A Compact Low-Power VLSI Architecture for Real-Time Sleep Stage Classification

Peter Zhi Xuan Li[1], Hossein Kassiri[1], Roman Genov[1]
[1] Department of Electrical and Computer Engineering, University of Toronto, Canada

***Abstract*—A wearable-optimized implementation of a sleep stage classification algorithm that has low detection latency, high detection accuracy and low resource consumption is developed and successfully implemented on a low-power FPGA microsystem for closed-loop electrical brain stimulation. This implementation uses EEG and EMG signals as inputs and classifies stages of sleep. By structurally merging multichannel FIR and window averaging filters into one reconfigurable, multipurpose filter, the new implementation maintains a sleep detection accuracy of 79.7%, a REM detection sensitivity of 98.2%, a REM detection specificity of 89.2% and a detection latency of 0.982 ms, while consuming 6.8 times fewer logic elements and 96.28% less power compared with the current state of the art implementation. With its high performance and low resource usage, this implementation enables a low-power wearable microsystem to perform neural recording, real-time REM sleep stage detection, and closed-loop responsive brain stimulation as a tool to study the mechanisms of neurodegenerative diseases.**

I. Introduction

Neurodegenerative diseases affect millions of people around the world. Electrical brain stimulation has shown to be an effective, low-side-effect option for treating neurodegenerative disorders compared with many conventional pharmaceutical methods [1]. In the case of Alzheimer's disease (AD), epidemiological studies have discovered a correlation between the development of AD and stages of sleep, which is dominated by cycles of REM (rapid eye movement) that disrupts memory consolidation and NREM (non-REM) sleep that promotes memory consolidation [1], [2]. Studies found that stimulating the lateral hypothalamus at the peaks of θ-oscillation (5-10 Hz) significantly increases the chances of suppressing REM sleep [3]. This motivates the development of a sleep stage classifier with high accuracy and low detection latency ($<$ 1 ms) in order to deliver such timely stimuli by responsive stimulation.

Several software-based algorithms have already been developed for classifying sleep stages [4], [5]. Although these algorithms often achieve high detection accuracy, they require significant amounts of computational power, which prevents their implementation on low-power, wearable devices. Additionally, sending the recorded signal from a data acquisition module to a computer creates long delays that prevent responsive stimulation for REM suppression. These shortcomings motivate the design for a digital implementation on a FPGA-controlled wearable device capable of neural recording, real-time REM sleep stage detection, and close-loop brain stimulation (Fig 1, top).

Recently, several hardware based REM detection algorithms have been developed [7], [8]. In these algorithms, multiple bio-signals, such as electroencephalogram (hippocampus and cortex EEG) and electromyogram (EMG) signals in [7], are first bandpass-filtered and averaged (Fig. 2). Then, a sleep stage can be determined via thresholding because θ-oscillations from filtered hippocampus (5-10 Hz), Δ-oscillation from filtered cortex EEG (0-4 Hz), and EMG oscillations (100-200 Hz) are most prevalent in REM, NREM and AWAKE stages, respectively. Although these algorithms have been shown to achieve high REM classification accuracy with low detection latency, their high power and resource consumption make them unsuitable for implementing on low-power ($<$ 1 mW) wearable devices.

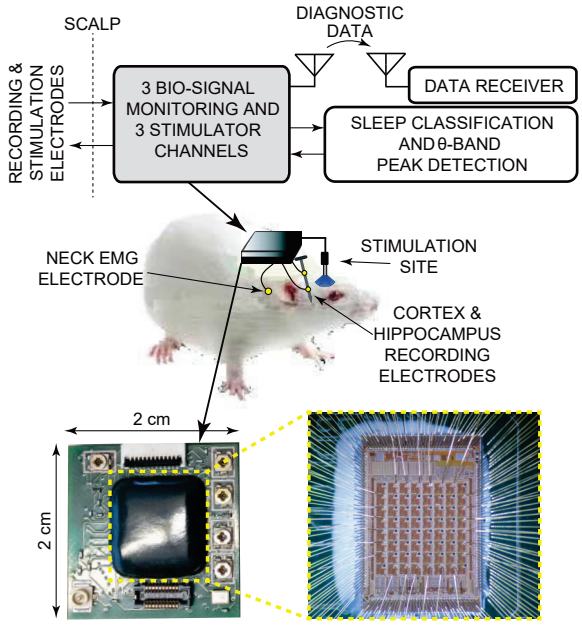


Fig. 1: A wearable system for sleep stage detection and responsive REM stage suppression. The integrated circuit was first reported in [6].

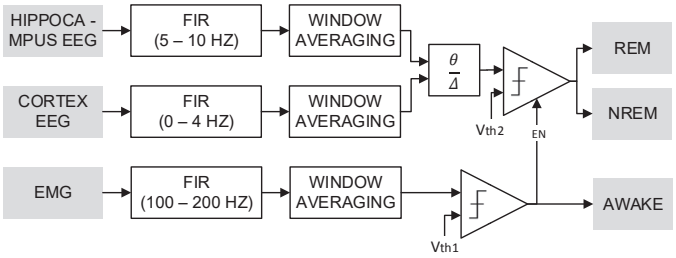


Fig. 2: Existing REM detection algorithm [7].

In our research, we have investigated the best existing hardware based REM detection algorithm [7] in order to identify causes for its high FPGA logic element (LE) and power usage. Then, an optimized implementation of the algorithm is proposed to significantly reduce its power and resource consumption. This implementation is validated with offline recordings from nine mice on a low-power, low-resource Actel AGL060 FPGA (1536 logic elements) that controls a wearable ASIC (Fig. 1, bottom). These recordings consist of icEEG and EMG signals that were collected with a 16 kHz sampling

978-1-4799-5341-7/16/$31.00 ©2016 IEEE

TABLE I: Comparison of logic element usage between the recently reported implementation [7] and the wearable-optimized implementation on an Actel IGLOO FPGA.

Module	Quantity	Total LE Consumption	% of Available Resources
Recently Reported Implementation [7]			
Signal acquisition	3	240	15.62%
FIR filter	3	6130	399.09%
Window averaging filter	3	2022	131.64%
Thresholding & classification	1	589	38.35%
Control logic	1	50	3.26%
Total	**11**	**9031**	**587.96%**
Wearable-optimized Implementation			
Signal acquisition	1	80	5.21%
FIR & window averaging filter	1	835	54.36%
Thresholding & classification	1	277	18.03%
Peak detection	1	49	3.19%
Control logic	1	76	4.95%
Total	**5**	**1317**	**85.74%**

rate using a headstage pre-amplifier with corresponding sleep stages manually scored by an expert [7]. After validation, the resource consumption, detection accuracy and detection latency of the optimized implementation on its FPGA are compared with the current state of the art implementations.

The rest of the paper is organized as follows: Section II outlines algorithmic design specifications and provides a resource utilization analysis. Section III highlights innovations in the optimized implementation that significantly reduces FPGA resource consumption. Section IV compares the performance of the new implementation with the current state of the art implementations.

II. RESOURCE ULTILIZATION ANALYSIS

The breakdown of LE usage in Actel IGLOO FPGA for a recently reported high-performance sleep stage classifier is shown in the upper half of Table I [7]. Optimum algorithmic specifications and main causes for their high-count LE implementation are discussed below.

A. Parallel Filtering of Input Signals

FIR filters with a minimum of 64 taps are chosen for all three channels of signals (cortex EEG, hippocampus EEG, EMG). Additionally, a maximum sampling frequency of 200 Hz and 800 Hz is chosen for filtering EEG and EMG signals respectively. Based on Matlab simulations, these filter specifications provide sufficient magnitude attenuation outside the corresponding bandpass frequency range for each signal. Furthermore, a window size of 10 seconds for the window averaging filter is chosen for all three channels to further improve sleep classification accuracy (Fig. 3).

The implementation in [7] requires three channels of signals to be processed separately by their dedicated FIR and window averaging filters, which are preconfigured with channel specific filtering parameters. Although parallel filtering shortens signal processing time, it consumes a significant amount of resources.

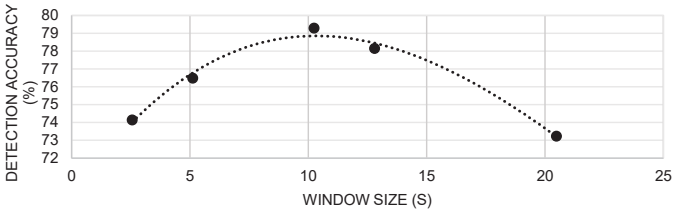


Fig. 3: Matlab simulation result of sleep stage classification algorithm: detection accuracy vs. window size of the window averaging filter.

B. Storage of Filter Data and Coefficients

Each FIR and window averaging filter needs to store and retrieve input data and filtering coefficients. For instance, each FIR filter requires the storage of 64 data inputs with its 64 corresponding filtering coefficients. Additionally, each window averaging filter needs to store and retrieve 512 data inputs.

Since the implementation in [7] of the FIR and window averaging filters stores data and coefficients in registers, a substantial amount of logic elements in the FPGA are consumed.

C. Integer Division

In an Actel FPGA, the integer division between two 8-bit unsigned numbers consumes 550 logic elements. This leads to high LE consumption in the thresholding module, which uses such division.

III. COMPACT VLSI ARCHITECTURE

Due to their high resource utilization, sharing the FIR and window averaging filters among the three channels significantly reduces the number of logic elements. It involves algorithmic structural sharing and low resource integer division, which are discussed below.

A. FSM Control of Time-Multiplexed Module Execution

Sharing the FIR and averaging filters requires time-multiplexing them among different channels. This is possible as the filter is clocked at 1.7 MHz, which is much faster than the highest required sampling frequency of 800 Hz (for the EMG channel).

Since the shared filter can only process one channel of signal at a time, a finite state machine (FSM) is needed to select the appropriate channel and trigger the execution of different data processing modules along the signal path. Major blocks inside such FSM are shown at the top of Fig. 5. At the rising edge of the 800 Hz sampling clock, the timing control block enables the deserializer module to convert the serialized EMG signal into a parallel 8-bit signal. Once the data is deserialized, it is filtered by the bandpass FIR and then by the window averaging filter, both implemented in the same module. The module's filtering mode is controlled by the FSM. Once filtering is completed, the FSM either waits for the next rising edge of the 800Hz clock for another EMG signal or immediately switches to one of the EEG signal for data processing. Since the two EEG channels are sampled at 200 Hz, the FSM enables the data processing of these channels once after every fourth processing of the EMG channel. Once filtering is complete, the FSM enables the detection modules. Different sleep stages are determined

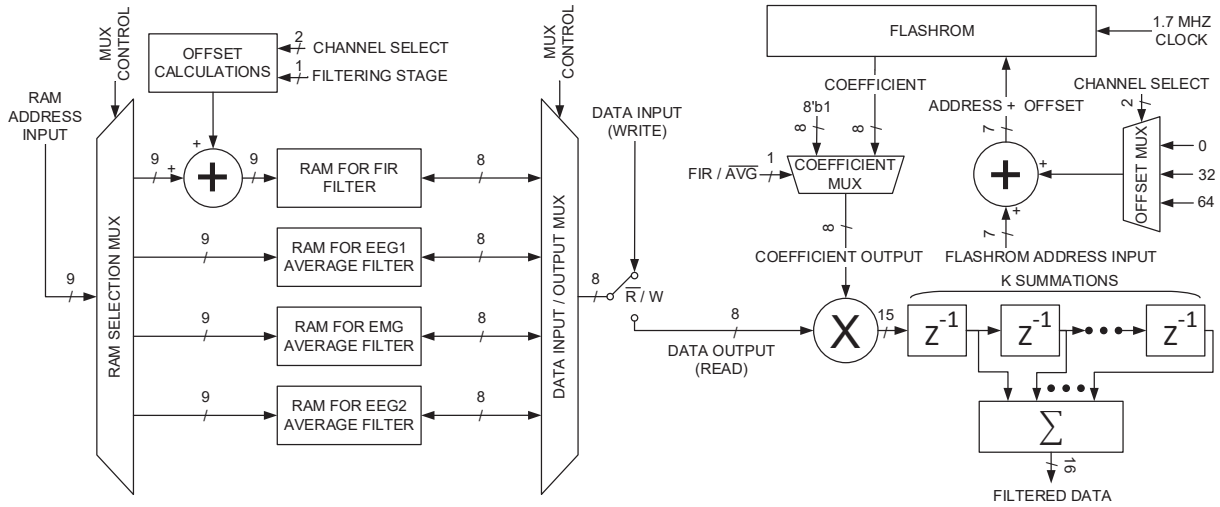
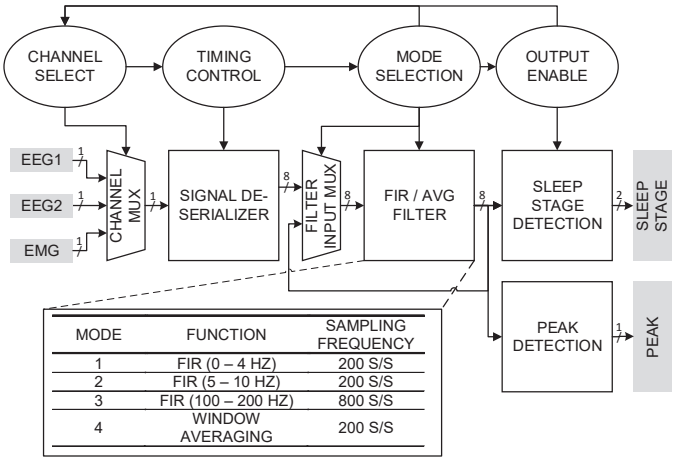


Fig. 4: Logic diagram for a reconfigurable multichannel FIR and window averaging filter.



MODE	FUNCTION	SAMPLING FREQUENCY
1	FIR (0 – 4 HZ)	200 S/S
2	FIR (5 – 10 HZ)	200 S/S
3	FIR (100 – 200 HZ)	800 S/S
4	WINDOW AVERAGING	200 S/S

Fig. 5: Block diagram for the presented implementation.

by thresholding the average filtered EMG signal and the ratio of two filtered EEG signals. Simultaneously, another module processes the filtered hippocampus EEG signal for θ-band peak detection.

B. Multichannel FIR Filtering Using Memory Address Offsets

Since one FIR filter is used for channels with different bandwidths, it has to be programmable. Also, to make the system work stand-alone for an implantable / wearable application, the coefficients must be saved on the FPGA. Since the filtering coefficients are constant and symmetric, only half of them (32 coefficients) for each channel are required to be stored. The coefficients are stored in the non-volatile FlashROM memory of the FPGA, which significantly reduces the number of required logic elements. Fig. 4 shows the process of coefficient retrieval for the shared FIR filter, using a multiplexer (OFFSET MUX) that chooses the proper address offset for every channel.

The same technique is used to set the offset for the FIR input data storage and retrieval from the RAM. In this case, 64 input data samples from each channel are needed to be stored and accessed in one RAM block on the FPGA. To store or retrieve different sets of channel-specific FIR input data with a constant range of RAM input address (0-63), a channel specific RAM offset address has to be selectively added to each input address by the Offset Calculations block shown in Fig. 4.

The bottom half of Table I summarizes the LE usage after sharing the FIR and window averaging filters, and applying the storage and retrieval techniques for the filter coefficients and data samples. The resource usage is reduced by approximately 6.8 times compared with the recently reported implementation.

C. Merging the FIR and the Window Averaging Filter

To further reduce the LE usage, the FIR and averaging filters are also merged. This is possible due to the similarity of logic structures for both filters. Both filters effectively require the addition of a specific number (k) of data samples (D_i), which are multiplied by a certain coefficient (C_i),

$$D_{out} = \Sigma_{i=1}^{k} C_i \cdot D_i \quad (1)$$

For the FIR filter, C_i is the filter coefficient, D_i is the input data, and k is the number of taps. For window averaging filter, C_i is always one, D_i is the input data, and k is the window size.

As shown in Fig. 4, a multiplexer (COEFFICIENT MUX) is used to choose between a FIR coefficient or a constant one for C_i. To choose the correct D_i, a multiplexer (RAM SELECTION MUX) directs the address to the appropriate memory block to retrieve the filter specific data via a second multiplexer (DATA INPUT / OUTPUT MUX). Since one addition operation is performed per clock cycle, the total number of summations (k) can be easily controlled by the number of clock cycles.

D. Integer Division Using High Rate Substraction

The thresholding module requires the integer division of two EEG signal amplitudes to make a detection. To avoid a multi-bit divider that increases resource utilization significantly, the division operation is performed by counting the number of times that the denominator must be subtracted from the numerator before the output becomes negative. A higher clock frequency (40 MHz) is used for this block to avoid any significant additional delay to the algorithm.

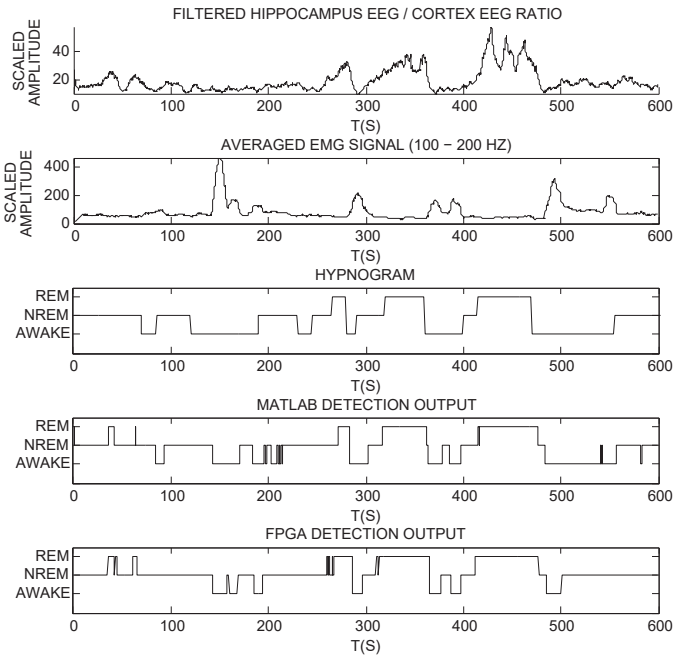


Fig. 6: Sample simulation results for the new implementation.

IV. Results

A. Classification Accuracy

The detection results are shown in Fig. 6. The first two plots show the EEG ratio and the EMG signal after filtering. From these plots, a higher EEG ratio corresponds to the REM stage and a higher EMG signal corresponds to the AWAKE stage. The third plot is the hypnogram, which is produced by manual sleep stage scoring and acts as a reference for detection outputs. The last two plots show the similarity of the Matlab and FPGA generated outputs with the hypnogram. The detection accuracy of the FPGA algorithm is 79.7% over all sleep stages. Furthermore, the sensitivity and specificity for REM sleep stage detection are 98.2% and 89.2% respectively.

B. Classification Latency

The wearable-optimized implementation is validated on the Actel IGLOO AGL060 FPGA, which controls the low-power body-interfacing ASIC. The algorithm needs 1670 clock cycles (using 1.7 MHz clock) to generate an output, which translates to a detection latency of 0.982 ms (< 1 ms). Although the implementation [7] in Table II has a lower detection latency, it is unable to perform peak detection and consumes significantly more LEs and power.

To further reduce latency, data from different channels can be processed in parallel instead of in series. Thus, a multi-stage pipeline system can be implemented so that data from different channels can be de-serialized, filtered and classified simultaneously.

V. Conclusion

The wearable-optimized implementation of the sleep classifier maintains high detection accuracy and low latency while consuming very low power. Based on the offline recordings from nine mice, the implementation has a detection accuracy of 79.7%, a REM detection sensitivity of 98.2%, a REM detection specificity of 89.2%, a detection latency of 0.982 ms and a power consumption of 0.346 mW. This new implementation enables a wearable closed-loop sleep control system to perform real-time REM suppression. As such, purposeful study of neurodegenerative diseases can be explored.

TABLE II: Comparison amoung existing hardware-based methods.

Ref.	[7]	[8]	[9]	This work
Stages classified	W,REM NREM	W, S1,S2, SWS, REM	W, Sleep	W,REM, NREM
Signal(s) used	EEG, EMG	ECG	ECG, Resp.	EEG, EMG
Method	Filtering+ thresholding	FNGLVQ	FFT, PSD, ANN	Filtering+ thresholding
Peak detection	No	No	No	Yes
Accuracy (%)	81.66	68.8	89.4-95.4	79.7
Sensitivity (%)	81.7	64.2	N/R	98.2
Specificity (%)	93.8	64.2	N/R	89.2
Computation time (ms)	0.039	790	3.75	0.982
Real-time	Yes	No	Yes	Yes
Order of filter	64	N/R	N/A	64
FPGA name	Actel ProASIC3	Spartan-3AN XCS700AN	N/A	Actel IGLOO
FPGA resource usage (LE)	9031	11108	N/A	1317
Power consumption (mW)	9.3	N/R	200	0.346

N/A: Not applicable , N/R: Not reported

References

[1] J. Horne and M. McGrath, "The consolidation hypothesis for REM sleep function: stress and other confounding factorsa review," *Biological psychology*, vol. 18, no. 3, pp. 165–184, 1984.

[2] A. Bianchetti, A. Scuratti, O. Zanetti, G. Binetti, G. Frisoni, E. Magni, and M. Trabucchi, "Predictors of mortality and institutionalization in Alzheimer disease patients 1 year after discharge from an Alzheimer dementia unit," *Dementia and Geriatric Cognitive Disorders*, vol. 6, no. 2, pp. 108–112, 1995.

[3] A. R. Adamantidis, F. Zhang, A. M. Aravanis, K. Deisseroth, and L. De Lecea, "Neural substrates of awakening probed with optogenetic control of hypocretin neurons," *Nature*, vol. 450, no. 7168, pp. 420–424, 2007.

[4] V. Bajaj and R. B. Pachori, "Automatic classification of sleep stages based on the time-frequency image of EEG signals," *Computer methods and programs in biomedicine*, vol. 112, no. 3, pp. 320–328, 2013.

[5] F. Ebrahimi, M. Mikaeili, E. Estrada, and H. Nazeran, "Automatic sleep stage classification based on EEG signals by using neural networks and wavelet packet coefficients," in *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*. IEEE, 2008, pp. 1151–1154.

[6] K. Abdelhalim, H. M. Jafari, L. Kokarovtseva, J. L. Perez Velazquez, and R. Genov, "64-channel UWB wireless neural vector analyzer soc with a closed-loop phase synchrony-triggered neurostimulator," *IEEE Journal of Solid-State Circuits*, vol. 48, no. 10, pp. 2494–2510, 2013.

[7] A. Chemparathy, H. Kassiri, M. T. Salam, R. Boyce, F. Bekmambetova, A. Adamantidis, and R. Genov, "Wearable low-latency sleep stage classifier," in *Biomedical Circuits and Systems Conference (BioCAS), 2014 IEEE*. IEEE, 2014, pp. 592–595.

[8] M. Fajar, W. Jatmiko *et al.*, "FNGLVQ FPGA design for sleep stages classification based on electrocardiogram signal," in *2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2012, pp. 2711–2716.

[9] W. Karlen, C. Mattiussi, and D. Floreano, "Adaptive sleep/wake classification based on cardiorespiratory signals for wearable devices," in *Biomedical Circuits and Systems Conference, 2007. BIOCAS 2007. IEEE*. IEEE, 2007, pp. 203–206.