. In each band, the LC-ADC encoded audio signal undergoes filtering using 16-tap continuous-time FIR filters. Then, an energy extractor computes the energy values, in each frequency band, of the filtered output. The per-band energy values are stored and concatenated together to construct the target spectrogram.

Firstly, the proposed system was modeled using *Matlab*® to validate its operation and extract the key design parameters. Then, these parameters were used as a basis for the hardware implementation of the feature extractor mainly. Using VHDL language, we designed an event-driven clock-gated feature extraction unit. Then, it was implemented and tested on an FPGA board to test the operation, validate the hardware feasibility, and evaluate the hardware complexity.

The latter validation step helped to start the physical implementation of the feature extraction unit using the 28nm FD-SOI CMOS technology. The difference in this case will be the replacement of the clock-gated delay cells with the asynchronous analog thyristor-based delay cells. the objective of this replacement is to lower the power consumption. The feature extractor becomes composed of analog delay cells and digital processing bands. This mixed-signal nature led to a multi-environment physical implementation. The full-custom

analog environment is used for the delay cells and the semi-custom digital environment is used for the digital processing bands implementation. However, the digital design flows were customized to be adapted to the asynchronous nature of the feature extraction unit. The analog and digital implemented blocks were assembled to form the final "Capybara" feature extraction chip.

The chip was tested for the objective of testing the operation and extracting the performances. Through chip measurement results, the fabricated feature extraction unit demonstrated the recognition of 11 audio classes (10keyword + background) with an accuracy equal to 90.08%, while having an activity-dependent power consumption of 291.6nW, for a speech rate of 12keywords/minute. These performance results are compared to the latest state-of-the-art feature extraction unit developed for keyword spotting. The conclusion is that the proposed system presents a scalable and programmable event-driven keyword spotting solution with a better or very competitive trade-off between accuracy and power consumption compared to the state-of-the-art.

7.2 Future research perspectives

The presented work demonstrated a new keyword spotting system with competitive performances. However, this work can be improved or complemented in multiple ways. These ways are described using the following research perspectives:

- **CNN hardware implementation:** To complete the keyword spotting system, the CNN needs to be integrated on-chip. For this reason, we need to ensure the hardware implementation. To do so, we need first to study the possibility of reducing the CNN size and the impact of this reduction on the recognition accuracy. After setting a final reduced size, we need to explore new methods to design the CNN, while maintaining a low power profile. One of the efficient methods is to implement only one basic kernel layer and use it iteratively for all the processing. This method reduces the needed hardware resources but will require a faster clock signal. Hence, optimizing the trade-off between the number of kernel layers to implement and the processing clock frequency is mandatory. After that, we can start the CNN physical implementation and connect it to the designed feature extractor in the global keyword spotting chip.
  Another possibility is to program the needed CNN on an external ready hardware platform such as an FPGA board or a microcontroller. This method will validate the hardware feasibility of the CNN, but will not be efficient in terms of power consumption.

- **New delay architecture:** The used asynchronous delay cells were implemented in a full custom analog way. This implementation method is done manually without any automation. This is efficient in terms of power consumption but it blocks the digital design flow automation. Hence, these delays can be designed differently to be compatible with the digital design flows while keeping a low power profile. For this reason, the delays can be implemented using power-gated counters or flip-flops. The power gating feature switches off the complete circuit during the absence of any significant input. Thus, the power efficiency will be improved. To validate this choice, we need to compare it to the actual solution and study between power efficiency and the gain in design automation.

- **Other applications:** The proposed circuit architecture can be used for other use cases such as speaker identification. To do so, the designed system needs to be improved in several aspects. The first one is the coverage of wider bandwidth and more selective filtering. This can be done through the augmentation of the CT FIR filter orders by inserting additional MAC units. By doing so, the extracted features will be more precise and relevant. Then, we need to increase the memory size to be able to host largely coded features. These improvements will lead to a more programmable architecture.

APPENDIX A

Chip assembly : Pad-ring

After validating the operation of the physically implemented system through post-layout and back-annotate simulations, the next step is to finalize the chip design. The first part of chip assembling is to design a compatible Input/Output (IO) pad ring. This IO ring should contain a minimal number of IOs sufficient to test the functionality of the designed system.

The defined IOs with their different functions are described in the table A.1. The total number of defined inputs/outputs is equal to 51.

Table A.1 Different Inputs/Outputs and their function

IO name	Type	Function
VDDE	Supply	IO Ring supply at 1.8V
GNDE	Supply	IO Ring ground voltage
VDD	Supply	Supply to shift the level of the VDD level in the core
VDD_DELAYS	Supply	Supply voltage for the delay line(0.7V)
VDD_DELAYS_TEST	Supply	Supply voltage for the delays alone to test (0.7V)
VDD_CORE	Supply	Supply voltage for the digital processing bands (0.61V)

VBB_P_CORE	Body bias	Positive body bias supply for PMOS transistors in the digital processing bands (0->1.8V)
VBB_N_CORE	Body bias	Negative body bias supply for PMOS transistors in the digital processing bands (-1.8->0V)
I_REF_DELAYS	Polarization current	Polarization current for the delays
I_REF_DELAYS_TEST	Polarization current	Polarization current for the delays alone to test
Start	Input	Inverted reset for the delay line
Start_Delays_Test	Input	Inverted reset for the delays alone to test
rst	Input	Inverted reset for the digital processing bands
Frame clock	Input	Low-frequency clock (1kHz) to generate the framing signals
SPI clock	Input	Clock (4MHz) to synchronize the charge and coefficients and number of memory lines to read
Read clock	Input	Clock(128kHz) to read the memory lines specified when the read enable is triggered
Serial clock	Input	Clock(3.3MHz) to serialize the energy
Enable	Input	Enable to trigger the programming of the coefficient bank

Enable lines	Input	Enable to trigger the charging of the number of lines to read
Serial coeffs	Input	Input coefficients in series to charge the coefficient bank
REQ	Input	REQ signal for the delay line
REQ_test	Input	REQ signal for the test delays alone
DIR	Input	DIR signal for the delay line
Read enable	Input	External pulse to trigger the reading of the memories in each band
rst	Input	Inverted reset for the digital processing bands
Acknowledge_PD	Input	Output acknowledge signal of the delay line
Acknowledge_ND	Output	Input acknowledge signal of the delay line
Frame1	Output	First framing signal
Frame2	Output	Second framing signal
Frame3	Output	Third framing signal
Read busy 1,2,...,8	Output	Signal at high level during memory reading in the first, second until the eightieth processing band
Read flag	Output	Signal synchronizing the memory reading process in all the bands
Serial energy	Output	Energy values stored in the memory output through an SPI interface

Serial busy	Output	Signal at high level whenever we read an energy value from the memory
Truncation position	Output	Truncation position output through an SPI interface
Truncation busy	Output	Signal at high level when reading the truncation position
Combined REQ signal 1	Output	Combined Req signal in the first band (16 concatenated delayed REQ signals)
Combined REQ signal 8	Output	Combined Req signal in the eightieth band (4 concatenated delayed REQ signals)
memory lines	Output	Number of lines to be read in series output from the SPI
Coeffs busy	Output	Signal at high level when we are programming the coefficient bank
Coeffs ready	Output	Pulse generated when finishing the programming of the coefficient bank
Coeffs out series	Output	Signal to check if the coefficients are correctly serialized through the SPI
Delays test out	Output	output of the delays used for stand-alone test

To construct the corresponding pad ring, we used the General Purpose Input Output (GPIO) library. These GPIOs can be programmed to act as input or output and their drive can also be programmed depending on the frequency of the signal that will pass through them. The designed pad ring is shown in Fig. A.1. It is composed of 54 pads. The

3 additional pads are VDDE pads. Based on the GPIO library specifications, we need a ring supply pad (VDDE) for every 10 GPIOs. Moreover, the pads (VDD_DELAYS, VDD_DELAYS_TEST, VDD_CORE), used to supply the different parts of the circuit, contain an integrated Electrostatic Discharges (ESD) protection. The different pads are arranged symmetrically resulting in a 1.38mmX1.38mm square shape.

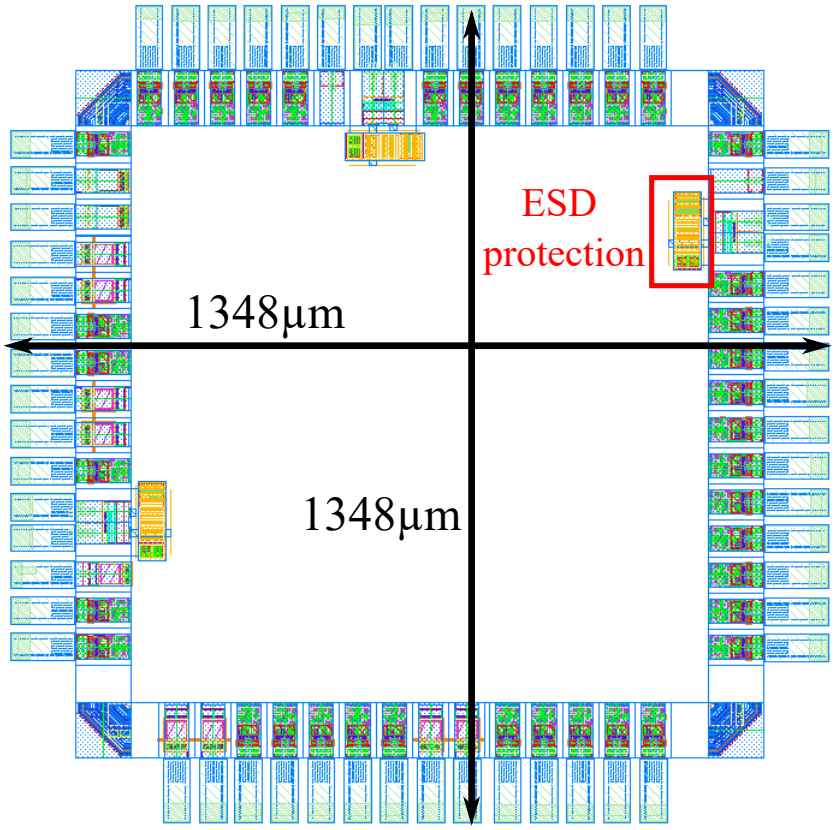


Figure A.1 Pad ring.

One of the common errors committed during the pad ring design is inverting the direction of a certain IO pad. This error will invert the direction of the clamp diode inside the pad, which will induce the appearance of a very high breakdown current. This current will completely short-circuit the chip and will never work. To prevent this error, a DC simulation needs to be performed to evaluate the leakage current of the pad ring. In our case, the leakage current was equal to 626.7nA, evaluated in the TT corner and at a temperature equal to 25°C.

APPENDIX B

Measurement configurations.

B.1 Chip alive test

The first test to be performed is the chip-alive test. This test has the objective of verifying if the chip reacts to the application of a supply voltage. During this test, we connect the power supply pins to the voltage source and we measure the current value at the same pins.

For this test, the different supply voltages are set using the configuration in table B.1.

Table B.1 Configuration of supply pins for the chip-alive test

Pin	Voltage (V)
VDDE	1.8
GNDE, GND_CORE	0
VDD_DELAYS	-0.25 -> 1.6
VDD_CORE	-0.25 -> 1.1
VBB_P_CORE	1
VBB_N_CORE	0

The graphs in Fig.B.1 and Fig.B.2 show the I-V characteristics for the delay and the digital processing bands. Based on these graphs, the two blocks react electrically to the application of a supply voltage. Moreover, the measured current values are in the same range as the simulated ones. Hence, the chip is electrically alive.

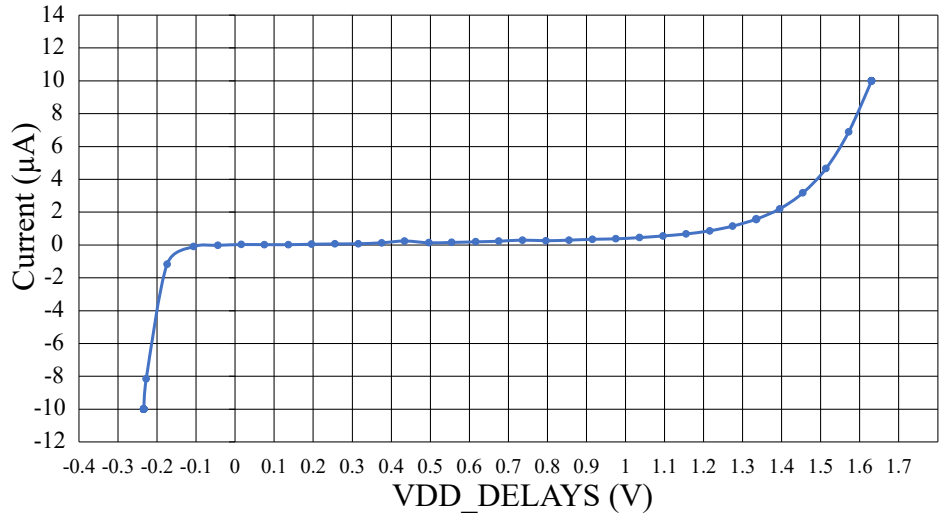


Figure B.1 I-V curve for the delays.

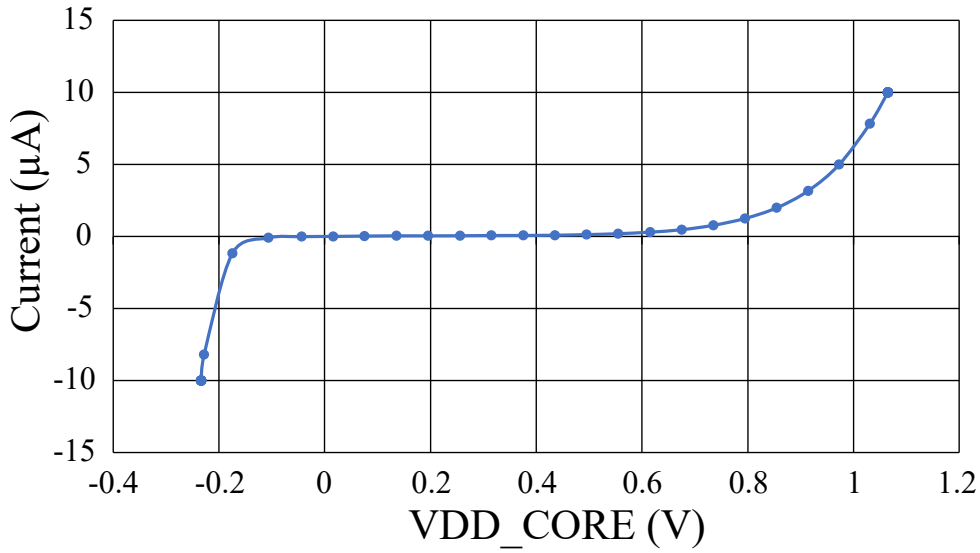


Figure B.2 I-V curve for the digital processing bands.

B.2 Pins configuration for the other tests

Table B.2 Configuration of pins for the SPI configuration test

Pin	Configuration
VDDE	1.8V
GNDE, GND_CORE	0V
VDD_DELAYS	0V
VDD_CORE	0.61V
VBB_P_CORE	1.8V
VBB_N_CORE	-1.8V
rst	activated
enable	activated

Table B.3 Configuration of pins to test framing signals.

Pin	Configuration
VDDE	1.8V
GNDE, GND_CORE	0V
VDD_DELAYS	0V
VDD_CORE	0.61V
VBB_P_CORE	1.8V
VBB_N_CORE	-1.8V
rst	activated
Frame clock	1kHz clock activated

Table B.4 Configuration of pins to test the delay line.

Pin	Configuration
VDDE	1.8V
GNDE, GND_CORE	0V
VDD_DELAYS	0.7V
VDD_CORE	0.61V
VBB_P_CORE	1.8V
VBB_N_CORE	-1.8V
rst	activated
Start	activated
I_REF_DELAYS	19nA