Design of MobileNetV1 SSD Target Detection Accelerator Based on FPGA

Luojia Shi, Chunyu Long, Jitong Xin, Jianhong Yang, Peng Wang(✉), and Fangcong Wang

Institute of Microelectronics, School of Physical Science and Technology, Lanzhou University, Lanzhou, China
{wangpeng,wangfc}@lzu.edu.cn

Abstract. Object detection based on convolutional neural network has become one of the important algorithms in the field of computer target detection. However, due to the speed and power limitation brought by convolution computation, the algorithm of the convolutional network used for target recognition generally needs to be accelerated by the convolution accelerator before it can be effectively deployed to edge computing devices. In this paper, a new architecture of the convolution accelerator is proposed. On this basis, the convolution accelerator is used to build and complete the convolutional acceleration system, and the MobileNetV1 SSD detection algorithm network is realized. The convolution accelerator has a special operation channel, which can normalize convolution kernels of different sizes. Besides, it innovatively optimizes the addition operation mode in the convolution kernels, realizes pipeline convolution calculation, shortens the image recognition time and increases the versatility of the convolution accelerator.

Keywords: Convolution accelerator · Architecture improvement · Target detection · MobileNetV1 SSD · FPGA

1 Introduction

In recent years, target detection technology has become an important research tool in the field of computer vision. As an important content of computer vision research, target detection is applied to judge the category of the target in the static image and generate multiple detection information such as the location of the target by using a certain target detection technology or calling some database for comparative analysis.

The algorithm for target detection is based on the convolutional neural network system, which has high image category detection accuracy, it supports high-precision positioning, and is more powerful than traditional target detection algorithms. Therefore, this algorithm has become the primary choice for the research and development of target detection schemes.

The MobileNet model is a lightweight deep neural network proposed by Google for mobile phones and other embedded devices using deeply separable convolution [1]. Compared with the traditional convolutional neural network, it has the advantage of

S. Yang and H. Lu (Eds.): ISAIR 2022, CCIS 1700, pp. 205–217, 2022.
https://doi.org/10.1007/978-981-19-7946-0_18

effectively reducing the amount of computation, saving a large number of parameters, speeding up the operation speed, and alleviating the training problems caused by excessive fitting [2]. MobileNet SSD [3] (Single Shot MultiBox Detector) algorithm is a target detection model based on MobileNet model, The MobileNetV1 SSD algorithm is very commonly used and has potential, so this paper chooses it as a research tool for the test of the target detection accelerator.

However, the convolutional neural network algorithm used in the target detection is not fast enough in CPU, but its power consumption is greatly limited. Therefore, the task of convolution calculation is generally assigned to a special convolution accelerator for calculation, and such a convolution accelerator generally takes logical resources of FPGA (Field Programmable Gate Array) [4] or ASIC (Application Specific Integrated Circuit) [5] as the carrier.

Although many architectures have been designed using FPGA, single-core convolution accelerators with traditional architectures have fixed operators, slow speed and poor compatibility on the one hand. On the other hand, few breakthroughs have been made in the architecture improvement of the accelerators recently, which requires research and improvement on the architecture of the accelerators.

Although at present there have been many FPGA convolution accelerator, but on the other hand due to the present most accelerators are doing translation software engineers and scholars of the algorithm based on convolution algorithm, using C language, the use of conversion tool for circuit transformation and generate the convolution accelerator [6], the convolution accelerator for algorithm architecture direct translation, Most of these convolution accelerators are single-core convolution accelerators [7]. The convolution accelerator with traditional architecture has fixed operator, slow speed and poor compatibility, so it is necessary to optimize the convolution accelerator from the perspective of hardware. On the other hand, there are few breakthroughs in the architecture innovation of the accelerator recently, which requires research and innovation on the architecture of the accelerator.

In this paper, a special pipeline convolution accelerator is designed, which has high flexibility and calculation speed. The MobileNetV1 SSD detection algorithm network can be implemented by this accelerator and complete the target detection function.

The rest of this paper is organized as follow. We first introduce the MobileNet SSD algorithm introduction in Sect. 2, we then describe the system configuration in Sect. 3. We give the accelerator design in Sect. 4, and experimental results and analysis are provided in Sect. 5. Finally, we conclude this paper in Sect. 6.

2 Algorithm Introduction

The architecture of the MobileNet SSD detection algorithm network is based on the VGG16-SSD algorithm structure [8]. It uses regression-based mode to directly return the category and location of the object in the network, and the concept of region-based makes it possible to use many candidates in the detection process which uses the anchor point method [9], and the idea of multi-scale detection is introduced. Feature maps of different sizes are used. The larger feature maps are used to detect relatively small targets, while the small ones are responsible for detecting large targets and predicting the next target, and generating target detection results by regression of the location [10].

Figure 1 shows the structure of the MobileNet SSD algorithm framework:

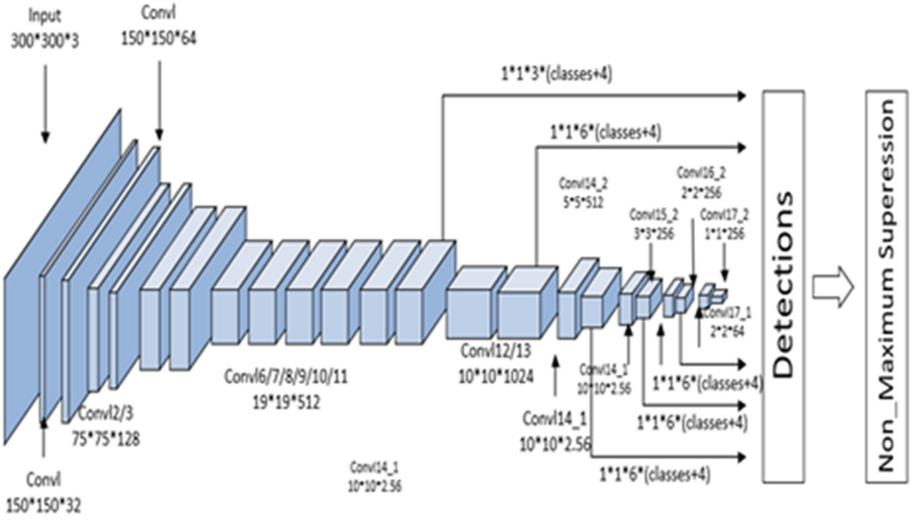

Fig. 1. The diagram of MobileNet-SSD algorithm framework structure

The MobileNet SSD network structure is similar to the VGG-SSD network structure, except that the number of convolutional layers is increased on the basis of the conv13 layer, and 6 layers are extracted from the 8 convolutional layers for target inspection, thereby improving the performance of target detection. The clever decomposition of convolutional kernel accelerates computation and effectively reduces network parameters. The backbone network of the MobileNet SSD algorithm studied in this work is the MobileNet network structure. Its core is the use of depthwise separable convolution [11], which is a typical mobile operating network structure. In the case that the detection accuracy is not affected, but the redundancy of convolution kernel of target detection is greatly reduced, which reduces the computation amount of the algorithm and the size of the model, thus helping to improve the efficiency of target detection.

3 System Configuration

The system as a whole is divided into two parts, namely PL part and PS part [12]. PL is the logic part, which is the FPGA, and the PS part is the ARM. Among them, the PL side mainly includes the accelerator, and the PS side mainly includes the fpgadrv driver, the ssd_detection user program, the Paddlelite framework [13], the mobilenet_v1 SSD model, the lib library, and the opencv library [14]. The PL side mainly realizes the acceleration of the convolution; the PS side mainly transmits the image data and the convolution kernel operator for the PL, and further processing of the result of the convolution operation is also required. The overall block diagram of the system is shown in Fig. 2, and the specific functions of each unit are shown below. The accelerator unit mainly accelerates the convolution part. The data enters the convolution kernel through

the Avalon bus [15], and the weight data and the image data are convolved through the convolution kernel [16]. Fpgadrv: Fpgadrv is the driver of FPGA accelerator on Linux side. Mobilenet_v1 SSD: Mobilenet_v1 SSD stores the SSD model and parameters. Lib [17] contains the library files needed to run the demo. Ssd_detection_src: Contains the demo source code. Opencv: The compiled opencv library.

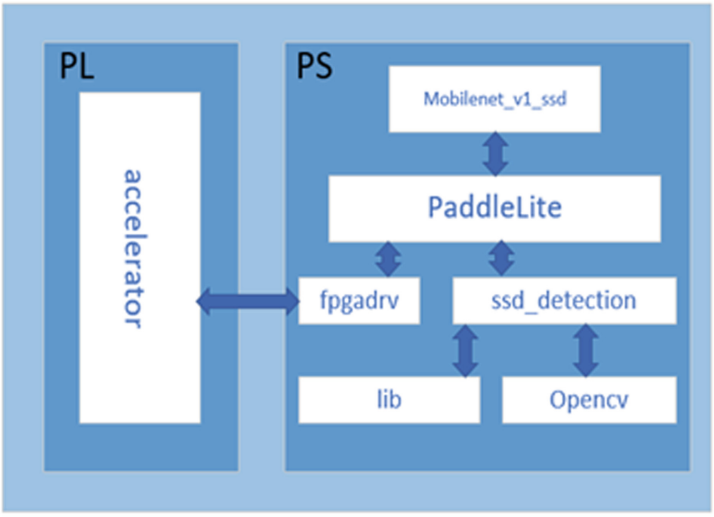

Fig. 2. System overall block diagram

4 Accelerator Design

4.1 Structure of Accelerator

As shown in Fig. 3, The accelerator consists of AvalonBusUpSizer ABUS module, AvalonBusMatrix MTX module, ConvCore module and sysCtrl module.

Among them, the Sysctrl module completes the configuration of the convolution kernel register through the configuration mode register. The AvalonBusUpsizer ABUS module mainly adapts the bus bandwidth. The data bit width of the Avalon bus on the ARM side is 128-bit, and the data bit width of the internal Avalon bus of the convolution acceleration kernel is 512-bit, so bus adaptation is required. The AvalonBusMatrix MTX module judges the data that enters the module, judges whether it is image data or convolution kernel data, transmits different types of data to the corresponding storage module, and provides the ARM side with a read map of the convolution result data.

As shown in Fig. 4, the ConvCore module is composed of din_fifo unit, ConvCompute unit, and dout_fifo unit:

Among them, din_fifo mainly completes the buffering of the input weight data and image data and splicing them into the convolution calculation module. As shown in

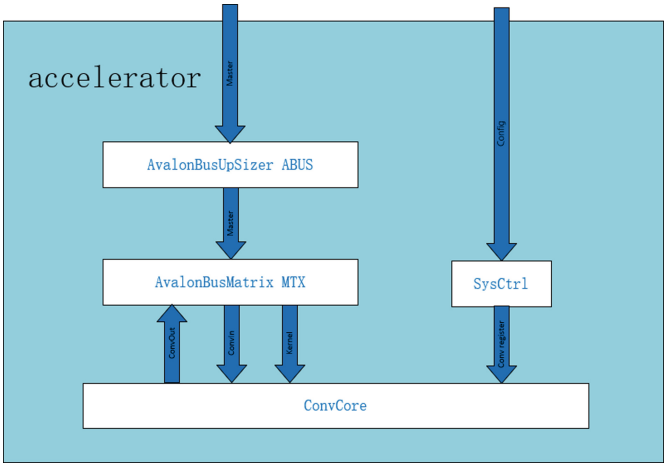

Fig. 3. Structure diagram of accelerator

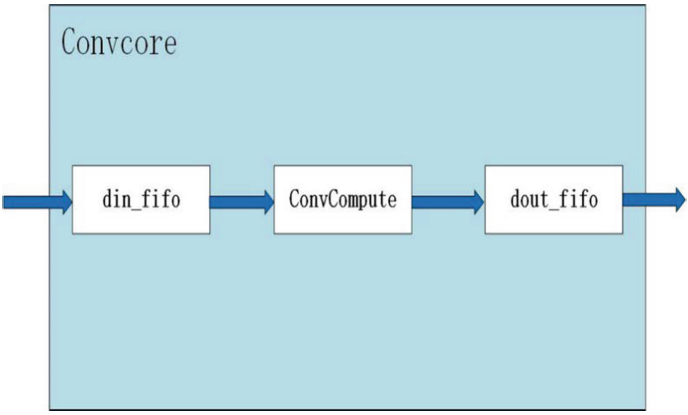

Fig. 4. Structure diagram of ConvCore

Fig. 5, the figure is a structural diagram of din_fifo. Input fifo completes the buffering of the input image data, and Kernal fifo completes the buffering of the weight data. When the data enters, the two fifo units perform pipeline operation at the same time. Convheight counts the number of kernel rows in each batch of kernel, and each 32-bit in the convolution kernel operator is regarded as a minimum arithmetic unit, and then it is expanded to 512-bit by repeating the 32-bit unit. Finally, the weight data and the image data are spliced, and the spliced result is input to the convolution calculation unit.

The structure of dout_fifo is shown in Fig. 6. This module is configured with a buffer with depth of 4096 and width of 512-bit. Each row contains 64-byte and 16 result points.

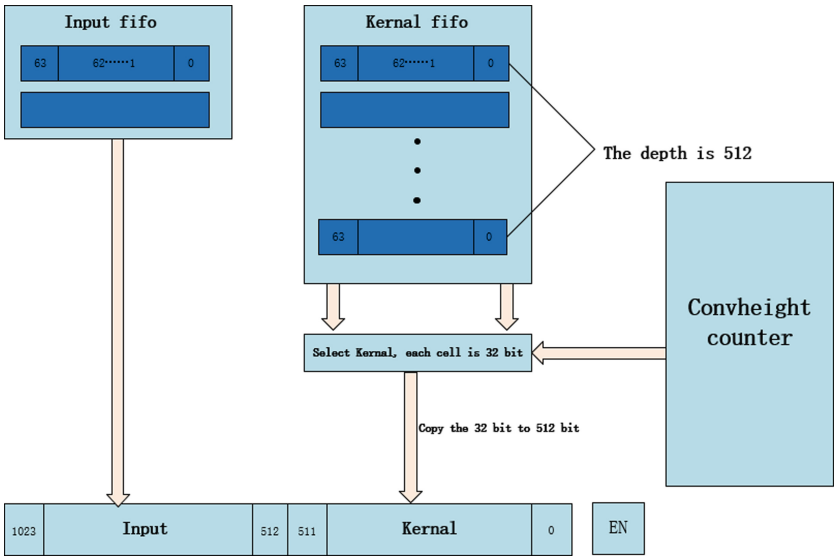

Fig. 5. The diagram of din_fifo module

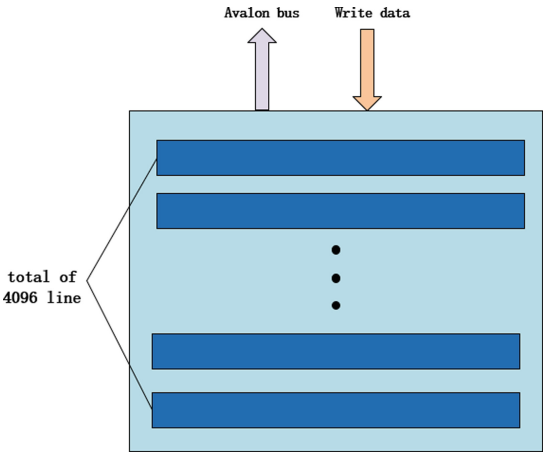

Fig. 6. The diagram of dout_fifo module

The overall architecture of ConvCompute is shown in Fig. 7. The function of the convolution calculation module is to perform multiplication and addition operations by the input image data and weight data, where the weight data and image data are multiplied by the 8 × 8 multiplier unit, and the result enters the adder array unit for addition calculation, where the ConvHeight counts the number of kernel rows in each batch, and the control output is effectively enabled.

For the multiplication calculation unit, as shown in Fig. 8. We use 64 multipliers to perform 8-bit fixed-point multiplication calculations on the input image data and weight

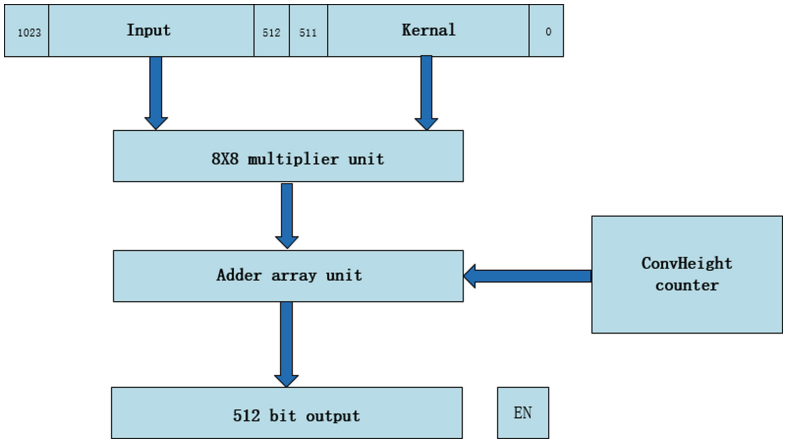

Fig. 7. The diagram of ConvCompute module

data, and insert registers for pipeline operation. Then the result into the addition module for addition operation. The multiplication calculation consumes 32 DSPs (Digital Signal Processor) [18].

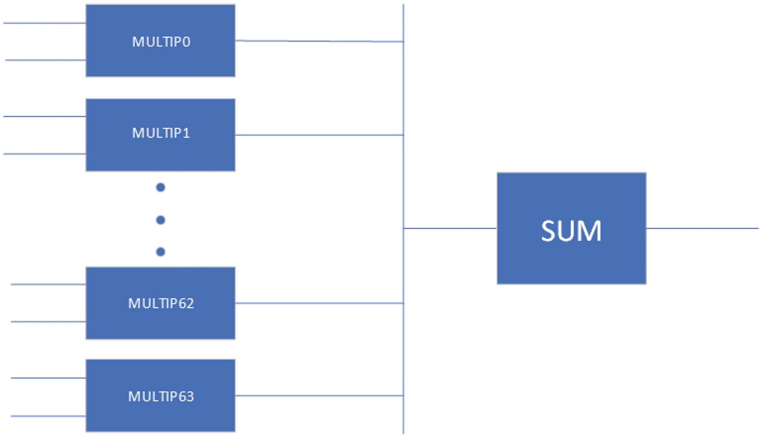

Fig. 8. The diagram of multiplier unit structure

For the addition calculation unit, as shown in Fig. 9, the adder array unit is composed of three-layer adders. The data first passes through 32 adders to complete an addition operation, then the calculation result is added again, next the data is registered. Then through a layer of adder to complete the self-adding operation. For the adder, we chose the Koggle-Stone tree adder parallel algorithm [19], which greatly improves the speed of the addition operation.

The Koggle-Stone adder is a tree adder generated by a parallel algorithm proposed by Peter M. Koggle and Harold S. Stone in 1972. In the tree adder, this kind of adder

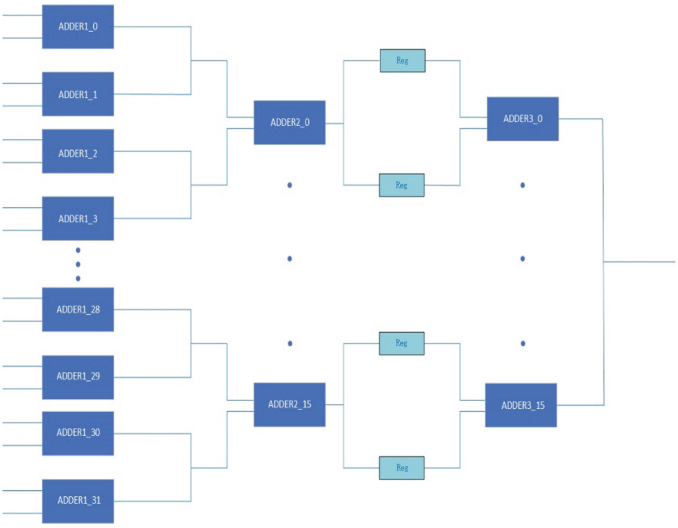

Fig. 9. The diagram of the adder array unit

has the characteristics of low logic layers and low fan-in and fan-out. This type of adder trades area for speed. The simulation of this algorithm is shown in Fig. 10 below.

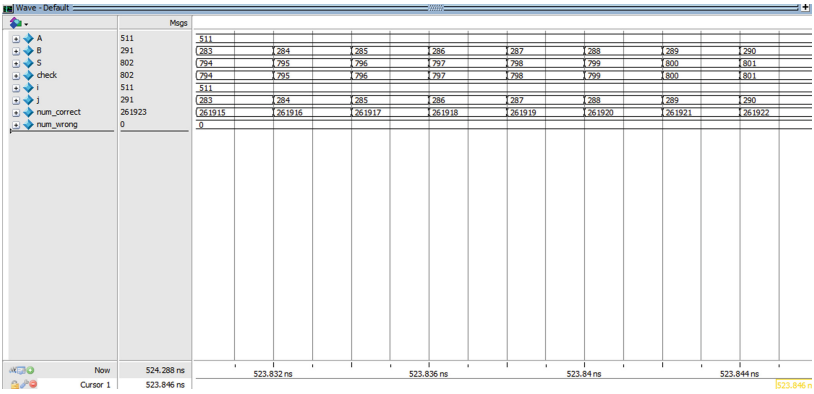

Fig. 10. Simulation result of Koggle-Stone adder

As shown in Fig. 11, the figure is a simulation diagram of the ConvCore module, in which LUT consumes 4226, DSP consumes 64, and bram consumes 2359296. The maximum operating clock frequency of the accelerator is 144.18 MHz.

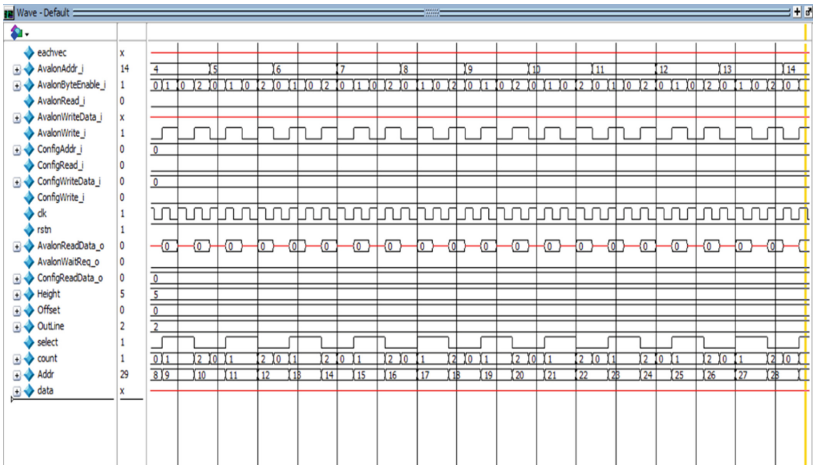

Fig. 11. Simulation of the ConvCore module

4.2 The Overall Use and Function of Accelerator

The driver on the ARM side integrates and assembles the image data according to the same batch of data of different channels. Each row of data is 64-bit, and the row of data contains 16 points of data. Each point data needed for segmentation and assembly, according to each of the 4-byte. The ConvHeight is the amount of data needed to calculate the result of a row divided by 4-byte and insufficient rounded up. And then output to the output memory with 16 points per row, and only the calculation depth (ConvHeight) and the result Depth (ConvOutMax) need to be configured in the configuration parameters, the ARM side needs to configure the kernel of each batch first, and then the data is fragmented and transmitted to the FPAG side (transmitted to the convolution kernel information through the AXI bus [20] to the Avalon bus [15]), and the accelerator performs convolution calculations as it receives data. After the transmission is completed, the data processing is obtained by reading the status register, and then the driver on the ARM side reads the results and reorganizes the data. Theoretically, the processing time is almost equal to the ARM data preprocessing time plus transmission time and then plus read back time, because of the accelerator architecture belongs to stream processing as transmission goes along. It only needs to store part of the input. In this design, the input memory is a 2-line memory structure (128-byte in total), while the Kernel memory is 512 row memory (following the original design), the output memory is 4096 row of memory (which can store 65536 result points). The input memory uses similar the structure of the ping-pong operation [21] ensures the accuracy of the input data (one row of memory is used to store the input data, and the other row is used to output the convolution calculation), and realizes a streaming architecture that transmits and processes at the same time, saving transmission time and storage space, use the limited embedded storage block as much as possible to store output data and kernel information. In addition, the convolution accelerator supports channel fusion and convolution kernel normalization [22] of convolution data, because any convolution calculation is regarded

as a 1 × 1 × multi-channel convolution calculation, which improves the versatility of the convolution accelerator.

4.3 Control Group Experiment

In the case of the same hardware platform and software configuration, only the accelerator was replaced. And the accelerator generated by a company using C language was used as the control group. The accelerator was a traditional fixed-point and single-core convolution accelerator, which was representative. The results are compared with the running results of the convolution accelerator of the new architecture adopted in this paper, as shown in Table 1:

Table 1. Performance comparison of convolution accelerators

Convolution accelerator	Accelerator frequency	Accelerator power consumption	Total system running time
The traditional architecture accelerator generated by C language	142.13 MHz	424.88 mW	3.42 s
The convolution accelerator of the paper used	144.18 MHz	430.55 mW	2.28 s

5 Experimental Results and Analysis

This system driver runs on the Linux-C5 SOC platform [23], the operating system is CentOS7. After the development board is powered on, configure the rbf file generated by quartus to the FPGA, then start the Linux kernel, and the Linux kernel loads the device tree file. The hardware development platform used in this experiment is Quartus 18.1 Standard. The simulation software is modelsimModelSim SE-64 10.5 [24], and the PL accelerator part is written in Verilog language. When recognizing a picture with three targets to be recognized, the recognition results are shown in Fig. 12 below. The total operation time of the system using the accelerator is 2.28 s. The recognition accuracy is more than 90%, and the target detection function can be realized. The working frequency of the accelerator is 144.18 MHz. The power consumption of the accelerator is 430.55 mW. Compared with the traditional convolution accelerator, the overall running time of the system is shortened from 3.42 s to 2.28 s under the premise that the accelerator's frequency and power consumption are not much different. These results show that the performance of convolution accelerator with the new architecture is improved, and its architecture innovation method has positive value.

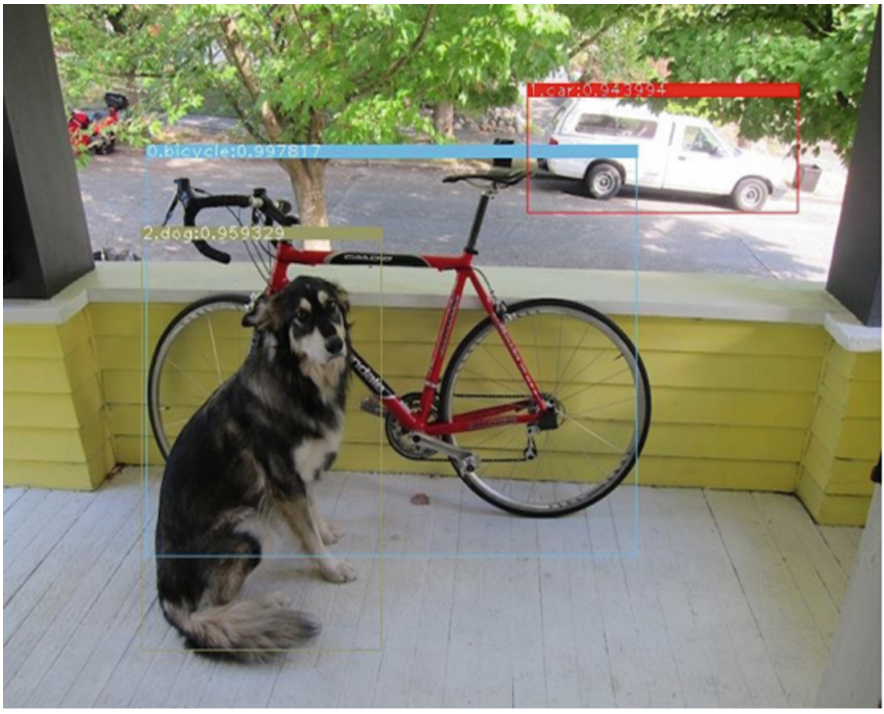

Fig. 12. Recognition effect using the MobileNet SSD network structure

6 Conclusion

The architecture of accelerator commonly used in target recognition network is innovated and the configurable storage computing system is redesigned. At the same time, we also use a high speed Koggle-Stone adder to optimize the addition operation, which has never been used in previous accelerators before. At the algorithm level, this accelerator adopts the method of channel fusion, which processes all the data of a batch of convolution operation at a time, normalizes the accelerator, and converts the acceleration kernel into a multi-channel 1×1 volume, which improves the calculation speed. In addition, because of the normalization process, the accelerator's universality is greatly enhanced. The working frequency of the accelerator is 144.18 MHz, and the power consumption is 430.55 mW. It takes about 2.28 s for the system to recognize images with three targets using the accelerator, and the target detection algorithm is successfully realized. The new architecture and method are of reference value to the future architecture innovation of the accelerator used for target recognition.

References

1. Li, D., et al.: Fast detection and location of longan fruits using UAV images. Comput. Electron. Agric. **190**, 106465 (2021)

2. Bilbao, I., Bilbao, J., Feniser, C.: Adopting some good practices to avoid overfitting in the use of machine learning. WSEAS Trans. Math **17**, 274–279 (2018)
3. Liu, W., et al.: Ssd: Single shot multibox detector. In: Proceedings European Conference on Computer Vision. Springer, Cham, pp. 21-37 (2016). https://doi.org/10.1007/978-3-319-46448-0_2
4. Zhang, C., Li, P., Sun, G., Guan, Y., Cong, J.: Optimizing FPGA-based accelerator design for deep convolutional neural networks. In: Proceedings The 2015 ACM/SIGDA International Symposium. ACM (2015)
5. Pan, H., Wang, M., Li, J.: Design of the key Structure of Convolutional Neural Network Reconfigurable Accelerator Based on ASIC. Proc.2018 International Conference on Network, Communication, Computer Engineering (NCCE 2018). Atlantis Press, pp. 301–304 (2018)
6. Cornu, A., Derrien, S., Lavenier, D.: HLS tools for FPGA: Faster development with better performance. In: Proceedings International Symposium on Applied Reconfigurable Computing. Springer, Berlin, Heidelberg, pp. 67-78 (2011). https://doi.org/10.1007/978-3-642-19475-7_8
7. Johnson, D., Johnson, M., Kelm, J., Tuohy, W., Lumetta, S., Patel, S.: Rigel: A 1,024-core single-chip accelerator architecture. IEEE Micro, Papers **31**(4), 30–41 (2011)
8. Kang, H.J.: Real-time object detection on 640x480 image with vgg16+ ssd. In: 2019 International Conference on Field-Programmable Technology (ICFPT). IEEE, pp. 419–422 (2019)
9. Yamada, Y., Ishida, M., Tsuzuki, S., Tazaki, S.: Simplified anchor point method for fast nearest neighbor search algorithm. Technical Research Report of Electronic Information Commun. Society **96**(78–88), 71–76 (1997)
10. Bao, D., Hao, L.: Survey of target detectionbased on neural network. J. Phys: Conf. Ser. **1952**(2), 022055 (2021)
11. Chollet, F.: Xception: Deep Learning with Depthwise Separable Convolutions. IEEE, pp. 1251–1258 (2017)
12. Zhang, Y., Liu, P., Wang, X.: Design of high speed 1553B bus test system based on ARM and FPGA. J. Changchun University of Science and Technology (Natural Science Edition) (2016)
13. Paddlepaddle: Paddle Lite (2021). https://www.paddlepaddle.org.cn/paddle/paddlelite
14. Culjak, I., Abram, D., Pribanic, T., Dzapo, H., Cifrek, M.: A brief introduction to OpenCV. In: 2012 Proceedings of the 35th International Convention MIPRO. IEEE, pp. 1725–1730 (2012)
15. Xu, N.Y., Zhou, Z.C.: Avalon bus and an example of SOPC system. Semiconductor Technology **28**(2), 17–20 (2003)
16. Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., Chen, T.: Recent advances in convolutional neural networks. Pattern Recognition **77**, 354–377 (2018)
17. Zheng, J.Y., Wang, L.: Static link library in implementing radar simulation system component. Communications Technology (2011)
18. Pailoor, R., Pradhan, D.: Digital signal processor (DSP) for portable ultrasound. Texas Instruments, Application Report SPRAB18A (2008)
19. Kogge, P.M., Stone, H.S.: A parallel algorithm for the efficient solution of a general class of recurrence equations. IEEE Trans. Comput. **100**(8), 786–793 (1973)
20. Math, S.S., Manjula, R.B., Manvi, S.S., Kaunds, P.: Data transactions on system-on-chip bus using AXI4 protocol. In: Proceedings IEEE, pp. 423–427 (2011)
21. Rui, L.I., Xiao, L.I., Wang, Z., Wang, G.: Design of data acquisition system based on SRAM ping-pong operation. J. University of Jinan (Science and Technology) (2015)
22. Sledevic, T.: Adaptation of convolution and batch normalization layer for CNN implementation on FPGA. In: 2019 Open Conference of Electrical, Electronic and Information Sciences (eStream) IEEE (2019)

23. El-Moursy, M.A., Sheirah, A., Safar, M., Salem, A.: Efficient embedded SoC hardware/software codesign using virtual platform. In: Proceedings 2014 9th International Design and Test Symposium (IDT). IEEE, pp. 36–38 (2014)
24. Alsharef, A.A., Ali, M., Sanusi, H.: Direct digital frequency synthesizer simulation and design by means of quartus-modelSim. J. Applied Sciences **12**(20), 2172–2177 (2012)
25. Huimin, L., Zhang, M., Xu, X.:Deep fuzzy hashing network for efficient image retrieval. IEEE Trans. Fuzzy Systems **29**(1), 166176 (2020). https://doi.org/10.1109/TFUZZ.2020.2984991
26. Huimin, L., Li, Y., Chen, M., et al.: Brain Intelligence: go beyond artificial intelligence. Mobile Networks Appl. **23**, 368–375 (2018)
27. Huimin, L., Li, Y., Shenglin, M., et al.: Motor anomaly detection for unmanned aerial vehicles using reinforcement learning. IEEE Internet Things J. **5**(4), 2315–2322 (2018)
28. Huimin, L., Qin, M., Zhang, F., et al.: RSCNN: A CNN-based method to en-hance low-light remote-sensing images. Remote Sensing **13**(1), 62, 2020
29. Huimin, L., Zhang, Y., Li, Y., et al.: User-oriented virtual mobile network resource management for vehicle communications. IEEE Trans. Intelligent Transportation Syst. **22**(6), 3521–3532 (2021)