Accepted for publication, J. of Neural Engineering, 2005.

Algorithms and Architectures for Low Power Spike Detection and Alignment

Alex Zviagintsev, Yevgeny Perelman and Ran Ginosar
VLSI Systems Research Center
Technion—Israel Institute of Technology, Haifa 3200, Israel
[alehan@tx.technion.ac.il]

Abstract: We introduce algorithms and architectures for automatic spike detection and alignment that are designed for low power. Some of the algorithms are based on principal component analysis. Others employ a novel Integral Transform analysis and achieve 99% of the precision of a PCA detector, while requiring only 0.05% of the computational complexity. The algorithms execute autonomously, but require off-line training and setting of computational parameters. We employ pre-recorded neuronal signals to evaluate the accuracy of the proposed algorithms and architectures: The recorded data are processed by a standard PCA spike detection and alignment software algorithm, as well as by the several hardware algorithms, and the outcomes are compared.

I. INTRODUCTION

Automatic and semiautomatic approaches to analysis of neuronal activity have been the subject of extensive research [1][2]. A typical setup for a neuronal recording experiment in an animal or human subject requires high bandwidth communications between the recording electrodes and the processing computer, where spikes are detected and sorted. When a large number of recording electrodes is employed, typical transmission resources are insufficient and power-hungry [3]. In addition, the large number of wires results in heavy cables that severely constrain the subject. Consequently, it is desirable to pre-process and reduce the volume of the recorded data so that it can be transmitted wirelessly.

We investigate the *Neuroprocessor*, an implantable integrated circuit that performs all neuronal interface and processing tasks, including spike acquisition, signal processing, and stimulation. In particular, it should contain power-efficient front-end processing of spikes, in order to minimize the communication bandwidth between the recording electrodes and the back-end computer [4]. In this paper we focus on *detection and alignment* (D&A) of spikes [5] as a pre-requisite to successful on-chip spike sorting [6][7]. For instance, given a sampling rate of 24Ksps and 12 bit sampling precision, the raw data rate is 288Kbits/second per electrode. Spike D&A enables transmission of only active spike data and filtering out the inter-spike noise [8]. Assuming a high rate of 100 spikes/sec/electrode and 2msec/spike, D&A reduces the data rate to 60Kbits/sec. If spike sorting were also added to the front-end processing, each spike would be converted into a short datagram (~20 bits), reducing the required data rate down to 2Kbits/sec per electrode, less than 1% of the original rate.

The computational task of such data reduction acquired by tens or hundreds of electrodes typically requires special purpose hardware. Conventional CPU would either be too large and dissipate too much power for a portable device, or would be too slow for the job. The special purpose hardware must implement a custom-tailored architecture that is carefully

tuned to perform the desired algorithm. In this paper we investigate the algorithms together with the architectures that implement them.

The most limiting constraint on implantable chips for spike detection of many electrodes is power dissipation [3]. While exact prediction of power requirements without completely designing the circuits is elusive, we investigate the computational complexity of several D&A algorithms as a reasonable predictor of their power. Before undertaking the complex endeavor of hardware design, an architectural comparative study is called for. Computational complexity is used to compare alternative algorithms and architectures.

The other figure of merit for D&A is the accuracy of subsequent spike processing, which depends heavily on the quality of D&A. We consider algorithms and architectures that trade off some subsequent classification accuracy in return for significant savings in power. The most favorable architecture is shown to achieve 99% of the accuracy of a "standard" algorithm, while incurring only 0.05% of its computational complexity. A similar approach has been presented in [9].

Spike detection algorithms have been discussed previously in [3],[9]-[14]. An adaptive threshold detection circuit that did not perform alignment was described in [3]. Computational complexity of several detection algorithms was discussed in [9]. D&A algorithms for multielectrode arrays were presented in [10], but power dissipation was not optimized. Wavelet based detection were described in [11]-[13], but without analysis of hardware complexity and power requirements. Detection by threshold crossing and initial alignment around the point of maximum slope was described in [14].

Section II presents the spike-processing section of the Neuroprocessor architecture in a top-down manner. D&A algorithms and architectures are specified in Section III, and their performance is compared in Section IV.

II. System Overview

In typical neuronal recording experiments, the signals recorded by the electrodes are amplified and transmitted over wires to a host computer where they are digitized and processed according to the experimental requirements [15]. The main disadvantage of that experimental arrangement is the need to connect a cable to the subject, restricting its movement. We investigate the *Neuroprocessor*, an implantable integrated circuit that performs all neuronal interface and processing tasks, including spike acquisition, signal processing, and stimulation. Its acquisition stage contains front-end analog processing, spike detection, alignment and sorting. Significant data reduction is achieved enabling wireless communications to replace cables and allow free movement of the patient or the test subject.

The architecture of the part of the Neuroprocessor that processes the signal from a single electrode is shown in Figure 1. Spike processing must be adaptable, due to unstable recording conditions [3][14]. Therefore, periodically, raw data is transmitted to the host computer for re-training, and the recalculated parameters are sent back to the Neuroprocessor (training algorithms, as well as the important question of how often they need to be executed, are beyond the scope of this paper). The Spike Detector detects the presence of spikes in the input, determines their starting point, and initiates the operation

of the Spike Sorter. The output logic produces the spike notification datagram. Although we consider a single channel in this paper, the results can be extended to Neuroprocessors handling multi-electrode arrays. Some of the hardware may be shared among the channels, if appropriate.

Performance of the spike sorter depends critically on the accuracy of the D&A algorithm. In this paper we focus on the D&A algorithms and assume a given Spike Sorter.

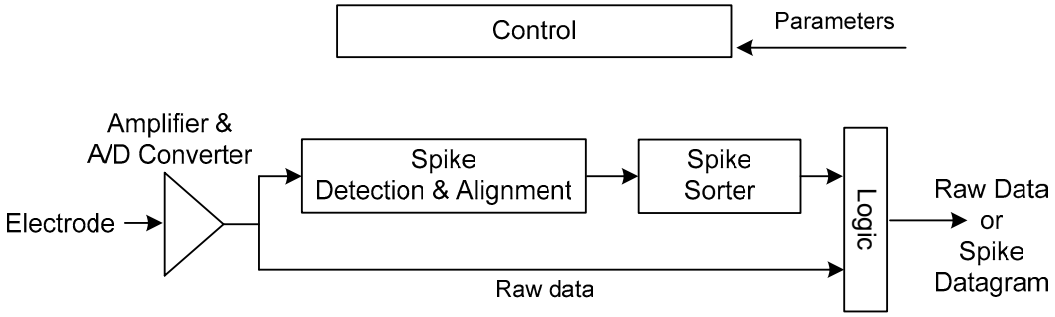

Figure 1: Neuroprocessor channel architecture: Each channel processes the signals of a single electrode, and comprises an analog and digital front-end and a programmable spike detector / sorter .

III. Hardware Algorithms and Architectures for Detection and Alignment

When exploring VLSI architectures for real time spike sorting to be carried out at the head-stage, we seek to minimize the required resources while still achieving acceptable levels of accuracy. The primary goal is to minimize power dissipation. Following [16], we consider the relative computational complexity of several architectures as a predictor of their power requirements.

Three different spike detection and alignment architectures and algorithms are considered: *Maximum Projection Alignment* (MPA), *Maximum Integral Transform Alignment* (MITA), and *Segmented PCA* (K-PCA).

Spike detection for all algorithms is based on threshold crossing. Threshold values are obtained by off-line training on the host computer and downloaded into the Neuroprocessor. They are set to optimize the ratio between the amount of threshold-crossing events due to the background noise (false alarms) and the amount of missed spikes.

We assume that most spike waveforms can be represented as a linear combination of a set of base vectors. Moreover, spikes generated by the same neuron are clustered together in the space spanned by the base vectors. The alignment procedure is based on the correlation of the input signal with the base vectors, which are determined off-line and downloaded to the Neuroprocessor.

A. Maximum Projection Alignment (MPA) Algorithm

The *Maximum Projection Alignment* (MPA) algorithm computes the correlations of the input signal with the first two principal components [17]. The spike is aligned to the point

of maximal correlation of the input signal with the first PC. It is a VLSI-oriented version of a common software detection algorithm [18]. The MPA algorithm comprises two steps: *extraction* and *alignment*, as follows.

1) MPA Extraction

During extraction, a segment of $M=K+N$ samples of the input signal is acquired. Extraction is triggered by threshold crossing at the input. The first K samples precede the triggering crossing event, and the remaining N samples follow it.

2) MPA Alignment

The alignment step seeks a spike of N samples within the M samples segment, starting at an offset $i \in \{1,...,K\}$ from the start of the segment. MPA alignment selects offset i that yields the maximum (absolute value of the) correlation of the segment with the first principal component (PC1). Thus, it finds offset i such that

$$P^1 = \max_{i=1}^{K} \left\{ \left| \sum_{r=1}^{N} s_{r+i-1} \mu_r \right| \right\} \tag{1}$$

where s_j are signal samples and μ_r are elements of PC1.

A VLSI architecture for the MPA algorithm is shown in Figure 2. The input is digitized by the ADC (analog to digital converter) and transferred through a FIFO. The Threshold block triggers operation of the PC1 projection unit, which computes the K projections according to (1).

We take advantage of the observation that the correlation function in (1) typically shows a single maximum over the entire range $i=1,\ldots,K$. The FindMax unit computes the maximum of P_i^1 by comparing each pair of consecutive projections. Upon detecting a maximum P^1, P^2 is also computed and both projections are sent to subsequent spike sorting. If no maximum is detected within the range of possible offsets, the last offset is selected (the spike is aligned with threshold crossing).

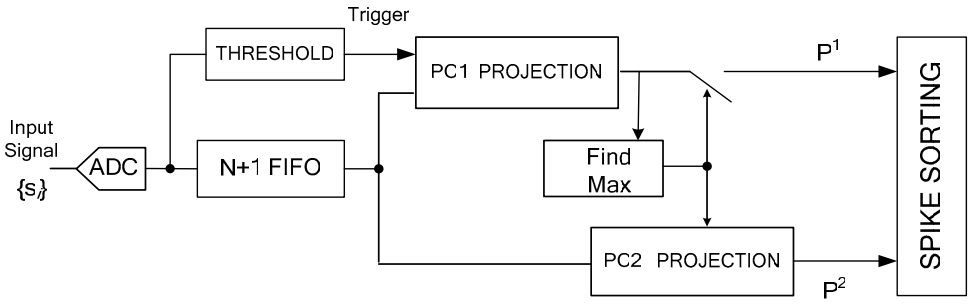

Figure 2: A VLSI architecture for the MPA algorithm

The MPA architecture provides an efficient real-time implementation of the common software PC-based detection algorithm. One potential shortcoming of MPA is that it employs only PC1 for computing the alignment.

B. Maximum Integral Transform Alignment (MITA) Algorithm

The *Maximum Integral Transform Alignment* (MITA) algorithm is based on separate integration of the positive and negative phases of the spike. The integral values are useful both for detection and for subsequent spike sorting [6]. As in the MPA case, the MITA algorithm can be divided into two steps, extraction and alignment. The extraction step is identical to that of MPA, whereas alignment uses a different set of base vectors for correlation. Figure 3 shows a typical spike. We define two time windows, α and β, matched (by off-line training) to the principal phases of the spike, positive and negative. Once trained, the sizes and relative positions of the two time windows remain fixed. The MITA algorithm computes the integral A of the input signal over window α, at all possible offsets, and selects the offset which yields the maximal absolute value of A. Once alignment has been determined, integral B is computed over the β window, the two integral values are produced at the output and can be employed for subsequent spike sorting.

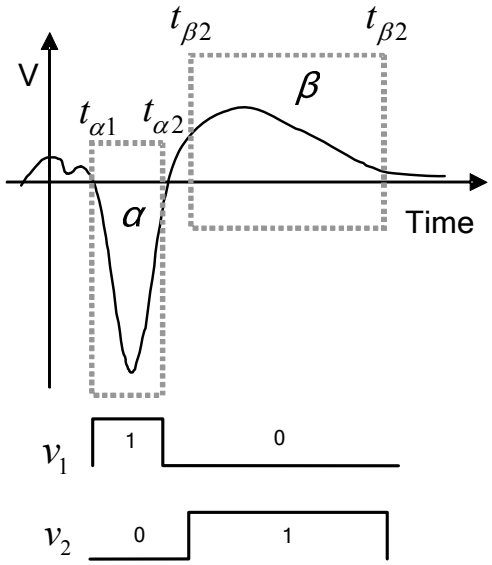

Figure 3: Spike projection on integral space.
The integrals over windows α and β can be considered as correlating the input signal with the vectors v_1 and v_2 ,

Computing the two integrals can be considered as correlating the input signals with the following two vectors:

$$v_1(r) = \begin{cases} 1, & t_{\alpha 1} \le r \le t_{\alpha 2} \\ 0, & otherwise \end{cases} \qquad v_2(r) = \begin{cases} 1, & t_{\beta 1} \le r \le t_{\beta 2} \\ 0, & otherwise \end{cases} \tag{2}$$

A VLSI architecture for the MITA algorithm is shown in Figure 4. Since integrals A and B do not overlap in time, we first compute integral A, find the spike alignment, and only then compute integral B.

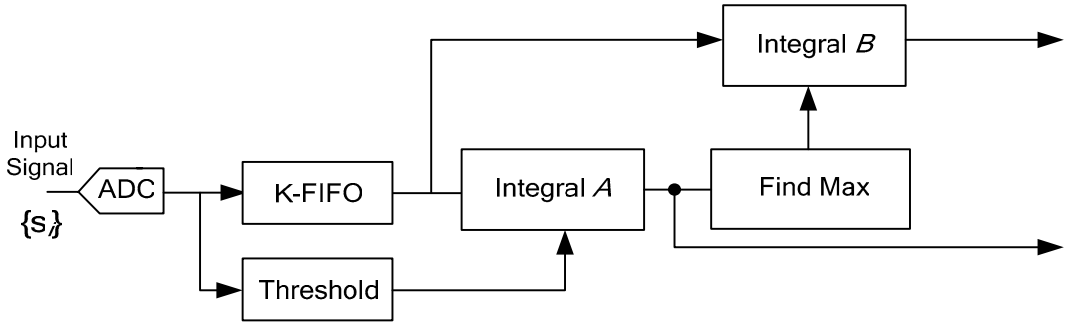

Figure 4: A VLSI architecture for the MITA algorithm.

Integration is implemented with a Moving Average filter, computing the sum of elements held in a K-stage FIFO. With every input sample, the oldest element is removed from the sum and the new sample is added. Consider the first integral,

$$A_i = \sum_{r=1}^{\alpha} s_{i+r-1}, \quad \alpha = t_{\alpha 2} - t_{\alpha 1} \tag{3}$$

Then

$$A_i = A_i + s_{i+\alpha+1} - s_i \tag{4}$$

A recursive implementation of a Moving Average filter for the A integral is depicted in Figure 5. Unlike the MPA architecture, there is no need to maintain K sums in parallel. Initially, the A-FIFO contains α zeroes. During the first α steps, the accumulator computes A1. Henceforth, one old element is subtracted from A1 and a new one is added. Thanks to eliminating multiplications, the MITA architecture incurs a lower hardware cost than MPA.

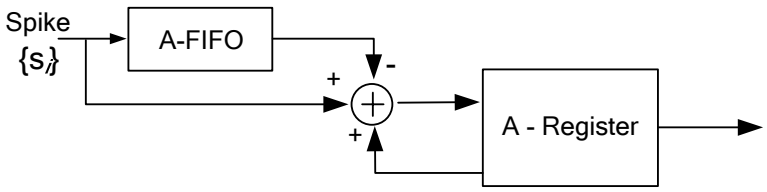

Figure 5: Architecture of the recursive Moving Average filter .

The following comments may be noted regarding the MITA algorithm. First, it is inspired by the observation that integral *A* shows a single maximum near the threshold. Second, aligning to the maximum of the signal integral results in lower noise sensitivity than aligning to the maximum of either the signal itself or its derivative (as proposed, e.g., in [14]). Third, an even more robust algorithm could apply threshold detection to the integral *A* rather than to the original signal.

The foregoing description of the MITA algorithm assumes that the two typical spikes recorded by the same electrode have roughly the same support, so that windows α and β roughly coincide with the first and second phases of both typical spikes, respectively. In many other cases, it is possible to either place the two windows at positions that do not necessarily overlap the first and second phases of either spike, but would still be sufficient for successful alignment (and even sorting). In yet other cases, it may be possible to use three windows instead of two.

C. Principal Component Detection

A common software algorithm for D&A is based on principal component analysis (PCA) [18]. For each potential offset, the signal is projected on the first two principal components, and those projections are used to estimate the signal. The offset which results in minimum estimation error is selected as the best alignment. Formally, the projections at offset *i* are

$$P_i^1 = \sum_{r=1}^{N} s_{r+i-1}\mu_r^1 \ , \quad P_i^2 = \sum_{r=1}^{N} s_{r+i-1}\mu_r^2 \qquad i = 1,\ldots,K \tag{5}$$

The estimated signal at offset *i* is the vector

$$\hat{\boldsymbol{s}}_i = P_i^1 \cdot \boldsymbol{\mu}^1 + P_i^2 \cdot \boldsymbol{\mu}^2 \tag{6}$$

and the algorithm seeks *i* that minimizes the error

$$Err(i) = \sum_{r=1}^{N} \left(s_{i+r} - \hat{\boldsymbol{s}}_{i,r} \right)^2 \tag{7}$$

VLSI architecture for on-chip D&A by means of PCA is shown in Figure 6. The input is transferred through a FIFO register of *K* stages. The Threshold unit triggers operation of the Estimation unit. The Estimation unit computes the 2*K* projections on the two PC vectors (two projections at each offset *i*) and produces the estimated signal per each *i*. Once the Min Error unit finds the offset that yields the minimal estimation error, the corresponding projections P_i^1 and P_i^2 are sent to the output. If no minimum is detected within the range of possible offsets, the last offset is selected.

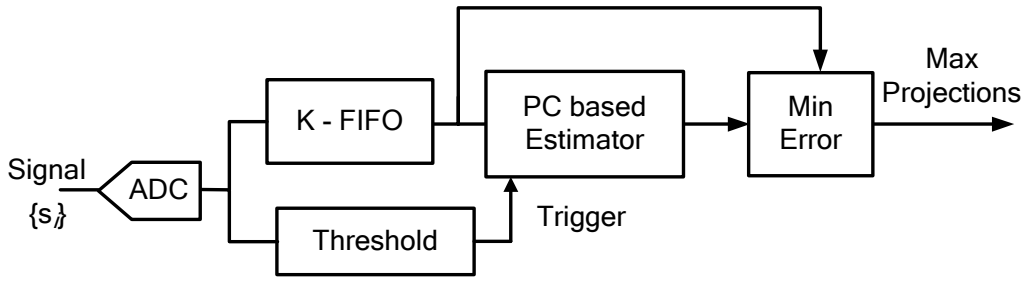

Figure 6: A VLSI architecture for PCA-based detection and alignment

D. Segmented PC Detection

The *segmented PC* ("K-PC") algorithm approximates PCA using a reduced number of multiplications. This is achieved by down-sampling the principal components. The signal is integrated over several time intervals (k_1 intervals for PC1 and k_2 intervals for PC2). Each integral is multiplied by the average of the principal component values over the respective interval, as follows:

$$\Phi_1 = \sum_{p=1}^{k_1} \Delta_p^1 \cdot \gamma_p^1 \tag{8}$$

where

$$\Delta_p^1 = \sum_{i \in \text{interval } p} s_i, \qquad \gamma_p^1 = \underset{i \in \text{interval } p}{Average} \left\{ \mu_i^1 \right\} \tag{9}$$

The expressions for Φ_2 are similar. Segmentation helps reduce only the rate of multiplications. The k integrals are computed by sliding windows similar to the moving average of Figure 5, incurring $2(k_1+k_2)$ additions per input sample. The two projections require additional k_1+k_2 multiplications and additions per each input sample. A VLSI architecture for the K-PC algorithm is shown in Figure 7.

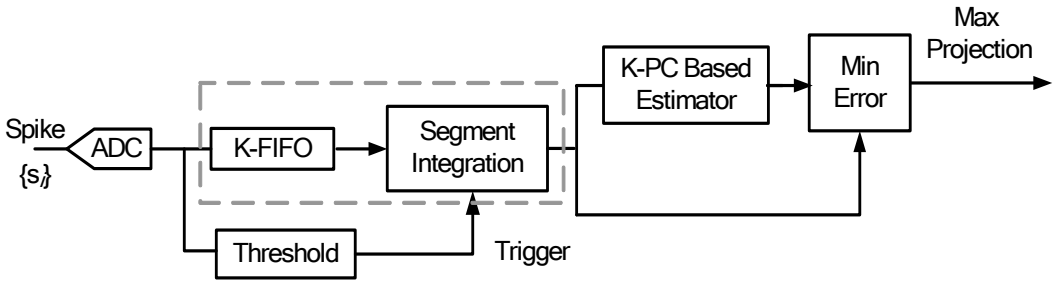

Figure 7: A VLSI architecture for the K-PC algorithm.

Another level of savings in computational complexity can be achieved by approximating the multiplication coefficients $\gamma^{1,2}_p$ by powers of 2 (the multiplications can be achieved by simple bit-shifting).

IV. RESULTS

A. Algorithm Validation

In order to evaluate our methods, all D&A algorithms are applied to the same data set, comprising a large number of digitized signals obtained from neuronal recordings. The signals are very similar to real spikes in the time domain.

We compare our results with a PCA based D&A algorithm [18]. A spike sorting algorithm (a standard PCA-based software sorter [18]-[20]) is executed subsequently to D&A, and the comparison is based on the accuracy of the sorting. Several other papers evaluate and compare D&A algorithms by considering the minimal distance between a noise-free template of the expected spike and the detected one [11][14][21][22]. However, since detection in our application is merely a preparation for sorting, it is more appropriate to evaluate D&A not in its own right but based on how it affects sorting. Note that such an approach may not apply to action potentials with significantly different shapes or SNR levels.

Figure 8 illustrates the algorithm validation scheme. First, part of the data is used for off-line training, producing configuration parameters for the hardware algorithm. Second, the parameters are downloaded into the Neuroprocessor. Third, the Neuroprocessor hardware D&A algorithm is applied to the entire data set. The software D&A algorithm is also applied to the same data; and the results of both hardware and software algorithms are processed by the software spike sorter [20] and compared. The results are reported in Table 1 below.

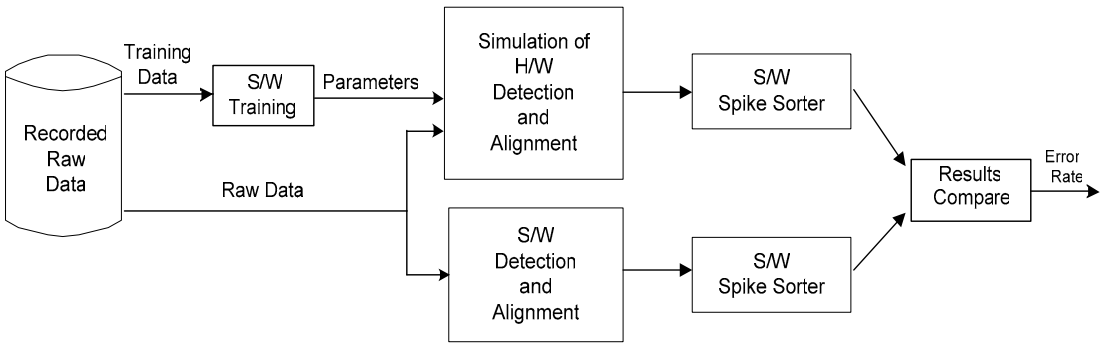

Figure 8: Validation scheme for detection and alignment algorithms .

B. Spike Data Preparation

Real spike data was taken from electrophysiological recordings of multiple spike trains, obtained from various cortical neurons [23]. Neuronal signals from the electrodes were amplified, bandpass filtered (300 – 6000 Hz, four poles Butterworth filter), and sampled at 24 Ksps/electrode. The data is up-sampled 4 times for improved alignment precision.

Spikes last about 2 msec, resulting in 200 samples per spike. Software spike detection and alignment were first applied on the data sets; only stable spike trains (as judged by stable spike waveforms and stable firing rate) were included in this study. The data set contained about 1,000 spikes per cluster. Samples of the recorded spikes and their projections on PC space are shown in Figure 9. To validate the performance of the proposed architectures on a particularly hard case, we have chosen one with near-by clusters sharing an edge (Figure 9b).

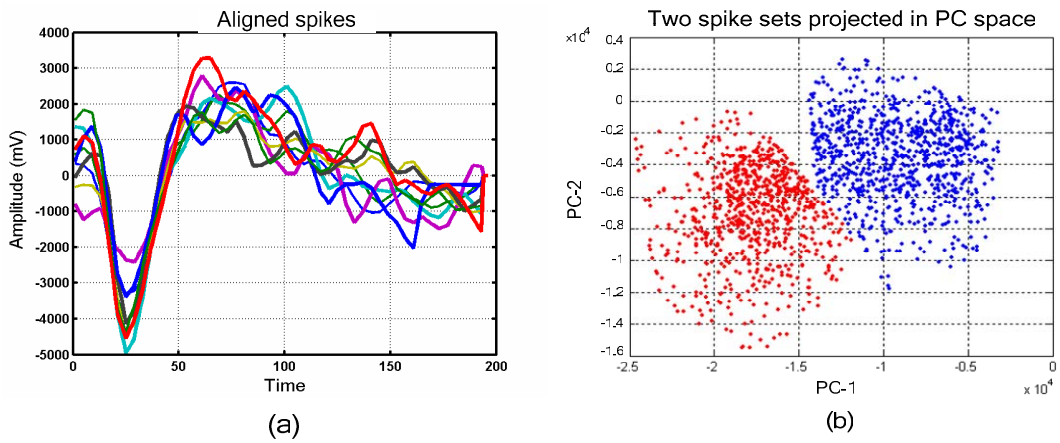

Figure 9: Neural spikes and their projection on PC space:
(a) Several recorded spikes after extraction and alignment,
(b) Clusters in PC space.

For evaluation of the hardware algorithms (implemented and simulated in MATLAB), segments of background noise, recorded form the same system, were added at the beginning and the end of every spike. An example of one such segment is shown in Figure 11a. That way we know the starting point and the cluster of every spike; yet such an arrangement is very close to the real data.

The principal cost measure of the various architectures is their computational complexity, which is roughly related to their power consumption. We count the number of additions and multiplications required in every algorithm for processing a single spike. Multiplication is counted as about ten additions, and computational complexity is expressed in the total number of equivalent additions.

C. Analysis of Spike D&A Algorithms

The results of a linear classifier applied to the output of the MPA D&A algorithm are shown in Figure 10. Most classification errors occur near the common edge of the two clusters. Compared to the PCA off-line algorithm, developed by Alpha-Omega Eng., [18], the MPA algorithm obtains similar results: the sorting error is small (0.3%, Table 1), and the cluster appear visually identical (see Figure 9b).

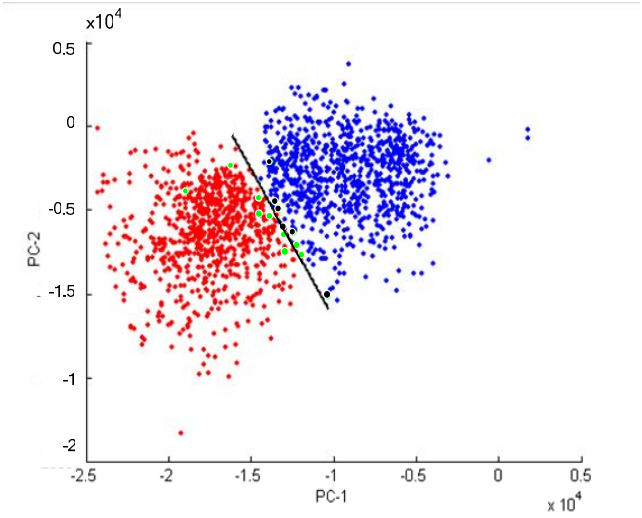

Figure 10: PC space representation of spikes detected by the MPA algorithm.
Mis-classified spikes are indicated as green and black points.

.

The operation of the MPA algorithm is illustrated on Figure 11. A sample input signal is shown as a blue waveform in Figure 11a. MPA continuously computes the correlation of the input signal with the first PC vector. This can be represented as a trace of points on the PC space (Figure 11b). When no spike is present at the input, the trace fluctuates around zero, as a result of correlation with the background noise. When a spike is present, the trace leads away from the origin, and the peak corresponds to the alignment point of the spike. The red waveform in Figure 11a has been reconstructed from the PC1 and PC2 values of the "best alignment" point in Figure 11b.

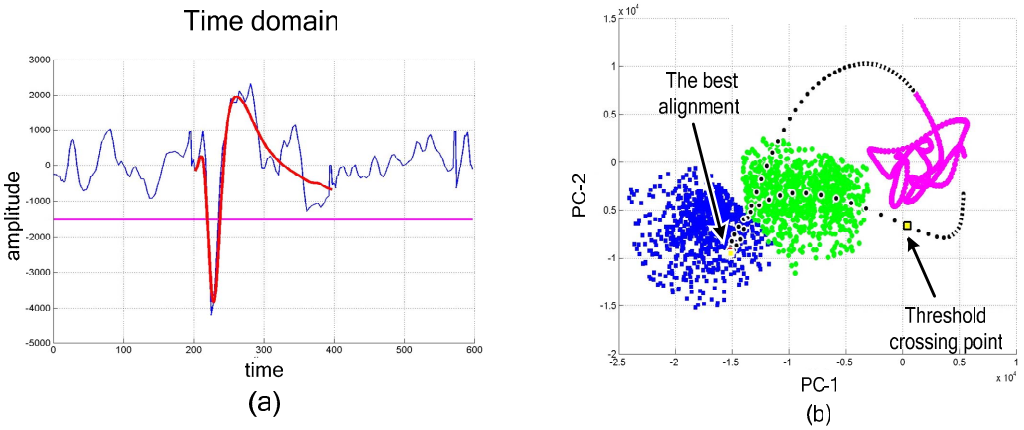

Figure 11: Spike detection and alignment using the MPA algorithm:
(a) Synthetic signal combining noise segments with a real spike (blue) and a best aligned reconstructed spike (red) (b) Trace of points on the PC space, progressively generated during the alignment process

Figure 12a shows an example where the offsets computed by MPA and by off-line PCA differ. The trace in Figure 12b represents the alignment process, and the green part of the trace shows points following threshold crossing. The two circles on PC space show that the two offsets lead to different mappings. These two mappings result in the two different

reconstructed spikes in the time domain (Figure 12a). However, both results map the spike into the same cluster.

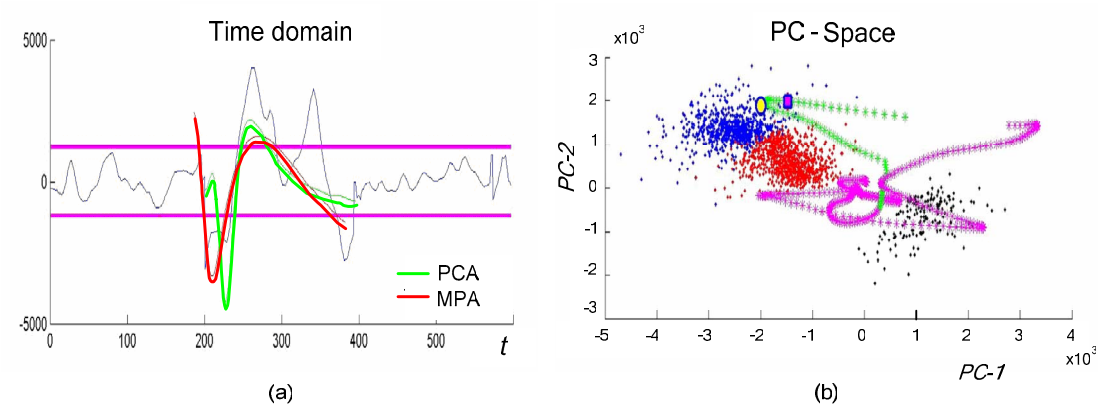

Figure 12: Spike detection and alignment using MPA and off-line PCA algorithms:
(a) Spikes reconstructed from estimations by MPA and PCA,
(b) Trace of points on the PC space. Both alignments map the spike onto the same cluster

Similar behavior characterizes the remaining simulated algorithms. Figure 13 shows the results of the MITA algorithm projected on PC space and linearly classified using SVM technique [24].

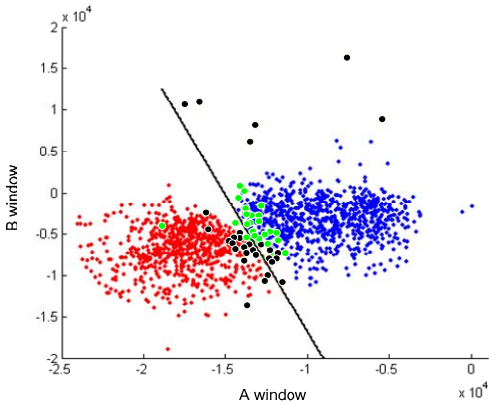

Figure 13: Integral space representation of spikes detected by the MITA algorithm.
Mis-classified spikes are indicated by green and black points.

D. Comparison

The computational complexity and error rate of the various spike D&A algorithms and architectures are compared in Table 1 and Figure 14 for K=50 and spikes of 200 samples. The algorithms were applied to a difficult data set, in which the two spike clusters were very close to each other. The error rate of the algorithms was maximal in this case (when

clusters are further apart, lower detection and alignment error rates are obtainable). For the sake of comparison, we also performed an experiment based solely on maximum detection (marked Maximum in the table), where spikes are aligned to the first maximum of the signal following threshold crossing.

TABLE 1: COMPUTATIONAL COMPLEXITY AND CLASSIFICATION ERRORS OF SPIKE DETECTION AND ALIGNMENT ALGORITHMS

Algorithm	Additions	Multiplications	Computational Complexity	Classification Error
Maximum	50	0	50	9.4%
MITA	250	0	250	1.2%
7-PCA	2,600	1,750	20,000	0.7%
MPA	10,250	10,200	112,250	0.3%
PCA	50,000	50,000	550,000	0.0%

K=50, N=200, M=K+N=250

Based on these results, three observations can be made: (a) the MPA algorithm performs as well as the software D&A, but incurs a high computational complexity; (b) the MITA algorithm achieves about 99% precision at about 0.05% of the complexity (relative to PCA); (c) MITA constitutes the "knee point" of the complexity versus error graph, and is thus suggested as the preferred architecture.

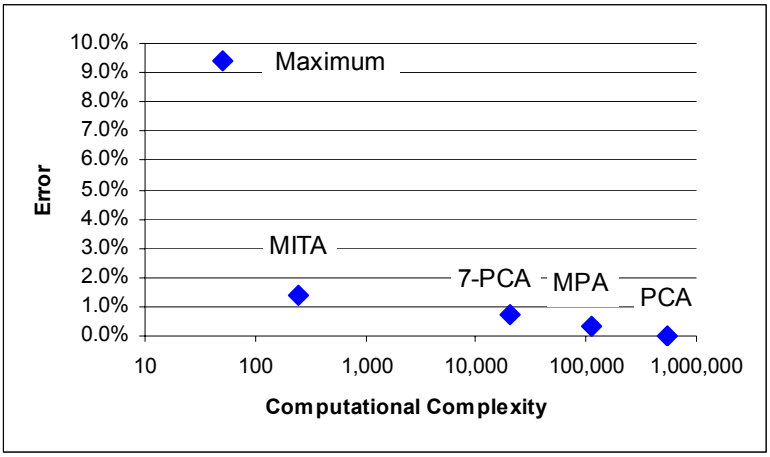

Figure 14: Classification error vs. computational complexity of several spike detection and alignment algorithms

V. CONCLUSIONS

We have considered low-power architectures and algorithms for spike detection and alignment (D&A). Such systems may be useful for implanting near recording electrodes, or for using in large multi-electrode arrays, in either research or clinical applications. These systems enable substantial reduction of the communication bandwidth, which is essential when a large number of recording electrodes is involved.

Three VLSI architectures have been described and analyzed *Maximum Projection Alignment (MPA), Maximum Integral Transform Alignment (MITA),* and *Segmented PC (K-PC)*. The algorithms have been simulated with real data obtained from neuronal recordings. The results are analyzed in terms of classification errors (relative to sorting achieved with software PCA classification) and computational complexity (estimated based on the number of additions and multiplications). The MITA algorithm yields only marginal accuracy degradation relative to MPA, while incurring only a very small fraction of the computational complexity. Thus, we have selected the MITA algorithm for power-efficient spike detection in a neuronal processing integrated circuit.

ACKNOWLEDGMENT

This research was funded in part by a grant from the Office of Chief Scientist, Israel Ministry of Industry and Trade. Guidance from Moshe Abeles, Hagai Bergman, Eilon Vaadia, Shimon Marom, Izhar Bar-Gad and Alpha-Omega, Inc. is greatly appreciated. Constructive comments by the anonymous referees helped improve this paper significantly.

REFERENCES

[1] Lewicki M.S., "A review of methods for spike sorting: the detection and classification of neural action potentials." *Network: Comp. Neural Syst.* 9(4): 53-78, 1998.

[2] Schmidt E. “Computer separation of multi-unit neuroelectric data: A review”. *J Neurosc Meth.,* 12:95-11. 1984.

[3] Harrison R., “A low-power integrated circuit for adaptive detection of action potentials in noisy signals,” *Proc. Inl. Conf. IEEE EMBS,* 17-21, 2003.

[4] Perelman Y. and Ginosar R., “An Integrated System for Multichannel Neuronal Recording with Spike / LFP Separation and Digital Output,” *2nd Int. IEEE EMBS Conf. Neural Eng.,* 377-380, 2005.

[5] Zviagintsev A., Perelman Y. and Ginosar R., “A Low-Power Spike Detection and Alignment Algorithm,” *2nd Int. IEEE EMBS Conf. Neural Eng.,* 317-320, 2005.

[6] Zviagintsev A., Perelman Y. and Ginosar R., “Low-Power Architectures for Spike Sorting,” *2nd Int. IEEE EMBS Conf. Neural Eng.,* 162-165, 2005..

[7] Wheeler B.C., Nicolelis M.A., “Automatic discrimination of single units,” *in Methods for Neural Ensemble Recordings, Ed. Boca Raton, FL: CRC Press LLC,* 1999.

[8] Moxon K., Morizio J., Chapin J., Nicolelis M., Wolf P. “Designing a brain-machine interface for neuroprosthetic control,” *in Neural Prostheses for Restoration of Sensory and Motor Function, J.K.Chapin and K. A. Moxon, Eds. Boca Raton, FL: CRC Press,* 2001.

[9] Obeid I., Wolf P., “Evaluation of Spike-Detection Algorithms for a Brain-Machine Interface Application”, *IEEE Trans. Biomed. Eng.,* 51:905-911, 2004

[10] Guillory K.S. and Normann R.A., “A 100-channel system for real time detection and storage of extracellular spike waveforms,” *J. Neurosci. Meth.*,91:21–29, 1999.

[11] Nakatani H., Watanabe T., and Hoshiyama N., “Detection of nerve action potentials under low signal-to-noise ratio condition”, *IEEE Trans. Biomed. Eng.*,48:845-849, 2001.

[12] Oweiss K. G., and Anderson D. J., "A multiresolution generalized maximum- likelihood approach for the detection of unknown transient multichannel signals in colored noise with unknown covariance," *in Proc. ICASSP,* 2993–2996, 2002.

[13] Kim K.H. and Kim S.J., "A Wavelet-Based Method for Action Potential Detection From Extracellular Neural Signal Recording With Low Signal-to-Noise Ratio," *IEEE Tans. Biomed. Eng.*, 50:999-1011, 2003.

[14] Chandra R., Optican L., "Detection, classification, and superposition resolution of action potentials in multiunit single channel recordings by an on-line real-time neural network," *IEEE Trans. Biomed. Eng.,* 44: 403–412,1997.

[15] Nicolelis M., "Actions from thoughts," *Nature,* 409:403-407, 2001.

[16] Zumsteg Z., Ahmed R., Santhanam G., Shenoy K., Meng T., "Power Feasibility of Implantable Digital Spike-Sorting Circuits for Neural Prosthetic Systems," *Proc. 26th Ann. Int. Conf. IEEE EMBS,* 4237-4240, 2004.

[17] Abeles M, Goldstein MHJ., "Multispike train analysis," *IEEE Trans. Biomed. Eng*., 65:762–73, 1977.

[18] Bar-Gad I., Ritov Y., Vaadia E., and Bergman H., "Failure in identification of multiple neuron activity causes artificial correlations," *J. Neurosci. Methods*., 107: 1-13, 2001.

[19] Alpha Omega Engineering Ltd., Multi-Spike Detector (MSD).

[20] Alpha-Omega Engineering Co., *Ltd, Alpha Sort Ref. Manual,* 1996-2002.

[21] Atiya, A. F., "Recognition of multiunit neural signals," *IEEE Trans. Biomed. Eng.* 39:723–9, 1992.

[22] Lewicki M. S., "Bayesian modeling and classification of neural signals," *Neural Comput.* 6:1005–30, 1994.

[23] Eytan D., Brenner N. and Marom S., "Selective Adaptation in Networks of Cortical Neurons, " *J. Neurosci. Methods*. 23(28):9349-9356, 2003

[24] Chang, C.C. and C.J. Lin (2001). LIBSVM: a library for support vector machines. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.