This article has been accepted for publication in IEEE Access. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2025.3538781

IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.Doi Number

Reconfigurable and Scalable Artificial Intelligence Acceleration Hardware Architecture with RISC-V CNN Coprocessor for Real-time Seizure Detection

SHUENN-YUH LEE [1], (Senior Member, IEEE), Ming-Yueh Ku[1], Sing-Yu Pan[1], Chou-Ching Lin[2]

[1] Department of Electrical Engineering, National Cheng Kung University, Tainan City 70101, Taiwan (R.O.C.)
[2] Division of Neurology, National Cheng-Kung University Hospital, Tainan City 70101, Taiwan (R.O.C.)

Corresponding authors: Shuenn-Yuh Lee (e-mail: ieesyl@mail.ncku.edu.tw)

This research was supported in part by the Taiwan Semiconductor Research Institute, the National Science and Technology Council (NSTC), Taiwan, R.O.C., under Grants NSTC 112-2221-E-006-181 MY3, NSTC 113-2640-E-006-006, and NSTC 113-2640-B-006-004.

ABSTRACT Epilepsy is a neurological disorder characterized by recurrent seizures. These seizures are caused by abnormal electrical activity in the brain. Seizures are often accompanied by involuntary partial or whole-body convulsions, frothing at the mouth, and possible loss of consciousness, putting a patient at high risk. Electroencephalogram (EEG) can be used to diagnose epilepsy. This study proposes a seizure detection algorithm to identify a seizure attack with EEG. This algorithm includes a simplified signal preprocessor and a nearly optimized convolutional neural network (CNN). This study also proposes an artificial intelligence acceleration (AIA) hardware architecture, including a deep learning accelerator (DLA) and a two-stage reduced instruction set computer-V (RISC-V) central control unit (CPU), to implement the detection algorithm in real-time operation. The accelerator is implemented in System-Verilog and validated on the Xilinx PYNQ-Z2 Field Programmable Gate Array (FPGA) board. The implementation consumes 3535 lookup tables, 2283 flip-flops, 28 KB of block random-access memory, six digital signal processors, and seven input/output (I/O). The total power consumption is 0.108 W in 1-MHz operation frequency. The detection algorithm provides 99.06% accuracy on fixed-point operations with a detection latency of 128 ms/class. The application-specific integrated circuit (ASIC) performance of the AIA hardware architecture is also tested with a 180 nm 1P6M process. The total power of the AIA is 1.29 mW. The core circuit of the RISC-V CPU and DLA consumes 80 μW and 84.5 μW, respectively. Moreover, the AIA can be reconfigurable. Thus, the accelerator can execute different deep-learning models to fit various wearable applications for biomedical acquisition systems.

INDEX TERMS Electroencephalography, bio-signal processing, seizure detection, artificial intelligence, convolutional neural network, reduced instruction set computer-V, reconfigurable hardware acceleration.

I. INTRODUCTION

According to the World Health Organization, approximately 50 million people worldwide have epilepsy [1]. Epilepsy, a noninfectious disease occurring in all age groups, can be caused by brain injuries, encephalitis, or genetic factors, among other things [2]. However, the cause is unknown in approximately 50% of patients. When an epileptic seizure occurs, the neurons in the brain begin to discharge abnormally, causing muscle spasms, foaming at the mouth, hallucinations, or loss of consciousness. The duration of each seizure can range from a few seconds to 5 min. The frequency of seizures can range from several times a day to several years.

Clinically, epilepsy is often diagnosed using an electroencephalogram (EEG) to evaluate the patient's condition [3]. Figure 1 illustrates the EEG waveform of normal and epileptic seizures. EEG is a technique that uses electrodes placed on the scalp or inside the skull to measure

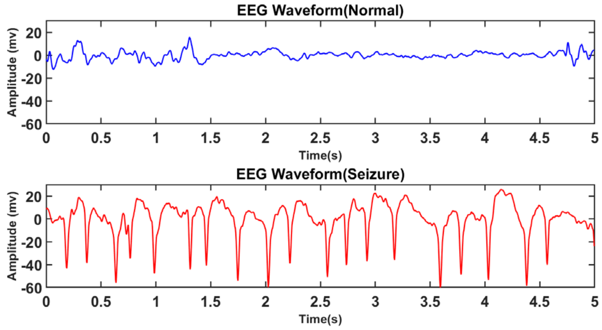


FIGURE 1. **EEG waveform of normal and epileptic seizures.**

the electrical activity of the brain. A waveform of the voltage differences between the electrodes is produced. The time and location of an epileptic seizure can be detected by recording and observing changes with EEG. Continuously measuring the brain's electrical activity using a wearable device can allow patients to monitor their condition for long periods, thereby making real-time detection of seizures possible [4]. According to artificial intelligence (AI) development, seizure attack measurement is a potential detection method.

A neural network is a type of AI model inspired by the structure and function of the human brain. The concept of deep learning was proposed by Hinton in 2006 [5]. Deep learning significantly increases the model's ability to learn features and predict accurately by stacking more than two layers of neural networks in an AI model. Among them, a convolutional neural network (CNN) can learn spatial or temporal features of the input through convolutional operations; it is also suitable for extracting local features of images or physiological signals, such as edges, corners, and turns [6]. Therefore, it is often used in image recognition or disease detection [7]. If a CNN model can be used to provide appropriate treatment in real-time by detecting epileptic seizures, it can protect the safety and quality of life of patients. Accurately recording the duration of seizures can also help with the subsequent tracking and treatment of epilepsy.

A neural network algorithm has thousands to millions of multiplication or addition operations. Moreover, the calculation process requires frequent memory access. If a general processor is used for calculation, considerable time and energy are taken. Thus, the demands of wearable devices and real-time detection cannot be met. In a previous work [8], a reduced instruction set computer-V (RISC-V) CNN coprocessor was proposed for the real-time detection of epilepsy. A special coprocessor was designed to complete a large number of multiplication and addition operations in CNN and achieve the requirement for the real-time identification of epilepsy. However, the signal preprocessing algorithm relies on other processors to complete, and the entire epilepsy recognition algorithm is not fully implemented on edge. Moreover, the cooperative processor can calculate only the fixed-structure CNN model, limiting the types of models that the cooperative processor can accelerate and reducing the usability of the architecture. Hence, we extended our previous work [8] in the present study with the following contributions:

1. Reduction of the algorithm's complexity while improving accuracy according to the proposed algorithm compilation.
2. Implementing signal preprocessing with RISC-V controller to achieve a complete algorithm for seizure detection.
3. Proposing a hardware architecture to make the CNN accelerator reconfigurable.

Many reconfigurable accelerators have been proposed; however, most focus on image classification with a quite large model [9]. This type of accelerator can achieve a very large amount of calculation. Therefore, much energy is consumed. Moreover, the flow of program control is rarely discussed. The accelerators for biomedical applications usually aim at the highest efficiency with a fixed application-specific integrated circuit (ASIC) design. The calculation process is usually not easy to modify after the hardware implementation. If a reconfigurable accelerator for biomedical applications exists [10], developers can apply the algorithm to different domains without redesigning the hardware. Hence, the proposed hardware architecture is also designed to be reconfigurable, thereby allowing it to adapt to many biomedical-related applications.

This paper is structured as follows. Section II describes the overview of the seizure detection system, the proposed seizure detection algorithm, and how the algorithm is converted into a configuration file for setting up the hardware. Section III represents the hardware architecture of the AI accelerator (AIA), including the system executing flow and the computation units. Section IV shows the experiment results of the algorithm and AIA. Finally, Section V concludes the research.

II. SYSTEM STRUCTURE AND THE SEIZURE DETECTION ALGORITHM

Figure 2 shows the proposed system structure for the real-time seizure detection application with three main parts, including the seizure detection algorithm, configuration file, and reconfigurable AIA hardware architecture. The seizure detection algorithm with the CNN model is first developed and validated using the software platform. The optimized CNN model is trained to reach sufficient accuracy in our Lab database [4][8]. The algorithm can be compiled and transformed into the configuration file after the algorithm is validated with the fivefold cross-validation [11]. The configuration file with RISC-V binary code, layer memory configuration, layer instruction, and layer data can be used to initialize the AIA with the serial peripheral interface (SPI). The main controller in the AIA starts to compute the seizure detection algorithm after a start command is received. The main controller of the AIA fetches the layer instructions and dispatches the task to the computing core, such as the deep

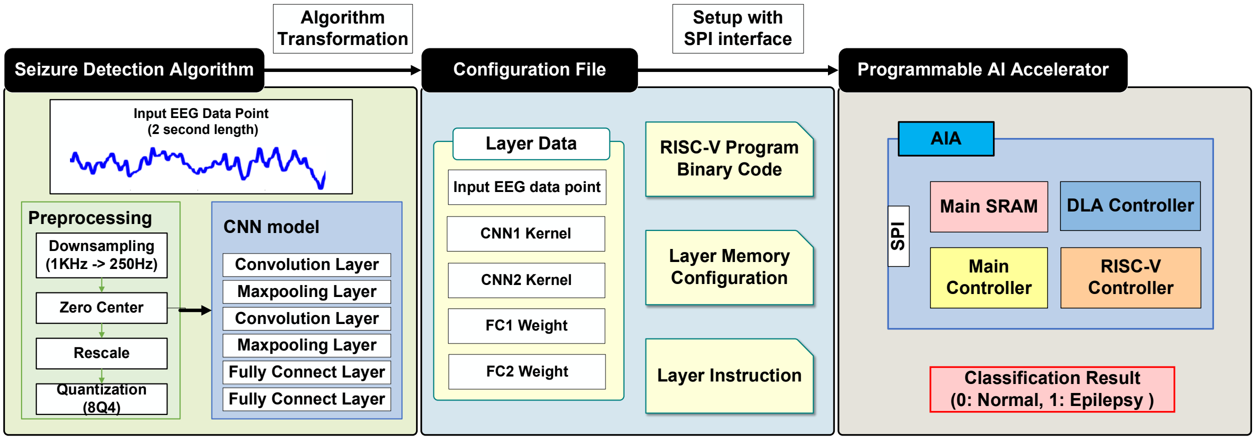


FIGURE 2. Proposed system structure of seizure detection application.

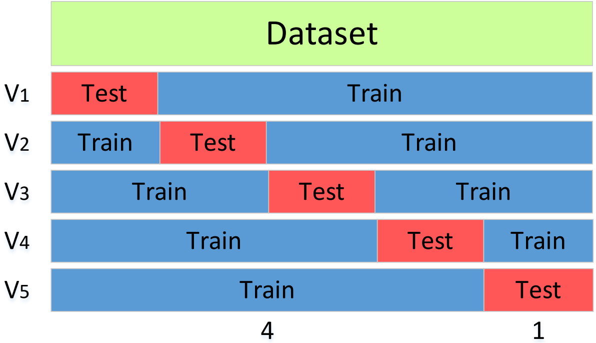


FIGURE 3. Fivefold cross-validation.

learning accelerator (DLA) controller or RISC-V controller. When the result has been calculated, the AIA sets the signal to high to indicate that the result is ready. The output has two possible values: 0 for normal EEG and 1 for epilepsy EEG.

A. ANIMAL EEG DATABASE

The proposed algorithm is validated with the animal EEG database elaborated in our previous research [4][8]. The database contains the data from the dual-channel intracranial EEG from six rats. The EEG signal is first amplified 500 times by a front-end amplifier and passed through a bandpass filter with a frequency band of 2.87–100 Hz. The signal is sampled with a 12-bit analog-to-digital converter with a resolution of 1.6 μV per bit. The sampling frequency is 1,000 Hz. The training and testing data used in this research include 8,340 s-long EEG data, of which 4,770 s is marked as seizure segments. The ratio between the normal and epileptic EEG data is approximately 3:4.

In this study, the algorithm is validated with the fivefold cross-validation. Figure 3 shows how the dataset is split into the training and testing datasets in each validation. The ratio of the training and testing data is 4:1. The validation is repeated five times to generate an average accuracy. Accuracy is the performance metric of the algorithm and the indicators to improve the design strategy.

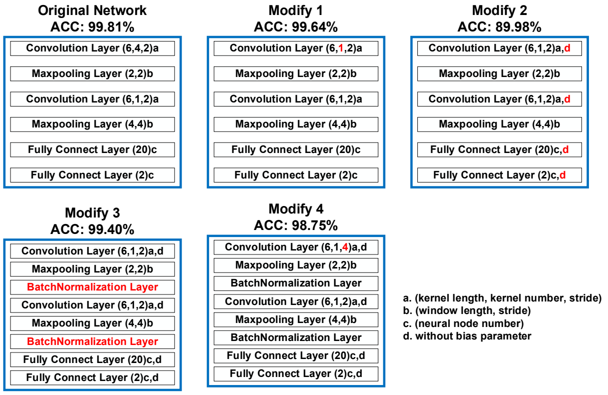

Original Network
ACC: 99.81%
Convolution Layer (6,4,2)a
Maxpooling Layer (2,2)b
Convolution Layer (6,1,2)a
Maxpooling Layer (4,4)b
Fully Connect Layer (20)c
Fully Connect Layer (2)c
Modify 1
ACC: 99.64%
Convolution Layer (6,1,2)a
Maxpooling Layer (2,2)b
Convolution Layer (6,1,2)a
Maxpooling Layer (4,4)b
Fully Connect Layer (20)c
Fully Connect Layer (2)c
Modify 2
ACC: 89.98%
Convolution Layer (6,1,2)a,d
Maxpooling Layer (2,2)b
Convolution Layer (6,1,2)a,d
Maxpooling Layer (4,4)b
Fully Connect Layer (20)c,d
Fully Connect Layer (2)c,d
Modify 3
ACC: 99.40%
Convolution Layer (6,1,2)a,d
Maxpooling Layer (2,2)b
BatchNormalization Layer
Convolution Layer (6,1,2)a,d
Maxpooling Layer (4,4)b
BatchNormalization Layer
Fully Connect Layer (20)c,d
Fully Connect Layer (2)c,d
Modify 4
ACC: 98.75%
Convolution Layer (6,1,4)a,d
Maxpooling Layer (2,2)b
BatchNormalization Layer
Convolution Layer (6,1,2)a,d
Maxpooling Layer (4,4)b
BatchNormalization Layer
Fully Connect Layer (20)c,d
Fully Connect Layer (2)c,d
a. (kernel length, kernel number, stride)
b. (window length, stride)
c. (neural node number)
d. without bias parameter

FIGURE 4. Comparison of different model structures.

TABLE 1. Comparison table of different models.

Model Structure	Number of parameters (Weight/Bias)	Number of multiplication	Accuracy
Original	388/27	7732	99.81%
Modify 1	352/24	2188	99.64%
Modify 2	352/0	2188	89.98%
Modify 3	**352/2**	**2188**	**99.40%**
Modify 4	352/2	1098	98.75%

B. SIGNAL PROCESSING

The EEG signal is adjusted to fit the input format of the CNN model and enhance the learning accuracy. Given that EEG is a nonperiodic physiological signal, a short input frame is often chosen [12]. In the present research, a 2 s-long EEG signal with 2000 data points (X) is used as the input frame of the algorithm. The input frame is first downsampled to 250 Hz to decrease the complexity of the CNN model. Then, the downsampled signal is applied with the zero mean processes to cancel the individual difference between rats. The zero mean processes can be expressed as

$$X_m = X - \mu, \quad (1)$$

where μ is the mean of the downsampled EEG signal.

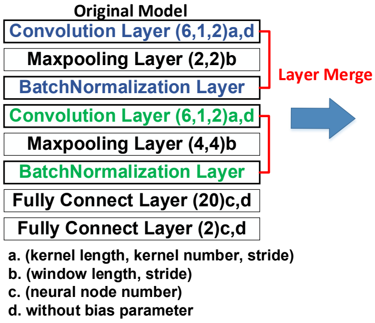

Original Model
Convolution Layer (6,1,2)a,d
Maxpooling Layer (2,2)b
BatchNormalization Layer
Convolution Layer (6,1,2)a,d
Maxpooling Layer (4,4)b
BatchNormalization Layer
Fully Connect Layer (20)c,d
Fully Connect Layer (2)c,d
Layer Merge
Model After Layer Merge
Convolution Layer (6,1,4)a
Maxpooling Layer (2,2)b
Convolution Layer (6,1,2)a
Maxpooling Layer (4,4)b
Fully Connect Layer (20)c,d
Fully Connect Layer (2)c,d
a. (kernel length, kernel number, stride)
b. (window length, stride)
c. (neural node number)
d. without bias parameter

FIGURE 5. Proposed CNN model with the layer merge method.

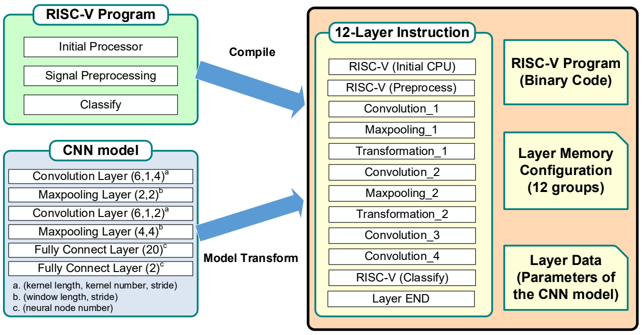


FIGURE 6. Conversion of the detection algorithm to the configuration file.

C. PROPOSED CNN MODEL

The CNN model is modified and optimized based on [8]. The optimization target is to reduce the amount of computation and maintain the accuracy of the model. Five different model structures were tested, as shown in Figure 4. The differences between the previous and current modifications are highlighted. Modify 1 reduces the number of kernels in the first CONV layer, resulting in a 75% operation reduction of that layer. Modify 2 attempts to remove the bias from the CONV and FC layers, resulting in a 9.66% decrease in accuracy. Since this degradation is not acceptable, Modify 3 introduces batch normalization (BN) layers to improve accuracy. Thanks to layer fusion, the BN layer does not add any extra computational overhead. Modify 4 increases the stride of the CONV layer to reduce computational load, but this comes at the cost of lower accuracy. After evaluating several trade-offs, we selected Modify 3 as the final model. A summary of the different modifications, including the number of parameters, multiplication operations, and accuracy, is provided in Table 1. The left part of Figure 5 shows the details of the proposed CNN model with the layer merge method. The feature extractor used in this study includes two convolutional layers, two max-pooling layers, and two batch normalization layers. The input window of the model is 2 s with 500 EEG data points. Each of the two convolutional layers contains 1 kernel with a length of 6. The stride of the two convolution layers is 2, followed by a max-pooling layer to concentrate the output features. The composition of the classifier is fully connected layers that can generate the seizure detection result. The activation function of the convolution layer and the fully connected layer are selected as a rectified linear unit (ReLU) to increase the nonlinearity of the model. The ReLU function can be expressed as

$$ReLU(x) = \begin{cases} 0\,, & x \le 0 \\ x\,, & x > 0^{\cdot} \end{cases} \qquad (2)$$

The batch normalization layer can limit the output range and decrease the possibility of failed training. The output of the batch normalization layer can be computed according to Equation (3). u_B and σ are batch mean and variance, respectively [14]. γ and β are the trainable variables in the batch normalization layer. ϵ is a constant preventing the denominator from being 0, and the value used in this research is 10^{-5}. After the training process, the batch and its variance can be obtained by passing through the training data again. All the parameters are constant after the training process. Therefore, Equation (3) can be simplified to Equation (4) with a gain of a and a constant of B.

TABLE 2. Comparison table of different data formats.

Data format	Bit-width	Accuracy
Floating-point	32-bit	99.81%
Fix-point: 32Q16	32-bit	99.64%
Fix-point: 16Q8	16-bit	89.98%
Fix-point: 8Q4	**8-bit**	**99.40%**
Fix-point: 4Q2	4-bit	98.75%

$$y = \left(\frac{\gamma}{\sqrt{\sigma^2 + \epsilon}}\right) X_i + (\beta - \frac{\gamma u_B}{\sqrt{\sigma^2 + \epsilon}}) \qquad (3)$$

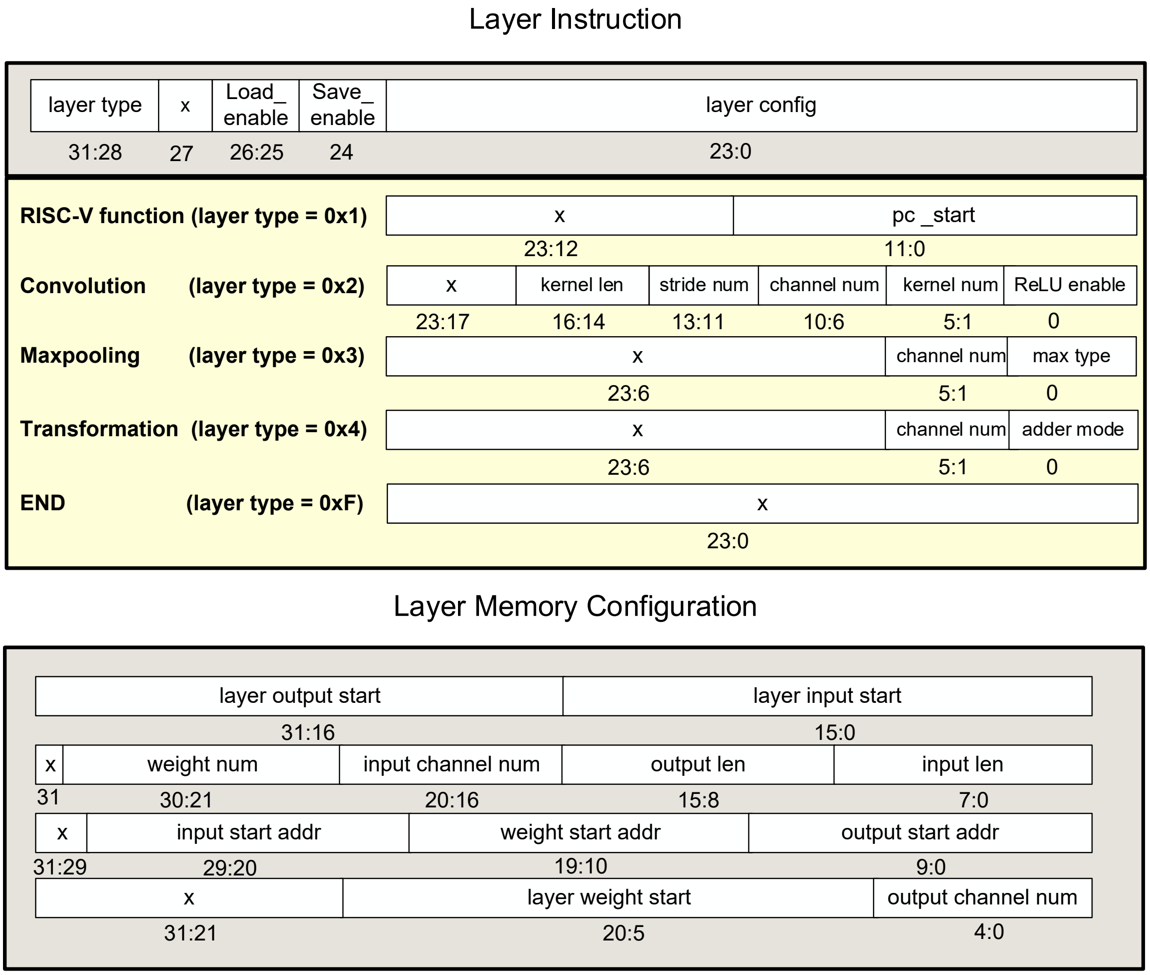
$$y = aX_i + B \qquad (4)$$

Convolutional layers can be combined with batch normalization layers when two conditions are met. The first condition: all the biases in the convolutional kernel must be 0. The second condition: the constant "a" in Equation (4) must be greater than 0. The reason for satisfying these two conditions is to maintain the consistency of the ReLU operation result. If two conditions are met, the new weight and the new bias can be expressed in Equations (5) and (6), whereas "a" and "B" are the constants in Equation (4). According to Figure 5, the number of layers in the CNN model can be decreased from 8 to 6 after the proposed layer merge technique is used. The merged model can save the operation of the original batch normalization layer. Thus, the computational complexity of the model can be further reduced.

$$W_{new} = aW \qquad (5)$$

$$b_{new} = B \qquad (6)$$

D. ALGORITHM COMPILATION



Layer Instruction

layer type	x	Load_ enable	Save_ enable	layer config
31:28	27	26:25	24	23:0

RISC-V function (layer type = 0x1)

x	pc _start
23:12	11:0

Convolution (layer type = 0x2)

x	kernel len	stride num	channel num	kernel num	ReLU enable
23:17	16:14	13:11	10:6	5:1	0

Maxpooling (layer type = 0x3)

x	channel num	max type
23:6	5:1	0

Transformation (layer type = 0x4)

x	channel num	adder mode
23:6	5:1	0

END (layer type = 0xF)

x
23:0

Layer Memory Configuration

layer output start	layer input start
31:16	15:0

x	weight num	input channel num	output len	input len
31	30:21	20:16	15:8	7:0

x	input start addr	weight start addr	output start addr
31:29	29:20	19:10	9:0

x	layer weight start	output channel num
31:21	20:5	4:0

FIGURE 7. Format of layer instruction and layer memory configuration.

The floating-point number format is used to train the learnable parameters in the CNN model of the training process. However, the hardware multiplication unit for the floating-point number is more complex to implement than the fixed-point multiplier. Therefore, the model is quantized into the fixed-point number to save the area and power consumption of the arithmetic modules in the hardware implementation. Several data formats are tested, and the results are summarized in Table 2. Finally, we chose 8Q4 format for acceptable accuracy and the least data storage, where 8 indicates that the data are represented by a total of 8 bits, and 4 indicates the four decimal places. Compared with the previous work with 32Q16 format [8], the proposed 8Q4 format can save four times the memory space with higher accuracy. The data format for each layer's input and output is also designed in 8Q4 to standardize the memory access strategy.

Figure 6 shows the flowchart for converting an algorithm into the hardware configuration file. The calculation process of the algorithm is implemented with two components: the RISC-V programs and the CNN model. A layer depicts a stage of the algorithm flow, including signal preprocessing, to simplify the description. The components can be converted into the configuration file with 12-layer instructions, 12 groups of layer memory configuration, binary codes of RISC-V programs, and the parameters of the CNN model (layer data). The configuration file can be used to set up and initialize the AIA to execute the algorithm computation.

Figure 7 represents the format of the layer instruction and the layer memory configuration. The layer instruction defines the computation process of the algorithm that AIA should start to execute. Each layer instruction, combined with a group of layer memory configurations, defines the memory settings of the operation. The lengths of a layer instruction and a group of layer memory configurations are 32 and 128 bits, respectively. The required calculation method is defined by the layer type in the layer instruction. At present, five-layer types exist: RISC-V function, convolution, max pooling, transformation, and END. The length of a layer type is 4 bits. Hence, 11 layer types are reserved for future application usage. The RISC-V program is also defined as a layer type (RISC-V function), which means that the time to start executing the RISC-V program

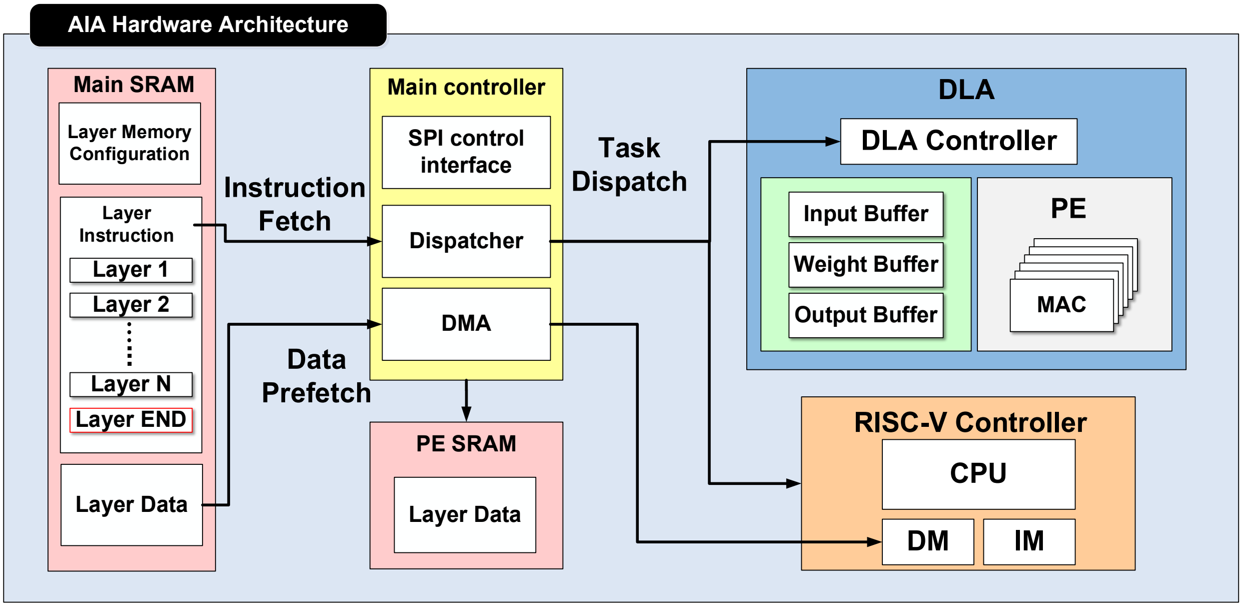


FIGURE 8. Proposed AIA hardware architecture.

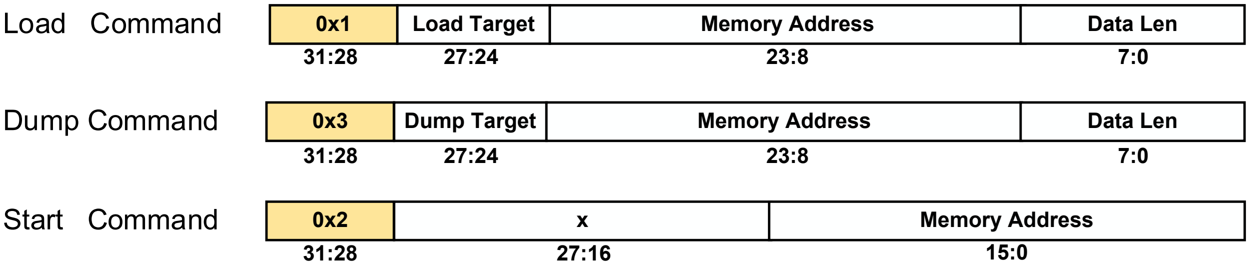


FIGURE 9. Format of SPI command.

is controlled by the main controller of AIA. Therefore, the algorithm computation processes are completely managed by the main controller. The RISC-V central control unit (CPU) is only used as a computation unit, responsible for receiving tasks assigned from the main controller.

The “load enable” in the layer instruction indicates whether prefetching the data from the main memory to the local memory of the target computation unit is required. Moreover, the “save_enable” in the layer instruction defines whether the result needs to be written back from the local memory to the main memory. The rest of the layer instruction is used to define the reconfigurable settings of this layer. For example, the starting position of the program counter (pc_start) is defined in the RISC-V function layer, and the convolution layer defines the size of the convolution kernel.

The layer memory configuration defines the memory settings for the layer instruction, including the length and address of the input and output. The advantage of representing data as memory pointers can reduce data movement in memory. If the data are moved because of the calculation, changing the value setting of the memory address without moving the data in the memory is necessary. The configuration definition of each block in the layer

TABLE 3. The block definition of layer memory configuration.

Block name	Bit length	Definition
layer output start	16	The first address of the output data in the main memory
layer input start	16	The first address of the input data in the main memory
layer weight start	16	The first address of the weight data in the main memory
weight num	10	The parameter number of the layer
input channel num	5	The channel number of the input data
output channel num	5	The channel number of the output data
output len	8	The length of the output data
input len	8	The length of the input data
input start addr	10	The first address of the input data in the local memory
weight start addr	10	The first address of the weight data in the local memory
output start addr	10	The first address of the output data in the local memory

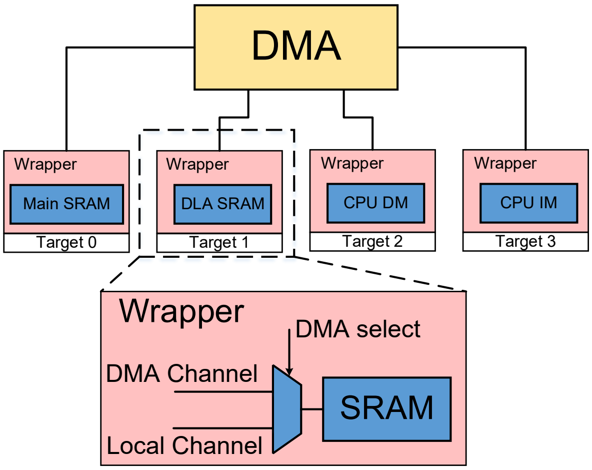


FIGURE 10. **Structure of DMA with each memory in AIA.**

memory can be found in Table 3 to reduce the data movement.

III. AIA HARDWARE ARCHITECTURE

In previous works, two types of AI accelerator hardware architecture are presented [15]-[17]. The first architecture is used to design the most efficient accelerator for a specific application [15][16]. This type of accelerator can complete the algorithm the fastest. However, the process that the accelerator can execute is relatively fixed, and it is difficult to be compatible with other application algorithms. The second architecture is reconfigurable [17]. The accelerator can calculate various AI models through different configuration files. Although its efficiency is not as high as the efficiency of the former, the reconfigurable feature of the accelerator enables it to support different algorithms. In this study, the second architecture is employed to implement the AIA combined with the proposed RISC-V coprocessor to enhance efficiency. The architecture can realize various AI algorithms without changing the hardware design by combining different layer instructions and layer memory configurations proposed in this research.

Figure 8 shows the proposed hardware architecture of the AIA. The main controller of the AIA has three modules: an SPI control interface, a dispatcher, and direct memory access (DMA). The AIA has two computation units to execute the computing task for completing the algorithm. The RISC-V CPU can execute general-purpose programs suitable for performing general operations in the algorithm, such as zero center, data moving, and sorting. The DLA can accelerate the computation flow of the AI model to meet the requirement of real-time detection.

A. MAIN CONTROLLER

In the hardware architecture, the main controller is responsible for three tasks: the control of SPI reading and writing, the data transfer in memory, and the task assignment of algorithm processes. The main controller acts as the SPI slave, which can be accessed with commands from other devices, such as an internet-of-thing device. Figure 9 represents the SPI command format. SPI commands have three kinds. The load command is used to upload the data to the memory in the AIA. A command may be followed by up to 256 words, each of which is 32-bit long. The dump command is used to download data from the memory of the hardware architecture, which can be used to confirm whether the algorithm in the hardware architecture is calculated correctly. After the hardware has completed the configuration setting with the load command or dump command, the start command can signal the AIA to start the execution of the algorithm.

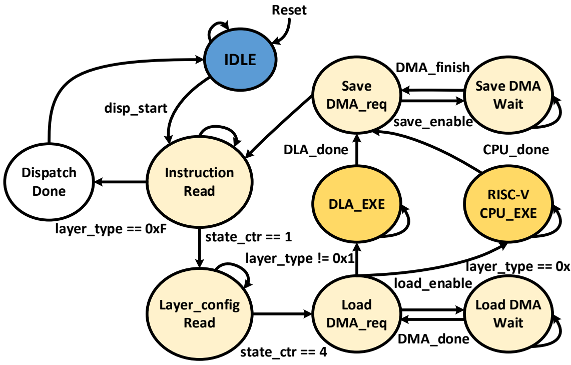


FIGURE 11. **FSM diagram of the dispatcher.**

The connecting diagram of DMA with each memory in AIA is shown in Figure 10. The DMA is used to move a large amount of data between memories without the intervention of the RISC-V CPU. The AIA hardware architecture has four memories: a main memory, a DLA memory, a CPU data memory (DM), and a CPU instruction memory (IM). Every memory has a wrapper that can be alternated between the local and DMA channels for different data access conditions.

Figure 11 represents the finite state machine (FSM) diagram of the dispatcher in the main controller. The purpose of the dispatcher is to manage the algorithm flow to generate the detection result. After the AIA receives the start command, the main controller sends a “disp_start” signal to instruction read to start the algorithm execution. In the beginning, the dispatcher fetches and decodes the layer instruction and the layer memory configuration from the main memory. If the load enabled in the layer instruction is set as one, the dispatcher prefetches the data from the main memory to the target memory defined by the layer type. If the load enabled is zero or the data prefetch phase is finished, the dispatcher assigns the task defined by the layer type to the target controller (DLA controller or the RISC-V CPU controller). At this moment, the target controller can access the DMA and the arithmetic unit in the AIA to calculate the result of the layer. After the execution of the target controller is finished, the dispatcher checks the “save_enable” in the layer instruction. If the “save_enable” is set as one, the dispatcher first writes back the result from the main memory. The dispatcher repeats the steps above until it fetches the END layer instruction. The

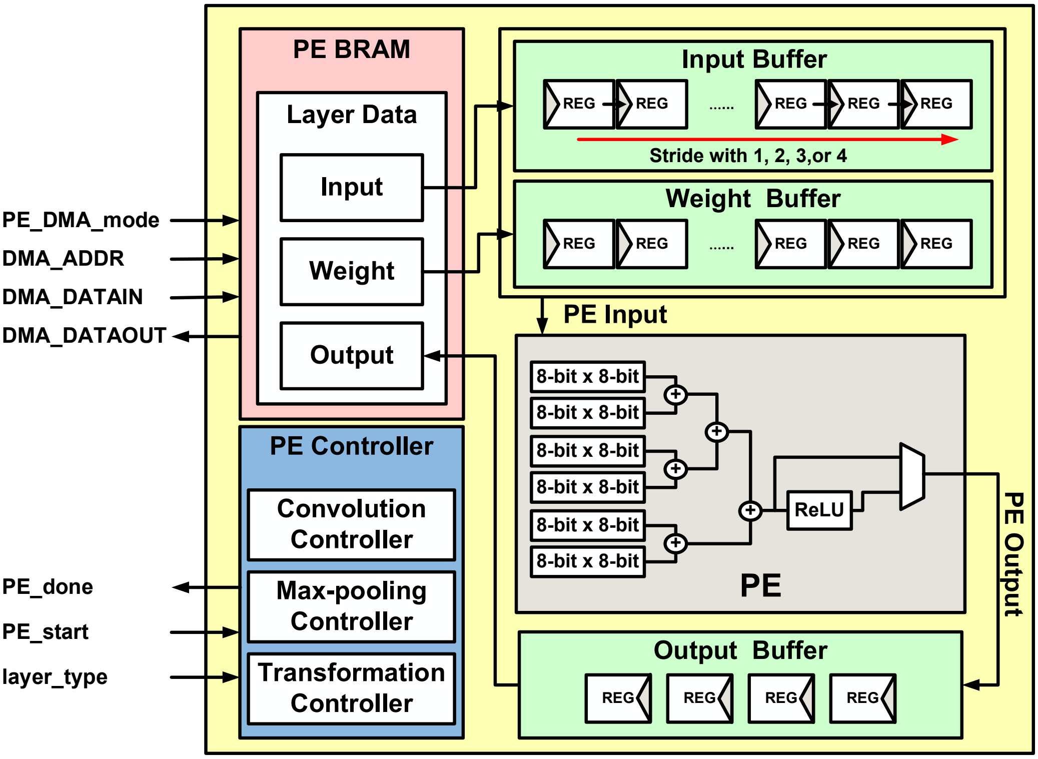


FIGURE 12. DLA hardware architecture.

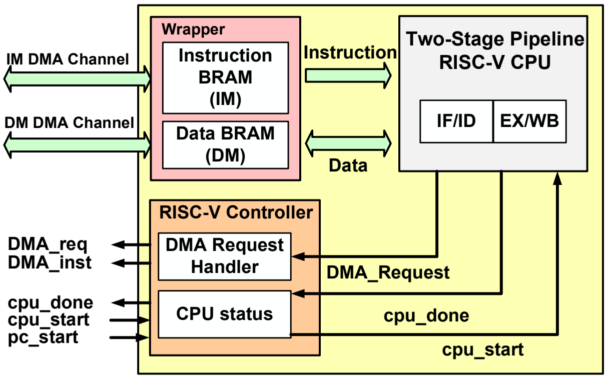


FIGURE 13. Proposed structure of the RISC-V system.

dispatcher pulls the "dispatch_done" signal as high to indicate that the algorithm is finished. Then, it turns back to the IDLE state.

B. STRUCTURE OF DLA

As shown in Figure 12, the DLA hardware architecture consists of a processing element (PE) block random-access memory (BRAM), PE controller, input/weight/output buffer, and PE. The dispatcher instructs the DLA to begin the computation process. Three subcontrollers, including convolution, max pooling, and transformation in the DLA, manage the calculating acceleration flow. These three subcontrollers are also reconfigurable. The operating function, limits of input, and reconfigurable parameters of the three subcontrollers are listed in Table 4. The DLA controller decides to use the subcontroller for the calculation based on the layer type.

The DLA manages the PE and data buffers to accelerate the AI computation process. The PE is composed of six multipliers and an adder tree. The reason we selected six multipliers is to align with the model's design. Through software-hardware co-design, we were able to reduce memory access and simplify the control path, resulting in a low-power design. Additionally, by setting different Layer Instructions, the multipliers can be gated, allowing control over the number of active multipliers. In the DLA, three data buffers are implemented, including input buffer, weight buffer, and output buffer, which can decrease the memory access time and make the computation efficient. The input buffer is designed as a first-in, first-out (FIFO) system to reuse the overlap data in the convolution operation. The weight buffer is used in reusing the weight data for the same kernel of the convolution layer. The output buffer receives the result from the PE. The size of the output buffer is equal to the maximum length that the memory can write at one time. When the output buffer is full, all results are written to the DLA memory simultaneously to save the number of memory access. When the subcontroller finishes all calculations, the DLA controller raises the DLA done signal to alert the dispatcher that the layer operation is complete.

C. RISC-V CPU

The purpose of the coprocessor in AIA is to execute the programs aside from AI model acceleration, such as signal preprocessing, data moving, or number sorting. The present research adopts the RV32I basic instruction set in the open-

TABLE 4. The subcontrollers in the DLA.

Name	Operating Function	Input Limitation (8 bits)	Reconfigurable Parameter
Convolution Controller	Operation flow control of the convolutional layer	Length: 1–1024 Channel: 1–32	Kernel length: 1–6 Kernel number: 1–32 Stride length: 1–4 Activation: None, ReLU
Max-pooling Controller	Operation flow control of the max-pooling layer	Length: 1–1024 Channel: 1–32	Stride length: 2, 4
Transformation Controller	Scaling or shifting the input data	Length: 1–1024 Channel: 1–32	Mode: Scaling, shifting

TABLE 5. Detection results in software

		Detected Class (Software)	
		Negative	Positive
Labeled Class	Negative	354 (TN)	2.8 (FP)
	Positive	1.8 (FN)	475.4 (TP)

TABLE 6. Detection results in hardware

		Detected Class (Hardware)	
		Negative	Positive
Labeled Class	Negative	352.8 (TN)	4 (FP)
	Positive	3.8 (FN)	473.4 (TP)

source RISC-V instruction set architecture to implement the coprocessor [18]. The RISC-V community has abundant resources. Developers do not need to redevelop software, such as compilers, greatly reducing the technical difficulty required for developing a coprocessor. RISC-V coprocessors are also well suited for small size, high speed, and low power consumption scenarios. Thus, they fit the requirements of wearable devices.

Figure 13 is the architecture diagram of the RISC-V controller and RISC-V CPU. The role of the RISC-V controller is to bridge the CPU and AIA hardware architecture. If the layer type implements the RISC-V function, the instruction should be executed by the RISC-V CPU. The dispatcher instructs the RISC-V controller to begin the program execution by waking up the two-stage pipeline CPU. The CPU uses the memory-mapped input and output (MIMO) method to communicate with the RISC-V controller. When the CPU execution is finished, it initiates the "cpu_done" signal in the RISC-V controller by writing to a specific memory address to inform the dispatcher that the RISC-V function has been done. The RISC-V controller can also handle the DMA request from the CPU. After issuing a request, the CPU enters a wait state. When the RISC-V controller knows that the transfer of DMA memory data is completed, it will wake the CPU to execute the program.

IV. EXPERIMENT RESULT

A. EXPERIMENT RESULT OF ALGORITHM ON ANIMAL TESTING EEG DATABASE

The correctness of the proposed epilepsy algorithm can be validated with the animal experiment database mentioned in Section II. Tables 5 and 6 show the confusion matrices representing the outcomes of the seizure detection algorithm on software and hardware, respectively. The positive result indicates that the input EEG is a seizure segment, whereas the negative result indicates that the segment is normal. The labeled class and the detection result in the confusion matrix have four possible relationships: true positive (TP), false negative (FN), false positive (FP), and true negative (TN). Accuracy, sensitivity, and specificity are also important metrics for evaluating the detection algorithm [19], which can be expressed as

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \times 100\%, \tag{6}$$

$$\text{Sensitivity} = \frac{TP}{TP + FN} \times 100\%, \tag{7}$$

$$\text{Specificity} = \frac{TN}{FP + TN} \times 100\%. \tag{8}$$

In each fivefold cross-validation, the number of test data is 834. The confusion matrix value is the average of five outcomes because of the test method. Hence, the result has decimal numbers. The accuracy of 99.45%, the sensitivity of 99.62%, and the specificity of 99.22% can be obtained on the software. Given the use of fixed-point arithmetic in the hardware architecture, the calculation causes some errors with the original model. These errors result in a slight decrease in prediction accuracy. The seizure detection algorithm implemented by the hardware has an accuracy of 99.06%, a sensitivity of 99.20%, and a specificity of 98.88%.

B. MEASUREMENT RESULTS OF THE PROPOSED SYSTEM

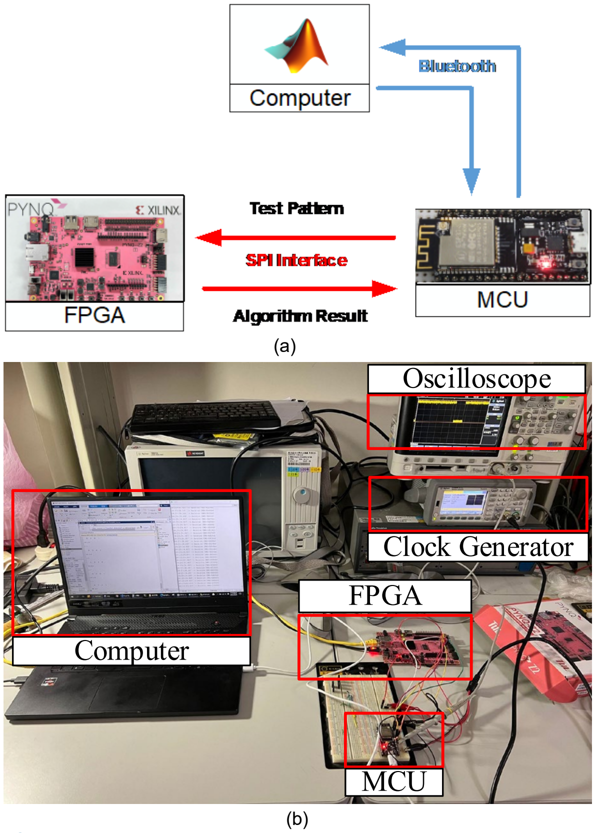


FIGURE 14. (a) AIA architecture test setup. (b) Measurement environment.

Figure 14 shows the AIA test setup and the measurement environment. The configuration file and test pattern are transmitted to the microcontroller unit (MCU) over Bluetooth by the MATLAB software on the computer. The MCU is the SPI master that communicates with the Field Programmable Gate Array (FPGA) board. The input pattern transmission takes 4.44 ms with a 2 MHz SPI interface. When the algorithm execution ends, the MCU can obtain the identification result through SPI. The result is sent back to the computer to analyze the algorithm's accuracy. The clock generator provides a square wave signal with a frequency of 1 MHz as the AIA's clock signal. The change in the "result_ready" signal can be observed on the oscilloscope. The signal switches from 0 to 1 after an algorithm computation. It remains at 1 until the AIA starts the next algorithm program. According to the observation of signals, the functional correctness of the AIA can be confirmed. The time required for the hardware architecture to complete the calculation of the seizure identification algorithm is close to 128 ms, which is acceptable for real-time seizure detection. The computation of signal preprocessing with CPU takes approximately 126 ms. The process of the CNN model can be completed in only 1.48 ms under the acceleration of DLA. The details of the layer inference time are shown in Table 7.

The AIA hardware architecture is implemented and validated on the PYNQ-Z2 FPGA board. The implementation consumes 3535 lookup tables (LUTs), 2283 flip-flops, 28 KB BRAM, and only six digital signal processors. The total power consumption is 0.108 W with a static power of 0.107 W and a dynamic power of 0.001W. This finding reveals that the power consumption of the FPGA board accounts for most of the power consumption. In comparison, the dynamic power consumption of the circuit is insignificant. The ASIC performance of the AIA hardware architecture is also tested under the simulation of the TSMC 180 nm 1P6M process. The performance is summarized in Table 8. The total power of the AIA is 1.29 mW. The core circuit of the CPU and DLA consumes 80 μW and 84.5 μW, respectively. Most of the power consumption of the hardware is consumed by the 28 KB memory with 1.13 mW. Given that this hardware architecture is reconfigurable, the large memory space implemented occupies the unknown storage capacity of the algorithm. Therefore, in the case of confirming the algorithm architecture, it may be considered to save power consumption further by reducing the memory size.

TABLE 7. Details of layer inference time.

Layer Instruction	Operation type	Time (μs)
RSIC-V (Initial CPU)	Pre-processing	112
RSIC-V (Preprocess)	Pre-processing	126230
Convolution_1	AI model	522
Maxpooling_1	AI model	139
Transformation_1	AI model	141
Convolution_2	AI model	112
Maxpooling_2	AI model	39
Transformation_2	AI model	33
Convolution_3	AI model	274
Convolution_4	AI model	104
RSIC-V (Classify)	AI model	112
Layer END	AI model	4

TABLE 8. ASIC performance summary

Chip specifications	
Technology	TSMC 180 nm 1P6M
Operating Frequency	1 MHz
Chip Area (μm^2)	4,851,385
AIA(w/o CPU)	
Logic Gates	126.8 k (NAND2 equiv.)
SRAM	8 KB Main Memory 4 KB DLA Memory
Power Consumption	84.5 μW
Detection Latency	1.48 ms
Energy Efficiency	109 nJ/class. (CNN Classification)
RISC-V CPU	
Logic Gates	155.8 k (NAND2 equiv.)
SRAM	8 KB IM 8 KB DM
Power Consumption	80 μW

V. CONCLUSION

Tables 9 and 10 list the recent works related to this research. Table 6 shows the seizure identification system implemented

TABLE 9. Comparison of seizure detection application in FPGA implementation

	ACCESS ’18 [20]	ASICON ’21 [21]	ASSCC’ ’22 [22]	**This Work**
Platform	ZYNQ-7000 XC7Z020	ZYNQ-ZC706	Alveo U250	**PYNQ-Z2**
Freq. (Hz)	100M	200M	75M	**1M**
Seizure Database	CHB-MIT	CHB-MIT	CHB-MIT	**Lab Rat**
Processor	ZYNQ7	No	RISC-V	**RISC-V**
Reconfigurable DLA	No	No	Yes	**Yes**
Detection Algorithm	Extractor: STFT Classifier: RBF-SVM	1D-CNN	SRNN	**1D-CNN**
Data Format	32-bit FP	16-bit FXP	--	**8-bit FXP**
LUT	11390	1480	21560	**3411**
Power	380 mW	N/A	787 mW	**0.108 mW**
Detection Accuracy	Sen: 98.4%	Acc: 97.35% Sen: 94.32%	91.09%	**99.06%, *90.70%**
Detection Latency (ms/class)	0.313 (STFT+RBF-SVM)	0.170	--	**Preprocessing: 126 AI model: 1.48**

*Using CHB-MIT, subject ID=chb08, input channel is modified to 4.

TABLE 10. Comparison of seizure detection application in ASIC implementation

	JSSC ’20 [23]	TBCAS ’21 [8]	TCAS II ’22 [24]	**This Work**
Technology (nm)	40	180	65	**180**
Supply Voltage (V)	0.58	1.8	0.9	**1.8**
Frequency (Hz)	130 k 65 k	1 M	10k	**1 M**
Seizure Database	CHB-MIT	Lab Rat	CHB-MIT	**Lab Rat**
Reconfigurable Processor	No	RISC-V	No	**AIA RISC-V**
Feature Extraction	FFT	CNN	FFT	**CNN**
Classifier	NL-SVM	Fully Connected	RBF-SVM	**Fully Connected**
Logic Gates	3.76M	130.5 k	--	**126.8 k[a]**
Data Format	24-bit FP	32-bit FXP	16-bit FXP	**8-bit FXP**
Power Consumption	1.9 mW	102 μW	200 μW	**84. 5 μW[a]**
SRAM	9 KB	6 KB	--	**12 KB[a]**
Detection Accuracy	96.6%	93.5%	Sen: 94.4%	**99.06%, *90.70%**
Detection Latency (ms/class)	710	12	280	**1.48[b]**

a: AIA w/o CPU
b: CNN acceleration
*Using CHB-MIT, subject ID=chb08, input channel is modified to 4.

using the FPGA platform. [20] and [21] used a database of CHB-MIT, which is measured in human scalp EEG. In comparison, the animal experiment database used in the present research is intracranial EEG. Although the objects and measurement methods are different, the clean and continuous measurement of intracranial EEG signals simplifies the use of the long-term seizure detection device. Moreover, this work also provides the detection accuracy using the CHB-MIT Database for reference. Compared with [20] and [21], our proposed method has the advantage of reconfigurability in terms of hardware architecture and is more lightweight than [22]. The CNN models of different sizes can be realized through the different settings of the configuration file. Thus, the execution of the hardware architecture becomes increasingly diverse and flexible. The proposed hardware architecture can also achieve high identification accuracy with low power consumption. The detection latency of the proposed hardware architecture is higher than that of the other two while meeting the requirement of real-time computing.

Table 10 presents a comparison table of accelerator chips applied to seizure identification. The hardware implemented by the chip can build a seizure identification system with low power consumption and volume. This system is suitable for the requirements of wearable devices. Compared with [23], our proposed DLA can realize many AI model acceleration processes by adjusting and setting the layer instruction. The number of logic gates used in the present study is smaller than that used in our previous work [8] to achieve lower power consumption. The amount of memory used is high to reserve space that may be required for reconfiguring. In terms of accuracy, the database employed in this study is the same as that in the previous work [8], but it can finish the calculation of the AI model in a short time, with slight power consumption and improved accuracy.

A highly accurate algorithm is proposed for seizure identification to summarize the contribution of this research. The algorithm has a two-step process, including the signal preprocessing algorithm and an optimized six-layer CNN model. An AIA hardware architecture is also proposed to implement an algorithm for real-time detection. The AIA has two computation units: DLA and RISC-V CPU. The DLA can accelerate the computation according to the AI model to meet the requirement of real-time detection. The RISC-V CPU can execute general-purpose programs aside from the AI model acceleration, such as the signal preprocessing step. The algorithm can achieve ~99% accuracy in both software and hardware with a detection latency of 128 ms/class. The process of the CNN model can be completed in only 1.48 ms under the acceleration of DLA with an energy efficiency of 109 nJ/class. This finding reveals the achievement according to the demonstration of hardware implementation. Moreover, the AIA is programable using layer instruction to manage the algorithm computation flow. Some layer types are also reserved for future scalable usage. According to the reconfigurable advantage, the present work can accelerate various application algorithms. It also improves the future utilization of the architecture.

ACKNOWLEDGMENT

The authors greatly appreciate the support of the Taiwan Semiconductor Research Institute, the National Science and Technology Council (NSTC), Taiwan (Grant NSTC 112-2221-E-006-181 MY3, NSTC 113-2640-E-006-006, and NSTC 113-2640-B-006-004.).

REFERENCES

[1] World Health Organization, "Epilepsy", Feb 9, 2022. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/epilepsy. (Accessed: May 5, 2024).

[2] A. H. Ropper, “Adams and victor’s principles of neurology, 8th ed.” New York, NY, USA: McGraw-Hill, 2005.

[3] K. E. Misulis., Atlas of EEG & seizure semiology, and management, Oxford University Press, Feb. 2014.

[4] S. -Y. Lee et al., “A programmable EEG monitoring SoC with optical and electrical stimulation for epilepsy control,” *IEEE Access*, vol. 8, pp. 92196-92211, 2020.

[5] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-2324, Nov. 1998.

[6] G. E. Hinton, S. Osindero and Y. Teh, “A fast learning algorithm for deep belief nets,” *Neural Comput.,* vol. 18, no. 7, pp. 1527-1554, 2006.

[7] B. Rim, N.-J. Sung, S. Min and M. Hong, “Deep learning in physiological signal data: A survey,” *Sensors*, vol. 20, no. 4, pp. 969, Feb. 2020.

[8] S. -Y. Lee, Y. -W. Hung, Y. -T. Chang, C. -C. Lin and G. -S. Shieh, “RISC-V CNN coprocessor for real-time epilepsy detection in wearable application,” *IEEE Trans. Biomed. Circuits Syst.*, vol. 15, no. 4, pp. 679-691, Aug. 2021.

[9] D. Shin and H. -J. Yoo, “The heterogeneous deep neural network processor with a non-von neumann architecture,” *Proceedings of the IEEE*, vol. 108, no. 8, pp. 1245-1260, Aug. 2020.

[10] S. -Y Pan, S. -Y. Lee, Y. -W Hung, C. -C. Lin, and G. -S. Shieh, “A Programable CNN Accelerator with RISC-V core in Real-Time Wearable Application,” in *Proc. IEEE Int. Conf. Recent Advances in Systems Science and Engineering (RASSE 2022)*, Nov. 2022, pp. 1-4.

[11] R. Kohavi, “A study of cross-validation and bootstrap for accuracy estimation and model selection,” in *Proc. 14th Int. Joint Conf. Artificial Intelligence*, 1995, pp. 338-345.

[12] Shoeb and J. Guttag, “Application of machine learning to epileptic seizure detection,” in *Proc. 27th Int. Conf. Mach. Learn.,* 2010, pp. 975-982.

[13] Y. LeCun, Y. Bengio and G. Hinton, "Deep learning", *Nature*, vol. 521, pp. 436-444, May 2015.

[14] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” *arXiv preprint* arXiv:1502.03167, 2015.

[15] Sahani, M.; Rout, S.K.; Dash, P.K., “FPGA implementation of epileptic seizure detection using semisupervised reduced deep convolutional neural network,” *Appl. Soft Comput*. vol. 110, pp. 107639, Oct. 2021.

[16] L. G. Rocha et al., “Binary CorNET: Accelerator for HR estimation from wrist-PPG,” *IEEE Trans. Biomed. Circuits Syst.*, vol. 14, no. 4, pp. 715-726, Aug. 2020.

[17] Y. -H. Chen, T. Krishna, J. S. Emer and V. Sze, “Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks,” *IEEE J. Solid-State Circuits*, vol. 52, no. 1, pp. 127-138, Jan. 2017.

[18] Andrew Waterman et al., “The RISC-V instruction set manual volume I: Unprivileged ISA,” *2CS Division*, EECS Department, University of California, Berkeley, August 2022.

[19] M. A. Bin Altaf et al., “A 16-Channel patient-specific seizure onset and termination detection SoC with impedance-adaptive transcranial electrical stimulator,” *IEEE J. Solid-State Circuits*, vol. 50, no. 11, pp. 2728-2740, Nov. 2015.

[20] H. Wang, W. Shi and C. Choy, “Hardware design of real time epileptic seizure detection based on STFT and SVM,” *IEEE Access*, vol. 6, pp. 67277-67290, 2018.

[21] L. Zhu, D. Liu, X. Li, J. Lu, L. Wei and X. Cheng, “An efficient hardware architecture for epileptic seizure detection using EEG signals based on 1D-CNN,” *IEEE 14th Int. Conf. on ASIC (ASICON)*, 2021, pp. 1-4.

[22] C. Fang, F. Tian, C. Wang, J. Yang and M. Sawan, "A 217.8 MSOPs/W FPGA-based Online Learning SNN Processor Using Unified Event-Driven Structure and Topology Aware Data Reuse Strategies," *2022 IEEE Asian Solid-State Circuits Conference (A-SSCC)*, Taipei, Taiwan, 2022, pp. 1-3.

[23] S. -A. Huang, K. -C. Chang, H. -H. Liou and C. -H. Yang, "A 1.9-mW SVM processor with on-chip active learning for epileptic seizure control," *IEEE J. Solid-State Circuits*, vol. 55, no. 2, pp. 452-464, Feb. 2020.

[24] Y. Su, W. Shi, L. Hu and S. Zhuang, "Implementation of SVM-Based Low Power EEG Signal Classification Chip," in IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 69, no. 10, pp. 4048-4052, Oct. 2022.