34th Annual International Conference of the IEEE EMBS
San Diego, California USA, 28 August - 1 September, 2012

A Dual Mode FPGA Design for the Hippocampal Prosthesis

Will X. Y. Li[1], *Student Member*, *IEEE*, Rosa H. M. Chan[1], *Member*, *IEEE*, Dong Song[2], *Member*, *IEEE*, Theodore W. Berger[2], *Fellow*, *IEEE* and Ray C. C. Cheung[1], *Member*, *IEEE*

***Abstract*—One important step towards the cognitive neural prosthesis design is to achieve real-time prediction of neuronal firing pattern. An FPGA-based hardware computational platform is designed to guarantee this hard real-time signal processing requirement. The proposed platform can work in dual modes: generalized Laguerre-Volterra model *parameters estimation* and *output prediction*, and can switch between these two important system functions. Compared with the traditional software-based platform implemented in C, the hardware platform achieves better efficiency in doing the biocomputations by up to thousandfold speedup in this process.**

I. INTRODUCTION

Neural prostheses are artificial devices that can be used to substitute brain modalities which might be damaged as a result of injury or disease. The research in neuroprosthetics is catching the attention of scientists from various disciplines. These prosthetic devices, once successfully developed, would provide more fundamental remedies for certain diseases that are considered incurable by traditional medicine [1].

We are now in the process of developing a hippocampal cognitive neural prosthesis. This biomimetic prosthetic device is expected to perform bidirectional interaction with the hippocampus. The prosthesis functions to transform the spatial temporal pattern of input spike trains recorded in CA3 region to the spatial temporal pattern of output spike trains recorded in CA1 region.

The mathematical underpinnings of this cognitive neural prosthesis is the generalized Laguerre-Volterra model (GLVM) which was established earlier in our group by Song *et al.* [2]. The GLVM is a rigorous mathematical model which describes the highly complicated neuronal process from a system input/output relationship standpoint. For a long time, the generalized Laguerre-Volterra algorithm is only implemented using digital software running on commercial desktops and workstations, which cannot guarantee hard real-time signal processing that is required by future prosthetic applications. This paper presents a hardware-based means of implementation of the GLVM employing modern field-programmable gate array (FPGA) devices, which we deem as an important step towards the cognitive neuroprosthetic system design.

There are some previous investigations regarding the hardware implementation of the GLVM. The initial work was carried out by Berger *et al.* in 2005 [3]. Later, Hsiao *et al.* designed a *in vitro* hippocampal CA3 silicon prosthesis and completed the fabrication using the TSMC 0.18*um* process [4]. Recently, Li *et al.* adopted the Xilinx Virtex-6 FPGA platform and conducted the GLVM parameters estimation with high efficiency [5]. Compared with previous works, the work presented in this paper has two distinct features. First, instead of implementing the single-input, single-output (SISO) model as done in [3] and [4], we are targeting the more complicated multi-input, multi-output (MIMO) model. The MIMO model is more physiologically plausible due to the fact that memory-related information in the brain is coded in a distributed manner among populations of neurons. Second, distinct from the work presented in [5], the upgraded platform can be used for conducting both GLVM parameters estimation and neuronal firing pattern prediction as shown in Fig. 1. It can switch between these two important system functions.

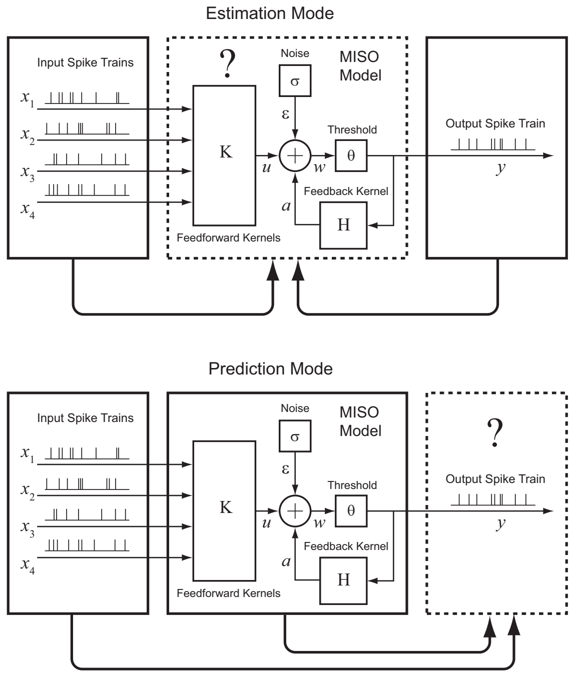

Fig. 1. The dual working mode of FPGA: model parameters estimation and model output prediction.

The major contributions of our work consist of two parts.

- We have successfully designed of an FPGA-based computational platform for conducting hippocampal neural firing pattern prediction. This work is an important step towards the hippocampal cognitive neural prosthesis design. (To be elaborated in Section IV)
- We have achieved good speedups in doing both GLVM

This work is supported by the CityU Start-up Grant 7200179.

[1]Will X. Y. Li, Rosa H. M. Chan and Ray C. C. Cheung are with the Department of Electrical Engineering, City University of Hong Kong, Hong Kong xyli@ee.cityu.edu.hk; rosachan@cityu.edu.hk; rcheung@cityu.edu.hk
[2]Dong Song and Theodore W. Berger are with the Department of Biomedical Engineering, University of Southern California, Los Angeles, CA 90089, USA dsong@usc.edu; berger@bmsr.usc.edu

parameters estimation and output prediction compared with the previous software-based approach. (To be elaborated in Section III)

II. System Identification

A. The Generalized Laguerre-Volterra Algorithm

In this subsection, we have a brief review on the generalized Laguerre-Volterra algorithm. The detailed description of this mathematical model can be found in Section 2 of [2].

The MIMO GLVM can be decomposed into a series of multi-input single-output (MISO) models which are structurally identical. Each MISO model projects to a separate output and has its own physiologically plausible components as shown in Fig. 1. One important variable in the MISO system is the pre-threshold membrane potential w. An all-or-none output spike is generated when w crosses a predefined threshold value θ.

w itself is determined by three system variables: 1) the synaptic potential u which is generated by a feedforward Volterra kernel K, 2) the after-potential a which is generated by a feedback Volterra kernel H and 3) a noise term ϵ which captures the influences of intrinsic neuronal noise and unobserved model inputs.

One of the major challenges in direct Volterra modeling is the large amount of open system parameters to be estimated, which brings considerable computational burden and makes the model easily untrackable. To overcome this problem, we employ the Laguerre expansion of Volterra kernel (LEV) technique by which both feedforward and feedback Volterra kernels are expanded with the Laguerre basis functions. With model input and output first convolving with the basis functions, the number model coefficients are largely reduced and can be updated in a iteratively way using the re-weighted least-squares method [6].

B. Architecture of the Hardware Platform

The general architecture of the hardware platform can be found in Fig. 2. The hippocampal neural firing recordings gathered from animal experiment are pre-stored in the host desktop. They are read and sent to the FPGA processing core via Ethernet connection. The hardware processing core is consisted of several important units, which are functional modules designed to address different stages of the algorithm. On completion of the DSP by these design units, the calculation results are sent back to the host desktop via the Ethernet IP. Some results are looped back to certain registers for the next round iteration. These design units can be categorized according to their specific functions as discussed below.

1) Calculating the Membrane Potential: The membrane potential w is an important parameter in the MIMO GLVM. At the estimation stage, it is linked to the firing probability via the Gaussian error function; at the prediction stage, it is one of the inputs to the threshold-trigger where the predicted spikes are generated. w is calculated by U1 and U2 in Fig. 2. U1 is designed to convolve the model input and previous-round model output with the Laguerre basis functions while

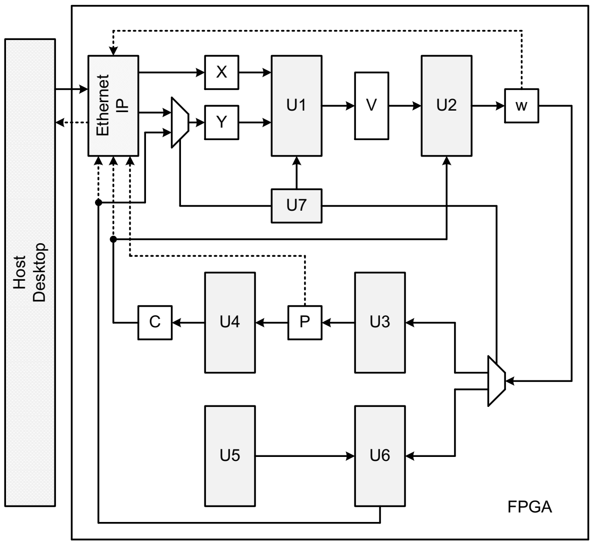

Fig. 2. Overview of the hardware architecture. (U1 is the Convolution Unit; U2 is the Multiplication and Accumulation Unit; U3 is the Firing Probability Calculation Unit; U4 is the Laguerre Coefficients Updating Unit; U5 is Gaussian Random Number Generator; U6 is the Threshold Trigger; U7 is the Control Unit; X and Y are model input and output; w is the pre-threshold membrane potential; P is the neural firing probability; C is the Laguerre coefficients.)

U2 performs the multiplication and accumulation (MAC) operation between the convolution products and the Laguerre coefficients. Unlike our previous design, in the new architecture, U1 and U2 can be fully pipelined when the system works in the prediction mode due to the elimination of feedback loop of the model coefficients. In U1 and U2, different number of processing element (PE) can be adopted, by making a tradeoff between circuit size and calculation efficiency. For the compact architecture, only 1 PE is adopted while for the fully paralleled architecture, the number of PEs is equal to the number of model inputs N (N=64 in current system). The architecture can thereby be implemented in a wide range of Virtex-6 family devices given its multi-fold scalability [5].

2) Calculating the Firing Probability and the Laguerre Coefficients: The neural firing probability P is calculated by the U3 component and the Laguerre coefficients C are updated iteratively by U4. P is acquired via a Gaussian error function $P(t) = 0.5 - 0.5erf\left(\theta - u(t) - a(t)/\sqrt{2}\sigma\right)$. There are several methods to calculate the error function in hardware. An easy approach is to transform the calculation of error function to the exponential function using $erf(z) \cong \sqrt{1-e^{-\frac{4}{\pi}z^2}}$. Other methods and detailed comparisons can be found in [5]. The exponential function itself is calculated by the method of Taylor expansion ($order = 18$) given the limited data range ($[-4\sqrt{2}, 4\sqrt{2}]$) of the independent variable in current system. Laguerre coefficients can thereby be acquired by $C = C + R*((1/P*dP)' + (\hat{y} - P))$, where dP is the gradient of P and $\hat{y}$ is the current value of model output. C is updated only when the system works in the estimation mode. In the prediction mode, it appears as a constant.

TABLE I
DESKTOP HOST CONFIGURATION

CPU	Intel Core i7-2620M (Turbo Boost to 3.40 GHz)
Memory size	8GB (DDR3 1333MHz)
C compiler	gcc 3.4.4-999 running on Cygwin 1.7 platform
Interface	Gigabit Ethernet with jumbo frame enabled

3) Predicting the Output Spikes: This part of circuity is activated only when the system works in the prediction mode. An output spike is generated when the summation of the membrane potential and a randomly generated Gaussian white noise (produced by U5) crosses a threshold value. U6 is the threshold trigger. The Gaussian random number generator (RNG) is designed based on the method first proposed by Tkacik [7]. It is implemented by the bitwise XOR operations between the lower 32 bits of a 43-bit Linear Feedback Shift Register (LFSR) and the lower 32 bits of a 37-bit Cellular Automata Shift Register (CASR).

4) The FSMs and the Control Circuit: The control circuit (U7 in Fig. 2) is designed for the functional switch between the two working modes and the control of their respective timing using Finite State Machines (FSMs). It also has other functions such as being in charge of RNG seed loading, FIFO r/w signal generation and component enabling.

III. RESULTS

As mentioned in Section I, one major contribution of our work is the acceleration of biocomputation utilizing the GLVM. With the aid of the FPGA-based hardware platform, the computation time of both model coefficients and predicted output is largely reduced. Computational efficiency is an important issue in our current research. A typical illustration is the process of conducting the model selection [8]. There are three steps to go through before we can make out whether a particular model input or input pairs contributes to the actual neuronal firing output, as listed:

1) Incorporate the spike train(s) of this particular input or input pair into the in-sample data and re-estimate the model coefficients;
2) Use the updated coefficients and the out-of-sample data to predict the novel model output;
3) Compare the predicted output data to the actual neuronal firing pattern and determine if or not this particular input or input pairs is to be included.

More detailed description regarding model selection can be found in [8]. Traditionally, the selection is conducted using the software suite. However, it is very inefficient considering the long computational time. By using the hardware platform, this process can be sped up.

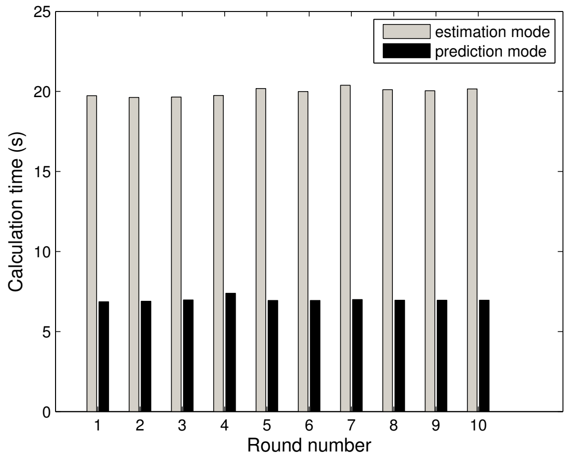

Fig. 3. Calculation time of the software-based platform.

For testing purposes, we use both the software and hardware platforms to process a session of neural recording data gathered from the male Long-Evans rats performing the delayed nonmatch-to-sample (DNMS) task [2]. For the software part, the generalized Laguerre-Volterra algorithm is run for successively 10 rounds. The configuration of the software platform we employ is shown in Table I. The time consumed for conducting parameters estimation and output prediction is recorded and shown in Fig. 3.

The data throughput of the software platform is calculated by $T_{sw} = D \times l/\sum_{i=1}^{l} t_i$. D represents the number of data frames utilized for the test (in our test, it is set as 10,000); l indicates the times of iteration and t is the execution time of each round. Using the data gathered, we can calculate the data throughputs of the software platform (implemented in C) are 500.90 data frames/sec (for parameters estimation) and 1430.94 data frames/sec (for output prediction) respectively.

For the hardware part, the calculation speed can be estimated using equation $T_{hw} = D/(\frac{D_e \times K_{ce}}{f_{clk}} + \frac{D_p \times K_{cp}}{f_{clk}})$. D represents the total number of data frames utilized for test (in our test, it is set as 10,000); D_e is number of data frames used for parameters estimation and D_p is the number of data frames used for output prediction. K is cycles needed for doing one round of calculation. For parameters estimation, $K_{ce} = 67$; for output prediction, $K_{cp} = 68$. During the test, the hardware switches from the mode of estimation to prediction from the 2,000th input data frame. The overall data throughput of the hardware platform can be calculated as 3.83×10^4 data frames/sec. At the stage of estimation, the data throughput can be calculated as 3.88×10^4 data frames/sec (D=De, Dp=0). At the stage of prediction, the data throughput can be calculated as 3.82×10^4 data frames/sec (D=Dp, De=0). The hardware to software speedups are 77.47x and 26.72x respectively during the two stages of calculation.

Our hardware system is very scalable by implementing different number of processing elements within each design component. For the fully paralleled architecture, where the

TABLE II
NORMALIZED MEAN SQUARE ERROR OF CALCULATION

Variable Name	Estimation Mode	Prediction Mode
Membrane potential	1.6202^{-12}	1.6777^{-13}
Firing probability	2.5843^{-11}	n/a
Laguerre coefficients	1.4812^{-11}	n/a
Predicted output	n/a	0

number of PEs equals the number of inputs, the data throughput can reach 1.33×10^6 data frames/sec at the estimation stage and 1.00×10^6 data frames/sec at the prediction stage. The speedups are 2.66×10^3x and 698.84x respectively.

The hardware platform achieves good precision when doing the computations. The IEEE single precision floating point standard is adopted for representation of most system variables. We utilize both the hardware and software platforms to run the two stages of the generalized Laguerre-Volterra algorithm. During the estimation stage, the values of the three important system parameters: 1) the membrane potential, 2) the firing probability and 3) the Laguerre coefficients are recorded. During the prediction stage, the values of the membrane potential and the predicted model output are recorded.

Defining the normalized mean square error (NMSE) as $NMSE = \sum_{t=1}^{T}(y(t) - \tilde{y}(t))^2 / \sum_{t=1}^{T} \tilde{y}(t)^2$, then we calculate the NMSEs for the membrane potential, the firing probability, the coefficients and the predicted output at each stage respectively. The results are shown in Table II.

IV. DISCUSSIONS

A. Cognitive Neural Prosthesis Design

We are now in the process of designing a clinical viable cognitive neural prosthesis that can be implanted into human brain as a substitute for the malfunctioned hippocampal CA3 region. This project is consisted of five important stages, as described in Subsection D, Section V of [5].

Thus far, we have completed the work of the third stage. Although the final hippocampal neural prosthesis will be an application-specific integrated circuit (ASIC) which owns many virtues such as ultra-low power consumption and better area efficiency, FPGAs are very suitable to be adopted as the computational platforms for the research at the second and third stages. This is largely due to their hardware level reconfigurability which brings great flexibility to the designers. Besides, FPGAs are more economical owning to the elimination of the non-recurring engineering (NRE) cost which is common in AISC design flow.

Our work bridges of gap between the mathematical abstraction of the GLVM and the VLSI prosthetic device. The current hardware architecture is to be further upgraded and made more tailed to future prosthetic applications.

B. Future work

In light of the fact that the design is targeted for clinical application, there are certain aspects in which the potential improvements lie.

The first one is the reduction of power consumption. For a brain implantable neural prosthesis, power consumption is of critical importance. The battery life should be ideally as long as possible. Down to the circuit level, techniques employing reduced characteristics length and high-K metal gate process optimized for low power applications can be well adopted.

The second one is the adoption of fault-tolerance design paradigm. It would result in disastrous consequences if the prosthesis fails in operation or produces erroneous predictions. Traditional fault-proof techniques can be utilized such as the employment of hardware, time and information redundancy.

The third one is by resorting to run-time partial reconfiguration technique which is specific for our FPGA platform. By modifying part of FPGA functions during its operation time, more flexibilities are brought in. This is also a good approach for further reduction of both dynamic and static power, and optimization of the circuit architecture.

V. CONCLUSIONS

We have designed an FPGA-based hardware platform for implementation of the generalized Laguerre-Volterra model. This platform achieves two-fold functionality: estimation of model parameters and prediction of model output. It attains significant speedup in conducting the biocomputation compared to the previous software platform. A successful development of the hardware platform is a key stage towards the fabrication of the clinical viable hippocampal neural prosthesis, which serves as our ultimate research objective.

REFERENCES

[1] M. A. Lebedev and M. A. L. Nicolelis, “Brain-machine interfaces: past, present, and future,” *Trends in Neurosciences*, vol. 29, no. 9, pp. 536–546, 2006.

[2] D. Song, R. H. M. Chan, V. Z. Marmarelis, R. E. Hampson, S. A. Deadwyler, and T. W. Berger, “Nonlinear modeling of neural population dynamics for hippocampal prostheses,” *Neural Networks*, vol. 22, pp. 1340–1351, 2009.

[3] T. W. Berger, A. Ahuja, S. H. Courellis, S. A. Deadwyler, G. Erinjippurath, G. A. Gerhardt, G. Gholmieh, J. J. Granacki, R. Hampson, M. C. Hsiao, J. LaCoss, V. Z. Marmarelis, P. Nasiatka, V. Srinivasan, D. Song, A. R. Tanguay, and J. Wills, “Restoring lost cognitive function: hippocampal-cortical neural prostheses,” *IEEE Engineering in Medicine and Biology Magazine*, vol. 24, pp. 30–44, 2005.

[4] M. C. Hsiao, C. H. Chan, V. Srinivasan, A. Ahuja, G. Erinjippurath, T. P. Zanos, G. Gholmieh, D. Song, J. D. Wills, J. LaCoss, S. Courellis, A. R. Tanguay, J. J. Granacki, V. Z. Marmarelis, and T. W. Berger, “VLSI implementation of a nonlinear neuronal model: a ‘neural prosthesis’ to restore hippocampal trisynaptic dynamics,” *Proceedings of the 28th IEEE EMBS Annual International Conference*, pp. 4396–4399, 2006.

[5] W. X. Y. Li, R. H. M. Chan, W. Zhang, R. C. C. Cheung, D. Song, and T. W. Berger, “High-performance and scalable system architecture for the realtime estimation of generalized Laguerre-Volterra MIMO model from neural population spiking activity,” *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 1, pp. 489–501, 2011.

[6] L. Paninski, J. W. Pillow, and E. P. Simoncelli, “Maximum likelihoodestimation of a stochastic integrate-and-fire neural encoding model,” *Neural Computation*, vol. 16, pp. 2533–2561, 2004.

[7] T. E. Tkacik, “A hardware random number generator,” *Proceedings of the 4th International Workshop on Cryptographic Hardware and Embedded Systems*, pp. 450–453, 2002.

[8] D. Song, R. H. M. Chan, V. Z. Marmarelis, R. E. Hampson, S. A. Deadwyler, and T. W. Berger, “Sparse generalized laguerre-volterra model of neural population dynamics,” *Proceedings of the 31st Annual International Conference of the IEEE EMBS*, pp. 4555–4558, 2009.