

ALIFE-BCI: An Adaptive Low-power Integrated Feature Extractor for Brain-Computer Interfaces

Joe Saad
Univ. Grenoble Alpes
CEA, LIST
Grenoble, France
joe.saad@cea.fr

Ivan Miro-Panades
Univ. Grenoble Alpes
CEA, LIST
Grenoble, France

Adrian Evans
Univ. Grenoble Alpes
CEA, LIST
Grenoble, France

Lorena Anghel
Univ. Grenoble Alpes
CNRS, Grenoble INP
Grenoble, France

Abstract—Brain-Computer Interfaces (BCIs) have the potential to restore motion for patients suffering from spinal cord injuries. Making such systems embedded, or even implantable, imposes strict low power constraints. Feature extraction, which transforms brain signals into intermediate representations before decoding motor intent, is typically the most compute intensive step. In this work, we introduce ALIFE-BCI, an Adaptive Quality Feature Extractor (AQFE), based on a Continuous Wavelet Transform (CWT) that captures the signal dynamics in both the time and frequency domains. The system is optimized with a top-down approach: (i) At the algorithmic level, it implements a piecewise linear approximation of the CWT that allows real-time energy-accuracy trade-offs. (ii) At the architectural level, memory reuse and parallelism are used to balance area and compute performance. (iii) At the circuit level, low-power techniques are used in a 22 nm FDSOI technology physical implementation flow. Three variants, with different levels of parallelism, are explored to extract 960 features at a rate of 10 Hz for a BCI motor application. The optimal variant, with an area of only 0.061 mm^2 , achieves $0.27 \mu\text{W}/\text{feature}$ at maximum quality, and $0.13 \mu\text{W}/\text{feature}$ at minimum quality, resulting in $8\times$ lower power than existing digital solutions. Combined, these characteristics make the system well-suited for ultra-low-power implantable BCI decoders.

Index Terms—Brain-computer interfaces (BCIs), Feature extraction, Low-power circuits, Approximate computing.

I. INTRODUCTION

Brain-Computer Interfaces (BCIs) continue to evolve from research prototypes to clinical systems [1], used for assistive devices and rehabilitation [2], [3]. Such systems must operate under power and area constraints [4], especially when targeting portable or implantable devices [5]. Most BCIs rely on extracting meaningful information from signals measured in specific brain areas such as the motor cortex. The extracted patterns can be used to drive an effector such as an exoskeleton [6] or a spinal stimulator [7], to induce controlled movement for patients with a spinal cord injury. The process is often performed in two steps: feature extraction and output decoding. A given system has a fixed set of feature extractors, chosen based on signal properties, such as time, frequency or chaotic parameters [8]. Output decoding consists of finding a mapping between the extracted features and the outputs.

Among brain signals that can be used for decoding, Electroencephalography (EEG) offers an attractive balance between spatial resolution and invasiveness [9], [10], since electrodes are placed on the cortical surface through a minimally invasive

surgical procedure. Previous studies have shown that these signals can be used for motor applications. More specifically, motor intent has been shown to modulate activity in distinct frequency bands [11], which can be detected in ECoG signals. These frequency components are analyzed over time using the Continuous Wavelet Transform (CWT) [12], which consists of convolving input signals with a wavelet, a frequency pattern localized in time. The extracted components can be used to compute spectral band energy features [6], [13]. For ECoG-based BCI systems, feature extraction often dominates the computational load, thus the power consumption of the system [4].

Several methods have explored approximations of the CWT, using polynomial [14] and spline-based approximations of the wavelets [15], but they have not been investigated in the BCI context. More recently, the Piecewise Linear Continuous Wavelet Transform (PLCWT) approximation [16] was introduced and validated for BCI decoding. This technique is attractive as the level of approximation can be adjusted, while preserving a high decoding accuracy. Although the software implementation reduces arithmetic operations, it remains to be shown that algorithmic benefits translate into energy savings.

In this paper, we present an Adaptive Quality Feature Extractor (AQFE) for BCIs, based on an algorithm-to-circuit optimization of the PLCWT. The system can dynamically adjust the approximation level of the features, based on the requirement of the decoding task. Three variants of the circuit were implemented in 22 nm FDSOI technology, to evaluate the impact of parallelism. The highly optimized module extracts up to 960 features at a rate of 10 Hz with an energy consumption of only $0.133\text{--}0.266 \mu\text{W}/\text{feature}$, depending on the approximation level. Furthermore, the architecture enables temporal re-use of hardware resources, resulting in a total silicon area between 0.061 to 0.120 mm^2 . Combined, these contributions will enable ultra-low-power implantable BCI motor decoders.

The remainder of this paper is organized as follows. Section II discusses related work in wavelet-based feature extraction and low-power BCI hardware. Section III introduces the AQFE module, describing the multi-layer optimizations applied at the algorithmic, architectural, and circuit levels. In section IV, we compare the system performance to the existing State-of-the-Art (SoA) BCI circuits, and we conclude in section V.

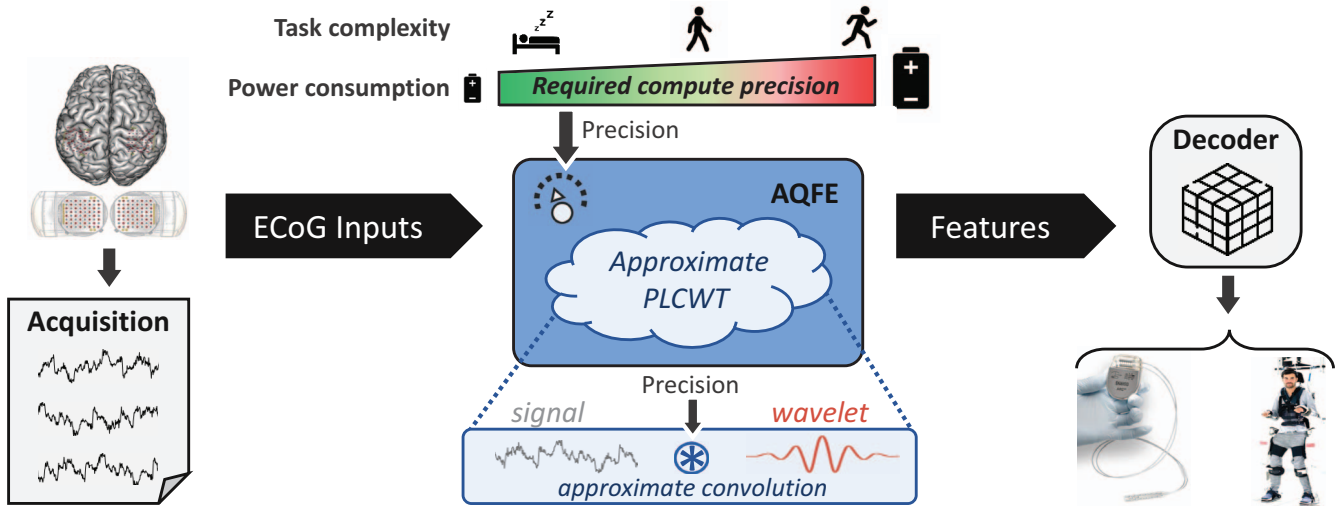


Fig. 1: Overview of a BCI System with an Adaptive Quality PLCWT-Based Feature Extraction

II. RELATED WORK

For BCI systems to better meet patients' expectations, they must be miniaturized to be integrated into portable or implantable devices. For the latter, power must be limited to meet thermal dissipation constraints, and more generally, reducing power extends battery life and enables long-term operation. Indeed, extracting meaningful information with a limited power budget has been the focus of several works aiming to balance accuracy and power constraints. Many BCI platforms have been proposed, including existing Microcontroller Units (MCUs) [17] and Digital Signal Processors (DSPs) [18]. Other custom hardware accelerators have been developed for specific applications such as epilepsy [19], [20] or motor decoding [21], [22]. While many of these circuits were optimized to minimize power consumption or area for specific applications, they lacked adaptability. For example, in [23], the authors applied a quantization of the sine function in a Discrete Cosine Transform (DCT), to simplify operations while achieving sufficient accuracy for their motor BCI. However, this quantization was fixed at design time and did not allow runtime reconfiguration. The circuit presented in [8] is able to dynamically select a subset of features per classification task. This selection is determined during training and hardwired for inference. While some features are computed using approximation techniques, the approximation is not updated at runtime, thus limiting adaptability once the system is deployed.

Although a large variety of features have been explored, in practice, most motor BCIs rely on spectral power band features, as these have been shown to be effective for motor intent decoding [6], [13]. Extracting power bands from non-stationary signals often relies on the CWT [12], which is compute intensive. Computing a CWT can be done efficiently in the frequency domain, without any approximation, using a Fast Fourier Transform (FFT) [24]. Many hardware architectures have implemented the FFT algorithm efficiently [25], [26], and some have used it to compute CWT [27]. The Discrete

Wavelet Transform (DWT) is a variant of the CWT that can be efficiently implemented using cascaded filter banks [28], requiring fewer operations than the FFT approach. However, when using the DWT, the wavelets are constrained to form an orthogonal basis and the extracted features are limited to power-of-two multiples of a base frequency. An approximate method, based on a piecewise linear approximation of the wavelet, has been previously introduced [29] to compute the CWT with fewer operations than conventional techniques. The PLCWT algorithm was tested on BCI data for motor decoding in [16]. In this paper, we propose a top-down optimization of the PLCWT, integrating algorithmic approximations with circuit-level design.

III. METHODS

A typical SoA BCI decoding flow begins with brain signal acquisition, which is then followed by feature extraction and task decoding, as shown in Fig. 1. In this system, the AQFE module extracts time-frequency features from ECoG signals by applying a PLCWT with adjustable quality, and a pre-trained model then decodes the extracted features. By adjusting this precision level, it is possible to control the number of compute operations, hence the energy consumption required for feature extraction. The required compute precision is estimated according to the task difficulty or using external feedback.

The AQFE is designed to enable a flexible trade-off between required compute precision and energy consumption through cross-layer approximation. At the algorithmic level, we implemented dynamic control of the approximation level through the subdivision parameter J . In addition, the wavelet symmetry was exploited to reduce the number of operations and the memory size. At the micro-architectural level, memory re-use and multi-lane processing balanced throughput and energy consumption. Finally, at the circuit-level, aggressive clock gating and exclusive use of ultra-low leakage standard cells, reduced dynamic and static power consumption of the physical implementation.

A. Algorithmic Level

The mathematical formulation of the CWT for a discrete signal X of length N is given as:

$$S(\tau) = \sum_{i=0}^{N-1} X[i + \tau] \psi[i], \quad (1)$$

where τ is the time shift, and ϕ a given wavelet. The PLCWT algorithm approximates the wavelet ϕ by ψ using a limited number of linear segments defining a subdivision of its interval [16]. The number of segments, referred to as the subdivision size, determines the approximation quality of the wavelet, as shown in Fig. 2.

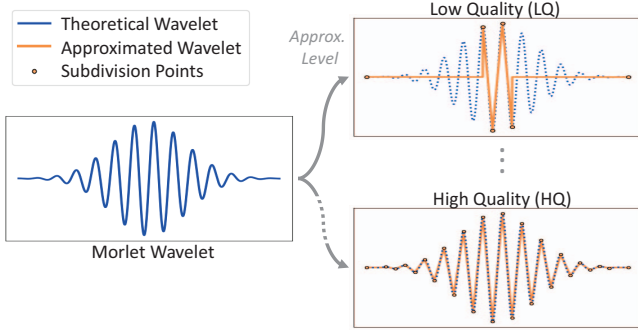


Fig. 2: Wavelet Approximation for Different Subdivision Sizes

When using this approximation, it is possible to control the number of operations required per convolution. More precisely, the PLCWT approximation is defined in (2),

$$S(\tau) = X^{(1)}(a_J + \tau)\psi(a_J) - X^{(1)}(a_1 + \tau)\psi(a_1) - I(\tau) \quad (2)$$

where $X^{(1)}$ is the first antiderivative of the input signal X , $\psi(a_1)$ (resp. $\psi(a_J)$) the wavelet values at the start (resp. end) of the subdivision, and $I(\tau)$ is an integral term given in (3).

$$I(\tau) = \sum_{i=1}^J b_i X^{(2)}(a_i + \tau) \quad (3)$$

$X^{(2)}$ is the second antiderivative of the input signal, and b_i are coefficients, pre-computed using the wavelet derivative values at the subdivision points. The number of multiplications in (3) is proportional to the subdivision size. As fixed-point multiplications require approximately $30\times$ more energy than additions [30], they dominate the overall power consumption. Moreover, when wavelets are symmetric (resp. antisymmetric), the coefficients b_i are antisymmetric (resp. symmetric). This allows an algorithmic factorization, which cuts the number of multiplications in half.

The pseudocode shown in Algorithm 1 describes the two main steps of the PLCWT: (1) signal integration using a trapezoid method, and (2) a dot product reduction using the coefficients b_i . To produce N convolution points, the input signal X must be of length $3N$ to avoid boundary effects, assuming a convolution window of length $\leq 2N$.

Algorithm 1: Feature Extraction using PLCWT

Input: Discrete ECoG signal $x[n]$ of length $3N$, J subdivision points $\{a_i\}_{i=1}^J$ of a wavelet support, border coefficients $\psi(a_1)$ and $\psi(a_J)$, precomputed coefficients $\{b_i\}_{i=1}^J$.

Output: Time-frequency components $\{S[\tau]\}_{\tau=0}^{N-1}$.

Step 1: Signal Trapezoid Double Integration;

$X^{(1)}[0] \leftarrow 0;$

$X^{(2)}[0] \leftarrow 0;$

for $n = 1$ to $3N - 1$ **do**

$X^{(1)}[n] \leftarrow X^{(1)}[n - 1] + (x[n - 1] + x[n])/2;$

$X^{(2)}[n] \leftarrow X^{(2)}[n - 1] + (X^{(1)}[n - 1] + X^{(1)}[n])/2;$

Step 2: Dot Product using Subdivision Points;

for $\tau = 0$ to $N - 1$ **do**

 // 2a. Multiply by border values

$S \leftarrow \psi(a_J) \times X^{(1)}[a_J + \tau] - \psi(a_1) \times X^{(1)}[a_1 + \tau];$

 // 2b. Reduce over the subdivision

for $i = 1$ to J **do**

$S \leftarrow S + b_i \times X^{(2)}[a_i + \tau];$

$S[\tau] \leftarrow S;$

return $\{S[\tau]\}_{\tau=0}^{N-1};$

The algorithm extracts time-frequency components from a single signal channel using a wavelet. To extract multiple frequency components from an input signal, only step 2 must be repeated using wavelet coefficients corresponding to each frequency band. For multi-channel systems, the whole algorithm must be repeated on all input signals either sequentially or in parallel. Trade-offs with different levels of parallelism are explored to balance power, area and compute time.

B. Architectural Level

1) *Hardware Blocks and Dataflow:* Fig. 3 shows the top-level block diagram of the AQFE. The input signals are provided by an upstream acquisition and pre-processing stage. These samples are stored in the *Input RAM*, separating the AQFE from the front-end. The module uses these signals to compute features that are stored in the *Output RAM*, to be read by the downstream decoding model. In addition, the *SpW0 RAM* (resp. *SpW1 RAM*) stores the wavelet border non-zero values (resp. b_i coefficients) in a sparse format {index, value}. The *Integ RAM* stores the $X^{(2)}$ values, which are then used during the reduction step. In addition to the RAMs, the architecture comprises four functional blocks: three compute units and a control unit.

- **Integrator Unit (IntU):** Computes the first and second antiderivatives $X^{(1)}$ and $X^{(2)}$ using a pair of trapezoidal integrators (Step 1).
- **Sparse Shift Multiplication Unit (SpSMU):** Fetches border values $\psi(a_1)$ and $\psi(a_J)$ of a wavelet from the *SpW0 RAM* and multiplies them with the corresponding points of the integrated signal $X^{(1)}$ (Step 2a). Only a limited number of Most Significant Bits (MSBs) are kept

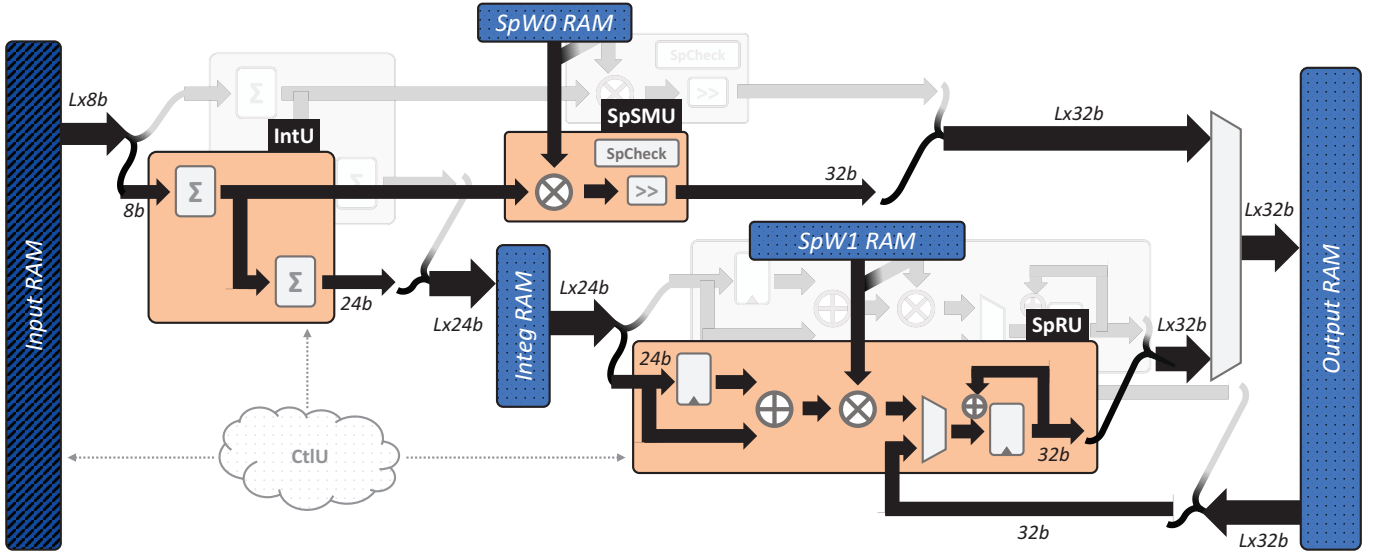


Fig. 3: Top-Level Block Diagram of AQFE showing the Compute Units (IntU, SpSMU and SpRU), the CtlU, and RAMs

after the multiplication, corresponding to a right shift applied to the result.

- **Sparse Reduction Unit (SpRU):** Implements the subdivision-based dot product between the pre-computed coefficients b_i and the corresponding samples of $X^{(2)}$ (Step 2b). For each coefficient fetched from the *SpW1 RAM*, two words can be read from the *Integ RAM*, and then be summed or subtracted before multiplication with the coefficient, as we exploit wavelet symmetry.
- **Control Unit (CtlU):** Coordinates the operations performed by the above compute units.

The AQFE can be implemented as a single-lane architecture where data from all channels is processed sequentially, or as multi-lane configurations able to process L channels in parallel, increasing the throughput by a factor L . In the latter, each lane contains its own IntU, SpSMU and SpRU, while sharing RAMs among the lanes. In Table I, we show the memory sizes needed for the three configurations. For simplicity, we consider input signals with $3N$ samples of 8 bits, recorded on C channels. Each input signal is convolved with W wavelets quantized over B bits. The *SpW1 RAM* holds $J/2$ coefficients for each wavelet, meaning that the maximum subdivision size is J , when exploiting wavelet symmetry. The *SpW RAMs* use a sparse format where each entry holds a coefficient (B bits), an index ($\log_2(3N)$ bits), and an additional control bit. When using L lanes, samples from L consecutive channels are stored in a single word in the *Input*, *Integ*, and *Output RAMs*. Note, the total size of the *Input RAM* remains constant regardless of the number of lanes. The number of channels (C) only affects the size of the *Input RAM*.

2) *Architectural Optimizations:* In addition to implementing the algorithmic optimizations of the PLCWT such as symmetry factorization and dynamic quality control, the architecture integrates further techniques to improve efficiency. These include memory re-use and multi-lane parallelism, described below.

TABLE I: RAM Sizes for Different AQFE Configurations

RAM	Width (bits)	Words
SpW0 RAM	$B + \log_2(3N) + 1$	$2 \times W$
SpW1 RAM	$B + \log_2(3N) + 1$	$J/2 \times W$
Input RAM	$L \times 8$	$(3 + 1) \times N \times (C/L)$
Integ RAM	$L \times 24$	$3N$
Output RAM	$L \times 32$	$2 \times N \times W$

a) *Memory Re-use:* The AQFE module is designed to ensure a good trade-off between area and energy consumption. As the PLCWT algorithm uses the antiderivatives $X^{(1)}$ and $X^{(2)}$ to compute time-frequency components through convolution, these values need to be stored, and then multiplied with different wavelet coefficients. A simple approach would be to store both signals in two memories which would then be read during the reduction step. To reduce memory accesses the SpSMU multiplies the border coefficients read from the *SpW0 RAM* with the corresponding points from $X^{(1)}$, on the fly, avoiding an additional memory to store the first antiderivative $X^{(1)}$. The border product results are then temporarily stored in the *Output RAM*, to be later read by the SpRU when computing the reduction. After accumulating each convolution result, the SpRU writes this final value in the *Output RAM*, overwriting the border products written by the SpSMU. By re-using the memory in which these products were stored, it was possible to reduce the *Output RAM* size by 33%.

b) *Multi-Lane Parallelism:* As highlighted in the architecture, the AQFE supports different lane configurations to process data from L channels in parallel. For a configuration with L lanes, the size of the *SpW0* and *SpW1 RAMs* remains unchanged, but the number of read requests in both RAMs is reduced by a factor of L , as the coefficients are read once for all lanes. For other memories, wider word accesses amortize

the energy cost per bit operation. Furthermore, increasing L increases the feature throughput, reducing the clock frequency needed to meet the needs of a given application, as will be shown next. More lanes, however, result in a larger area, as the *Integ* and *Output RAMs* are wider.

c) Cycle Count and Operating Frequency: The number of cycles required to process input signals depends on the selected quality and the lane configuration. The worst case occurs with maximum quality, where the largest number of subdivision points must be processed. The cycle count also scales with the signal length and the decision rate imposed by the decoding model. These parameters determine the minimum operating frequency of the AQFE module. With L lanes, this frequency is reduced by $1/L$, as multiple channels are processed in parallel. For the design implementation, the maximum quality configuration is used to dimension the clock frequency to guarantee real-time operation.

C. Circuit Level

While the algorithm reduces arithmetic complexity and the architecture balances memory and parallelism, the physical implementation further reduces power at the lowest level. The micro-architecture balanced the pipeline stages so that timing could be closed using only ultra-low leakage (UHVT28) standard cells, minimizing leakage power. All RAMs were implemented as single port memories, since data dependencies prevent meaningful parallelism and do not justify dual port access. Memory cuts were selected to minimize the overall power consumption. Careful RTL coding enabled the synthesis engine to automatically infer clock gates on 96% of the registers. In addition, unused RAM macros were clock-gated, reducing the dynamic power. Furthermore, we approximate the arithmetic operations, mainly multiplications, using a fixed-point stage-specific precision, which enabled the synthesis tool to simplify the datapath by removing unused or redundant logic. Configurable shifting and bit truncation allow tuning to minimize the number of significant bits required to maintain high accuracy. As an illustrative example, Figure 4 shows how wavelet bitwidth affects the average Correlation Coefficient (CC) between hardware and software features, guiding the minimum bitwidth choice of 17 in the final design.

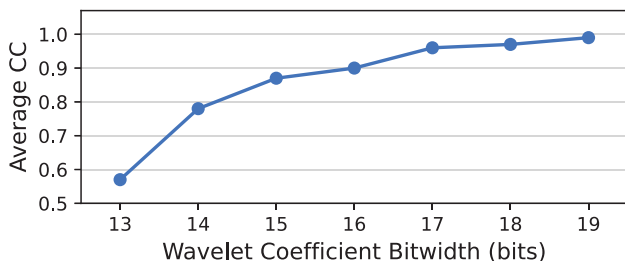


Fig. 4: Average Feature CC vs. Wavelet Bitwidth

A full physical implementation of three AQFE variants was performed to evaluate the impact of parallelism, enabling post-implementation selection of the optimal architecture, which highlights the importance of cross-layer optimization.

IV. EXPERIMENTAL RESULTS

The three variants of the design were implemented in a 22 nm FDSOI technology. We benchmark them using actual ECoG data from a BCI dataset [6], to evaluate their energy consumption and algorithmic performance. The algorithmic performance is assessed by comparing the output features to a floating-point reference model. Finally, we compare the AQFE implementations with SoA feature extractors used in BCIs.

A. BCI System Use Case

For benchmarks, we consider a BCI system with input signals from 64 electrodes, sampled at 590 Hz [10]. Each signal is convolved with 15 wavelets to compute time-frequency components at different frequency bands, resulting in 960 features. The features are extracted at a rate of 10 Hz, to meet the motor decoder requirements [6]. Three quality parameters (**LQ**, **MQ** and **HQ**) corresponding to J values of 4, 18 and 32 are tested to evaluate how the approximation affects the energy consumption, and accuracy.

We evaluate three variants of the architecture: a single-lane (**AQFE-1**), a two-lane (**AQFE-2**), and a four-lane (**AQFE-4**) implementation. The **AQFE-1** variant requires an operating frequency of 50 MHz, while the **AQFE-4** only requires 12.5 MHz to meet the real-time constraint.

B. Physical Implementation and Power Analysis

The physical implementation of the three variants is done using Synopsys DCTM synthesis and Cadence InnovusTM placement tools. Fig. 5 shows the layout of the **AQFE-1**, occupying an area of 0.061 mm². The area is dominated by memory cuts, mainly the *Input* and *Output RAMs*. The designs were implemented with a 0.8 V supply voltage and were simulated at gate-level using SDF, and their power was evaluated using Synopsys PrimePowerTM.

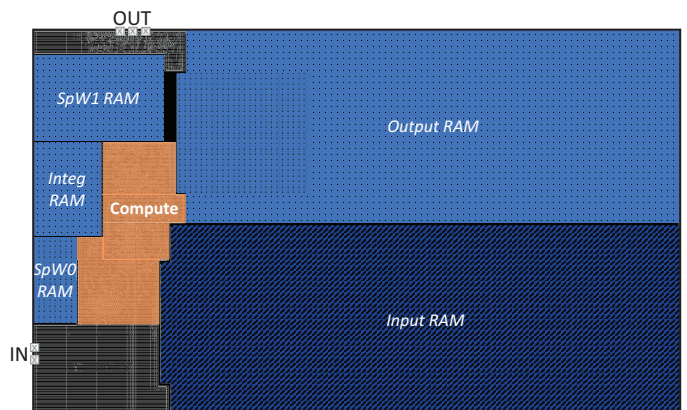


Fig. 5: Post-layout Floorplan of the Single-lane **AQFE-1**

Fig. 6 shows the power breakdown of the AQFE module for three lane configurations at different qualities. When reducing the quality from **HQ** to **LQ**, the overall power decreases, for all configurations. For the **AQFE-1** configuration, at **HQ**, 75% of the power consumption is dynamic, including logic (arithmetic operations and control) and the memory accesses.

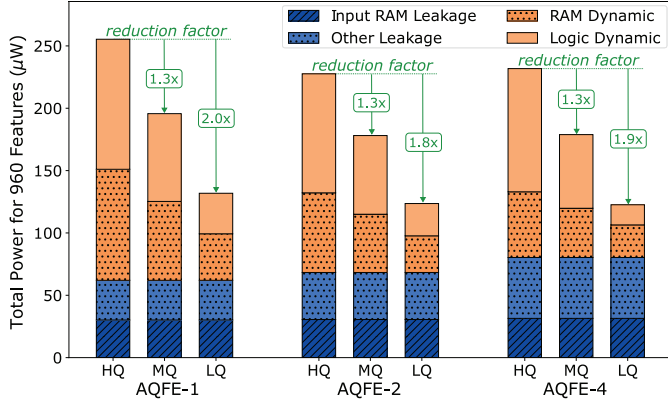


Fig. 6: Power Composition across Lane Configurations for Different Quality Parameters

When reducing the quality, the total power consumption drops from $255\mu\text{W}$ to $128\mu\text{W}$, achieving a $2\times$ reduction.

Multi-lane versions benefit from lower operating frequency, L times fewer RAM accesses, and a lower per-bit energy cost for wider RAMs, reducing the RAM dynamic power consumption. At **HQ**, the **AQFE-2** (resp. **AQFE-4**) consumes 18% (resp. 22%) less dynamic power than the **AQFE-1**. As a result, the multi-lane configurations achieve comparable power consumption to the single lane despite having larger *Integ* and *Output RAMs*, thus higher leakage. More precisely, at **HQ**, multi-lane circuits consume slightly less power than the single-lane, and they are able to reach the same low power at **LQ**.

These findings highlight the trade-off between different levels of parallelism. Multi-lane designs benefit from reduced operating frequency, while the single-lane occupies less area. The total power remains comparable across all configurations, especially when operating at **LQ**. The power reduction achieved through quality control remains effective even when processing up to four lanes in parallel.

C. Comparative Evaluation

1) *Comparison with Software Implementation*: To evaluate the impact of fixed-point custom width quantization, we compare the outputs of the proposed AQFE with a reference implementation in MATLAB R2022, using 32-bit floating-point precision. This reference [16] has been used to validate the decoding accuracy of the PLCWT algorithm on a BCI dataset [6] recorded during exoskeleton motor control. The comparison is performed for different approximation levels using quality parameters between 4 and 32. To quantify the precision loss due to approximation, we compute the Pearson CC between the MATLAB and hardware output features. Depending on the approximation level, the mean CC value over all frequencies ranges from 0.968 to 0.998, confirming that the fixed-point quantization does not significantly affect the accuracy of the extracted features, hence the quality of the decoding.

2) *SoA Feature Extractors*: We compare our circuit with existing SoA feature extractors based on FFT, filters, CWT and other custom methods. As shown in Table II, the power consumption per feature, at high precision, is at least $4\times$ lower

than that of existing digital circuits. The module extracts up to 960 features, $8\times$ more than prior extractors, at a rate of 10 Hz, offering a sufficient throughput for motor decoding. The best implementation (**AQFE-1**) for our use case has a minimal silicon area 0.061 mm^2 , $2\times$ smaller than existing circuits.

TABLE II: Summary of SoA Brain Signal Decoding Hardware Implementations

Ref. (Year)	Ch.	Features		Power ($\mu\text{W}/\text{feat.}$)	Area (mm^2)	Implementation
		Method	Nb. Rate			
2010 [20]	16	Custom	16 10 Hz	1.4	2.1	ASIC (90 nm)
2011 [19]	4	FFT	1 2 Hz	0.99	8.88	ASIC (130 nm)
2014 [23]	7	DCT	42 3.3 Hz	17.1	-	FPGA
2016 [22]	62	PCA	3 2 Hz	50666	-	FPGA
2016 [18]	4	FFT	1 0.5 Hz	34000	100	DSP (90 nm)
2017 [13]	1	CWT	1 1 Hz	47300	1.12	FPGA
2019 [21]	10	Filter	3 1 Hz	8333	-	FPGA
2019 [17]	7	Filter	1 1.3 Hz	150000	49	MCU
2020 [31]	32	Analog	32 -	0.205	0.145	ASIC (180 nm)
2022 [8]	64	Custom	64 4 Hz	1.42	0.12	ASIC (65 nm)
2024 [32]	192	Filter	128 1 Hz	1.68	0.71	ASIC (65 nm)
2025 [33]	32	PAC/PLV	48 -	0.88-1.92	5.46	ASIC (65 nm)
AQFE-1	64	PLCWT	960 10 Hz	0.13-0.27	0.061	ASIC (22 nm)
AQFE-4	64	PLCWT	960 10 Hz	0.13-0.24	0.12	ASIC (22 nm)

Our design achieves a balance between area, energy efficiency, and adaptability. While previous works use static configurations, our architecture supports real-time approximation adjustment to meet dynamic quality requirements. This tuning mechanism can dynamically reduce the power consumption by a factor of $1.8\times$ to $2\times$. Multi-lane implementations reach the same power per feature as the single-lane, while reducing the system clock frequency.

V. CONCLUSIONS

This work introduces ALIFE-BCI, an ASIC implementation of a wavelet-based adaptive time-frequency feature extractor for ultra-low-power BCIs. By leveraging approximate computing and multi-layer optimizations, from algorithmic approximations to architectural and circuit-level techniques, the AQFE consumes $4\times$ less power per feature, compared to SoA digital feature extractors. In addition, when lower quality features are sufficient, dynamic control of the approximation level can further reduce the energy consumption by up to $2\times$, reaching $0.13\mu\text{W}/\text{feature}$.

Future work will explore online closed-loop control using system-level feedback, such as task performance or user satisfaction, to dynamically adjust the approximation quality of features in a BCI system.

ACKNOWLEDGMENTS

The authors thank Sylvain Choisnet for his help with the place and route flow. They also thank Clinatex team members for their help with BCI datasets. This work is funded by the NEMO-BMI project (101070891) that received funding from the European Innovation Council, Pathfinder Challenge.

REFERENCES

- [1] X. Chen, Y. Yu, J. Tang, L. Zhou, K. Liu, Z. Liu, S. Chen, J. Wang, L.-L. Zeng, J. Liu, and D. Hu, "Clinical Validation of BCI-Controlled Wheelchairs in Subjects With Severe Spinal Cord Injury," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 30, 2022.
- [2] H. Wang, J. Guo, Y. Zhang, Z. Fu, and Y. Yao, "Closed-loop rehabilitation of upper-limb dyskinesia after stroke: from natural motion to neuronal microfluidics," *Journal of NeuroEngineering and Rehabilitation*, vol. 22, no. 1, Apr. 2025.
- [3] M. A. Younessi Heravi, K. Maghooli, F. Nowshiravan Rahatabad, and R. Rezaee, "A New Nonlinear Autoregressive Exogenous (NARX)-based Intra-spinal Stimulation Approach to Decode Brain Electrical Activity for Restoration of Leg Movement in Spinally-injured Rabbits," *Basic and Clinical Neuroscience*, vol. 14, no. 1, Jan. 2023.
- [4] J. Saad, A. Evans, I. Jaoui, V. Roux-Sibillon, E. Hardy, and L. Anghel, "Comparison metrics and power trade-offs for BCI motor decoding circuit design," *Frontiers in Human Neuroscience*, vol. 19, Mar. 2025.
- [5] K. C. Davis, K. R. Wyse-Sookoo, F. Raza, B. Meschede-Krasa, N. W. Prins, L. Fisher, E. N. Brown, I. Cajigas, M. E. Ivan, J. R. Jagid, and A. Prasad, "5-year follow-up of a fully implanted brain-computer interface in a spinal cord injury patient," *Journal of Neural Engineering*, vol. 22, no. 2, Apr. 2025.
- [6] A. L. Benabid, T. Costecalde, A. Eliseyev, G. Charvet, A. Verney, S. Karakas, M. Foerster, A. Lambert, B. Morinière, N. Abroug, M.-C. Schaeffer, A. Moly, F. Sauter-Starace, D. Ratel, C. Moro, N. Torres-Martinez, L. Langar, M. Oddoux, M. Polosan, S. Pezzani, V. Auboiroux, T. Aksenova, C. Mestais, and S. Chabardes, "An exoskeleton controlled by an epidural wireless brain-machine interface in a tetraplegic patient: a proof-of-concept demonstration," *The Lancet Neurology*, vol. 18, no. 12, Dec. 2019.
- [7] H. Lorach, A. Galvez, V. Spagnolo, F. Martel, S. Karakas, N. Interling, M. Vat, O. Faivre, C. Harte, S. Komi, J. Ravier, T. Collin, L. Coquoz, I. Sakr, E. Baaklini, S. D. Hernandez-Charpak, G. Dumont, R. Buschman, N. Buse, T. Denison, I. van Nes, L. Asboth, A. Watrin, L. Struber, F. Sauter-Starace, L. Langar, V. Auboiroux, S. Carda, S. Chabardes, T. Aksenova, R. Demesmaeker, G. Charvet, J. Bloch, and G. Courtine, "Walking naturally after spinal cord injury using a brain-spine interface," *Nature*, May 2023.
- [8] U. Shin, C. Ding, B. Zhu, Y. Vyza, A. Trouillet, E. C. M. Revol, S. P. Lacour, and M. Shoaran, "NeuralTree: A 256-Channel 0.227- μ J/Class Versatile Neural Activity Classification and Closed-Loop Neuromodulation SoC," *IEEE Journal of Solid-State Circuits*, vol. 57, no. 11, Nov. 2022.
- [9] K. Matsushita, M. Hirata, T. Suzuki, H. Ando, T. Yoshida, Y. Ota, F. Sato, S. Morris, H. Sugata, T. Goto, T. Yanagisawa, and T. Yoshimine, "A Fully Implantable Wireless ECoG 128-Channel Recording Device for Human Brain-Machine Interfaces: W-HERBS," *Frontiers in Neuroscience*, vol. 12, 2018.
- [10] C. S. Mestais, G. Charvet, F. Sauter-Starace, M. Foerster, D. Ratel, and A. L. Benabid, "WIMAGINE: Wireless 64-Channel ECoG Recording Implant for Long Term Clinical Applications," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 23, no. 1, Jan. 2015.
- [11] W.-k. Tam, T. Wu, Q. Zhao, E. Keefer, and Z. Yang, "Human motor decoding from neural signals: a review," *BMC Biomedical Engineering*, vol. 1, Sep. 2019.
- [12] C. E. Heil and D. F. Walnut, "Continuous and Discrete Wavelet Transforms," *SIAM Review*, vol. 31, no. 4, Dec. 1989.
- [13] A. Chamanzar, M. Shabany, A. Malekmohammadi, and S. Mohammadnejad, "Efficient Hardware Implementation of Real-Time Low-Power Movement Intention Detector System Using FFT and Adaptive Wavelet Transform," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 11, no. 3, Jun. 2017.
- [14] M. Omachi and S. Omachi, "Fast calculation of continuous wavelet transform using polynomial," in *2007 International Conference on Wavelet Analysis and Pattern Recognition*, vol. 4, Nov. 2007.
- [15] A. Muñoz-Barrutia, R. Ertlé, and M. Unser, "Continuous Wavelet Transform with Arbitrary Scales and $O(N)$ Complexity," *Signal Processing*, vol. 82, May 2002.
- [16] J. Saad, A. Evans, I. Miro-Panades, T. Aksenova, and L. Anghel, "Low-Power Feature Extraction for BCI Applications Using Approximate Continuous Wavelet Transform," in *2025 13th International Conference on Brain-Computer Interface (BCI)*, Feb. 2025.
- [17] P. T. Wang, E. Camacho, M. Wang, Y. Li, S. J. Shaw, M. Armacost, H. Gong, D. Kramer, B. Lee, R. A. Andersen, C. Y. Liu, P. Heydari, Z. Nenadic, and A. H. Do, "A benchtop system to assess the feasibility of a fully independent and implantable brain-machine interface," *Journal of Neural Engineering*, vol. 16, no. 6, Nov. 2019.
- [18] P. T. Wang, K. Gandasetiawan, C. M. McCrimmon, A. Karimi-Bidhendi, C. Y. Liu, P. Heydari, Z. Nenadic, and A. H. Do, "Feasibility of an ultra-low power digital signal processor platform as a basis for a fully implantable brain-computer interface system," in *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, Aug. 2016.
- [19] S. R. Sridhara, M. DiRenzo, S. Lingam, S.-J. Lee, R. Blazquez, J. Maxey, S. Ghanem, Y.-H. Lee, R. Abdallah, P. Singh, and M. Goel, "Microwatt Embedded Processor Platform for Medical System-on-Chip Applications," *IEEE Journal of Solid-State Circuits*, vol. 46, no. 4, Apr. 2011.
- [20] T.-C. Chen, T.-H. Lee, Y.-H. Chen, T.-C. Ma, T.-D. Chuang, C.-J. Chou, C.-H. Yang, T.-H. Lin, and L.-G. Chen, "1.4 μ W/channel 16-channel EEG/ECoG processor for smart brain sensor SoC," in *2010 Symposium on VLSI Circuits*, Jun. 2010.
- [21] X. Ma, W. Zheng, Z. Peng, and J. Yang, "FPGA-Based Rapid Electroencephalography Signal Classification System," in *2019 IEEE 11th International Conference on Advanced Infocomm Technology (ICAIT)*, Oct. 2019.
- [22] M. Agrawal, S. Vidyashankar, and K. Huang, "On-chip implementation of ECoG signal data decoding in brain-computer interface," in *2016 IEEE 21st International Mixed-Signal Testing Workshop (IMSTW)*, Jul. 2016.
- [23] M. Won, H. Albalawi, X. Li, and D. E. Thomas, "Low-power hardware implementation of movement decoding for brain computer interface with reduced-resolution discrete cosine transform," in *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, Aug. 2014.
- [24] L. P. A. Arts and E. L. van den Broek, "The fast continuous wavelet transformation (fCWT) for real-time, high-quality, noise-resistant time-frequency analysis," *Nature Computational Science*, vol. 2, no. 1, Jan. 2022.
- [25] M. Garrido, "A Survey on Pipelined FFT Hardware Architectures," *Journal of Signal Processing Systems*, vol. 94, no. 11, Nov. 2022.
- [26] S. He and M. Torkelson, "Design and implementation of a 1024-point pipeline FFT processor," in *Proceedings of the IEEE 1998 Custom Integrated Circuits Conference (Cat. No.98CH36143)*, May 1998.
- [27] C. Bae, S. Lee, and Y. Jung, "High-Speed Continuous Wavelet Transform Processor for Vital Signal Measurement Using Frequency-Modulated Continuous Wave Radar," *Sensors*, vol. 22, no. 8, Jan. 2022.
- [28] M. Shensa, "The discrete wavelet transform: wedding the a trous and Mallat algorithms," *IEEE Transactions on Signal Processing*, vol. 40, no. 10, Oct. 1992.
- [29] N. Arizumi and T. Aksenova, "Fast Continuous Wavelet Transform for Brain Computer Interface using piecewise polynomials," in *2019 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, Dec. 2019.
- [30] M. Horowitz, "1.1 Computing's energy problem (and what we can do about it)," in *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, Feb. 2014.
- [31] O. Malekzadeh-Arasteh, H. Pu, J. Lim, C. Y. Liu, A. H. Do, Z. Nenadic, and P. Heydari, "An Energy-Efficient CMOS Dual-Mode Array Architecture for High-Density ECoG-Based Brain-Machine Interfaces," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 14, no. 2, Apr. 2020.
- [32] M. Shaeri, U. Shin, A. Yadav, R. Caramellino, G. Rainer, and M. Shoaran, "A 2.46-mm² Miniaturized Brain-Machine Interface (MiBMI) Enabling 31-Class Brain-to-Text Decoding," *IEEE Journal of Solid-State Circuits*, 2024.
- [33] D. Alex, A. Yadav, J. Joo, U. Shin, A. Afzal, J. Liu, G. Diehl, A. Widge, and M. Shoaran, "A 32-Channel 196- μ W Logarithmic SoC for Brain Network Connectivity Extraction and Adaptive Psychiatric Symptom Classification," in *2024 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*. IEEE, Jun. 2025. [Online]. Available: <https://infoscience.epfl.ch/handle/20.500.14299/251241>