Towards A Low Power FPGA Implementation for A Stand-Alone Intraoral Tongue Drive System

Sina Viseh[1], Abner Ayala-Acevedo[2], Maysam Ghovanloo[2] and Tinoosh Mohsenin[1]
[1]University of Maryland Baltimore County, [2]Georgia Institute of Technology

Abstract— Tongue Drive System (TDS) is a new assistive, unobtrusive, wireless, and wearable device that allows for real time tracking of the voluntary tongue motion in the oral space for communication, control, and navigation applications. The latest TDS prototype appears as a wireless headphone and has been tested in human subject trials. In this paper, we propose a low power FPGA implementation of the TDS. Implementing the computational engine on an IGLOO Nano FPGA reduces the volume of data that needs to be wirelessly transmitted per command by a factor of 24. As a result, the power consumption and size of the device will be significantly reduced. With this integration, the whole system can be implemented inside the mouth and operate longer with every recharge of the battery. iTDS can be used as an augmentative human motor output, like hands and fingers, and act like the 3rd arm, where physical motion has been hindered by the environment or by the nature of the task. Alternatively, by concealing visible motions, iTDS can be used for communication and control in stealth operations.

I. INTRODUCTION

Tongue motion as an untapped modality for motor function has the potential to serve as a control mechanism for the disabled. It can also be an additional resource for complex control during high workload situations for able-bodied individuals especially in battlefields and military missions. It is evident from the motor homunculus that the tongue and mouth occupy a considerable area of the motor cortex, comparable with that of the hand [1]. The tongue is connected to the brain over a shorter distance via the hypoglossal nerve, while the hand and fingers are connected through the spinal cord, a much longer neuronal extension. Considering these anatomical innervations and that the tongue's motion in the mouth is rapid, intuitive, dexterous, and does not require much concentration, the tongue seems to be quite appropriate as a control interface. To this end, a few tongue-operated assistive technologies (AT), such as the Tongue-Touch-Keypad [2], Jouse2 [3] and Integra Mouse [4], have been developed. However, these technologies are limited by their large size, requirements for specific head movement, and potential for causing fatigue.

Tongue Drive System (TDS) developed at GT-BIONICS LAB in Georgia Institute of Technology is an unobtrusive, minimally invasive, and wireless, tongue-operated AT that can potentially substitute some of the hand functions with tongue functions. The TDS architecture and performance by able-bodied subjects and those with severe physical disabilities (tetraplegia) have been previously evaluated and reported [6]-[7]. Fig. 1 shows one of the experimental setups for testing the concurrent arm and tongue movements and TDS function. The major findings of previous studies were: (1) the movement speed was slowed with the concurrent hand-tongue task for the hand speed, but not for the tongue speed; (2) the movement speed of the hand was not slowed with the concurrent hand-cognitive task, but the movement speed of the tongue was slowed with the concurrent tongue-cognitive task; and (3) the accuracy (correctness) of goal-directed movements was reduced with the concurrent hand-tongue task for both hand and tongue, especially when the task was difficult. These findings suggested that the speed of hand and tongue movements would be influenced in different ways by introducing additional motor control modality or cognitive task to individual hand or tongue tasks. The tongue control via TDS has the advantage of maintaining comparable speeds between independent and concurrent use over the hand movement control during the tasks that require rapid repetitive goal-directed movements [8].

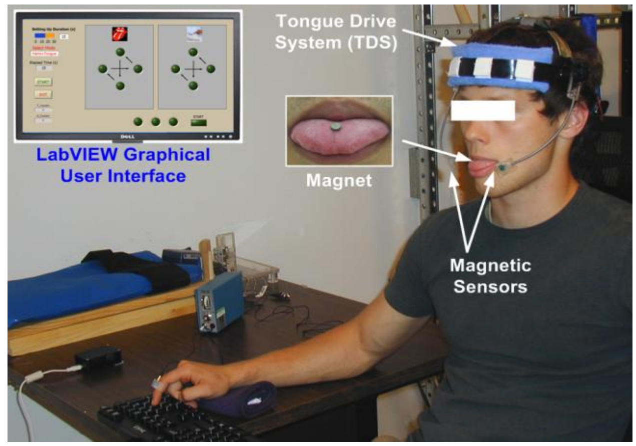

Fig. 1. A subject using TDS to perform tongue tasks and tapping a standard keyboard with his right index finger to perform finger tasks. He was presented by visual cues through a customized GUI (inset) on a 22" LCD monitor during the experiment [8].

So far, most studies on tongue motor abilities have been related to natural tongue functions such as respiration, speech, and swallowing [8], [9]. Training tongue with simple protrusion task was also reported to observe the neural plasticity [10]. However, none of them have studied the tongue performance in providing voluntary motor control. The ability of the TDS to incorporate tongue movements into performing tasks allows us to evaluate the efficacy of tongue motion as a voluntary motor modality on human performance in realistic environments as well as its human factors.

Fig. 2. Military applications of low power FPGA implementation for iTDS.

A TDS prototype is proposed in this paper in the form of an intraoral TDS (iTDS), which consists of four 3-D magnetic sensors, a processing FPGA, and a wireless control unit [11]. TDS detects user's tongue movements by sensing the changes in the magnetic field generated by a small magnetic tracer, the size of a lentil, attached to user's tongue using adhesives. In previous versions of TDS all raw data which has been sensed with magnetic sensors was sent to a smartphone where sensor signal processing was performed. This high volume wireless data transmission consumes significant amount of power and requires a bulky battery for operation. Unlike previous versions, the FPGA performs onboard analysis of the magnetic signals and only transmits minimal amount of data that is needed to deliver the detected command to the target. The FPGA takes magnetic field samples with a determined interval between each command and uses new signal processing and machine learning algorithms to determine movements of tongue and classify them into the appropriate commands. Implementing the computational engine on an ultra low power FPGA reduces the volume of data to be transmitted from 24 to 1 byte per command. As a result, it reduces the power consumption in the transmission, resulting in longer iTDS operating time with a single charge.

II. Proposed FPGA-based TDS Implementation

The high-level block diagram of the FPGA design is shown in Figure 3 which consists of three blocks, Serial Peripheral Interface (SPI), Electromagnetic Fields (EMF) attenuation, and a classifier. In the following, details of the design and optimization are described. The FPGA should be in charge of controlling communication among four sensors and processing engine while being sure that detected command is correctly delivered to target device.

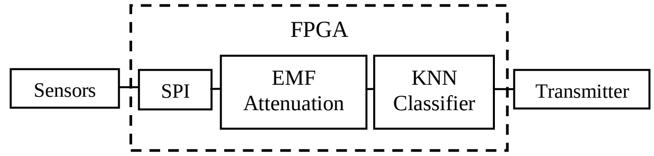

Fig. 3. Basic block diagram for the FPGA Implementation of the iTDS digital signal processing engine.

A. *Choosing the Right Classifier*

Although a considerable number of FPGA chips with large capacity can be found in the market on these days, however an FPGA with an ultra low power consumption and low area which suits for the iTDS purpose is very space limited. Thus the main challenge is to choose the right classifier which has a low misclassification rate while leaving enough space for the EMF cancelation and SPI blocks on FPGA. In a previous study [12], it is shown that if we divide classification process into two stages, the misclassification rate drops by 2%. For the two stage classifier, the first stage only classifies the command for left and right, and in the second stage we classify for the final decision. Among several classifiers that have been investigated including Mahalanobis, Diagonal Quadratic, Quadratic, KNN city block, KNN cosine, KNN correlation and KNN Euclidean, the KNN Euclidean classifier had a relatively low misclassification rate of 7.3% It also can fit in our target FPGA, IGLOO Nano AGLN250. With the optimized bit resolution that will be discussed in the following, the KNN Euclidean occupied almost 70% of our selected FPGA versus the Quadratic classifier which occupied almost double the size of an AGLN250.

B. *Data path Wordwidth Reduction and Bit Resolution Optimization*

Accurately presenting the biological signals has profound impact on final result, we should have enough number of bits to fully represent a biological signal. On the other hand, the datapath wordwidth of the processing blocks directly determines the required memory capacity, routing complexity, circuit area, and critical path delays. Moreover, it affects the amount of switching activity on wires and logic gates, and thus affecting the power dissipation. Figure 3 shows the impact of wordwidth bit resolution on the final misclassification rate after a two stage KNN classification. We start with a 12-bit input which has the same number of bits that we receive from ADC and is the most achievable accuracy with the error rate of 7.3%. Then we decreased the number of bits, as shown in the figure, the accuracy nearly stays the same with up to 9 bit input implementation. Similarly, for training data storage in lookup tables, arithmetic unit and a 12-bit training set(of 280 entries) requires 20,160 bits storage versus the same training set with 11 bits data word has 18,480 bits.
In the next experiment, assuming 7 bit integer data, the number of decimal bits in the EMF cancelation coefficients is optimized. Figure 4 shows the impact of the number of

decimal bits in EMF cancellation coefficients on final misclassification rate. As shown in the figure, using 2 decimal bits the error rate increases by 8.9% versus with 5 bits, the error rate is 7.3% which is similar to a floating point full precision implementation in MATLAB.

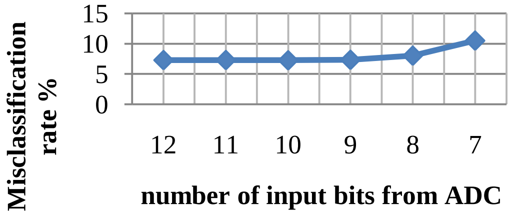

Fig. 4. Impact of decreasing the number of input bits on the misclassification rate

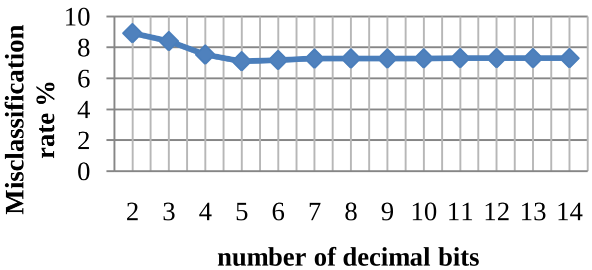

Fig. 5. Impact of the number of decimal bits in EMF cancelation's coefficients on Misclassification rate

C. Shared Processing

Solely relying on optimizing the classifier and datapath wordwidth does not provide enough space for all components to be implemented in the AGLN250 FPGA. Since the EMF cancellation unit and KNN classifier are both using an adder and a multiplier, with a well-designed architecture, these two resources can be shared by them in order to save space. As a result the next important consideration is to choose the right multiplier and adder's input size in a way that they can be used for both KNN classifier and EMF cancelation and without sacrificing accuracy. Figure 6 shows a high level block diagram of the FPGA design with shared resources. The state machine is in charge of issuing all control signals. Sensor data are read through the SPI interface and stored in an FIFO, the SPI also checks sensors' IDs to guarantee that the received data is not corrupted or there is no malfunction in the communication. After that the state machine issues required signals to carry The output of EMF cancelation is fed into the KNN classifier, which calculates the euclidean distance of data and training, finds and the the three shortest distance among all, and classifies the command based on the majority vote of three labels. . The final vote is transmitted to the microcontroller to be sent to the target device.

D. Implementation results

The design is fully placed and routed on IGLOO Nano AGLN250 and the implementation results are shown in Table. 1. As shown in the table, the total device utilization proves the importance of using resources wisely. In order to use the space in the best way and given that the SPI only works in the master mode with predefined operations, instead of using an IP core, we used our own simplified code which leads to 5% device utilization. Total logic usage is 98% percent of the AGLN250 FPGA.

Area			
	Used	Available	%
Core cells	6036	6144	98
CLKBUF	1	68	1.5
INBUF	2	68	3
OUTBUF	10	68	15
Total IO cells	13	68	19

Table 1. The post place and route results and design utilization on the AGLN 250 FPGA.

As mentioned before, the main reason of implementing signal processing on FPGA is to make ITDS operate longer with one single battery charge. Comparing with previous version, in which all raw data needs to be transmitted to another device for processing, ITDS with local processor only transmits final vote. The previous version [11] consumes 16.25 mW in transmission and ignoring analog parts with a 50 mA/h battery it can operates for 8.6 hours while the local processor with 2.709 mw power consumption (Table 2) and with the same battery can operates for 51.7 hours.

Power (mW)	
Static	0.046
Dynamic	2.663
Total	2.709

Table 2. Static, dynamic and total power consumption on AGLN250 with the operating frequency of 14 Mhz

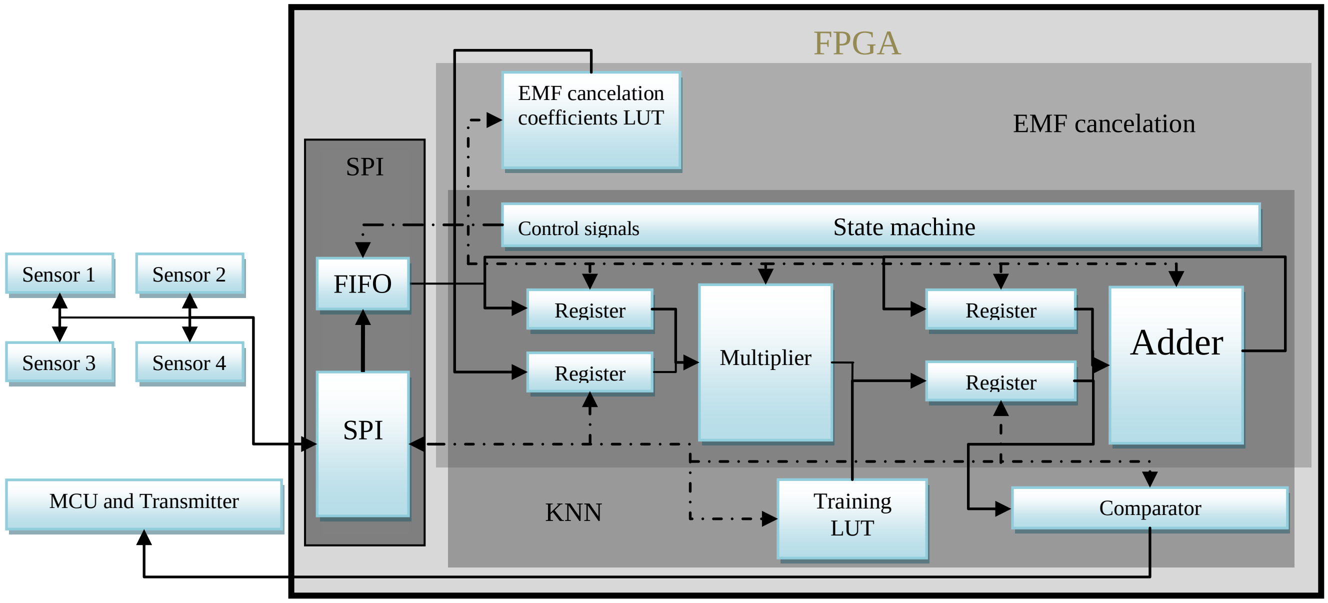

Fig. 6. Top level block diagram of Sensor Signal Processing engine unit and it's interconnection with sensors, transmitter and microcontroller

	Power(mw)	Percentage of being in active
Net	2.212	81.60%
Gate	0.153	5.60%
I/O	0.298	11.00%
Core Static	0.022	0.80%
Banks Static	0.025	0.90%

Table 3. Dynamic power consumption break down on different FPGA cores on AGLN250 with operating frequency of 14 Mhz.

III. CONCLUSION

This paper presents a local signal processor that reduces conventional tremendous data communication between ITDS and smartphone.

The processor consists of an SPI interface, the EMF cancelation and a KNN Euclidean classifier. For the FPGA implementation we used fixed point architecture in order to save resources and to find the most suited number of integer and decimal bits, some experiments have been carried out to find the optimum architecture with a reasonable error rate and appropriate complexity for AGLN250. Implementation results show a promising power consumption which means we can use a smaller battery instead of the conventional bulky battery and make the ITDS more user friendly. Analyzing detected values on a local processor and reduction in power consumption helps ITDS to operate for a longer period of time with a single battery charge and makes it an operational device that can be used as a third hand in the real world.

REFERENCES

[1] E.R. Kandel, J.H. Schwartz, and T.M. Jessell, *Principles of Neural Science*, 4th ed., New York: McGraw-Hill, 2000.

[2] TongueTouch Keypad™ (TTK), [Online]. Available: http://www.newabilities.com/

[3] Jouse2, Compusult Limited, [Online]. Available: http://www.jouse.com/

[4] USB Integra Mouse, Tash Inc., [Online]. Available: http://www.tashinc.com/catalog/ca_usb_integra_mouse.html

[5] X. Huo, and M. Ghovanloo, "Using unconstrained tongue motion as an alternative control mechanism for wheeled mobility" *IEEE Trans. Biomed. Eng.*, vol. 56, pp. 1719–1726, June 2009.

[6] X. Huo and M. Ghovanloo, "Evaluation of a wireless wearable tongue computer interface by individuals with high level spinal cord injuries," *Journal of Neural Engineering*, vol. 7, #026008, Mar. 2010.

[7] X. Huo and M. Ghovanloo, "Tongue Drive: A wireless tongue-operated means for people with severe disabilities to communicate their intentions," *IEEE Communications Magazine*, vol. 50, no. 10, pp. 128–135, Oct. 2012.

[8] A.N. Johnson, X. Huo, C.W. Cheng, M. Ghovanloo, and M. Shinohara, "Effects of additional load on hand and tongue performance," Proc. *IEEE 32nd Eng. in Med. and Biol. Conf.*, pp. 6611-6614, Sep. 2010.A. Sawczuk and K.M. Mosier "Neural control of tongue movement with respect to respiration and swallowing," *Crit Rev Oral Biol Med* vol. 12, 18–37 , Jan. 2001

[9] K. Bunton, and G. Weismer, "Evaluation of a reiterant force-impulse task in the tongue," *J Speech Hear Res*, vol. 37, pp 1020-1031, Oct. 1994.

[10] P. Svensson, A. Romaniello, K. Wang, L. Arendt-Nielsen and B.J. Sessle, "One_hour tongue-task training associated plasticity corticomotor control of the human tongue musculature", *Experimental Brain Research*, vol. 173, pp 165-173, Aug. 2006.

[11] H. Park, M. Kiani, H.M. Lee, J. Kim, B. Gosselin, and M. Ghovanloo, "A wireless magnetoresistive sensing system for an intraoral tongue-computer interface," *IEEE Trans. on Biomed. Circuits and Systems*, vol. 6, no. 6, pp. 571–585, Dec. 2012.

[12] Abner Ayala-Acevedo and Maysam Ghovanloo, Senior Member, IEEE " Quantitative Assessment of Magnetic Sensor Signal Processing Algorithms in a Wireless Tongue-Operated Assistive Technology"