Hardware evaluation of spike detection algorithms towards wireless brain machine interfaces

Alexandru Oprea*, Zheng Zhang*†, Timothy G. Constandinou*†‡
*Department of Electrical and Electronic Engineering, Imperial College London, SW7 2BT, UK
†Centre for Bio-Inspired Technology, Institute of Biomedical Engineering, Imperial College London, SW7 2AZ, UK
‡UK Dementia Research Institute, Care Research & Technology Centre at Imperial College London and University of Surrey
Email: (alexandru.oprea18, zheng.zhang18, t.constandinou)@imperial.ac.uk

***Abstract*—The current trend for implantable Brain Machine Interfaces (BMIs) is to increase the channel count, towards next generation devices that improve on information transfer rate. This however increases the raw data bandwidth for wired or wireless systems that ultimately impacts the power budget (and thermal dissipation). On-implant feature extraction and/or compression are therefore becoming essential to reduce the data rate, however the processing power is of concern. One common feature extraction technique for intracortical BMIs is spike detection. In this work, we have empirically compared the performance, resource utilization, and power consumption of three hardware efficient spike emphasizers, Non-linear Energy Operator (NEO), Amplitude Slope Operator (ASO) and Energy of Derivative (ED), and two common statistical thresholding mechanisms (using mean or median). We also propose a novel median approximation to address the issue of the median operator not being hardware-efficient to implement. These have all been implemented and evaluated on reconfigurable hardware (FPGA) to estimate their hardware efficiency in an ultimate ASIC design. Our results suggest that ED with average thresholding provides the most hardware efficient (low power/resource) choice, while using median has the advantage of improved detection accuracy and higher robustness on threshold multiplier settings. This work is significant because it is the first to implement and compare the hardware and algorithm trade-offs that have to be made before translating the algorithms into hardware instances to design wireless implantable BMIs.**

I. Introduction

BMIs have enabled disabled patients to control neuroprosthetics [1] or communicate through thought [2] [3]. In such circumstances, intracortical implants record the electrical activity of the neurons and subsequently analyze the spikes (observed action potentials) through a behavioral decoding algorithm in order to control external assistive technology such as a prosthetic limb or a text-to-speech prosthesis.

With the recent trend of designing wireless distributed BMIs with increased channel counts [4], the growing amount of data requires unfeasible bandwidth and wireless transmission power for the implants. Real-time on-implant feature extraction and compression are therefore essential to distill the informative features, reduce bandwidth and consequently reduce the transmitting power. The firing rate is one of the most commonly used features [5]–[8] representing essential information correlated to subject behaviors with significantly reduced bandwidth. Using firing rate reduces the bit rate linearly with the increase of bin count period [4]. However, on-implant feature extraction and compression bring extra hardware complexity and face great challenges in power consumption. It has been demonstrated in [9] that in practice, the implant power consumption should be less than 0.8 mW/mm^2.

To obtain the firing rate of neurons, the neuron action potential spikes have to be detected and counted. The detection of spikes represents the process of discerning between the background noise of the signal and the action potential. Multiple spike detection techniques have been developed based on template matching [10] [11], wavelet transform [12], and statistics [13]–[15]. Due to its simplicity, the latter class of methods has generated substantial interest in fully-implantable BMIs. The expectation is generally the most used statistics due to its low implementation cost in the form of a moving average. The mean is usually paired with the NEO or its variants (i.e., ASO, ED, Smoothed-NEO, multi-NEO) [16]–[19] which emphasizes the spikes before averaging. The median is regarded in the literature as a measure of the standard deviation of the noise through the sensible assumption of normal distribution [20]. However, the successive comparison and large data buffer involved make it less feasible in hardware use. Similarly, due to high hardware cost, other thresholding mechanisms using statistics such as standard deviation, RMS value, and cross-correlation [15] have also not been widely used in hardware implementations.

There are significant studies on implementing and comparing various spike detection algorithms, but they rarely experimentally compare their hardware trade-offs. In this work, we have implemented three different spike emphasizers (NEO, ASO, ED), two thresholding algorithms (mean and median), and a compression module (spike binner). We have also proposed a novel implementation of median thresholding to fit the median operation in an allowable hardware budget. A comprehensive comparison is drawn in terms of spike detection performance of algorithm combinations and Field Programmable Gate Arrays (FPGA) power consumption and resource usage.

II. System Architecture

Fig. 1 depicts the general signal path of a fully-implantable BMI with on-implant signal processing capability. This work will focus on the last three phases (i.e., Emphasizers, Thresholding, and Compression). Our implementation assumes an

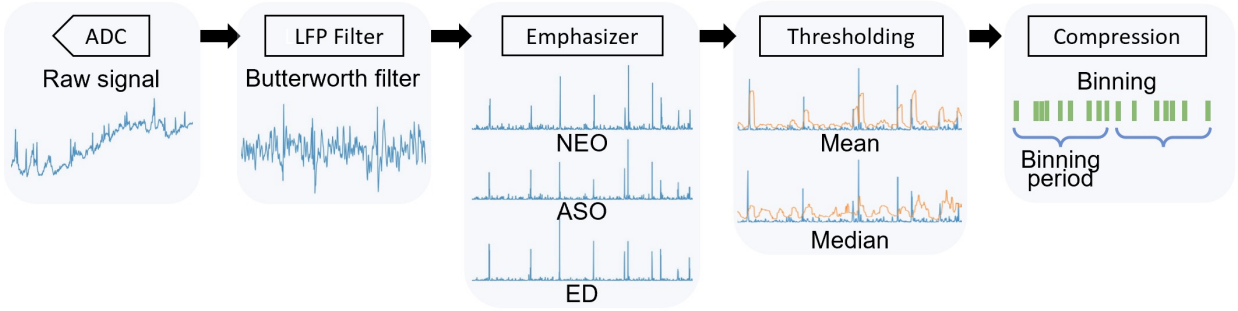


Fig. 1. The architecture of the firing rate-based BMI. The waveform under each module represents the output wave of the respective stage

analog fronted of 7 kHz and 10-bit resolution per channel, as they have been proved to be the minimum requirements to contain the essential information of the spikes [21]. As this literature also suggested, a two-pole Butterworth filter is used to remove the LPF leaked to the passband.

A. Emphasizer

Before finding the threshold, a pre-emphasis step is applied to the LPF removed signal. Traditionally, NEO is used due to its characteristics of accentuating high energy, high-frequency signals (i.e., spikes) to suppress the noise.

$$\psi_{NEO}[n] = x[n]^2 - x[n-1]x[n+1] \tag{1}$$

Eq. 1 depicts the formula of the digital NEO operator ψ_{NEO}. There are multiple variants of NEO; however, some involve digital filter smoothing [18] or more samples to be buffered [19], adding extra complexity. Therefore, the only variants of NEO considered in this work are ASO [16] and ED [17], due to their low hardware cost, requiring one less multiplication and one less sample than the NEO. ASO and ED are shown in formula 2 and 3 respectively.

$$\psi_{ASO}[n] = x[n](x[n] - x[n-1]) \tag{2}$$

$$\psi_{ED}[n] = (x[n] - x[n-1])^2 \tag{3}$$

After the computation of the emphasized signal, the absolute values are then taken to preserve the amplitude and gradients on both sides.

B. Thresholding Algorithm

Subsequent to pre-emphasis, the signal is processed with the thresholding module. Traditionally, after the NEO pre-emphasis, the threshold would be set using a moving average as in the formula below.

$$Thr[n] = p * \frac{\sum_{i=0}^{N-1} \psi[n-i]}{N} \tag{4}$$

Where N is the current sample, ψ is the output of the emphasizer, p is a constant, and N is the number of samples included in the average. This solution is employed due to its very low computational cost as on every cycle, an addition, a subtraction, and a shift is performed (Given that p and N are carefully chosen). This method was proved to yield good performance [13] considering its simplicity. N here is set to be 16, yielding good performance, low resource occupation, and ease of bit shift.

The median operation has a high implementation complexity of over O(n^2), which is unsuitable for real-time processing. In order to mitigate these shortcomings, we propose to successively find the median of medians within a set of samples to estimate the median values. The approach divides the sequence into sets of 5 and then finds the median of each set. The process then repeats to the medians found until only one number remains. This process does not guarantee to find the median but an estimation. However, our results suggest that the accuracy drop is under 3% in low to moderate noise scenarios, reaching approximately 5% in high noise ones (Fig. 3).

This approach represents a hardware-efficient implementation as the implementation complexity has decreased from over O(n^2) to O(nlog(n)). Furthermore, the instances of the modules can be set up through simple interconnections, allowing for scalability. The set size of 5 was chosen as the minimum interval that can contain a spike by itself. A higher interval would have generated a behavior closer to a median. However, it would have come at the cost of multiple successive comparisons. The implementation of the median of 5 number module achieves the optimal resource consumption by using only 6 comparators. Notice that the median mentioned later is this median approximation using 25 samples unless specified.

C. Compression

The compression module is constructed as a series of counters (one for each channel) that count the number of threshold crossing in every 800 samples (about 114 ms). To allow more robustness in the binning process and avoid multiple detections of one spike, the binner will not increment the value after one detection, until one refractory period has passed, usually 1 ms. At the end of the set binning period, counts are output, and the counters reset.

III. Results

A. Evaluation metrics and hardware platform setting

To test the spike detection performance of the algorithms in combination with the different emphasizers, the algorithms have been simulated offline on a widely used dataset from [20]. It consists of real spike shapes positioned in time with Poisson distribution. The superimposed realistic noise

consists of randomly selected spikes with a relative standard deviation of 0.05, 0.1, 0.15 and 0.2 respectively. Accuracy, a generally used metric to compare spike detection algorithms, is presented below.

$$Accuracy = \frac{TP}{TP + FP + FN} \tag{5}$$

Where True Positive (TP) represents truly detected spikes, False Positive (FP) represents wrongly detected spikes, and False Negative (FN) represents the undetected spikes.

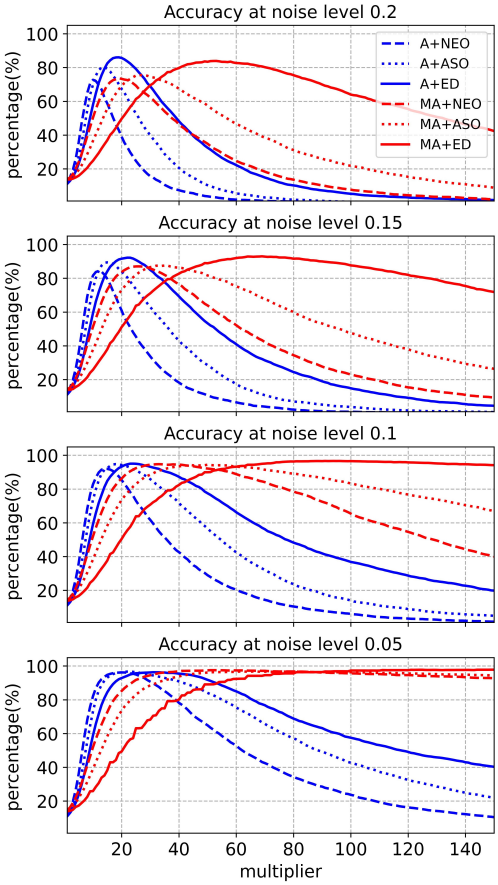
Different methods have been implemented on the XC7A35T AMD FPGA built on the 28nm technology. The resource utilization was reported by the synthesizer and the dynamic power was measure by subtracting between the FPGA core power with and without the methods implemented. In order to avoid simplification in the synthesis stage, the out-of-context mode is used. These steps assure that the reported power is directly linked to the complexity of the algorithm and not to synthesis factors. The FPGA hardware results would not translate directly to the ultimate ASIC design, but it is still a reliable estimation for the ease of algorithm complexity comparison and implementation selection.

B. Algorithm performance

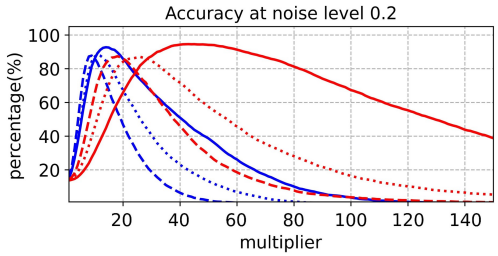
From Fig.2 (a), out of the three emphasizers, ED (Solid lines) yields the highest accuracy across all noise levels and with both methods. ED also exhibits the highest adaptability through robustness against multiplier choice (flatter curves). ASO (Dotted lines) shows similar characteristics but reaches lower accuracy at higher noise levels. The performance of NEO (Dash lines) deteriorates rapidly as the level of noise increases. This finding suggests that the gradient is more discriminative than amplitude in spike emphasizing, especially when noise increases, ED outstands.

As for median approximation (MA, Red lines) and average (A, Blue lines), their performance is similar w.r.t the highest accuracy they can achieve in different noise levels as the figure shows. However, it is a common drawback of the statistical-based thresholding that multiplier choosing can affect the detection performance [22] and therefore introduces the effort of manually tuning the threshold in practice. The median, as the figure shows, is more robust to the choice of the multiplier as it generates a flatter response close to its maximum point. This behavior is most noticeable at 0.05 noise, where the algorithm can always yield high accuracy as long as the multiplier is large enough. On the other hand, when using the average, the accuracy decays very fast as the multiplier gets further from the peak values. This susceptibility is mitigated by using ED, while it results in the most narrow interval by using NEO. Furthermore, The interval for choosing a multiplier narrows as the noise increases for the median method, however, its effect is less noticeable across all emphasizer choices.

Based on the findings above,using ED alongside median approximation can provide the highest spike detection performance while it is the least affected by the choice of the multiplier.



(a) Average (A) with 16 samples vs Median Approximation (MA) with 25 samples



(b) Average (A) with 90 samples vs Median Approximation (MA) with 50 samples with different emphasizers

Fig. 2. The performance difference of Average (A) vs. Median approximation (MA)

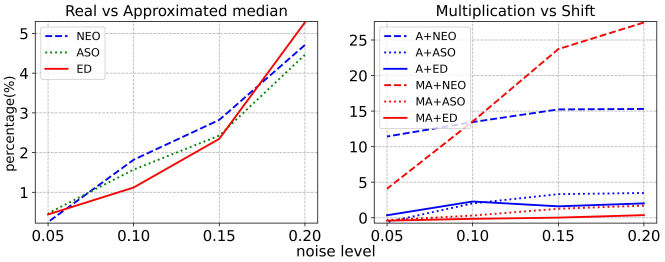


Fig. 3. Accuracy degradation of approximations median (left) and multiplication (right)

TABLE I
THE POWER AND RESOURCE UTILIZATION OF DIFFERENT MODULES

Module name	LUTs	Registers	Block RAM	DSPs	Power(μW)	Power per channel (μW)
Butterworth filter	196	0	1	0	71.8	0.75
NEO multiplier	235	0	1	0	58.6	0.61
NEO bitwise shift	120	0	1	0	53.0	0.55
NEO DSP	30	0	1	2	46.6	0.49
ASO DSP	40	0	0.5	1	40.0	0.42
ED DSP	13	0	0.5	1	36.4	0.38
Average	41	0	5	0	461.8	4.81
Median unrolled	1495	0	7	0	634.6	6.61
Median rolled	465	184	7	0	1107.4	11.54
Counter	85	32	0.5	0.5	25.2	0.26

C. The effect of the number of samples

We also assessed how the number of samples used to calculate the median and average affects the detection performance in high noise level cases. Detecting spikes at high noise levels can be challenging as more numbers need to be buffered for mean/median calculation, but the number buffering can be resource-consuming.

The average starts outperforming the median method as the noise increases with 16 samples for the mean and 25 samples for the median. However, suppose we allow more samples to use. In that case, when the number of samples is increased to 90 samples for the average and 50 for the median, the detection accuracy can be improved by 5%, and the median outperforms the average at all noise levels; Fig. 2(b) shows this behavior. Therefore, if the available memory is increased, the median yields both higher accuracy and better robustness.

D. Resource consumption

One of the most costly hardware resources in the presented emphasizers is the multiplier. They was implemented in three methods: Look-Up Tables (LUTs), Digital Signal Processing units (DSP), and an approximation by left shifting according to the index of the most significant bit. The latter has been previously employed in spike detection yielding promising results [16]. Table. I shows bit shift approximation uses half the resources of the LUT-based multiplier representing a reliable comparison to their ASIC implementation. However, the shift approximation is unsuitable when using the NEO emphasizer as the accuracy degradation overpasses 10% shown in Fig.3. The other two emphasizers exhibit a 4% degradation at worst. The median with ED, in particular, exhibits almost no accuracy degradation. Using DSP yields the lowest power consumption as it is customized for such operation, but DSP would occupy the extra area in ASIC design. Only the NEO emphasizer was used to compare the multiplier methods because it would indicate a larger power difference. The rest of the emphasizers are implemented using DSPs.

The ED uses the least resources and power compared to the other methods as expected. ASO and ED use half the DSP units and the memory used by NEO. ASO uses, unexpectedly, more LUTs than NEO, because the subtraction is fused with multiplication in a single multiply and accumulate (MAC) operation in the DSP core.

The average is implemented through a moving average; thus, it uses few resources outside of the memory. The ram blocks hold both previous samples and the previous value of the average. For our median algorithm, we tested two implementations: one consists of six median modules that divide the 25 samples into groups of 5, and then the median of medians is taken (unrolled version), and the other one has one module fed with 5 groups of five samples and, subsequently, with the medians (rolled version). The first solution aims to reduce the dynamic power consumption at the cost of resource usage, while the other targets the opposite effect. The rolled version uses approximately a third of the logic units used by the unrolled version. However, this improvement comes at the cost of registers that feed the median module and a substantially increased power consumption. The steep increase in state transitions causes the power to almost double in the case of the rolled version of the median.

The counter module uses memory to store the number of arrival spikes in each channel and the bin period counts. The output remains unchanged until the bin period is reached. Due to the low output frequency, it has a lower power consumption than other modules using similar resources (e.g., emphasizers).

IV. CONCLUSION

This paper discusses the trade-off between hardware complexity and detection performance of the median approximation and moving average preceded by three hardware-efficient emphasizers (NEO, ASO, ED) for spike detection. Multiple considerations were taken into account, such as the number of samples, different multiplication methods, and median module reusing.

ED demonstrated superior performance with both thresholding algorithms and used fewer resources. It not only reaches a larger peak accuracy but also enables adaptability to multiple noise levels. On the other hand, the proposed median approximation makes the median operation achievable in hardware with an acceptable budget and easily scalable. Using the median approximation consumes more power and resources than the average thresholding, but it results in better spike detection performance and improved robustness to signal variations. At low noise levels, the median approach does not require tuning in multiplier choice, regardless of which emphasizer is used.

REFERENCES

[1] J. L. Collinger, B. Wodlinger, J. E. Downey, W. Wang, E. C. Tyler-Kabara, D. J. Weber, A. J. McMorland, M. Velliste, M. L. Boninger, and A. B. Schwartz, "High-performance neuroprosthetic control by an individual with tetraplegia," *The Lancet*, vol. 381, no. 9866, pp. 557–564, 2013.

[2] C. Pandarinath, P. Nuyujukian, C. H. Blabe, B. L. Sorice, J. Saab, F. R. Willett, L. R. Hochberg, K. V. Shenoy, and J. M. Henderson, "High performance communication by people with paralysis using an intracortical brain-computer interface," *Elife*, vol. 6, p. e18554, 2017.

[3] F. R. Willett, D. T. Avansino, L. R. Hochberg, J. M. Henderson, and K. V. Shenoy, "High-performance brain-to-text communication via handwriting," *Nature*, vol. 593, no. 7858, pp. 249–254, 2021.

[4] A. B. Rapeaux and T. G. Constandinou, "Implantable brain machine interfaces: first-in-human studies, technology challenges and trends," *Current opinion in biotechnology*, vol. 72, pp. 102–111, 2021.

[5] R. D. Flint, Z. A. Wright, M. R. Scheid, and M. W. Slutzky, "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.

[6] R. Quian Quiroga and S. Panzeri, "Extracting information from neuronal populations: information theory and decoding approaches," *Nature Reviews Neuroscience*, vol. 10, no. 3, pp. 173–185, 2009.

[7] R. A. Andersen, S. Musallam, and B. Pesaran, "Selecting the signals for a brain–machine interface," *Current opinion in neurobiology*, vol. 14, no. 6, pp. 720–726, 2004.

[8] R. D. Flint, Z. A. Wright, M. R. Scheid, and M. W. Slutzky, "Long term, stable brain machine interface performance using local field potentials and multiunit spikes," *Journal of neural engineering*, vol. 10, no. 5, p. 056005, 2013.

[9] K. M. Silay, C. Dehollain, and M. Declercq, "Numerical analysis of temperature elevation in the head due to power dissipation in a cortical implant," in *2008 30th annual international conference of the IEEE engineering in medicine and biology society*. IEEE, 2008, pp. 951–956.

[10] 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.

[11] S. Luan, I. Williams, M. Maslik, Y. Liu, F. De Carvalho, A. Jackson, R. Q. Quiroga, and T. G. Constandinou, "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.

[12] Y. Yang, C. S. Boling, A. M. Kamboh, and A. J. Mason, "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.

[13] M. H. Malik, M. Saeed, and A. M. Kamboh, "Automatic threshold optimization in nonlinear energy operator based spike detection," in *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, 2016, pp. 774–777.

[14] S. Shaikh, R. So, C. Libedinsky, and A. Basu, "Experimental comparison of hardware-amenable spike detection algorithms for ibmis," in *2019 9th International IEEE/EMBS Conference on Neural Engineering (NER)*. IEEE, 2019, pp. 754–757.

[15] L. Schaffer, S. Pletl, and Z. Kincses, "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.

[16] Z. Zhang and T. G. Constandinou, "Adaptive spike detection and hardware optimization towards autonomous, high-channel-count bmis," *Journal of Neuroscience Methods*, vol. 354, pp. 109–103, 2021.

[17] Y.-G. Li, Q. Ma, M. R. Haider, and Y. Massoud, "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.

[18] H. Semmaoui, J. Drolet, A. Lakhssassi, and M. Sawan, "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.

[19] J. H. Choi, H. K. Jung, and T. Kim, "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.

[20] R. Q. Quiroga, Z. Nadasdy, and Y. Ben-Shaul, "Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering," *Neural computation*, vol. 16, no. 8, pp. 1661–1687, 2004.

[21] J. Navajas, D. Y. Barsakcioglu, A. Eftekhar, A. Jackson, T. G. Constandinou, and R. Q. Quiroga, "Minimum requirements for accurate and efficient real-time on-chip spike sorting," *Journal of neuroscience methods*, vol. 230, pp. 51–64, 2014.

[22] Z. Zhang and T. G. Constandinou, "Selecting an effective amplitude threshold for neural spike detection," *bioRxiv*, 2022.