ESC499 Thesis Interim Report

Automatic sleep staging transformer model and ASIC accelerator

Tristan Robitaille (1006343397)
tristan.robitaille@mail.utoronto.ca

Supervisor: Professor Xilin Liu

April 11, 2024

Contents

List of Figures

List of Tables

1 Introduction[1]

As reported by Chaput *et al.* [1], insomnia impacts around 24% of Canadians adults. Detection and classification of sleep stages, known as *sleep staging*, followed by neuromodulation has been recently found by Yoon [2] to be a promising treatment against insomnia. The current stage-of-the-art for sleep staging involves the use of polysomnography to measure biosignals (at least 19 sensors are required, as explained by Levin and Chauvel [3]) and manual annotation by a sleep expert, which requires, on average, 2 hours of work [4]. This technique also does not provide neuromodulation. To address these downsides while effectively treating insomnia, we propose an in-ear device performing electroencaphalogram (EEG) sensing, sleep staging and neuromodulation. To maximize treatment potential, the device should be as small and portable as possible such that it can be used at home.

This thesis focuses on the development of a deep learning model to perform sleep staging and on the design of an accelerator ASIC module to perform in-situ inference with said model. In the end, we aim to prove, by simulations, the merit of such an accelerator in order to potentially integrate it in the in-ear device. Multiple authors [5]–[7] have published high-accuracy results using a deep learning approach to sleep staging, and have done so with significantly fewer sensors than polysomnography. However, these AI models run on standard computers as software frameworks and are thus unsuitable for a lightweight integrated solution. Google sells small custom AI-accelerators (such as the Coral Edge TPU) that could run these AI models, but they still consume too much power (1W, [8]) and do not readily integrate with custom neuromodulation hardware.

The proposed solution should match the accuracy of traditional polysomnography and published models in the literature with a power consumption low enough that the whole system can be powered for at least a full-night on a battery that fits in-ear.

This document describes the high-level technical direction of the project, serves to report the current state of literature in both sleep staging using deep learning and AI accelerator hardware in order to define a gap that is filled by this project. It also discusses the progress made to date and the work that is left to complete.

1.1 Detailed design constraints and direction

Table 1 indicates precise design goals and their justification, which helps guide design decision and development effort. For example, to reach the target model size, time will be spent evaluating the impact of hyperparameters to find the combination that gives the lowest size while meeting the desired accuracy. Furthermore, quantization and pruning will be explored to reduce model size. For the AI accelerator, since inference

[1]Adapted from the *Thesis Proposal*, submitted in October 2023.

Table 1: Design goals for AI model and ASIC accelerator

Type	Goals	Justification
Model	Size $< 125\,\text{kB}$	Help reach ASIC area/power goals
	Accuracy $> 80\%$	Competitive with state-of-the-art
ASIC	$P_{\text{avg}} < 1\,\text{mW}$	System to function for whole night
	$T_{\text{inference}} < 3\,\text{s}$	Maximum phase offset of 10%
	$A_{\text{total}} < 1.5\,\text{mm}^2$	Fit in ear (65nm node)

power and clock frequency are inversely proportional, we must focus on reducing energy per inference. From first principles, this implies reducing the amount of charge that is displaced within the chip. Since the physical properties are locked for the target 65nm node, we focus on reducing the number of operations, simplifying operations, limiting data movement and reducing control logic.

To determine the average power consumption constraint, the battery capacity of the Airpods Pro, which is 49.7mAh, can be considered as a reference [9]. Assuming a supply voltage of 1V and 10h of battery life, the overall power consumption must stay below 5mW, on average. The speaker coil driving circuitry will consume most of the power and, leaving enough power budget for the analog front-end and overhead in the system, in addition to a safety factor, 1mW seems like a reasonable target for the accelerator. Regarding the target area and model size, a reasonable chip package for this application is a 4mm x 4mm Chip-Scale Package (CSP). According to IPC standard J-STD-012, a CSP overall area is no more than 120% of the die, which leaves us with $13mm^2$ of die area [10]. Again, a significant portion will be consumed by the coil driver, RISC-V processor and memory. Therefore, a maximum area of 10%, or $1.5mm^2$, for the accelerator seems to be a safe option. According to Liu and Kursun, the area of a standard 6T SRAM cell is $0.75\mu m^2$ [11]. To leave enough area for the compute elements, controllers and routing, reserving 50% of the accelerator area to weights is reasonable. Thus, with $0.75mm^2$ that can be allocated to weights, the total model size can be 1Mb, or 125kB. Finally, as mentioned above the accuracy should be close to published literature in this field to provide effective treatment. Current PSG divides the night in 30s “sleep epochs”. To limit too much of an offset between the end of a 30s epoch and its predicted sleep stage (essentially a phase offset), which would add inaccuracy to the overall system and potentially reduce its effectiveness, a maximum phase offset of 10% (3s) is appropriate.

2 Literature review

2.1 ML for sleep staging

Deep learning for sleep staging has been studied since around 2017. Broadly speaking, basic deep neural networks (DNN) came first, followed by convolutional neural networks (CNN) and recurrent neural-networks (RNN) [12]. The transformer is a relatively new type of neural network based around the concept of "attention" and particularly suited for sequence inputs since it can process the input in parallel [13]. Since its introduction in 2017 [14], the transformer has been used for sleep staging tasks. Indeed, Dai *et al.* developed a transformer-like model without decoders which used three input EEG channels and achieved an impressive 87.2% accuracy on the popular SleepEDF-20 dataset [15]. Similarly, Phan *et al.* developed a model with a focus on outputting easily-interpretable confidence metrics for clinicians. They found that a significant impediment to the adoption of automatic sleep staging in clinics is lack of trust from clinicians as they perceive the system to be a "black box". Their model ingests multiple sleep epochs for each inference, which allowed the team to achieve 84.9% accuracy on the SleepEDF-78 dataset. Eldele *et al.* managed an accuracy of 85.6% on SleepEDF-78 using a single-channel, single-epoch attention-based model [7].

In recent years, the accuracy of sleep staging by ML models has plateaued. In fact, Phan *et al.* claim that AI-based sleep-staging in heathly patients has been solved fully as the accuracy has reached the "almost perfect" level of Cohen's kappa [4]. However, none of the models presented above meet our constraints. Indeed, we require a lightweight, single-channel, single-epoch model. Most models have more than 1M parameters [12]; even the smallest model by Eldele *et al.* has above 500k 32-bit float weights, which far exceeds the 125kB constraint. Furthermore, none have been optimized to run on custom hardware. Thus, there is a need to develop a novel lightweight transformer.

2.2 AI accelerator hardware

AI accelerators are a very active area of research at the moment. Significant gains in latency and power are possible by designing hardware optimized for machine learning. Indeed, CPUs lack in memory bandwidth and parallelism and have significant control overhead as required to process any arbitrary program. GPUs have high cost and high power consumption and typically have less available memory than a CPU. Hardware (FPGA and ASICs), on the other hand, can be fast and energy efficient but suffer from limited flexibility [16]. The first AI accelerator ASIC, publised in 2014 by Chen *et al.* and named *DianNao*, could perform 452G 16b fixed-point computations per second [17]. It was quickly surpassed by *ShiDianNao*, which was 1.87x faster and used 60x less energy [18].

Modern models are massive and suffer from memory access penalties. Thus, the most optimized architectures limit data movement as much as possible by placing the compute elements directly *in memory*, meaning that less charge is switched per operation (DRAM access requires 200x more energy than on-chip memory [19]) and the overhead and bottleneck effect of the data bus is drastically reduced. For example, Arora *et al.* describe "CoMeFa", a compute-in-memory (CiM) module for FPGA BRAM which managed to reduce energy comsumption by 55%. They placed up to 160 single-bit processing element per BRAM (20kbit) and perform operations in a bit-serial manner instead of the traditional bit-parallel [20]. Similarly, Wang and colleagues present a similar BRAM CiM module which speeds up compute by up to 2.3x at the cost of 1.8% increase in area [21]. A high-level architecture that often uses a number of CiM is known as "dataflow". Unlike the traditional von Neumann architecture, it does not rely on instructions and banks of memory cells to perform the necessary computation, but instead organizes different compute elements sequentially and in parallel to match a model's architecture. For instance, Farabet *et al.* present a runtime-reconfigurable dataflow architecture based on 9 pipelined "processing tiles" that can be reconfigured in $\sim 10^4$-10^5 permutations [22].

Another area of research is concerned with the software-hardware co-design opportunities afforded by tight integration between the model and the hardware running its inference. Ding and team describe an impressive design where both the training backpropagation and inference hardware are modified. By ensuring that weight matrices consist of an array of circulant matrices, the team is able to reduce time-complexity of vector-matrix mutliply from $\mathcal{O}(n^2)$ to $\mathcal{O}(n \log n)$. This technique reduced storage needs by 400x-4000x, enabling the model to be stored in on-chip memory and reducing energy consumption by 60x-70x compared to naïve FPGA implementation [19]. Taking a different route, by developing a so-called "Block-Balanced Pruning" technique to pack a pruned weight matrix and its index information in an FPGA BRAM, Qi *et al.* managed to double inference speed compared to a GPU [23]. Another example of software-hardware co-design is the work by Zhi *et al.* who developed a "Clipped staircase ReLU" activation function optimized for their custom CiM processing element and use of quantized weights. Their work consumed 5x less energy and 100x-1000x fewer memory accesses [24].

2.3 Lessons from the literature

The literature reviews presented above offer promising design directions to help reach the design goals. Firstly, a small model is critically missing from the literature since even the smallest model is 16x too large for our project. Secondly, CiM and dataflow architectures are proven techniques to decrease power consumption and offer promising inspiration for the design of the ASIC. Finally, good software-hardware co-design should not be overlooked as it can offer significant gains.

3 Progress to date

3.1 Transformer model and edge TPU

Since August 2023, I have completed Andrew Ng's Machine Learning Specialization course to build enough knowledge to tackle the transformer model. The model is based on a vision transformer [25], which accepts a 30s epoch and return the most probable sleep stage. The architecture is shown in Figure 1. Since sleep staging as presented here is a "sequence-to-one" problem, feedback is not applicable and thus the decoder stack present in a more traditional transformer is not needed. The model is written in Python with TensorFlow due to its straightforward conversion to TensorFlow Lite, which is compatible with the Coral Edge TPU. As can be seen in Figure 1, the model has an simple averaging layer downstream of the softmax layer. This provides more stable results as they tend to oscillate between light and deep sleep stages. This feature boosted accuracy by 3%. Table 2 indicates the main hyperparameters of the model and its training. These parameters have been found through hyperparameter search on the Compute Canada cluster, where lowest total model weight and reasonable accuracy are prioritized. It should be noted that these are subject to change as the search is not so far complete. The model developed contains only 63k parameters (pre-pruning), which can be quantized to 16-bit integers with a slight gain of accuracy. The accuracy on a 31-fold validation set is 80.2 %. To help determine design priorities for the accelerator, the training script also exports the total number of different type of operations, as presented in Table 3. Clearly, the vast majority of operations are Multiplies and Additions (either as part of MACs or individually, such as in LayerNorm layers), so most time should be spent on optimizing these operations over others. Diverging from [25], the activation function chosen for the Dense layers is SWISH (instead of GeLU), as described by [26]. SWISH helps resolve the vanishing gradient problem in backpropagation, leading to a higher accuracy for a given number of training epochs. In this project, an accuracy increase of roughly 0.25% was observed when using SWISH over GeLU.

Furthermore, a script was needed to extract and preprocess raw EDF data. The script parses raw EDF files, applies signal processing, prunes unknown sleep stages and concatenates all nights into one Tensor which is saved to disk. The Tensor format is useful as it permits the use of all of TensorFlow's methods such as auto-caching, paired shuffling, etc., without having to convert at every training session. The signal processing consists of a 60Hz notch filter (to remove any coupling with AC mains), a 0.3Hz-100Hz bandpass (as recommended in [27]) and adjustable quantization (currently to 16-bit integer instead of the default 32-bit float in order to decrease disk usage and speed up saving/loading without accuracy loss). The script can also up- or downsample the signal to evaluate the impact of sampling frequency on accuracy. The MASS dataset signals are sampled at 256Hz, but the sampling frequency can be reduced to 64Hz without loss of accuracy. This can lower the power draw of the ASIC's analog front-end and decrease storage requirements by 4%-9% depending on the final model size. In addition, the clip length can be specified, again in an effort to observe whether storage can be saved or accuracy can be increased. 30s has been found as a good medium between providing enough data to accurately perform sleep staging and a short enough update period. The script also exports a "pseudo-random" EEG signal (whose samples follow $\mathcal{N}(\mu = 2^{16} * stage/(\#stages + 1), \sigma = 2000)$). This proved useful when debugging the model initially, as any functional model should have an accuracy of 100% on this Gaussian distribution centred around the sleep stage number. Finally, the data processing script is multithreaded, which cuts down processing time by 12x on the Compute Canada server (maximum of 6 processors per job, each with 2 threads).

Table 2: Hyperparameters for vision transformer model

Hyperparameter	Value
Input channel	Cz-LER
Embedding depth (d_{model})	64
# of attention heads	8
# of encoder layers	2
MLP depth	32
Patch length	256 samples
Clip length	30s
Sampling frequency	256 Hz
Output averaging depth	3 samples
Learning rate schedule	$\sqrt{d_{model}} * min(\sqrt{step}, step/4000^{1.5})$
Batch size	16
# of epochs	100
Dropout rate	30%
Class weights	1.0 $\forall$ {Wake, REM, N1, N2, N3/N4}
Optimizer	Adam
Activation function	SWISH
Data processing	60Hz notch $\rightarrow$ 0.3-100Hz bandpass $\rightarrow$ 16b quantization

Both the TensorFlow model training program and the data extraction script are fully parametrizable via bash files and take advantage of the SLURM scheduler features on the Compute Canada server, notably the array input argument jobs (automatically spawn a number of job while varying a single input argument), which is useful for hyperparameter search.

In addition, the model was ported to TensorFlow Lite to run on the Coral Edge TPU.

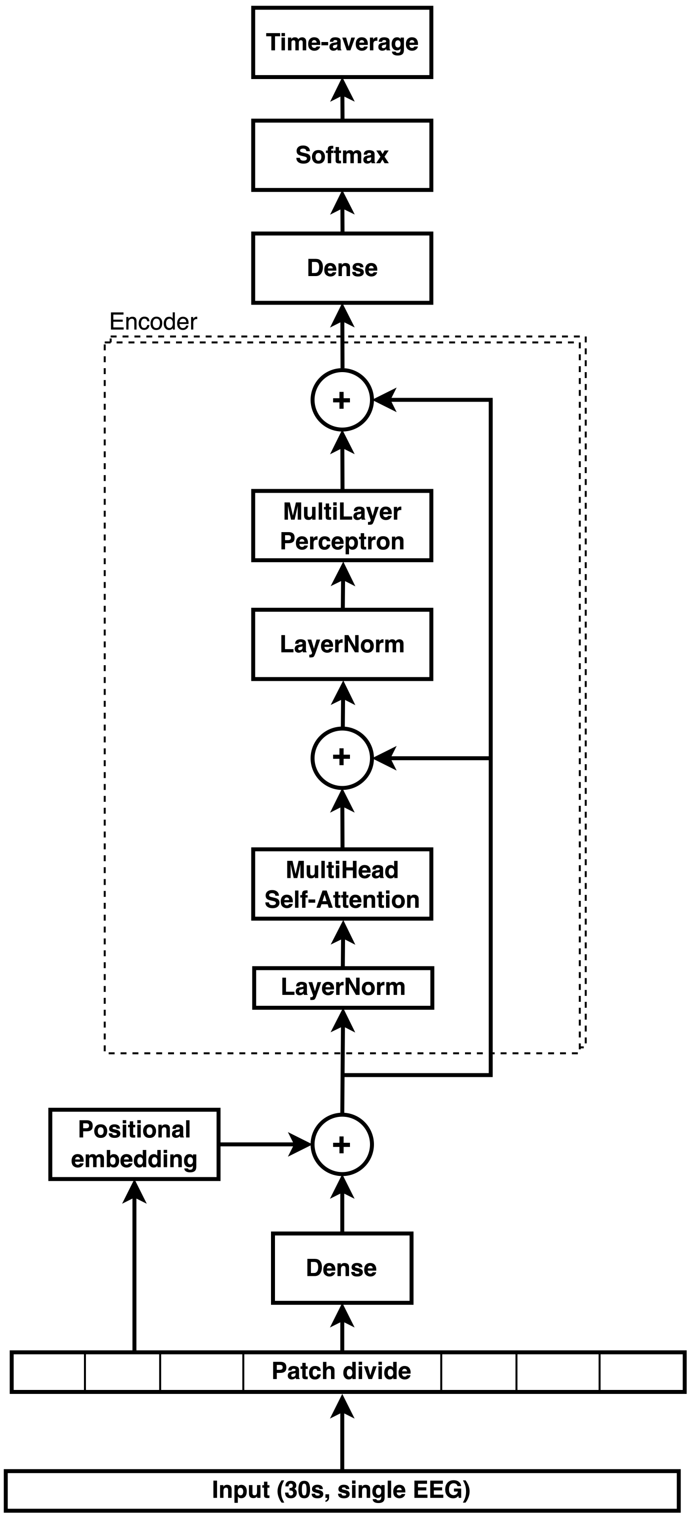

Figure 1: High-level transformer architecture for in-situ sleep staging

This provided a reference latency and power figures to size what is considered to be the most promising commercial alternative to a custom ASIC given this application. This knowledge can be useful should for a functional proof-of-concept prototype. In regular frequency mode (200MHz), the average inference time on the Edge TPU is 0.754ms. Finally, I have researched and documented initial ways the Edge TPU can be used with a microcontroller, which will be needed should we wish to develop a prototype in the future.

Table 3: Count of different types of operations in the transformer model

Operation	Total count	Percent of total
Index increment*	2475068	N/A
Multiplication	2449232	49.7%
Addition	2431056	49.3%
Division	33782	0.68%
Substraction	9920	0.20%
Activation (SWISH)**	7099	0.14%
Exponent	496	0.01%
Square root	157	0.003%

* Increments are excluded from total as they can be executed in parallel of any other element-wise operation.
** The count of elementary operations for activation depends on the approximation used in hardware.

3.2 ASIC accelerator

I have so far been granted access to the Synopsys Design Vision (DV), ModelSim and TSMC 65nm design kit on the EECG machines, and was able to simulate and synthesize a simple design by following the ECE1388 user manual in addition to becoming familiar with Synopsys's DesignWare IP library. Since Synopsys DV and ModelSim are fairly slow and overkill for modules that do not require IP, I have researched and gained familiarity with lighter weight open-source tools for Verilog logic simulation and testbenching, such as Verilator and CocoTB, which is a Python module for hardware simulations that has been rapidly gaining in popularity recently.

Finally, the architecture of the accelerator, which is shown in Figure 2, was designed. It contains 64 Compute-in-Memory modules to parallelize computations and minimize expensive memory accesses. Most of the matrices in the model have been designed to have at least one dimension equal to 64, or a multiple of 64. This maximizes utilization of the area, ultimately leading to lower area needs. Centre to each CiM is the "SuperMAC" which, in tandem with the local controller, can perform LayerNorm, Softmax and SWISH activation approximated computations in addition to Multiply-and-Accumulate. To minimize power consumption, most computations can be done fully internally, and done so

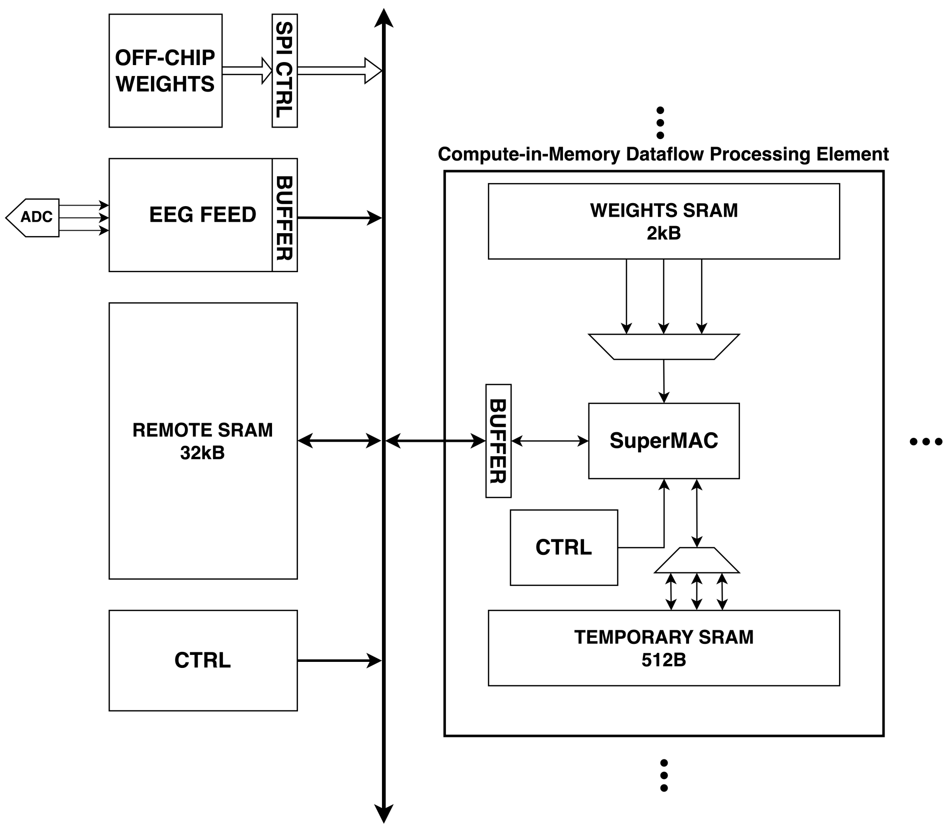

Figure 2: High-level architecture of the ASIC accelerator

without constant relying on the external controller or instructions. The weights SRAM is packed with the correct vectors/sub-matrices to proceed with computations with inter-CiM data transfer when needed. The only transfer needed occurs when a reshape and transpose is inevitable, such as within the Multi-Head Self-Attention (MHSA) layer. For this, a remote SRAM bank is used. Each CiM also contains an additional SRAM cell for temporary results storage (i.e. for residual connections or multi-stage computations such as LayerNorm). Finally, provisions are made to load weights from off-chip non-volatile storage, and from the chip's analog front-end.

4 Future work

4.1 Transformer model

The future work needed for the model is minor. There are three improvements that could further increase accuracy or reduce model size. Firstly, the model could be pre-trained on the Stanford Technology Analytics and Genomics in Sleep (STAGES [28]) dataset, which contains PSG recordings from 1500 patients, significantly more than the 62 nights available in the dataset (MASS SS3 [29]) currently used. Although MASS is widely used for model benchmarking, it is worthwhile to try pre-training the model on STAGES and perform transfer learning and k-fold validation on the MASS dataset. In addition, the model should be trained and validated on the Expanded Sleep-EDF dataset, which is the most widely used dataset in the literature [30]. Finally, weight pruning could further reduce the size of the model. Pruning consists of eliminating a subset of weights to reduce model size and compute with minimal loss of accuracy at the cost of more complicated control logic. In some cases, pruning may also increase accuracy. Indeed, Chen *et al.* applied 50% pruning on DeiT-Small, a vision transformer model, and observed a 0.28% accuracy boost.

4.2 ASIC accelerator

In order to obtain representative power, area and latency figures, the design will be synthesized using the same tools (Synopsys Design Vision) and process node as the target for the full ASIC (TSMC 65nm). Firstly, a C model will be developed to fully exercise the control logic, weight/data movement and compute approximation as those are anticipated to be the hardest part of the design and a software proof-of-concept will prove very handy in writing the RTL. Following that, the SRAM cells will be synthesized as it is expected that this synthesis will require to most research so more time rather than less should be allocated to it. The master control FSM and inter-CiM communication fabric will then be designed along with their testbench. Then, the “SuperMAC” will be developed in two phases; the basic Multiply-and-Accumulate functionality followed by the control logic to repurpose it to perform LayerNorm, Softmax and SWISH. Because AI is probabilistic in nature, there is room for area/power/latency improvements by relaxing the exactness requirement for these operations and proceeding with approximations. For instance, Schraudolph proposes an algoritm to approximate the exponential function (used in the Softmax layer) with only one multiplication, two additions and bit manipulations [31]. Finally, the weight packing architecture and process will be developed. Once all functional simulations pass and sleep staging inference equals results from the TensorFlow model, the design will be fully synthesized to provide the final power, area and latency figures.

Figure 3 plans the future work for this thesis on a week-by-week basis (starting on the first week of the year). In addition to the points discussed above, the last two thesis deliverables, namely visuals submission and final thesis report, as included.

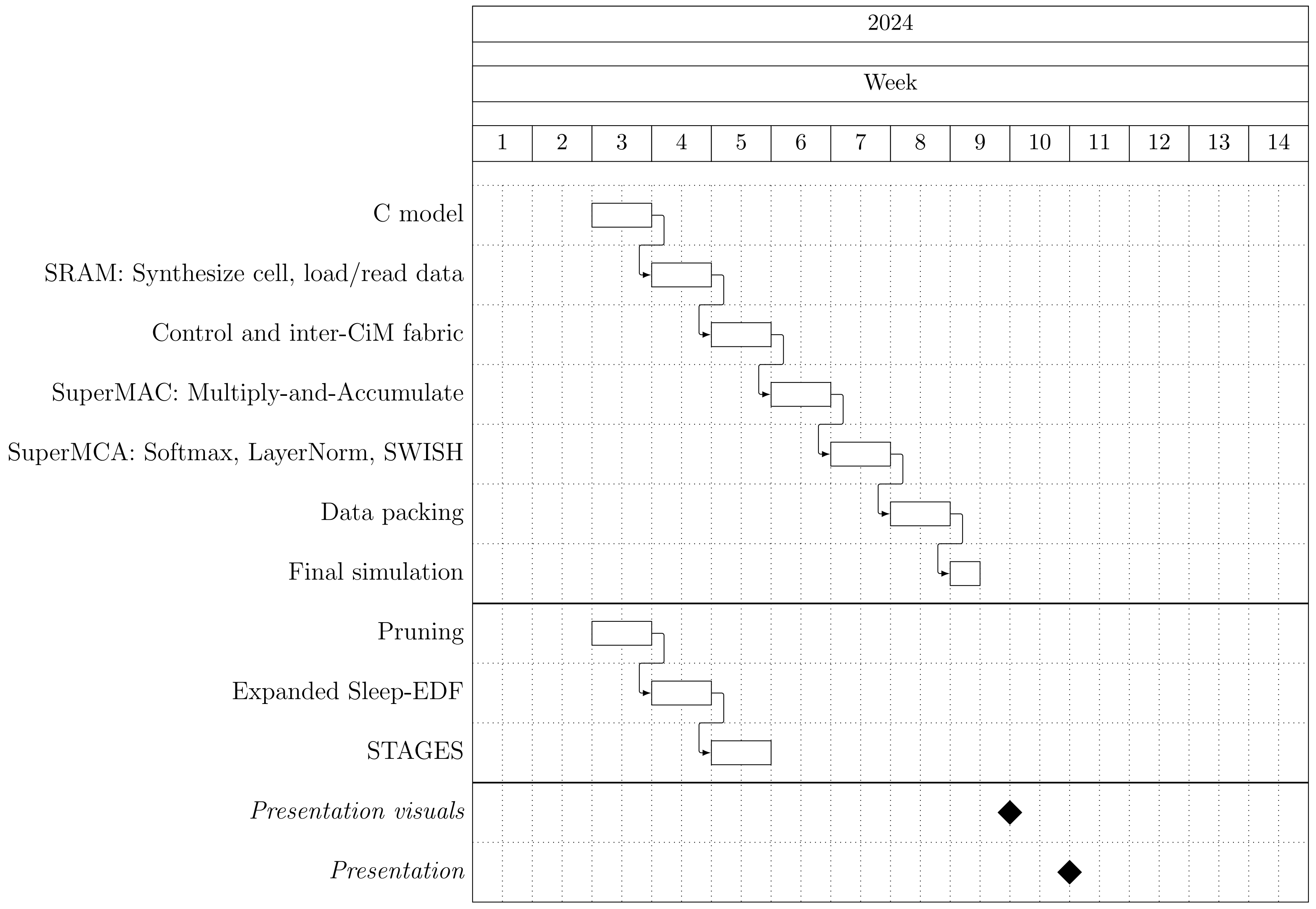

Figure 3: Future work for transformer model and ASIC accelerator

References

[1] Jean-Philippe Chaput, Jessica Yau, Deepa P. Rao, et al. "Prevalence of insomnia for Canadians aged 6 to 79". In: *Health Reports* 29.12 (2018).

[2] Ho-Kyoung Yoon. "Neuromodulation for Insomnia Management". In: *Sleep Medicine and Psychophysiology* 28.1 (2021), pp. 2–5.

[3] Jessica Vensel Rundo and Ralph Downey. "Chapter 25 - Polysomnography". In: *Clinical Neurophysiology: Basis and Technical Aspects*. Ed. by Kerry H. Levin and Patrick Chauvel. Vol. 160. Handbook of Clinical Neurology. Elsevier, 2019, pp. 381–392. DOI: `https://doi.org/10.1016/B978-0-444-64032-1.00025-4`. URL: `https://www.sciencedirect.com/science/article/pii/B9780444640321000254`.

[4] Huy Phan and Kaare Mikkelsen. "Automatic sleep staging of EEG signals: recent development, challenges, and future directions". In: *Physiological Measurement* 43.4 (2022), 04TR01.

[5] Micheal Dutt, Surender Redhu, Morten Goodwin, et al. "SleepXAI: An explainable deep learning approach for multi-class sleep stage identification". In: *Applied Intelligence* 53.13 (2023), pp. 16830–16843.

[6] Mingyu Fu, Yitian Wang, Zixin Chen, et al. "Deep learning in automatic sleep staging with a single channel electroencephalography". In: *Frontiers in Physiology* 12 (2021), p. 628502.

[7] Emadeldeen Eldele, Zhenghua Chen, Chengyu Liu, et al. "An attention-based deep learning approach for sleep stage classification with single-channel EEG". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 29 (2021), pp. 809–818.

[8] *USB Accelerator datasheet*. Version 1.4. Coral. 2019. URL: `https://coral.ai/static/files/Coral-USB-Accelerator-datasheet.pdf`.

[9] Venkatesh Gorantla. *Apple AirPods Pro 2 Battery Capacity Revealed Via 3C Listing*. Website. 2022.

[10] *Implementation of Flip Chip and Chip Scale Technology*. Joint Industry Standard, 1996.

[11] Zhiyu Liu and Volkan Kursun. "Characterization of a novel nine-transistor SRAM cell". In: *IEEE transactions on very large scale integration (VLSI) systems* 16.4 (2008), pp. 488–492.

[12] Huy Phan, Kaare Mikkelsen, Oliver Y Chén, et al. "Sleeptransformer: Automatic sleep staging with interpretability and uncertainty quantification". In: *IEEE Transactions on Biomedical Engineering* 69.8 (2022), pp. 2456–2467.

[13] Kai Han, Yunhe Wang, Hanting Chen, et al. “A survey on vision transformer”. In: *IEEE transactions on pattern analysis and machine intelligence* 45.1 (2022), pp. 87–110.

[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, et al. “Attention is all you need”. In: *Advances in neural information processing systems* 30 (2017).

[15] Yang Dai, Xiuli Li, Shanshan Liang, et al. “MultiChannelSleepNet: A Transformer-based Model for Automatic Sleep Stage Classification with PSG”. In: *IEEE Journal of Biomedical and Health Informatics* (2023).

[16] Yunxiang Hu, Yuhao Liu, and Zhuovuan Liu. “A survey on convolutional neural network accelerators: GPU, FPGA and ASIC”. In: *2022 14th International Conference on Computer Research and Development (ICCRD)*. IEEE. 2022, pp. 100–107.

[17] Tianshi Chen, Zidong Du, Ninghui Sun, et al. “Diannao: A small-footprint high-throughput accelerator for ubiquitous machine-learning”. In: *ACM SIGARCH Computer Architecture News* 42.1 (2014), pp. 269–284.

[18] Zidong Du, Robert Fasthuber, Tianshi Chen, et al. “ShiDianNao: Shifting vision processing closer to the sensor”. In: *Proceedings of the 42nd Annual International Symposium on Computer Architecture*. 2015, pp. 92–104.

[19] Caiwen Ding, Siyu Liao, Yanzhi Wang, et al. “Circnn: accelerating and compressing deep neural networks using block-circulant weight matrices”. In: *Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture*. 2017, pp. 395–408.

[20] Aman Arora, Tanmay Anand, Aatman Borda, et al. “CoMeFa: Compute-in-Memory Blocks for FPGAs”. In: *2022 IEEE 30th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*. IEEE. 2022, pp. 1–9.

[21] Xiaowei Wang, Vidushi Goyal, Jiecao Yu, et al. “Compute-capable block RAMs for efficient deep learning acceleration on FPGAs”. In: *2021 IEEE 29th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*. IEEE. 2021, pp. 88–96.

[22] Clément Farabet, Yann Lecun, Koray Kavukcuoglu, et al. “Large-Scale FPGA-Based Convolutional Networks”. In: *Scaling up Machine Learning: Parallel and Distributed Approaches*. Ed. by Ron Bekkerman, Mikhail Bilenko, and JohnEditors Langford. Cambridge University Press, 2011, pp. 399–419. DOI: `10.1017/CBO9781139042918.020`.

[23] Panjie Qi, Yuhong Song, Hongwu Peng, et al. "Accommodating transformer onto FPGA: Coupling the balanced model compression and fpga-implementation optimization". In: *Proceedings of the 2021 on Great Lakes Symposium on VLSI*. 2021, pp. 163–168.

[24] Ruoyu Zhi, Ryan Jurasek, Wolfgang Hokenmaier, et al. "Opportunities and Limitations of in-Memory Multiply-and-Accumulate Arrays". In: *2021 IEEE Microelectronics Design & Test Symposium (MDTS)*. IEEE. 2021, pp. 1–6.

[25] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, et al. "An image is worth 16x16 words: Transformers for image recognition at scale. arXiv 2020". In: *arXiv preprint arXiv:2010.11929* (2010).

[26] Prajit Ramachandran, Barret Zoph, and Quoc V Le. "Searching for activation functions". In: *arXiv preprint arXiv:1710.05941* (2017).

[27] Akara Supratak, Hao Dong, Chao Wu, et al. "DeepSleepNet: A model for automatic sleep stage scoring based on raw single-channel EEG". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 25.11 (2017), pp. 1998–2008.

[28] Guo-Qiang Zhang, Licong Cui, Remo Mueller, et al. "The National Sleep Research Resource: towards a sleep data commons". In: *Journal of the American Medical Informatics Association* 25.10 (2018), pp. 1351–1358.

[29] CEAMS. *SS3 Biosignals and Sleep stages*. Version V1. 2022. DOI: `10.5683/SP3/9MYUCS`. URL: `https://doi.org/10.5683/SP3/9MYUCS`.

[30] PhysioToolkit PhysioBank. "Physionet: components of a new research resource for complex physiologic signals". In: *Circulation* 101.23 (2000), e215–e220.

[31] Nicol N Schraudolph. "A fast, compact approximation of the exponential function". In: *Neural Computation* 11.4 (1999), pp. 853–862.