33rd Annual International Conference of the IEEE EMBS
Boston, Massachusetts USA, August 30 - September 3, 2011

FPGA Implementation of Hardware Processing Modules as Coprocessors in Brain-Machine Interfaces

Dong Wang, Yaoyao Hao, Xiaoping Zhu, Ting Zhao, Yiwen Wang, Yaowu Chen, Weidong Chen, Xiaoxiang Zheng

Abstract—**Real-time computation, portability and flexibility are crucial for practical brain-machine interface (BMI) applications. In this work, we proposed Hardware Processing Modules (HPMs) as a method for accelerating BMI computation. Two HPMs have been developed. One is the field-programmable gate array (FPGA) implementation of spike sorting based on probabilistic neural network (PNN), and the other is the FPGA implementation of neural ensemble decoding based on Kalman filter (KF). These two modules were configured under the same framework and tested with real data from motor cortex recording in rats performing a lever-pressing task for water rewards. Due to the parallelism feature of FPGA, the computation time was reduced by several dozen times, while the results are almost the same as those from Matlab implementations. Such HPMs provide a high performance coprocessor for neural signal computation.**

I. Introduction

BRAIN-machine interface (BMI) aims to build a completely new communication channel between brain and external world to restore lost capabilities for people with damaged sensory/motor functions. One key point of BMI is the real-time mapping from high-throughput neural signals to external kinematic variables [1]. Although a general-purpose processing infrastructure (e.g. a PC) is convenient for implementing the mapping, its currently computational power cannot meet the real-time performance of some complicated algorithms in BMI studies [2]. Meanwhile, practical BMIs must be reconfigurable, portable and flexible, which current platforms cannot meet too. So it is necessary to find a higher performance platform to achieve efficient processing and practical applications.

Recently, some high performance hardware platforms have been brought into BMI research to provide possible solution of current problems. Mehdi et al. [3] designed an implantable Very Large Scale Integrated circuits (VLSI) architecture for real time spike sorting, which reduces telemetry bandwidth of BMIs and improves the practicality of BMI systems in clinical applications. Both Digital Signal Processors (DSP) and FPGA based neural signal processing (NSP) were proposed for real time processing and portable computation platform [4-5]. Although VLSI implementation can provide high processing speed and compact structures, it is more time consuming and cannot be reconfigured. DSP based embedded system can take advantage of its dedicated floating point/integer multiplier and highly specialized processor, but their architecture is sequential, which is not consistent with the parallel characteristics of neural channels. FPGA preserves parallel processing architecture that can simultaneously execute a variety of operations. Besides, FPGA has many dedicated computing units which can further reduce the computation time. Moreover, FPGA can be reconfigured and scaled to specific application and has the advantages of lower cost, higher density and shorter design cycle. Recently some researchers have made full use of parallelism characteristics of FPGA to process neural signal and dramatically improved the computation speed [6-8]. Especially, Zhou et al. [8] proposed the FPGA implementation of neural network for decoding motor cortical ensemble recordings in BMIs and got impressive results.

This work tries to bring forward another framework, i.e. FPGA implementation of hardware processing modules (HPMs) as coprocessors for neural signal processing. HPMs realize BMI algorithms on hardware platform which can accelerate processing. Moreover, HPMs are highly modular and have common interfaces which can facilitate different applications. Implementing the signal processing steps only rather than the entire BMI frame work on the hardware platform is a compromise way that can take advantage of both general-purpose unit and dedicated hardware computation modules. Due to its inherent parallel computing ability, we chose FPGA as the hardware processing platform for two HPMs, probabilistic neural network (PNN) based spike sorting and Kalman filter (KF) based neural ensemble decoding. This paper is organized as follows. Part II introduces the experimental system setup. Part III describes the details of the two HPMs realization. Part IV demonstrates

Manuscript received March 26, 2011. This work was supported by grants from the National Natural Science Foundation of China (No. 60873125, 61031002, 61001172). Zhejiang provincial key science and technology program for international cooperation(No. 2011C14005)

Dong Wang and Yaoyao Hao are with the Qiushi Academy for Advanced Studies, and the Department of Biomedical Engineering and Instrument Science, Zhejiang University, Hangzhou, China.

Xiaoping Zhu and Yaowu Chen are with the Institute of Advanced Digital Technologies & Instrumentation, Zhejiang University, Hangzhou, China

Tingzhao and Yiwen Wang are with Qiushi Academy for Advanced Studies, Zhejiang University, Hangzhou, China.

Weidong Chen is with Qiushi Academy for Advanced Studies, and College of Computer Science and Technology, Zhejiang University, Hangzhou, China

Xiaoxiang Zheng is with Qiushi Academy for Advanced Studies, the Department of Biomedical Engineering and Instrument Science, Zhejiang University, HangZhou, China. (phone: +86 571 87952838; fax: +86 571 87952865; e-mail: qaas@zju.edu.cn).

978-1-4244-4122-8/11/$26.00 ©2011 IEEE

the experiment results. Part V discusses the performance and concludes the paper.

II. System Architecture

A. Neural Data Acquisition

The experiment paradigm is similar to that described in [8]. In brief, rats were trained to learn pressing a lever using their forelimb for water reward, i.e. each rat press a lever equipped in an operant conditioning chamber and then get a sip of water reward. After an animal achieved a good task performance level, a chronic 16-channel microwire electrodes array was implanted into the forepaw region of its primary motor cortex (M1). The 16-channel neural signals were recorded using Cerebus Data Acquisition System (Cyberkinetics Inc., USA) at a sample rate of 30 KHz when the experiments were carried out. And, the pressure of the lever, which is the indicator of the forepaw position, was recorded synchronously by a pressure sensor at a sample rate of 500Hz for neural mapping. Spikes were detected automatically in real time through a thresholding method in the recording system. Both neural signals and lever pressure signals were stored for offline algorithm evaluation and FPGA test. All the experimental procedures were approved by the Animal Care Committee at Zhejiang University.

B. System Setup

The Xilinx Virtex-6 FPGA is chosen to implement the HPMs in this application. Built on a 40 nm copper CMOS process technology, Virtex-6 FPGAs offer 50% lower power, 20% lower cost and higher-speed transceiver capabilities than the previous generation (e.g. Virtex-5). On the FPGA board, a soft processor, namely MicroBlaze, is used to manage state logic of processing blocks and communicate with external master. It is configured under the Xilinx Embedded Design Kit (EDK) environment. In this design, two HPMs, i.e. PNN based spike sorting and KF based neural signal decoding, were developed and taken as intellectual property (IP) for the processor. The system is developed under the Xilinx ISE 11.4 Foundation Design Software Environment using Verilog HDL. Both Matlab and FPGA based realization are developed for evaluation and the data type used in FPGA implementation is single floating point (32-bit) according to IEEE 754 floating point standard.

III. HPMs Implementation

A. Spike Sorting Module

Spike sorting algorithms use the spike shapes information recorded in the vicinity of the electrodes to distinguish one or more neurons from background activity. Probabilistic neural network (PNN), a kind of radial basis networks, is usually used for classification problems. PNN was first described by Specht in detail [9]. By replacing the sigmoid activation function often used in neural networks with an exponential function, PNN can compute nonlinear decision boundaries approaching the Bayes optimal. PNN with different variations can be used for numerous applications, and recently, PNN was first used for neural ensemble decoding in BMIs [10]. In this paper, PNN is realized for spike sorting and its FPGA version is presented.

The dataflow diagram for FPGA implementation of PNN model is shown in Fig.1. Several modules including control module, training module, calculation module, and summation compare module are realized according to the PNN network architectures. Actually, the training process of PNN is just configuring training data to blocks of random access memory (RAM) for later calculation. Here, 48-point spike data and corresponding class numbers are stored into *Training RAMs* and *Class RAM*, respectively. The calculation module has two sub-modules, distance and exponential calculation modules. When an input is presented, the calculation module computes the distances from the input vector to each training vector and feed the results to *Exponential Cal* module. The activation function of the PNN model is the exponentiation function as described in (1).

$$f(X) = \exp(-(W_i - X)(W_i - X)^T / 2\sigma^2) \qquad (1)$$

where X and W_i are input vectors and training vectors respectively, and σ is the smooth parameter. This study uses Taylor series and look-up table (LUT) method to approximate the exponentiation function. For one input value x, exp(x) can be composed of two parts as follows,

$$\exp(x) = \exp(z) \times \exp(f) \qquad (2)$$

where x=z+f, z and f are the integral and fraction part of x, respectively. The value of exp (z) is obtained by LUT and the value of exp (f) is obtained by the Taylor progression as described in (3).

$$\exp(x) = 1 + \sum_{n=1}^{+\infty} \frac{x^n}{n!} \qquad (3)$$

where n is the order of the Taylor series for the input value x.

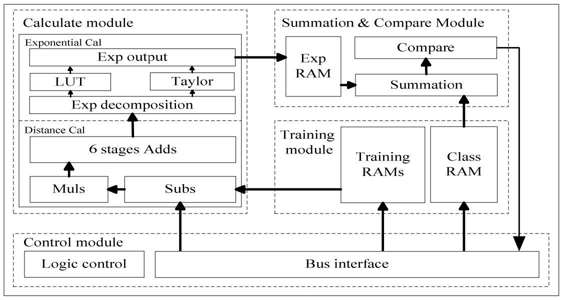

Fig. 1. Dataflow diagram of FPGA implementation of PNN.

Here, we choose n = 7 for the hardware approximation of the exponentiation function. According to the corresponding class number of each spike in the training data, the

Summation module sums the value of exp (x) for each class. Then by comparing the sum of each class the compare module finds the corresponding index of the maximum value, which is the class number of the current spike vector input.

Besides the standard 32-bit single floating point implementation, another 16-bit version was designed to reduce the FPGA resource utilization. The structure of 16-bit floating point is custom-defined, which is composed of 8-bit exponent and 7-bit fraction. The 16-bit results are compared with 32-bit version to test if lower precision can achieve the same sorting accuracy with the reduced resource utilization.

B. Neural Decoding Module

Kalman filter (KF) and its variant, a kind of recursive Bayesian decoder with guaranteed stability and robustness, have been successfully used as a decoding algorithm in BMI for several years [11-12]. It provides optimal state estimates along with the associated confidence regions for a linear Gaussian dynamic system. In the KF framework, the pre-limb movement of rat (position, velocity and acceleration) is modeled as the system state and the neural firing rate in 100ms window is modeled as the observation. FPGA implement of KF is described in detail as follows.

The system architecture of FPGA realization of KF is showed as in Fig.2. Both encoding (training) and decoding (test) parts in the KF algorithm are realized in our FPGA implementation. The communication module takes charge of exchanging data and state information with the external master, the encoding module and decoding module realize the KF algorithm by constantly transmitting data to calculate module and getting its processing results. And the calculate module, including basic processing elements (PEs) and matrix inversion modules, is responsible for almost all the computation task of KF to achieve a high performance.

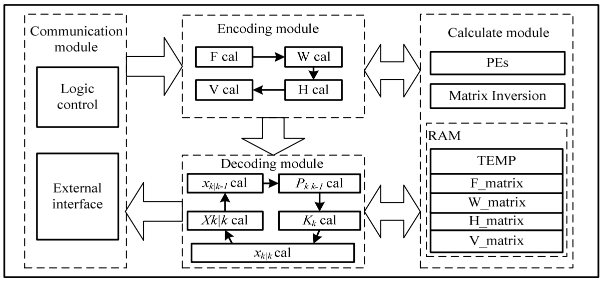

Fig. 2. System architecture of FPGA realization of KF.

For accelerating matrix operations, a row/column-based method is proposed to construct a novel processing element (PE) in our implementation. The row/column based method is realized by storing matrix in row/column and simultaneously choosing the row/column elements to do the matrix operations in one cycle in parallel which is easily realized in FPGA architecture by on-chip block RAM resources. The structure of the novel PE is shown in Fig.3. This PE consists of a floating-point multiplier, a floating-point adder, a block RAM, a user defined FIFO, an AND operator, and eight multiplexers. It can work as an adder, a subtractor, a multiplier, and a Multiply-Accumulator (MAC) depending on the state flag. Several novel PEs are enrolled to form a PE array for parallel operation. The novel PE has uniform structures for easy extension and could be reused for reducing resources usage.

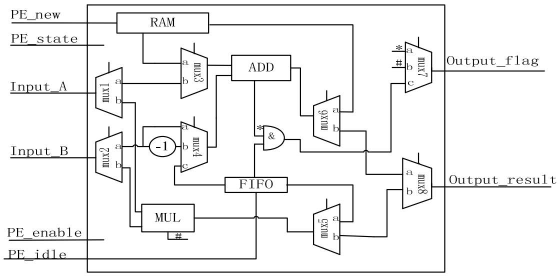

Fig. 3. The structure of the novel processing element proposed.

The matrix inversion module is the key part of Kalman filter, whose implementation is formulated as:

$$B^{-1} = (QR)^{-1} = R^{-1}Q^{-1} = R^{-1}Q^{T} \tag{4}$$

where B is a real matrix, Q is orthogonal matrix and R is a upper triangular matrix. Firstly, QR-decomposition is used to obtain an orthogonal matrix Q and an upper triangular matrix R; Then the inversion of the triangular matrix and transposed matrix of the orthogonal matrix are calculated; finally the result is obtained by the inversion of the triangular matrix left multiplying transposed orthogonal matrix. All the adders, subtrators, multipliers and divisions of floating-point involved here are generated by the Xilinx Floating point IP with a nine-stage pipeline structure.

IV. Results

Three rats were trained and implanted with electrodes successfully and the data from one of the rats are used for test in this experiment.

A. PNN Results

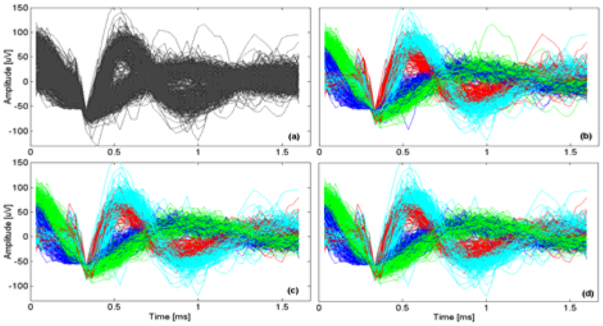

Fig. 4. The representative sorting results. (a) original spikes; (b) manual sorted result; (c) and (d) FPGA and Matlab sorted results.

As shown in Fig.4, The representative results of Matlab and FPGA-based implementation are compared with standard manual spike sorting results in this section. The 32-bit FPGA implementation results get a high accuracy of 93.83%, which is also as accurate as the Matlab version. The accuracy of

16-bit version is a little lower (93.67%), but the performace drop is neglectable. The results indicate that FPGA implementation of PNN is feasible for spike sorting and lower precision realization using fewer resources does not affect the results in this application.

B. KF Results

A total of 32 neurons were sorted from the 16 electrodes from one rat and used to decode forelimb pressure trajectory.

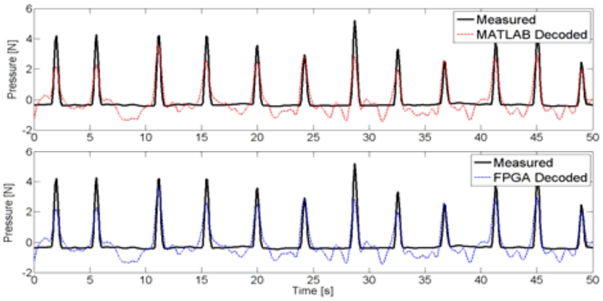

Fig. 5. Matlab and FPGA-based decoding results are compared with the measured pressure signals.

The decoding results of Matlab and FPGA-based implementation are compared with the measured pressure signals in Fig.5.The peaks correspond to the press action of the rat, and it can be seen obviously that all press actions are correctly predicted by our FPGA implemention of the KF algorithm. The correlation coefficient between Matlab decoded predictions and the real trajectory is 0.7725. The mean square error between FPGA and Matlab implementation results is only 2.3750×10^{-4}, which indicates the fidelity of FPGA.

C. Resource Utilization and Speed

Table I summarizes the resources utilization in the two HPMs implementations and Table II shows the corresponding computation time. We can find that both PNN and KF based FPGA realization make the computation speed improved by several dozen times. The 16-bit version of PNN dramatically reduces resources utilization while maintaining the same computation time as the 32-bit version.

TABLE I
RESOURCES UTILIZATION OF FPGA IMPLEMENTATION OF HPMS

Resources	Available	32-bit PNN	16-bit PNN	32- bit KF
Slice registers	393600	1%	1%	16%
Slice LUTs	196800	9%	7%	55%
LUT_FF pairs	24460	11%	18%	24%
Block RAM	704	1%	1%	45%
DSP48E	1344	13%	4%	42%

TABLE II
COMPUTATION TIME COMPARISON

Platform	32-bit PNN	16-bit PNN	32- bit KF
Matlab(Dual CPU @2.3GHz)	300	—	10,447
FPGA (@100MHz)	6.7	6.7	427

The unit of the time is μs.

V. DISCUSSION AND CONCLUSION

The two hardware processing modules (HPMs), PNN based spike sorting and Kalman based neural decoding, have been successfully realized on FPGA platform. They could achieve the same results as PC based and greatly reduce the computation time. The row/column based method proposed for accelerating matrix computation and the novel PE designed for saving hardware resources achieved good results. The lower precision implementation for spike sorting algorithm is also feasible and reduces FPGA resources utilization strikingly. With the features of re-configurability and scalability, these HPMs, as the coprocessors of the general purpose system, could greatly improve the overall performance of BMIs. Moreover, they are reusable and suitable for portable applications. Therefore, a wide adoption of FPGA in BMI systems will lead to more clinic applications of BMIs.

REFERENCES

[1] M. A. Lebedev and M. A. Nicolelis, "Brain-machine interfaces: past, present and future," *Trends Neurosci,* vol. 29, pp. 536-46, 1990-09-01 2006.

[2] Z. Li, J. E. O'Doherty, T. L. Hanson, M. A. Lebedev, C. S. Henriquez, and M. A. L. Nicolelis, "Unscented Kalman Filter for Brain-Machine Interfaces," *PLoS ONE,* vol. 4, p. e6243, 2009-07-15 2009.

[3] M. Aghagolzadeh, Z. Fei and K. Oweiss, "An implantable VLSI architecture for real time spike sorting in cortically controlled Brain Machine Interfaces," in *Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE*, 2010, pp. 1569-1572.

[4] S. Darmanjian, G. Cieslewski, S. Morrison, B. Dang, K. Gugel, and J. Principe, "A Reconfigurable Neural Signal Processor (NSP) for Brain Machine Interfaces," in *Engineering in Medicine and Biology Society, 2006. EMBS '06. 28th Annual International Conference of the IEEE*, 2006, pp. 2502-2505.

[5] K. Balasubramanian and I. Obeid, "Reconfigurable embedded system architecture for next-generation Neural Signal Processing," in *Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE*, 2010, pp. 1691-1694.

[6] A. Keuer, R. Schrott, J. Taube, D. Schmuck, H. Beikirch, and W. Baumann, "FPGA based time detection of spikes within neural signals," in *Sensor Array and Multichannel Signal Processing Workshop Proceedings, 2004*, 2004, pp. 186- 190.

[7] E. Biffi, D. Ghezzi, A. Pedrocchi, and G. Ferrigno, "Development and validation of a spike detection and classification algorithm aimed at implementation on hardware devices," *Comput Intell Neurosci,* p. 659050, 2010-01-20 2010.

[8] F. Zhou, J. Liu, Y. Yu, X. Tian, H. Liu, Y. Hao, S. Zhang, W. Chen, J. Dai, and X. Zheng, "Field-programmable gate array implementation of a probabilistic neural network for motor cortical decoding in rats," *Journal of Neuroscience Methods,* vol. 185, pp. 299-306, 2009.

[9] D. F. Specht, "Probabilistic neural networks and the polynomial Adaline as complementary techniques for classification," *IEEE Trans Neural Netw,* vol. 1, pp. 111-21, 1990-01-19 1990.

[10] Y. Yu, S. M. Zhang, H. J. Zhang, X. C. Liu, Q. S. Zhang, X. X. Zheng, and J. H. Dai, "Neural decoding based on probabilistic neural network," *Journal of Zhejiang University-Science B*, vol. 11, pp. 298-306, 2010.

[11] W. Wei, M. J. Black, D. Mumford, G. Yun, E. Bienenstock, and J. P. Donoghue, "Modeling and decoding motor cortical activity using a switching Kalman filter," *Biomedical Engineering, IEEE Transactions on,* vol. 51, pp. 933-942, 2004-01-01 2004.

[12] W. Wu, Y. Gao, E. Bienenstock, J. P. Donoghue, and M. J. Black, "Bayesian population decoding of motor cortical activity using a Kalman filter," *Neural Computation,* vol. 18, pp. 80-118, 2006.