Design and exploration of neural network microsystem based on SiP

Hao Lv^{1,2} · Shengbing Zhang¹ · Bao Deng^{1,2} · Jia Wang¹ · Desheng Jing² · Yaoqin Chu² · Kun Zhang²

Received: 2 May 2021 / Accepted: 12 August 2021

Published online: 21 August 2021

© The Author(s) 2021

Abstract

In recent years, microelectronics technology has entered the era of nanoelectronics/integrated microsystems. System in package (SiP) and system on chip (SoC) are two important technical approaches for the realization of microsystems. Deep learning technology based on neural networks is used in graphics and images. Computer vision and target recognition are widely used. The deep learning technology of convolutional neural network is an important research field in the miniaturization and miniaturization of embedded platforms. How to combine the lightweight neural network with the microsystem to achieve the optimal balance of performance, size, and power consumption is a difficult point. This article introduces a micro-system implementation scheme that combines SiP technology and FPGA-based convolutional neural network. It uses Zynq SoC and FLASH and DDR3 memory as the main components, and uses SiP high-density system packaging technology to integrate. PL end (FPGA) design Convolutional Neural Network, convolutional neural network accelerator, adopt the method of convolution multi-dimensional division and cyclic block to design the accelerator structure, design multiple multiplication and addition parallel computing units to provide the computing power of the system. Improving and accelerating perform on the YOLOv2_Tiny model. The test uses the COCO data set as the training and test samples. The microsystem can accurately identify the target. The volume is only $30 \times 30 \times 1.2$ mm. The performance reaches 22.09GOPs and the power consumption is only 0.81 W under the working frequency of 150 MHz. Multi-objective balance (performance, size and power consumption) of lightweight neural network Microsystems has realized.

Keywords Micro system · SiP · CNN · YOLO · Target detection

1 Introduction

With the development of deep learning (Deep Learning) technology, Convolutional Neural Network (CNN) has been widely used in machine vision fields such as target detection and face recognition in recent years. Due to the high computational requirements of the CNN algorithm on the operating platform, GPU (Graphics Processing Unit) is usually used to implement it. Deep learning is generally divided into two stages: training and inference. The learning stage can be implemented by GPU or CPU.

However, in the inference stage, it is mostly embedded use scenarios, and the system often has multiple goals such as volume, power consumption, real-time and performance. Constraints, FPGA (Field Programmable Gate Array) implementation has more advantages. FPGA has low power consumption and low latency, and is suitable for parallel acceleration of CNN. In particular, the recent SOPC technology integrates processors such as ARM and FPGA through a high-speed bus to give full play to their respective advantages.

✉ Hao Lv, lvhao20032003@mail.nwpu.edu.cn | ¹School of Computer Science and Engineering, Northwestern Polytechnical University, Xi'an 710072, China. ²Xi'an Aeronautics Computing Technique Research Institute, AVIC, Xi'an 710065, China.

Microsystem technology has received more and more attention with regard to the miniaturization of embedded electronic systems. The SiP and SoC are two important technical approaches for the realization of microsystems. SiP and SoC can stack different types of devices and circuit chips together to build a more complex and complete system with the development of micro-assembly technology. Its integration method is more flexible in terms of R&D cycle and cost. It has advantages and makes up for the shortcomings of SoC. The combination of SiP and SoC is the trend of future miniaturization. CNN accelerator based on SiP or SoC chip is a new research hotspot.

In order to meet the requirements of target detection or terminal guidance in aviation field (such as UAV), a microsystem chip is needed to meet the following characteristics:

- a. It has the minimum system resources such as processor, FPGA and memory.
- b. It can load lightweight neural network model, and its computing power is no less than 2GIPS.
- c. Low power consumption, easy to natural heat dissipation, minimizing in volume and weight.

In this paper, SiP microsystem integrated packaging technology is combined with convolutional neural network to construct a dynamic and reconfigurable deep learning CNN microsystem. The XC7Z020 chip is integrated and packaged with DDR3, Flash memory, etc.. It reduces the volume and power consumption, and improves the signal integrity. An accelerator is built on the FPGA side of the SiP chip based on the YOLOv2_Tiny neural network model, which is controlled by the ARM processor side to form a neural network SiP microsystem. The COCO data is set to verify finally.

2 Related work

2.1 SiP technology status

As it becomes more and more difficult to shrink the feature size of the semiconductor process, whether Moore's Law has reached its limit has become a concern of the semiconductor industry and the entire society. The microscopic size of integrated circuits has reached the atomic limit. Various integrated packaging technologies must be used to solve the bottleneck problem. According to the forecast of the international integrated circuit technology development roadmap, the future development of integrated circuit technology will be concentrated in the following three directions. Continue to follow Moore's Law to reduce the feature size of transistors in order to continue to improve

circuit performance and reduce power consumption, that is, More Moore. Develop in the direction of multiple types and expand Moore's Law, namely More Than Moore [1]. Integrate System on Chip (SoC) and System in Package (SiP) to build a high-value integrated system.

SiP technology integrates a variety of single-function units, such as processors, memories and other chips in the same package, to achieve a basically complete function. SoC and SiP technology are similar. They both integrate systems containing logic components, memory components, and even passive components into one unit. SoC is based on the system-on-chip design concept, which highly integrates the various components that make up the system on a single chip. SiP integrates different chips through a side-by-side and superimposed packaging process based on the packaging perspective. These active devices with different functions and optional passive devices are integrated into a whole to achieve certain functional standard components including devices containing MEMS and optical components.

In the past ten years, the integration of new interconnection and packaging technologies has pushed the MCM technology to a higher level namely SiP (System in Package) technology. It is used to achieve more powerful functions. At this stage, the United States, Europe (Germany, Belgium, etc.) and Asia (Japan, South Korea, Singapore, Taiwan) and other countries or regions are relatively leading in the research and development of system-in-package technology. The United States was the first country to start system-level packaging research. As early as the 1990s, MCM was listed as one of the top ten dual-use high-tech developments. Qiao Yeya Institute of Technology Packaging Research Center is a world-renowned packaging technology research center. Apple's A4 processor and DDR SDRAM use PoP (PoP) technology. Xilinx uses 2.5D stacking technology to package FPGAs. Multiple FPGA chips are interconnected through an interposer board. The interposer board is interconnected with the substrate through C4 solder joints. AMD uses 2.5D stacking technology in AMD's GPU chip packaging structure. Samsung has developed a SiP that integrates ARM processors, NAND flash memory and SDRAM.

2.2 Neural network model

The main task of target detection is to locate the target of interest from the input image, and then accurately determine the category of each target of interest. The current target detection technology has been widely used in daily life safety, robot navigation, intelligent video surveillance, traffic scene detection, aerospace and other fields. At the same time, target detection is the foundation of other advanced visual problems such as behavior

understanding, scene classification, and video content retrieval. However, because there may be great differences between different instances of the same type of object, objects of different types may be very similar. Different imaging conditions and environmental factors will have a huge impact on the appearance of the object. It's a big challenge. Target detection mainly uses convolutional neural network algorithms to achieve at present.

Convolutional neural networks, as one of the current mainstream neural networks, have shown amazing effects in the fields of image classification, target detection, and video encoding and decoding. In the 2012 Image-Net ILSVRC [2] (Large-scale Visual Recognition Challenge), the convolutional neural network model AlexNet proposed by Krizhevsky et al. has a top-5 accuracy rate of 83.6% and a depth of 8 layers. In the next few years, network models such as VGG16, VGG19, and ResNet were successively proposed. While the classification accuracy rate continues to increase, the depth of the network model and the computational complexity are gradually increasing. The current mainstream target detectors include R-CNN, Fast R-CNN, Faster R-CNN, R-FCN based on the region proposal method (region proposal method), and YOLO and SSD [3] of single object detection are all implemented on the basis of convolutional neural networks.

At present, common target detection networks include Region with Convolutional Neural Network (R-CNN), Faster R-CNN and YOLO. Among them, R-CNN is the first method to successfully use CNN for target detection [4]. Compared with traditional methods, it is based on the concept of region suggestion and has achieved higher recognition accuracy. However, R-CNN divides the detection process into two steps: target recognition and positioning, which makes the algorithm execution rate slower. Even if it is simplified, Faster R-CNN cannot meet the requirements of real-time detection [5, 6]. YOLO transforms the target detection problem into a regression problem, and simultaneously performs the two tasks of target recognition and positioning, which greatly improves the detection speed. Since the selection of the size of the YOLO candidate frame is relatively fixed, its detection accuracy has decreased [7]. In order to solve this problem, YOLOV2 [8] was proposed, which can well balance the requirements of target detection accuracy and speed. In this design, we design the hardware of YOLOV2, a simplified version of YOLOV2-Tiny. While ensuring detection accuracy, YOLO-Tiny2 has a smaller network scale and computational complexity compared with other YOLO versions. It has now developed to the YOLOv3 version.

The YOLO target detection model based on the darknet framework was first proposed in 2016. YOLO uses a grid segmentation method to return to the target bounding box and predict the category at the same time. Compared

with the Faster-RCNN target detection model that performed well at the time, YOLO's detection efficiency is significant. It has better applicability to target detection tasks with high real-time requirements. In 2017, the YOLO model was further developed to YOLOv2. Its feature extraction was based on darknet-19. By adding batch normalization to the network, multi-scale feature extraction, and introducing anchor boxes instead of fully connected layers for bounding box regression, etc. Improve to optimize the YOLO classification and positioning effect, and further improve the detection efficiency.

Every time the author of YOLO releases a new model, he will give a lightweight Tiny version, YOLO1-Tiny, YOLOV2-Tiny. The idea of lightweight is to greatly compress the parameters of the network, reduce the convolutional layer, and make the network shallow. And thin. The network structure of YOLOV2-Tiny contains a total of 9 convolutional layers and 6 pooling layers.

2.3 Neural network accelerator

GPU (Graphics Processing Unit) is usually used to implement the CNN algorithm due to the high computational requirements of the CNN algorithm on the operating platform [9]. Deep learning is generally divided into two stages: training and inference. The learning stage can be implemented by GPU or CPU. However, it is mostly embedded use scenarios in the inference stage. The system often has multiple goals such as volume, power consumption, real-time and performance. Constraints, FPGA (Field Programmable Gate Array) implementation has more advantages. FPGA has low power consumption, low latency and reconfigurable. It's suitable for parallel acceleration of CNN. There have been some researches on FPGA-based neural networks. For example, CNN's FPGA digital recognition simulation results can reach a recognition rate of 95.4%. In particular, the recent SOPC technology integrates processors such as ARM and FPGA through a high-speed bus, which can give full play to their respective advantages.

As shown in Fig. 1, current FPGA-based neural network accelerators basically use a similar architecture. Considering the scarcity of FPGA on-chip storage resources (generally less than 10 MB), a three-tier storage architecture is usually used: off-chip storage, on-chip cache, and local registers in the processing unit. The difference from Cache is that the dedicated accelerator can know in advance the data address and amount of data to be read or written in the next memory access, so it can be specifically optimized for the memory access delay. As can be seen from the above figure, the accelerator delay is mainly composed of three parts: memory access delay, on-chip transmission delay and calculation delay. In addition to using the ping-pong buffer design to cover the overlap of delays, the current

Fig. 1 FPGA neural network accelerator framework

optimization strategies to reduce delays can be summarized into three categories [10]: Loop Unrolling, Loop Blocking (Loop Unrolling), and Loop Blocking (Loop Unrolling). Tiling, Loop Interchange.

Lu et al. [11] introduced the convolution calculation method based on the Winograd algorithm into the hardware deployment of convolutional neural networks to reduce the number of multiplications required for convolution, so that the same number of DSP modules can be deployed with higher throughput Accelerator. Alwanietal. On this basis, a new calculation method based on inter-layer fusion was proposed. A prototype accelerator based on FPGA was constructed. On the basis of inter-layer fusion, Xiao et al. An accelerator structure based on the Winograd algorithm and traditional convolution computing components is deployed on the chip [12]. Fowers et al. describes Microsoft's hardware deployment of neural network applications based on FPGA in the data center. The deployment adopts a design concept similar to ASIC. An FPGA can support different network types such as convolutional neural networks and recurrent neural networks at the same time through operator-level application segmentation and scheduling.

3 Methodology

3.1 SIP chip

SIP chip packaging and manufacturing mainly include bare chip stocking, substrate manufacturing and bare

chip packaging. The substrate generally adopts BT resin copper clad rigid substrate. SIP packaging processes mainly include FC (Flip chip) and WB (Wire Bonding). In FC, the IC is placed upside down on the bumps of the substrate and aligned with the corresponding soldering area of the chip, which can make full use of the space and stack three-dimensionally, but the flatness of the substrate is relatively high. If the FC process cannot guarantee the flatness of the substrate, it will reduce the reliability of the chip, resulting in a desoldering failure or a decrease in the yield. WB is a relatively mature traditional wire bonding process with relatively low manufacturing difficulty and higher reliability of the finished product.

After the hardware principle is designed, tools (such as Cadence) are used to carry out the layout and routing of the SIP chip, as well as various simulation tasks, including thermal simulation, structural strength simulation, signal integrity simulation, and power integrity simulation. Figure 2 is a block diagram of the SIP chip design [13].

SIP technology is used to realize the hardware chip platform of the CNN microsystem. This avionics SIP microsystem chip is based on Zynq SoC (XC7Z020), which includes Processing System (PS) and Programmable Logic (PL). The PS side is Dual-core ARM® Cortex™-A9, and the Maximum Frequency is 667 MHz. The PL side contains Artix™-7 FPGA, 28 K Logic Cells, 17600LUTs. It can be used in the PL through programmable logic to build an interface or CNN logic IP core. PS and PL end through the industry standard AXI bus interconnection communication.

Fig. 2 Schematic diagram of SIP chip structure

With Zynq SoC as the core, 1Gbytes of DDR3 Memory, 128Mbit of SPI_FLASH, and interface drivers are configured on the periphery. All these and correspondingly configured resistors, capacitors and other devices are packaged into the SIP chip, forming the smallest computing system. The chip is a plastic package BGA480 package. The chip size is 30×30 mm. The chip thickness is 1.2 mm. Figure 3 is a bare core layout of the SIP chip. The Fig. 4 is a photo of the SIP chip.

3.2 YOLOV2_Tiny neural network

YOLOV2-Tiny can recognize 80 kinds of objects based on the coco data set. The YOLOV2-Tiny network is mainly composed of 9 convolutional layers and 6 pooling layers. There are two convolution kernels with size 3×3 and 1×1, step size 1, and two pooling kernels with size 2×2 and step size 1 and 2.

The input feature image is 416×416×3, and the output of 13×13×425 is output after CNN operation characteristic map.

The basic process of YOLOV2-Tiny [14]:

- (1) Adjust the size of the external image to 416×416 as the input feature image of YOLOV2-Tiny;
- (2) Run the YOLO network to predict the probability of the type of object in the image and the location of the image;
- (3) Finally, after non-maximum suppression, the best results are selected.

The YOLOV2-Tiny network structure selected in this design is shown in Fig. 5, which mainly includes a convolutional layer, a batch normalization layer (BN) and a pooling layer. The BN layer can solve the hidden layer covariate offset problem among them. The input value of the nonlinear transformation function falls into the area sensitive to the

input as to avoid the problem of gradient disappearance and speed up the training process.

In the reasoning stage, the operation of BN layer is shown in formula [15] (1)

$$y = \frac{\gamma}{\sqrt{\text{Var}[x] + \epsilon}} \times x + \left(\beta - \frac{\gamma \times E(x)}{\sqrt{\text{Var}[x] + \epsilon}} \right) \quad (1)$$

In the formula, x is the convolution calculation result, $\text{Var}[x]$ is the mathematical expectation of variance x , $E(x)$ is the mathematical expectation of mean value x , γ is the scaling factor, ϵ is the smaller constant, the prevention denominator is 0, and the bias is β .

The activation function performs a non-linear transformation on each output feature map pixel, which is mainly used to increase the ability of non-linear fitting to the network, generally after batch normalization. The activation function used in YOLOV2-Tiny is Leaky ReLU [16]:

Leaky ReLU function:

$$f = \begin{cases} y, & y \geq 0 \\ 0.1y, & y < 0 \end{cases} \quad (2)$$

3.3 Accelerator architecture

3.3.1 Convolution module

Due to the lack of on-chip storage resources in the FPGA, the convolution loop is divided into blocks, so that each fetch only involves the $T_{ix} \times T_{iy}$ pixel block of the T_{if} input feature map, and the corresponding $T_{if} \times T_{ky} \times T_{kx}$ of the T_{of} convolution kernels. T_{kx} weight parameters, and the $T_{oy} \times T_{ox}$ pixel block of the T_{of} sheet output feature map. Combined with cyclic exchange. Multiplexing the data in the on-chip storage reduce the number of memory accesses and the amount of data to be accessed after determining the data multiplexing mode by rationally

SIP Internal Layout

D9-Zynq Die Local Photo

D6-1553B Driver Die Local Photo

D11-DDR3 Die Local Photo

D10-Flash Die Local Photo

D13 is another backup flash die. No bonding.

Fig. 3 Internal bare core layout of SIP chip

designing the block parameters (T^*). The memory access will not become an accelerator. It's the main bottleneck of time delay. Figure 6 is the loop block.

By partially expanding the number of output feature maps and the number of input feature maps in the convolution loop, $Pof \times Pif$ parallel multiplication calculation units and $Pof \lceil \log_2 Pif \rceil$ depth addition trees are formed, and the multiplication and addition calculations are processed by pipeline [17]. Among them, $Pof = Tof$, $Tif = Pif$. Take $Tof = 2$ and $Tif = 4$ as an example, as shown in Fig. 6. After the pipeline is full, the convolution module reads

Tif pixels at the same position from the corresponding Tif independent input feature map buffers in each clock cycle, and also reads the same position from the $Tof \times Tif$ independent convolution kernel buffers. Weight, $Tof \times Tif$ parallel multiplication units multiplex Tif input pixels for multiplication calculation. Tof addition trees add the products two by one, and after the result and partial sum are accumulated, they are written back to the corresponding output buffer. Figure 7 is the convolution module. The time delay corresponding to the convolution calculation is:

Fig. 4 The photo of the SIP chip

$$\text{Latency Computer} = \frac{\text{Const} + N_{ky} \times N_{kx} \times T_{oy} \times T_{ox}}{\text{Freq}} \quad (3)$$

From the convolution calculation principle, it can be seen that the new convolution module consumes most of the multiplier and adder resources in the FPGA [18]. The data accuracy used in the operation is the main factor in the consumption of resources. It can be obtained by changing the different data accuracy. With the corresponding resource consumption, the DSP (Digital Signal Processing) and LUT (Look-Up Table) used by the adder and multiplier with 16-bit fixed-point data accuracy are far less than those with 32-bit floating-point data accuracy. The consumption of resources further verified the feasibility of fixed-point floating-point numbers.

Fig. 5 YOLOv2_Tiny Network structure

Fig. 6 Loop block

Fig. 7 Convolution module **Fig. 8** The relationship between detection accuracy and quantization bit width

In this paper, the activation value and weight of YOLOV2-Tiny are quantized between 6 and 32 bits, and the corresponding relationship between the accuracy and the quantization bit width is shown in Fig. 8. In order to minimize the pressure of hardware calculation while ensuring the detection accuracy, in this design, the network activation value and the weight width are set to 16 bits.

3.3.2 Pooling module

The effect of pooling is to reduce the dimensionality of the feature map and reduce overfitting. The pooling layer in YOLOv2 is all the maximum pooling layer with $N_kx = N_ky = S = 2$. For a single input feature map, slide a window with a size of 2×2 and a step size of $S = 2$, and take the largest pixel as the output. The max pooling operation is similar to the convolution operation, the difference is:

- (1) Only a single input feature map needs to be sampled.
- (2) The arithmetic unit is not a multiplier and an adder, but a comparator. In each clock cycle, the pooling module reads a pixel at the same position from the independent input feature map buffer of T_{pool} and compares it with the current maximum value. At the same time, there are T_{pool} comparators that compare different input feature maps. After the comparison, the maximum value obtained is written into the output buffer. As shown in the overall architecture diagram of the accelerator in Fig. 8, since each layer shares the input and output modules, so the parallelism of the pooling modules.

Use input and output modules, so the parallelism of the pooling module is $T_{pool} \leq \min(Tif_{max}, Tof_{max})$, and $T_{pool} = Tif = Tof$.

The hardware structure of the pooling layer is shown in Fig. 9.

3.4 Roofline module

The Roofline model is an intuitive performance evaluation model used to evaluate and explore the peak performance and optimization directions that can be achieved for various applications for a given hardware system (ie, known peak computing performance and theoretical maximum bandwidth). Therefore, it can also be used to evaluate and explore the design space of FPGA-based neural network accelerators [19]. A typical Roofline model is shown in Fig. 10.

Among them, the computational upper limit (Computational roof) represents the theoretical peak computing performance that a given hardware system can achieve, and the bandwidth upper limit (Bandwidth roof) represents the maximum bandwidth for the given hardware

Fig. 9 Pooling module **Fig. 10** Poofline module

system to interact with the memory. The abscissa calculation density (Operation Intensity) represents the average amount of calculation involved in reading or writing unit data from the memory. It is generally obtained by dividing the total calculation amount by the total memory access data volume. The ordinate represents the system and each application can achieve performance. Application 1 falls in the bandwidth-constrained area (blue area) in Fig. 9. If the total amount of calculation and memory access of application 1 does not change, the maximum performance can only reach the bandwidth limit. Application 2 falls in the area of the upper performance limit (orange area). At this time, application 2 can reach the maximum performance of the system. In actual application, it is hoped that the application can fall in the orange area, and the closer the upper right calculation limit is, the better. There are two main reasons: (1) The closer to the upper limit of performance, the closer the

actual performance is to the maximum performance, and the system utilization Higher. (2) The closer to the right, the greater the calculation density, that is, the greater the amount of calculation corresponding to unit memory data exchange, and the better the energy efficiency.

The multiple design parameter combinations found can be evaluated by the Roofline model introduced in Sect. 2.3.5. The calculated density I and performance P corresponding to the accelerator design can be obtained by the following formula at this time:

$$I = \text{Model Operations} / N_{MA}^{\text{total}} \quad (4)$$

$$P = \text{Model Operations} / \text{Latency}^{\text{total}} \quad (5)$$

N_{MA}^{total} indicates the amount of data accessed by the accelerator (read and write back), $\text{Latency}^{\text{total}}$ is the accelerator total delay.

4 Experimental design and analysis

4.1 Functional testing

The collected images which stored in the internal memory are obtained from the USB camera as shown in Fig. 11. They are pre-processed by the ARM and handed over to the YOLOv2-Tiny accelerator on the FPGA side for target detection. The relevant data is still stored in the memory after detection. After ARM post-processing, the image with the detection category and location is written back to a certain address in the memory, and is displayed on the display screen by the HDMI controller on the FPGA side.

The CPU on the PS side can control all interfaces between PS and PL. The accelerator uses CPU scheduling to input the feature map and YOLO network parameters to be input into the DDR3 buffer, and interact with the peripheral interface circuit through the AXI (Advanced eXtensible Interface) bus. The CPU uses the AXI bus to read the calculation results of the acceleration circuit, and executes the image preprocessing application and displays it on the PS side. On the PL side, the data in the external DDR3 is read to the on-chip RAM. The hardware design bit stream file (Bit) and the design instruction file (Tcl) are passed to the PS side., the various YOLO accelerators described above after identification and analysis. The module design is deployed on the FPGA. The YOLO IP core is added to the PYNQ Overlay library for top-level calls. The internal accelerator architecture design of the SIP chip is shown in Fig. 12.

CNN model: YOLOv2-Tiny (416×416), the data set is the COCO data set. The test platform uses SIP chip: XC7Z020

(Dual-core ARM-A9 + FPGA), in which the number of BRAM_18Kb, DSP48E, FF and LUT resources of FPGA are 280, 220, 106,400 and 53,200 respectively. Dual-core ARM-A9, clock frequency 667 MHz, 1 GB DDR3 memory. Use Vivado HLS 2018.2 HLS for accelerator design, and Vivado v2018.2 for synthesis and placement and routing. The power consumption uses the external measurement system power consumption of the VC470 power consumption meter, and uses the thermal design power (TDP) as the total power consumption.

4.2 Model parameters

The YOLOv2-Tiny accelerator is built in the XC7Z020 inside the SIP chip using the method of loop unrolling. (Noy, Nox), Nof, Nif, (Nky, Nkx), S represent output characteristics respectively Figures, the number of output feature maps, the number of input feature maps, the size and step size of the convolution kernel. The convolution loop is divided into blocks, so that Tof pixel blocks of the size of Tiy×Tix of the input feature map and the weight corresponding to the size of Tif×Tof×Nky×Nkx are required each time. The output feature map pixels are reused as much as possible at this time. The intermediate results of the output feature map are kept in the on-chip buffer or local register. All output feature map pixels will be written back off-chip once only after the final result is obtained. The convolution kernel parameters must be read off-chip ([Noy/Toy] × [Nox/Tox]) times and input the feature map [Nof/Tof] times. The input pixel block width and height (Tiy, Tix) formula required to generate the output feature map of Toy×Tox size is as follows:

Fig. 11 The overall framework of the target detection system

Fig. 12 System hardware accelerator architecture

$$T_{iy} = (T_{oy} - 1) \times S + N_{ky} \quad (6)$$

$$T_{ix} = (T_{ox} - 1) \times S + N_{kx} \quad (7)$$

Mainly because the current mainstream convolution kernel is small, such as 1×1 or 3×3 , which can be read in the on-chip cache, $T_{kx} = N_{kx}$, $T_{ky} = N_{ky}$. Considering that the convolutional layer tends to occupy more than 90% of the network amount of calculation, here is the design situation of the parameters of each layer of the convolutional layer shows in Table 1.

The Zynq7020 chip has 220 DSPs. Each multiplier will consume 1 DSP under the dynamic fixed-point 16-bit data accuracy according to Table 4–1. The adder does not consume DSP. Considering that the current working clock is 150 MHz, the upper limit of system performance is 66 GOPS ($220 \times 150 \times 2 / 1000 = 66$).

The memory model is 32bit DDR3-1066, and the theoretical maximum bandwidth is 4.264 GB/s. The actual

bandwidth efficiency is only about 75% [20]. The maximum bandwidth is 3.198 GB/s. However, the improving can be seen from that the signal integrity of DDR3 due to the use of SIP technology [21]. The actual bandwidth has been slightly increased, which can reach 3.312 GB/s.

The design space of the accelerator is evaluated with the Roofline model. For the position of point A ($T_{if} = 4$, $T_{of} = 32$, $T_{oy} = 26$, $T_{ox} = 26$, 23.1GOPS) (estimated by Eqs. 4, 5) in the figure, as shown in Fig. 13, the design is in performance limitation area.

4.3 Test results

A test board equipped with a SIP chip is used in the test environment. The SIP chip integrates XC7Z020, 1Gbytes of DDR3(1066 Mb/s), and 128Mbit FLASH. The operating frequency of the single-core ARM-A9 is set to 400 MHz, and the FPGA frequency is set to 150 MHz. The test platform is shown in the Fig. 11. The YOLOv2-Tiny model accelerator is built into the FPGA of Zynq SoC. The data

Table 1 Convolutional layer accelerator parameter list

Name	Input	Input expansion	Convolution kernel	T _{if}	T _{of}	T _{ox} × T _{oy}	T _{ix} × T _{iy}
Conv1	416 × 416 × 3	448 × 448 × 3	3 × 3 × 16	3	16	26 × 26	28 × 28
Conv2	208 × 208 × 16	224 × 224 × 16	3 × 3 × 32	4	32	26 × 26	28 × 28
Conv3	104 × 104 × 32	112 × 112 × 32	3 × 3 × 64	4	32	26 × 26	28 × 28
Conv4	52 × 52 × 64	56 × 56 × 64	3 × 3 × 128	4	32	26 × 26	28 × 28
Conv5	26 × 26 × 128	28 × 28 × 128	3 × 3 × 256	4	32	26 × 26	28 × 28
Conv6	13 × 13 × 256	15 × 15 × 256	3 × 3 × 512	4	32	13 × 13	15 × 15
Conv7	13 × 13 × 512	15 × 15 × 512	3 × 3 × 1024	4	32	13 × 13	15 × 15
Conv8	13 × 13 × 1024	15 × 15 × 1024	3 × 3 × 512	4	32	13 × 13	15 × 15
Conv9	13 × 13 × 512	13 × 13 × 512	1 × 1 × 425	4	32	13 × 13	13 × 13

Fig. 13 Accelerator model evaluation diagram

Fig. 14 The diagram of the test environment

Fig. 15 Target detection test results

set corresponding to the model is the COCO data set. The Vivado HLS is used for accelerator design, Vivado v2018.2 is used for synthesis, placement and routing. The camera is OV5640 which is a CMOS type digital image sensor. The sensor supports the output of images up to 5 million pixels (max resolution 2592×1944). The VC470 power consumption meter is used to externally measure the system power consumption [22]. The thermal design power is used as the total power consumption [23]. The diagram of the test environment is shown in Fig. 14. The situation of target recognition through the CNN micro system is shown in Fig. 15.

Table 2 is the logic resource consumption table of the SIP microsystem. DSP is mainly used for the design of adders and multipliers in the convolution module,

Table 2 SIP test programmable logic resource usage

Resources	BRAM_18K	DSP48E	Flip-Flops	LUT
Total resources	280	220	106,400	53,200
Involved dosage	158	150	19,380	26,072
Utilization/%	56	68	18	49

BRAM_18Kb is used to implement a large storage cache, and the AXI Master interface will also consume BRAM to implement the interface cache. The 16-bit fixed-point precision convolution module consumes 19,968 LUTs, which consumes about 26,072 LUTs in total.

The comparison is shown in Table 3. The design in [24] uses the YOLOv2-Tiny network to map all layers to the FPGA, but does not use the ping-pong buffer, which results in the memory access and data transmission delays that cannot overlap with the calculation delays. The design [25] converts the convolution operation into a general matrix multiplication operation, but converting the convolution into a general matrix multiplication requires copying and reordering the convolution kernel parameters before each calculation, which adds additional delay and the complexity. Literature [26] uses the optimized YOLOv2-Tiny model accelerator, and its power consumption and performance are close to the original design, but SIP technology is not used, and the area and volume of the PCB design are larger. Literature [27] uses the YOLOv2 model, which consumes more resources. Although the performance is higher, the power consumption is greater. In contrast, due to the SIP technology and optimized YOLOv2-Tiny model accelerator, this design achieves a higher balance in terms of performance, power consumption, and area (volume).

5 Conclusions

In this paper, a target detection micro system is constructed based on the neural network algorithm YOLOv2_Tiny and structured with SIP integrated packaging technology. The accelerator is designed by the method of cyclic block expansion, convolution and multi-dimensional expansion. The parallel computing units are designed to provide the computing power of the system. the number of memory accesses and the amount of data is reduced. The Rooflin model is used for evaluation. The microsystem encapsulates Zynq-7020 and DDR3 memory together, reducing the chip area (volume) and improving signal integrity. It focuses on the accelerator design and parameter structure, and conducts testing and analysis. The target detection SIP microsystem is well balanced in many aspects such as performance, power consumption and volume. It provides an idea and method for the development of neural network micro-system. Some analog circuits are not included in the microsystem package, including power conversion circuits, clock circuits, and reset circuits. The feasibility of incorporating these circuits into the SIP chip can be considered in the future. At the same time, the target detection algorithm model has also been updated. For example, the latest YOLOv5 algorithm model is also producing its Tiny version [28]. Of course, the complexity of these algorithms is higher, the resource consumption is greater. Whether higher performance processors and FPGA, whether it will cause higher difficulties in the packaging process, these are still worth studying. These await future target detection micro-system analysis and research.

Table 3 Test comparison

Accelerator	CNN model	Architecture	Frequency/ MHz	Power/W	Performance/ GOPs	Area/mm ²
Ref [24]	YOLOv2_Tiny	Zynq7020	142	–	7.46	49 × 42
Ref [25]	YOLOv2_Tiny	Cyclone V	117	–	19.45	–
Ref [26]	YOLOv2_Tiny	Zedboard	150	0.83	21.97	50 × 45
Ref [27]	YOLOv2	PYNQ-Z2	150	2.96	26.98	49 × 40
This work	YOLOv2_Tiny	SIP(7020 + ddr3)	150	0.81	22.09	31 × 31

Declarations

Conflict of interest The author(s) declare that they have no competing interests.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. Faber H, Anthopoulos TD (2019) Adding a new layer to “more than moore.” *Nat Electron* 2(11):497–498
2. Russakovsky O, Deng J, Su H et al (2015) Imagenet large scale visual recognition challenge. *Int J Comput Vision* 115(3):211–252
3. Liu W et al (2016) SSD: single shot multibox detector. Springer, Cham
4. Bharati P, Pramanik A (2019) Deep learning techniques—R-CNN to Mask R-CNN: a survey. In: 1st International conference on computational intelligence in pattern recognition (CIPR 2019)
5. Wang K, Zhou W (2019) Pedestrian and cyclist detection based on deep neural network fast R-CNN. *Int J Adv Robot Syst* 16(2):1729881419829651
6. Ren S, He K, Girshick R, Sun J (2017) Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Trans Pattern Anal Mach Intell* 39(6):1137–1149
7. Chen W, Huang H, Peng S, Zhou C, Zhang C (2020) Yolo-face: a real-time face detector. *Vis Comput* 37:805
8. Li R, Yang J (2018) Improved YOLOv2 object detection model. In: 2018 6th international conference on multimedia computing and systems (ICMCS)
9. Liu C (2021) YOLOv2 acceleration using embedded gpu and fpgas: pros, cons, and a hybrid method. *Evolut Intell*. <https://doi.org/10.1007/s12065-021-00612-y>
10. Ma Y, Cao Y, Vruthula S et al (2017) Optimizing loop operation and dataflow in FPGA acceleration of deep convolutional neural networks. In: Proceedings of the 2017 ACM/SIGDA international symposium on field-programmable gate arrays. New York: 45–54
11. Lu L, Yun L, Xiao Q, Yan S (2017). Evaluating fast algorithms for convolutional neural networks on FPGAs. In: IEEE international symposium on field-programmable custom computing machines. IEEE
12. Shi F, Li H, Gao Y, Kuschner B, Zhu S (2019). Sparse winograd convolutional neural networks on small-scale systolic arrays. In: The 2019 ACM/SIGDA international symposium
13. Lv H, Zhang S, Han W, Liu Y, Liu S, Chu Y (2020) Design and realization of a aviation computer micro system based on SiP. *Electronics* 9(5):766
14. Chen C, Wei Y, Jun X et al (2019) Design and implementation of FPGA-based deep learning object detection system. *Appl Electron Tech* 45(8):40–43
15. Cen Li, Guang-hui He (2020) Design of FPGA-based neural network accelerator for real-time objective detection. *Microelectron Comput* 37(7):6–11
16. Huang H, Liu Z, Chen T, Hu X, Zhang Q, Xiong X (2020) Design space exploration for yolo neural network accelerator. *Electronics* 9:1921
17. Nakahara H, Fujii T, Sato S (2017). A fully connected layer elimination for a binarized convolutional neural network on an FPGA. In: 27th international conference on field programmable logic and applications (FPL). IEEE
18. Wang Z, Xu K, Wu S, Liu L, Wang D (2020) Sparse-yolo: hardware/software co-design of an fpga accelerator for yolov2. IEEE Access. <https://doi.org/10.1109/ACCESS.2020.3004198>
19. Zhang C, Li P, Sun G et al (2015) Optimizing FPGA-based accelerator design for deep convolutional neural networks. In: Proceedings of the 2015 ACM/SIGDA international symposium on field-programmable gate arrays. New York: 161–170
20. Zynq-7000 SoC Technical Reference Manual. https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf, 2019-03-26
21. Yi J, Wang M, Bai L (2021) Design of ddr3 sdram read-write controller based on fpga. *J Phys Conf Ser* 1846(1):012046
22. Huang X, Wang Y, Zhong Z (2019) Bonding technology of thermal conductive adhesive at interface. *Electronics Process Technology*
23. Sang KK, Oh D, Hwang S, Bang WL, Kim TH (2019) Electrical and thermal co-analysis of thermally efficient SiP for high performance applications. In: 2019 Electrical design of advanced packaging and systems (EDAPS)
24. Yunke Z, Dan L (2019) Real-time implementation of improved TINY YOLO vehicle detection algorithm based on Zynq SoC hardware acceleration. *J Comput Appl* 039(001):192–198
25. Wai YJ, Bin-Mohd YZ, Bin-Salim SI et al (2018) Fixed Point Implementation of Tiny-Yolo-v2 using OpenCL on FPGA. *Int J Adv Comput Sci Appl* 9(10):506–512
26. Chen C (2019) Research of scalability on FPGA-based neural network accelerator (Doctoral dissertation, Jiangnan University China)
27. Qidi X, Bingyong Y, Jiale Z, Huifeng W (2020) Architecture design of target detection hardware accelerator based on heterogeneous FPGA. *J East China Univ Sci Technol* 31(1691):1–10
28. leamsaard J et al (2021) Deep learning-based face mask detection using yoloV5. In: 2021 9th international electrical engineering congress (IEEECON 2021): 428–431

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.