Embedded Spike Sorter

Design of an unsupervised spike sorting accelerator

Zhewei Jiang, Jamis Johnson
zj2139@columbia.edu, jmj2180@columbia.edu

March 26, 2014

1 Overview

In this project, we will design and implement a specialized hardware to perform unsupervised spike sorting which is the process of classifying extracellular action potential of neurons by spike waveform. We will build the sorter on FPGA to simulate the operation of the device as an implanted chip without the need for user input.

2 Motivation

Brain-computer-interface (BCI) research on neural prosthesis or other applications rely on neural spike rate of target region. Implanted electrodes are extracellular and records activities in noisy environment. Each electrode may record the action potential of multiple neurons. It is necessary to cluster the spikes by their spike waveform so accurate spiking rates can be computed and used. In order to improve the practicality of BCI platforms, it is important to develop a closed-loop spike sorting system which can can autonomously operate with minimal parameter or user interference.

3 Algorithm

The spike sorting algorithm we will implement has a training phase and an operation phase, including these major operations: (1) feature extraction; (2) informative sampling, (3) dimensionality reduction; (4) clustering.

3.1 Feature Extraction

We will use discrete wavelet transform (DWT) based feature extraction method to perform multiresolution analysis on input spikes as opposed to other popular methods

such as PCA and time-domain features.Caltech's Wave_Clus package uses four-level wavelet decomposition with Haar wavelet.[1] This technique has been proposed as fixed-delay discrete derivative for on-chip feature extraction.[2] We perform $\sqrt{2}$ scaled Haar decomposition on spike samples to extract coefficient feature.

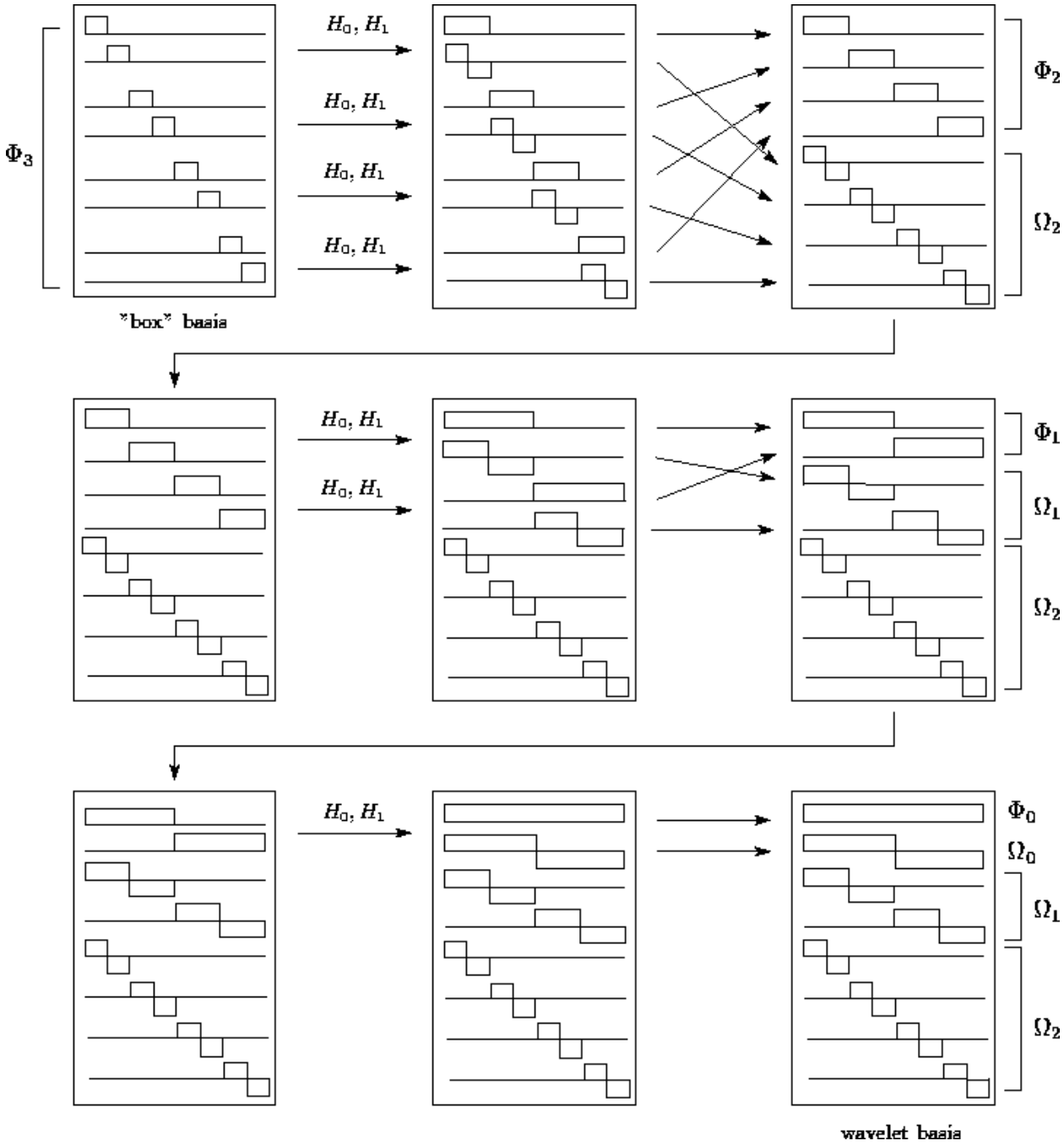

Figure 1: Haar wavelet basis [3]

DWT based features is less susceptible to noise. The use of Haar decomposition strengthens the feature character in near proximity, making this algorithm better suited for sorting highly similar spikes.

3.2 Informative Sampling

During training, spike sample's absolute values and wavelet coefficients are used to learn the feature distribution. Informative samples are defined as features belonging to the modal peaks in the multimodal distribution.
Convolve the distribution with negative Laplace operator [-1,2,-1] gives the negative of distribution's 2nd discrete derivative which indicates modal peak within one standard deviation if the resulting value is positive. To segment the distribution range by mode, minimum between modal peaks is set as boundary of the modes which favors the modes with higher variance.

$D = d * L$
For ith component of D
    If $D_{i+1}<0$ & $D_i < 0$
        D_{tmp} = minimal of (D_i, D_{tmp})
    If $D_{i+1}>0$ & $D_i < 0$
        Set D_{tmp} as a new boundary
    if $D_i > 0$
        Set as modal peak

Ideally, cluster centroid features are 64 dimensional vector with each feature in modal peak. In reality, overlapping modes, dissimilar membership, and other factors result in spike centroids without corresponding modes. A neural network for checking waveform similarity can be used to find cluster centroids without fixating on modal distribution.

3.3 Adaptive Resonance Theory

ART is an unsupervised neural network, also known as Carpenter-Grossberg Classifier. Classic ART1 accepts binary input, therefore we encode the informative sample features as one hot values indicating modal segment.
The basic topology of ART1 is given in Figure 2. There are two layers in this network, where F_1 is the input layer, F_2 is the decision layer, and a single inhibition neuron R with predefined vigilance parameter. Neurons in different layers are fully connected and there are no synapses between neurons of the same layer.

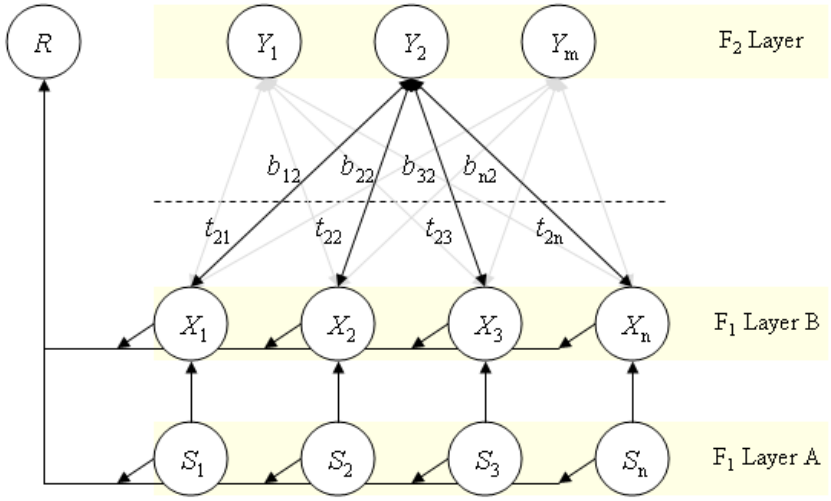

Figure 2: Basic ART1 Topology [4]

The algorithm [5]:

Step 0. Initialize parameters:

$L > 1$,

$0 < p < 1$.

Initialize weights:

$0 < b_{ij}(0) < L/(L-1+n)$,

$t_{ji}(0) = 1$.

Step 1. While stopping condition is false, do steps 2-13.

Step 2. For each training input, do step 3-12.

Step 3. Set activations of all F_2 units to zero.

Set activation of $F_1(a)$ units to input vector s.

Step 4. Compute the norm of s:

$$||s|| = \sum_i s_i$$

Step 5. Send input signal from $F_1(a)$ to the $F_1(b)$ layer:

$x_i = s_i$

Step 6. For each F_2 node that is not inhibited:

if $y_j \neq -1$, then

$$y_j = \sum_i b_{ij} x_i$$

Step 7. While reset is true, do step 8-11.

Step 8. Find J such that $y_J \geq y_j$ for all nodes j.

If $y_J = -1$, then all nodes are inhibited

This pattern cannot be cluster.

Step 9. Recompute activation x of $F_1(b)$:
$x_i = s_i t_{Ji}$

Step 10. Compute the norm of vector x:

$$||x|| = \sum_i x_i$$

Step 11. Test for reset:
If $||x||/||x|| < p$, then
$y_J = -1$ (inhibit node J) (goto Step 7)
If $||x||/||x|| \geq p$
goto step 12.

Step 12. Update the weights for node J (fast learning);
$b_{iJ}(new) = Lx_i/(L-1+||x||)$
$t_{Ji}(new) = x_i$

Step 13. Test for stopping condition.

n number of components in input vector
b_{ji} bottom-up weights (from $F_1(b)$ unit X_i to F_2 unit Y_j)
t_{ji} top-down weights (from F_2 unit Y_j to F_1 unit X_i)
p vigilance parameter
s binary input vector (n-tuple)
x activation vector for $F_1(b)$ layer (binary)
||x|| norm of vector x, defined as the sum of the components of x_i

3.4 Clustering

After training, ART determines the cluster centroids and allows clustering be merely assignment based on a specific distance metric. Due to non-orthogonality of the features, euclidean distance metric is biased against noise portion of the signal. Hence simple L1 distance metric is used for clustering.[6]

3.5 Hierarchical ART Organization

Time domain features have positive correlation with proximity, hence it may be beneficial to stack several layers of smaller ART to perform dimensionality reduction instead of a large one. Depending on later design choice, the ART can be a $\sum_i^{64} mode_i$ input network, or a four layer ART with top-layer as 16 $\sum_i^{3} mode_i$ input network.

4 Software Prototype

The verification of algorithm has been under study for some time, all components of the algorithm have been simulated and tested during prior research. A low dimensional variant of the design without ART and with different modal metric is currently under publication [7], and a 32 dimensional ART design with different modal metric has been fully verified in software, providing justification for the feasibility of the project.

5 Hardware

Implantable BCI system has stringent silicon area constraint and the algorithm we propose reflects such need. Hence, the resources on the FPGA is more than enough to implement all aspects of the algorithm and allows us to use high dimensional feature extraction to boost sorting accuracy than typical SoC implementations which are restricted to operate with low dimensional features.

5.1 Multiresolution Analysis

The feature extraction step performs a three-level multiresolution decomposition with Haar wavelets, resulting 32 wavelet coefficients. As the wavelet transform is performed with the fixed Haar wavelet packet, the discrete wavelet transform can be implemented without multipliers as shown in figure 3.

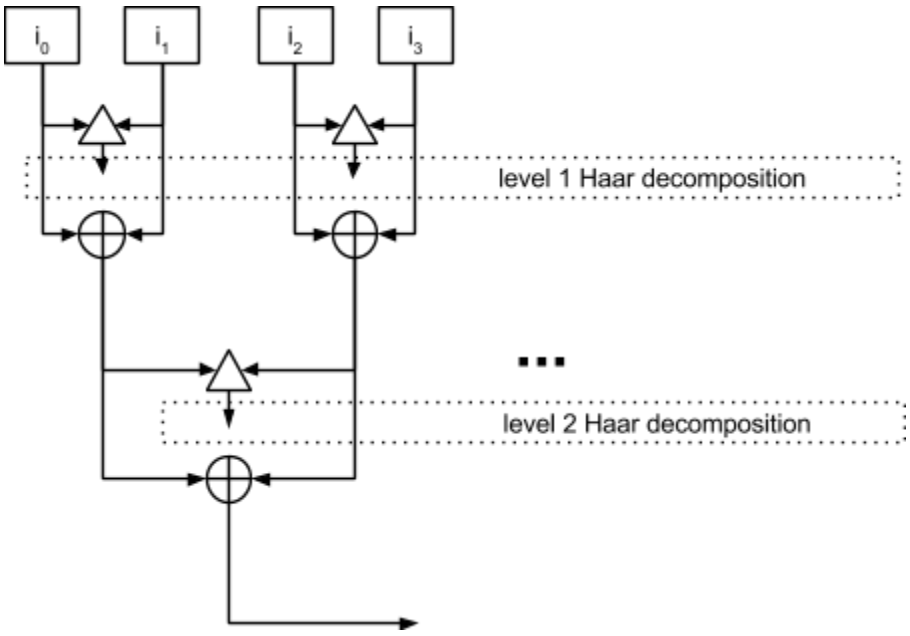

Figure 3: Haar decomposition hardware

The decomposition results using simple adders and subtractors givens linearly scaled wavelet coefficients which does not impact the feature distribution.

5.2 Discrete Laplace Operator

Use 1-D discrete Laplace operator as sharpening kernel, we can find the 2nd order discrete derivative of feature distribution as informative sampling metric as well as modal boundary. Each distribution is segmented by the absolute minimum index of sharpened distribution between positive islands. Shown in figure 4.

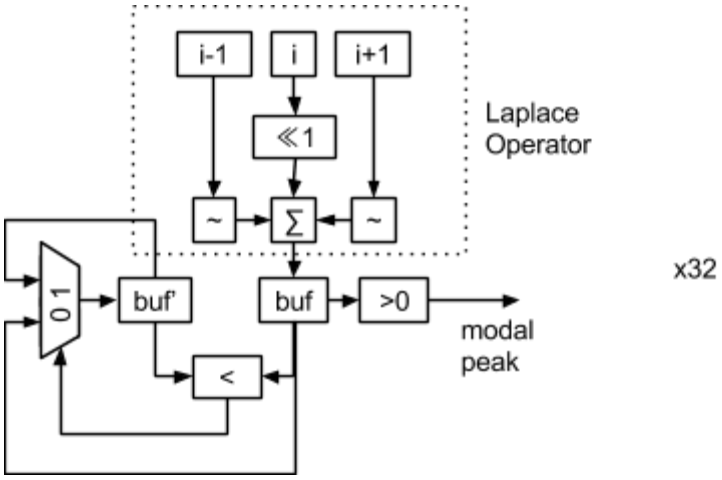

Figure 4: Laplace operator based distribution segmentation

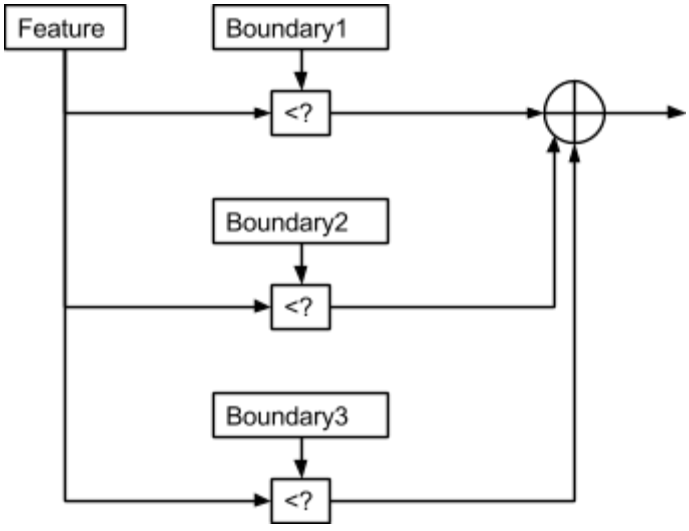

Figure 5: Feature index assignment hardware

The boundaries for each feature are stored separately. After training phase, new incoming spikes features are each broadcasted to banks of comparators to locate the feature segments. Shown in Figure 5.

5.3 Adaptive Resonance Theory

The ART design is yet to be finalized. The tested design of 32 dimensional ART has the following architecture.

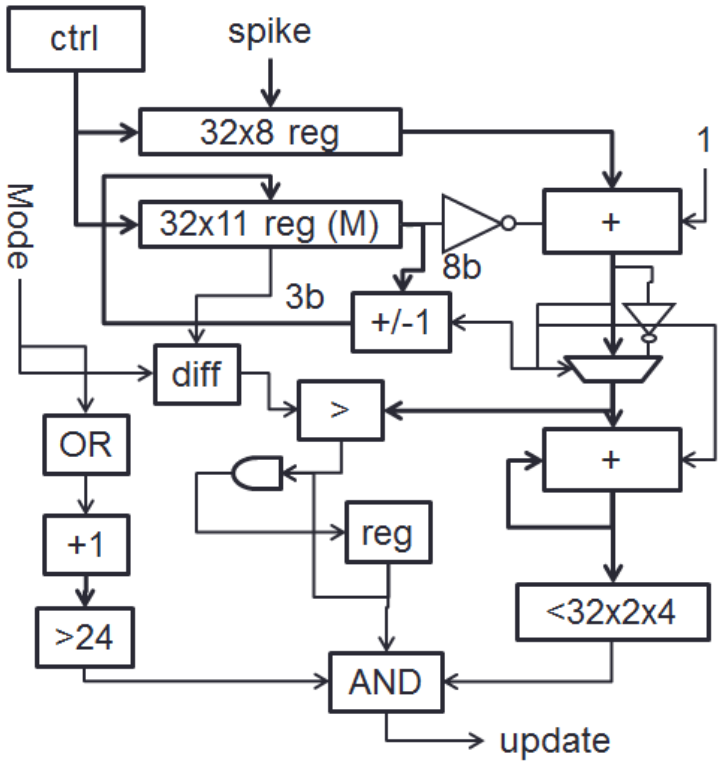

Figure 6: Absolute value feature ART hardware

For stacked ART with low dimensional input for each top-level network, the vigilance parameter should be 100% such that the inhibition neuron can be omitted and reduce to template matching.
Low dimensional template matching design is shown in figure 7.
Final design will settle somewhere between the two, with the low dimensional design output as ART input, layer partition between input and cluster decision is at the moment still fluid.

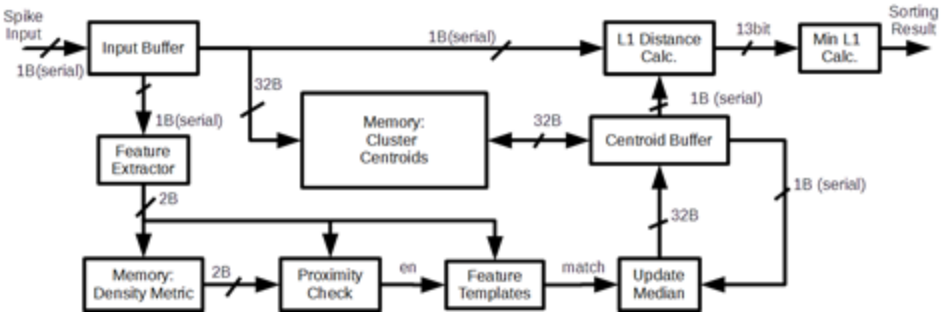

Figure 7: Top level hardware block diagram

6 Software

Software schedule phases of the spike sorting algorithm, and may perform metric evaluation of informative samples.

6.1 Kolmogorov–Smirnov Test (?)

One possible alternative of Laplace operator based modal peak selection is k-s test.

$$F_n(x) = \frac{1}{n}\sum_{i=1}^{n} I_{[-\infty,x]}(X_i)$$

$$I_{[-\infty,x]}(X_i) = \{1 \ if \ X_i \le x; \ 0 \ otherwise$$

$$D_n = sup\,|F_x(x) - F(x)|$$

By using chi-squred approximation to the likelihood ratio, multimodal distribution fitting can be done in software to potentially prevent distribution non-ideality artifact from creating false modes. This may or may not work without reference distribution.
This remains a design decision to be determined.

6.2 Phase Scheduler

As our design has multiple operation phases, each has unique data path and operations. It is the software's role to schedule the hardware for different operations. The hardware can be considered a RISC processor that only handles these instructions:

I1: Haar decomposition + update distribution
I2: Convolve distributions with Laplace operator (segment feature, find modes)
I3: Haar decomposition + informative sample screen + update cluster feature
I4: Haar decomposition + L1 distance metric + assign to cluster

After each instruction, the hardware waits for next input and acknowledges result.

7 Milestones

Completed Milestones

1. Written Matlab code that implements the spike sorting algorithm
2. Design block diagram of the hardware

Milestone 1

1. Finalize design choices on hardware and software partition
2. Determine hardware/software interface based on task partition
3. Design control component of device driver

Milestone 2

1. Begin RTL coding of major hardware blocks
2. Establish control scheme in software

Milestone 3

1. Full implementation of the design
2. Finish testbench of the overall design

Final Project Presentation

1. Finish testing design in FPGA
2. Report and presentation write-up

References

[1] R. Q. Quiroga, Z. Nadasdy and Y. Ben-Shaul, “Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering,” *Neural Comp*, no. 16, pp. 1661-1687, 2004.
[2] M. Zamani and A. Demosthenous, “Feature Extraction Using Extrema Sampling of Discrete Derivatives for Spike Sorting in Implantable Upper-Limb Neural Prostheses,” IEEE Trans. on Neural Sys. and Rehab Eng, vol. 22, no. 4, July 2014.
[3] D. Zorin, S.Cohen, and J. Owens http://robotics.stanford.edu/~scohen/lect02/
[4] J. McCullock, http://mnemstudio.org/neural-networks-adaptive-resonance-theory.htm
[5] LV. Fausett. *Fundamentals of Neural Networks: Architectures, Algorithms And Applications*, 1993.
[6] V. Karkare, S. Gibson and D. Markovic, "A 75-μW, 16-channel neural spike-sorting processor with unsupervised clustering," JSSC, vol. 48, no. 9, 2013.
[7] Z. Jiang, Q. Wang, and M. Seok, “A Low Power Unsupervised Spike Sorting Accelerator Insensitive to Clustering Initialization in Sub-optimal Feature Space,” *ACM IEEE DAC Design Automation Conference (DAC)*, 2015, accepted for publication.