Chip Design of Convolution Computation for AI Network

Meng-Ju Hsieh
Department of Electronics Engineering
National Yunlin University of Science and Technology
Douliou, Yunlin, Taiwan
M10813047@yuntech.edu.tw

Shih-Chang Hsia
Department of Electronics Engineering
National Yunlin University of Science and Technology
Douliou, Yunlin, Taiwan
hsia@yuntech.edu.tw

Szu-Hong Wang
Department of Electronics Engineering
National Yunlin University of Science and Technology
Douliou, Yunlin, Taiwan
wangsr@yuntech.edu.tw

2021 IEEE 4th International Conference on Knowledge Innovation and Invention (ICKII) | 978-1-6654-2307-6/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICKII51822.2021.9574782

***Abstract*—Artificial intelligence (AI) has been developed a long time ago. In deep learning, the convolution neural network (CNN) architecture is a major focus for image recognition. Owing to the advancement of GPU, most of the computation relied on CPU and GPU to co-compute the AI model. Currently, AI applications are embedded in hardware owing to the smaller size and lower cost. AI chips would be developed quickly in the future. Thus, in this study, we designed a fast convolutional computing chip for AI. There are many convolutional operations in the CNN architecture, and the images are convolved to generate feature maps. This chip is used to achieve the function of convolutional computation. The chip is designed for a cell-based convolution module that was manufactured by using TSMC 0.18um Mixed Signal/RF process, and measured on an ADVANTEST V93000 PS1600 machine. This chip includes nine multipliers and eight adders for 3x3 convolutional computation. The area occupied by the chip is 0.98804x0.9836 mm^2. The power consumption of the chip is 462.8052 uW in an idle state and 1,909.5246 uW in an operating state. The chip can be used for the convolution part of the CNN network architecture.**

Keywords—CNN, Algorithm, Chip, Cell-based

I. INTRODUCTION

In artificial intelligence (AI), the convolutional network architecture is important. The convolutional network architecture uses convolutional operations to extract features. Convolutional operations are mainly performed by a convolutional kernel on a matrix. In a convolutional network architecture, convolution combines two operations, sliding and inner accumulation. After sliding the input image with the convolution kernel and continuously performing the inner product operation, the new image is obtained by manipulating the whole image, which is usually called a feature map. For hardware design, FPGA is widely used [1,2]. Dundar *et al*. [3] designed a novel schedule to reuse the data for computations, which achieved the maximum utilization of resources. Wang [4] proposed a deep learning accelerator unit (DLAU) with three full pipeline processing units that included a tiled matrix multiplication unit (TMMU), part sum accumulation unit (PSAU), and activation function acceleration unit (AFAU).

We proposed the timing-sharing idea to design a fast convolutional chip. Reading the whole picture into the hardware at the beginning requires a large register to store it. Therefore, we use the concept that the convolution operation is a continuous slide of the convolution kernel on the picture and the inner product so that we do not need to read the whole picture at once when processing the convolution. One only reads the part of the convolution kernel with the weight and the relative picture first. Then after a cycle of operation, the convolution kernel does the inner product on the next part of the picture. At this time, we operate again after shifting and reading the value in the registers. After each operation, one of the output pins is pulled to a high level, which allows the receiver to know that the result is read after the operation so that the receiver receives the data correctly. Finally, after the whole operation, the other output pin is pulled high for one cycle to inform the receiver that the convolution of the whole picture has been completed.

II. DESIGN FLOW

This design does not read the pixel values of the whole picture at one time, which requires a large number of registers for storage. Based on timing control, the required convolutional kernel weights and the required picture pixel values are read first. Figure 1 below shows the design flowchart, which is explained as follows.

Initially, the chip is in idle state and waiting for the input kernel_en pin is high when activated. When the chip receives the kernel_en pin high, it first reads the eight input data, and then the eight bits convolutional kernel weights and stores them in the registers, after reading all the eight bits convolutional kernel weights. It then reads the required picture pixel values, for example, if the convolutional kernel is 3x3, it only needs to read the corresponding nine picture pixel values first for operation, not all the picture pixel values at once. It is not necessary to read all the pixel values at once, and then shift the register and read the next required pixel value after the operation. After then, the whole picture is processed. When the registers have the convolutional kernel weights and read the required pixel values, the chip enters the operation state, and the pixel values of the feature map are obtained after the operation state per cycle. After obtaining the pixel values of the feature map, the number of bits obtained from the convolutional operation of the eight bits kernel and the eight bits data is too large. Thus, the pixel values of the feature map need to be quantized to reduce the number of bits and round them to the nearest fraction. After the quantization process, one pixel of the feature map is outputted to I/O pins. After the whole picture is finished, the chip gives the high ready signal for reading data and return to the idle state, waiting for the next block to be calculated.

ISBN: 978-1-6654-2307-6

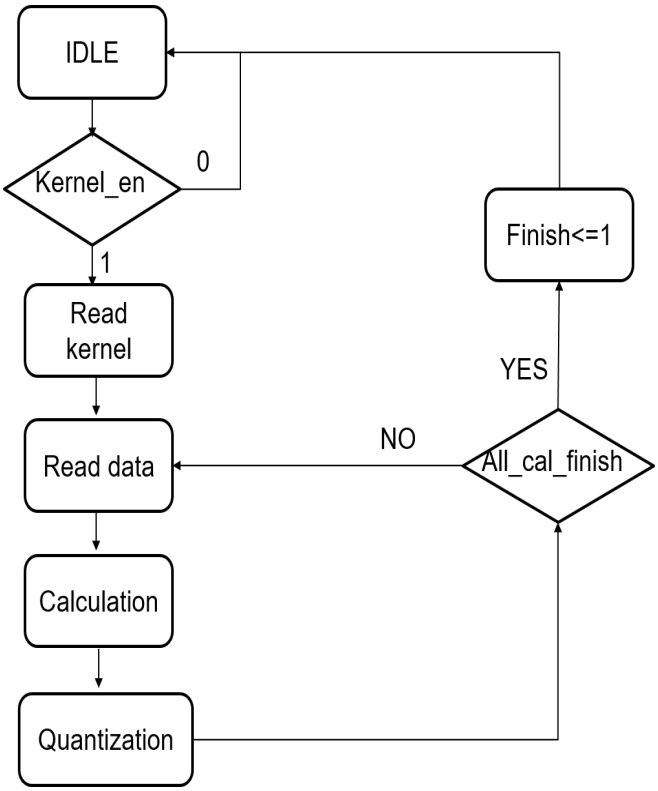

Fig. 1. Design flow chart.

III. System Architecture

The circuit architecture of the input and output connections for the convolution operation is designed. Figure 2 shows the schematic diagram of the circuit architecture for inputs and outputs. Figure 3 shows the architecture of the computing unit, which can calculate one convoluted pixel per cycle.

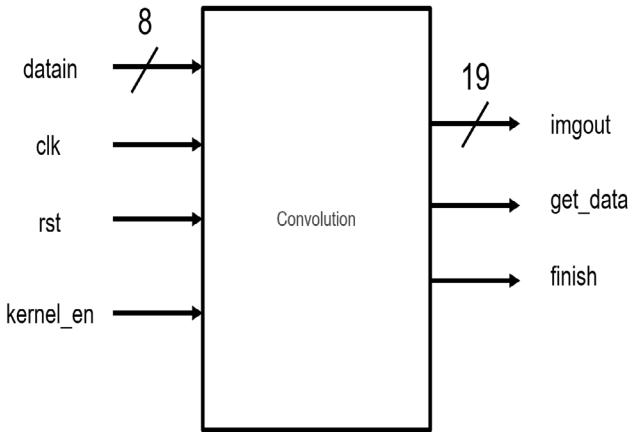

Fig. 2. Diagram of the circuit I/O architecture.

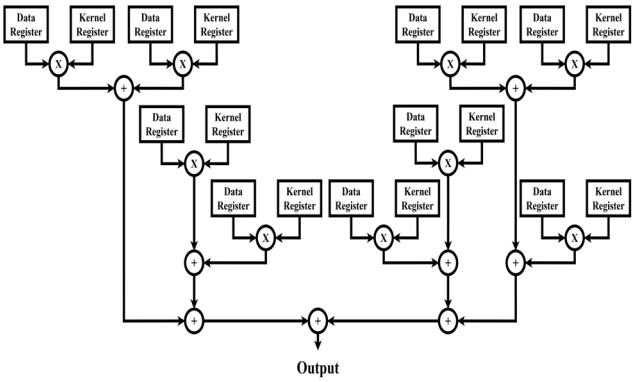

Fig. 3. The architecture of the computing unit.

The clock signal with clk pins is used for the timing control circuit. The rst pin is the function of reset for all registers. Datain pin has a total of eight pins, mainly used to input the eight-bit convolution kernel weight and the eight-bit pixel value of each point of the picture, with time-sharing input. As the input of the octet of the convolution kernel weight and the picture of each point of the octet pixel value uses the datain pins, we set kernel_en pins to let the chip know whether to input the convolution kernel weight or the picture pixel value. After the convolutional operation, the 3x3 convolutional kernel of eight bits and the pixel value of the picture of eight bits. The maximum value with 20 bits is for 3x3 convolution. Due to the chip package pin limit of 40 pins, even if choosing a smaller driving force of the output PAD, the value of the output feature map is still 19 pins at most. Therefore, the value of the convolutional operation still needs to be quantized. Regarding the quantization process, the original result has 20 bits, but it needs to be quantized to 19 bits for the chip package. Thus, we round the lowest value as a decimal point. After quantization, each point of the feature map is exported to the output from the imgout pin. The following explains how to communicate with the receiver. First of all, after the computation, the receiver become to know that the chip has finished the internal computation and when to grab the chip output data. This part all relies on the output pin of get_data, when the pixel value of the feature map is finished, the chip pulls the output pin of get_data to the high level to tell the receiver to grab the data at this time. When the whole feature picture value is finished, the chip pulls the output pin of finish to tell the receiver that the whole picture is finished. At this point, the chip returns to the idle state and wait for the next picture to be input.

IV. Software Simulation

In this section, we present the results of the input to the convolutional chip on the computer. The simulation software is NCVERILOG, and a total of five images are input to the convolutional chip to obtain the output feature maps, which are then compared with the standard answers computed by tensorflow. If the comparison is successful, it means the chip is correct. The five images are also imported in sequence to ensure that the chip can continuously process the convolutional computation. Figure 3 shows the input and output waveforms of the five images, followed by the shape and local waveform of each image compared to the standard tensorflow answer.

Fig. 4. Five pictures of the complete wave pattern.

Figs. 5, 7, and 9 present the shapes of the pictures respectively, while Figs. 6, 8, and 10 show the partial waveforms of the eigenmaps generated after the computation of the pictures respectively.

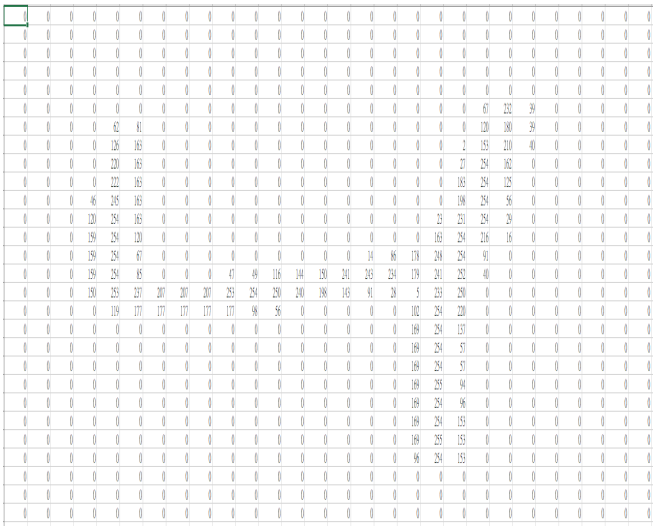

Fig. 5. The first picture.

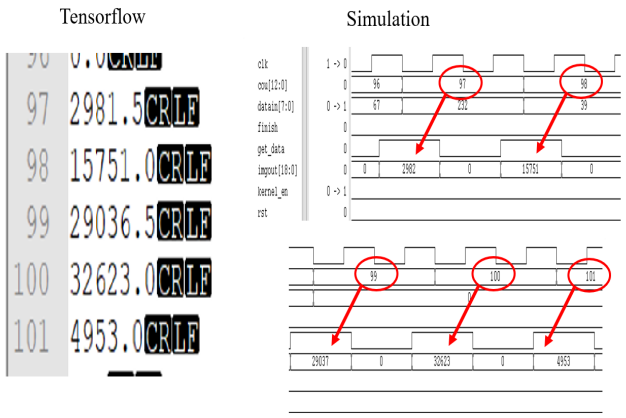

Fig. 6. The first picture of a partial waveform.

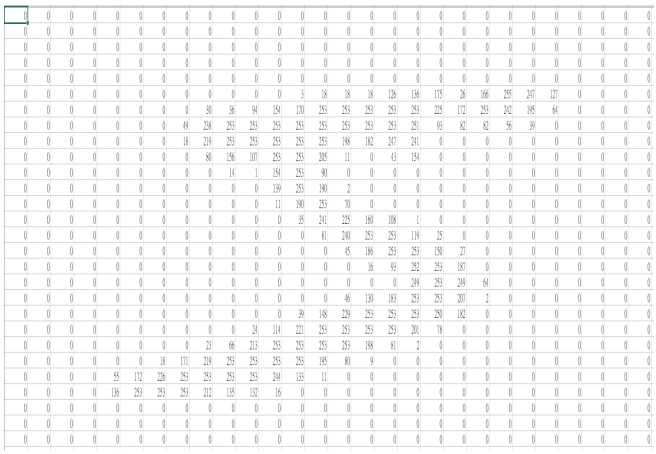

Fig. 7. The second picture.

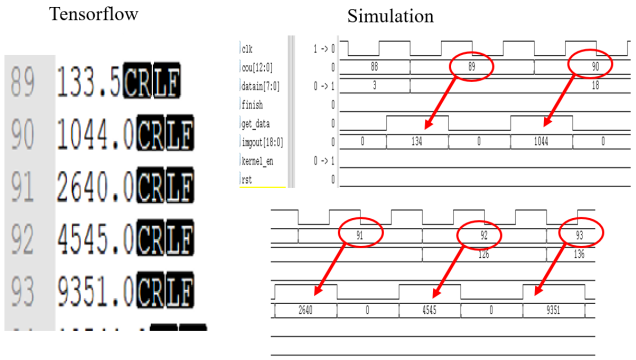

Fig. 8. The second picture of a partial waveform.

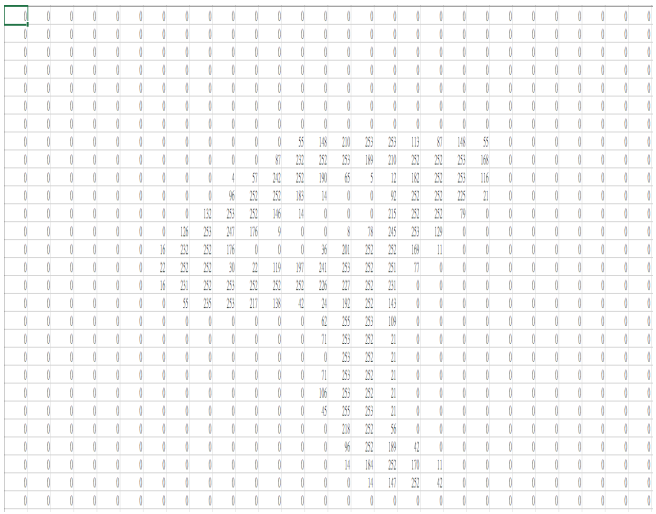

Fig. 9. The third picture.

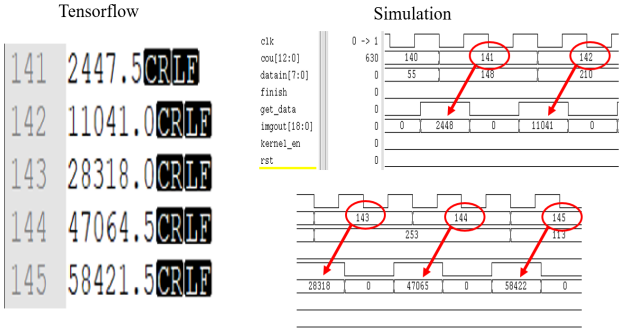

Fig. 10. The third picture of a partial waveform.

V. CHIP LAYOUT AND MEASUREMENT

This chip uses a Cell-Based design. Table 1 shows technology, the chip size, and the number of gate counts used. Figure 11 shows the chip layout.

TABLE I. TECHNOLOGY, THE CHIP SIZE, AND THE NUMBER OF GATES COUNTS USED.

Technology:	TSMC 0.18um Mixed Signal/RF Process
Chip Size:	0.98804 X 0.9836 (mm^2)
Gate Count:	98,372

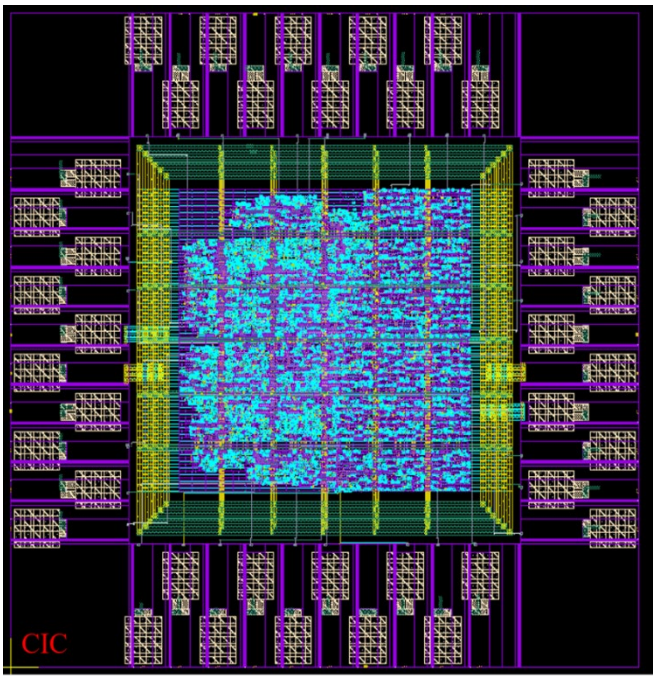

Fig. 11. The chip layout.

The part using ADVANTEST V93000 PS1600 machine measurement is measured for the input of five pictures into the chip. One reads the chip output results and compares them with the expected answer. If it is the same, the chip functions normally. The measurement is divided into five main parts. The first part is the Continuity test to measure the connection between ATE and the chip to check if there is any problem, and the test result is PASS. The second part is the Standby current test, in which the power/ground pad of the DUT is connected to the VDD and GND of the ATE DPS module channel, and no test signal is input. Therefore, the DUT needs to be in a standby state, so theoretically the DUT in this state only consumes a small amount of static power for the CMOS process. If the power and ground are short-circuited, it causes a large short-circuit current consumption. The third part is the functional test, which is conducted under normal ATE and DUT power/ground operation, with the data from Vector Setup to generate driver data and send it to the DUT. The real response data output from the DUT is compared with the expected data set in Vector Setup in a cycle-based manner. If any pattern is different, it means the test fails. The test result is PASS, which means the chip functions normally. The fourth part is the operation current test. The operation current is measured as the average current, so the functional test needs to be continued in order to measure the stable average current. Thus, this step requires the stimulus patterns to be looped. The test result is PASS, and the power consumption is 1,909.5246 uW. The fifth part is the shmoo test, "shmoo" is plotted as a functional test using any two test parameters in the combined working range of the environment parameter. Figure 12 shows the test results, with the horizontal axis showing the operating environment at different frequencies, and the vertical axis shows the environment at different operating voltages, the chip is operating at 20 MHz without using the pipeline stage.

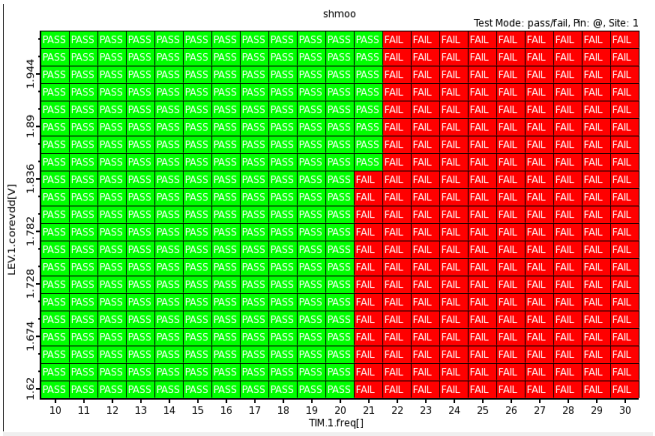

Fig. 12. shmoo plotting.

TABLE II. CHIP TEST RESULT TABLE

Test Items	Result
Continuity test	PASS
Standby current test	PASS Power:462.8052 uW
Functional test	PASS
Operation current test	PASS Power:1,909.5246 uW
Shmoo test	PASS Frequency:20MHz

VI. COMPARISON

Table 3 shows the expected results and the actual chip measurement results. The power consumption performance of the actual chip is less than that in the simulated stage. However, in terms of operating frequency, the chip is expected to reach 40MHz, but the actual chip can only run at 20MHz, which is probably due to the insufficient driving force of the low-power PAD selected during the design. Therefore, the insufficient driving force of the PAD of the chip output causes the frequency to fail to reach the expectation.

TABLE III. CHIP TEST COMPARISON TABLE

	Expected specifications	Test Results
Power Supply (V)	1.8	1.8
Total Current (mA)	2.24	1.06
Power Dissipation (mW)	4.03	1.91
Frequency (MHz)	40	20
Chip size (mm^2)	0.98804 X 0.9836	

VII. CONCLUSIONS

In this research, we designed a fast convolutional computing chip for AI. This chip is used in the present time when artificial intelligence is popular. CNNs account for a large part of AI, and many of them use convolutional processing. Therefore, this chip is used as the hardware for computing. The power consumption of this chip is 462.8052 uW in an idle state and 1,909.5246 uW in an operating state. The chip's operating frequency is 20MHz, which is less than the expected 40MHz. The frequency can be further increased by improving the driving problem. For high speed, more convolution modules can be duplicated in parallel. At present, a large part of artificial intelligence still relies on computing on the computer. Recent AI applications are gradually found in low-power embedded systems. More chips about AI are designed for low-cost systems. AI chips will be a major trend in the future.

REFERENCES

[1] C. Farabet, C. Poulet, J. Y. Han and Y. LeCun, "CNP: An FPGA-based processor for Convolutional Networks," 2009 International Conference on Field Programmable Logic and Applications, Prague, 2009, pp. 32-37, doi: 10.1109/FPL.2009.5272559.

[2] K. Benkrid and S. Belkacemi, "Design and implementation of a 2D convolution core for video applications on FPGAs," Third International Workshop on Digital and Computational Video, 2002. DCV 2002. Proceedings., Clearwater Beach, FL, USA, 2002, pp. 85-92, doi: 10.1109/DCV.2002.1218747.

[3] A. Dundar, J. Jin, B. Martini, and E. Culurciello, “Embedded Streaming Deep Neural Networks Accelerator With Applications”, *IEEE Trans. Neural Networks and Learning Systems,* VOL. 28, NO. 7, pp. 1572-1583, JUL. 2017.

[4] C. Wang, L. Gong, Q. Yu, X. Li, Y. Xie, and X. Zhou, “DLAU: A Scalable Deep Learning Accelerator Unit on FPGA”, *IEEE Trans. Computer-Aided Design of Integrated Circuit and System*,, VOL. 36, NO. 3, pp. 513-517, MAR. 2017