A Neuromorphic Brain Interface based on RRAM Crossbar Arrays for High Throughput Real-time Spike Sorting

Yuhan Shi, Akshay Ananthakrishnan, Sangheon Oh, Xin Liu, Gopabandhu Hota, Gert Cauwenberghs, Duygu Kuzum

***Abstract* — Real-time spike sorting and processing are crucial for closed-loop brain-machine interfaces and neural prosthetics. Recent developments in high-density multi-electrode arrays with hundreds of electrodes have enabled simultaneous recordings of spikes from a large number of neurons. However, the high channel count imposes stringent demands on real-time spike sorting hardware regarding data transmission bandwidth and computation complexity. Thus, it is necessary to develop a specialized real-time hardware that can sort neural spikes on the fly with high throughputs while consuming minimal power. Here, we present a real-time, low latency spike sorting processor that utilizes high-density CuO_x resistive crossbars to implement in-memory spike sorting in a massively parallel manner. We developed a fabrication process which is compatible with CMOS BEOL integration. We extensively characterized switching characteristics and statistical variations of the CuO_x memory devices. In order to implement spike sorting with crossbar arrays, we developed a template matching-based spike sorting algorithm that can be directly mapped onto RRAM crossbars. By using synthetic and *in vivo* recordings of extracellular spikes, we experimentally demonstrated energy efficient spike sorting with high accuracy. Our neuromorphic interface offers substantial improvements in area (~1000× less area), power (~200× less power), and latency (4.8μs latency for sorting 100 channels) for real-time spike sorting compared to other hardware implementations based on FPGAs and microcontrollers.**

***Index Terms*—RRAM, Memory, Non-volatile memory, Crossbar, Copper Oxide, Real-time, Spike Sorting, High Throughput, Template Matching.**

I. Introduction

Extracellular recordings of neuronal spikes using microelectrode arrays have been widely used in studying neural circuits involved in sensory [1], motor [2], and navigation [3] functions in the brain [4]. The recorded signals are a mix of activities from multiple neurons and a crucial processing step, called spike sorting, is required to separate the firing activities and assign the recorded spikes to individual neurons from the recordings. Spike sorting is an indispensable tool in neuroscience for studying neural circuits [5], connectivity, causality and decoding brain activities [6, 7]. It is also fundamental in decoding intentions from neural activity in brain-machine interfaces (BMIs) [8] and neural prosthetics [9]. Conventionally, spike sorting is performed offline by transmitting raw digitized signals recorded by neural electrodes to a nearby computer. However, the off-line processing approach becomes impractical for sorting neural recordings generated from advanced high-density microelectrode arrays (HDMEAs) that comprise hundreds or thousands of recording sites in a single probe, such as recently developed Neuropixels probe [10]. Transmitting vast amounts of neural recording data from HDMEAs to an off-line spike sorter leads to excessive power dissipation which poses a serious risk of damage for the surrounding tissues [11]. For example, a 100-channel microelectrode array with a 16-bit ADC operating at 30kHz sampling frequency generates 3MSamples/s and dissipates mW-level power to nearby tissues. More importantly, to enable the closed-loop BMIs for prosthetics with multiple degrees of freedom, hundreds of neurons distributed in multiple cortical areas need to be monitored in real-time with minimal delay [12]. An 8-hour recording experiment using a 100-channel microelectrode array would accumulate ~200GB of data [13], demanding at least a few hours to sort the recorded spikes off-line [14]. The high latency associated with spike sorting becomes a limiting factor for closed-loop applications requiring rapid feedback. These drawbacks highlight the need for developing compact, low-power and high throughput hardware that can be integrated with high density implantable microelectrode arrays to perform on-chip spike sorting in real-time.

Although there have been sustained efforts to develop real-time spike sorting in FPGAs, most implementations are inefficient in terms of area and power consumption. Want et al., demonstrated a single channel real-time spike sorting while using >90% FPGA resources [15]. Laszlo et al. implemented the "Osort" algorithm in FPGA for sorting 128 recording channels, using hundreds of block RAM and DSP units. However, this approach does not scale well with channel count [16]. On the other hand, resistive switching random access memory (RRAM) has been considered as a promising next-generation memory technology due to its low switching energy,

This paper was submitted on Nov 2021 for review. This work was supported by Office of Naval Research (N000142012405), the National Science Foundation (ECCS-1752241, ECCS-2024776), and the National Institutes of Health (DP2 EB030992).The fabrication of the devices was performed at the San Diego Nanotechnology Infrastructure (SDNI) of the University of California San Diego, supported by the National Science Foundation (ECCS-1542148). Y. Shi, A. Ananthakrishnan, S. Oh, X.Liu, G. Hota and D. Kuzum are with the Electrical and Computer Engineering Department. G. Cauwenberghs are with Bioengineering Department, University of California at San Diego, San Diego, CA 92093, USA (e-mail: dkuzum@eng.ucsd.edu).

non-volatility, high switching speed and small footprint [17]. In-memory computing based on RRAM arrays has been widely used in accelerating data intensive applications such as neural network inferences, computer vision, and compressed sensing [18]. A crossbar array consisting of thousands of RRAM devices offers large non-volatile memory storage and facilitates massive parallelization of matrix-vector multiplications. These advantages make RRAM crossbars uniquely poised to implement a large number of dot-products in real-time with high energy-efficiencies. However, to the best of our knowledge, no studies have yet shown RRAM-based brain interfaces for real-time spike sorting.

In this paper, we designed a compact, energy-efficient, and high throughput neuromorphic brain interface based on CuO_x crossbar arrays that can perform spike sorting for extracellular neural recordings. On the hardware front, we developed a low-temperature fabrication process that is compatible with BEOL CMOS integration to fabricate high-density CuO_x crossbars. We developed a template matching-based spike sorting algorithm that is a hardware-friendly and scalable for mapping onto crossbars. In our neuromorphic brain interface, low amplitude neural signals (few µVs) from an implanted neural probe were amplified and digitized using an Intan amplifier. The neural templates were encoded into device conductances and stored in columns of CuO_x crossbars (**Fig. 1**). Template matching was achieved by feeding neural signals to the wordlines (WLs) and using the crossbar architecture to compute their dot products with corresponding neural templates in each column. The sorting results were obtained parallelly by processing the weighted sum currents in the bitlines (BLs). We experimentally demonstrated the ability of our CuO_x crossbar arrays to sort simulated synthetic spikes as well as extracellular recordings from *in vivo* animal experiments with high accuracy i.e. close to ideal software implementation. Based on experimental results, we also performed a system-level simulation and estimated that our approach can sort 100-channel recordings within 4.8µs with ~1000× reduction in chip area, ~200× reduction in power, and ~50× less energy per channel compared to the state-of-the-art FPGA and microcontroller implementations.

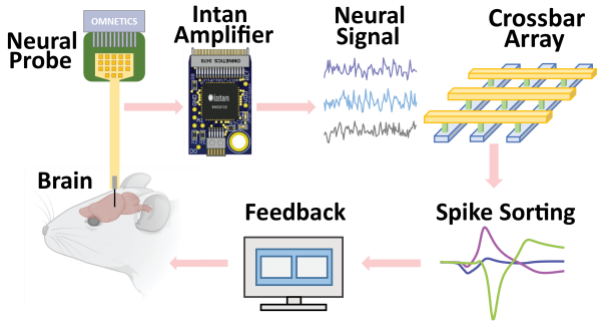

Fig.1 Proposed neuromorphic brain interface based on CuO_x crossbar array for spike sorting. Neural signals recorded by the multichannel neural probe are amplified and digitized using an Intan amplifier and ADC respectively. CuO_x crossbar array performs spike sorting in real-time. That can be used as real-time feedback for a closed-loop neural interface.

The rest of this paper is organized as follows. Section II presents device characterization results for CuO_x devices, including DC switching, transient pulse responses, cycle-to-cycle, and device-to-device variations and retention. Section III describes the template matching algorithm and two datasets used in the hardware demonstration. Section IV explains how the algorithm is mapped to the hardware and the spike sorting in the crossbar. Section V discusses the system-level benchmarking results of our approach in comparison to other hardware implementations. Section VI summarizes this paper.

II. CuO_x Resistive Crossbars

We developed a wafer-scale process for fabricating 16×16 crossbar arrays of Au/CuO_x/Au resistive switching devices (**Fig.2a**). The SEM image of the crossbar array and the cross-section schematic are shown in **Fig.2b** and **d**. The fabrication flow is illustrated in **Fig.2c**. First, Au with Cr adhesion layer (100 nm) is sputtered and patterned via photolithography and lift-off for bottom electrodes (or WLs) with 1µm linewidth and a 2um pitch. Then, 70 nm of CuO_x switching layer is deposited and patterned with reactive sputtering of Cu and Ar/O_2 (95 %/5 %) gas. After that, top electrodes are deposited and patterned following the same fabrication steps as the bottom electrodes. Lastly, 300 nm of SiO_2 layer is deposited and patterned to passivate the device active region to ensure long-term stability. Since all the processes for the CuO_x crossbars are low-temperature process, it can be built directly on the BEOL of CMOS circuits.

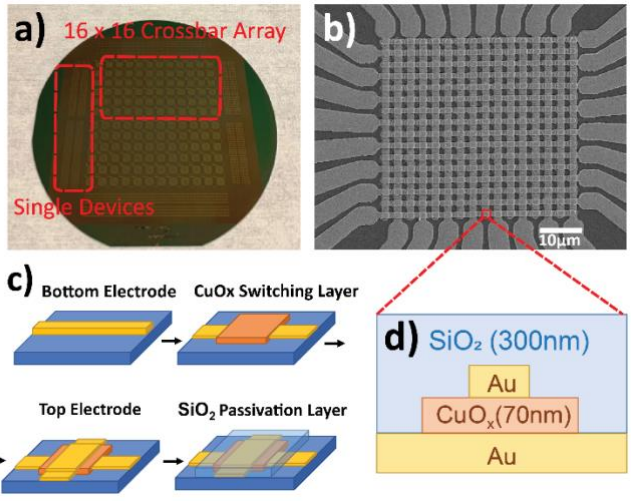

Fig.2 (a) Image of a wafer including fabricated 16×16 CuO_x crossbar arrays and single devices for testing. (b) SEM images of 16×16 crossbar with 4µm^2 cross point. Scale bar: 10µm. (c) Fabrication process for CuO_x -based single devices and 16×16 crossbar. (d) Device cross-section (callout window) highlighting the 70nm CuO_x resistive switching layer sandwiched between 100nm Au electrodes. A 300nm SiO_2 passivation layer is deposited on top of the stack.

After fabricating Au/CuO_x/Au resistive switching devices, we extensively characterized them (**Fig.3** and **Fig.4**). The Au/CuO_x/Au devices displayed consistent bipolar switching in

response to 30 DC voltage sweeps (**Fig. 3a**). They could be set to a low resistance state of ~100Ω at V_{SET} = ~1.5V whereas applying V_{RESET} = ~-0.7V increased device resistances to as high as ~1GΩ with low cycle-to-cycle variations (**Fig.3b**). The high ON/OFF ratio (~ 10^7) of the device resistances (**Fig. 3c**) provides a sufficiently large window for implementing the neuromorphic brain interface. Furthermore, the relatively low SET and RESET voltages **(Fig. 3b)** is desirable for future integration with peripheral CMOS circuitry.

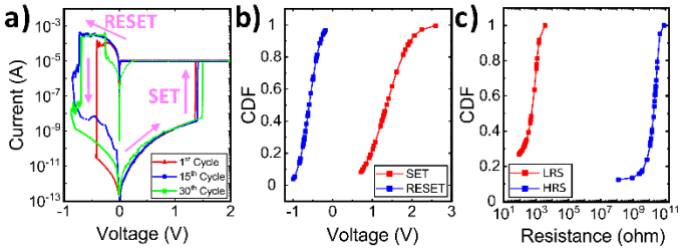

Fig.3 (a) DC switching characteristics of single devices for 30 cycles. (b) Cumulative distribution function (CDF) of SET (1V to 2.5V) and RESET (-1V to -0.2V) voltages. (c) CDF of high resistance state (100MΩ to 100GΩ) and low resistance state (100Ω - 1kΩ) resistances.

Low device-to-device variations are important to ensure accurate mapping templates to the crossbar. To quantify this, we randomly selected 120 Au/CuO_x/Au devices from different regions of the wafer. Cumulative distribution (CDF) of switching voltage and resistance is shown in **Fig.4a** and **b** respectively. The measured SET and RESET latencies are presented in [19]. The RESET transition (~80µs) was significantly faster than the SET process, highlighting the scope for further device optimization. Non-volatility of low resistance state (LRS) and high resistance state (HRS) was characterized by reading the device (V_{read} = 0.1V) at regular time intervals immediately after a successful SET or RESET process. The Au/CuO_x/Au devices could retain their LRS and HRS for more than >10000 seconds, indicating these devices can faithfully store the neuron templates needed for real-time spike sorting and periodic refresh operations could be utilized if experiments taking longer than this time period (**Fig. 4c**).

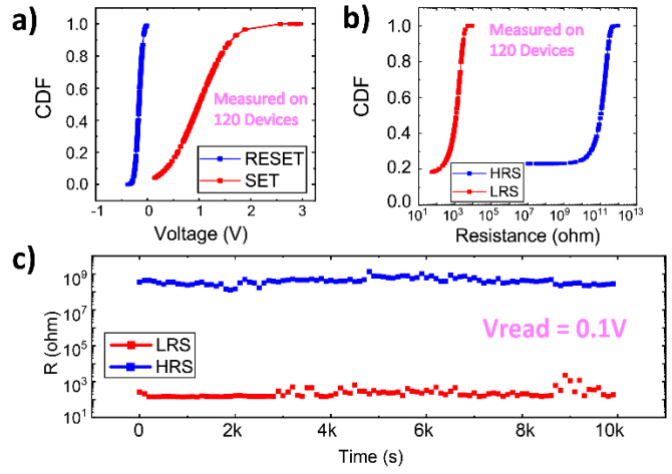

Fig.4 CDF of the (a) switching voltages and (b) HRS/LRS resistances measured across 120 devices randomly selected on the wafer. c) Retention characteristics. Device resistance was monitored intermittently using 0.1V read pulses.

III. Template Matching Algorithm

A. Algorithm Overview

Spike sorting is a challenging clustering problem and many algorithms have been developed over the past years such as principal component analysis [20], template matching [21], Bayesian statistical frameworks [22], and hidden Markov models [23]. Among these, template matching is the most efficient approach to sort neural spikes [24]. It assumes a pre-existing database of neuron templates; the goal is to assign the best-fit templates to the detected spike waveform, hence clustering the spikes to specific neuron units. Motivated by this, we developed a template matching algorithm that can be directly mapped to the crossbars to achieve real-time spike sorting.

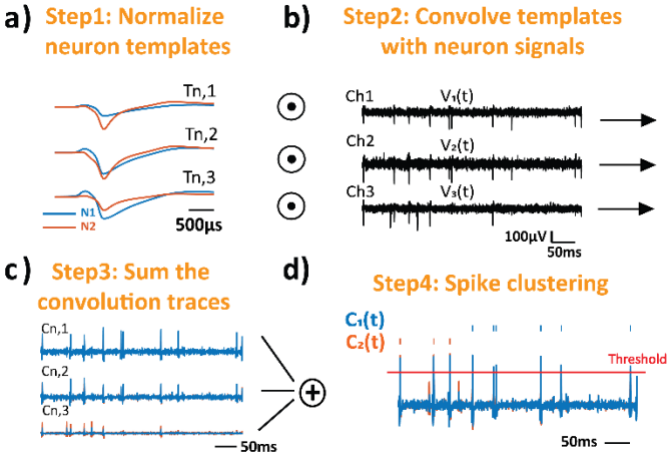

Fig.5 (a) Normalized templates of N1 and N2. (b) Neural recordings of three channels. (c) Computing the overall activation of neuron n neural recordings i.e., voltage traces with normalized templates N1 and N2. Summing the convolution traces ($C_{n,m}(t)$) corresponding to each neuron. d) Thresholding and assigning spikes to neurons N1 or N2 based on whether $C_1(t) > C_2(t)$ (assign to N1) or $C_1(t) < C_2(t)$ (assign to N2).

Figure.5 outlines the algorithm (Step1-Step4) by showing a simplified example for classifying two neurons (n=2) from three-channel recordings (m=3). The same methodology can be used to classify a larger number of neurons recorded across hundreds of channels. Each neuron had a template matrix T_n = $[T_{n,1}, T_{n,2}, \ldots, T_{n,m}]$, where column $T_{i,j}$ represented the template for neuron i corresponding to channel j (**Fig.5a**). The $T_{i,j}$ is a vector with S samples with S = $f_s \times k$, where f_s is the sampling frequency and k is the user-define window that determines the duration of templates. In this example, f_s = 30kHz and k = 3ms. T_n is built by horizontally concatenating these templates across m electrodes (m=3). The template matrix T_n was normalized by its Frobenius Norm ($T_n/\|T_n\|_F$) to maintain the amplitude of the spikes in the same range (**Fig.5a**). Similarly, we defined the neural signal V(t) = $[V_1(t), V_2(t)\ldots, V_m(t)]$, where $V_j(t)$ is the recorded signal from channel j. **Fig.5b** shows an example of recording in three channels at 30kHz. To perform the template matching, we first computed the waveform similarity $C_{n,m}(t)$, which is the convolution between signals from channel m and

the template of neuron n on channel m measured at time t. The convolution can be expressed as $C_{n,m}(t)=V_m(t)*T_{n,m}$, which is simply a sliding dot product between the signal and template. Then, the resulting waveform similarities from all m channels were summed up for each neuron (**Fig.5c**) to give $C_n(t) = \sum_1^m C_{n,m}(t)$, the overall activation of neuron n at time t. In the final step (**Fig.5d**), we applied a threshold, which is ~3 standard deviation of the $C_n(t)$ to identify the spike times. After that, we assigned the spikes to the neuron having the largest $C_n(t)$. Note that these templates are typically obtained offline through a semi-automatic algorithm with human curation to ensure accuracy. The details of mapping templates to the hardware are discussed in Section IV.

B. Datasets

We implemented the aforementioned template matching algorithm on two neural recordings: (1) a synthetic "NeuroNexus-32" data [25] and (2) "real" spikes from *in vivo* animal experiments recorded with the NeuroFITM probe [6] for validating our spiking sorting hardware with different neural electrode technologies. In the "NeuroNexus-32" dataset, the extracellular spiking activities with ground truth were generated using MEArec [25]. MEArec generated data in two phases. In the template generation phase, biophysically realistic neuron models were positioned at different locations of the NeuroNexus-32 probe model to produce extracellular potentials to form a template library. In the recording generation phase, it convolved the templates selected from the library with randomly generated spike trains. Additive Gaussian noise was added to the convolution results to obtain the final recording data. Typically, a channel can record activities of 2-3 neurons nearby. Our synthetic dataset contains extracellular recordings of twelve neurons from 32 channels sampled at 30kHz [19]. The "real" dataset contains 1 hour recordings sampled at 32kHz from an *in-vivo* animal experiment recorded with the 32-channel NeuroFITM probe (**Fig.6a**) [6], where spike sorting results from offline Kilosort algorithm [14] was considered as the ground truth. **Figure 6b** and **c** show representative neuron templates and the recordings in Ch4. Top of **Fig.6c** shows the predicted spike train as square symbols and the clustered neuron spike waveforms are presented in **Fig.6d**. As can be seen, for each neuron, the shape of the clustered spike waveforms closely matched their respective templates. A similar waveform example of NeuroNeuxus-32 and the complete template libraries of both probes can be found in our previous work [19].

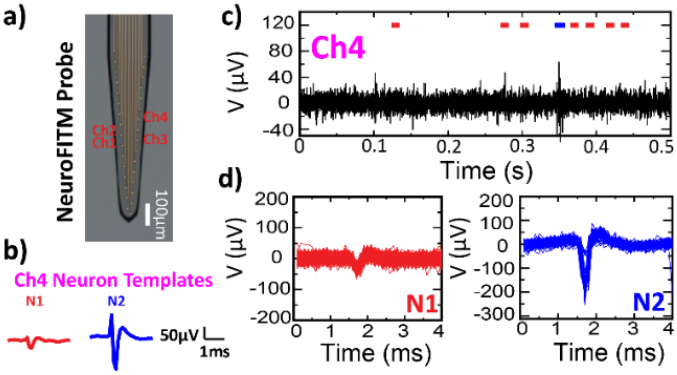

Fig.6 (a) Image of a 32-channel NeuroFITM probe with four representative channels highlighted as red. (b) Representative templates for the two neurons in Ch4. (c) Example 500ms-recordings from Ch4 with predicted spike train marked in colored squares. (d) Clustered spikes for N1 and N2 for Ch4.

C. Sorting Performance

The sorting outcome of our algorithm is determined against the ground truth spikes by comparing the spike time. To quantify the sorting performance, we employed the commonly-used F1 score (in %) given by 2TP/ (2TP+FP+FN), where TP, FP, and FN denote the true positive, false positive, and false-negative outcomes. A TP is defined as a spike that has been classified correctly by the algorithm. An FP is defined as a spike that is classified as spiking activity but does not exist in ground truth data. An FN is defined as a spike that exists in the ground truth data but is not detected by our algorithm. The spike predictions from our algorithm agree with the ground truth well. Eleven out of twelve neurons in the NeuroNexus-32 dataset have F1 score > 90% (**Fig.7a**), whereas all the two neurons in the NeuroFITM "real" dataset have F1 score > 85% (**Fig.7b**). The F1 score of the "real" dataset is slightly less than the synthetic dataset due to higher noise and probe drifting [26] during the recording, making the classification more difficult. To map the templates to the hardware, we investigated how quantization impacts the F1 score. The template was quantized to 2^N discrete levels between the min and max amplitude range of the normalized template library. After quantization, we followed the same sorting pipeline to obtain the F1 score. **Figure.7c** shows that the performance could be retained if the templates are quantized to at least 4-bit resolution for NeuroNeuxus-32 dataset, which is also applied for NeuroFITM dataset.

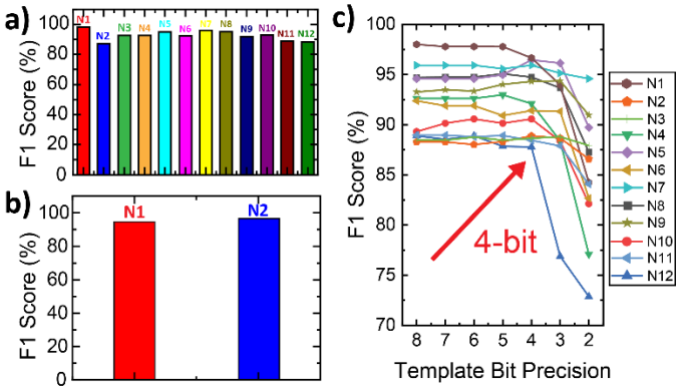

Fig.7 F1 scores (%) for (a) NeuroNexus-32 and (b) NeuroFITM dataset. (c) F1 score (%) as a function of template precision for 12 neurons in NeuroNexus-32 dataset. 4-bit quantized templates are used in hardware experiments.

IV. Hardware Implementation of Spike Sorting

A. *Hardware Mapping*

To process hundreds of spikes per second, it would be necessary to adopt a multi-core architecture (**Fig.8a**) where each core consists of a crossbar that stores the templates for a specific set of neurons (**Fig.8b**). **Figure 8c** illustrates how a set of templates could be mapped on to a crossbar core.

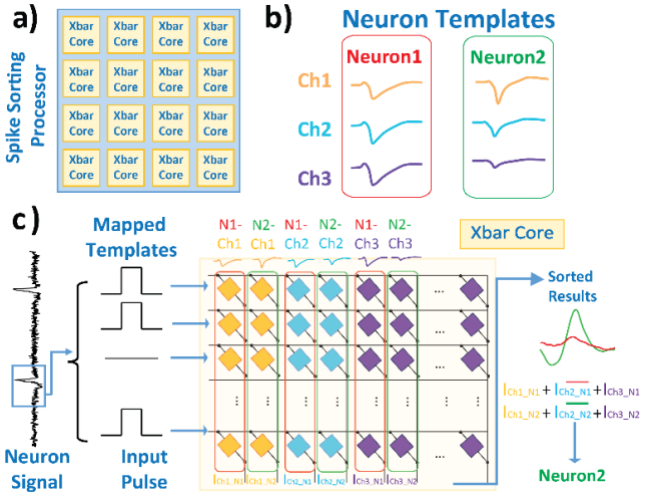

Fig.8 (a) Real-time spike sorting processor with multiple crossbar cores. (b) Representative templates of two neurons with three channels. (c) Crossbar spike sorting: each crossbar column stores a neuron template. 8-bit digitized neural signals are provided as voltage inputs and weighted-sum currents from convolutions are obtained on the BLs. Neuron-wise aggregation of channel currents determines the sorting result.

In the illustration, we assume that three channels (m=3) record spike activities of two neurons (n=2), resulting in a total of 6 templates. The templates from the same channel are mapped to the adjacent columns in the crossbar. The devices in the crossbar can store the templates using multi-level for analog implementation or binary (HRS or LRS) conductance states for digital implementation [27]. A column of devices with 16 (4-bit) multi-level can be used to map a template directly as shown in this example (i.e. templates of N1-Ch1 and N2-Ch2 are mapped to the first two columns of the crossbar respectively). Similarly, templates from other channels are mapped to the rest of the columns to achieve the maximum usage of the array (**Fig.8c**). If binary conductance state is used, four columns are required to map a template from MSB to LSB. Although device with multi-level states can achieve maximum area efficiency, it has been shown that these multi-level states may exhibit high device to device variations, non-linearity and resistance drift due to unstable filament formation [18]. In contrast, digital implementation is more robust against of variations [28], which makes it a better approach to realize high sorting accuracy for template matching task. In addition to conductance states, differential pair scheme is commonly used to represent both negative and positive values of the templates [29].

After all templates are mapped on a core, the voltage spike inputs on WLs (V_{WLi}) are convolved with the templates stored as cross point conductances (G_{ij}). The columns of the crossbar can perform template matching (BL currents $I_{BLj}=\sum G_{ij}V_{WLi}$) in parallel. Since a set of templates from each channel need to convolve with neural signals from the corresponding channel, recordings from Ch1-Ch3 are processed in a time-multiplexed manner, the matching results ($I_{BLn,j}$) for each channel are collected from the corresponding BLs in parallel (n: neuron; j: channel number). The final classification result is obtained by adding the BL currents for each neuron i.e., $I_n = \sum_1^m I_{BLn,j}$ from all m channels and then assigning the spike to the neuron with the maximum I_n. For the sake of illustration, we show all templates mapped to a single crossbar. For practical applications involving large channel counts, a multicore architecture can be adopted, where each core is dedicated to a channel and stores all templates belonging to the assigned channel. As a result, all channels can be processed at the same time to achieve higher parallelism.

B. *Hardware Demonstration*

A custom PCB board was used to access the WLs and BLs of the wire-bonded CuO_x crossbar (**Fig. 9a** and **b**). Before mapping the templates, array read was performed to confirm the initial states of the crossbar. To read a single device, the selected WL was biased to V_{read} = 0.25V while all other lines were grounded. The as-fabricated devices had initial resistances greater than 500kΩ (**Fig.9c**). As explained in Subsection A. Hardware Mapping, digital implementation is adopted in our demonstration. Neuron templates were quantized, binarized, and mapped onto crossbar columns using differential pair scheme. To program the devices to different states, we used $V_{dd}/2$ write scheme, where the selected WL and BL were biased to $V_{dd}/2$ and $-V_{dd}/2$, and all other unselected lines were grounded to prevent sneak paths (SET: V_{dd} = 4V and RESET: V_{dd} = 3V).

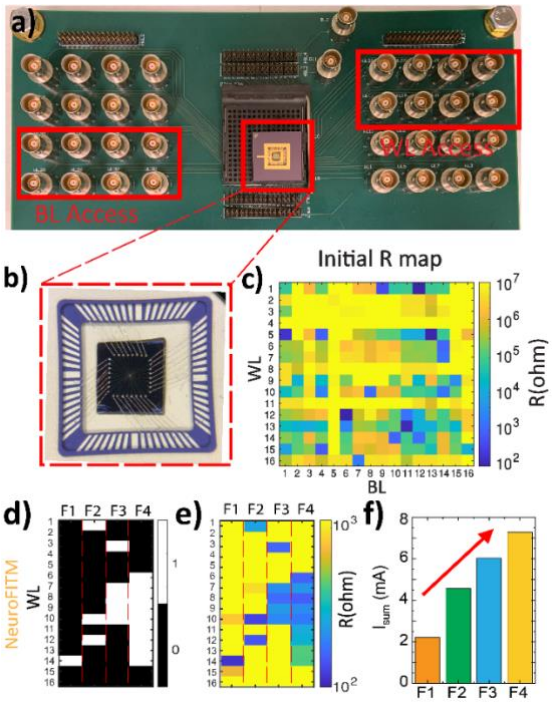

Fig.9 (a) Custom PCB board to access individual WLs and BLs of the CuO_x crossbar for the write and read operations. BLs can be accessed through the connectors shown in the lower left while WLs can be accessed through the connectors in the top right. (b)16×16 crossbar wire-bonded onto a PGA package. (c) Initial resistance map of a 16×16 CuO_x crossbar. (d) Four representative binarized (black=0 and white=1) filters (F1-F4) from NeuroFITM. (e) Programmed crossbar columns implementing these filters. (f) I_{sum} measured at V_{WLs}=0.25V for four filters.

Figure.9d-f shows four representative templates (F1-F4) of Neuro-FITM implemented in the crossbar. The templates were quantized to 4-bit and then binarized to two levels ("0"-black or "1"-white) off-line (**Fig.9d**). "0" is mapped to HRS and "1" is mapped to LRS of the device respectively (**Fig.9e**). Since the crossbar was initially off, only "1" needs to be programmed accordingly. The patterns of the hardware templates match well with software templates, indicating precise write operation. To validate the accuracy of crossbar convolutions, we biased all WLs to high (V_{WLs}=0.25V) and measured the BL currents. As shown in **Fig.9f**, the weighted-sum BL currents (I_{sum}) increased proportionately with the number of LRS devices in the columns. Templates from NeuroNexus-32 dataset are mapped in the same way [19].

Using the programmed templates, we performed spike sorting on NeuroNexus-32 and NeuroFITM recordings. Neural data encoded as 8-bit voltage pulse trains were fed into the WLs and I_{sum} were measured on the BLs. **Figure.10a and Figure.11a** show the NeuroNexus-32 and NeuroFITM recordings and the hardware spike sorting results implemented to sort representative three neurons (N1-N3) from the NeuroNexus-32 data and two neurons (N1,N2) from the NeuroFITM data. The neural voltage traces from the recording channels (Ch1-3 in NeuroNexus-32 and Ch1-4 in NeuroFITM) are shown at the bottom. Hardware convolution trace generated by CuO_x crossbar represents final current $I_n = \sum_1^m I_{BLn,j}$ by adding weighted sum currents measured in each $I_{BLn,j}$ for "m" channels and "n" neurons (NeuroNexus-32: m = 3, n =3; NeuroFITM: m = 4, n = 2). The raster plots on the top of **Fig. 10a** and **Fig.11a** show the spike train predicted in hardware compared with the ground truth spikes for Neuronexus-32 and NeuroFITM dataset, respectively.

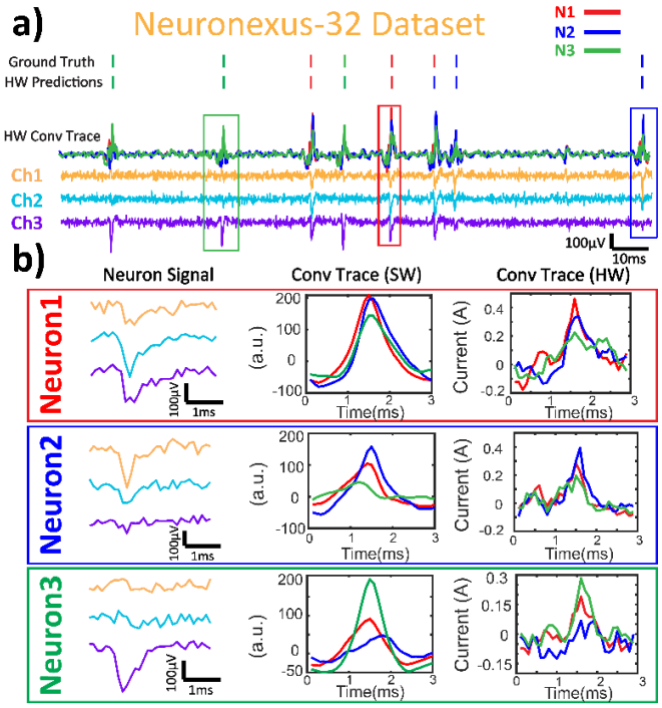

Fig.10 (a) NeuroNexus-32: Ch1,2,3 are used to classify neurons N1, N2, N3. A segment of recordings from Ch1 to Ch3 and predicted hardware (HW) convolution (Conv) traces for three neurons. (b) Representative spike sorting results for N1-N3 showing convolution implemented in HW agrees with the software (SW) implementation.

Figure10b and **Fig.11b** show the callouts for the spikes highlighted in rectangular boxes (**Fig.10a** and **Fig.11a)**. Inside the boxes, the snippet spike waveform of each neuron is shown in the left. Channels are coded in different colors that match with the signal traces above. The template matching results in software and hardware are shown as convolution traces in the middle (SW) and right (HW) respectively. Different colors represent N1-N3 of NeuroNexus-32 and N1-N2 of NeuroFITM. The software convolution traces are shown as arbitrary units while hardware traces are shown as measured weighted sum currents. For each spike, the neuron with the highest peak in the convolution trace was assigned to the spike. The shapes of convolution traces produced by the CuO_x crossbars matched closely with software, thereby confirming our hardware can reliably sort neural spikes. Note that the off-peak regions of the hardware convolution traces are slightly noisy compared with software mainly due to variations in the programmed device conductances across crossbar columns. This issue can be alleviated by adopting a more robust "program and verify" scheme in storing the templates in the crossbar [30].

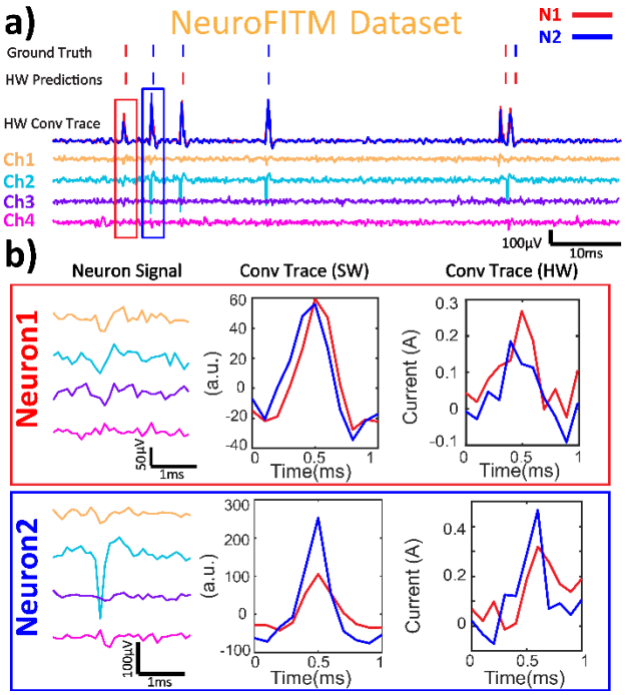

Fig.11 (a) NeuroFITM: Ch1,2,3,4 are used to classify neurons N1, N2. Segments of recordings from Ch1 to Ch4 and predicted HW conv traces. (b) Representative spike sorting results for N1, N2 implemented in HW agrees with the SW implementation.

B. System-level Performance Benchmarking

Based on the hardware spike sorting results obtained over a 100ms time window (**Fig.10** and **Fig.11**), we evaluated F1 scores on the entire 30s-wide recordings in both neural data and compared them with software predictions. The hardware F1 scores were calculated by performing template matching between neural signals with hardware templates that contain measured device resistances. To evaluate sorting performance across multiple neurons, we averaged F1 score based on neuron number. **Table I** shows neurons could be sorted with high mean accuracy (~92.5% for NeuroNexus-32, ~94.6% for NeuroFITM).

Table I F1 Score of SW and HW

	NeuroNexus-32 F1 Score (%)	NeuroFITM F1 Score (%)
SW	92.89%	96.04%
HW	92.48%	94.62%

Table I. F1 score for software and hardware implementations.

To project the sorting performance of multi-core architecture (**Fig.8a**) with our crossbar-based spike sorting hardware, we performed a system-level benchmarking to estimate area, power, and latency and compared it with the state-of-the-art FPGA and microcontroller implementations. All implementations included in **Table II** use *in-vivo* experimental datasets and template matching based approach for a fair comparison. Our work and microcontroller implementation [31] demonstrated sorting for 32-channel probe while FPGA [15] implemented sorting for a single channel. The area per channel was estimated by the number of columns used in mapping a neuron template of a channel (i.e. ~8 columns are used for a channel template and it occupies 40 µm × 20 µm = 8×10^{-4} mm^2). Power per channel was calculated by averaging power consumption $P_{avg} = \sum_1^N I_{sum} \times V_{read}$ across a representative spike waveform snippet during template matching. Here, N is number of samples in the spike waveform (N=30), I_{sum} is the weighted sum current for processing a sample measured crossbar.

Table II Performance Benchmarking

Reference	This Work	[15]	[31]
Hardware	Crossbar	FPGA	Microcontroller
Recording Data	Simulated, in-vivo experiments	in-vivo experiments	in-vivo experiments
No. of channel	32	1	32
Area/Channel (mm^2)	8e-4	> 10	0.78
Power/Channel (mW)	2.15	460	3.11
Sorting Latency (µs)	4.8 per 100 channel	0.72 per channel	169 per channel
Energy/Channel (nJ)	10.3	331.2	525.6

Table II. Benchmarking our results against previous works [15, 31] in terms of hardware type, recording data used in the studies, channel count, area/channel, power/channel, sorting latency, and energy/channel. The accuracy obtained onNeuroNexus-32 and NeuroFITM data from software (SW) and hardware (HW) experiments.

Overall, our crossbar-based spike sorting hardware promises ~1000× smaller (area/channel) [15] and ~200× reduction in power consumption [31] compared to state-of-the-art spike sorting hardware implementations that rely on FPGAs (**Table II**). To better understand the sorting latency in the multicore architecture, we assume one crossbar core can have size up to 256×256 and 10ns read latency. Unlike previous works that rely on sequential processing, each crossbar core in the multi-core architecture can process multiple recording channels in a highly parallelized manner. We estimated twelve CuO_x crossbar (256×256) cores can sort 100 channel recordings within 4.8µs using the same mapping scheme of our hardware demonstration. As a result, it consumes ~30-50× less energy (energy = power × latency) [15, 31]. These performance gains make real-time spike sorting possible using our crossbars for high throughput BMI applications.

V. Conclusion

We presented a high throughput neuromorphic brain interface for real-time spike sorting based on resistive crossbar arrays. We fabricated CuO_x crossbars using a simple low-temperature process enabling easy 3D BEOL integration with underlying CMOS circuits. In order to realize real time spike sorting, we developed a hardware compatible template matching algorithm and developed methods for mapping onto crossbar arrays. We demonstrated that hardware implementation of template matching using CuO_x crossbars can accurately classify spikes from individual neurons recorded *in vivo*. Our neuromorphic approach offers substantial performance gains in area, power, latency, and energy for spike sorting hardware designed for processing recordings from neural probes with high channel counts. Our work paves the

way towards in-memory computing-based real-time spike sorting and processing hardware for next-generation closed-loop brain interfaces.

REFERENCES

[1] M. A. Nicolelis, A. A. Ghazanfar, C. R. Stambaugh, L. M. Oliveira, M. Laubach, J. K. Chapin, R. J. Nelson, and J. H. Kaas, "Simultaneous encoding of tactile information by three primate cortical areas," *Nature neuroscience,* vol. 1, no. 7, pp. 621-630, 1998.

[2] A. P. Georgopoulos, A. B. Schwartz, and R. E. Kettner, "Neuronal population coding of movement direction," *Science,* vol. 233, no. 4771, pp. 1416-1419, 1986.

[3] E. I. Moser, E. Kropff, and M.-B. Moser, "Place cells, grid cells, and the brain's spatial representation system," *Annu. Rev. Neurosci.,* vol. 31, pp. 69-89, 2008.

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

[5] L. Luo, E. M. Callaway, and K. Svoboda, "Genetic dissection of neural circuits: a decade of progress," *Neuron,* vol. 98, no. 2, pp. 256-281, 2018.

[6] X. Liu, C. Ren, Y. Lu, Y. Liu, J.-H. Kim, S. Leutgeb, T. Komiyama, and D. Kuzum, "Multimodal neural recordings with Neuro-FITM uncover diverse patterns of cortical–hippocampal interactions," *Nature Neuroscience,* vol. 24, no. 6, pp. 886-896, 2021.

[7] X. Liu, C. Ren, Z. Huang, M. Wilson, J.-H. Kim, Y. Lu, M. Ramezani, T. Komiyama, and D. Kuzum, "Decoding of cortex-wide brain activity from local recordings of neural potentials," *Journal of Neural Engineering,* 2021.

[8] U. Chaudhary, N. Birbaumer, and A. Ramos-Murguialday, "Brain–computer interfaces for communication and rehabilitation," *Nature Reviews Neurology,* vol. 12, no. 9, pp. 513-525, 2016.

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

[10] N. A. Steinmetz, C. Aydin, A. Lebedeva, M. Okun, M. Pachitariu, M. Bauza, M. Beau, J. Bhagat, C. Böhm, and M. Broux, "Neuropixels 2.0: A miniaturized high-density probe for stable, long-term brain recordings," *Science,* vol. 372, no. 6539, 2021.

[11] S. Kim, P. Tathireddy, R. A. Normann, and F. Solzbacher, "Thermal impact of an active 3-D microelectrode array implanted in the brain," *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* vol. 15, no. 4, pp. 493-501, 2007.

[12] M. A. Nicolelis and M. A. Lebedev, "Principles of neural ensemble physiology underlying the operation of brain–machine interfaces," *Nature reviews neuroscience,* vol. 10, no. 7, pp. 530-540, 2009.

[13] S. Gibson, J. W. Judy, and D. Marković, "An FPGA-based platform for accelerated offline spike sorting," *Journal of neuroscience methods,* vol. 215, no. 1, pp. 1-11, 2013.

[14] M. Pachitariu, N. A. Steinmetz, S. N. Kadir, M. Carandini, and K. D. Harris, "Fast and accurate spike sorting of high-channel count probes with KiloSort," *Advances in neural information processing systems,* vol. 29, pp. 4448-4456, 2016.

[15] P. K. Wang, S. H. Pun, C. H. Chen, E. A. McCullagh, A. Klug, A. Li, M. I. Vai, P. U. Mak, and T. C. Lei, "Low-latency single channel real-time neural spike sorting system based on template matching," *PloS one,* vol. 14, no. 11, p. e0225138, 2019.

[16] L. Schäffer, Z. Nagy, Z. Kincses, R. Fiáth, and I. Ulbert, "Spatial information based OSort for real-time spike sorting using FPGA," *IEEE Transactions on Biomedical Engineering,* vol. 68, no. 1, pp. 99-108, 2020.

[17] S. Yin, Y. Kim, X. Han, H. Barnaby, S. Yu, Y. Luo, W. He, X. Sun, J.-J. Kim, and J.-s. Seo, "Monolithically integrated RRAM-and CMOS-based in-memory computing optimizations for efficient deep learning," *IEEE Micro,* vol. 39, no. 6, pp. 54-63, 2019.

[18] A. Sebastian, M. Le Gallo, R. Khaddam-Aljameh, and E. Eleftheriou, "Memory devices and applications for in-memory computing," *Nature nanotechnology,* vol. 15, no. 7, pp. 529-544, 2020.

[19] Y. Shi, S. Oh, X. Liu, G. Hota,G. Cauwenberghs, D.Kuzum, "High Throughput Neuromorphic Brain Interface with CuOx Resistive Crossbars for Real-time Spike Sorting," *International Electron Devices Meeting,* vol. In press, 2021.

[20] D. A. Adamos, E. K. Kosmidis, and G. Theophilidis, "Performance evaluation of PCA-based spike sorting algorithms," *Computer methods and programs in biomedicine,* vol. 91, no. 3, pp. 232-244, 2008.

[21] J. Wouters, F. Kloosterman, and A. Bertrand, "Towards online spike sorting for high-density neural probes using discriminative template matching with suppression of interfering spikes," *Journal of neural engineering,* vol. 15, no. 5, p. 056005, 2018.

[22] A. Bar-Hillel, A. Spiro, and E. Stark, "Spike sorting: Bayesian clustering of non-stationary data," *Journal of neuroscience methods,* vol. 157, no. 2, pp. 303-316, 2006.

[23] J. A. Herbst, S. Gammeter, D. Ferrero, and R. H. Hahnloser, "Spike sorting with hidden Markov models," *Journal of neuroscience methods,* vol. 174, no. 1, pp. 126-134, 2008.

[24] H. G. Rey, C. Pedreira, and R. Q. Quiroga, "Past, present and future of spike sorting techniques," *Brain research bulletin,* vol. 119, pp. 106-117, 2015.

[25] A. P. Buccino and G. T. Einevoll, "Mearec: a fast and customizable testbench simulator for ground-truth extracellular spiking activity," *Neuroinformatics,* vol. 19, no. 1, pp. 185-204, 2021.

[26] C. Rossant, S. N. Kadir, D. F. Goodman, J. Schulman, M. L. Hunter, A. B. Saleem, A. Grosmark, M. Belluscio, G. H. Denfield, and A. S. Ecker, "Spike sorting for large, dense electrode arrays," *Nature neuroscience,* vol. 19, no. 4, pp. 634-641, 2016.

[27] X. Hong, D. J. Loy, P. A. Dananjaya, F. Tan, C. Ng, and W. Lew, "Oxide-based RRAM materials for neuromorphic computing," *Journal of materials science,* vol. 53, no. 12, pp. 8720-8746, 2018.

[28] S. Yu, Z. Li, P.-Y. Chen, H. Wu, B. Gao, D. Wang, W. Wu, and H. Qian, "Binary neural network with 16 Mb RRAM macro chip for classification and online training," in *2016 IEEE International Electron Devices Meeting (IEDM)*, 2016, pp. 16.2. 1-16.2. 4: IEEE.

[29] P. Yao, H. Wu, B. Gao, S. B. Eryilmaz, X. Huang, W. Zhang, Q. Zhang, N. Deng, L. Shi, and H.-S. P. Wong, "Face classification using electronic synapses," *Nature communications,* vol. 8, no. 1, pp. 1-8, 2017.

[30] W. Shim, J.-s. Seo, and S. Yu, "Two-step write–verify scheme and impact of the read noise in multilevel RRAM-based inference engine," *Semiconductor Science and Technology,* vol. 35, no. 11, p. 115026, 2020.

[31] S. Luan, I. Williams, M. Maslik, Y. Liu, F. De Carvalho, A. Jackson, R. Q. Quiroga, and T. G. Constandinou, "Compact standalone platform for neural recording with real-time spike sorting and data logging," *Journal of neural engineering,* vol. 15, no. 4, p. 046014, 2018.