CLINK: Compact LSTM Inference Kernel for Energy Efficient Neurofeedback Devices

Zhe Chen
University of California, Los Angeles
Los Angeles, California
zhechen@ucla.edu

Andrew Howe
University of California, Los Angeles
Los Angeles, California
ahowe@ucla.edu

Hugh T. Blair
University of California, Los Angeles
Los Angeles, California
tadblair@ucla.edu

Jason Cong
University of California, Los Angeles
Los Angeles, California
cong@cs.ucla.edu

ABSTRACT

Neurofeedback device measures brain wave and generates feedback signal in real time and can be employed as treatments for various neurological diseases. Such devices require high energy efficiency because they need to be worn or surgically implanted into patients and support long battery life time. In this paper, we propose CLINK, a compact LSTM inference kernel, to achieve high energy efficient EEG signal processing for neurofeedback devices. The LSTM kernel can approximate conventional filtering functions while saving 84% computational operations. Based on this method, we propose energy efficient customizable circuits for realizing CLINK function. We demonstrated a 128-channel EEG processing engine on Zynq-7030 with 0.8 W, and the scaled up 2048-channel evaluation on Virtex-VU9P shows that our design can achieve 215x and 7.9x energy efficiency compared to highly optimized implementations on E5-2620 CPU and K80 GPU, respectively. We carried out the CLINK design in a 15-nm technology, and synthesis results show that it can achieve 272.8 pJ/inference energy efficiency, which further outperforms our design on the Virtex-VU9P by 99x.

CCS CONCEPTS

• **Hardware** → **Digital signal processing**; *Reconfigurable logic and FPGAs*; • **Computing methodologies** → Machine learning;

KEYWORDS

Electroencephalography (EEG), filtering, long short-term memory (LSTM), neurofeedback

ACM Reference Format:
Zhe Chen, Andrew Howe, Hugh T. Blair, and Jason Cong. 2018. CLINK: Compact LSTM Inference Kernel for Energy Efficient Neurofeedback Devices. In *ISLPED '18: ISLPED '18: International Symposium on Low Power Electronics and Design, July 23–25, 2018, Seattle, WA, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3218603.3218637

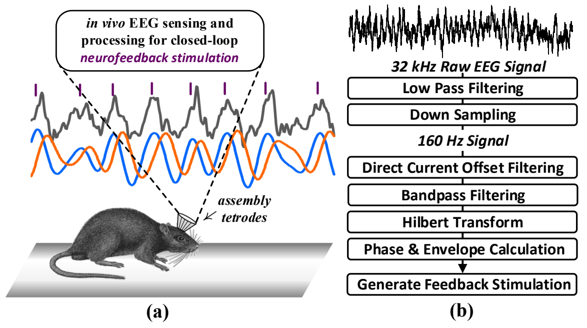
in vivo EEG sensing and processing for closed-loop *neurofeedback stimulation*
assembly tetrodes
(a)
32 kHz Raw EEG Signal
Low Pass Filtering
Down Sampling
160 Hz Signal
Direct Current Offset Filtering
Bandpass Filtering
Hilbert Transform
Phase & Envelope Calculation
Generate Feedback Stimulation
(b)

Figure 1: (a) EEG neurofeedback application on a freely behaving rat and (b) a conventional EEG signal processing flow for neurofeedback stimulation.

1 INTRODUCTION

Neurological diseases such as Parkinsonism and epilepsy can be treated using neurofeedback *deep brain stimulation* (DBS). Next-generation DBS devices will be able to constantly monitor the patients' brain activity recorded from neural probes, sense when disease symptoms are occurring, and automatically deliver neural stimulation to immediately relieve those symptoms. Such neurofeedback device requires specialized hardware with high energy efficiency, because it needs to be worn or surgically implanted into a human patient or an experimental subject and support long battery life time when frequent charging is not possible.

Electroencephalogram (EEG) is widely utilized for a broad range of clinical and research applications. A typical closed-loop neurofeedback stimulation case with conventional EEG signal processing flow is illustrated in Figure 1. In order to select specific band of signals from the EEG, a series of filtering and transformation operations are required. Although this flow works effectively during offline analysis, it causes two problems when it is applied to a neurofeedback device. First, the low pass filter operating at the raw sampling rate is energy consuming, especially when the number of EEG channels is high. Secondly, the bandpass filter and the Hilbert transform operator are implemented by FIR or IIR filters [8], which cause undesirable acausal delay whey they are deployed to real-time neurofeedback applications.

Meanwhile, the temporal and spatial resolution of the brain signal sensing is increasing rapidly. According to [13], researchers are aiming at making neurofeedback devices that can monitor the brain activity using 1 million electrodes simultaneously and generate feedback stimulation on up to 100,000 selected neurons. The increased workload will drastically increase the requirement on both the performance and the energy efficiency of the neurofeedback devices. Considering 128 channels of EEG signal sampled at 32 kHz and assuming each sample consumes ~600 operations, it requires a 2.5-GOPS throughput, which is hard to be satisfied under the temperature increase limit of $2°C$ [11] for bio-embedding applications.

In this paper, we propose a compact long short-term memory (LSTM) inference kernel (CLINK) for energy efficient EEG signal processing. First, we introduce the method in reducing computation cost and acausal delay for neurofeedback devices. Then we present the circuit design for an efficient CLINK implementation. Finally, we evaluate our design by synthesizing the CLINK with a 15nm CMOS technology, and make comparisons to scaled-up designs on high-end FPGAs and highly optimized implementations on multi-core CPUs and GPUs. The contributions of this paper are summarized as follows:

- To our best knowledge, we are the first to propose using the compact LSTM network to perform the EEG filtering. Our method can save computational operations by 84% and reduce the acausal delay incurred by digital IIR/FIR filters for energy efficient neurofeedback devices.
- Design fully pipelined processing circuits with reconfigurable multiplier to realize the CLINK function with high energy efficiency and short processing latency.
- Implement a 128-channel EEG processing and neurofeedback prototype on Zynq-7030 with 0.8 W power consumption. Scale up the design on Virtex-VU9P achieving 215x and 7.9x energy efficiency against implementations on E5-2620 CPU and K80 GPU, respectively. CLINK circuits synthesized in a 15-nm process achieves 272.8 pJ/inference energy efficiency.

2 BACKGROUND

2.1 EEG Basics

EEG is a widely used technique for sensing electrical activity generated by brain at sub-millisecond time resolution, either invasively from electrodes implanted in the brain, or non-invasively from scalp electrodes. In humans and other mammals, the brain generates rhythmic oscillations that can be detected in the EEG across a range of different frequencies. Major EEG frequency bands includes: delta rhythm (δ; <4 Hz), theta rhythm (θ; 4–10 Hz), alpha rhythm (α; 10–15 Hz), beta rhythm (β; 15–30 Hz) and gamma rhythm (γ; >30 Hz). In addition to these oscillatory rhythms, transient EEG signals such as K-complexes, P-300 waves, and sharp-wave ripple (SWR) events are also known to occur in specific brain regions [2]. Research has shown that certain brain rhythms and transient EEG events can be used as biomarkers of abnormal brain activity, such as seizures or anxiety attacks [2]. In this paper, we focused on a processing flow that can be generally used to detect such EEG rhythms or transient signals, and deliver closed-loop stimulation that is precisely synchronized to the detected signals.

2.2 Conventional EEG Processing Flow

Raw EEG signals are typically sampled at a fast rate such as 32 kHz. To analyze a specific frequency band, raw EEG data is often downsampled to the lowest rate correspondingly. For example, to analyze data in the theta band, the raw EEG data needs to be downsampled to 160 Hz, which is about ten times the upper cutoff frequency of the passband. A bandpass filter is then applied to this downsampled signal, and a standard method for extracting the phase information to synchronize neurofeedback stimulation involves taking the Hilbert transform which shifts the phase of the bandpass filtered signal by 90° [8].

The ideal bandpass and Hilbert transform filters can be approximated by FIR or IIR digital filters, but these filters are acausal and they can incur unacceptably long delay, so they are more useful for offline analysis. In a real-time application, FIR or IIR implementations can impose unwanted delay longer than 10 ms in a neurofeedback closed loop, and it can cause the stimulation not working properly.

The final step in the EEG signal processing flow is the phase and envelope calculation. We denote the real and imaginary components of the analytic signal as the u_r and the u_i, and they are computed from the bandpass filter and the Hilbert transform, respectively. Finally, the phase and the envelope are calculated by the equations:

$$\begin{aligned} \phi(n) &= \arctan(u_i(n)/u_r(n)) \\ u_A(n) &= \sqrt{u_r^2(n) + u_i^2(n)} \end{aligned} \tag{1}$$

where $\phi(n)$ and $u_A(n)$ represent the calculated phase and envelope for the time step n, respectively. The phase and envelope information can be used to detect specified events in the EEG, and the detection can be used to generate real-time closed-loop feedback control to accomplish the therapy or research tasks.

2.3 LSTM Model

LSTM [7] is a type of recurrent neural networks (RNNs) that has been successfully used for many temporal signal prediction tasks, such as the handwriting recognition [4] and the speech recognition [5]. A typical LSTM network architecture is made up of the hidden layer and the output layer, as Figure 2 shows. The hidden layer consists of four types of gates: the input gate $\boldsymbol{I}$, the forget gate $\boldsymbol{F}$, the cell gate $\boldsymbol{G}$ and the output gate $\boldsymbol{O}$, and two types of nodes: the cell node $\boldsymbol{C}$ and the hidden node $\boldsymbol{H}$. At each time step n, the LSTM takes the input *In(n)*, updates the values of all the gates and the nodes, and then generates output *Out(n)* based on the hidden node value. The LSTM inference is defined by the equations:

$$\boldsymbol{I}(n) = \sigma\left(In(n) \cdot \boldsymbol{W}_{Ii} + \boldsymbol{H}(n-1) \cdot \boldsymbol{W}_{IH} + \boldsymbol{B}_I\right) \tag{2}$$

$$\boldsymbol{F}(n) = \sigma\left(In(n) \cdot \boldsymbol{W}_{Fi} + \boldsymbol{H}(n-1) \cdot \boldsymbol{W}_{FH} + \boldsymbol{B}_F\right) \tag{3}$$

$$\boldsymbol{G}(n) = \tanh\left(In(n) \cdot \boldsymbol{W}_{Gi} + \boldsymbol{H}(n-1) \cdot \boldsymbol{W}_{GH} + \boldsymbol{B}_G\right) \tag{4}$$

$$\boldsymbol{O}(n) = \sigma\left(In(n) \cdot \boldsymbol{W}_{Oi} + \boldsymbol{H}(n-1) \cdot \boldsymbol{W}_{OH} + \boldsymbol{B}_O\right) \tag{5}$$

$$\boldsymbol{C}(n) = \boldsymbol{F}(n) \odot \boldsymbol{C}(n-1) + \boldsymbol{G}(n) \odot \boldsymbol{I}(n) \tag{6}$$

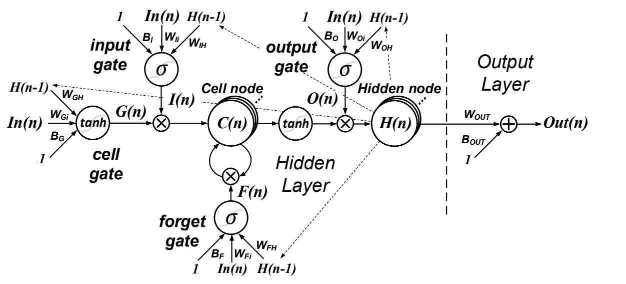

Figure 2: Typical LSTM network architecture with one hidden layer and an output layer.

$$H(n) = \boldsymbol{O}(n) \odot \tanh(\boldsymbol{C}(n)) \tag{7}$$

$$Out(n) = \boldsymbol{H}(n) \cdot \boldsymbol{W}_{OUT} + B_{OUT} \tag{8}$$

among which Eq. 2–5 describe the update of the gates, and Eq. 6 and Eq. 7 define how the cell node and hidden node values are updated based on the updated gate values. Eq. 8 shows the computation of the inference result $Out(n)$. σ and $tanh$ are non-linear sigmoid and hyperbolic tangent functions, and $\odot$ represents for the dot product operation. Suppose the LSTM has one hidden layer, and the hidden layer consists of S hidden nodes, then the input weights and bias are denoted as S-dimensional vectors $\{\boldsymbol{W}_{Ii}, \boldsymbol{W}_{Fi}, \boldsymbol{W}_{Gi}, \boldsymbol{W}_{Oi}\}$ and $\{\boldsymbol{B}_I, \boldsymbol{B}_F, \boldsymbol{B}_G, \boldsymbol{B}_O\}$, and the $\{\boldsymbol{W}_{IH}, \boldsymbol{W}_{FH}, \boldsymbol{W}_{GH}, \boldsymbol{W}_{OH}\}$ are $S \times S$ internal weight matrices. The W_{OUT} is the S-dimensional output weight vector and the B_{OUT} is the bias for the output layer.

3 CLINK: OUR PROPOSED METHOD

3.1 LSTM-based EEG Processing Flow

We propose an EEG signal processing flow based on a pair of compact LSTM networks to reduce the computation cost and latency, as Figure 3 shows. The LSTM networks are independently trained to generate predictions of the u_r and u_i described in Section 2.2. In the offline training, both LSTMs take the same downsampled and direct current offset (DCO) filtered EEG signal as the training input, and they take the bandpass filtered signal u_r and the phase shifted signal u_i as the training targets. Once the training is accomplished, the well-trained LSTMs are used to generate predictions on new input data and approximate to the acausal filtering functions. In the final step, the inference results are relayed to a calculator to compute the phase and envelope values which are used to perform event detection for the neurofeedback stimulation.

3.2 Algorithm Evaluation

We carried out a simulation with 6-minute EEG data recorded from a freely behaving rat. The sampling rate of the raw EEG data is 32 kHz. The 4-12 Hz theta band is chosen as the analyzing frequency range, and the down sampling rate and the DCO size are set to be 200 and 256, respectively. The offline FIR and IIR implementations [8] of the Butterworth bandpass filter and the Hilbert transform operator are used to generate target sequences for

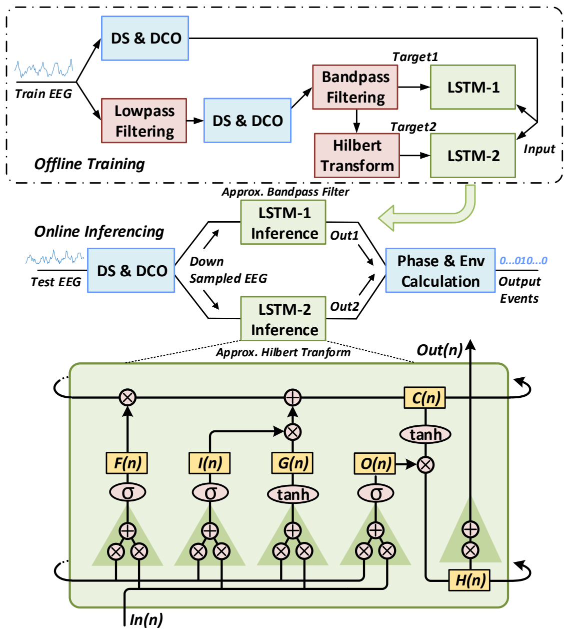

Figure 3: Proposed LSTM based EEG signal processing flow.

the LSTM training. The training program normalizes the input and target sequences and then performs training based on the standard LSTM model supported by Caffe [9]. We started the training from a sufficiently large 50-hidden-node LSTM network, and then shrank the network size to achieve a compact implementation. For a well-trained 50-node network, the inference results match the training targets quite well, and they are used to compute the phase and the envelope according to Eq. 1. The mean phase error is within ±3° of zero after calibration and it is accepted as a highly accurate result for real-time neurofeedback stimulation.

We used a pair of measurement errors ε_R and ε_A, referred to as the real signal error and the envelope error, to evaluate the accuracy of the LSTM inference across different frequency bands. In general, the ε_R and ε_A measure the variance of the deviation of the inference and target values across the data samples, for the real signal and the envelope amplitude, respectively, and they are computed as:

$$\varepsilon_R = \sigma^2(\widetilde{x_R} - \widetilde{y_R}), \tag{9}$$

$$\varepsilon_A = \sigma^2(\widetilde{x_A} - \widetilde{y_A}), \tag{10}$$

where $\widetilde{x_R}$ and $\widetilde{y_R}$ are z-score normalized inference and target real signals, and $\widetilde{x_A}$ and $\widetilde{y_A}$ are z-score normalized inference and target envelope amplitudes, respectively. We first trained the LSTMs on multiple EEG bands such as the theta band and the gamma band with 50 hidden nodes, and then we gradually reduced the number of nodes and retrained the network. Figure 4 shows the ε_R and ε_A, and the normalized computational complexity in correspondence with the number of nodes in the LSTM network for the gamma band, which is more difficult to get converged in the training than

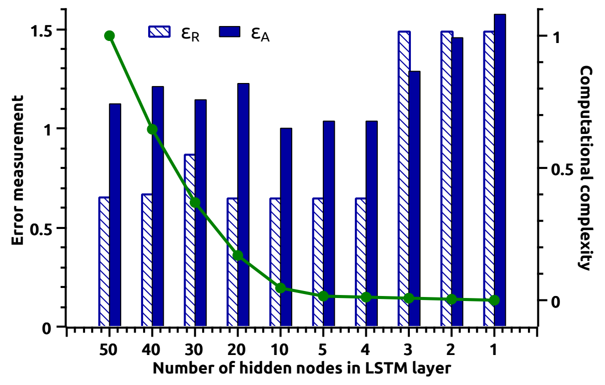

Figure 4: The error measurement and the computation complexity against the number of hidden nodes in the LSTM network for the Gamma band.

Table 1: Comparison of the EEG Filtering Methods

Methods	LSTM-Based	Conventional
Implementation	5-node LSTM	5-tap IIR
Latency @100 MHz	2.09 μs	12.5 ms
Operation Count	646	4,094

the theta band. As Figure 4 shows, neither of the ε_R and ε_A shows an obvious increase until the LSTM layer size is reduced to 3 hidden nodes. We observed similar trends of error increase along with the network shrinking for the theta band and the SWR band. In general, as the LSTM network shrinks, the error increases because the general inferencing capability of the network decreases. In order to provide a compact LSTM network with the general inferencing capability for various bands, we set the layer size to be 5 hidden nodes based on our algorithm exploration.

The main benefit of the proposed method is that the LSTM inference can largely reduce the computation cost, while remaining high accuracy in making an approximation to the filter function. As Figure 3 shows, when we deploy the well-trained LSTMs for online inferencing, we can bypass the low pass filter, which operates at the raw sampling rate and is the most energy consuming part based on our evaluation. Considering 32 kHz sampling rate and a down sample rate of 200 for the theta band, the LSTM inference can save 84% computational operations while keeping acceptable accuracy for the neurofeedback stimulation. Another benefit of the proposed method is that it reduces the latency of acausal FIR/IIR filters and shortens the processing delay. For example, a 5-hidden-node LSTM network costs 296 operations to accomplish an inference, and the implementation achieves 2.09 μs latency under 100 MHz clock frequency. On the other hand, a 5-tap IIR filter results in 12.5 ms acausal delay considering a 160 Hz rate after the down sampling. In this way, the proposed method can generate a quick response for stimulation in neurofeedback devices. Table 1 shows the comparison between the proposed method based on the LSTM and the conventional digital signal processing flow.

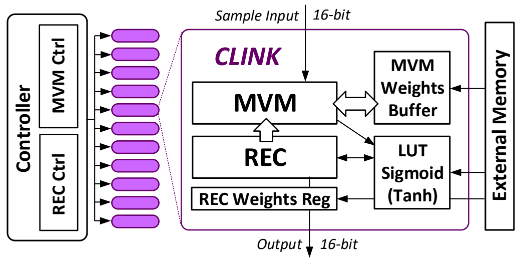

Figure 5: General architecture of the CLINK engine.

The offline training is performed in floating point, and we converted it into 16-bit fixed point for online inference. Evaluation results show that it does not cause essential accuracy loss. For the specific theta band, we in further carried out weights pruning and low bit-width quantization. As a result, we can prune over 35% weights and reduce the bit-width of the weights to 9-bit for 3-hidden-node LSTMs without significant accuracy degradation, but this step of optimization does not work consistently for various bands. In this paper, we focus on the 5-hidden-node LSTM in 16-bit, which provides general inferencing capability for a variety of EEG bands, including theta, gamma, high gamma (>60 Hz), low gamma (30-60 Hz), and SWR bands.

4 CLINK: CIRCUIT DESIGN

Based on CLINK method, we designed a customizable processing engine for efficient LSTM inference. Figure 5 shows the general architecture of the proposed design. The design is composed of an array of CLINK processing elements (PEs) and a controller. The CLINK PEs operate in a massively parallel way to perform filtering for multiple EEG channels. Each PE consists of a matrix-vector multiplication (MVM) module, a recurrent state update (REC) module, a look-up table (LUT) and weight buffers corresponding to the MVM and the REC modules. The MVM and the REC circuits operate in full pipeline under the control of submodules of the controller. The LUT is used for performing non-linear *sigmoid* and *tanh* operations. The MVM weight buffer stores input and hidden layer weights, while the REC weight buffer stores the output weights.

4.1 MVM Design

Figure 6 shows the MVM circuit design. The MVM is composed of a 1-D multiplier bank and an adder tree. In each clock cycle, the MVM receives one 16-bit input sample, fetches N 16-bit weights from the MVM weight buffer and performs $N + 1$ multiplication operations, where N is the number of hidden nodes. The multiplication results are right shifted by 12 bits and the adder tree calculates the sum of the shifted values. The sum is shifted again by 4 bits and a 2-bit sign is calculated by comparing the result x to the LUT size *Range* according to Eq. 11. The MVM operates in N iterations for each inference, and in each iteration, the input gate, cell gate, forget gate and output gate are updated in full pipeline since there is no data dependency between consecutive two updates. For updating the cell gate, the MVM performs an additional 1-bit shift on the result, because the *tanh* and the *sigmoid* operations can be unified by Eq.

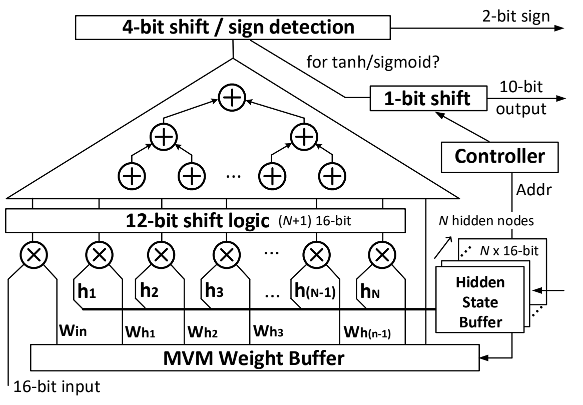

Figure 6: Schematic diagram of the MVM module.

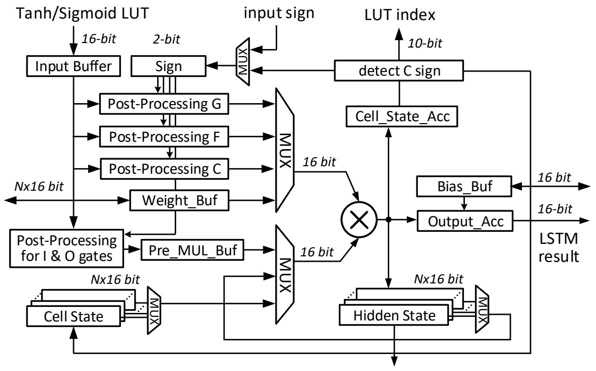

Figure 7: Schematic diagram of the REC module.

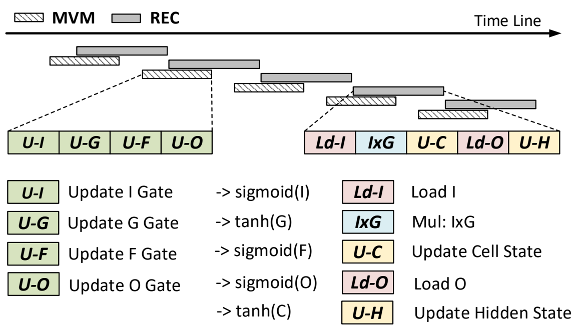

Figure 8: Timing diagram of the pipeline operations of the MVM and the REC modules.

12 and only one 2kB LUT is needed. Finally, the hidden node values are updated at the end of each inference by the REC module.

$$Sigmoid(x) = \begin{cases} 2'b00 : MAX \; [\, x > Range] \\ 2'b01 : LUT(x) \; [0 \le x \le Range \,] \\ 2'b10 : MAX - LUT(-x) \; [-Range < x < 0] \\ 2'b11 : MIN \; [x \le Range] \end{cases} \tag{11}$$

$$\tanh(x) = 2sigmoid(2x) - 1 \tag{12}$$

4.2 REC Design

Figure 7 shows the REC circuit. The REC consists of an input buffer, a temporary buffer *Pre_MUL_Buf*, cell and hidden node registers, a cell state accumulator *Cell_State_Acc*, an output accumulator *Output_Acc* and REC weight buffers composed of *Weight_Buf* and *Bias_Buf*. Two multiplexers are used to reconfigure data paths for the multiplication operation, and dedicated logic is used to detect the 2-bit sign from the accumulated C value and perform non-linear operations according to Eq. 11 and Eq. 12. The novelty of the REC design is to reconfigure the data path to reuse the multiplier and fully pipeline update operations described by Eq. 6–8. Figure 8 shows the timing diagram of the MVM and the REC. Each iteration of the REC update is pipelined into five consecutive cycles, in which the input buffer is updated by the LUT readouts corresponding to $sigmoid(I)$, $tanh(G)$, $sigmoid(F)$, $sigmoid(O)$ and $tanh(C)$ in sequential order. In the first cycle, the $sigmoid(I)$ stored in the input buffer is post processed according to Eq. 11 and transferred to the *Pre_MUL_Buf*. Meanwhile, the selected weight in the *Weight_Buf* is multiplied by the corresponding hidden node value to update the *Output_Acc*. In the second cycle, the $sigmoid(I)$ stored in the *Pre_MUL_Buf* is multiplied by the $tanh(G)$ stored in the input buffer, and the multiplication result is transferred to the *Cell_State_Acc*. In the third cycle, the $sigmoid(F)$ from the input buffer is multiplied by the selected cell value and the result is accumulated to the *Cell_State_Acc* to update the cell value. In the fourth cycle, the $sigmoid(O)$ is stored in the *Pre_MUL_Buf* and in the last cycle, the $tanh(C)$ from the input buffer is multiplied by the $sigmoid(O)$ stored in the *Pre_MUL_Buf* to update the corresponding hidden node value. Once all of the N iterations are finished, the hidden node values are used to update the hidden node buffer in the MVM.

5 EVALUATION

We first implemented a 128-channel EEG processing and neurofeedback prototype design based on the CLINK designed with Vivado HLS on Zynq-7030. Each CLINK PE is shared by 16 EEG channels. Under 100 MHz clock frequency, the EEG processing latency is 2.68 μs, and the estimated power consumption for the FPGA kernel and the SoC device are 0.80 W and 2.79 W, respectively.

We then scaled up the CLINK design on Zynq-7045, Virtex-690t and Virtex-VU9P devices by 3x, 6x and 16x, respectively. We also evaluated the peak performance of the CLINK on multi-core CPUs and GPUs with OpenMP and CUDA programming to understand the performance and efficiency achieved by the FPGA designs. Under 300 MHz clock frequency, evaluation results show that the design on the Virtex-VU9P achieves 315 M samples/s throughput and 54 nJ/sample energy efficiency, which outperforms designs on the E5-2620 CPU with 12 threads and on the K80 GPU by 215x and 7.9x, respectively. Fig. 9 shows comparison of energy efficiency and throughput on selected platforms.

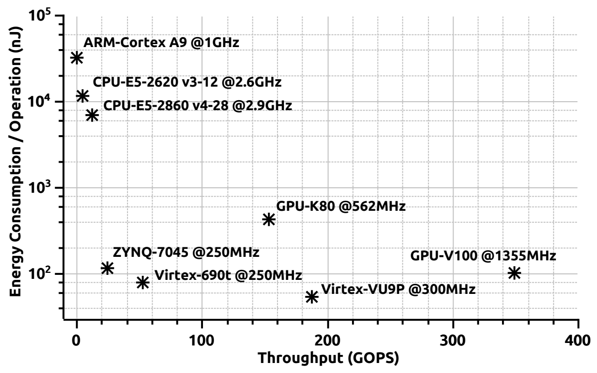

Figure 9: Energy efficiency and performance of the CLINK implementation on different platforms.

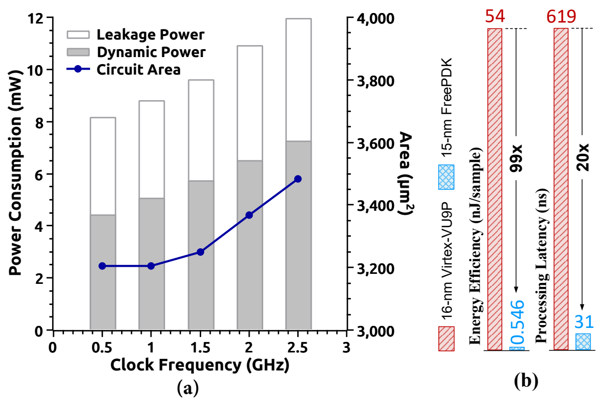

Figure 10: (a) Power consumption and area evaluation under different clock frequency. (b) Energy efficiency and latency comparison between FPGA and ASIC designs of the CLINK.

We further evaluated the CLINK design by synthesizing the MVM and REC circuits design in the 15-nm FreePDK [1] process using the Synopsys Design Compiler. The estimation of power consumption and circuit area under various clock frequency is shown in Figure 10(a). The circuit area gradually increase as the clock frequency increases over 1 GHz, while the dynamic power linearly increases with the frequency. Figure 10(b) shows the comparison on energy efficiency and processing latency between the CLINK implementation on Virtex-VU9P and the circuit design in the FreePDK process. The circuit design achieves 272.8 pJ/inference energy efficiency at 1 GHz clock frequency. Compared to the VU9P design, it achieves 99x higher energy efficiency, and reduces 95% processing latency.

6 RELATED WORK

LSTM for EEG The LSTM method has been used for EEG signal analysis in previous literature [3, 10]. However, to our best knowledge, no previous works used LSTM methods to predict filtered EEG signals or aimed at reducing the computation cost and the acausal filter delay for neurofeedback device. Some papers used the LSTM to generate prediction on future EEG samples [10]. Compared to this work, our method uses the LSTM to generate inference on the filtered EEG signal, and our algorithm evaluation shows that it achieves not only high accuracy but also low computation cost and short latency.

Hardware for LSTM Some recent works on FPGA-based LSTM acceleration achieved both high performance and high energy efficiency [6, 12]. However, these implementations targeted at different applications such as the speech recognition, which requires much larger LSTM model compared to this work. The compact LSTM proposed in this paper not only reduces the computation cost on algorithm level, but also provides opportunity to leverage the parallelism by carrying out customized circuit design. Our CLINK circuit shows additional advantage over the conventional FPGA design in energy efficiency and is suitable for the neurofeedback application which requires high energy efficiency.

7 CONCLUSIONS

In this paper, we proposed the CLINK as an energy efficient EEG processing method. It largely reduces computation cost and latency for closed-loop neurofeedback applications. We introduced the CLINK circuit design as a highly energy efficient implementation. Our design can support long battery life time for future neurofeedback devices, and it has the potential to be used as treatments for a variety of neurological disorders such as epileptic seizures, major depression and Parkinsonian motor tremors.

ACKNOWLEDGMENTS

This work is partially supported by the Depression Grand Challenge (DGC) HSSEAS Foundation Fellows Gift and the NSF under Grant No.: CCF-1436827 and No.:DBI-1707408.

REFERENCES

[1] Kirti Bhanushali and W Rhett Davis. 2015. FreePDK15: An Open-Source Predictive Process Design Kit for 15Nm FinFET Technology. In *Proc. Int. Symp. Phys. Des. (ISPD '15)*. 165–170.
[2] György Buzsaki. 2006. *Rhythms of the brain.* Oxford University Press.
[3] P R Davidson, R D Jones, and M T Peiris. 2005. Detecting behavioral microsleeps using EEG and LSTM recurrent neural networks.. In *27th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.*, Vol. 6. 5754–5757.
[4] Alex Graves, Marcus Liwicki, Santiago Fernandez, and et al. 2009. A novel connectionist system for unconstrained handwriting recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* 31, 5 (May 2009), 855–868.
[5] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. 2013. Speech recognition with deep recurrent neural networks. In *IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP '13)*. 6645–6649.
[6] Song Han, Junlong Kang, Huizi Mao, and et al. 2017. ESE: Efficient speech recognition engine with sparse LSTM on FPGA. In *Proc. ACM/SIGDA Int. Symp. Field-Programmable Gate Arrays (FPGA '17)*. 75–84.
[7] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.* 9, 8 (Nov 1997), 1735–1780.
[8] Forester W. Isen. 2009. *DSP for MATLAB and LabVIEW III: Digital filter design.* Morgan and Claypool Publishers, New York, NY.
[9] Yangqing Jia, Evan Shelhamer, Jeff Donahue, and et al. 2014. Caffe: Convolutional architecture for fast feature embedding. In *Proc. 22nd ACM Int. Conf. Multimedia (MM '14)*. 675–678.
[10] Louis Kim, Jacob Harer, Akshay Rangamani, and et al. 2016. Predicting local field potentials with recurrent neural networks. In *38th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.* 808–811.
[11] G Lazzi. 2005. Thermal effects of bioimplants. *IEEE Eng. Med. Biol. Mag.* 24, 5 (Sep 2005), 75–81.
[12] Shuo Wang, Zhe Li, Caiwen Ding, and et al. 2018. C-LSTM: Enabling Efficient LSTM Using Structured Compression Techniques on FPGAs. In *Proc. ACM/SIGDA Int. Symp. Field-Programmable Gate Arrays (FPGA '18)*. 11–20.
[13] Rafael Yuste, Sara Goering, Blaise Aguera Y Arcas, and et al. 2017. Four ethical priorities for neurotechnologies and AI. *Nature* 551, 7679 (Nov 2017), 159–163.