NEUSORT2.0: A Multiple-channel Neural Signal Processor with Systolic Array Buffer and Channel-interleaving Processing Schedule

Tung-Chien Chen[1,2], Zhi Yang[1], Wentai Liu[1,3] and Liang-Gee Chen[2]
[1]University of California, Santa Cruz, CA, USA; [2]National Taiwan University, Taipei, Taiwan;
[3]National Chiao-Tung University, Hsinchu, Taiwan; Email: djchen@video.ee.ntu.edu.tw

***Abstract*— An emerging class of neuroprosthetic devices aims to provide aggressive performance by integrating more complicated signal processing hardware into the neural recording system with a large amount of electrodes. However, the traditional parallel structure duplicating one neural signal processor (NSP) multiple times for multiple channels takes a heavy burden on chip area. The serial structure sequentially switching the processing task between channels requires a bulky memory to store neural data and may has a long processing latency. In this paper, a memory hierarchy of systolic array buffer is proposed to support channel-by-channel signal processing in cycle basis. The neural data from multiple channels can thus be interleavingly processed in real time with the minimum processing latency, and the NSP can be tightly coupled to the analog frontend interface circuitry without any bulky memory. Based on our previous one-channel NSP of NEUSORT1.0 [1], the proposed memory hierarchy is realized on NEUSORT2.0 for a 16-channel neural recording system. Compared to 16 of NEUSORT1.0, NEUSORT2.0 demonstrates a 81.50% saving in terms of area×power factor.**

I. Introduction

Recent research field of neuroprosthetics has demonstrated that monkeys and human can move computer cursors and robotic arms directly by thought. These proof-of-concept laboratory demonstrations motivate the development of a wireless implantable neural prosthesis that will reduce the surgical infection risk and enable the free movement of test subjects by eliminating chronic transcutaneous connectors and bulky host computers. In such an approach, frontend integrated neural signal processors detecting spike events, extracting features, classifying individual neurons, and decoding inherent meaning can significantly reduce the data bandwidth and is a key to enable such wireless neuroprosthetic device [2], [3].

Several state-or-art neural signal processing hardware along with the analog frontend interface circuitry have been proposed. In [4], [5], comparator-based spike detection modules are integrated in multiple-channel neural recording systems. However, it transmit the binary event streams of the detected neural events, and the significant loss of information limits the ability of classification and sorting of individual neuron signal sources. In [6], a lossy wavelet transformation algorithm and the corresponding architecture are proposed. However, the data reduction ratio is small, and the effect of compression loss in sorting feasibility is still unknown.

Recently, an emerging class of neuroprosthetic devices aims to provide aggressive performance by realizing more advanced signal processing algorithms in particular real-time spike sorting on chips. In [1], a neural signal processor composed of nonlinear-energy-operator-based (NEO-based) spike detector [7], a noise shaping programmable filter, and a maximum-minimum feature extraction engine is integrated in a 128-channel neural recording system. However, it can only support realtime processing on one of the 16 channels, which is still far from the expectation.

It is challenging to provide real-time low-latency signal processing hardware for spike sorting algorithms for multiple channels with low-power and small-area constraints in neuroprosthetic devices. To support multiple channels, the parallel structure is typically used by duplicating the basic processing units multiple times for multiple channels [4]–[6]. However, this approach will take a heavy burden in chip area by duplicating the powerful basic processing units supporting more complicated algorithms.

The folding technique that sequentially does data processing channel by channel on one hardware unit is possible since the modern fabrication processes allow us to drive hardware with more than thousand times of the frequency of the neural recording sampling rate. However, a sophisticated memory hierarchy and the corresponding processing schedule are essential in order to not increase the area and power burdens, and also meet the real-time and short latency requirements.

In this paper, based on the NEUSORT1.0—an one channel neural signal processor [1], a systolic register array structure along with a channel-interleaving schedule is proposed to efficiently support multiple-channel neural signal processing with one processing unit. The reminder of this paper is organized as follows. In section II, the NEUSORT1.0 and the interface of analog frontend interface circuitry are briefly reviewed, and the problems of parallel and serial structures are analyzed. In section III, the memory hierarchy of systolic buffer array structure is proposed along with the channel-interleaving schedule. Section IV presents the implementation results of NEUSORT2.0, a multiple-channel version of NEUSORT1.0, and Section V concludes this work.

II. Background and Problem Statement

A. NEUSORT1.0—One Channel Neural Signal Processor

Figure 1 shows the block diagram of the one-channel neural signal processor [1] that has a 32-tap programmable noise shaping filter, a NEO-based spike detector [7], and a maximum/minimum feature extraction engine. For the

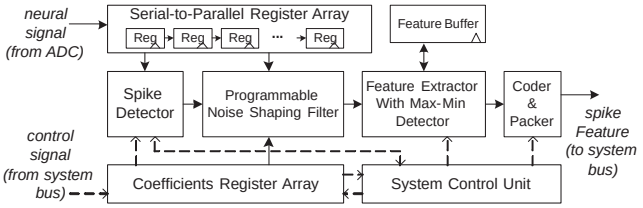

Fig. 1. The block diagram of NEUSORT1.0, an one-channel neural signal processor [1].

detailed algorithm description, please refer to [8]. The input are 9-bit serial neural samples from the analog frontend interface circuitry. The output data after the signal processing consist of 64 bits of three features and timing information of the spike events. Each spike is processed, encoded and sent out on-the-fly within 41 cycles (1.025msec, the processing delay time between the peak sample of a spike event input to the hardware and the last feature score output from the hardware).

During the initial configuration, 32 9-bit filter coefficients for noise shaping operation and a 16-bit threshold value for spike detection are programmed into the coefficient register array. These values can be trained off-chip based on few seconds of initial recordings. During the regular operation, the neural samples are serially input cycle by cycle and shifted forward in the serial-to-parallel register array. After 39 cycles of the data fetch latency, this register array is fully filled with a moving segment of neural waveform for the following signal processing.

There are three processing units—NEO-based spike detector, noise shaping filter, and max-min feature extractor. The spike detector and noise shaping filter are operated in cycle basis. Spike detector calculates the energy function for a latest 7 samples of neural waveform inside the register array every cycle. Once the result reaches the threshold at the peak of the convex curve, a find-spike-event signal will be fired. The noise shaping filter takes the rest 32 neural samples, and output one filtered result after the convolution between neural samples and the filtering coefficients in the coefficient register array. This filter serves a two-folded purpose. First, it is used as the band-pass filter to reject the low-frequency noise and high-frequency thermal noise. Second, the filter outputs the derivative of the spike waveforms for the following feature extraction. The feature extraction engine is operated in event basis. If a spike event is recognized, the feature generation engine is reset and then continuously monitor the maximum and minimum values from the filtered spike samples in the next 32 cycles. During these 32 cycles, the intermediary results of maximum and minimum values of the detected spike event are stored in the feature buffer. The final feature scores along with event timing information are packed and output through the coder&packer. These procedures, including detection, filtering, feature extraction, and coding, are operated in parallel to meet the real-time requirement.

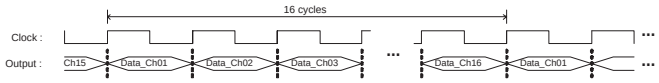

Fig. 2. In order to optimize the trade-off between power consumption and chip area, an ADC and part of amplifier are usually shared by multiple channels. The output are the channel-interleavingly time-multiplexed serial data.

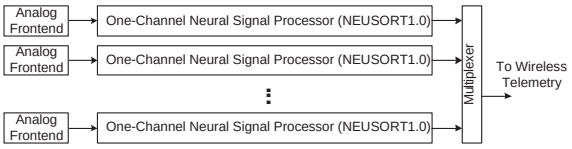

Fig. 3. The parallel structure to support multiple-channel neural signal processing.

B. Analog Frontend Interface Circuitry

An analog frontend interface circuitry of neural recording systems is generally composed of three main parts—pre-amplifier, analog filter, and analog to digital converter (ADC). In order to optimize the trade-off between power consumption and chip area, an ADC and part of amplifier are usually shared by multiple channels. The output are the channel-interleavingly time-multiplexed serial data as shown in Fig. 2. In this paper, the frontend interface circuitry with 16 channel per ADC is used as the optimal case in our integrated mixed-signal system [9].

C. Serial and Parallel Structures for Multiple Channels

Two basic system structures, parallel and serial, are investigated and compared. Figure 3 shows the parallel structure. In order to support multiple channels, multiple neural signal processors along with their analog frontend interface circuitries are used in parallel. This structure may have the smallest power consumption, because there is no additional data buffering and processing requirement. However, it has a large burden on chip area for both analog interface and digital processing hardware.

Based on the optimized analog frontend interface circuitry described in Section. II-B, a serial structure and the correspond schedule is shown in Fig. 4. The neural data are output interleavingly channel by channel from analog frontend interface circuitry and buffered in a memory in order to rearrange the data order for the following processing requirement. During the data processing, the processor is locked by the selected channel, and the corresponding neural data are read sequentially from the memory. Several latency cycles are required to pre-load data in the serial-to-parallel register array (39 cycles in our case). This latency happens once

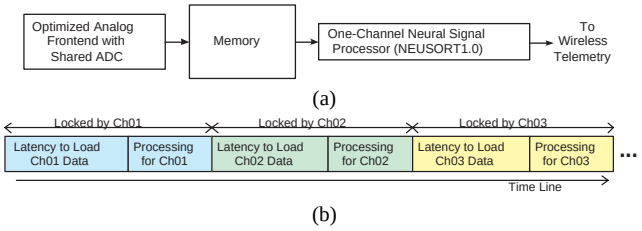

Fig. 4. (a) The parallel structure to support multiple-channel neural signal processing. (b)The correspond schedule.

when the processor switches from one channel to another. In order to increase the hardware utilization, the processor should be locked by one channel as long as possible. In this way, a prohibitively large memory is required for the data rearrangement. In this structure, the power consumption is increased because of an addition buffering requirement for the rearrangement. The chip area may be even larger because of the bulk memory, and the processing latency will become longer.

III. NEUSORT2.0—Multi-channel Neural Signal Processor

A. Proposed Systolic Array Buffer and Channel-Interleaving Processing Schedule

In order to have a small power consumption and chip area, a systolic array buffer and the corresponding channel-interleaving processing schedule are proposed to efficiently support multiple-channel signal processing with one neural signal processor. The idea is to do signal processing interleavingly channel by channel in cycle basis to match up with the data flow of the optimized frontend interface circuitry described in Section II-B.

Figure 5 (a) shows the systolic array buffer for the input neural data. In this case, the processing units process a segment of spike waveforms in parallel within one cycle. There are 16 rows of registers to store the spike waveforms for 16 channels. Every cycle, the data shift forward in parallel according to the arrows shown in Figure 5 (a). An example of the data flow is shown in Fig. 6. The "s01-39" means the 39th neural sample of channel #01. In Fig. 6 (a), "s01-39" is output from the analog frontend interface circuitry to the systolic array. The 1st to 39th neural samples of channel #01 are output from the systolic array to the processing unit for signal processing. At this cycle, the processor is locked by channel #01. At the next cycle, the spike sample of the next channel, "s02-39", is output from the analog frontend interface circuitry. The data in systolic array are shifted forward. As shown in Fig. 6 (b), another 1st to 39th neural samples of channel #02 are output from the systolic array and processed by the processor. At this cycle, the processor is locked by channel #02.

Figure 6 (c) summarizes the channel-interleavingly processing schedule. The neural data are interleavingly output channel by channel from analog frontend interface circuitry. With the proposed input systolic array buffer, the signal processing for different channels can also be scheduled interleavingly in cycle basis. In this way, the neural signal processor can be directly connected to the optimized analog frontend interface circuitry without any bulky memory.

Figure 5 (b) shows another case of systolic array buffer to store the intermediary result output from the processing unit with the same channel-interleaving schedule. The processing units here take more than one cycle to finish the procedure. With the channel-interleaving schedule, the intermediary results are stored and shifted in the systolic array buffer. Every 16 cycles, this intermediary result is rotated back and processed with the new input data.

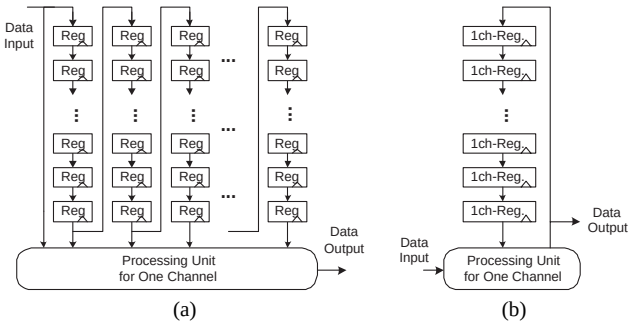

Fig. 5. (a) The systolic array buffer for the input neural data. (b) The systolic array buffer for the intermediary output result.

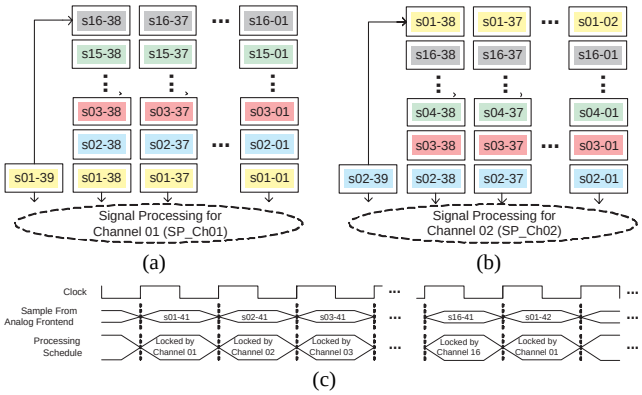

Fig. 6. (a-b) An example of the data flow in the input systolic buffer array. (c) The channel-interleaving schedule for the systolic buffer array.

B. Overall Architecture

The overall architecture of NEUSORT2.0 are shown in Fig. 7. The input are 9-bit channel-interleavingly time-multiplexed neural samples from the optimized multiple-channel analog frontend interface circuitry. The output data after the signal processing consist of 68 bits of three features, timing information, and the channel index of the spike events. The NEUSORT2.0 has exactly the same three processing units as NEUSORT1.0—a 32-tap programmable noise shaping filter, NEO-based spike detector, and maximum/minimum feature extraction engine. However, the data registers in the data path are replaced to the proposed systolic array buffers. The first systolic array buffer stores 16 segments of spike waveforms for the spike detection and noise shaping filter engines. The other systolic array buffer stores 16 sets of the intermediary maximum, minimum values, and time information for feature extraction engine.

In NEUSORT1.0, the operation frequency is 40 kHz that is the sampling rate of the neural signal. In NEUSORT2.0, because 16 times of data are input and processed, 640 kHz operation frequency is used. For each channel, the data are processed ones per 16 cycles. If there is a spike event fired in one channel, it requires 41×16 cycles to complete the detection, filtering, feature extraction, and coding procedures. Compared with NEUSORT1.0, Because the hardware is driven by a 16-times faster operation frequency, the processing delay time is still 1.025 msec.

After the system configuration, the neural samples are input interleavingly channel by channel and shifted in the

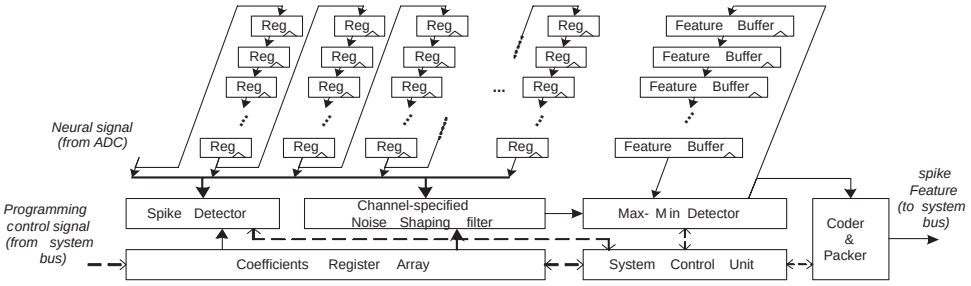

Fig. 7. The overall architecture of NEUSORT2.0.

TABLE I
GATE COUNT PROFILE OF NEUSORT1.0

Hardware Component	Gate Count (k)	Ratio (%)
Serial-to-Parallel Register Array	1.51	8.39
Feature Buffer	0.25	1.39
Spike Detection Unit	1.22	6.78
Noise Shapping Filter	11.53	64.09
Coefficient Register Array	1.19	6.61
Feature Extraction Engine	0.49	2.72
Coder&Packer	0.46	2.56
System Control Unit	0.46	2.56
Others	0.88	4.89
All	17.99	100.00

TABLE II
COMPARISON BETWEEN NEUSORT2.0 AND 16 OF NEUSORT1.0S

	Processing Latency (msec)	Area (Mm^2)	Power (uW)	Area $\times$Power ($um^2 \times uW$)
NEUSORT1.0$\times$16	1.025	1.21$\times$16	106.7$\times$16	33.26×10^9
NEUSORT2.0	1.025	3.36	1830	6.16×10^9
Saving	0	82.74%	-7.19%	81.50%

systolic array buffer. It takes 39$\times$16 latency cycles to preload the systolic array buffer. After this latency, the processor is continuously operated and automatically switched from channel to channel without any bubble and latency cycle. 100% hardware utilization is achieved.

IV. IMPLEMENTATION RESULTS

In order to have a fair comparison between the parallel structure and the proposed serial structure, we implement both NEUSORT1.0 and NEUSORT2.0 with .35 μm 2P4M process. Table I shows the gate count profile of NEUSORT1.0. More than 70% of area is used for three processing units. Only 10% of area is used for the data buffer in the data path. If the parallel structure is adopted to support multiple channels, the area increases linearly with the channel number. With the proposed systolic array buffer and channel-interleaving schedule, 70% area of the processing units are reused with 100% hardware utilization. Only 10% area of storage buffer increases linearly with the channel number. In this way, a neural signal processor can support multiple channels with a very small area overhead.

Table II shows the comparison between the usage of NEUSORT2.0 and 16 of NEUSORT1.0s in a 16 channel neural recording system. With the systolic array buffer and channel-interleaving schedule, 82.74% chip area is saved. There is 8.2% power overhead for the systolic array buffer. Therefore, the totally saving is 81.50% if the power consumption and the chip area are jointly considered.

V. CONCLUSION

To provide high performance neural signal processor (NSPs) for multiple-channel neural recording systems is challenging. The parallel structure duplicating the NSPs takes a heavy burden on chip area, while the serial structure requires a bulky memory between the recording circuitry and NSP. In this paper, a memory hierarchy of systolic array buffer is proposed to support signal processing interleavingly channel by channel in cycle basis to match up with the data flow of the optimized multiple-channel frontend interface circuitry. Based on the one-channel NSP of NEUSORT1.0 [1], the proposed memory hierarchy is realized on NEUSORT2.0, a 16-channel version of NEUSORT1.0. Compared to 16 of NEUSORT1.0, NEUSORT2.0 demonstrates a 81.50% saving in terms of area$\times$power factor.

REFERENCES

[1] M. Chae et al., "A 128-channel 6mw wireless neural recording ic with on-the-fly spike sorting and uwb transmitter," in *Proc. of ISSCC*, Feb. 2008, vol. 7, pp. 146–147.

[2] M.D. Linderman et al., "Signal processing challenges for neural prostheses," *IEEE Signal Processing Magazine*, vol. 25, no. 1, pp. 18–28, 2008.

[3] Z. Zumsteg et al., "Power feasibility of implantable digital spike sorting circuits for neural prosthetic systems," *IEEE Trans. on Neural Systems and Rehabilitation Engineering*, vol. 13, no. 3, pp. 272–279, 2005.

[4] R. H. Olsson et al., "A three-dimensional neural recording microsystem with implantable data compression circuitry," *IEEE Journal of Solid-State Circuits*, vol. 40, no. 12, pp. 2796–2804, 2005.

[5] R. R. Harrison et al., "A low-power integrated circuit for a wireless 100-electrode neural recording system," *IEEE Journal of Solid-State Circuits*, vol. 42, no. 1, pp. 123–133, 2007.

[6] K. G. Oweiss et al., "Optimizing signal coding in neural interface system-on-a-chip modules," in *Proc. of IEEE EMBS Conference*, Sept. 2003, vol. 3, pp. 2216–2219.

[7] K. H. Kim et al., "Neural spike sorting under nearly 0-db signal-to-noise ratio using nonlinear energy operator and artificial neural-network classifier," *IEEE trans. on Biomedical Engineering*, vol. 47, no. 10, pp. 1406–1411, 2000.

[8] Z. Yang et al., "A neuron signature based spike feature extraction algorithm for on-chip implementation," in *Proc. of IEEE EMBS Conference*, Aug. 2008.

[9] M. Chae et al., "Design optimization for integrated neural recording systems," *to appear, IEEE Journal of Solid-State Circuits*, June 2008.