IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received August 6, 2018, accepted September 6, 2018, date of publication September 17, 2018, date of current version October 12, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2870273

EERA-ASR: An Energy-Efficient Reconfigurable Architecture for Automatic Speech Recognition With Hybrid DNN and Approximate Computing

BO LIU, HAI QIN, YU GONG, WEI GE, MENGWEN XIA, AND LONGXING SHI, (Senior Member, IEEE)

National ASIC System Engineering Technology Research Center, Southeast University, Nanjing 210096, China

Corresponding author: Bo Liu (liubo_cnasic@seu.edu.cn)

This work was supported in part by the National Science and Technology Major Project under Grant 2018ZX01031101-005 and in part by the National Natural Science Foundation of China under Grant 61404028 and Grant 61574033).

ABSTRACT This paper proposes a hybrid deep neural network (DNN) for automatic speech recognition and an energy-efficient reconfigurable architecture with approximate computing for accelerating the DNN. To accelerate the hybrid DNN and reduce the energy consumption, we propose a digital–analog mixed reconfigurable architecture with approximate computing units, including a binary weight network accelerator with analog multi-chain delay-addition units for bit-wise approximate computing and a recurrent neural network accelerator with approximate multiplication units for different calculation accuracy requirements. Implemented under TSMC 28nm HPC+ process technology, the proposed architecture can achieve the energy efficiency of 163.8TOPS/W for 20 keywords recognition and 3.3TOPS/W for common speech recognition.

INDEX TERMS Hybrid deep neural network, binary weight network, reconfigurable architecture, approximate computing.

I. INTRODUCTION

Deep Neural Networks (DNNs) that have many hidden layers have been proven to outperform traditional models (i.e., Markov models, Gaussian mixture models) on a variety of speech recognition benchmarks by a large margin [1], [2]. High performance and extreme energy efficiency are very critical for deployments of DNNs, especially in speech recognition systems for the internet of things. As the large amounts of neurons and synapses in DNNs incur intensive computation and power consumption, efficiently processing DNNs with limited resources and low energy cost remains a challenging problem.

To overcome the challenges, many accelerators are designed, which can be mainly divided into two categories: digital architectures and analog architectures. For digital architectures, four categories are widely used in DNN accelerators: FPGAs, customized DSPs with specified instruction sets, ASICs, and coarse-grained reconfigurable architectures (CGRAs). FPGAs have been frequently utilized as DNN accelerators to achieve speed-up over software simulations [3], [4]. For example, Chang's team mapped a Recurrent Neural Network (RNN) to FPGA platform and achieved up to 23× better performance per power than GPU Tegra-X1 [4]. However, FPGA is not suitable for small, low-power application because of its large footprint and power consumption. Chen, *et al*. published a serial work of DianNao [5], DaDianNao [6] and Cambricon-X [7] which are specified DSPs for DNNs. Relative to the FPGAs and specified DSPs, ASIC implementations are always highly optimized for chosen DNN models and topologies. M. Price, *et al*. presented an ultra-low power speech recognizer which reduced the power to 7.78mW @40MHz with WER of 8.78% under TSMC 65nm low-power logic process [8]. In the past decade, CGRAs have been introduced as an alternative way to the conventional designs with high flexibility and energy efficiency for DNNs. Yin, *et al*. proposed a CGRA called Thinker with two 16 × 16 reconfigurable processing elements arrays, which can support variant bit-width computing of DNNs [9]. The processor can achieve at most 1.27TOPS/W in energy efficiency and outperform the state-of-the-art architectures up to 5.2×. In work [10], an energy-efficient DNN accelerator is proposed with

2169-3536 © 2018 IEEE. Translations and content mining are permitted for academic research only.
Personal use is also permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

a network compression approach. With this compression approach, most of the computation can be eliminated before they are mapped to the DNN accelerator. Analog architectures have also been commonly adopted in DNN accelerators to implement the processing components of neurons and synapses because of the natural similarity to biological systems. However, the unreliability in the analog system is still a difficult problem to be solved. Besides, they also lack system flexibility and adaptability. For example, Badami, *et al*. proposed an analog DNN architecture for speech/non-speech classification in a voice activity detection system [11]. Comparisons show that the use of analog analytics brings increased energy efficiency by 10×. However, due to the limitation of analog circuit characteristics, this architecture can only work well under specific conditions, such as a particular SNR/database. Since the Binary Weight Network (BWN) has demonstrated its high energy efficiency and outperformed classification capability which is close to full precision networks in many databases, a BWN accelerator called YodaNN is proposed for ultra-low power BWN acceleration and can achieve 61.2TOPS/W in energy efficiency at 0.6V [12].

In this paper, we propose an energy-efficient reconfigurable hybrid DNN architecture and its circuit implementation for speech recognition. A hybrid DNN consisting of two network models is proposed for speech recognition, including a BWN for twenty frequently used keywords recognition, and a recurrent neural network based on the long-short term memory (LSTM-RNN) structure for processing acoustic model in common speech recognition. To accelerate the hybrid DNN and make it more energy efficient, we propose a digital-analog mixed reconfigurable architecture consisting of two accelerators: the BWN accelerator and the RNN accelerator. For the BWN accelerator, an analog multi-chain delay-addition unit is proposed for bit-wise approximate computing. For the RNN accelerator, a reconfigurable approximate multiplication unit is proposed for different calculation accuracy requirements.

This paper makes the following contributions:

1) We present a hybrid DNN model for speech recognition including a BWN and an LSTM-RNN. The BWN first process the input voice data for recognizing twenty words that frequently used (called keywords), and then the LSTM-RNN structure is used to process the acoustic model for high precision common speech recognition. Since BWN can be calculated with only bit-wise operations (i.e., XOR and addition operations), it can significantly reduce the memory access and computing energy.

2) We propose a BWN accelerator architecture and its circuit implementation for twenty frequently used keywords recognition. To achieve high energy efficiency, an analog multi-chain delay-addition unit is proposed for bit-wise approximate computing. Besides, the scheduling method for mapping different layers of proposed BWN onto the reconfigurable architecture is also present.

3) We propose an RNN architecture with approximate multiplication units to process different layers of the LSTM-RNN network. The approximate multiplication units can be dynamically reconfigured and adaptive to different accuracy requirements. Simulation results show that it can efficiently reduce power consumption with negligible loss of recognition accuracy.

II. PRELIMINARIES

A. HYBRID BIT-WIDTH WEIGHT COMPRESSION AND DYNAMIC DATA-WIDTH ADAPTIVE COMPUTING

On the one hand, DNNs always consist of a lot of synapses in which the values are close to zero, and therefore the network can be firstly compressed offline with pruning, hybrid bit-width weights and so on. With this weight compression approach, we can reduce the memory size, transmission time and access power of weights. In our previous works [13], [14], we have proposed an efficient network compression method with hybrid bit-width weights scheme. The key points of this hybrid bit-width weights compression approach are as follows: the weight magnitude of neural network reflects the significance of synapses. In terms of the distribution, the closer to the zero point, the more densely the weights are distributed. Therefore, we can divide the 16-bit weights into several levels with a weight grading scheme. And then, we can use different bit widths to represent the weights of different grading level. As shown in Table 1, we use five grading levels for compressing the weights. In Table 1, the leading ''1'' index indicates the position of first '1' in a weight (except the sign bit).

TABLE 1. Hybrid bit-width weights grading table.

Grading Level	Leading "1"	Length of Bits
0	<2	0
1	2-6	2
2	7-12	4
3	13-14	6
4	15	8

The compression method with hybrid bit-width weights scheme discussed above is also adopted in this work to compress the proposed LSTM-RNN network. In our work, we use 2/4/6/8 bits to represent each weights. In addition, the level 0 shown in Table 1 means these weights with a grading level 0 will be eliminated and not be transmitted to the hardware for processing. For convenience of hardware designs, the compressed weights will be decoded to 16-bit before the calculation by the RNN accelerator. As shown in the previous work [14], the compression ratio of RNN is 7×, which means 85.7% of the operations performed with uncompressed RNN will be eliminated in our work.

On the other hand, for speech recognition, the frequency of speech sampling is much higher than that of changing phonemes in speeches. Therefore, each phoneme sustains for several frames, which leads to the continuity of frames.

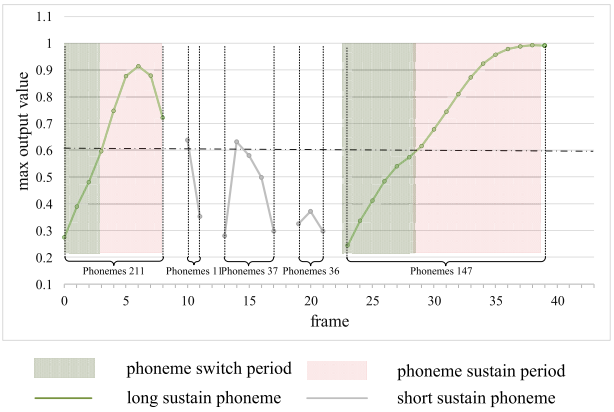


FIGURE 1. Trend of RNN maximum output values in speech recognition.

As discussed in our previous work [15], the trend of RNN network maximum output values for speech recognition is evaluated and shown in Fig. 1. It can be seen that the calculation accuracy required at time step $t+1$ can be estimated based on the outputs quality at time step t, so different calculation accuracy can be predicted accordingly. In this work, we set three thresholds to estimate and choose the calculation accuracy, which can be classified into 4 levels. Therefore, the approximate multiplier adopted in our work is designed for processing multiplication operations with four kinds of bit widths, including 4-bit $\times$ 16-bit, 8-bit $\times$ 16-bit, 12-bit $\times$ 16-bit, and 16-bit $\times$ 16-bit operations. In our work, the compressed weights with different bit widths will be firstly decoded into uncompressed weights with a 16-bit fixed-width, before they are transmitted to the computing units. For the bit width of input data for each layer of the RNN, there are four candidates: 4-bit, 8-bit, 12-bit and 16-bit, each for a different accuracy level. Therefore, the basic approximate multiplication unit in our work is designed as a 4-bit $\times$ 16-bit multiplication unit. The dynamic data bit-width adaptive computing is implemented by precision adjustment, which is distinguished by the thresholds.

B. APPROXIMATE COMPUTING MULTIPLIER

In LSTM-RNN, which is used for high precision common speech recognition in our work, the numbers of multiplication and addition operations are almost the same, but the power consumption of multiplication operation accounts for up to 96% of the total power consumption [16]. Thus, a convincing idea to reduce power consumption for processing LSTM-RNN is to improve the energy efficiency of multiplication operations. LSTM-RNN have been proven to be naturally fault-tolerant, and the calculation accuracy requirements for various application scenarios are also in large variations [17]. Therefore, we can use approximate multipliers with reduced power consumption to replace the traditional multipliers adopted in LSTM-RNN.

In our work, we proposed a reconfigurable approximate multiplication unit for approximate calculation of the LSTM-RNN based on Liu's work [18]. For a multiplication operation, the product of A and B can be calculated by the sum of each partial products of A_i and B_j (A_i and B_j are each bits of A and B respectively, and i, j $= 1, 2, 3, \cdots, 16$). Therefore, we can design a power efficient approximate multiplier by replacing the adders adopted with approximate adders.

This approximate adder firstly simplifies the carry calculation by a pre-processing operation. For a certain bit i, if A_i is 0 and B_i is 1, then we first exchange the A_i and B_i to make sure the bits of value 1 in A is as much as possible. The logical expressions can represent this pre-processing operation as below:

$$A'_i = A_i + B_i \tag{1}$$
$$B'_i = A_i B_i \tag{2}$$

After the pre-processing operation, the carry calculation can be approximately simplified. The truth table of this approximate adder is shown in Table 2, where A and B are the two preprocessed addends, C represents a carry, S is the addition result, and E represents a calculation error with this approximate approach.

TABLE 2. Truth table of the approximate adder.

Input			Output	
$B'_i B'_{i-1}$	A'_i	C_{i-1}	S_i	E_i
00	A'_i	0	A'_i	0
01	A'_i	1	1	A'_i
10	1	0	0	0
11	1	1	1	0

From Table 2, the logical formulas of Si and Ei can be shown in (3) and (4), where preprocessed logic equations in (1) and (2) are brought into calculations.

$$S_i = (A_i \oplus B_i) + A_{i-1}B_{i-1} \tag{3}$$
$$E_i = (A_i \oplus B_i)A_{i-1}B_{i-1} \tag{4}$$

We can furtherly optimize the approximate adder logic as shown above. The XOR logic is calculated as $A \oplus B = \overline{A}B + A\overline{B}$. With an inversion operation, the XOR logic can be changed to an XNOR logic of $A \odot B = AB + \overline{A}\,\overline{B}$. The XNOR operation $A \odot B = AB + \overline{A}\,\overline{B} = \overline{(A+B)\cdot\overline{AB}}$, and the operation "$\odot$" can be split into two parts: one is the logic function $A + B$, the other is the logic function $\overline{AB}$, therefore the hardware resources can be mostly reused. The approximate adder circuit design will be further discussed in next section.

$$S_i = \overline{(A_i \oplus B_i)\overline{(A_{i-1}B_{i-1})}} \tag{5}$$
$$E_i = \overline{(A_i \oplus B_i) + \overline{A_{i-1}B_{i-1}}} \tag{6}$$

Implemented and simulated under TSMC 28nm HPC+ process at 0.8V 25°C TT corner, the optimized adder only cost 55.7% power of the standard adder. Besides, the circuit latency can be significantly reduced, because the carry chain which is the critical path of standard adder circuits, is removed in this approximate adder. For a multiplication

operation, the addition of each partial product should be finally accumulated, where the error of each approximate adder will also be accumulated. Therefore, we propose an error recovery method and its circuits design in the next section.

III. ENERGY-EFFICIENT RECONFIGURABLE HYBRID DNN ARCHITECTURE WITH APPROXIMATE COMPUTING

A. SPEECH RECOGNITION SYSTEM BASED ON HYBRID DEEP NEURAL NETWORK MODEL

The hybrid DNN model for speech recognition is shown in Fig. 2. This model consists of a keywords recognition unit using BWN and a high precision speech recognition unit using RNN. Besides, the BWN unit can also be used as a detection module for the on/off control of the high precision recognition RNN unit.

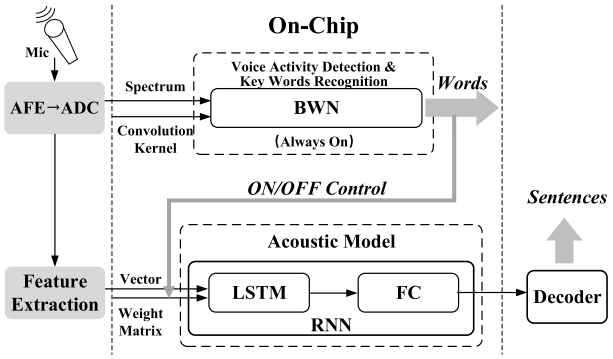


FIGURE 2. Speech recognition system based on hybrid deep neural network model.

We trained a BWN with six convolution layers and three fully-connected layers. It can detect and recognize twenty words at the accuracy of 99.6% in software simulation. The neural networks with binarized weights can drastically reduce the memory size and the data access, as well as make multiplication operations replaced by bit-wise operations, which can significantly reduce the power consumption and improve the energy efficiency. Once the BWN detects a continuous speech, the RNN will be activated. We stack four recurrent layers with 640 LSTM units as the building blocks of each recurrent layer, a fully-connected layer and a soft-max layer for our acoustic model for Chinese speech recognition. We also stack two recurrent layers with 256 LSTM units of each recurrent layer, a fully-connected layer and a soft-max layer for our acoustic model for English speech recognition, because the recognition complexity of English is much less than that of Chinese. This acoustic model gives the most likely label sequences (phonemes or characters) for speech frames input. Finally, the decoding of label sequences is done offline by software, combined with the dictionaries and language models, to evaluate the recognition accuracy.

As shown in Fig. 3, the top level architecture consists of a system controller implemented with ARM7TDMI, a 24KBytes scratch-pad memory (SPM), two heterogeneous

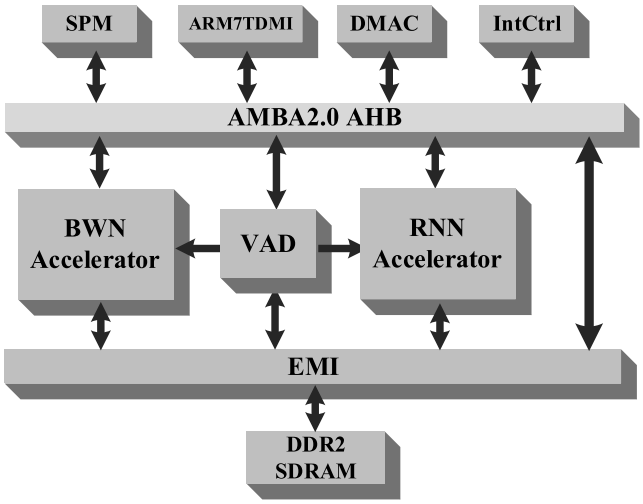


FIGURE 3. Top architecture of the porotype system.

accelerators for BWN and RNN respectively, a VAD module for detecting the incoming voice signal and determining whether it is noise or not, and several assistant modules for system scheduling. All of the modules are AMBA2.0-AHB-compatible and connected to the 32-bit AHB bus module, used as the system bus.

B. ENERGY-EFFICIENT RECONFIGURABLE BWN ACCELERATOR FOR TWENTY KEYWORDS RECOGNITION

As shown in Fig. 4, we trained a BWN for a twenty keywords recognition system with six convolution layers and three fully-connected layers. During the training process, the whole forward pass and the calculation of the gradient during the back propagation, the weight is binarized to either $+1$ or -1. The feature-mapped data is quantized to 16-bit. The size of the convolution kernel is 3×3, and the number of convolution cores per layer is 16, 16, 32, 32, 64 and 64. A 2×2 max pooling layer follows each two convolution layers.

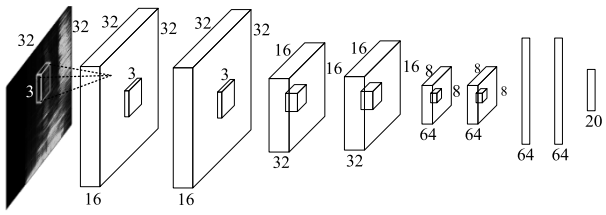


FIGURE 4. BWN topology for keywords recognition.

Fig. 5 shows the architecture of the BWN accelerate processor in our work. The processor mainly consists of a reconfigurable computing array (RCA) with 4 processing elements (PEs) and a Post-Processing Unit. Each PE can be configured to perform the calculation of different network layers, including six convolution layers and three fully-connected layers with different settings. The Post-Processing Unit is used to process the convolution results accumulation, activation, normalization, and pooling operations. The voice

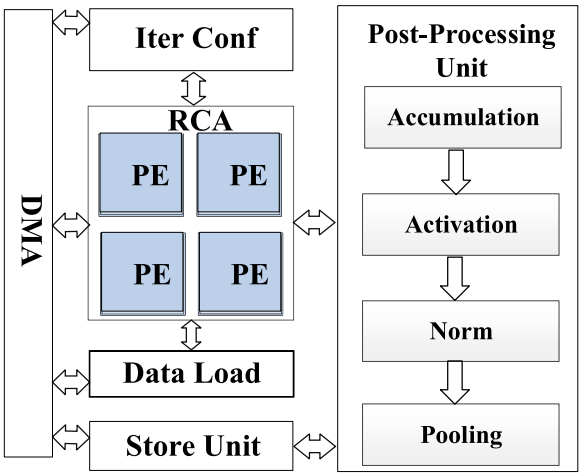


FIGURE 5. BWN accelerator architecture.

data is first input and pre-processed by the VAD module, and then it will be transmitted into the BWN accelerator if it is not a noise signal. The BWN accelerator will then be configured to process the six convolution layers and three fully-connected layers as shown in Fig. 4, to recognize the incoming voice signal and determine whether it is one of the twenty keywords.

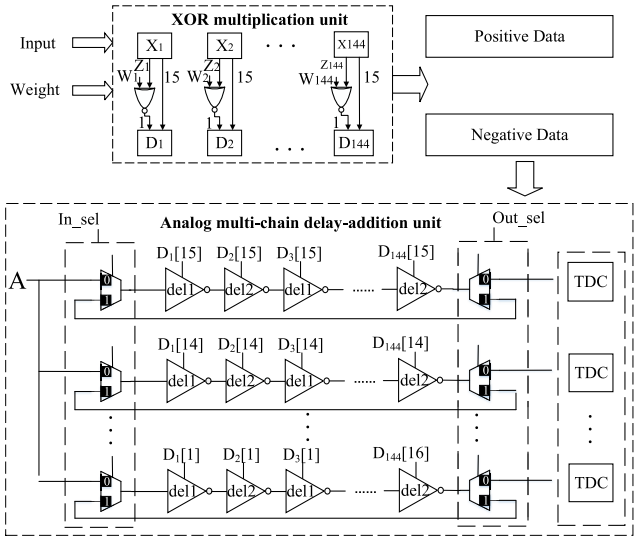


FIGURE 6. PE architecture of the BWN accelerator.

As shown in Fig. 6, the PE for processing BWN consists of the XOR multiplication units, an analog multi-chain delay-addition unit with an iteration controller and a Time-to-Digital Converter (TDC). The data of each layer in the topology based on the binarized network is a multiple of 144. Therefore the BWN circuit processes multiplication of n × 144 data (n = 1, 2, 3, ···) controlled by the iteration controller at one time. 144 or 3 × 144 data in one layer will be transmitted to the circuit unit at one time, then multiplied with the corresponding binarized weight through the XOR multiplication units. The resulting data is fed into the analog multi-chain delay-addition unit for bitwise accumulation operations, then the number of '1' is summed up and converted into the delay. The data is divided into positive and negative parts before being accumulated. Both parts of the data remain the same size as the original data (fill zero in excess). Finally, the sum of the 16 corresponding bits is sent to the Post-Processing Unit, which conducts the accumulation, activation, normalization and pooling.

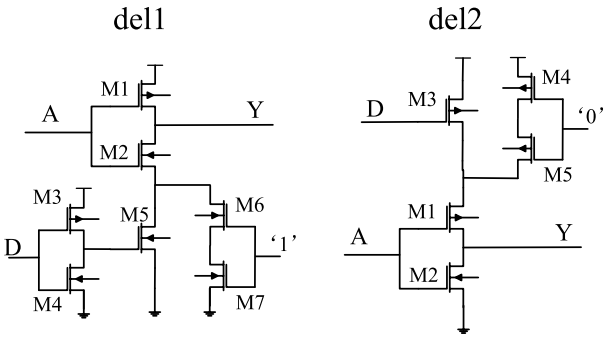


FIGURE 7. The two delay blocks for proposed BWN accelerator.

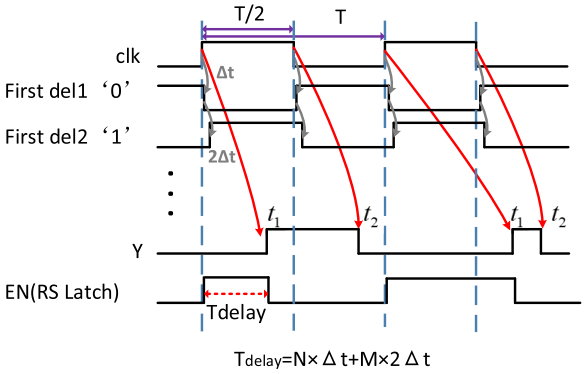


FIGURE 8. Example of a delay chain working process.

For the implementation of addition operations in the BWN, we design an efficient approximate addition tree based on the inverter delay. And the analog module consists of two kinds of delay blocks: the del1 and the del2 as shown in Fig. 7. The del1 is a controllable delay block triggered by rising the edge of the clock while the del2 is a controllable delay block triggered by the falling edge of the clock. For the del1, when its control signal D is '1', M5 is turned on, and connected with M6 and M7 to form a pull-down NMOS array. The pull-down resistance is small, and the delay is Δt. When the input signal of D is '1', M5 is turned off, and the pull-down network has only one path formed by M6 and M7. The pull-down resistor is large, and the delay is 2Δt. The del2 works in the same way as the del1. By changing PMOS with control D, different delays for falling edges can be obtained. Fig. 8 shows the operation example of a delay chain, in which N is the number of 0 and M is the number of 1, and then the delay can be calculated by N × Δt + M × 2Δt.

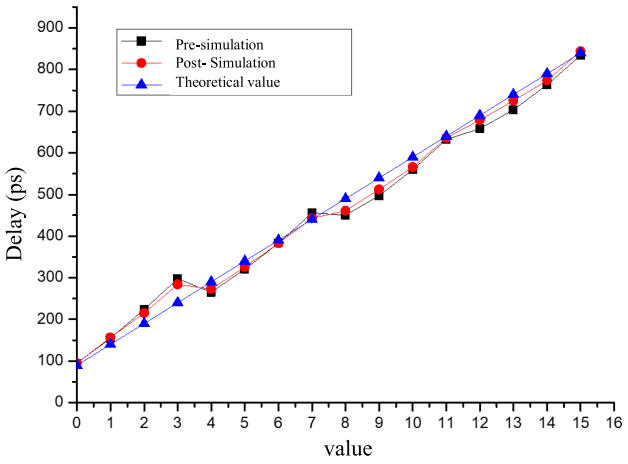


FIGURE 9. Impact of PVT on time-delay based addition.

The fluctuation of delay affected by the factors such as process angle and temperature of CMOS also brings some calculation errors. Simulation results show that under TSMC 28nm process, the error of the quantization results of a 144-level delay chain in SS and FF process corner can reach 10%, as shown in Fig. 9.

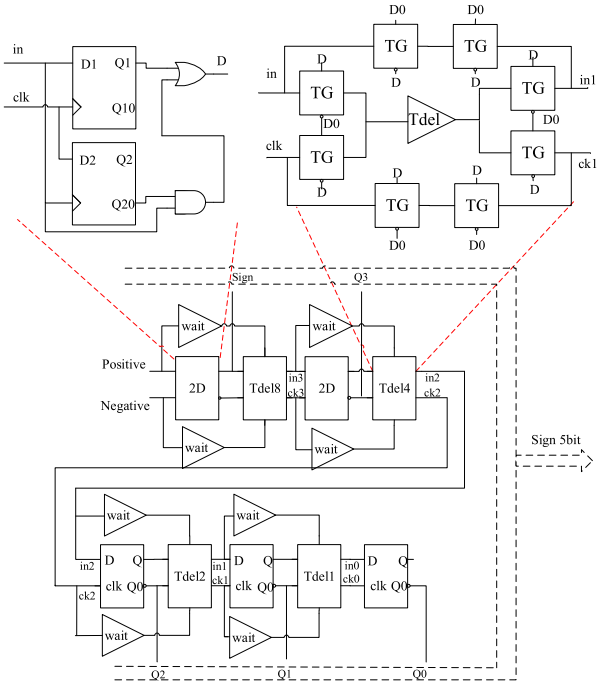


FIGURE 10. TDC architecture for proposed BWN accelerator.

In this work, a time-to-digital converter module (TDC) is implemented to process the delay signal in the BWN accelerator. The n-bit TDC based on a binary-search algorithm is composed only of n FFs and n delay blocks with different depth, which results in the area and power reduction. To further reduce the impact of process, PVT and other factors on the delay of the MOS transistor, the delay block of the TDC module is designed with the same delay chain as the calculation block (e.g., Tdel1, Tdel2, Tdel4 and Tdel8 shown in Fig. 10). The double-trigger structure named 2D in TDC is used to prevent the misjudgments. Besides, we designed the Tdel selection unit to choose the delay for the trigger's clock or data-in path. The wait module is used to compensate for the delay of the trigger.

Unlike traditional analog circuit designs, complex DAC/ADC modules are not required in the analog multi-chain delay-addition unit proposed in this work. The conversion of digital signals into the controllable delay signals is implemented by the two kinds of delay blocks, while the conversion of delay signals to digital signals is implemented by the proposed TDC module. Therefore, the power consumption and area overhead of these modules can be effectively reduced.

When the convolution operation is carried out, it can be divided into two cases: when the number of feature map's channel D<16, the PE array does not iterate. When D>16, the PE array iterate D/16 times. Also, fully-connected layer can also be transformed into one-dimensional multiplication-accumulation of the vector when calculated. The pseudo-code of the scheduling method is described as follows: X and Y are the dimensions of the input data feature, w_i denotes the i-th weight, ch represents the channel of feature maps, *Partialsum* represents the partial sum generated by the convolution operations, and CONV represents a $3\times3\times16$ convolution operation.

01: Supposed that: input feature map: f(X, Y)
02: PE convolution: $3\times3\times16$
03: PE Weights collection: $1, \cdots, w_n$
04: for all $Y = Y + 1$ do
05: for all $X = X + 1$ do
06: for all $w_i \in \{1, 2, \ldots, w_n\}$ do
07: for all $ch = ch + 16$ do
08: $PartialSum(i) = CONV(3\times3\times16)$;
09: end for
10: $PartialSum(i) += PartialSum(i)$;
11: end for
12: output $PartialSum(i)$;
13: end for
14: end for

C. ENERGY-EFFICIENT RECONFIGURABLE RNN ACCELERATOR FOR COMMON SPEECH RECOGNITION

We trained two LSTM-RNN for acoustic model of common speech recognition based on EESEN [19]. One LSTM-RNN is for Chinese speech recognition with the following settings: four recurrent layers and 640 LSTM units in each layer, one fully-connected layer and a soft-max layer. The other LSTM-RNN is for English speech recognition with the following settings: two recurrent layers and 256 LSTM units in each layer, one fully-connected layer and a soft-max layer. The LSTM-RNN used for Chinese speech recognition

is more complex than that for English, and contains more neurons and synapses. This is because there are 213 Chinses phonemes and only 48 English phonemes. For Chinese speech recognition, we use the THCHS-30 Speech Corpus [20], including 750 Chinese sentences as the training set and benchmarks to test the recognition accuracy and energy efficiency. For English speech recognition, we use the TIMIT Speech Corpus [21], including 6300 English sentences as the training set and benchmarks.

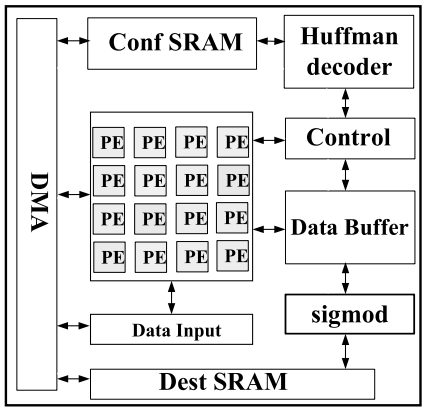


FIGURE 11. RNN accelerator architecture.

Fig. 11 shows the architecture of the RNN accelerator in our work. The accelerator consists of a controller, a sigmoid function module, 4×4 process elements (PEs), a configuration SRAM (Conf SRAM) and a data SRAM (Dest SRAM). The accelerator can directly access both the input data and the weights via DMA. Each PE consists of a proposed approximate multiplier, eight 16-bit data input/output registers and multiplexers, the weight SRAM with the size of 2KB and a data input FIFO. The PE can be reconfigured to process multiplication or addition operation with adder adopted in the multiplier architecture. Besides, the interconnections between PEs can also be reconfigured by setting the data input/output multiplexers of each PE, and therefore the RNN accelerator can process matrix multiplication and addition with various sizes of different layers of the two adopted LSTM-RNN networks for Chinese and English speech recognition respectively. Besides, the Huffman decoder is used to dynamically decodes the compressed weights in Conf SRAM and transmit them to the control module; And then the control module distributes the weights to the PEs.

In our work, an approximate multiplier with error recovery for the RNN accelerator is proposed as shown in Fig. 12. According to the discussion in Section. I, we take into account the effects of accumulated error in our approximate multiplier to improve the accuracy. Requirements of different calculation precision can be satisfied with the different number of MSBs used for error reduction. The approximate multiplier with error recovery can meet the LSTM-RNN computing accuracy requirements for speech recognition and contribute a significant decrease in energy cost compared to that with

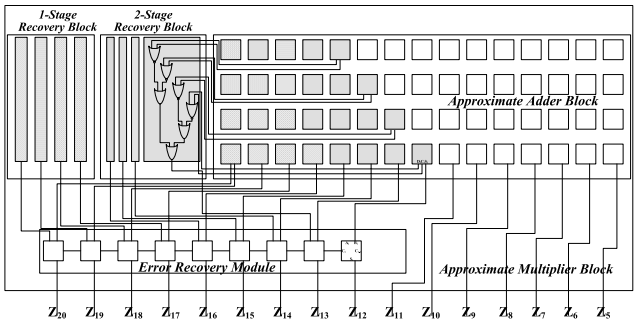


FIGURE 12. Approximate multiplier with error recovery.

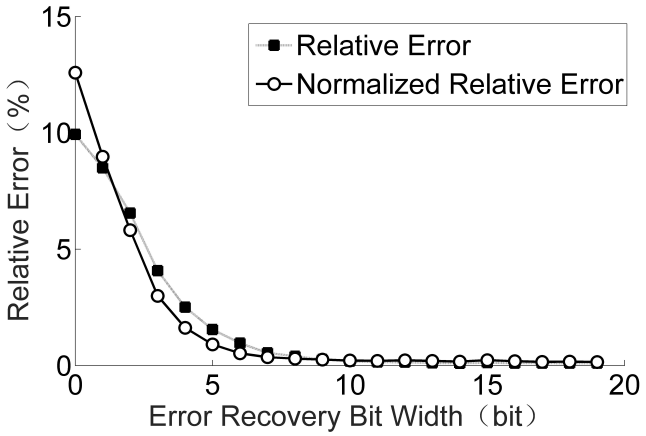


FIGURE 13. Relative error with different bit widths of error-recovery.

traditional multipliers, especially for low-precision calculation cases.

To study the characteristics of error recovery, we simulate the approximate multiplication algorithm discussed in Section.II via Python with 100,000 random numbers. The relative error of the error recovery under different calculation bit width is shown in Fig. 13. The dotted line in the figure indicates the relative error (E_r), which is the ratio of the absolute difference ($|N_{appr} - N_{std}|$) between the approximate multiplication calculation result (N_{appr}) and the standard multiplication calculation result (N_{std}): $E_r = |(N_{appr} - N_{std})|/N_{std}$. In the figure, the solid line represents the normalized relative error (E_{nr}), which is the ratio of the absolute difference to the average multiplication result ($N_{appr-avg}$): $E_{nr} = |(N_{appr} - N_{std})|/ N_{appr-avg}$. The $N_{appr-avg}$ is the average of 100,000 multiplication results of random numbers. For some smaller numbers, the MSBs are 0, the error cannot be reduced with MSBs error recovery, so the relative error E_r is high. The LSBs in the neural network have lower impacts on the network accuracy. Therefore, the normalized relative error E_{nr} is also taken into consideration in our work. As seen in Fig. 13, when the error recovery circuit has less bit width, the E_{nr} is smaller than the E_r. When the recovery bit width increases, the benefit of error recovery is more significant correspondingly.

Based on the analysis above, we choose 4-bit and 8-bit as the two-stage error recovery bit width in this work. The approximate multiplication unit with error recovery is shown in Fig. 12. The basic module is a 4×16 bit approximation multiplier (the input data is 4-bit, and the weight is 16-bit). There are two error calculation modules adopted in this approximate multiplication unit, called the 1-stage and the 2-stage error recovery block respectively. The approximate adders and the error recovery modules with the same remarks as shown in Fig. 12 are connected (only one group of these connections are shown in the figure). The 1-stage/2-stage error recovery blocks calculate the corresponding E_i as shown in (5) and (6). And then the error results are merged though an OR logic. Finally, the error recovery module will produce the approximate multiplication results by calculating the sum of each partial products and the error results.

The basic component of approximate multiplication with error recovery is an approximate adder. The logical formula of an approximate adder is shown in (5) and (6). In order to reduce power consumption, only the approximation adders for high-bit calculation need error recovery calculations. Therefore, the hardware design of the basic 1-bit approximate adder does not consist a calculation module for E_i. The circuit diagram of the 1-bit approximation adder is shown in Fig. 14. It has three inputs: A_i, B_i, and $\overline{C_{i-1}}$, where A_i and B_i represent the i-th-bit of the two input data A and B, and $\overline{C_{i-1}}$ is the negation value of the calculation carry of (i-1)-th-bit. It has three outputs: D_i, S_i, and $\overline{C_i}$, in which S_i is the result bit, C_i is the carry to high bit and D_i is used for the implementation of the error recovery.

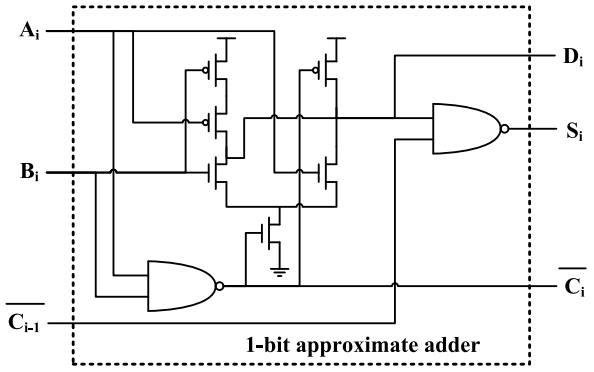


FIGURE 14. 1-bit approximation adder circuit diagram.

IV. IMPLEMENTATION RESULTS

The prototype system as shown in Fig. 3 with proposed hybrid DNNs and accelerator architectures is implemented and evaluated under TSMC 28nm HPC+ process technology. The BWN accelerator is customized with Cadence Virtuoso Tool using LVT transistors, and the other digital modules are described with Verilog HDL language and synthesized by Synopsys Design Compiler (DC). The BWN accelerator and VAD module are functional with logic supply voltage 0.58V (10MHz), the RNN accelerator and other modules is functional with 0.8V (400MHz). The on-chip memory are functional with 0.8V, and the bit-line voltage drop (V_{BL}) is 430mV (swing from 0.8V to 0.37V). The timing and energy cost are evaluated with Synopsys HSIM at 25°C TT corner. The area and power consumption are shown in Table 3.

TABLE 3. Implementation details of proposed system.

Module	Energy Power (mW)	consumption Percent (%)	Hardware Area (μm^2)	overhead Percent (%)
BWN accelerator and VAD	0.3	0.5%	315703	9.40%
RNN accelerator	21.7	40.4%	725068	21.60%
Memory	27.8	51.8%	1965476	58.55%
Others	3.9	7.3%	350640	10.45%
Total	53.7	100%	3356887	100%

A. RECOGNITION ACCURACY EVALUATION

For Chinese speech recognition, the trained LSTM-RNN with four recurrent layers and 640 LSTM units, can achieve the training accuracy of 82.93% and the valid accuracy of 81.83% (token accuracy) with software simulation on desktop computers. For English speech recognition, the training accuracy of our LSTM-RNN with two recurrent layers and 256 LSTM units can achieve the training accuracy of 82.89% and the valid accuracy of 81.78% (token accuracy) with software simulation. To compress the LSTM-RNN, reduce calculation operations mapped to RNN accelerator and the energy sonsumption, we first adopt the typical pruning method in work [10], and then use the hybrid bit-width scheme proposed in our previous work [14] to compress the networks further. Taking the LSTM-RNN for Chinese speech recognition as example, the network parameters and accuracy before and after compression are shown in Table 4. The total parameters size of the network decreased from 17 MB to 0.23 MB (the compression rate is 7×), with negligible accuracy loss (the Top-5 error of the network increased by only 0.1%). Since the network structures used for both two LSTM-RNN are similar, only the number of neurons composed is different. The compression result for the network used for English speech recognition is similar to that of the network for Chinese speech recognition.

TABLE 4. Parmeter size and accuracy after compressions.

Compression Scheme	Top-1 Error	Top-5 Error	Parameters	Compression Rate
Original	8%	0	17MB	-
Compressed with work [10]	8.1%	0	5.86MB	2.9
Compressed with work [10] and [14]	11.6%	0.1%	0.23MB	7.0

To verify the system accuracy with proposed approximate multiplication units, a reference design with standard multipliers is also implemented. As shown in Table 5, the comparison results show that the architecture with the proposed approximate multipliers can achieve negligible loss of recognition accuracy (<0.26%).

TABLE 5. Recognition accuracy with approximate multipliers.

Data Set	Speech Noise (SNR)	Token using standard multiplier	Accuracy using proposed approximate multiplier
THCHS-30 (Chinese)	clean	79.83%	79.78%
	20 dB	74.18%	73.92%
	10 dB	61.28%	61.02%
TIMIT (English)	clean	79.66%	79.64%
	20 dB	73.82%	73.57%
	10 dB	61.09%	60.84%

For the proposed BWN accelerator, to further reduce the error caused by analog computing, we add some errors to the input of each layer to analysis the error model of analog computing. The training approach by adding some errors to improve the robustness of the network to computational errors is called fault-tolerance training. In this work, the added errors are random values sampled from a normal distribution with a standard deviation of 0.8, which is determined to be large enough compared with the Monte-Carlo simulation results of our PE circuit. The Google's Speech Commands [22] is used to test our BWN for keyword recognition, and we choose 20 words of the dataset as our test benchmarks, as shown in Table 6. The accuracy comparison results are shown in Table 7.

TABLE 6. The 20 words chosen for key word recognition.

Key Words	Zero, One, Two, Three, Four, Five, Six, Seven, Eight,Nine, Yes, No, On, Off, Stop, Go, Up, Down, Left, Right.

TABLE 7. Accuracy comparison in different SNRS.

Speech Noise (SNR)	Recognition Accuracy		
	software	This work (without fault-tolerance training)	This work (with fault-tolerance training)
clean	95.40%	91.88%	93.51%
20 dB	94.63%	90.75%	92.10%
10 dB	94.55%	90.12%	91.69%

B. ENERGY EFFICIENCY COMPARISONS

The comparisons with other BWN accelerators are shown in Table 8. The results show that, evaluated on TSMC 28nm HPC+ process technology and 25 °C TT corner, our design can achieve 49.1GOPS at 10MHz, while the power is 300μW for twenty keywords recognition. Comparing to state-of-the-art BWN accelerator architecture YodaNN, the energy efficiency of our work is 163.8TOPS/W which is 2.6× better than YodaNN. The benefits of energy efficiency are mainly due to two reasons: firstly, the more advanced technology process is adopted for our work; secondly, compared with YodaNN, which is designed with digital computing architectures and standard circuits, our BWN accelerator is implemented with digital-analog mixed approximate computing.

TABLE 8. Comparison with other BWN accelerators.

Platforms	YodaNN[12]	Proposed BWN accelerator
Bit Width of Weight	1	1
Technology (nm)	65	28
Voltage (V)	0.6	0.6
Power Efficiency (TOPS/W)	61.2	163.8

TABLE 9. Comparison with other RNN accelerators.

Platforms	RNN-LM[3]	EIE[10]	Thinker[9]	This work
Architecture	FPGA	Reconfigurable Architecture		
Technology (nm)	45	65	65	28
Frequency (MHz)	150	800	200	400
Bit width of Data	16	16	8/16	4/8/12/16
Bit width of Weight	16	16	8/16	2/4/6/8
Computing circuits	Standard Multipliers			Approximate Multipliers
Power cost (W)	25	0.590	0.290	0.054
Peak Throughput (GOPS)	9.6	401.2∼ 1103*	368	179.2*
Power Efficiency (GOPS/W)	0.38	680∼ 1870	1268	3318

**The equivalent throughputs are multiplied by their compression rates (4x-11x for EIE and 7x for our work, respectively).*

The comparisons with other RNN accelerators are shown in Table 9. Generally speaking, the reconfigurable architectures can achieve much higher power efficiency than FPGAs for accelerating RNNs. In work [3] and [9], the RNNs are implemented and evaluated in their original form without compression. In work [10] and our work, all the networks are firstly compressed, so the equivalent throughput should be multiplied by the compression rates as discussed in [10]. Comparing with the state-of-the-art architectures, EIE, which is also implemented on compressed networks, our work achieves over 1.7× better in energy efficiency because the approximate computing units adopted in our work can significantly reduce computing energy cost. And comparing with state-of-the-art architecture Thinker, our work can achieve up to 2.6× better in power efficiency.

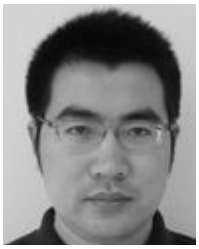
V. CONCLUSIONS

This paper proposes a hybrid Deep Neural Network (DNN) for speech recognition which includes a Binary Weight Network (BWN) for keywords recognition and a Recurrent Neural Network (RNN) based on LSTM structure for high precision common speech recognition. To accelerate the hybrid DNN and make it energy efficient, a digital-analog mixed reconfigurable architecture consists of two accelerators is proposed in our work: for the BWN accelerator, an analog multi-chain delay-addition unit is proposed for bit-wise approximate computing; and for the RNN accelerator, a reconfigurable approximate multiplication unit is proposed for different calculation accuracy requirements.

The energy efficiency of the proposed architecture can achieve 163.8TOPS/W for the twenty keywords recognition and 3.3TOPS/W for the common speech recognition. Comparing with state-of-the-art architectures, this work achieves over 1.7× better in energy efficiency.

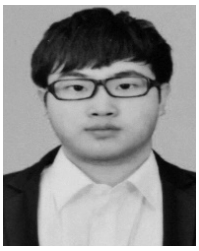
REFERENCES

[1] M. L. Seltzer, D. Yu, and Y. Wang, "An investigation of deep neural networks for noise robust speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Vancouver, BC, Canada, May 2013, pp. 7398–7402.
[2] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Trans. Audio, Speech, Language Process.*, vol. 20, no. 1, pp. 30–42, Jan. 2012, doi: 10.1109/TASL.2011.2134090.
[3] S. Li, C. Wu, H. Li, B. Li, Y. Wang, and Q. Qiu, "FPGA acceleration of recurrent neural network based language model," in *Proc. IEEE 23rd Annu. Int. Symp. Field-Program. Custom Comput. Mach.*, Vancouver, BC, Canada, May 2015, pp. 111–118.
[4] A. X. M. Chang and E. Culurciello, "Hardware accelerators for recurrent neural networks on FPGA," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Baltimore, MD, USA, May 2017, pp. 1–4.
[5] T. Chen *et al.*, "DianNao: A small-footprint high-throughput accelerator for ubiquitous machine-learning," *ACM SIGPLAN Notices*, vol. 49, no. 4, pp. 269–284, 2014, doi: 10.1145/2541940.2541967.
[6] Y. Chen *et al.*, "DaDianNao: A machine-learning supercomputer," in *Proc. 47th Annu. IEEE/ACM Int. Symp. Microarchitecture*, Cambridge, U.K., 2014, pp. 609–622.
[7] S. Zhang *et al.*, "Cambricon-X: An accelerator for sparse neural networks," in *Proc. 49th Annu. IEEE/ACM Int. Symp. Microarchitecture (MICRO)*, Taipei, Taiwan, Oct. 2016, pp. 1–12, doi: 10.1109/MICRO.2016.7783723.
[8] M. Price, J. Glass, and A. P. Chandrakasan, "A low-power speech recognizer and voice activity detector using deep neural networks," *IEEE J. Solid-State Circuits*, vol. 53, no. 1, pp. 66–75, Jan. 2018, doi: 10.1109/JSSC.2017.2752838.
[9] S. Yin *et al.*, "A high energy efficient reconfigurable hybrid neural network processor for deep learning applications," *IEEE J. Solid-State Circuits*, vol. 53, no. 4, pp. 968–982, Apr. 2018, doi: 10.1109/JSSC.2017.2778281.
[10] S. Han *et al.*, "EIE: Efficient inference engine on compressed deep neural network," in *Proc. Int. Symp. Comput. Archit.*, vol. 44, no. 3, 2016, pp. 243–254, doi: 10.1145/3007787.3001163.
[11] K. M. H. Badami, S. Lauwereins, W. Meert, and M. Verhelst, "A 90 nm CMOS, 6μW power-proportional acoustic sensing frontend for voice activity detection," *IEEE J. Solid-State Circuits*, vol. 51, no. 1, pp. 291–302, Jan. 2015, doi: 10.1109/JSSC.2015.2487276.
[12] R. Andri, L. Cavigelli, D. Rossi, and L. Benini, "YodaNN: An architecture for ultralow power binary-weight CNN acceleration," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 37, no. 1, pp. 48–60, Jan. 2018, doi: 10.1109/TCAD.2017.2682138.
[13] B. Liu *et al.*, "An energy-efficient accelerator for hybrid bit-width DNNs," in *Proc. IEEE Symp. Series Comput. Intell.*, Honolulu, HI, USA, Nov. 2017, pp. 1–8.
[14] Z. Wang *et al.*, "EERA-DNN: An energy-efficient reconfigurable architecture for DNNs with hybrid bit-width and logarithmic multiplier," *IEICE Electron. Express*, vol. 15, no. 8, pp. 1–10, 2018, doi: 10.1587/elex.15.20180212.
[15] B. Liu *et al.*, "E-ERA: An energy-efficient reconfigurable architecture for RNNs using dynamically adaptive approximate computing," *IEICE Electron. Express*, vol. 14, no. 15, pp. 1–11, 2017, doi: 10.1587/elex.14.20170637.
[16] M. Horowitz, "Computing's energy problem (and what we can do about it)," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, San Francisco, CA, USA, Feb. 2014, pp. 10–14.
[17] M. Pietras, "Error analysis in the hardware neural networks applications using reduced floating-point numbers representation," in *Proc. AIP Conf.*, vol. 1648, no. 1, 2015, pp. 239–255, doi: 10.1063/1.4912881.
[18] C. Liu, J. Han, and F. Lombardi, "A low-power, high-performance approximate multiplier with configurable partial error recovery," in *Proc. 19th Design, Automat. Test Eur. Conf. Exhib. (DATE)*, Dresden, Germany, Mar. 2014, pp. 1–4.
[19] Y. Miao, M. Gowayyed, and F. Metze, "EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding," in *Proc. IEEE Workshop Autom. Speech Recognit. Understanding*, Scottsdale, AZ, USA, Dec. 2015, pp. 167–174.
[20] D. Wang and X. Zhang. (2015). "THCHS-30: A free chinese speech corpus." [Online]. Available: https://arxiv.org/abs/1512.01882
[21] V. Zue, S. Seneff, and J. Glass, "Speech database development at MIT: Timit and beyond," *Speech Commun.*, vol. 9, no. 4, pp. 351–356, 1990, doi: 10.1016/0167-6393(90)90010-7.
[22] P. Warden. (2018). "Speech commands: A dataset for limited-vocabulary speech recognition." [Online]. Available: https://arxiv.org/abs/1804.03209



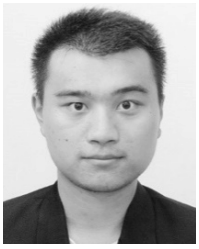
BO LIU was born in Taizhou, Jiangsu, China, in 1984. He received the B.S. and Ph.D. degrees in electrical engineering from Southeast University in 2006 and 2013, respectively.

He is currently a Lecturer with the National ASIC system Engineering Research Center, Southeast University. His research interests include chip architecture design, reconfigurable computing, and related VLSI designs.



HAI QIN received the B.S. degree from Yangzhou University, China, in 2016. He is currently pursuing the M.S. degree with the School of Microelectronics, Southeast University, China.

His current research interests include neural network chip design, speech recognition, and digital–analog hybrid circuit design.



YU GONG received the B.S. degree in mathematics and the M.S. degree in integrated circuits from Southeast University in 2013 and 2016, respectively, where he is currently pursuing the Ph.D. degree in electronics engineering.

His research interests including approximate computing, reconfigurable computing, deep learning accelerator, and related VLSI design.

WEI GE received the B.S. and Ph.D. degrees in electronics engineering from Southeast University in 2006 and 2015, respectively.

He is currently an Assistant Researcher with the Electrical Engineering Department, Southeast University. His research mainly focuses on SoC design technology, reconfigurable computing, and related VLSI design.

MENGWEN XIA received the B.S. degree from Xidian University, China, in 2016. She is currently pursuing the M.S. degree with the School of Microelectronics, Southeast University, China.

Her current research interests include neural network chip design, speech recognition, and digital–analog hybrid circuit design.

LONGXING SHI (SM'06) received the B.S., M.S., and Ph.D. degrees from Southeast University, Nanjing, China, in 1984, 1987, and 1992, respectively.

From 1992 to 2000, he was an Associate Professor with the School of Electronic Science and Engineering. Since 2001, he has been a Professor and the Dean of the National ASIC System Engineering Research Center. He has authored one book and over 130 articles. His current research interests include ultra-low-power IC design.