

PAPER • OPEN ACCESS

Event-driven processing for hardware-efficient neural spike sorting

To cite this article: Yan Liu *et al* 2018 *J. Neural Eng.* **15** 016016

View the [article online](#) for updates and enhancements.

Related content

- [A non-parametric method for automatic neural spike clustering](#)
Z Tiganj and M Mboup
- [Noise-robust unsupervised spike sorting based on discriminative subspace learning with outlier handling](#)
Mohammad Reza Keshtkaran and Zhi Yang
- [A single-chip neural signal processing and telemetry engine](#)
Michael Rizk, Iyad Obeid, Stephen H Callender *et al.*

Event-driven processing for hardware-efficient neural spike sorting

Yan Liu^{1,3,4} , João L Pereira^{1,2} and Timothy G Constandinou^{1,3}

¹ Centre for Bio-Inspired Technology, Imperial College London, SW7 2AZ, United Kingdom

² Dept. of Bioengineering, Instituto Superior Técnico, 1049-001 Lisboa, Portugal

³ Dept. of Electrical & Electronic Eng., Imperial College London, SW7 2BT, United Kingdom

E-mail: yan.liu06@imperial.ac.uk

Received 18 May 2017, revised 28 September 2017

Accepted for publication 5 October 2017

Published 11 January 2018



Abstract

Objective. The prospect of real-time and on-node spike sorting provides a genuine opportunity to push the envelope of large-scale integrated neural recording systems. In such systems the hardware resources, power requirements and data bandwidth increase linearly with channel count. Event-based (or data-driven) processing can provide here a new efficient means for hardware implementation that is completely activity dependant. In this work, we investigate using continuous-time level-crossing sampling for efficient data representation and subsequent spike processing. **Approach.** (1) We first compare signals (synthetic neural datasets) encoded with this technique against conventional sampling. (2) We then show how such a representation can be directly exploited by extracting simple time domain features from the bitstream to perform neural spike sorting. (3) The proposed method is implemented in a low power FPGA platform to demonstrate its hardware viability. **Main results.** It is observed that considerably lower data rates are achievable when using 7 bits or less to represent the signals, whilst maintaining the signal fidelity. Results obtained using both MATLAB and reconfigurable logic hardware (FPGA) indicate that feature extraction and spike sorting accuracies can be achieved with comparable or better accuracy than reference methods whilst also requiring relatively low hardware resources. **Significance.** By effectively exploiting continuous-time data representation, neural signal processing can be achieved in a completely event-driven manner, reducing both the required resources (memory, complexity) and computations (operations). This will see future large-scale neural systems integrating on-node processing in real-time hardware.

Keywords: event driven, spike sorting, level crossing sampling, inverse derivatives

(Some figures may appear in colour only in the online journal)

1. Introduction

Electrophysiology provides a powerful tool for observing neural signals by recording extracellular biopotentials [1]. This allows us to study single unit activity that is of great importance to understanding fundamental mechanisms of the

brain, such as neural coding and plasticity [2]. Such high resolution information can also be used for decoding of intentions in neural prostheses [3, 4], and has great potential to impact clinical applications, for example epileptic seizure therapy [5].

With recent advances in microfabrication and integrated electronic technology, it is now possible to integrate electrodes with amplification and data conversion electronics, creating several hundreds of recording channels to monitor the spiking activity of neurons in real-time [1, 6, 7]. In fact, the number of channels in a single device has been doubling every 7 years [8]. On-node wireless data communication with a battery or

⁴ Author to whom any correspondence should be addressed.



Original content from this work may be used under the terms of the [Creative Commons Attribution 3.0 licence](https://creativecommons.org/licenses/by/3.0/). Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI.

wireless power transmission makes it possible for long-term implantable and portable instrumentation [9].

The recording data produced from thousands of channels raises hardware limitations in terms of power consumption and data throughput bandwidth. For instance, a system with 128 channels can easily require data rates in excess of 20 Mbps, most of which to accommodate non-active signal or redundant data, as single unit activity can be picked up by multiple adjacent electrodes [10]. Moreover, the power consumed in data conversion and processing dissipates heat to the adjacent biological tissue. The current literature states a maximum temperature increase allowed for cortical implants of 1 °C and a maximum 40 mW cm⁻² of heat flux [11].

Spike sorting, the method of classifying recorded spikes [12–16], can be integrated with the recording front-end to perform on-site processing and reduce the data bandwidth and power consumption. There have been numerous studies focusing on spike sorting in the last decades, most of them based on processing data sampled at a rate higher than the Nyquist frequency [17]. On-chip spike sorting has also been implemented with CMOS technology [18–20]. Previous works, however, rely on fixed synchronous sampling rates, which imposes a trade-off between accurately capturing the spikes and producing large amounts of data. Compressed sensing or feature classification are also used with reduced sampling rates, however these methods require hardware intensive algorithms, increasing the area of integrated implantable devices [21].

Therefore, efficient on-node neural spike sorting methods are critical for large-scale neural recording systems in chronic implants. These methods should preserve the essential information of spike activity, with minimum power consumption and reduced hardware resources.

In [22], we presented a hardware solution of a spike sorting system based on event-driven data conversion [23]. However the proposed implementation relied on a relatively complicated ADC, and only presented some preliminary details towards hardware spike processing.

In this paper we investigate the performance of event-driven data conversion in neural recording, and propose a fully asynchronous spike sorting method based on event-driven data conversion. Due to the sparse nature of extracellular recordings, as shown in figure 1(c), our claim is that event-driven sampling and signal processing can improve the overall efficiency of spike sorting implants by:

- Reducing the data bandwidth, due to the activity dependent sampling;
- Reducing the dynamic power consumption, due to both sampling and processing being purely activity dependent; and
- Allowing efficient feature extraction by using continuous-time based features.

We first compare the efficiency of level-crossing (LC) sampling with the Nyquist-rate uniform sampling (US) scheme. We then select a set of 4 simple event-driven features (EDF), namely the maximum, minimum and the incremental and decremental minimum inverse derivatives, that can be extracted on the fly using an asynchronous digital signal processor. We

develop noise filtering techniques based on pattern recognition to reduce the generated data and improve feature accuracy.

A hardware-demanding sorting algorithm can then be performed off-node using the reconstructed raw recording of LC sampling, to generate the templates for the selected features. The selection and optimisation of this algorithm is user dependant and out of scope of this work.

The generated templates can then be stored on-node and used to perform L1-norm based clustering. The sorting accuracies are subsequently benchmarked, relative to feature extraction complexity and feature dimensionality, against several hardware-feasible reference methods: template matching (TM) [17], principal component analysis (PCA) [24], first and second derivative extrema (FSDE) [25], and the equivalent features to those we propose, but extracted from a conventional US system (US_Eq).

The paper is organised as follows: section 2 summarises previous studies on spike sorting and asynchronous data conversion; section 3 describes the datasets and methodology used in this work. In section 4 we present the event-driven data conversion and EDF sorting results, while in section 5 we extend the discussion on these findings. Finally, in section 6, concluding remarks are made on this novel approach for spike sorting.

2. Background

2.1. On-node spike sorting stages

A conventional neural recording system with an embedded spike sorting engine is shown in figure 1(a). This consists on the analogue front-end, analogue-to-digital converter (ADC), and digital back-end, for signal processing [26]. The analogue front-end includes several stages of low-noise amplifiers and filters to remove out-of-band noise and unwanted signal components, such as local field potentials, and amplify the sensitive neural signal of interest for data conversion [1, 7].

The ADC converts the analogue signal to its digital equivalent using a constant sampling rate between 10–24 kHz. The number of bits used for the conversion determines the value of the least significant bit (LSB) i.e. the resolution. These parameters determine both the numerical accuracy and computational cost of the following signal processing stages.

The digital back-end for spike sorting includes spike detection, feature extraction and a classification system. Spike detection methods consist on thresholding the input signal, either directly, i.e. detection by amplitude threshold crossing [15], or after applying an operator with the aim of emphasising spikes, such as the non-linear energy operator [27], or matched-filters [28]. With each detection, the spike can be aligned to a reference point, such as its peak or maximum derivative [18], or left unaligned, which is equivalent to be aligned to the detection point.

Features are then extracted to reduce sampling dimension for classification. With TM [17], the n samples of each spike are used directly as features, i.e. no explicit feature extraction stage is needed. However, by extracting a smaller set of m features, such that $m < n$, the required memory and computational cost of classification can be reduced. Possible features

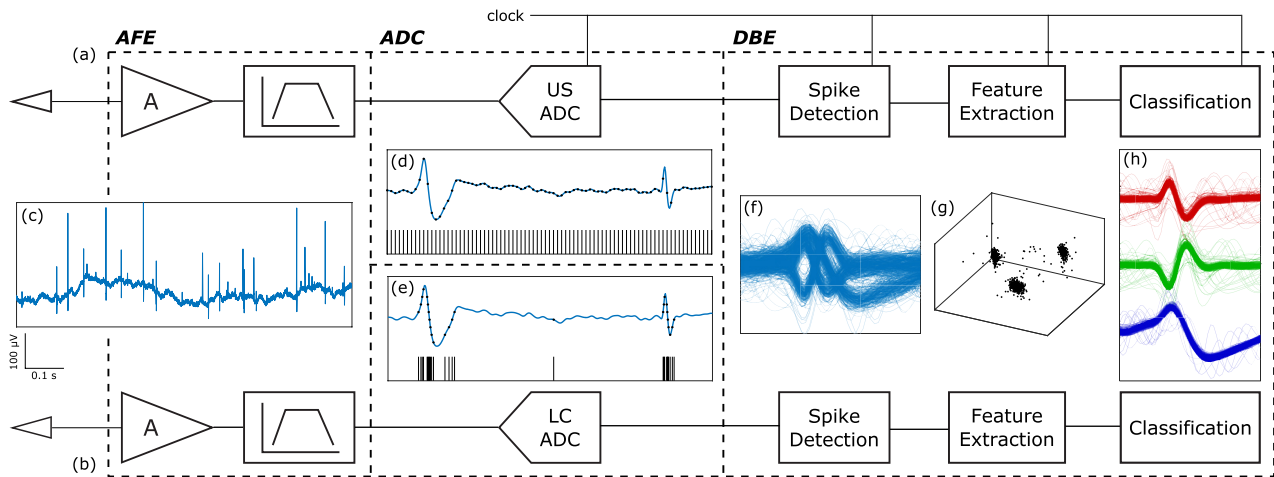


Figure 1. (a) Schematic of conventional US based spike sorting, composed of the analogue front-end, AFE, ADC and digital back-end, DBE, stages; (b) Event-driven data conversion and spike sorting; Example data: (c) Raw extracellular neural recording; (d) US based on the Nyquist rate; (e) Continuous-time LC based data conversion; (f) Detected spikes; (g) Clustering of spikes in the feature space; and (h) Sorted spikes.

include the maximum, minimum and width of peaks [29], zero-crossing features [30], principal components of spikes, obtained with PCA [24], the coefficients from the discrete wavelet transform [15], the derivatives of spikes [25, 31, 32] and the integral transform [33].

These features can then be clustered to identify the spike classes using methods such as k -means [25], superparamagnetic clustering [15] or others [32, 34]. Noise minimisation and dealing with overlapping spikes can also be performed beforehand to increase the sorting accuracy [35–37].

Conventionally, dimensionality reduction and spike sorting are performed offline due to the computational complexity and memory requirements of the algorithms typically used. However, computationally demanding algorithms can still be used offline for clustering (e.g. to determine the classes and define mean template features) and then low complexity functions can be implemented on-node for real-time classification (e.g. assigning spikes to pre-defined templates [17, 26, 38, 39]). Completely unsupervised on-chip sorting has also been recently reported based on computing the L1-norm distance [20]. However these algorithms still occupy a relatively large area, and thus are challenging to scale to large channel counts.

2.2. Event-driven sampling and signal processing

A constant sampling frequency is normally used in conventional neural recording systems. Nyquist’s sampling theorem needs to be met to preserve the spectrum of spikes. However, neuron activity is not driven by a consistent central clock. A fixed sampling frequency will inevitably compromise the accuracy of a recording based on the sample and hold theory, and possibly distort the signal due to aliasing.

On the other hand, extracellular recordings show a bursty behaviour, figure 1(c), with high frequency spikes intercalated by periods of lower activity. For a channel capturing spikes, each lasting 2 ms, from 3 different source neurons firing at 20 Hz each, around 88% of the data produced will not contain any useful information.

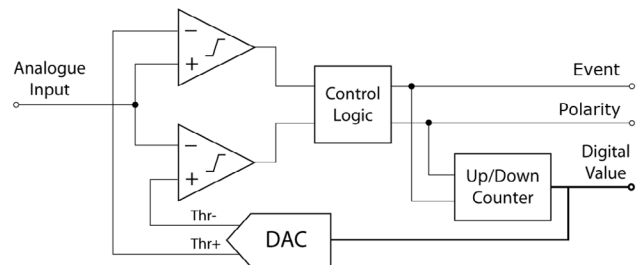


Figure 2. Generic architecture of a LC ADC [23]. No clock is required. All signals are a function of continuous-time, i.e. their value can change at any moment, driven by the activity of the input. DAC represents a digital-to-analogue converter.

Therefore, continuous-time ADCs based on the level-crossing scheme have been proposed for biomedical applications [40–42]. With this design, ADCs only operate when an event occurs, i.e. when the input signal crosses a certain threshold window, set dynamically by the ADC. Hence, the input is tracked closely, without a constant high sampling frequency, thus potentially producing fewer data, as shown in figures 1(d) and (e).

The information within the pulse train generated by LC ADCs is essentially another form of representing the recorded spikes, thus, rather than reconstructing the spikes in the time domain, an asynchronous digital signal processor [23, 43] can be used for spike sorting [22]. Furthermore, the pulse train itself can be used to trigger the following signal processing stages. Therefore, a completely activity dependent asynchronous sorting engine, driven by the level-crossing events, is possible, figure 1(b).

The proposed method avoids using a high frequency control clock signal, significantly saving power, while also reducing the memory required during processing, as the information is carried by pulses rather than n -bit samples.

2.2.1. Level-crossing sampling. Figure 2 shows a typical LC ADC architecture. The analogue input is compared with an upper (Thr+) and lower (Thr-) threshold voltages, as shown

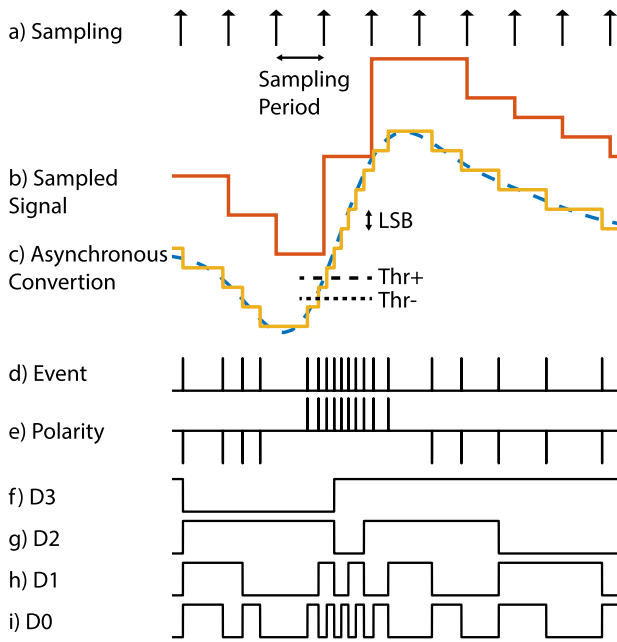


Figure 3. Schematic of level-crossing sampling. (a) Sampling index in the time domain for uniform sampling; (b) Uniform sampled signal; (c) The input analogue signal is compared with the Thr+ and Thr- thresholds. If either is crossed, both change by V_{LSB} to follow the input; (d) and (e) Asynchronous delta-modulation: the input signal is represented by a pair of bits stream: event indicates whenever an event occurs and Polarity indicates if it is an incremental ('1') or decremental ('-1' or '0') change; (f)–(i) Instantaneous 4-bit digital representation of the analogue input. The change can occur at any moment.

in figure 3(c), which are defined by an internal digital-to-analogue converter (DAC). This window sets the resolution of the ADC. The outputs of the two comparators drive control logic to produce two digital pulses indicating whenever an event occurs (event) and the polarity of the crossing (polarity), figures 3(d) and (e). These are then used to update the DAC by triggering a counter.

Instead of producing an n -bit word for each sample, the output of this ADC can be the pulse train composed by the Event and Polarity bits. Meanwhile, the real-time digital equivalent of the input is stored in the counter, figures 3(f) to (i). Compared to uniform sampling, the LC sampling in figure 3 could track the input signal (blue dash) much closer while having the same resolution, due to depending only on signal activity and not a fixed sampling rate.

As LC ADCs operate completely asynchronously, without a clock, no aliasing is present. However, as level-crossings can occur at any moment, the sampling rate cannot be specified by design. The resolution, V_{LSB} , on the other hand, is defined by:

$$V_{LSB} = \frac{V_{FS}}{2^N} \quad (1)$$

where V_{FS} is the maximum input range and N is the number of bits used for amplitude quantisation.

3. Materials and methods

In this study we propose a generic spike sorting method based on LC sampling. This method consists of three phases as illustrated in figure 4:

- A pre-recording phase: LC samples are continuously streamed out without any data processing (i.e. raw data). The spike waveforms are reconstructed in a computer.
- A training phase: reconstructed waveforms are processed by a computationally-demanding sorting algorithm to cluster units and extract accurate features, e.g. templates. These features are then sent back to the on-node processor.
- Online, real-time spike sorting phase: features of LC sampled data are extracted, and classified on-node using a distance metric computed between the current spike and stored template features.

We firstly analyse the performance of LC sampling on neural extracellular recording datasets, then detail the proposed algorithm including features and clustering methods. The presented work was implemented in *Mathworks* MATLAB v8.5 (R2015a), and the hardware implementation validated via *TerasIC*'s DE0 FPGA platform.

3.1. Test datasets

Initially we used 16 synthetic extracellular recording datasets, provided by [15]. Synthetic datasets were used as the ground truth is already known *a priori*, i.e. the time and class of each spike, used to evaluate the different sorting methods. The datasets are divided in 4 groups and labelled as either 'Easy' or 'Difficult' given the similarity between their spikes waveforms. Each group includes the same combination of source neurons but with 4 different noise levels: 0.05, 0.1, 0.15 and 0.2 (the peaks of spikes are normalised to 1). The noise levels are computed using equation (2).

$$Noise_{level} = median \left\{ \frac{|signal|}{0.6745} \right\}. \quad (2)$$

In vivo data obtained from CRCNS [14, 44], was also used to validate our results. This dataset is based on data recorded from a rat hippocampus and labelled as HC1.

All datasets were filtered using a 4th order Elliptic band-pass filter [17], to emulate the effect of the analogue front-end filtering [45]. The low- and high-pass cutoff frequencies were set to 4 kHz and 200 Hz respectively.

The datasets were upsampled to 10 MHz using linear interpolation to simulate continuous-time analogue signals. As the interpolation frequency is much higher than the low-pass cutoff frequency of the used filter, the spike distortion and in-band noise introduced by interpolation are both negligible. It should also be noted that this upsampling procedure is only required to conduct simulations on MATLAB using the same pre-recorded uniformly sampled datasets. On hardware, the LC ADC can directly output a continuous-time bitstream as described in the previous section.

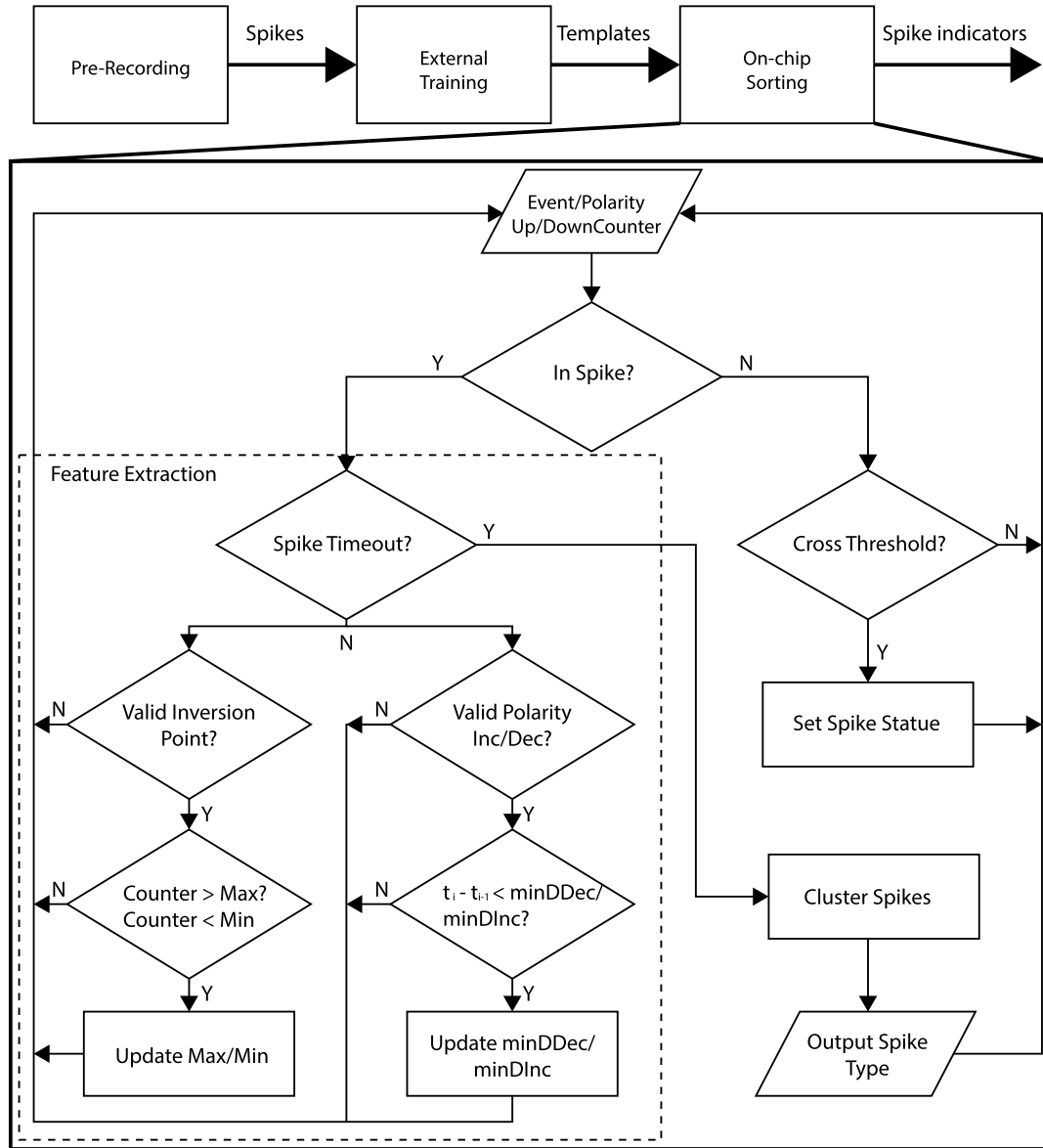


Figure 4. Flow chart describing the proposed spike sorting process.

3.2. LC ADC model

The LC ADC has been implemented in MATLAB. The difference between consecutive thresholds, V_{LSB} , was found for each dataset as:

$$V_{LSB} = \frac{\max(data) - \min(data)}{2^N}. \quad (3)$$

With N the number of bits used for amplitude quantisation.

3.3. Spike detection

Amplitude threshold crossing was used for spike detection. The detection threshold coefficient was first empirically set to 0.25 for all datasets, based on [15], then rounded to coincide with the closest LC ADC threshold in each simulation. Overlapping spikes were not included.

3.4. Event-driven features—EDF

For each spike, 4 event-driven features were extracted using the Event and Polarity outputs of the LC ADC. These are the peaks and the minimum inverse derivatives. It should be noted that possible event-driven features are not exhausted by those presented here, however, the examples mentioned in [22] require a dedicated LC ADC with a relatively complicated structure. Hence, the features in this work were selected for being considered the simplest example for further hardware implementation.

3.4.1. Peak values. The *max* and *min* features correspond to the values of the positive and negative peaks respectively.

Conventionally, to find these, every sample needs to be compared with the current values. With LC sampling, however, we can ‘economically’ find local extrema by tracking the patterns of the event and polarity pulse trains. If a downwards

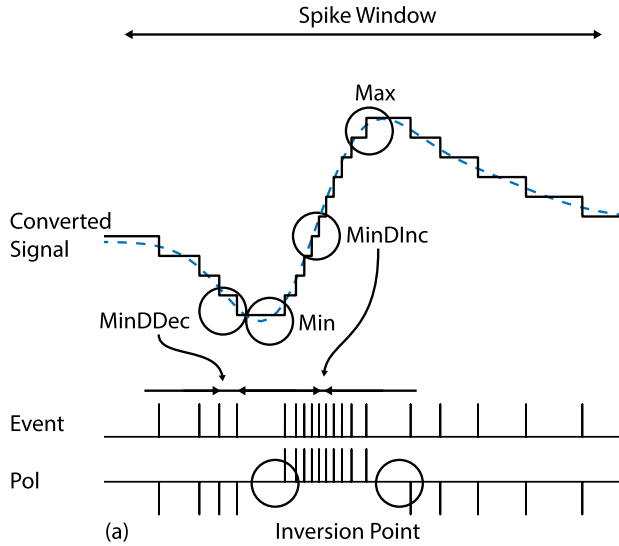


Figure 5. Spike and corresponding EDF. The time between consecutive level-crossings with the same polarity is quantised into $2^N \Delta t$ -wide intervals, with N the number of bits used to store the inverse derivative features: *minDInc* and *minDDec*.

level-crossing follows an upwards one, we have a local maximum; if an upwards level-crossing follows a downwards one, we have a local minimum, figure 5. Thus, we can reduce the number of comparisons for extrema finding to the number of local extrema.

3.4.2. Inverse derivatives. In synchronous systems, the derivative is found as the difference between consecutive samples. With LC sampling, however, the time between samples is not constant. But on the other hand, the difference in value is always 1 LSB. Thus, an inverse derivative can be computed through equation (4), which corresponds to measuring the time between consecutive level-crossings, figure 5.

$$\frac{\Delta t_i}{\Delta x} = \frac{t_i - t_{i-1}}{x_i - x_{i-1}} = t_i - t_{i-1}. \quad (4)$$

A fast input change will give rise to a dense pulse train, with short time intervals between events, while a smooth or flat input will generate few or even no pulses. Separate minimum inverse derivatives can be computed for the rising (incremental) and falling (decremental) slopes of spikes, resulting in two different features: *minDInc* and *minDDec*.

Only the event and polarity bits need to be handled to extract these features. Time-to-digital converters with inverter based delay lines can be used as in the FPGA implementation [46, 47]. As we only store the minimum inverse derivatives of each spike, the resolution and number of comparisons done can be greatly reduced, with larger time intervals between level-crossings being discarded as an overflow.

3.5. Noise considerations

With noise presented to the system, the features mentioned above will undoubtedly be influenced and impact the

classification sensitivity. As shown in figure 6, to the original noiseless spikes we superimpose:

- Overlapping neural spikes with varying amplitudes and shapes;
- Thermal noise presented before the recording circuits, which will be filtered;
- Electronic noise in the recording circuitry, which is predominantly thermal noise and wideband.

The resulting spike shapes will subsequently be distorted depending on the magnitude of the interfering signal (noise), however, the pulse trains generated for spike events and polarities are extended by summing these pulse trains together. This results in either the polarity or event pulses being shifted or reduced during this process due to quantisation. These introduce a pattern noise that results in a change in feature value.

We have therefore introduced bitstream pattern noise filtering to reduce noise, as shown in figure 6(c). This implements a 3-bit polarity buffer that stores the last three polarity values, and only triggers an inversion if there is a majority change. Figure 6(d) illustrates the noise filtering scheme that correctly identifies positive and negative inversion points using bitstream majority patterns, before extracting the derivative features between every valid consecutive level-crossing events.

The buffer size can be further extended to achieve a higher noise filtering performance depending on noise power. For instance, in a real implementation, a 5-bit buffer with the corresponding pattern recognition method could be used to balance resources and performance.

It should be noted that if there exists a large spike overlap or in-band electrode noise, it is particularly challenging to isolate the original spikes from the interference without using computationally-demanding methods such as averaging spike waveforms across a recording array [37] and calculating the correlations [36]. These methods, however, are not feasible on a single chip solution for on-site spike sorting.

3.6. Spike sorting

Once the features have been extracted from the current spike, the L1-norm distance between these and the values in the template memory is computed. The offline training stage, i.e. generating spike templates using an external clustering algorithm, is beyond the scope of this study. For our work, we used the ground truth of the simulated datasets to generate template features, whereas the real *in vivo* data is processed using the Wave_Clus software [15]. All features required to perform online classification are provided by the mean templates.

Each spike is then classified to the closest centroid based on the L1-norm distance, equation (5), whose computation is exempt from multiplications, allowing to reduce hardware resources.

$$Class_j = \min_i ||Spike_j - Centroid_i||_1 \quad (5)$$

$i = 1, 2, 3; \quad j = 1, \dots, N_{spikes}$

i indicates the template centroid and j the spike being classified.

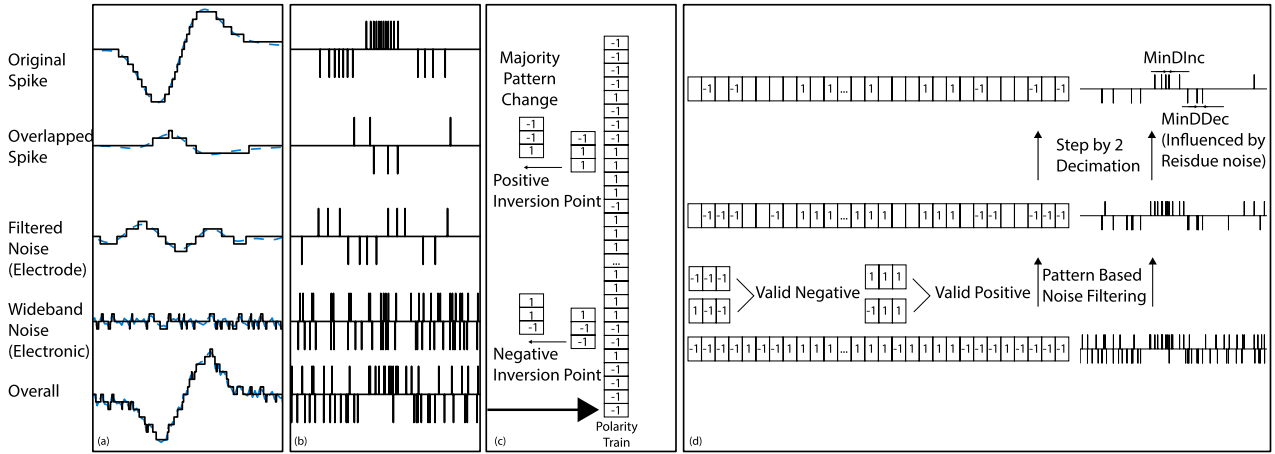


Figure 6. Event-based conversion of a neural signal which contains both biological and electronic noise. Shown are: (a) Original (blue dashed) and converted (solid) waveforms; (b) Polarity pulse trains indicating events for the different signal and noise components; (c) Majority pattern based inversion point locator; and (d) Bitstream-based noise filtering.

The quality of the selected feature space can be evaluated by calculating the similarity between the mean of one cluster to each of the other clusters [35]. The weighted average across all clusters then provides a measure of the overall quality of the selected feature space:

$$q = \sum_i \frac{w_i^2 N(\mu_i | \mu_i, \sigma_i)}{\sum_j w_j N(\mu_i | \mu_j, \sigma_j)} \quad (6)$$

where w_i is the weight of the number of spikes of type i out of the total amount of spikes, $N(\mu_i | \mu_j, \sigma_j)$ is the posterior probability for assigning mean of cluster i into cluster j . A value close to 1 corresponds to a high feature separation.

The sorting performance is evaluated by the accuracy (Acc) and the false positive rate (FPR), averaged between the 3 spike classes.

3.7. Reference methods

Benchmarking the proposed system against the reference methods required uniformly sampling the datasets at various frequencies plus quantising the amplitude with the same resolutions as those used with the LC ADC.

To compare different feature extraction methods, the discrete derivatives (DD) [13, 25], PCA [24], and discrete wavelet transform [15], are used.

Moreover, to benchmark the sorting results, template matching [13], PCA using the first 10 principal components (PCA₁₀), and the first and second derivative extrema (FSDE) [25], are used with a similar training process. In addition to these, US_Eq, a set of features similar to the EDF composed by two amplitude extrema and two derivative extrema, is extracted with a US ADC. These methods are chosen based on their hardware feasibility. Peak alignment was performed before extraction or clustering in PCA, wavelet and TM methods.

3.8. Hardware implementation

The proposed system has been implemented on an FPGA to demonstrate the feasibility of the event-driven approach for spike sorting. *TerasIC's* DE0 development board with *Altera's*

Cyclone III FPGA was used as the implementation platform. Pre-converted data was stored as the event and polarity bitstream in the flash memory, with a 0.1 μ s time resolution.

The system architecture of the FPGA implementation is shown in figure 7. This includes a core sorting engine and time-to-digital converters. The memory interface and user control blocks are not shown. The inputs to the system are the Pol (polarity) and event signals that represent the CT converted signal. A counter-based accumulator converts the bitstream to a digital representation of the recorded signal. A spike detection block then allocates a spike window based on absolute threshold crossing and a spike duration timer (set to 1.4ms). A pattern based peak/valley finder then determines and updates the extrema values. The time between two successive increments (or decrements) is quantised using the time-to-digital converters, driven by a 10 MHz clock, which is also the memory access clock.

Once the spike window ends, both the extrema and derivative EDFs are fed to the L1-norm classification block, which computes the distance between the current spike and 3 stored templates. By comparison, the system outputs the spike class (i.e. with the minimum distance) and signals a ready flag.

4. Results

4.1. Event-driven sampling

4.1.1. Data throughput versus ADC resolution. Figure 8 shows the data throughput produced by US and LC ADCs for different noise levels. It can be observed that the data throughput of LC ADCs displays an approximately exponential behaviour with increasing resolution, while for US ADCs it is linear. At lower resolutions, considerably lower data rates are achieved with LC ADCs, while at resolutions higher than 7 bits, LC ADCs produced more data due to the excessive amount of events triggered on the slopes of spikes.

It can also be found that increasing the noise levels increases the number of events, with noise level 0.2 producing approximately $2.5 \times$ higher data rates than noise level 0.05. These results suggest that the benefit of LC sampling

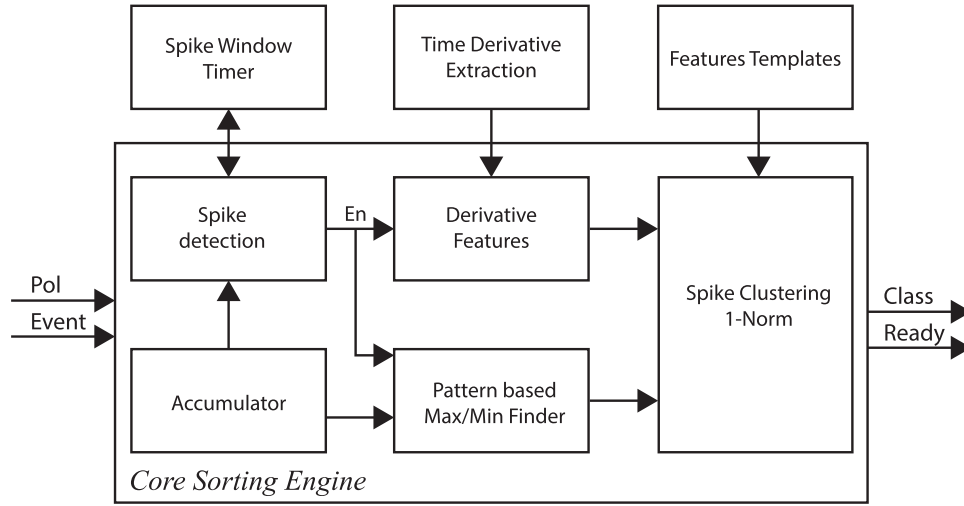


Figure 7. System architecture of the FPGA implementation, including the core sorting engine, spike timer and template storage.

in terms of data throughput can be compromised with higher resolutions, higher spike rate and noise levels. Therefore, it is worthwhile to compare the quality of conversion with respect to the events and data throughput as discussed in the following section.

The average number of events for each spike is summarised in table 1. It can be found that with 7-bit resolution, an average 84.4 pulses are required to represent the spike. With increased resolution, the number of detected inversion points, used for extrema extraction, also increases due to the presence of noise.

4.1.2. Data conversion quality versus data rate. To examine the conversion quality, the data sampled with US and LC sampling is reconstructed using zero-order interpolation, at a consistent higher sampling frequency, then compared with the original spikes by computing the error in equation (7) and averaging over all spikes.

$$E = \sum_{n=1}^{N_{\text{samples}}} |\text{spike}_{\text{rec}}(n) - \text{spike}(n)|. \quad (7)$$

Figure 9 shows the normalised error versus sampling rate. For LC sampling, the sampling rate corresponds to the average over all datasets. It can be seen that the conversion accuracy increases with the total data output, however, similarly to [17, 45, 48], a resolution increase from 6 to 8 bits does not result in a significant improvement.

LC sampling achieves lower conversion errors than US when the same data rate is obtained, which becomes more evident at lower data rates. This results from LC sampling preserving the main characteristics of spikes, in terms of magnitude and peaks, with fewer total samples, as shown in figure 1.

4.2. Event-driven spike sorting

Based on the analysis and discussion in the previous subsections, a resolution of 6 bits is set for the behavioural model of the LC ADC. Δt is set such that a 6-bit dynamic range for the EDF can be achieved.

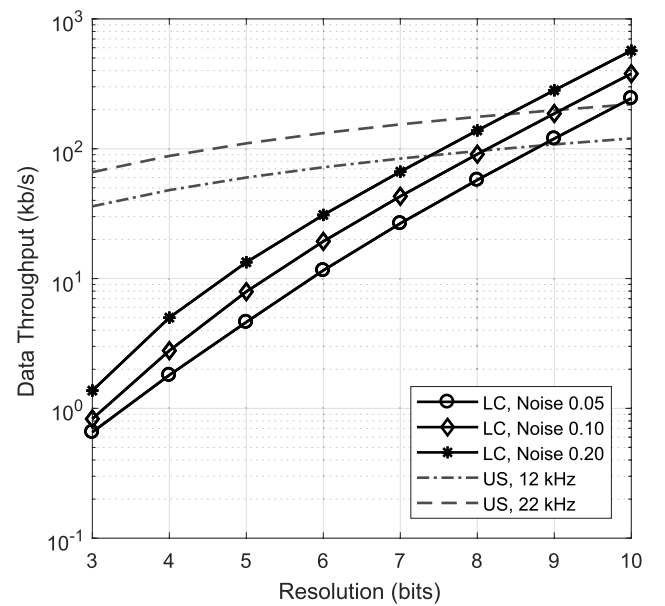


Figure 8. Data throughput versus ADC resolution. For US, each sample outputs N bits, while LC ADCs output only 2 bits for each level-crossing. For LC sampling, the data throughput shown was averaged over the datasets with each noise level.

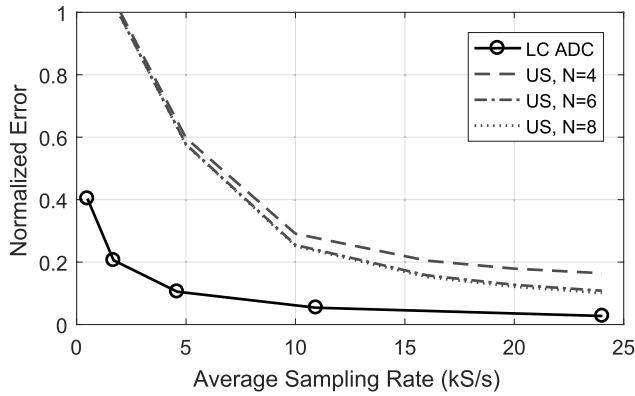
From left to right, figure 10 shows the extracted spikes compared to average spike profiles, the EDF features (minDInc and minDDec with a 6-bit resolution), DD features (maximum and minimum 1st derivatives [25] for both 6 and 12-bit resolutions), PCA (largest scores using a 12-bit resolution), and wavelet features (largest coefficients using a 12-bit resolution). The ‘Difficult1’ datasets with noise levels 0.05 to 0.2 are used for each row. The feature extraction quality is labelled in each figure accordingly.

It should be noted that all methods are sensitive to spike overlap, while PCA and the wavelet method require peak alignment [13]. Waveform adjustment, i.e. overlap removal and peak alignment, has been performed before feature extraction.

Although clusters begin to overlap with higher noise levels, the majority of spikes is still correctly identified. Compared

Table 1. Average number over all datasets, and corresponding standard deviation, of events and inversion points per spike (1 ms window), using a LC ADC.

N	3	4	5	6	7
Events	4.0	9.2	19.8	41.2	84.4
Std	0.7	1.2	2.2	4.2	8.3
Inv. points	1.1	1.5	1.8	2.0	2.2
Std	0.3	0.3	0.4	0.4	0.4

**Figure 9.** Comparison between the conversion accuracies of US and LC sampling versus data rate. For LC sampling, N increases from 3 to 7, while for US, different curves are shown for different resolutions.

to DD, PCA and the wavelet method, the feature space of the proposed EDF achieved a better quality factor across all noise levels. Moreover, the extracted EDF are discrete integers, which is more evident in low noise signals. This dramatically reduces the clustering computational complexity in hardware.

A similar performance can be found in figure 11, when pre-recorded *in vivo* data is used [14, 44].

Figure 11 also shows the clustering results using both Wave_Clus and EDF based sorting. The majority of spikes are correctly classified, however some error still exists due to the spread in the feature space and accuracy of the distance calculation.

Table 2 summarises the EDF sorting accuracy under different resolutions and noise levels with the synthetic datasets. With the proposed method and pattern based filtering, the average sorting accuracy across all datasets increased from 72% to 95% with a resolution increase from 3 to 7 bits.

4.2.1. Hardware cost. The hardware cost is a key constraint in designing large-scale on-node spike sorting systems. Therefore, the sorting accuracy is benchmarked with respect to two key specifications: computational complexity and memory utilisation. Feature extraction complexity corresponds to the number of operations required per spike to extract the features, table 3. A single operation is defined as a 1-bit addition, subtraction or comparison, while a multiplication corresponds to 10 operations, as in [25].

The reference methods rely on US, hence the feature extraction complexity depends only on the ADC resolution, N , and the number of samples taken from each spike, N_{samples} , which in turn varies with sampling rate. For the EDF, the number of

events and inversion points varies between spikes. Hence, we use the average values found over all datasets, as in table 1, to compute the complexity. The noise pattern filtering method proposed prior to finding the extrema will reduce the amount of events and inversion points. However, it will increase the complexity even if the algorithm is based on combinational logic rather than subtractions. Therefore, the worst case scenario for the number of events, i.e. obtained without filtering, is used to simplify the calculations.

Neither the cost of aligning spikes to their maximum, for TM and PCA, nor of quantising time, for the inverse derivative features, has been included in the reported complexity. With PCA, the cost of finding the principal directions is also not included, as they are only computed once for each dataset.

The memory utilisation depends on the feature size and is directly related to the resolution in bits, table 3. It should be noted that while PCA uses a smaller number of features than TM, the whole spike still needs to be stored on-node prior to feature extraction. For the EDF, FSDE and US_Eq, features can be extracted as the spike is being recorded in real-time, requiring less memory.

The average sorting accuracy versus complexity and required memory (estimated by the feature dimensionality) for the EDF, FSDE, US_Eq, TM and PCA using 10 principal components are shown in figures 12 and 13. For the EDF, the same parameters as those in table 2 are used. For US_Eq, TM and PCA₁₀, N varies from 3 to 7, as further increasing the resolution did not improve sorting results. For the FSDE, however, the resolution goes up to 9. The sampling rate varies between 10 kHz and 24 kHz.

The FSDE and US_Eq methods use features which aim at describing similar characteristics to those of the EDF. Nonetheless, the latter can outperform both these methods. With identical hardware expense, the EDF obtain better results than all reference methods.

Overall, only TM and PCA₁₀ can outperform the EDF on certain datasets, however by only around 1%. To achieve a similar accuracy to the EDF with $N = 7$, TM (with $N = 5$ and a sampling rate of 16 kHz), already has approximately a $4 \times$ larger feature size, while PCA₁₀ (with $N = 3$ and a sampling rate of 14 kHz) has a feature extraction complexity approximately $14 \times$ higher.

4.3. Measured FPGA implementation results

The core sorting engine synthesised on the FPGA occupies 377 logic cells and 73 registers for a 6-bit sorting resolution, i.e. with both input and feature templates encoded using a 6-bit digital representation. It should be noted that the spike window timer and time feature extraction are implemented using a counter driven by a 10 MHz clock signal. These have been excluded from the sorting engine.

A total of 14 segments from different data sources, lasting 1 s each, are used as test input. The implemented system was characterised using Saleae's logic pro16 digital oscilloscope, with an example of a measured output shown in figure 14. The reconstructed waveform based on digital outputs D5–D0 is

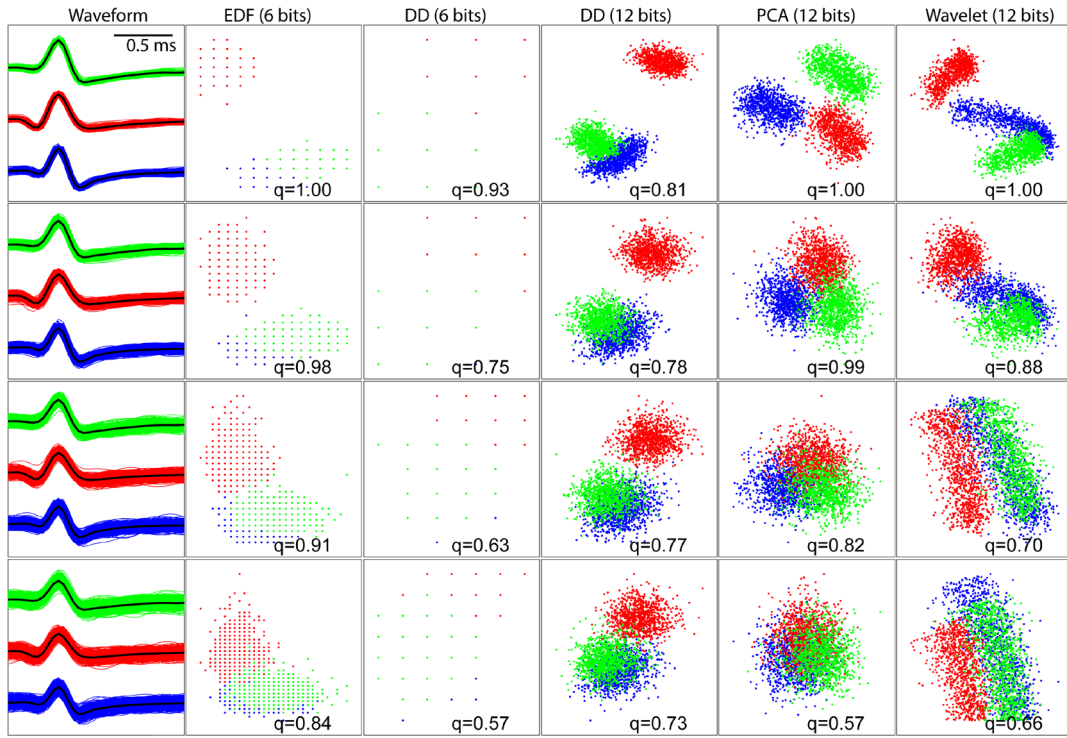


Figure 10. From top to bottom: extracted spikes and corresponding features from the ‘Difficult1’ datasets with noise levels from 0.05 to 0.20; From left to right: extracted spikes and average waveforms, event-based features, discrete derivative features, maximum scores from PCA, and wavelet features. All the sub figures are colour coded with respect to ground truth in the simulated data.

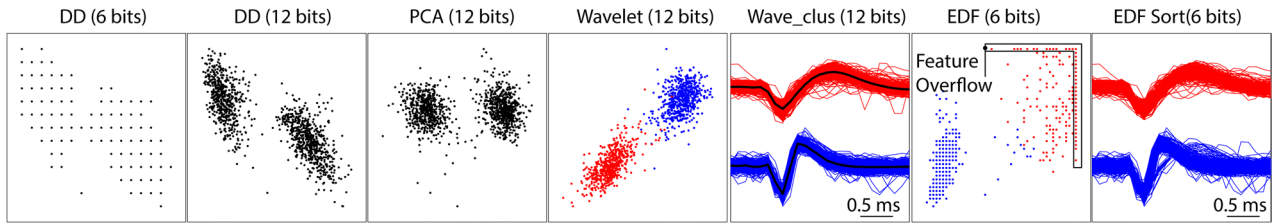


Figure 11. Feature extraction and sorting results for *in vivo* data from CRCNS.org [14, 44]. From left to right: discrete derivative features (resolutions of 6 and 12-bits), PCA (12-bit), wavelet features (12-bits), sorted spikes based on Wave_Clus, EDF, and sorted spikes based on EDF and L1-norm clustering.

Table 2. Sorting accuracy (Acc) and FPR (false positive rate) per dataset, for EDF features with different resolution (bits).

		3		4		5		6		7	
Bits		Acc	FPR	Acc	FPR	Acc	FPR	Acc	FPR	Acc	FPR
Easy1	0.05	0.84	0.12	1.00	0.00	1.00	0.00	1.00	0.00	1.00	0.00
	0.1	0.81	0.14	1.00	0.00	1.00	0.00	1.00	0.00	1.00	0.00
	0.15	0.80	0.15	1.00	0.00	1.00	0.00	1.00	0.00	1.00	0.00
	0.2	0.73	0.20	0.99	0.00	1.00	0.00	1.00	0.00	1.00	0.00
Easy2	0.05	0.97	0.02	0.93	0.05	0.96	0.03	0.99	0.00	0.99	0.00
	0.1	0.79	0.16	0.79	0.16	0.89	0.08	0.93	0.06	0.93	0.06
	0.15	0.69	0.23	0.66	0.25	0.83	0.13	0.87	0.10	0.87	0.10
	0.2	0.63	0.28	0.71	0.22	0.79	0.16	0.83	0.13	0.83	0.13
Difficult1	0.05	0.77	0.18	0.88	0.09	0.99	0.01	1.00	0.00	1.00	0.00
	0.1	0.73	0.20	0.85	0.11	0.96	0.03	0.98	0.01	0.99	0.01
	0.15	0.66	0.26	0.87	0.10	0.92	0.06	0.94	0.04	0.94	0.04
	0.2	0.66	0.26	0.83	0.13	0.88	0.09	0.90	0.07	0.91	0.07
Difficult2	0.05	0.58	0.32	0.96	0.03	0.99	0.01	1.00	0.00	1.00	0.00
	0.1	0.61	0.30	0.89	0.08	0.93	0.05	0.94	0.04	0.95	0.04
	0.15	0.59	0.30	0.86	0.10	0.87	0.09	0.89	0.08	0.89	0.08
	0.2	0.62	0.28	0.82	0.14	0.83	0.13	0.83	0.13	0.84	0.12
Mean		0.72	0.21	0.88	0.09	0.93	0.05	0.94	0.04	0.95	0.04

Table 3. Feature extraction complexity per spike and feature dimensionality for each sorting method used. N is the feature resolution and N_{samples} the number of samples taken from each spike. The $|\cdot|$ operand indicates the average number of events and inversion points per spike, computed over all datasets.

Method	Feature extraction Complexity	Feature Dimensionality
EDF	$N \times (events + inv.points - 1) + events $	$4 \times N$
TM	—	$N_{\text{samples}} \times N$
PCA _n	$n \times N \times (12 \times N_{\text{samples}} - 1)$	$n \times N$
FSDE	$N \times (6 \times N_{\text{samples}} - 9)$	$4 \times N$
US_Eq	$N \times (5 \times N_{\text{samples}} - 3)$	$4 \times N$

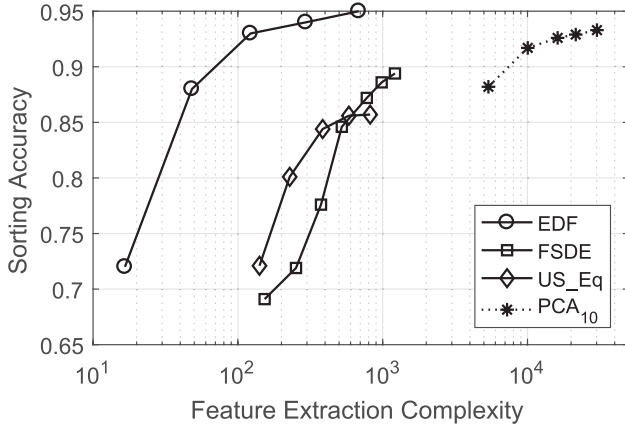


Figure 12. Average sorting accuracy, over all 16 datasets, versus feature extraction complexity for the EDF, FSDE, US_Eq and PCA using 10 principal components. The dotted line distinguishes methods using a larger number of features than the EDF.

shown for reference purposes only, as it is not necessary in a final implementation. It can be observed that the counter based data converter recovers the spike shape from the asynchronous events and polarity signals, with minimal distortion compared to uniform sampling. When a spike is detected, a fixed time window is allocated to enable feature extraction. Once the timer exceeds the spike window, the EDF features are latched and the distance between the current spike and the three templates are calculated and compared. Finally, a ready signal is triggered and a flag indicating the spike type is presented.

Compared to the MATLAB implementation results with the same algorithm and datasets ('easy1' and 'difficult1'), the detection scheme on the FPGA platform achieved 99.94% (695/696) of true positive rate; the sorting true positive rate (categorizing spikes with the same type) was 96.4% (669/696) and the false positive rate was 0.94% (averaged from the three types). The observed discrepancy is mainly due to a quantisation error.

5. Discussion

5.1. Raw recording

It has been demonstrated that data throughput can be dramatically reduced with LC based sampling. However, increased resolution and noise levels can introduce redundant crossing

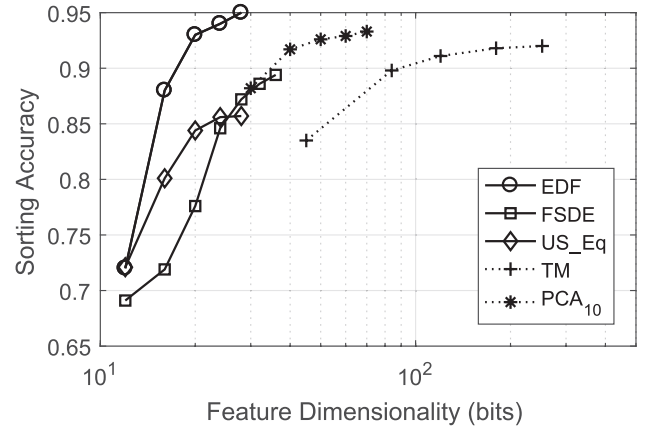


Figure 13. Average sorting accuracy, over all 16 datasets, versus feature size for the EDF, FSDE, US_Eq, TM and PCA using 10 principal components. The dotted line distinguishes methods using a larger number of features than the EDF.

events and increase the data rate. For a Gaussian noise signal with bandwidth f and power φ , the average number of samples produced per second is given by equation (8), [49].

$$\bar{N}_s = \sqrt{\frac{8\pi}{3}} \frac{\sqrt{\varphi}}{V_{\text{LSB}}} f. \quad (8)$$

For this reason, low noise analogue front-end filters will be needed to filter both electronic and electrode thermal noise.

Moreover, higher spike rates will also increase the data throughput. The synthetic datasets used have spike rates of 50–60 per second, which is more than has been reported *in vivo* [44]. These observations indicate that under extreme conditions, i.e. high noise and high spike rates, this approach could lead to higher output data streams. However, the fidelity of the reconstructed spikes is still higher for LC sampling even with considerably lower data bandwidth. Secondly, noise suppression techniques, as previously mentioned, can be used to minimise the unnecessary data output, for example, induced by thermal noise. Last but not least, since the desired output is the sorted spike information, the data rate in the raw recording stage will only exist for a short period and can be reduced by sequentially selecting sorting channels.

It should also be noted that tracking an input with LC ADCs only requires changing the output by 1 LSB between events, as shown in figure 3, resulting in reduced switching and comparison power in the ADC. Moreover, adaptive resolution LC ADCs [50], could be used to further reduce the data conversion rate, and consequently power consumption, even in the presence of high noise.

Nonetheless, the latency of LC sampling induced by the delay of the comparators, control logic, counter and DAC, must be minimized such that the threshold window defined by the DAC can accurately track the input signal with minimum distortion. A large latency could distort the features by shifting the crossing events. Assuming the delay introduced by updating the loop after each level-crossing is Δ , the maximum input gradient to be sampled without introducing distortion is:

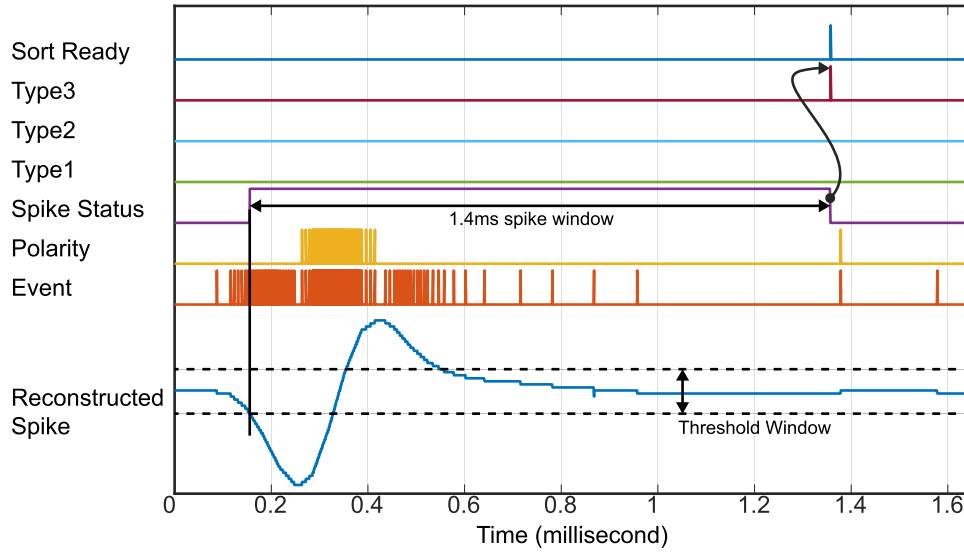


Figure 14. Measured FPGA output showing (from bottom to top): reconstructed waveform from the 6-bit digital output; triggered events; polarity (1 for a positive level-crossing and 0 for a negative one); spike status indicating when the feature extraction is on; three spike type indicators; and sort ready signal.

$$\left| \frac{dV_i}{dt} \right| < \frac{V_{LSB}}{\Delta}.$$

(9)

5.2. EDF features and clustering

Although the results demonstrate that EDF features achieve a high sorting accuracy, the best performance is still achieved with conventional features sampled with relatively high resolutions and sampling rates (e.g. PCA at 24kHz and with 12-bit). The selected features (extrema and minimum time derivatives) are susceptible to noise and the similarity of the derivatives of spikes, as shown in table 2. Most importantly, the noise induced feature distortion, specially by overlapped spikes, can significantly degrade the sorting results.

It should be noted, however, that only feature extraction and classification are done using the proposed method, while clustering is still performed offline. This requires a training phase using established (and computationally more demanding) methods, such as *k*-means or supermagnetic clustering. This division of spike sorting into two stages allows to strike a good trade-off between clustering accuracy and online spike sorting efficiency.

Additionally, event-based sampling can track the signal with less quantisation and aliasing noise, as shown in figure 3. This is because the level-crossing scheme provides timely and accurate comparison during the conversion, and a fixed sampling frequency is no longer needed. Therefore the extracted features can represent the spikes with higher fidelity compared to the same features extracted by uniform sampling with the same resolution, as shown in figure 10.

Finally, this new representation of spikes in the time and magnitude domain, through bit streams, can significantly change the way spikes are interpreted, and inspire new methods for processing sparse signal. For instance, bits stream based digital filters [23] can provide robust noise removal.

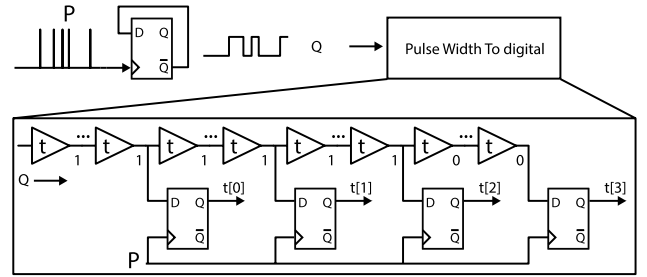


Figure 15. Illustration of a complete clock-less inverse time derivative feature extractor using delay elements. In this example, the feature resolution is 4 bits.

A similar methodology as in [31] can also be implemented with the proposed method, such that templates can be extracted without external assistance.

5.3. Hardware implementation

In the FPGA implementation, the time derivative features are extracted by using clock-driven counters for demonstration purposes. However, a complete clock-less time feature extractor has also been implemented, as shown in figure 15. The input pulse, *P*, is converted to pulse-width with different polarities, *Q*, and a pulse-width-to-digital converter is then used, which includes a delay line and output latches. The pulse-width *Q* propagates through the delay elements and is quantized by the delay unit of *t*. Essentially, the delay line takes a snapshot of the pulse-width (continuous 1 or 0 from left) and latches the thermal-coded time derivative results in the registers, whose amount is defined by the resolution of the features.

A total of 4096 delay elements provided 1.02 μ s of capture range. However, this is smaller than the minimum time derivative value when a low resolution is chosen. Moreover, it did not maintain consistent results due to jitter noise and occupied

a large area [46, 47]. This is due to FPGAs not being designed for asynchronous processing in general, while propagation delays, t , are minimized to achieve fast operation.

With a custom hardware implementation, such as an ASIC (application-specific integrated circuit), these delay elements can be realised using current starved inverters [47, 51], which allow a much larger and tunable propagation delay, t , to minimize power and area.

6. Conclusion

This work has proposed event-driven asynchronous data representation for spike-based processing. It has been demonstrated that this can be applied to achieve effective spike sorting with the prospect of efficient hardware implementation.

We conclude that: (1) level-crossing based data conversion can provide a more accurate representation of the recorded spike signals whilst also minimising the data rate (compared to uniform sampling with the same number of quantisation levels). This is because level crossing conversion has a significantly reduced quantisation noise; (2) the proposed EDF can achieve efficient (low complexity) and effective (high accuracy) sorting performance (feature extraction and classification—not clustering). We have demonstrated how peak values and minimum inverse derivatives can be used as sorting features together with L1-norm classification; and (3) these proposed methods can be efficiently implemented on-node for spike sorting, thus significantly reducing hardware cost.

Compared with the reference methods of FSDE, US_Eq, TM and PCA₁₀, applied to the same datasets, the proposed system achieved similar sorting performances while producing fewer sampled data throughput and requiring less hardware resources.

Acknowledgments

This research was funded by Engineering and Physical Sciences Research Council (EPSRC) EP/I000569/1 and EP/K015060/1. We would like to thank Dr Ian Williams for useful discussion.

Appendix. Acronyms

ADC	Analogue-to-digital converter
DAC	Digital-to-analogue converter
EDF	Event-driven features
FSDE	First and second derivative extrema
FPGA	Field-programmable gate array
LC	Level-crossing
LSB	Least significant bit
PCA	Principal component analysis
DD	Discrete derivative
TM	Template matching
US	Uniform sampling
US_Eq	Uniform sampling equivalent (features)

ORCID iDs

Yan Liu  <https://orcid.org/0000-0001-5616-9428>

References

- [1] Liu Y, Luan S, Williams I, Rapeaux A and Constandinou T G 2017 A 64-channel versatile neural recording soc with activity-dependent data throughput *IEEE Trans. Biomed. Circuits Syst.* **11** 1344–55
- [2] Gerstner W and Kistler W M 2002 *Spiking Neuron Models. Single Neurons, Populations, Plasticity* (Cambridge: Cambridge University Press)
- [3] Todorova S, Sadtler P, Batista A, Chase S and Ventura V 2014 To sort or not to sort: the impact of spike-sorting on neural decoding performance *J. Neural Eng.* **11** 056005
- [4] Wodlinger B, Downey J E, Tyler-Kabara E C, Schwartz A B, Boninger M L and Collinger J L 2014 Ten-dimensional anthropomorphic arm control in a human brain-machine interface: difficulties, solutions and limitations *J. Neural Eng.* **12** 016011
- [5] Truccolo W, Donoghue J A, Hochberg L R, Eskandar E N, Madsen J R, Anderson W S, Brown E N, Halgren E and Cash S S 2011 Single-neuron dynamics in human focal epilepsy *Nat. Neurosci.* **14** 635–41
- [6] Berényi A, Somogyvári Z, Nagy A J, Roux L, Long J D, Fujisawa S, Stark E, Leonardo A, Harris T D and Buzsáki G 2014 Large-scale, high-density (up to 512 channels) recording of local circuits in behaving animals *J. Neurophysiol.* **111** 1132–49
- [7] Luan S, Liu Y, Williams I and Constandinou T G 2016 An event-driven soc for neural recording *IEEE Biomedical Circuits and Systems Conf.* pp 404–7
- [8] Stevenson I H and Kording K P 2011 How advances in neural recording affect data analysis *Nat. Neurosci.* **14** 139–42
- [9] Yin M 2009 A multi-channel wireless implantable neural recording system *PhD Thesis* North Carolina State University
- [10] Pedreira C, Martinez J, Ison M J and Quiroga R Q 2012 How many neurons can we see with current spike sorting algorithms? *J. Neurosci. Methods* **211** 58–65
- [11] Wolf P D 2008 Thermal considerations for the design of an implanted cortical brain-machine interface (BMI) *Indwelling Neural Implants: Strategies for Contending with the In Vivo Environment* (Boca Raton, FL: CRC Press) pp 63–83
- [12] Abeles M and Goldstein M H 1977 Multispike train analysis *Proc. IEEE* **65** 762–73
- [13] Lewicki M S 1998 A review of methods for spike sorting: the detection and classification of neural action potentials *Netw.: Comput. Neural Syst.* **9** R53–78
- [14] Harris K D, Henze D A, Csicsvari J, Hirase H and Buzsáki G 2000 Accuracy of tetrode spike separation as determined by simultaneous intracellular and extracellular measurements *J. Neurophysiol.* **84** 401–14
- [15] Quiroga R Q, Nadasdy Z and Ben-Shaul Y 2004 Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering *Neural Comput.* **16** 1661–87
- [16] Gibson S, Judy J W and Markov D 2012 Spike sorting: the first step in decoding the brain: the first step in decoding the brain *IEEE Signal Process. Mag.* **29** 124–43
- [17] Navajas J, Barsakcioglu D Y, Eftekhari A, Jackson A, Constandinou T G and Quiroga R Q 2014 Minimum requirements for accurate and efficient real-time on-chip spike sorting *J. Neurosci. Methods* **230** 51–64
- [18] Karkare V, Gibson S and Markovic D 2011 A 130 μ m, 64-channel neural spike-sorting dsp chip *IEEE J. Solid-State Circuits* **46** 1214–22

- [19] Chen T, Ma T, Chen Y and Chen L 2012 Low power and high accuracy spike sorting microprocessor with on-line interpolation and re-alignment in 90nm cmos process *Annual Int. Conf. of the IEEE Engineering in Medicine and Biology Society* pp 4485–8
- [20] Karkare V, Gibson S and Markovi D 2013 A 75 μ W, 16-channel neural spike-sorting processor with unsupervised clustering *IEEE J. Solid-State Circuits* **48** 2230–8
- [21] Liu X, Zhang M, Xiong T, Richardson A G, Lucas T H, Chin P S, Etienne-Cummings R, Tran T D and der Spiegel J V 2016 A fully integrated wireless compressed sensing neural signal acquisition system for chronic recording and brain machine interface *IEEE Trans. Biomed. Circuits Syst.* **10** 874–83
- [22] Liu Y, Pereira J L and Constandinou T G 2016 Clockless continuous-time neural spike sorting: method, implementation and evaluation *IEEE Int. Symp. on Circuits and Systems* pp 538–41
- [23] Tsvividis Y 2010 Event-driven data acquisition and digital signal processing; a tutorial *IEEE Trans. Circuits Syst. II: Express Briefs* **57** 577–81
- [24] Gibson S, Judy J W and Markovic D 2010 Technology-aware algorithm design for neural spike detection, feature extraction and dimensionality reduction *IEEE Trans. Neural Syst. Rehabil. Eng.* **18** 469–78
- [25] Paraskevopoulou S E, Barsakcioglu D Y, Saberi M R, Eftekhar A and Constandinou T G 2013 Feature extraction using first and second derivative extrema (FSDE) for real-time and hardware-efficient spike sorting *J. Neurosci. Methods* **215** 29–37
- [26] Williams I, Luan S, Jackson A and Constandinou T G 2015 A scalable 32-channel neural recording and real-time FPGA based spike sorting system *IEEE Biomedical Circuits and Systems Conf.* pp 188–91
- [27] Koutsos E, Paraskevopoulou S E and Constandinou T G 2013 A 1.5 μ W NEO-based spike detector with adaptive-threshold for calibration-free multichannel neural interfaces *IEEE Int. Symp. on Circuits and Systems* pp 1922–5
- [28] Obeid I and Wolf P D 2004 Evaluation of spike-detection algorithms for a brain-machine interface application *IEEE Trans. Biomed. Eng.* **51** 905–11
- [29] Bonfanti A, Borghi T, Gusmeroli R, Zambra G, Oliyink A, Fadiga L, Spinelli A S and Baranauskas G 2008 A lowpower integrated circuit for analog spike detection and sorting in neural prosthesis systems *IEEE Biomedical Circuits and Systems Conf.* pp 257–60
- [30] Kamboh A M and Mason A J 2013 Computationally efficient neural feature extraction for spike sorting in implantable high-density recording systems *IEEE Trans. Neural Syst. Rehabil. Eng.* **21**
- [31] Zamani M and Demosthenous A 2014 Feature extraction using extrema sampling of discrete derivatives for spike sorting in implantable upper-limb neural prostheses *IEEE Trans. Neural Syst. Rehabil. Eng.* **22** 716–26
- [32] Paraskevopoulou S E, Wu D, Eftekhar A and Constandinou T G 2014 Hierarchical adaptive means (ham) clustering for hardware-efficient, unsupervised and real-time spike sorting *J. Neurosci. Methods* **235** 145–56
- [33] Zviaginsev A, Perelman Y and Ginosar R 2005 Low-power architectures for spike sorting *Conf. Proc. 2nd Int. IEEE EMBS Conf. on Neural Engineering* pp 162–5
- [34] Xu R and Wunsch D C 2010 Clustering algorithms in biomedical research: a review *IEEE Rev. Biomed. Eng.* **3** 120–54
- [35] Rossant C et al 2016 Spike sorting for large, dense electrode arrays *Nat. Neurosci.* **19** 634–41
- [36] Pillow J W, Shlens J, Chichilnisky E J and Simoncelli E P 2013 A model-based spike sorting algorithm for removing correlation artifacts in multi-neuron recordings *PLoS One* **8** 1–14
- [37] Paralakar K J, Rao C R and Clement R S 2009 New approaches to eliminating common-noise artifacts in recordings from intracortical microelectrode arrays: inter-electrode correlation and virtual referencing *J. Neurosci. Methods* **181** 27–35
- [38] Yang Y, Boling C and Mason A 2014 Power-area efficient VLSI implementation of decision tree based spike classification for neural recording implants *IEEE Biomedical Circuits and Systems Conf. Proc.* pp 380–3
- [39] Barsakcioglu D Y and Constandinou T G 2016 A 32-channel MCU-based feature extraction and classification for scalable on-node spike sorting *IEEE Int. Symp. on Circuits and Systems* pp 1310–3
- [40] Li Y, Zhao D and Serdijn W A 2013 A sub-microwatt asynchronous level-crossing adc for biomedical applications *IEEE Trans. Biomed. Circuits Syst.* **7** 149–57
- [41] Zhang X and Lian Y 2014 A 300 mv 220 nw event-driven adc with real-time qrs detection for wearable ecg sensors *IEEE Trans. Biomed. Circuits Syst.* **8** 834–43
- [42] Maslik M, Liu Y, Lande T and Constandinou T 2017 A charge-based ultra-low power continuous-time ADC for data driven neural spike processing *IEEE Int. Symp. on Circuits and Systems* pp 1–4
- [43] Guo N, Huang Y, Mai T, Patil S, Cao C, Seok M, Sethumadhavan S and Tsvividis Y 2016 Energy-efficient hybrid analog/digital approximate computation in continuous time *IEEE J. Solid-State Circuits* **51** 1514–24
- [44] Henze D A, Borhegyi Z, Csicsvari J, Mamiya A, Harris K D and Buzsáki G 2000 Intracellular features predicted by extracellular recordings in the hippocampus *in vivo* *J. Neurophysiol.* **84** 390–400
- [45] Barsakcioglu D Y, Liu Y, Bhunjun P, Navajas J, Eftekhar A, Jackson A, Quiroga R Q and Constandinou T G 2014 An analogue front-end model for developing neural spike sorting systems *IEEE Trans. Biomed. Circuits Syst.* **8** 216–27
- [46] Szplet R and Klepacki K 2010 An FPGA-integrated time-to-digital converter based on two-stage pulse shrinking *IEEE Trans. Instrum. Meas.* **59** 1663–70
- [47] Chen C-C, Chang W and Chen P 2004 A precise cyclic CMOS time-to-digital converter with low thermal sensitivity *IEEE Symp. Conf. Record Nuclear Science* vol 3 pp 1364–7
- [48] Barsakcioglu D Y, Eftekhar A and Constandinou T G 2013 Design optimisation of front-end neural interfaces for spike sorting systems *IEEE Int. Symp. on Circuits and Systems* pp 2501–4
- [49] Inose H, Aoki T and Watanabe K 1966 Asynchronous delta-modulation system *Electron. Lett.* **2** 95–6
- [50] Weltin-Wu C and Tsvividis Y 2013 An event-driven clockless level-crossing ADC with signal-dependent adaptive resolution *IEEE J. Solid-State Circuits* **48** 2180–90
- [51] Johnson M G and Hudson E L 1988 A variable delay line pll for cpu-coprocessor synchronization *IEEE J. Solid-State Circuits* **23** 1218–23