Toward an Ultralow-Power Onboard Processor for Tongue Drive System

Sina Viseh, *Student Member, IEEE*, Maysam Ghovanloo, *Senior Member, IEEE*, and Tinoosh Mohsenin, *Member, IEEE*

***Abstract*—The Tongue Drive System (TDS) is a new unobtrusive, wireless, and wearable assistive 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. However, the robustness of the external TDS (eTDS) in real-life outdoor conditions may not meet safety regulations because of the limited mechanical stability of the headset. The intraoral TDS (iTDS), which is in the shape of a dental retainer, firmly clasps to the upper teeth and resists sensor misplacement. However, the iTDS has more restrictions on its dimensions, limiting the battery size and consequently requiring a considerable reduction in its power consumption to operate over an extended period of two days on a single charge. In this brief, we propose an ultralow-power local processor for the TDS that performs all signal processing on the transmitter side, following the sensors. Assuming the TDS user on average issuing one command/s, implementing the computational engine reduces the data volume that needs to be wirelessly transmitted to a PC or smartphone by a factor of 1500×, from 12 kb/s to ∼8 b/s. The proposed design is implemented on an ultralow-power IGLOO nano field-programmable gate array (FPGA) and is tested on AGLN250 prototype board. According to our post-place-and-route results, implementing the engine on the FPGA significantly drops the required data transmission, while an application-specific integrated circuit (ASIC) implementation in a 65-nm CMOS results in a 15× power saving compared to the FPGA solution and occupies a 0.02-mm^2 footprint. As a result, the power consumption and size of the iTDS will be significantly reduced through the use of a much smaller rechargeable battery. Moreover, the system can operate longer following every recharge, improving the iTDS usability.**

***Index Terms*—Application-specific integrated circuit (ASIC), field-programmable gate array (FPGA), low power, machine learning, onboard processor, personalized assistive device, wearable biomedical device.**

I. Introduction

TONGUE motion as an untapped modality for motor function has the potential to serve as a control mechanism for disabled individuals. To this end, a few tongue-operated assistive technologies (ATs), such as the TongueTouch Keypad [1], Jouse2 [2], and Integra Mouse [3], 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) is an unobtrusive,

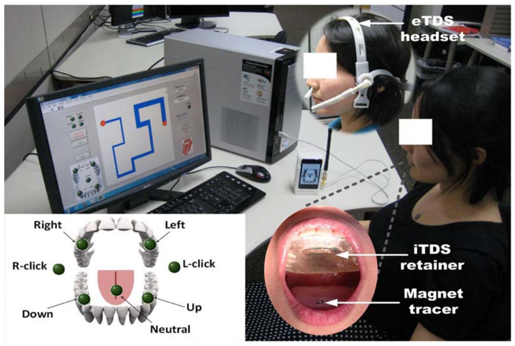


Fig. 1. Test setup for the maze navigation task with both iTDS dental retainer and eTDS headset. Inset: six recommended command positions plus neutral.

minimally invasive, wireless, and wearable tongue-operated AT that can potentially substitute some of the hand functions with voluntary tongue motion [5], [6]. The TDS architecture and performance by able-bodied subjects and those with severe physical disabilities (tetraplegia) have been previously evaluated and reported [6]. Prior versions of the TDS are categorized into two main groups: intraoral (iTDS) and external (eTDS). The iTDS is in the shape of a dental retainer, which is placed inside the mouth, whereas the eTDS is in the shape of a headset. Fig. 1 shows both eTDS and iTDS in the test setup and six recommended commands.

In all previous TDS versions, the raw sensed data are transmitted to a PC or a smartphone, to perform all signal processing and command classification. This large amount of data communication results in high power consumption, and consequently, the device needs a bulky battery. Taking advantage of a local processor that can perform all signal processing at the sensors instead of sending raw data will significantly lower the transmission power consumption and subsequently increase the operating time or alternatively abate the battery size [7]–[9]. In this brief, we propose to implement a local processor for the current version of the TDS. Similar to the previous versions, this prototype consists of four off-the-shelf three-axial magnetic sensors; a dual-band wireless transmitter, which uses OOK modulation; and the local processor. The sensors, analog front end, and transmitter have been extensively explained, in detail, in an earlier publication [10]. The local processor communicates with sensors and transmitter through a serial peripheral interface (SPI) and a 3-bit parallel interface, respectively. TDS detects a user's tongue movements by sensing the changes in the magnetic field generated by a small magnetic tracer, i.e., the size of a lentil, attached to the user's tongue using adhesives. Unlike previous versions, our custom low-power processor performs onboard analysis of the raw magnetic sensor signals and only transmits minimal amount of processed data that are needed to deliver the detected command to the PC/smartphone. The processor takes the magnetic field samples within a

Manuscript received June 27, 2014; revised September 5, 2014 and October 21, 2014; accepted November 18, 2014. Date of publication January 6, 2015; date of current version February 7, 2015. This brief was recommended by Associate Editor R. Rieger.

The authors are with the University of Maryland, Baltimore County, Baltimore, MD 21250 USA (e-mail: tinoosh@umbc.edu).

Color versions of one or more of the figures in this brief are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSII.2014.2387683

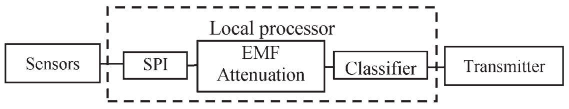


Fig. 2. Basic block diagram for the local processor.

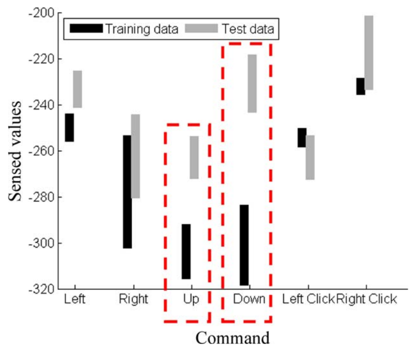


Fig. 3. Range of training and test data in one single feature for one of the patient data sets after using the linear regression model to attenuate the earth electromagnetic impact.

predetermined interval between each command through the SPI. Afterward, the processor uses signal-processing and machine-learning algorithms to determine movements of the tongue and classify them into the appropriate commands. The selected command is sent to the transmitter via a 3-bit parallel interface. Implementing the local processor reduces the volume of data to be transmitted from 12 kb/s to only 8 b/s, i.e., by a factor of 1500×. As a result, it reduces the power consumption in the transmission, resulting in a lighter and smaller device and consequently user comfort and longer TDS operating time with every recharge. Fig. 2 shows a high-level block diagram of the proposed local processor, which consists of SPI, Earth's magnetic field (EMF) attenuation, and classifier blocks [11].

The rest of the brief is organized as follows: Section II describes existing issues in eTDS. Section III goes over implementation challenges, classifier selection, training data, and hardware optimization. Sections IV and V present field-programmable gate array (FPGA) and application-specific integrated circuit (ASIC) implementations of the proposed processor, respectively. Finally, Section VI concludes the brief.

II. Why iTDS?

In previous work, the robustness of the eTDS in real-life conditions could not be guaranteed because of the limited mechanical stability of the headset. In addition, the classification error with basic machine-learning algorithms could not be lowered than 4.8% even by taking the majority vote among nine different classifiers, which is not area and energy efficient for hardware implementation [12]. This section investigates the cause of the errors in eTDS, starting by comparing the range of training and test data. We use the same 40-trial data set used in previous work for evaluation. Fig. 3 shows the range of both training and test data for one feature of six different commands in one of the patient's data sets that has a 55.17% misclassification error. As shown in the plot, the range of training and test data is very different in some commands, which are highlighted with dashed rectangles. This difference could be due to the residual electromagnetic field from the EMF attenuation algorithm or sensors' movement between test and train sessions. In the current noise cancelation method, a set of

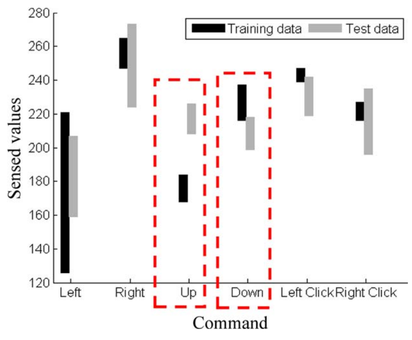


Fig. 4. Range of training and test data in one single feature for one of the patient data sets after using subtraction approach to attenuate earth electromagnetic impact.

precollected samples is processed by linear regression to derive four coefficients to approximate the EMF. The approximated EMF is subtracted from all subsequent sensed values to eliminate external noise for each axis of the sensed value [13]. To reduce the EMF approximation error, in this brief, we used one sensor in the rear as a reference to be subtracted from two front sensors. The notion behind this approach is that the sensed value from each sensor can be modeled as the summation of data and noise (1). Subsequently, subtraction of reference value from another sensor value results in the elimination of noise (2). That is

$$\begin{aligned} X_{\text{Left front}} &= X_{\text{data}_{\text{Left front}}} + X_{\text{Noise}} \\ X_{\text{Right front}} &= X_{\text{data}_{\text{Right front}}} + X_{\text{Noise}} \qquad (1) \\ X_{\text{Ref.}} &= X_{\text{data}_{\text{Ref.}}} + X_{\text{Noise}} \\ X_{\text{Left front}} - X_{\text{Ref.}} &= X_{\text{data}_{\text{Left front}}} - X_{\text{data}_{\text{Ref.}}} \\ X_{\text{Right front}} - X_{\text{Ref.}} &= X_{\text{data}_{\text{Right front}}} - X_{\text{data}_{\text{Ref.}}}. \qquad (2) \end{aligned}$$

Fig. 4 shows the range of results from the subtraction approach for the same patient and data set shown in Fig. 3. Results from the subtraction approach do not completely eliminate the range difference in the commands shown in Fig. 3. One possibility is that the difference is caused by sensor displacement. We tried to address the problem from a domain change perspective, including normalizing both training and test data, which did not improve the results. In the next experiment, the training and test data were swapped (i.e., labeled test and train data), and the results showed that even a simple classifier can discriminate them with 99% accuracy. These experiments indicate that the range difference in eTDS is mostly caused by the displacement and movement of the headset, during training and real-life conditions, and emphasize the importance of the iTDS design, which is potentially more mechanically stable and accurate, as it is customized to clasp onto the user's teeth.

III. Proposed Low-Power Onboard Processor

A. Challenges

In pursuit of a reduction in power consumption and size of the iTDS, the proposed processing algorithm should be small enough to fit in an ultralow-power FPGA, and its ASIC implementation should result in a small footprint. Furthermore, similar to other applications [14]–[16], applied hardware optimizations should not negatively impact the accuracy of the algorithm and the overall device reliability. Consequently, classifier, data-path word width design, resource allocation, and training data size should be arranged in a manner to satisfy all

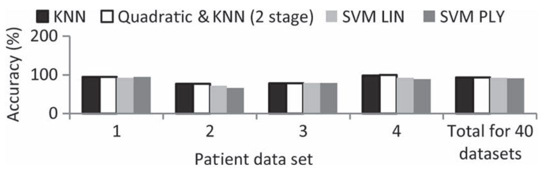


Fig. 5. Command detection accuracy comparison of four different classifiers for sample data sets and the total average for 40 data sets.

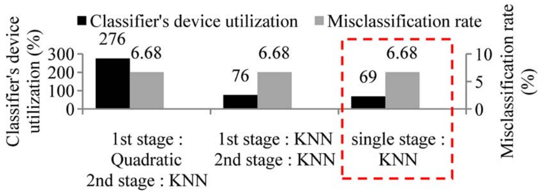


Fig. 6. Comparison between device utilization on AGLN250 and accuracy in single-stage and two-stage classification algorithms. Dashed rectangle represents the chosen algorithm for this brief.

the restrictions. We initially designed the prototype on a small and low-power FPGA and then implemented it on an ASIC to be fabricated in a 65-nm CMOS process. At the time of this publication, the only available FPGA in the market, which can meet our design criteria, was the AGLN250 IGLOO nano that has a 6 × 6 mm^2 package.

B. Classifier Algorithm Selection

Among different classification algorithms and configurations, such as single- and double-stage classifiers, that we have explored [12], a single-stage k-nearest neighbor (KNN) Euclidean, a support vector machine (SVM) with linear and polynomial kernels, and a two-stage algorithm with quadratic classifier as the first stage and KNN Euclidean as the second stage have reasonable accuracy. Fig. 5 shows the accuracy of four different classification algorithms in four data sets and the average accuracy through all 40 data sets. Although none of the four presented classifiers has a consistent accuracy advantage across all data sets, the KNN Euclidian and the two-stage quadratic and KNN Euclidean with 93.32% accuracy are the most accurate. The SVM with linear and polynomial kernels provided 92.83% and 91.14% accuracy, respectively. We also compared the device utilization of classifiers on an AGLN250 IGLOO FPGA.

Fig. 6 shows the accuracy and device utilization of single-stage and two-stage algorithms with different combinations of KNN and quadratic classifiers. Considering both of these factors, since the classification problem in this application is relatively uncomplicated, a single-stage KNN Euclidean classified the data set that was available to us with the same accuracy as a two-stage classifier, while utilizing four times less resources on AGLN250. The remaining inaccuracy is mainly due to the residual EMF and sensor displacements, as discussed in Section II.

C. Training Optimization

Additionally, we investigated the impact of the number of training samples representing every single command in a training set on both accuracy and hardware complexity. Fig. 7(a) shows the impact of training samples on accuracy. As shown in this plot, using less than ten samples results in considerable inaccuracy. However, the accuracy reaches an acceptable level

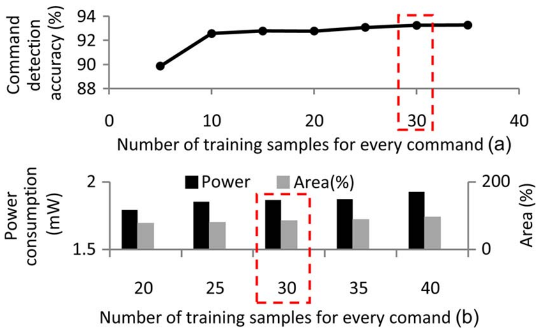


Fig. 7. Impact of increasing the training size (a) on the command detection and (b) on the post-place-and-route power consumption and area of the design on AGLN250, with the optimal value of 30 training samples, which is used for the design in this brief.

at 25 samples per command. The accuracy plot flattened after 30 samples and reached 93.32%. The bar plots in Fig. 7(b) show the impact of increasing training size on the post-place-and-route power consumption and area, respectively. Reducing training samples can result in a considerable reduction in device utilization, which reduces by 10% by going from 40 to 30 samples per command.

D. Data-Path Word Width Reduction and Bit Resolution Optimization

The data-path word width of the processing blocks in the proposed architecture 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, thus affecting the power dissipation. In order to measure the impact of word length on design size, ignoring the arithmetic unit and considering only the training lookup table (LUT), a training set with 12-bit data word has 20 160 bits (6 features × 12 bits × 40 samples × 7 commands), whereas the same training set with 11-bit data word has 18 480 bits. The proposed architecture uses all 12 bits of the analog-to-digital converter (ADC) to make the system resistant to signal saturation. Moreover, it has eight decimal bits fixed point for EMF attenuation coefficients. Further analysis has been carried out to assure that the aforementioned fixed-point structure does not impact the accuracy.

E. Shared Processing

Solely relying on optimizing the classifier and data-path word width does not provide enough space for all components to be implemented in the AGLN250 FPGA. Thus, with a careful design, the EMF attenuation unit and KNN classifier are using a shared adder, multiplier, and registers to save space. As a result, the next important consideration is to choose the right multiplier and adder input lengths in a way that they can be used for both KNN classifier and EMF cancelation without affecting accuracy. Fig. 8 shows a high-level block diagram of the proposed design. The minimum word width (in bits) used for the main signals is shown in the figure. The local processor reads all four sensors' values with 50 samples/s through the SPI serial interface. Each sensor reads three-axis values (X, Y, and Z), 12 bits each. The EMF attenuation coefficients and sensor values in the back are used to estimate the interference according to (3). The precalculated coefficients (explained in Section II) are stored in the EMF Attenuation Coefficient LUT.

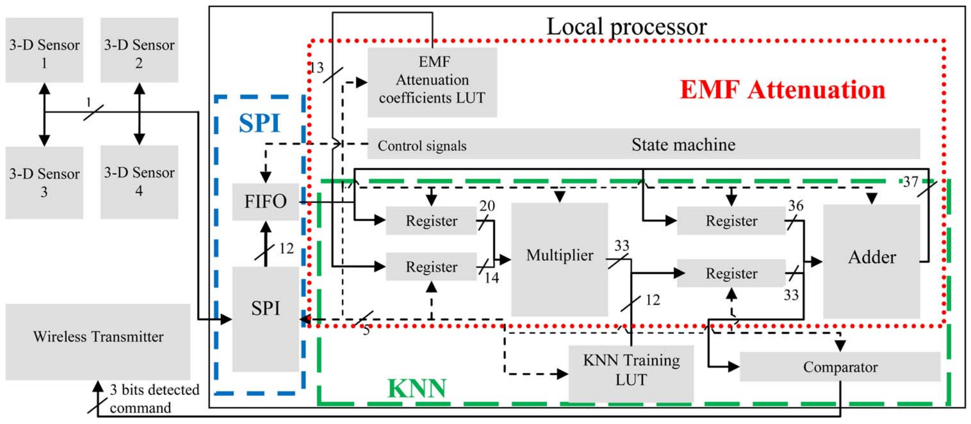


Fig. 8. Top-level block diagram of the proposed local processor and its interconnection with wireless transmitter and three-axial sensors. Dashed lines represent the control signals from state machine.

The estimated interference is subtracted from sensed values to eliminate noise for each axis of the sensed value [13]. Next, the result is used to find three minimum Euclidean distances from all training samples, according to (4); here, we use squared distance. That is

$$X_{\text{Test}_1} = X_{\text{Sensed}} \times \text{Coeff.}_{X_1} + \text{Y}_{\text{Sensed}} \times \text{Coeff.}_{Y_1} + Z_{\text{Sensed}} \times \text{Coeff.}_{Z_1} + \text{Coeff.}_d \quad (3)$$

$$d_n = \left(X_{\text{train}_{1_n}} - X_{\text{test}_1}\right)^2 + \cdots + \left(Z_{\text{train}_{2_n}} - Z_{\text{test}_2}\right)^2. \quad (4)$$

After going through all the training data in the KNN Training LUT, final vote is generated by taking the majority vote on the labels of the three minimum distances. The final vote for each command is transmitted to the PC/smartphone. The *state machine* is in charge of issuing all control signals to carry out the EMF interference estimation and KNN Euclidean distance calculations. The calculations are performed in a fully serial workflow to save resources, and it takes almost 12 500 clock cycles to process a received command. The EMF Attenuation Coefficient LUT is 480 bits (4 coefficients × 20 bits × 3 axis × 2 sensors), and the KNN Training LUT is 15 120 bits (6 features × 12 bits × 30 samples × 7 commands).

IV. FPGA Implementation and Results

Table I and Fig. 9 show the power breakdown of post-place-and-route implementation on AGLN250. As shown in this figure, nearly 81% of power is consumed in the FPGA interconnection logic. In the next section, we show the impact of ASIC implementation on reducing the interconnection and overall power consumption. Table II shows the device utilization on the AGLN250 FPGA. Discussed architecture with a training data set, which has 30 samples representing each command, utilizes 88% of the device. Fig. 10 shows the device utilization breakdown. The majority of the resources are utilized by the LUTs, which are 20 kb in total.

V. ASIC Implementation and Results

To reduce the overall power consumption, we used a standard-cell register-transfer level (RTL) to Graphic Data System (GDSII) flow using synthesis and automatic place-and-route. The hardware was designed using Verilog to describe the architecture, synthesized with Synopsys Design Compiler, and placed and routed using Cadence SOC Encounter. Fig. 11 shows the layout of the iTDS local processor in a 65-nm Taiwan Semiconductor Manufacturing Company Ltd. (TSMC) CMOS technology. White dashed rectangles indicate the multiplier, adder, LUTs, SPI, first-in first-out circuit (FIFO), and state machine. The processor is able to operate at 900-MHz clock frequency. However, the clock frequency has been reduced to 10 MHz abate the power consumption. The ASIC implementation reduces the

TABLE I
Power Consumption of Fully Implemented Design on AGLN250 at the Operating Frequency of 10 MHz

	Power (mw)	% of being active
Network dynamic	1.504	81.60%
Gate dynamic	0.107	5.60%
I/O dynamic	0.211	11.00%
Core Static	0.022	0.80%
Banks Static	0.025	0.90%
Total static	0.046	-
Total dynamic	1.821	-
Total	1.868	-

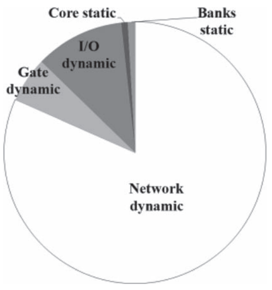


Fig. 9. Power consumption breakdown of the proposed design.

TABLE II
Proposed Design on FPGA and Its Utilization on AGLN250

Area utilization			
	Used	Available	%
Core cells	5377	6144	88
CLKBUF	1	68	1.5
INBUF	2	68	3
OUTBUF	10	68	15
Total IO cells	13	68	19

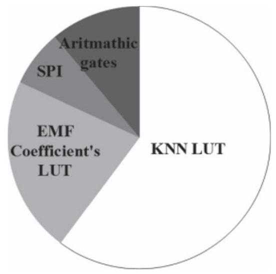
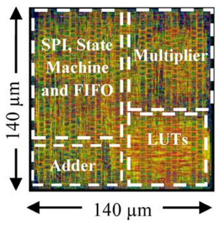


Fig. 10. Device utilization breakdown of the proposed design.



LAYOUT SYNTHESIS RESULTS

Technology	65 nm, 1 V CMOS
Processor area (mm^2)	0.02
Frequency (MHz)	10
Latency to detect (mSec)	20
Power (mW)	0.128

Fig. 11. Layout view of the iTDS local processor in a 65-nm CMOS and its specifications.

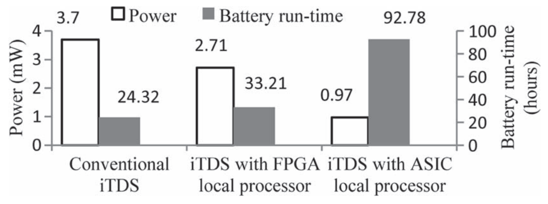


Fig. 12. Power consumption and estimated battery runtime comparison among the conventional iTDS and the iTDS with local processor with a 50-mAh battery.

processor's power consumption by a factor of 15, and voltage reduction decreases the power consumption by another factor of 2. Fig. 12 shows the comparison between power consumption and estimated battery runtime in the conventional iTDS and the proposed iTDS with FPGA and ASIC local processors utilizing a 50-mAh battery. Compared to the conventional iTDS, the total power consumption of the proposed iTDS with the local processor is reduced by 27% and 61% in FPGA and ASIC, respectively.

VI. CONCLUSION

The limited mechanical stability of the eTDS makes the iTDS an attractive option for daily life use. However, the impractical battery lifetime of iTDS leads us into designing a local processor that performs all signal processing and command detection locally before wireless transmission. The original TDS transmits all the raw data to an external computer to detect a command, while a TDS with a local processor only transmits the detected command, thus scaling down the data transmission by a factor of 1500×. The local processor design should be small enough to fit in our target ultralow-power FPGA and a small footprint ASIC. Furthermore, it should be accurate enough not to impact the TDS reliability. The FPGA implementation results show a considerable reduction on data communication, from 12 kb/s to 8 b/s. Implementation of the design on the AGLN250 FPGA results in 1.86-mW power consumption. The amount of power, which is consumed on network interconnect logic of the FPGA, directs us toward ASIC implementation. The ASIC implementation in a 65-nm CMOS technology reduces the power consumption by a factor of 15 when operating at 1 V, and by the factor of 30 when operating at 0.7 V, as compared to the FPGA prototype. The ASIC footprint occupies 0.02 mm^2, and it satisfies the required command detection latency of 20 ms when running at 10-MHz clock frequency. According to the post-place-and-route results, implementing the engine on the FPGA significantly drops the required data transmission and saves up to 13% power consumption in transmission and 27% in total power consumption by removing the microcontroller, which consumes 2.4 mW. In ASIC, the total power consumption reduces by 61% compared to the conventional iTDS. The FPGA and ASIC processors respectively occupy 4.11% and less than 1% of the iTDS's 25 mm × 35 mm printed circuit board (PCB). This optimization has a profound impact on the future editions of the iTDS, and it can eliminate the need for a bulky battery and increase the operating time following every recharge. This is expected to reduce the iTDS size and improve the user's comfort level and the system command detection accuracy.

REFERENCES

[1] TongueTouch Keypad (TTK). [Online]. Available: http://www.newabilities.com/
[2] Jouse2, Compusult Limited. [Online]. Available: http://www.jouse.com/
[3] USB Integra Mouse, Tash Inc. [Online]. Available: http://www.tashinc.com/catalog/ca_usb_integra_mouse.html
[4] 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," *Exp. Brain Res.*, vol. 173, no. 1, pp 165–173, Aug. 2006.
[5] X. Huo and M. Ghovanloo, "Using unconstrained tongue motion as an alternative control mechanism for wheeled mobility," *IEEE Trans. Biomed. Eng.*, vol. 56, no. 6, pp. 1719–1726, Jun. 2009.
[6] X. Huo and M. Ghovanloo, "Evaluation of a wireless wearable tongue computer interface by individuals with high level spinal cord injuries," *J. Neural Eng.*, vol. 7, no. 2, p. 26008, Mar. 2010.
[7] T. Mohsenin and A. Page, "Towards a low power wearable personalized seizure detection system," in *Proc. IEEE EMBS Brain Grand Challenges*, Nov. 2014, to be published. [Online]. Available: http://gcbme.embs.org/brain2014/program/poster-list/
[8] K. Li, S. Warren, and B. Natarajan, "Onboard tagging for real-time quality assessment of photoplethysmograms acquired by a wireless reflectance pulse oximeter," *IEEE Trans. Biomed. Circuits Syst.*, vol. 6, no. 1, pp. 54–63, Feb. 2012.
[9] H. Kim *et al.*, "A configurable and low-power mixed signal SoC for portable ECG monitoring applications," *IEEE Trans. Biomed. Circuits Syst.*, vol. 8, no. 2, pp. 257–267, Apr. 2014.
[10] H. Park *et al.*, "A wireless magnetoresistive sensing system for an intraoral tongue-computer interface," *IEEE Trans. Biomed. Circuits Syst.*, vol. 6, no. 6, pp. 571–585, Dec. 2012.
[11] S. Viseh, A. Acevedo, M. Ghovanloo, and T. Mohsenin, "Towards a low power FPGA implementation for a stand-alone intraoral tongue drive system," in *Proc. 39th Annu. GOMACTech Conf.*, Apr. 2014, pp. 1–4.
[12] A. Ayala-Acevedo and M. Ghovanloo, "Quantitative assessment of magnetic sensor signal processing algorithms in a wireless tongue-operated assistive technology," in *Proc. IEEE EMBC*, 2012, pp. 3692–3695.
[13] X. Huo, J. Wang, and M. Ghovanloo, "A wireless tongue-computer interface using stereo differential magnetic field measurement," in *Proc. 29th Annu. Int. IEEE EMBS*, Aug. 2007, pp. 5723–5726.
[14] A. Page, J. T. Turner, T. Mohsenin, and T. Oates, "Comparing raw data and feature extraction for seizure detection with deep learning methods," in *Proc. 27th Int. Conf. FLAIRS*, 2014, to be published. [Online]. Available: https://www.aaai.org/ocs/index.php/FLAIRS/FLAIRS14/rt/captureCite/7885/7853/BibtexCitationPlugin
[15] A. Page *et al.*, "A flexible multichannel EEG feature extractor and classifier for seizure detection," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 62, no. 2, pp. 109–113, Feb. 2015.
[16] A. Kulkarni, H. Homayoun, and T. Mohsenin "A parallel and reconfigurable architecture for efficient OMP compressive sensing reconstruction," in *Proc. 24th Annu. GLSVLSI*, pp. 299–304.