

Calibration-free and hardware-efficient neural spike detection for brain machine interfaces

Zheng Zhang, *Student Member, IEEE*, Peilong Feng, *Member, IEEE*, Alexandru Oprea and Timothy G. Constandinou, *Senior Member, IEEE*

Abstract—Recent translational efforts in brain-machine interfaces (BMI) are demonstrating the potential to help people with neurological disorders. The current trend in BMI technology is to increase the number of recording channels to the thousands, resulting in the generation of vast amounts of raw data. This in turn places high bandwidth requirements for data transmission, which increases power consumption and thermal dissipation of implanted systems. On-implant compression and/or feature extraction are therefore becoming essential to limiting this increase in bandwidth, but add further power constraints – the power required for data reduction must remain less than the power saved through bandwidth reduction.

Spike detection is a common feature extraction technique used for intracortical BMIs. In this paper, we develop a novel firing-rate-based spike detection algorithm that requires no external training and is hardware efficient and therefore ideally suited for real-time applications. Key performance and implementation metrics such as detection accuracy, adaptability in chronic deployment, power consumption, area utilization, and channel scalability are benchmarked against existing methods using various datasets. The algorithm is first validated using a reconfigurable hardware (FPGA) platform and then ported to a digital ASIC implementation in both 65 nm and 0.18 μm CMOS technologies.

The 128-channel ASIC design implemented in a 65 nm CMOS technology occupies 0.096 mm^2 silicon area and consumes 4.86 μW from a 1.2 V power supply. The adaptive algorithm achieves a 96% spike detection accuracy on a commonly used synthetic dataset, without the need for any prior training.

Index Terms—Spike detection, calibration-free, adaptive threshold, low power, hardware-efficient, ASIC

I. INTRODUCTION

IMPANTABLE Brain Machine Interfaces (iBMIs) have advanced significantly over the past two decades, already demonstrating their utility and potential impact through neuro-prosthetic control [1], as well as decoding of handwriting [2] and speech [3] for communication. Research in iBMI technology is targeting devices with higher channel counts to access more neural data, with the prospect of this achieving a higher information transfer rate and better robustness/longevity through redundancy. Translational and commercialization efforts are also aiming to reduce the level of invasiveness

through miniaturization and avoiding percutaneous connectors, i.e. wired implants.

Wireless connectivity is an essential feature for the widespread clinical adoption of iBMIs to reduce the risk of infection [4]–[6]. However, with an ever-increasing number of channels, the raw data bandwidth required for wireless transmission also increases and can become prohibitive. There, therefore, exists a real need for efficient and effective on-implant processing to reduce the amount of data through data compression or feature extraction.

The neural firing rate is a commonly used feature [4]–[7] that distills important information related to subject behavior or intention from the observed intracortical signals. By using this feature, the required data bandwidth can be significantly reduced, in turn resulting in reduced transmission power requirements [8]. The firing rate feature is often obtained by counting the number of spikes fired in a given period from either a single neuron (Single-Unit Activity, SUA) or multiple neurons (Multi-Unit Activities, MUAs). The occurrences of these spikes are captured using spike detection algorithms. These algorithms typically work as follows: they first remove any remaining Local Field Potential (LFP) component that typically escapes the passband of the analog front-end. They then use a number of different techniques to emphasize the spikes and/or suppress the noise in order to enhance the signal-to-noise ratio (SNR). A threshold is then established as a reference for spike detection. Any signal passing above this threshold triggers a spike detection – used as the key input feature for brain signal decoding.

Although SUA features are highly informative (encoding the activity of single neurons), and often used in scientific studies (e.g. understanding underlying neural circuit dynamics), they require additional processing (i.e. spike sorting). For iBMI applications however, the consensus is that MUA features are sufficient and do not lead to significant degradation of decoding performance [9]–[11]. This is particularly important considering the stringent power requirements for any implantable hardware – both SUA and MUA features provide a similar level of data reduction, but spike sorting (for extracting SUA features) is significantly more computationally intensive than spike detection (for extracting MUA features).

Various spike detection algorithms have been proposed for systems. Unlike offline algorithms, on-implant signal processing algorithms must be operable in real-time and be hardware-efficient (i.e. both low power and low resource) to ensure a fast response and that the on-implant processing power does not exceed the transmission power reduced by bandwidth

Z. Zhang, P. Feng, A. Oprea and T. G. Constandinou are with the Department of Electrical and Electronic Engineering, Imperial College London, South Kensington Campus, London SW7 2AZ, UK (e-mail: {zheng.zhang18, peilong.feng14, alexandru.oprea18, t.constandinou}@imperial.ac.uk).

Z. Zhang, P. Feng, and T. G. Constandinou are also with the UK Dementia Research Institute, Care Research & Technology Centre at Imperial College London and University of Surrey).

P. Feng and T. G. Constandinou are also with Mint Neurotechnology Ltd, London, UK.

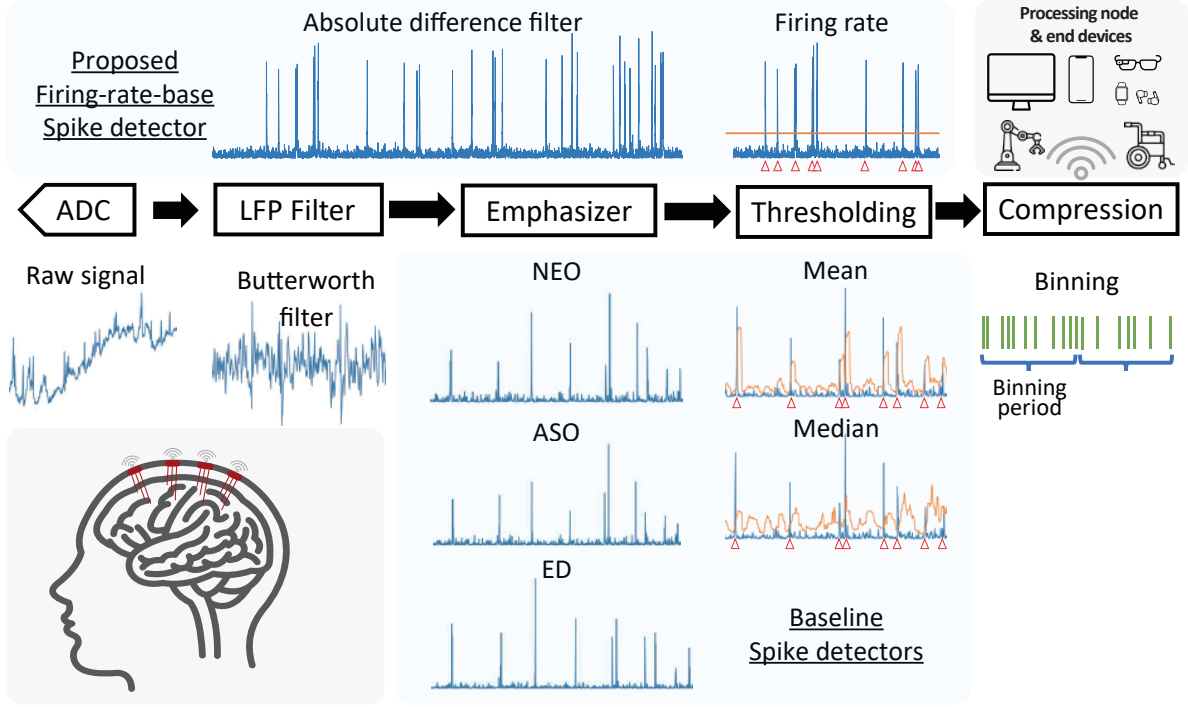


Fig. 1. An overview of on-implant signal processing: the intracortical signal, after digitization, is low-pass filtered to remove the local stationary LFP offset. The resulting signal is then ‘emphasized’ to enhance the SNR. A threshold is then set to detect this emphasized signal, with the threshold crossings serving as the features to be transmitted from the implant. To reduce data bandwidth, various compression techniques can be employed. The proposed spike detection method utilizes an absolute difference filter to improve the SNR, and incorporates firing rate information to set the threshold. The proposed spike detector is compared to three other baseline emphasizees (NEO, ASO, ED) and two threshold mechanisms.

reduction. Template-matching-based [12], [13] and wavelet-based spike detection [14], [15] achieve good spike detection performance, but their complexity makes them relatively inefficient for wireless implantable BMIs. Simple spike detection algorithms are most commonly used with thresholds defined using signal statistics, for example, using the local mean to capture a level that is somewhere between the spike and noise levels [16]. However, the local mean value-based threshold can be overly sensitive to changes in spike activity resulting in unstable thresholds. Some studies have tried eliminating the impact of spikes for a more reliable threshold [17], [18]. Median can also be used for a similar purpose, but calculating the median value often requires a large buffer making it inefficient for on-implant use. Rooted-Mean-Square (RMS) values, standard deviations (STD) and cross-correlations [19] are also used in some studies but are not preferred for implants due to their high hardware cost. There have also been works on estimating RMS and STD values resulting in reduced hardware complexity despite needing multiple cycles to converge, leading to bottlenecks in throughput. In combination with statistics-based thresholding, multiple pre-processing operators have been proposed to enhance the signal SNR. Nonlinear Energy Operator (NEO) is the most widely used operator [20]. Following NEO, multiple variants have been proposed, such as smoothed NEO [21], multi-resolution NEO [22], Amplitude Slope Operator (ASO) [17] and Energy of Derivative (ED) [23], achieving better performance and/or reduced computational

complexity.

Beyond conventional statistical methods for defining the detection threshold, our earlier work proposed using a firing-rate-based spike detection algorithm [24]. Here, instead of manually defining the threshold based on training data, a target detection rate is set, and the threshold adjusts toward maintaining that target. This is based on the assumption that neurons in a certain brain region over time maintain an average spike rate, and when the detection rate meets this target, the threshold is expected to be reliable. This work also established a mechanism to automatically update the target rate, reducing the sensitivity to an inaccurately selected initial value. More generally, this work demonstrated that relatively simple methods can perform at least as well as statistics-based methods, but without requiring a training history, and can provide good robustness to different or varying signals.

Although there do exist several spike detection algorithms that have been previously reported in the literature, few compare spike detection performance from a computational or hardware perspective (unless a full custom circuit). In [25], [26], several spike detection algorithms are assessed and minimum requirements are determined for optimizing hardware implementation. In [27], the detection and decoding performance of different algorithms are compared, and the hardware cost of the best-performing algorithm is presented. We also recently reported [28] an FPGA-based hardware comparison of common statistical-based spike detection algorithms

previously reported in the literature.

The work presented herein builds on our earlier work [24], [28] by further optimizing the firing-rate-based spike detection algorithm to adapt to signals of varying dynamics (SNR and nominal firing rate) whilst also improving the efficiency of the hardware implementation. This is then evaluated on a variety of different datasets, including both synthetic (with known ground truth) and real recordings (with more realistic signal dynamics) taken with different electrode technologies (e.g. Utah array, Neuropixels), across different animal models, neural targets and test subjects (rodent visual cortex, non-human primate motor cortex). By testing across this set of diverse recording setups we demonstrate both the robustness of our technique to different signal dynamics and to varying signal dynamics through testing chronic datasets. From the circuit and system standpoint, we then demonstrate the algorithmic equivalence of the proposed algorithm using a reconfigurable hardware (FPGA) implementation. We then port to a standard digital ASIC implementation in two different CMOS technology nodes (65 nm and 0.18 μ m) to quantify hardware efficiency both in terms of area utilization and power consumption. We then compare our results to the state-of-the-art as previously reported. An overview is given in Fig. 1

The remainder of this paper is organized as follows: Section II reviews selected low-complexity statistical-based spike detection algorithms and their hardware implementation; Section III introduces the proposed firing-rate-based threshold algorithm and the hardware implementation; Section III describes the experimental setup and datasets used in this work; Section IV compares the spike detection algorithm and decoding performance of different algorithms; Section V compares the resource utilization and power consumption for an FPGA target, and also area utilization and estimated power consumption on an ASIC target. Finally, Section VI concludes this work.

II. BASELINE HARDWARE-EFFICIENT SPIKE DETECTION ALGORITHMS

We regularly see new advances reported in spike detection algorithms, including different pre-processing, feature extraction methods, and threshold mechanisms. These are however not all directly suitable for real-time application, particularly for on-implant use due to their computational complexity and hardware requirements such as silicon area and power budget. We have selected some of the more commonly used methods that are amenable to hardware implementation using fixed-point representation, with minimal requirements for hardware-intensive operations such as multiplications. This section introduces the selected methods that are illustrated through a high-level functional schematic shown in Fig. 2. In the interests of hardware efficiency, we have constrained the sampling rate to 7 kHz and data resolution to 10 bits, as previous work shows this is sufficient for detection purposes [25]. Although the 7 kHz sampling frequency used in this study is typically lower than that of the vast majority of the electrophysiology research tools. As most of the spike energy is below 3 kHz, 7 kHz is however sufficient to capture the spike event, according to the

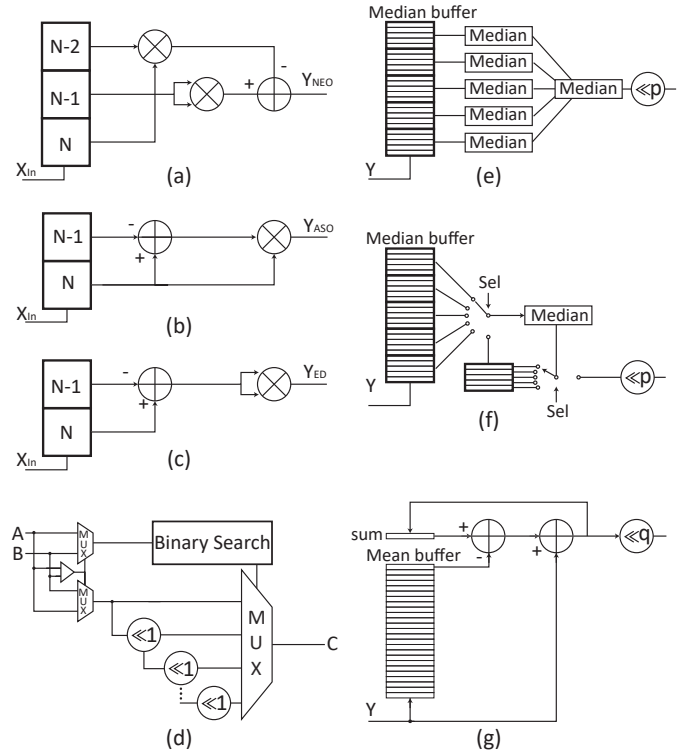


Fig. 2. The circuit diagram of baseline emphasizeers and thresholding modules. (a) NEO (b) ASO (c) ED (d) Shift-based multiplier for product operation (e) Unrolled median recursion (f) Rolled median recursion (g) Accumulated mean

Nyquist Theorem with 1 kHz oversampling. For applications like spike sorting where spike waveform information is essential, a higher sampling rate is desirable. Nonetheless, previous works have shown that a 7 kHz sampling rate is sufficient for accurate spike detection applications [25], [29].

A. LFP filters

There are several studies (e.g. [25], [27]) that investigate the most efficient filter topologies for rejecting the local field potentials (LFPs). Generally, a linear phase filter is considered ideal, as this does not introduce any phase distortion and therefore the spike shape is preserved. Phase distortion is especially critical for applications requiring single unit activity, as spike sorting relies on feature differences in the spike shape dynamics. However, as it has been shown [2], [3], [9], [26] that neural decoding for iBMIs can be achieved effectively using only threshold crossings (i.e. MUA data), a linear phase response is not critical, whereas filter complexity is important. Therefore, for our baseline method, a simple two-pole Butterworth filter is used as recommended in [25] to be shared for all channels.

B. Emphasizers

Using an emphasizeer to enhance the SNR of the LFP-removed signal is critical for improving detection performance, in particular for low SNR recordings.

The circuits of the selected emphasizeers are shown in Fig. 2(a)-(c). NEO in (1) and its variations are the most widely used emphasizeers. ASO [17] and ED [23] both used one less multiplication to emphasize the spikes and were reported to outperform NEO, so they were chosen for this work given in (2) and (3).

$$Y_n^{NEO} = X_n^2 - X_{n-1}X_{n+1} \quad (1)$$

$$Y_n^{ASO} = X_n(X_n - X_{n-1}) \quad (2)$$

$$Y_n^{ED} = (X_n - X_{n-1})^2 \quad (3)$$

Absolute values are taken after the operation allowing the algorithm to detect both positive and negative peaks with a single threshold.

Other variations may necessitate smoothing filters, deeper buffers, or multiplexing between different emphasizeers based on detected noise levels. Although the detection performance can be improved, this comes with an additional hardware burden, and thus are less attractive for efficient on-implant use. As such we have chosen to exclude these more advanced methods from our study.

In order to optimize our hardware implementation, we chose to target a multiplication-free system, but instead estimate multiplication using bit shift according to (4) as shown in Fig. 2.(d).

$$\begin{aligned} C &= B * A \\ &= \sum^i \{B \ll [(i-1) * b_i(A)]\} \\ &\approx B \ll (\text{MSB}(A) - 1) \end{aligned} \quad (4)$$

where $A > B$, b_i stands for the binary value of number at i -th digit and MSB finds the index of the most significant bit.

Instead of multiplying two values, the larger value is left-shifted by N bits, where N is the index of the most significant bit of the smaller value. A binary search is adopted to find the most significant bit of one number. The circuit is shown in Fig. 2.(d).

In order to test whether such an estimation is more efficient than using a look-up table (LUT) to implement a multiplier for product operation, we have implemented three different multipliers: 1) Digital Signal Processor (DSP) Multiplier 2) LUT Multiplier 3) Bit shift multiplier. These different implementations are compared both from algorithmic and hardware perspectives.

C. Thresholding

Thresholding is the most important step in spike detection. Most commonly, the threshold is defined using signal statistics – including the mean, root-mean-square (RMS)/standard deviation (STD), and median values.

The mean value is the most preferable for hardware systems due to its simplicity. Instead of summing all values using a buffer to capture a training history, it is preferred the mean is calculated incrementally using an accumulator as shown in Fig. 2.(g). The sum of the buffer is stored, the oldest value is subtracted from the mean, and latest value is added to the result. Such an operation can be completed in a single clock cycle requiring only two adders, reducing the hardware cost

to a minimum. The threshold is then calculated through bit shifting operation.

The median is however often preferable to the mean because it is less affected by outliers (spikes), but it is inefficient on hardware due to its $O(n^2)$ computational complexity. To address this shortcoming, we proposed using median recursion to estimate global median values from subset medians in [28], reducing the complexity to $O(n \log(n))$. In further detail (see Fig. 2(e,f)), to calculate the median of 25 numbers, we divide all values into five subsets and take the median value of each subset. Finally, we use the median of these medians as the output. Although this does not guarantee to find the real median value, it is a close enough estimate to provide a reliable threshold.

Three different median implementations are adopted in this work: (1) Normal median; (2) rolling median recursion: the subset median search logic is shared across the subset; and (3) rolling median recursion: where each subset has its own median search logic.

We have chosen to exclude the RMS pre-operator (or RMS estimation [29]) in our comparison due to the high computational cost. This makes it less desirable for on-implant use. It should be noted however that there does exist past work that implements this using analog circuits.

D. Compression

Instead of using a pulse stream output, i.e. the raw threshold crossing binary output, several studies use binning to define a spike rate that is then used for neural decoding [10]. This is shown to both reduce data bandwidth requirements whilst also improving decoding performance. The binning period is typically of the order of 1 to 50 ms. It is however essential to mitigate for detecting the same spike multiple times. This is achieved by setting a refractory period of 1 ms. The count will not increase until the end of this refractory period. More advanced compression can be applied to reduce the data bandwidth [30] further.

Different emphasizeers and thresholding mechanisms are investigated in this work, while the filters are shown to have less impact on spike detection [27], and compression algorithms have been compared in [31].

III. PROPOSED ULTRA HARDWARE EFFICIENT SPIKE DETECTION ALGORITHM

Although the previously described baseline algorithms are used widely, there do exist several drawbacks. Firstly, there is no heuristic way to find optimized multiplications such as to reliably estimate the local statistics. The standard approach is searching through trial and error using synthetic data. It is however not guaranteed that selected parameters will be generalized well if applied to real recordings, in particular with changing noise levels. In fact, in [32], we found that using statistics-based thresholding methods, it is hugely challenging to establish a reliable threshold that can adapt to different noise levels. Another problem is that calculating local statistics can require significant memory to maintain sufficient history – memory requirements increase linearly with the number

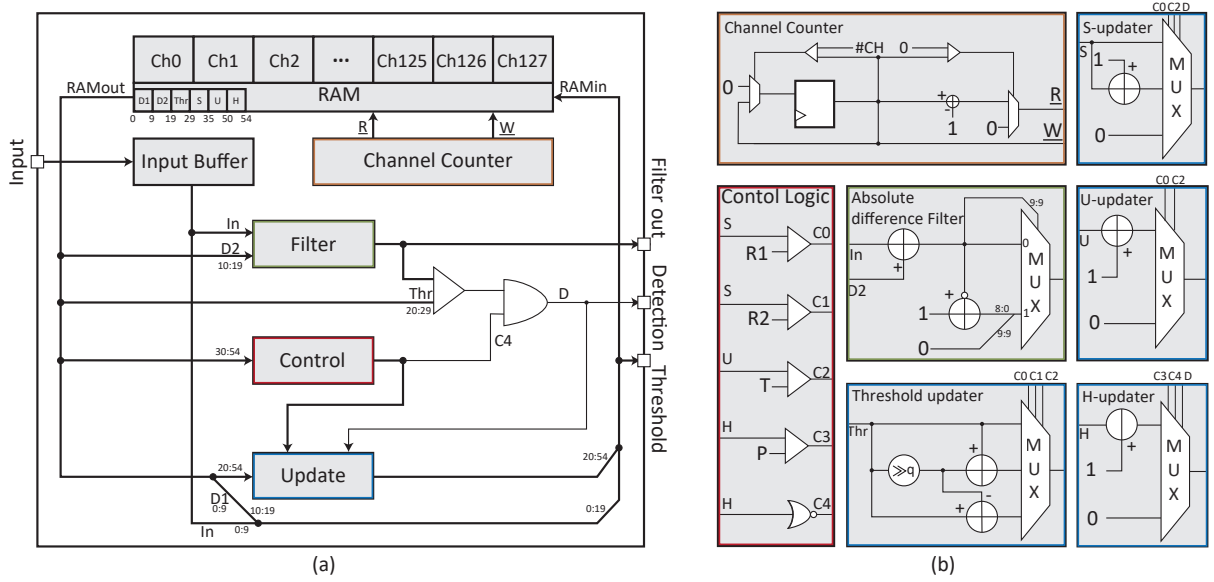


Fig. 3. A diagram for the proposed spike detector: (a) The system overview. The system comprises an input buffer that temporarily stores the input samples, a RAM that maintains the status of various channels, a channel counter that rotates among channels, a filter that improves the signal-to-noise ratio (SNR), a control unit that updates the RAM status, and a control unit that generates the control signals for threshold updating. The filter output, threshold, and detection binary stream are the final outputs from the system; (b) Detailed digital logic for different modules.

of channels and sampling frequency. Finally, [26] suggests that the neural decoding performance is quite robust to the spike detection algorithm missing small spikes (i.e. there is only a small reduction in accuracy). Fewer spikes detected mean fewer data to be communicated. Statistics-based spike detection is also sensitive to the scaling factor that is used to map local statistics to the threshold level – it is not clear how to balance the scaling factor to intentionally detect fewer spikes and trade-off accuracy with power consumption.

To avoid these challenges in conventional methods, we propose a firing-rate-based spike detection algorithm, originally reported in [24]. Here, instead of needing an arbitrary scaling factor to define the threshold from the local statistics, the proposed algorithm only requires a detection rate interval. We hypothesize that the threshold is reliable if it detects spikes at a rate close to the actual local firing rate. The target detection rate can therefore be set according to neuroscience observations heuristically. For example, it has been reported in [33] that the firing rate of human motor neurons varies from a nominal rate of about 10 Hz when the subject is performing minimum effort tasks, to over 20 Hz for maximum effort tasks. Additionally, it has been reported that on average 3 to 4 neurons can be observed per Utah array electrode in motor cortex [34] (i.e. through spike sorting efforts). Thus, the average firing rate of the ticker reaching task (low effort) is expected to be around 40 Hz. A target detection rate interval can then be set to, for example, 30 to 60 Hz, and the threshold is dynamically updated to maintain the detection rate within this interval. In the meantime, the firing-rate-based algorithm can intentionally ‘miss’ spikes (i.e. set a slightly lower detection rate) for better power efficiency.

We first reported our hardware-efficient firing-rate-based spike detection method in [24]. In this work, we further

TABLE I
BIT WIDTH AND DESCRIPTIONS FOR THE VALUES STORED IN RAM

	Width	Description
D1	10	ADF buffer for the first previous sample
D2	10	ADF buffer for the second previous sample
Thr	10	Current threshold
S	7	Number of detections within the current duty cycle
U	13	Count for the update duty cycle
H	3	Count to hold after detection avoiding redetection

improve its performance and simplify it to use fewer resources and power. The circuit schematic and timing diagrams are shown in Figs. 3 and 4 respectively. This consists of four key blocks: (1) the system status memory; (2) the input buffer and pre-processing filter; (3) the control; and (4) update logic.

A. RAM

The system status random access memory (RAM) stores the current status of each channel, and the channel counter schedules channels sharing the same combinational logic. The proposed algorithm only needs one clock cycle to process each sample, so there is no need for any register except the memory output buffer.

The RAM is a single-clock, dual-port block RAM. On each rising clock edge, the memory stores the updated status of the current channel read from the combinational logic and outputs the current status of the next channel into the output buffer. The channel counter updates the current and next channel addresses for the memory cyclically. Table I lists the various statuses stored in memory. This requires a total of 53 bits per channel.

B. Absolute difference filter

As first proposed in [24], both the LFP signal removal and spike signal emphasizing are achieved using the Absolute Difference Filter (ADF) as (5).

$$Y_n = \text{abs}\{X_n - X_{n-k}\} \quad (5)$$

where $k=2$ for a sampling rate of 7 kHz.

The ADF operates by reading the new input sample together with previous samples from a rolling buffer. The filter output is obtained by subtracting a previous sample (two samples back) from the new sample. If the difference is negative, the absolute value is calculated according to a two's complement representation.

No additional digital filters are used to remove the LFP. In practice, the LFP is mostly rejected using a high-pass or band-pass filter in the analog front-end. The digital filter is used to remove the residual LFP component that may have leaked into the passband from the front-end. Hence, a simple difference operation is sufficient.

The filter output is compared with the threshold value stored in RAM. When the filter output exceeds the threshold, and no spike is detected in five samples before, a valid spike detection signal is generated.

C. Control logic

The control unit reads three values (S, U, and H) and generates five control signals. C0 - C4, according to the logic circuits plotted in the control logic block. C0 and C1 control two conditions for the threshold to update. C0 is set when the current number of detections exceeds R1, and C1 is high when the current number of detections is below R2. C2 is high at the end of one update duty cycle. C3 and C4 control the signal entering and exiting the period when a spike is detected. C3 is high when it reaches the preset length of a spike after a spike is detected, and C4 is high when H is zero, i.e. not in a period of the presence of the spike.

D. Update logic ('updaters')

Four updaters update the values of Thr, S, U and H. When the number of detections S within the current duty cycle exceeds the detection rate upper limit R1 (C0 is high), the threshold is increased by $2^{-q} * \text{Thr}$ (q is four). When S is below the lower limit of R2 (C1 is high) at the end of each duty cycle (C2 is high), the threshold is reduced by $2^{-q} * \text{Thr}$. Otherwise, the threshold stays unchanged for the next duty cycle. This threshold update rule is different from where this firing-rate-based algorithm is proposed in [24]. The threshold is originally increased or decreased by half. We have found that finer grain control in the threshold update scale can provide more accurate threshold levels. We have additionally removed the block allowing for the target detection rate to update automatically as this added some hardware complexity, but does not significantly improve the performance.

When a validation spike is detected (D is high), the S is increased by one as long as it does not reach R1 (C0 is high), while it does reach the R1 or one duty cycle is over (C2 is

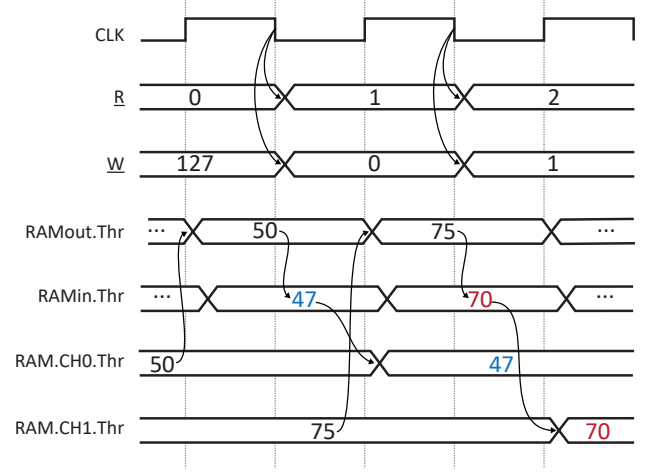


Fig. 4. Threshold update timing diagram. The memory read/write addresses are updated on each falling clock edge. On each rising clock edge, the previous threshold value (50) is retrieved from the RAM based on the read address. The new threshold (47) is calculated using combination logic. At the next rising clock edge, the updated threshold is stored in RAM, overwriting the previous value.

high), S is reset. There is a special case when a valid spike is detected at the end of one duty cycle. In this case, S is set to 1. In all other cases, S stays unchanged.

U is increased by one every clock cycle and reset when the threshold needs to be updated, i.e. C0 is high, or C2 is high. H is increased by one when a D is high or when C3 is low, and C4 is high. Otherwise, it stays at zero.

The new values are concatenated as follows: [Input, D1, new Thr, new S, new U, new H] replacing the old values [D1, D2, S, U, H] of the current channel stored in RAM. The values for R1 and R2 have been set to 30/60 according to the expected firing rate of the recordings as stated at the start of the Section.III. As a spike typically lasts for 7 samples (1 ms), skipping 5 samples after detection can effectively avoid re-detection while maintaining sensitivity. P is thus set to 5. The threshold updating duty cycle is empirically set to 1 s. T is thus 7000.

E. Timing

The hardware implementation is designed such that a single spike detection system can serve multiple channels through multiplexing input and maintaining channel-specific memory. Therefore, a 128-channel spike detection system operating at a 7 kHz sampling rate would require a system clock speed of 896 kHz (i.e. 1 clock cycle is needed for each independent sample). As an example, Fig.4 shows a timing diagram illustrating how the threshold value in RAM can be updated for channels 0 and 1. D1, D2, S, U, and H are updated following a similar procedure.

At the CLK posedge, the threshold value 50 is read from RAM location R(CH0) as RAMOut.Thr. Here we suppose the threshold is to be decreased by 1/16. The threshold updater updates RAMOut.Thr as 47 into RAMIn.Thr. At the next CLK negedge, the new R/W channel addresses will be updated. At the coming CLK posedge, the RAMIn.Thr is

writes into the $W(CH0)$ position of RAM as $RAM.CH0.Thr$. In the meantime, the threshold value of CH1 is read into the RAM output buffer and the threshold value will be updated accordingly.

F. System scalability

As the proposed spike detection system is efficient both in terms of hardware complexity (number of logic gates, memory) and computation (only 1 clock cycle is needed per input sample), this implementation is easily scalable to higher channel counts. The number of channels supported can be increased by linearly increasing the RAM and clock frequency. The LUTs usage is less affected because all channels share the same filter, control logic, and updaters in a time-sharing manner. Without considering the on-implant area and power constraints, the system can be scaled up to N channel as long as (6) is satisfied.

$$T_{delay}(RAM_{out}, RAM_{in}) + T_{RAM} < \frac{1}{7000 * N} \quad (6)$$

where $T_{delay}(RAM_{out}, RAM_{in})$ is the propagation delay from RAM_{out} to RAM_{in} and T_{RAM} is the delay of RAM read and write. The system can be easily scaled up to more than 1000 channels (7MHz), without violating timing constraints.

IV. SPIKE DETECTION PERFORMANCE

As previously mentioned this work further develops the firing-rate-based spike detection originally introduced in [24] by optimizing further for hardware implementation. Our previous work implemented an additional feedback loop, i.e. to adjust for target spike detection rate to provide good robustness to inaccurately defined initial values. Although for optimal spike detection performance, this work requires a better estimated target detection rate, this work achieves 1% higher spike detection accuracy, significantly improves long-term detection stability, and reduced hardware resources by 67%.

In this section, we compare the firing-rate-based spike detection with the baseline statistical-based spike detection algorithms w.r.t spike detection accuracy and adaptiveness on the synthetic dataset. We also compare the neural decoding performance, and long-term detection stability on the Utah array recording. The firing-rate-based spike detection is also validated on Neuropixel recordings.

All algorithms are validated using MATLAB R2020a (v9.8) for detection performance evaluation. To ensure the MATLAB algorithm is equivalent to a hardware implementation (referring to both FPGA and ASIC), we defined all parameters to use 10-bit integers and ensured all arithmetic functions use fixed point operations.

A. Datasets

This work uses three datasets to validate the algorithm, one of which is a commonly used synthetic dataset [35] and two based on real recordings (Utah electrode recordings [34] and Neuropixels recordings [36]).

1) *Synthetic dataset*: The synthetic dataset is first generated in [35] and widely used for evaluating spike detection and spike sorting algorithms [17], [24], [27]. This dataset simulates neuron activities using a Poisson process. The average firing rate is 20 Hz per neuron, and it simulates three neurons. It has 16 signals divided into four groups: Easy1, Easy2, Difficult1, and Difficult2. Within each group, the noise standard deviation to spike peak ratio is 0.05 to 0.2 with a step of 0.05. The sampling frequency of the original recordings is 24 kHz. It is downsampled to 7 kHz on MATLAB to align with the hardware implementation.

2) *Utah array recordings*: Another dataset is a real recording publicly available in [34]. This was collected on the Motor Cortex of a non-human primate using a Utah array when the subject was operating a ticker simultaneously. The finger movement displacement and velocity were also recorded. It contains 5 hours of recordings over 200 days, from June 27, 2016, to January 13, 2017. The recording is collected at 24,414 Hz and is downsampled to 7 kHz using MATLAB for our experiment.

Since there is no ground truth, quantitatively evaluating spike detection on real recordings is challenging. Instead of explicitly evaluating the spike detection performance, we decode the spike detection result from the raw recording and use the decoding performance to evaluate how much useful information is preserved after spike detection. This allows us to evaluate the spike detection performance on real recordings implicitly. Along with this dataset, threshold crossings are provided, which are obtained using 3-5 times STD values. The STD-based spike detection is expected to have a more accurate detection performance than the mean or median ones. Therefore, we used the provided STD-threshold crossings to compare with the threshold crossings from the proposed algorithm on their decode accuracy.

The threshold crossings are binned at 50 ms. Two decoding models are used to decode such features. The first model is Wiener Cascade Filter used in [37]–[39]. A linear Wiener filter is first fitted with Lasso regression between the spike crossings and kinematic data with a regularisation factor of 0.01. Then a third-order polynomial is fitted between the Wiener filter and the kinematic data to add the nonlinearity to the model. Another model is an LSTM following [10]. It contained 150 cells and was trained with Adam optimizer at a learning rate of 0.0035, and the batch size is 64.

3) *Neuropixel recordings*: This dataset was obtained from the rodent visual cortex and collected by Cortex Lab at University College London using Neuropixel probes. Each shank of the probes contained 384 channels, with each channel sampled at 30 kHz. The dataset is publicly available at [36]. The recordings were also downsampled to 7 kHz using MATLAB before being processed by the spike detection algorithm.

B. Evaluation metrics

Three metrics: Sensitivity (Sens), False Detection Rate (FDR) and Accuracy (Acc) are used to evaluate the

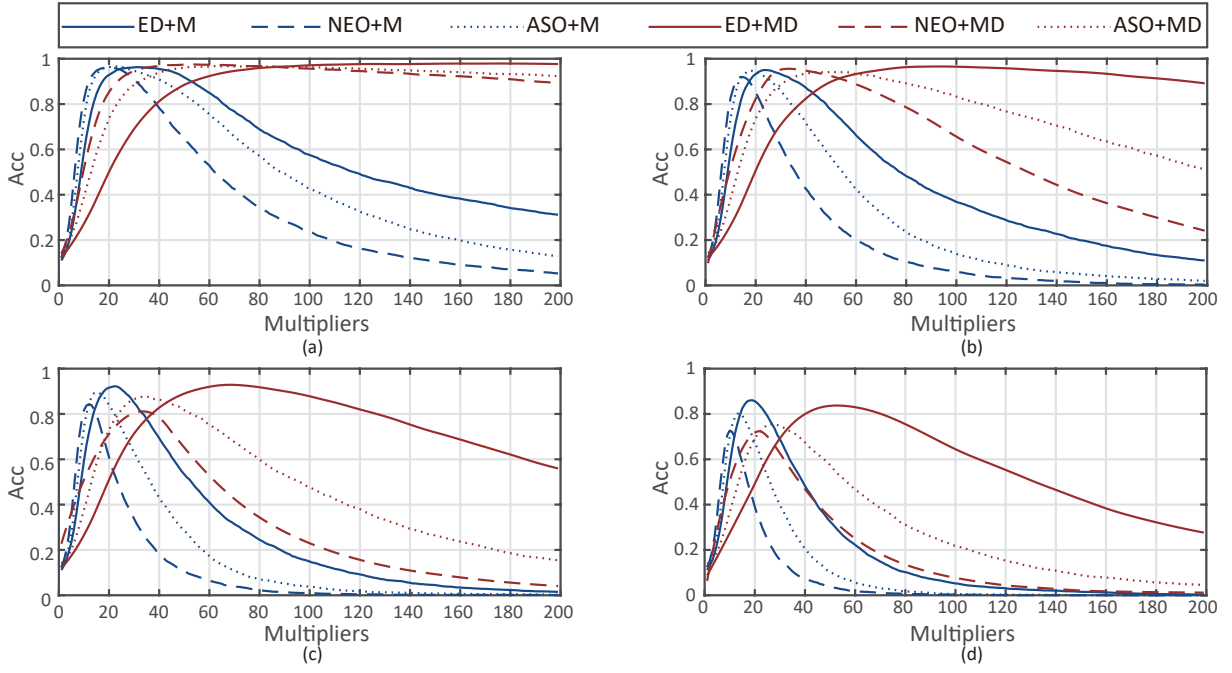


Fig. 5. The performance of different combinations of the baseline methods. M refers to mean thresholding and MD refers to median thresholding. MD typically archives higher accuracy and flatter curves, especially when combined with ED. Four subfigures are the accuracy at different noise levels. (a) 0.05 (b) 0.1 (c) 0.15 (d) 0.2

spike detection performance quantitatively in the synthetic dataset. The formula is given in (7) - (9).

$$\text{Sens} = \frac{TP}{TP + FN} \quad (7)$$

$$\text{FDR} = \frac{FP}{TP + FP} \quad (8)$$

$$\text{Acc} = \frac{TP}{TP + FP + FN} \quad (9)$$

A True Positive (TP) refers to the count of spikes that have been accurately identified by the algorithm. A False Positive (FP) represents the number of instances where the algorithm mistakenly identified non-spikes as spikes. Lastly, a False Negative (FN) is the number of actual spikes that went undetected by the algorithm.

The metric Sens describes the capability of the algorithm in refining the spike detection, while FDR measures its ability to filter out the noise. It is not possible for an algorithm to perform optimally on both metrics at the same time. Acc serves as a compromise between Sens and FDR, and is the primary metric used in this study.

The performance of the decoding on a real dataset is evaluated by computing the correlation coefficients (CC) between the ground truth finger velocity and the prediction. This approach indirectly assesses the effectiveness of spike detection in a practical setting. CC is formulated as (10)

$$\text{CC} = \frac{\sum_{t=1}^N (Y_t - \bar{Y})(\hat{Y}_t - \bar{\hat{Y}})}{\sqrt{\sum_{t=1}^N (Y_t - \bar{Y})^2} \sqrt{\sum_{t=1}^N (\hat{Y}_t - \bar{\hat{Y}})^2}} \quad (10)$$

where Y_t and \bar{Y} are the true target velocities at timesteps t and the average, \hat{Y}_t and $\bar{\hat{Y}}$ are the predicted velocities and the average. N is the total number of samples in this trial. CC is measured on both the x and y-axis. The models are evaluated using 10-fold validation, and the final CC is obtained by taking the average of the CCs from all ten folds.

C. Baseline algorithm performance

Fig. 5 is the average detection accuracy of different combinations of ED, NEO or ASO, and 16-point mean or 25-point median at different levels, obtained from tests on the synthetic data. The 16/25 buffer thresholds buffer size is considered to consume acceptable among resources for on-implant use. A detailed analysis of the emphasizeers and thresholding mechanisms is given below.

1) *Emphasizers*: Based on Fig. 5, ED (solid lines) is the most accurate emphasizeers among the three baseline methods at all noise levels for both thresholding mechanisms. ED is also the most adaptive emphasizeer in terms of multiplier choice robustness (flatter curves).

ASO (dotted lines) has comparable peak Acc, but its performance decreases faster with increasing noise levels, while the performance of NEO (Dash lines) degrades the most rapidly. The observation above implies that the gradient is more discriminative than amplitude in spike emphasizing, particularly as noise levels rise and ED becomes more effective than NEO or ASO.

2) *Thresholding*: The performance of the estimated median (MD, Red lines) and the Mean (M, Blue lines) is comparable in terms of the highest detection Acc they can achieve. However, a typical shortcoming of statistically-based thresh-

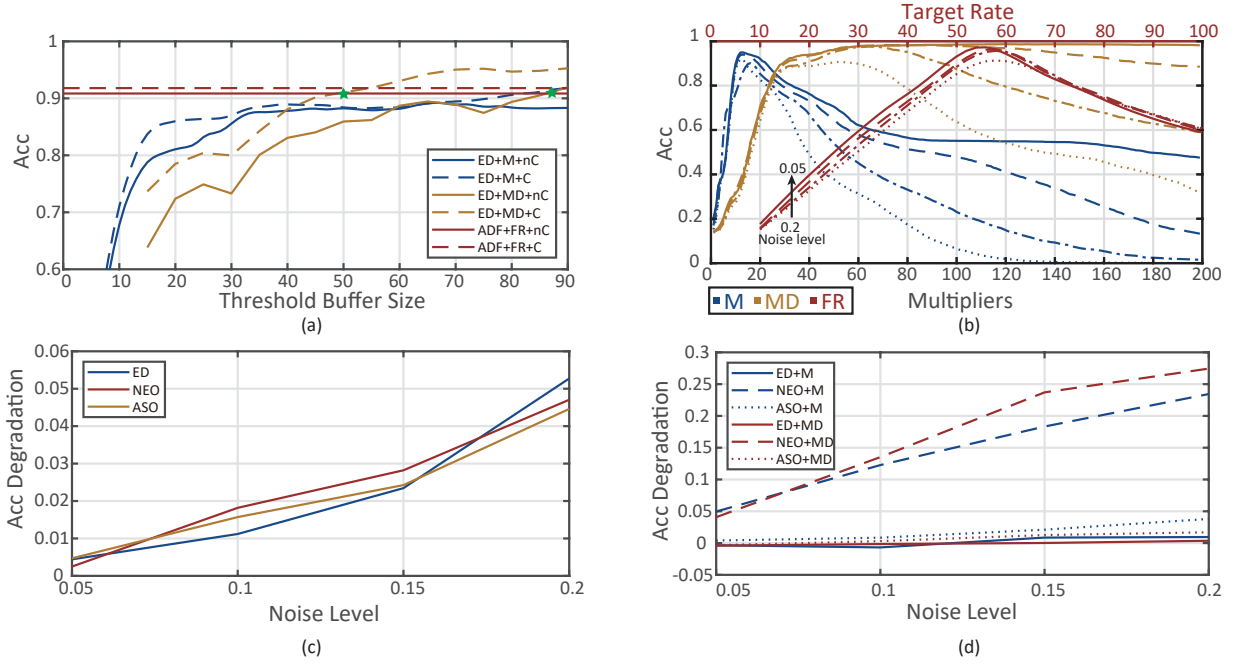


Fig. 6. (a) The spike detection accuracy is evaluated with (C) and without calibration (nC). The results show that the ADF+FR algorithm demonstrates the best performance when no calibration is available. Conversely, the performance of the ED+MD algorithm deteriorates significantly without calibration. Two green stars indicate the intersection points of the baseline algorithms to the proposed algorithm. (b) The spike detection accuracy is analyzed at the intersection points. Results indicate that the FR algorithm displays the greatest adaptiveness to varying noise levels, whereas the MD performance is the weakest. This highlights why calibration is necessary for M or MD algorithms, whereas it is not for the FR algorithm. (c) Performance degradation when using median recursion to estimate median values. (d) Performance degradation when using shift-based multipliers to approximate the product.

olding is that multiplier selection might impact detection performance, requiring manual threshold calibration in practice. As seen in Fig. 5, the median is more robust to the choice of multiplier since it produces a flatter curve nearer to its peak. This characteristic is particularly apparent around 0.05 noise level when the method can nearly always achieve high Acc. In contrast, when utilizing the mean, the Acc degrades rapidly as the multiplier moves away from the maximum values. The median outperforms the mean according to the above results because of its robustness to the outliers. Spikes can affect local statistics estimation, especially when the buffer size is limited. Though several samples are skipped after one detection to reduce such an impact, some residual energy of spikes and undetected spikes can still be contaminated, thus disturbing the threshold calculation. The outlier robustness of the median then plays a critical role in relaxing the impact of the spikes.

D. Proposed algorithm performance and adaptiveness analysis

We have compared the proposed algorithm with shift-based ED emphasize combined with mean or median with different threshold buffer sizes. The results are shown in Table. II. Compared to the 16-sample Mean (M16) or 25-sample Median (MD25). The proposed algorithm achieves much higher detection accuracy across four noise levels. The detection performance is only minorly degraded (1.3%) if the noise level increases from 0.05 to 0.15. It is 5.3% for both M16 and MD25. At 0.2 noise level, the proposed algorithm can still

TABLE II
THE SPIKE DETECTION ACCURACY OF THE PROPOSED AND BASELINE ALGORITHMS AT DIFFERENT NOISE LEVELS

Noise levels	M16	M90	MD25	MD50	ADF+FR
0.05	0.966	0.95	0.983	0.987	0.98
0.1	0.957	0.94	0.968	0.983	0.974
0.15	0.913	0.90	0.93	0.976	0.967
0.2	0.851	0.90	0.835	0.907	0.919
Avg	0.922	0.923	0.929	0.963	0.96

maintain over 90% detection accuracy while it becomes 85% and even lower for the other two.

The median achieves compatible performance if the threshold buffer size increases to 50. However, the improvement in mean-based spike detection is limited.

To investigate how the threshold buffer size can affect different algorithms and their adaptiveness, we simulated the scenario in practice that calibration recordings are statistically different from the recordings onward. The recordings of noise levels 0.05 - 0.15 are used to find the best parameter settings, and the recordings of noise level 0.2 are then tested on these settings. These test results for different buffer size is then plotted as the solid lines in Fig. 6.(a). The dashed lines are the best detection accuracy algorithms can achieve if the settings are appropriately set.

For median-based spike detection, the detection accuracy can be improved effectively with the increased buffer size, while it becomes less effective for the mean for a buffer size larger than 30. This is also because of the outlier robustness of

the median operation. A large mean buffer can help estimate more accurate mean values but also increases the chance of including undetected small spikes. There is no threshold buffer used in the proposed algorithm therefore unaffected.

If the parameters are appropriately set, median-based spike detection can achieve the highest spike detection accuracy when the buffer size is large enough, as the yellow dashed line shows. However, as the solid yellow line shows, when the noise level is different between calibration and experimental recordings, median-based threshold detection performance can be degraded for at least 5% according to our simulation. Firing-rate based algorithm is less affected, and the mean-based one comes in second place. The two green stars are the intersections when mean or median-based detection achieves a similar performance as the firing-rate-based one.

Fig. 6.(b) shows how their detection accuracy varies with the multipliers or the target detection rate at different noise levels. It explains why firing-rate-based spike detection can achieve the highest adaptiveness. The response of the accuracy to the target rate is similar across different noise levels for the firing-rate-based algorithm. However, the median varies significantly while the mean is milder. Therefore, even though the median can achieve the highest detection performance when threshold buffers are enough, it is not suitable in practice.

Aside from the high level of detection accuracy, the ability to operate without calibration is a crucial feature for a spike detection algorithm in Brain-Machine Interface (BMI) systems with thousands of channels, where frequent calibration of all channels is not feasible. The firing-rate-based spike detection algorithm proposed in this study is highly adaptive to varying noise levels, making it a suitable choice for this requirement.

E. Estimation error

In order to reduce hardware complexity, we used recursive median and shift-based multipliers as approximations for the actual median values and multipliers. While these approximations are not part of the proposed spike detection algorithm, it is still useful to quantify the impact on performance.

The results, shown in Fig. 6, indicate that the approximation of the median introduces less than a 5% degradation, and the trends are similar across all three emphasizeers. However, when using estimated multiplication, the degradation is more severe for the NEO approach, as two successive multiplications amplify the estimation error. The ED and ASO methods only involve a single multiplication and show only minor degradation.

The proposed spike detection algorithm does not require any multiplication and thus does not require any approximation. As a result, there is no compromise in achieving a multiplication-free fixed-point spike detection solution.

F. Long-term detection stability on Utah array recordings

Long-term stability is another important aspect of implantable BMI applications. Fig. 7 shows the real detection rate of three different spike detection outcomes from this work, previous work [24] and adaptive STD threshold.

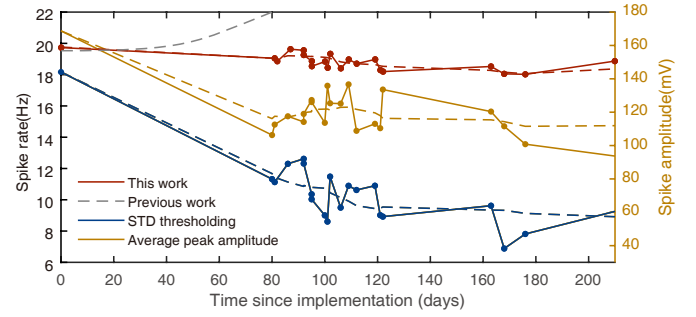


Fig. 7. Spike rates of the Utah array recording detected from this work detected at 20 Hz (Red), previous (Grey), STD thresholding (Blue), and the average spike peak amplitude (Yellow) over 200 days of recording started from 27 July 2016. The grey curve quickly disappears, indicating that the previous method is not long-term stable.

It can be clearly observed from Fig. 7 that the detection rate of the STD threshold (Blue) is reduced from nearly 18 kHz to 10 kHz, while the threshold of [24] can reduce to an unreasonable low level detecting too many spikes (Grey) if not reset constantly. However, the detection rate of this work (Red, target detection rate is set at 20 Hz) is stable over 200 days of recordings.

The growth of scar tissue can push the neurons further away from the implants. The neural activities are thus weakened over time, leading to less significant spikes [40]. Statistical-based spike detection can therefore detect fewer spikes over time.

In our previous work [24], we observed inadequate performance in long-term detection stability. To be robust to the inaccurate initial settings, that work sets the threshold according to both the spike peak mean values and target detection rate. However, due to the sub-optimal spike peak mean value estimation method in that work, false detection of noise peaks can reduce the estimated peak mean value below the real spike peak level. After a long time of detection, the spike peak means can potentially become too low, which guides the threshold to be even lower, creating even more false detections. Such positive feedback can eventually cause the algorithm to fail unless the system is reset. This impact is especially prominent when the signal-to-noise ratio is low.

The firing-rate-based spike detection in this work however eliminates the dependence on spikes. Although the target detection rate should be properly set to achieve optimal spike detection performance, the algorithm can now provide long-term stable detection outcomes as shown in Fig. 7. The threshold can be automatically adjusted to a lower level to detect the activities from neurons that have been pushed further away by the scar tissue maintaining the preset target detection rate. It is reasonable to assume that such stable detection results are expected to provide better long-term decoding performance compared to conventional statistical-based spike detection.

G. Decoding performance on Utah array recordings

The vast majority of spike detection algorithms are validated using synthetic data – as the ground truth is known *a priori*. Rarely are they validated using real recordings, as manual

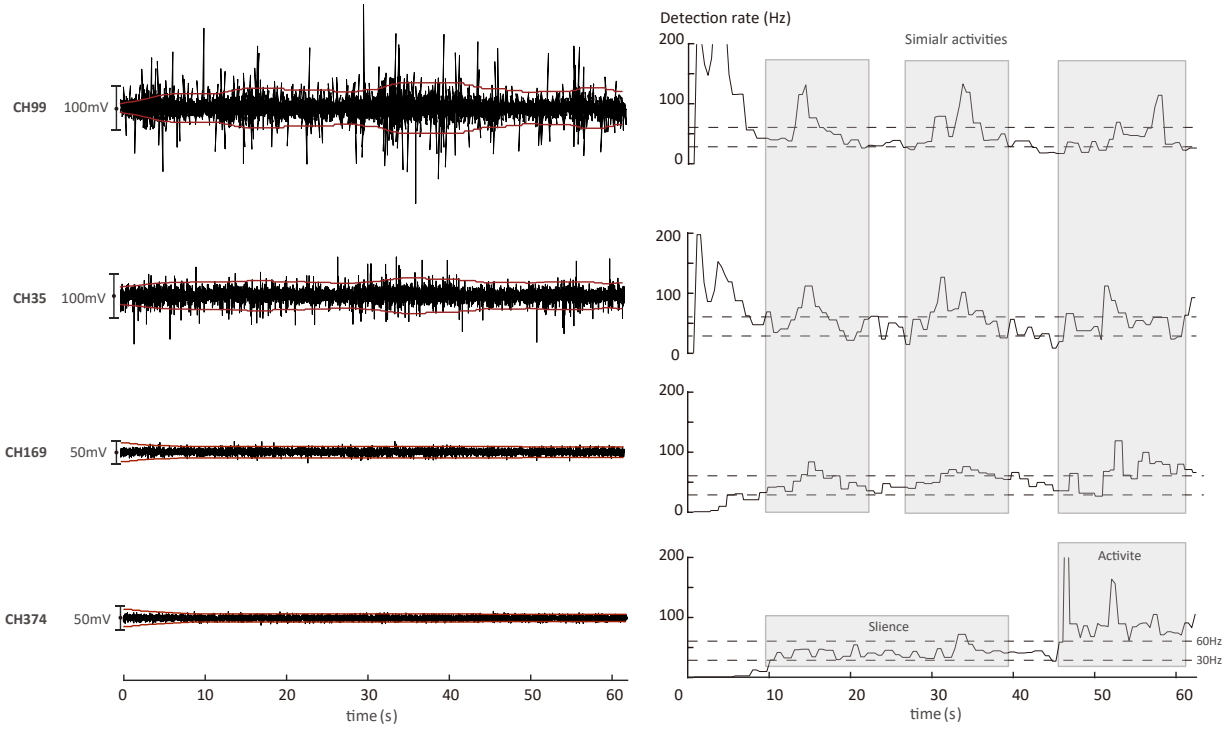


Fig. 8. The Neuropixels recordings and adaptive threshold of four different channels with the corresponding instant detection rate. For better visualization, the recording is plotted at a lower sampling rate of 800 samples/s without taking absolute values, but the algorithm still operates at 7 kHz with absolute values being taken. Some spike peaks may appear missing in the plot due to the lower sampling rate, but they are accurately detected by the algorithm.

TABLE III

DECODING PERFORMANCE WHEN USING THE RESULT FROM STD-BASED SPIKE DETECTION AND THE PROPOSED SPIKE DETECTION ALGORITHM

	STD w calibration	FR w/o calibration	FR w calibration
WCF	0.636	0.663	0.668
LSTM	0.738	0.753	0.759

labeling can be time-consuming. Some studies use the recording of paired electrodes to obtain the neural firing ground truth of one signal neuron. However, this only allows us to evaluate the algorithms' sensitivity, not the false detection rate or accuracy. Other studies have attempted to use new metrics to evaluate spike detection performance in practice [41] or design algorithms to automatically label the spikes [42], but these methods have not been widely used.

We used CC to evaluate the corresponding neural decoding performance and implicitly evaluate the spike detection performance. The results are given in Table III.

We can observe that for both conventional filter-based decoding and machine-learning-based decoding, with a fixed parameter setting across all channels, the proposed spike detection algorithm achieves 1% to 3% higher decoding accuracy compared to the results from calibrated STD-based spike detection. The corresponding mean or median-based result is expected to be even lower as their detection performance is supposed to be worse than the STD-based one. We also notice that channel-wise and day-to-day calibration only increased decoding accuracy by less than 1%, which means that after we find suitable parameters at the start, there is no need for further

parameter calibration for the proposed algorithm in practice.

H. Validation on Neuropixels recordings

We have shown the effectiveness of the firing-rate-based spike detection algorithm on Utah array recordings. The synthetic dataset is also created from Utah array templates. To investigate the adaptability of our algorithm to other types of recordings, we also tested it on Neuropixels recordings, which use a high-density electrode array, and were provided in [36]. Moreover, this high-density recording can provide a diverse collection of the different signal SNRs, allowing us to evaluate the algorithm's robustness in varying brain environments. However, since there is no ground truth available to evaluate the spike detection performance quantitatively, we visually inspected the spike detection results.

The detection outcomes are shown in Fig. 8, where four different channels with varying SNRs are plotted at the same scale. The original recording is down-sampled to 800 Hz before plotting for better visualization, but the detection algorithm still operates at 7 kHz with a target detection interval of [30 Hz, 60 Hz].

Even though the neural activities are significantly attenuated across CH99, CH35, and CH169, similar neural activities are detected from these channels according to the detection rate plot on the right. It is especially notable for CH169 that the spikes are nearly invisible from the noise. It has been shown in [9], [43] that detecting noise-level spikes is beneficial for neural decoding. Such an observation provides strong support for the robustness of the proposed algorithm to be used without calibration.

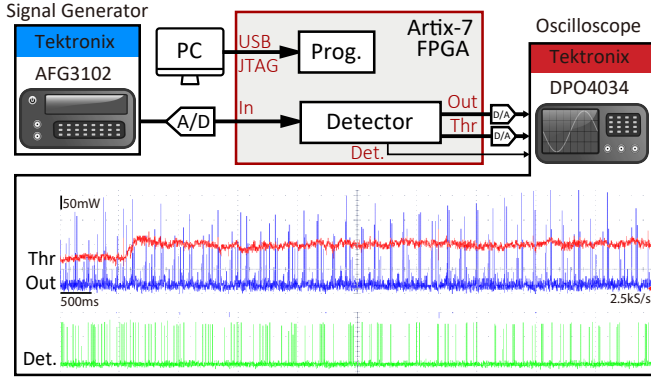


Fig. 9. The experimental setup used to validate the spike detector FPGA implementation, showing the respective instruments that were used.

We can also observe that in CH374, the detection rate is relatively stable at the start. We assume the reason is that the neural activity could be attenuated to be invisible from the noise making this channel to be ‘silence’ for a while. Once the activities happen in a closer region, this channel is activated.

Another observation is that even though a target detection interval is set, the instant detection rate can exceed or fall below this range, rather than being confined within it. This is because the threshold is updated only gradually after the target interval is not met. Burst activities or inactive states can still be detected, even if the rate is expected to be outside that interval.

Although we cannot provide a quantitative analysis of the spike detection accuracy in Neuropixels recording, the visualization and qualitative analysis above can demonstrate the potential of the proposed algorithm to be applied to new recordings at varying noise levels without calibration.

V. HARDWARE IMPLEMENTATION PERFORMANCE

The baseline and proposed algorithms have first been implemented on a Digilent Artix-7 FPGA development board featuring the XC7A35TICSG324-1L FPGA core built on 28 nm technology using Vivado 2022.2. A Digilent PMOD DA2 dual channel 12-bit DAC and Digilent PMOD AD1 dual channel 12-bit ADC are used for I/O conversion. The RTL design is also implemented using TSMC 0.18 μm and 65 nm technology for the ASIC designs. The experimental setup is shown in Fig. 9.

A. Resource occupation and power on FPGA

The FPGA implementation provides us a rapid workflow to compare hardware complexity, identify bottlenecks, and optimize hardware implementation. Resource utilization provides a relative measure of the area and power consumption of the ultimate ASIC design. We synthesized the algorithms into different modules. The resource utilization is shown in Table. IV. We also report the resources of the most resource-saving baseline implementation, ED+Mean spike detection, compared to the proposed firing-rate-based spike detection algorithm.

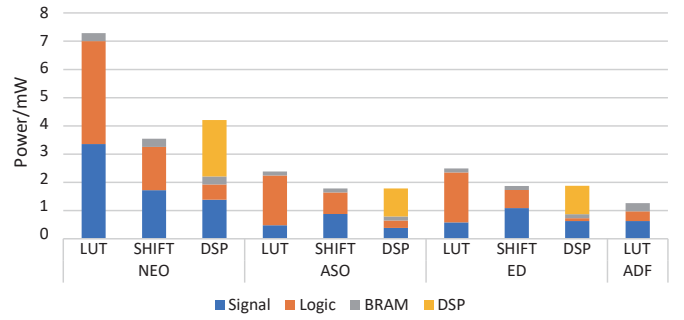


Fig. 10. Power breakdown for different implementations of NEO, ASO, ED and ADF emphasizeers simulating at 100 kHz.

1) *Emphasizers*: The absolute difference filter is obviously the most hardware-efficient emphasizeer. 2-6 times fewer LUTs can be saved compared to other LUT multiplication implementations, without considering the usage of an LFP filter for baseline emphasizeers. Even though we implement NEO, ASO or ED using shift-based multipliers, they still consume more LUTs than ADF.

We also investigated how three different implementations of the multipliers can affect the hardware-efficiency. Though none of these are used in the proposed spike detection algorithm, these results can provide useful information to design new emphasizeers. The LUT-based multiplier provides a fair reference for the corresponding ASIC design, while the shift-based multiplication has shown promising performance in several works [17]. Compared to the LUT-based multipliers, half of the LUT can be saved with the cost of a maximum of 5% accuracy excluding NEO, as shown previously in Fig. 6(c).

The power breakdown is shown in Fig. 10 according to the power estimator with the implementation simulated at 100 kHz. ADF is estimated to consume the least amount of power which is 7 times less compared to the LUT implementation of NEO. It can also be observed that shift implementation is supposed to reduce 25% of the power consumption for ASO and ED, while it becomes 50% for NEO. However, the DSP implementation of NEO consumes more power than the shift implementation becomes two DSPs are used. Their power consumption becomes similar when only one DSP is used for ASO and ED.

2) *Thresholding*: The 16-sample mean and 25-sample median are two compact baseline algorithms, while the 64-sample mean and 50-sample median are two settings achieving competitive detection performance as the firing-rate-based algorithm. The RAM bandwidth increases linearly with an increase in buffer size. It eventually requires more than 30-40 times more RAM for median or mean to achieve similar performance as the firing-rate-based spike detection. The proposed firing-rate-based spike detection utilizes half the LUTs and significantly less RAM compared to the mean or any median implementation because no buffer is needed for calculating the statistics.

The estimated median consumes much more resources compared to others. However, the conventional median operation for 25 numbers can consume over 7500 LUTs. More than 5

TABLE IV
FPGA IMPLEMENTATION RESOURCE UTILISATION

		LUT	Registers	RAM Width (Bits)	DSPs
LFP filter	Butterworth filter	196	0	20	0
emphasizers	NEO LUT Mul	272	0	20	0
	NEO shift	147	0	20	0
	NEO DSP	50	0	20	2
	ASO LUT Mul	140	0	10	0
	ASO shift	75	0	10	0
	ASO DSP	29	0	10	1
	ED LUT Mul	121	0	10	0
	ED shift	48	0	10	0
	ED DSP	10	0	10	1
	ADF	30	0	20	0
	Mean (16/64)	95	0	380/1300	0
	Median unroll (25/50)	1495/2579	0	500/1000	0
Thresholding	median roll (25/50)	468	184	500/1000	0
	median real 25	>7500	-	-	-
	FR	53	0	33	0
Full system	ADF+FR	78	16	53	0
	Previous ADF+FR	193	106	90	0
	ED+Mean16	140	24	390	0

TABLE V
COMPARISON TO STATE-OF-THE-ART ASIC ADAPTIVE SPIKE DETECTORS

	This work (0.18 μm)	This work (65 nm)	BioCAS2022 [29]	JNE2022 [27]	TBCAS2020 [16]	AEU2018 [18]	TBCAS2017 [44]
Technology (nm)	180	65	65	180	180	180	130
Implementation	Digital	Digital	Digital	Digital	Analog	Analog	Digital
Supply voltage	1.8	1.2	1.1	1.8	0.5	0.8	1.2
Clock frequency (MHz)	0.896	0.896	4	0.8	0.54	-	0.16
Preprocessing	ADF	ADF	Dual NEO	NEO	MAE	ED	NEO
Thresholding	FR	FR	STD	RMS	Mean	RMS	RMS
Channel	128	128	256	128	-	-	32
Resolution (Bits)	10	10	7	-	-	-	10
Sample frequency (kHz)	7	7	16	24	30	16	20
Power per channel ($\mu\text{W}/\text{Ch}$)	0.28	0.038 ^b	0.07	4.9	0.116	5.1	0.52 ^d
Area per channel (mm^2/Ch)	6.76×10^{-3}	7.51×10^{-4}	9.69×10^{-4} ^c	0.02	0.27	0.018	3.82×10^{-3} ^c
Accuracy	0.96 ^a	0.96 ^a	0.97	0.92	<0.97 ^e	0.95	<0.95 ^e

^a The accuracy is 0.97 @ 12 kHz sampling rate.

^b At best FoM (32 channels), this design only takes $0.01 \mu\text{W}/\text{ch}$ and $1.01 \times 10^{-3} \text{mm}^2/\text{ch}$.

^c The area is obtained from multiplexing eight 32-channel modules.

^d Only NEO processor and threshold estimator modules in the spike sorting design are included.

^e Maximum accuracy on a different dataset.

times reduction has been made through the median estimation. The recursion implementation of median estimation makes the median operation achievable in hardware, especially in the rolled version, where it can be implemented using less than 1000 logic cells.

The power of the thresholding algorithm is highly data-dependent and therefore is not estimated. However, it is no doubt that the firing-rate-based thresholding can consume less power than the mean or median because of the very low RAM requirements.

Compared to our previous work, our new design has achieved significant resource savings. Specifically, one-third of the LUTs+registers and half of the RAM bandwidth have been saved. Furthermore, Our design consumes $0.21 \mu\text{W}$ power

in Lattice ice40lp1k platform, which features a low-power FPGA with 40 nm technology. In comparison, the power consumption was $0.28 \mu\text{W}$ in our previous work. We should note that the Digilent Artix-7 FPGA platform can consume considerable static power, which can lead to inaccurate power measurements. Therefore, we report the power consumption on a lower-end FPGA platform to provide more representative values for power.

According to the spike detection accuracy in Table. II and the resource usage in Table. IV, we can see that combining the shift-based ED with 16-point mean thresholding can be the best trade-off of statistical-based spike detection. LFP filter is not used as the difference operation in ED already removes LFP and there is no performance degradation. Therefore,

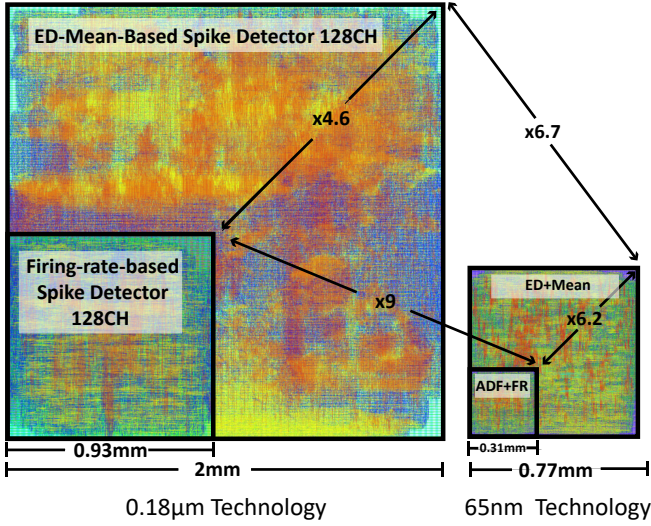


Fig. 11. The layouts of 128 channel proposed firing-rate-based spike detector and ED-Mean-based spike detector in 0.18 μm and 65 nm technology. The ratio of the area occupation is annotated.

we selected ED+Mean16 as a generic spike detection model following the statistical approach. It achieves adequate spike detection performance and adaptiveness while being the most hardware efficient among other statistical-based algorithm combinations. The ADF+FR spike detection in this work consumes only half LUTs and one-seventh RAM bandwidth compared to the ED+Mean16 spike detector. Such improvement in hardware efficiency is expected to reduce the ASIC area and power consumption significantly.

B. Area and power estimation on ASIC

The register transfer level (RTL) implementation of the Firing-rate-based spike detection is mapped to two ASIC designs using TSMC 0.18 μm BCD Gen II and 65 nm LP technology. The synthesis and place & route were performed by Cadence Genus and Innovus by using standard digital cells. The entire implementation is composed of sequential and combinational logic cells. Based on the synthesized reports, sequential logic instances (mainly the memory) occupy four times of silicon area than combinational logic ones. At 0.18 μm technology, the 128-channel FR spike detection occupies $0.93 \times 0.93 \text{ mm}^2$ and consumes $35.7 \mu\text{W}$ at 1.8 V supply voltage. When it is upgraded to 65 nm technology, it only occupies $0.31 \times 0.31 \text{ mm}^2$ and consumes $4.86 \mu\text{W}$ at 1.2 V supply voltage, resulting in 7-fold reduction in power consumption and 9-fold reduction in area. These two designs can provide a good reference to study how the technology node can impact the power and area, and allow us to compare our work with state-of-the-art implementations using different technology.

As shown in Table. V, our design has achieved the lowest power consumption and area occupation while still maintaining spike detection accuracy similar to the highest performing designs that have been included in the table. Compared the 65 nm implementation with the work in [29], the power

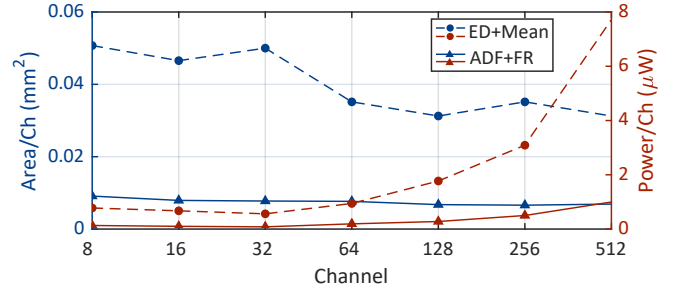


Fig. 12. The area occupation and power consumption scaling with channel count for two different spike detector implementations in 0.18 μm technology.

consumption is nearly halved and the area occupied is reduced by about 1/4.

The power of our 0.18 μm design only consumes 1/20 of the power reported in [27] and occupies 2/3 less area. Although analog designs are expected to have better hardware efficiency than digital designs, our digital design reduced the area occupation by nearly two orders of magnitude compared to the analog design in [16], and more than one order of magnitude of the power compared to [18]. Even the 0.13 μm design in [44] requires twice as much power and area as our 0.18 μm design.

Another interesting observation in Table. V can be found by comparing the work in [16] and [18]. They both use 0.18 μm analog design, while the former one uses mean thresholding and the latter one uses RMS thresholding. The mean thresholding leads to significantly reduced power consumption but requires more area, as sample buffers are needed to calculate the mean values, increasing the area requirement. However, the latter one estimate RMS values without using data buffers, but it typically requires multiple clock cycles (e.g. 200 [29]) and complex logic, leading to less area occupation but significantly increased power consumption. However, firing-rate-based spike detection requires no buffer and can be calculated with only one clock cycle, making it the best choice for spike detection when jointly considering the power, area, and accuracy.

While we implemented our design in two different technology sizes, it is important to note that the hardware specifications, experimental setups, and power estimation strategies can vary, making it challenging to compare designs directly. Comparing our design to the ED+Mean16 algorithm allows us to control for these variables and compare the hardware efficiency difference resulting solely from the algorithm. We can use them as two generic models for comparing statistical and firing rate approaches for thresholding. Furthermore, we can investigate how these two approaches scale with increased channel counts. As a result, we implemented both designs in 0.18 μm technology with channel counts ranging from 8 to 512 channels. The layouts of 128 channel designs of both algorithms in two technology sizes are given in Fig. 11 with the relative area ratios among them.

The 128-channel ED+Mean16 requires $2 \times 2 \text{ mm}^2$ and consume $35.7 \mu\text{W}$, which is 0.03 mm^2 and $1.77 \mu\text{W}$ per channel.

It already achieves a similar area occupation and much lower power consumption compared to the work in [27] with the same spike detection accuracy (0.92). Compared to the firing-rate-based spike detection in this work, it requires 4.6 times more area and 6.3 times more power while the detection accuracy becomes 4% lower.

Fig.12 shows the scalability of these two implementations with an increased channel count. The area occupation per channel decreased slowly for both spike detectors because combinational circuits are shared across channels. The power consumption of ED+Mean16 scales much faster than ADF+FR, ED+Mean consumes $0.64\ \mu\text{W}$ more power than ADF+FR per channel when the channel number is 8 and it becomes $6.7\ \mu\text{W}$ per channel when the channel count increased to 512. The reason is that ED+Mean16 requires much more memory than ADF+FR and the memory is supposed to be the most power-consuming module [29]. The observation above demonstrates that the firing-rate-based thresholding has better scalability with increased channel count compared to the mean-based thresholding. As the RMS-based spike detection can consume even more power compared to the mean-based one, our design potentially also outperforms the RMS-based spike detection as well.

In this section, we have demonstrated improved hardware efficiency over other works. Such an advantage comes from various aspects. On the one hand, since ADF requires no multiplication, it requires fewer logic cells, and the data bandwidth is maintained at 10 bits. In contrast, emphasizees using multiplications are more complex and require 20-bit output bandwidth. Reduced logic complexity and data width lead to less resource usage and more efficient placing and routing. On the other hand, the firing-rate-based thresholding of our algorithm does not require buffering of samples, reducing RAM usage. Additionally, it only requires one clock cycle for each sample. As a result, power and area can be significantly reduced.

VI. CONCLUSION

This paper reviews, compares and further develops spike detection methods for real-time on-implant operation in BMI applications. We start by comparing common spike detection methods that focus on balancing spike detection performance with hardware efficiency, and then further develop a firing-rate-based spike detection algorithm. We compare implementation and performance metrics including synthetic neural spike detection accuracy, real recording decoding accuracy, algorithm adaptiveness, long-term detection stability, hardware power consumption, and area utilization.

The firing-rate-based spike detection algorithm we propose achieves over 96% detection accuracy and the best adaptiveness across noise levels. It also achieves higher decoding accuracy without any calibration compared to the calibrated conventional spike detection algorithms. Moreover, it can provide stable detection results for long-term recordings over 200 days and well-adapt to different types of recordings. With 65 nm technology, the ASIC design only occupied $7.51 \times 10^{-4} \text{mm}^2$ area and consumes $0.038\ \mu\text{W}$ power per channel making it one of the most hardware-efficient spike detectors in literature.

A high-performance and hardware-efficient spike detection algorithm is crucial for next-generation iBMIs to achieve thousands of channels. Spike detection can reduce data bandwidth by approximately $200\times$, enabling the transmission of $200\times$ more channels with the same transmission power. This is however only possible if spike detection can be achieved without distorting features and consuming significant additional power. The proposed spike detection algorithm strikes a very good balance between adaptiveness, hardware efficiency and detection accuracy.

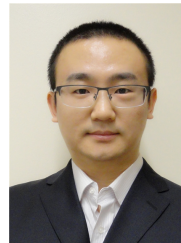
REFERENCES

- [1] J. L. Collinger, B. Wodlinger *et al.*, "High-performance neuroprosthetic control by an individual with tetraplegia," *The Lancet*, vol. 381, no. 9866, pp. 557–564, 2013, [Online].
- [2] F. R. Willett, D. T. Avansino *et al.*, "High-performance brain-to-text communication via handwriting," *Nature*, vol. 593, no. 7858, pp. 249–254, 2021, [Online].
- [3] F. R. Willett, E. Kunz *et al.*, "A high-performance speech neuroprosthesis," *bioRxiv*, pp. 2023–01, 2023, [Online].
- [4] A. B. Rapeaux and T. G. Constantinou, "Implantable brain machine interfaces: first-in-human studies, technology challenges and trends," *Current Opinion in Biotechnology*, vol. 72, pp. 102–111, 2021, [Online].
- [5] Y. Liu, A. Urso *et al.*, "Bidirectional bioelectronic interfaces: System design and circuit implications," *IEEE Solid-State Circuits Magazine*, vol. 12, no. 2, pp. 30–46, 2020, [Online].
- [6] N. Ahmadi, M. L. Cavuto *et al.*, "Towards a distributed, chronically-implantable neural interface," in *2019 9th International IEEE/EMBS Conference on Neural Engineering (NER)*. IEEE, 2019, pp. 719–724, [Online].
- [7] B. Dutta, "Eavesdropping on the brain: With 10,000 electrodes, this neural implant senses more than ever before," *IEEE Spectrum*, vol. 59, no. 6, pp. 30–35, 2022, [Online].
- [8] A. Eftekhari, E. P. Sivylla *et al.*, "Towards a next generation neural interface: Optimizing power, bandwidth and data quality," in *2010 Biomedical Circuits and Systems Conference (BioCAS)*. IEEE, 2010, pp. 122–125, [Online].
- [9] S. Todorova, P. Sadtler *et al.*, "To sort or not to sort: the impact of spike-sorting on neural decoding performance," *Journal of Neural Engineering*, vol. 11, no. 5, p. 056005, 2014, [Online].
- [10] N. Ahmadi, T. G. Constantinou *et al.*, "Robust and accurate decoding of hand kinematics from entire spiking activity using deep learning," *Journal of Neural Engineering*, vol. 18, no. 2, p. 026011, 2021, [Online].
- [11] G. Baranauskas, "What limits the performance of current invasive brain machine interfaces?" *Frontiers in Systems Neuroscience*, vol. 8, p. 68, 2014, [Online].
- [12] S. Kim and J. McNames, "Automatic spike detection based on adaptive template matching for extracellular neural recordings," *Journal of Neuroscience Methods*, vol. 165, no. 2, pp. 165–174, 2007, [Online].
- [13] S. Luan, I. Williams *et al.*, "Compact standalone platform for neural recording with real-time spike sorting and data logging," *Journal of Neural Engineering*, vol. 15, no. 4, p. 046014, 2018, [Online].
- [14] Y. Yang, C. S. Boling *et al.*, "Adaptive threshold neural spike detector using stationary wavelet transform in cmos," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 23, no. 6, pp. 946–955, 2015, [Online].
- [15] A. M. Kambh, M. Raetz *et al.*, "Area-power efficient vlsi implementation of multichannel dwt for data compression in implantable neuroprosthetics," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 1, no. 2, pp. 128–135, 2007, [Online].
- [16] R. Fiorelli, M. Delgado-Restituto *et al.*, "Charge-redistribution based quadratic operators for neural feature extraction," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 14, no. 3, pp. 606–619, 2020, [Online].
- [17] Z. Zhang and T. G. Constantinou, "Adaptive spike detection and hardware optimization towards autonomous, high-channel-count bmis," *Journal of Neuroscience Methods*, vol. 354, pp. 109–103, 2021, [Online].
- [18] S. Dwivedi and A. K. Gogoi, "A novel adaptive real-time detection algorithm for an area-efficient cmos spike detector circuit," *AEU-International Journal of Electronics and Communications*, vol. 88, pp. 87–97, 2018, [Online].

- [19] L. Schaffer, S. Pletl *et al.*, "Spike detection using cross-correlation based method," in *2019 IEEE 23rd International Conference on Intelligent Engineering Systems (INES)*. IEEE, 2019, pp. 000 175–000 178, [Online].
- [20] S. Mukhopadhyay and G. Ray, "A new interpretation of nonlinear energy operator and its efficacy in spike detection," *IEEE Transactions on Biomedical Engineering*, vol. 45, no. 2, pp. 180–187, 1998, [Online].
- [21] H. Semmaoui, J. Drolet *et al.*, "Setting adaptive spike detection threshold for smoothed teo based on robust statistics theory," *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 2, pp. 474–482, 2012, [Online].
- [22] J. H. Choi, H. K. Jung *et al.*, "A new action potential detector using the MTEO and its effects on spike sorting systems at low signal-to-noise ratios," *IEEE Transactions on Biomedical Engineering*, vol. 53, no. 4, pp. 738–746, 2006, [Online].
- [23] Y.-G. Li, Q. Ma *et al.*, "Ultra-low-power high sensitivity spike detectors based on modified nonlinear energy operator," in *2013 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2013, pp. 137–140, [Online].
- [24] Z. Zhang, O. W. Savolainen *et al.*, "Algorithm and hardware considerations for real-time neural signal on-implant processing," *Journal of Neural Engineering*, vol. 19, no. 1, p. 016029, 2022, [Online].
- [25] J. Navajas, D. Y. Barsakcioglu *et al.*, "Minimum requirements for accurate and efficient real-time on-chip spike sorting," *Journal of Neuroscience Methods*, vol. 230, pp. 51–64, 2014, [Online].
- [26] N. Even-Chen, D. G. Muratore *et al.*, "Power-saving design opportunities for wireless intracortical brain–computer interfaces," *Nature Biomedical Engineering*, 2020, [Online].
- [27] D. Valencia, P. P. Mercier *et al.*, "In vivo neural spike detection with adaptive noise estimation," *Journal of Neural Engineering*, vol. 19, no. 4, p. 046018, 2022, [Online].
- [28] A. Oprea, Z. Zhang *et al.*, "Hardware evaluation of spike detection algorithms towards wireless brain machine interfaces," in *2022 IEEE Biomedical Circuits and Systems Conference (BioCAS)*. IEEE, 2022, pp. 60–64, [Online].
- [29] X. Guo, M. Shaeri *et al.*, "An accurate and hardware-efficient dual spike detector for implantable neural interfaces," in *2022 IEEE Biomedical Circuits and Systems Conference (BioCAS)*. IEEE, 2022, pp. 70–74, [Online].
- [30] O. W. Savolainen, Z. Zhang *et al.*, "Hardware-efficient compression of neural multi-unit activity," *IEEE Access*, vol. 10, pp. 117 515–117 529, 2022, [Online].
- [31] O. Savolainen, Z. Zhang *et al.*, "Ultra low power, event-driven data compression of multi-unit activity," *bioRxiv*, pp. 2022–11, 2022, [Online].
- [32] Z. Zhang and T. G. Constandinou, "Selecting an effective amplitude threshold for neural spike detection," in *2022 44th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*. IEEE, 2022, [Online].
- [33] V. Macefield, S. Gandevia *et al.*, "The firing rates of human motoneurons voluntarily activated in the absence of muscle afferent feedback," *The Journal of Physiology*, vol. 471, no. 1, pp. 429–443, 1993, [Online].
- [34] J. E. O'Doherty, M. M. B. Cardoso *et al.*, "Nonhuman primate reaching with multichannel sensorimotor cortex electrophysiology," May 2017, [Online].
- [35] R. Q. Quiroga, Z. Nadasdy *et al.*, "Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering," *Neural Computation*, vol. 16, no. 8, pp. 1661–1687, 2004, [Online].
- [36] N. Steinmetz, M. Carandini *et al.*, "'Single Phase3" and "Dual Phase3" Neuropixels Datasets," 2019, [Online].
- [37] R. D. Flint, Z. A. Wright *et al.*, "Long term, stable brain machine interface performance using local field potentials and multiunit spikes," *Journal of Neural Engineering*, vol. 10, no. 5, p. 056005, aug 2013, [Online].
- [38] R. D. Flint, E. W. Lindberg *et al.*, "Accurate decoding of reaching movements from field potentials in the absence of spikes," *Journal of Neural Engineering*, vol. 9, no. 4, p. 046006, 2012, [Online].
- [39] R. D. Flint, M. R. Scheid *et al.*, "Long-term stability of motor cortical activity: implications for brain machine interfaces and optimal feedback control," *Journal of Neuroscience*, vol. 36, no. 12, pp. 3623–3632, 2016, [Online].
- [40] J. C. Barrese, N. Rao *et al.*, "Failure mode analysis of silicon-based intracortical microelectrode arrays in non-human primates," *Journal of Neural Engineering*, vol. 10, no. 6, p. 066014, 2013, [Online].
- [41] A. H. Barnett, J. F. Magland *et al.*, "Validation of neural spike sorting algorithms without ground-truth information," *Journal of Neuroscience Methods*, vol. 264, pp. 65–77, 2016, [Online].
- [42] Z. Zhang and T. G. Constandinou, "A robust and automated algorithm that uses single-channel spike sorting to label multi-channel neuropixels data," in *2021 10th International IEEE/EMBS Conference on Neural Engineering (NER)*. IEEE, 2021, pp. 783–787, [Online].
- [43] E. R. Oby, S. Perel *et al.*, "Extracellular voltage threshold settings can be tuned for optimal encoding of movement and stimulus parameters," *Journal of Neural Engineering*, vol. 13, no. 3, p. 036009, 2016, [Online].
- [44] Y. Yang, S. Boling *et al.*, "A hardware-efficient scalable spike sorting neural signal processor module for implantable high-channel-count brain machine interfaces," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 11, no. 4, pp. 743–754, 2017, [Online].



Zheng Zhang became a Student Member of IEEE in 2022. He received the B.Eng degree jointly from Beijing University of Technology (BJUT), China, and University College Dublin (UCD), Ireland, in 2018, and the M.Sc degree from Imperial College London in 2019. He is currently pursuing his Ph.D. at Imperial College London. His research interests are in hardware-efficient neural signal processing, brain-machine interfaces and artificial intelligence. His main research focus is on low-complexity neural signal processing and hardware co-design.



Peilong Feng (Member, IEEE) received the B.Eng degree in electrical engineering from the Henan Polytechnic University, China, in 2011, the first M.Sc degree in microelectronic systems design from University of Southampton, UK, in 2012, the second M.Sc degree in analogue and digital integrated circuit design from Imperial College London, UK, in 2015, the Ph.D. degree from the Imperial College London, in 2020. He is currently research associate at the Next Generation Neural Interfaces (NGNI) Laboratory. From 2012 to 2014, he worked as an electronic engineer in Shanghai Research Institute, China Coal Technology and Engineering group. His current research focuses on completely wireless infrastructure for distributed mm-sized neural implants and CMOS-memristor technologies.



Alexandru Oprea received his M.Eng in Electrical and Electronic Engineering from Imperial College of London, in 2022. His dissertation was focused on real-time, low-power spike detection and interpretation methods. He is currently pursuing a career in digital design.



Timothy G. Constandinou (Senior Member, IEEE) received the B.Eng. and Ph.D. degrees in Electronic Engineering from Imperial College London, London, U.K., in 2001 and 2005, respectively. He is currently Professor of Bioelectronics at Imperial College London, Director of the Next Generation Neural Interfaces (NGNI) Lab and Head of the Circuits & Systems Research Group. He is also a Group Leader within the UK Dementia Research Institute, Care Research & Technology Centre. His research interests include biomedical microsystems, implantable medical devices, neural interfaces, brain-machine interfaces, research platforms, and remote sensing using ultra-wideband radar. His lab focuses on creating innovative neurotechnologies to enable communication between the nervous system and electronic devices to study, manage, or treat neurological conditions. He recently co-founded MintNeuro, an Imperial College spinout to translate his research in the field of implantable neurotechnology. Within the IEEE, he regularly serves on Circuits and Systems Society conference committees including Technical Program Co-Chair (BioCAS 2010, 2011, 2018), General Chair (BrainCAS 2016, NeuroCAS 2018), Plenary Co-Chair (ICECS 2020, ICECS 2022), Special Session Co-Chair (ISCAS 2017, BioCAS 2019), Tutorial Co-Chair (ISCAS 2021, BioCAS 2022, BioCAS 2024). Previously he served on the IEEE Circuits and Systems (CAS) Society, Board of Governors (2017–19) and was Associate Editor-in-Chief of IEEE TRANSACTIONS ON BIOMEDICAL CIRCUITS AND SYSTEMS (2020–2021).