Parallelism Factor of 2D Convolution Circuit in 28-nm CMOS Technology

Chun-Yen Yao (B04901032)

Po-Jen Chen (B04901144)

***Abstract*—Convolutional neural network is now common in image recognition, so a reasonable convolution architecture can greatly improve energy efficiency of real-time edge-computing devices nowadays. By simulation of one convolution layer with 8×8 input image, 2×2 filter weight, and 7×7 output image, we find that the design with four process elements optimizes this task, and earn 2.87× energy reduction under the same throughput compared to the reference design without parallelism.**

Keywords-2D-convolution; parallelism; energy-delay tradeoff;

I. INTRODUCTION

Convolutional neural network (CNN) is now common in image recognition since convolution (CONV) layers can decrease the number of training weights compared to that of fully-connected ones. It is also hardware-friendly because fewer parameters are required to store in memory and fewer operations are needed to complete a task. These advantages make CNN suitable for edge-computing devices with low power and local real-time applications (e.g., self-driving cars) [1]. In a typical CNN, convolutions occupy most of the operations. For instance, LeNet-5 includes 344,468 connections in the hidden layers, 326,424 (94.7 %) of which are for convolutions [2]. As a result, a hardware implementation with reasonable parallelism at the cost of area can greatly improve energy and throughput [3]. Hence, our work focuses on comparison among different parallelism factors (PFs) of convolution architectures. Delay-supply, energy-supply, and energy-delay analysis will be presented in this report.

II. DESIGN SPECIFICATIONS

Fig. 1 shows a visualization of convolution operations of one input feature map (ifmap), one filter, and one output feature map (ofmap) [4] (ignoring bias). The computation is defined as

$$\boldsymbol{O}[x][y] = b + \sum_{k=0}^{C-1}\sum_{i=0}^{R-1}\sum_{j=0}^{S-1} \boldsymbol{I}[k][Ux+i][Uy+j]\times\boldsymbol{W}[k][i][j]\,,$$

$$0 \le y < E,\ 0 \le x < F,\ E = (H-R+U)/U,\ F = (W-S+U)/U, \quad (1)$$

where $\boldsymbol{O}$, $\boldsymbol{I}$, and $\boldsymbol{W}$ are the matrices of the ofmap, ifmap, and filter, respectively. U is a given stride size, and b is bias. For simplicity, we define our specifications as C=1, H=W=8, R=S=2, U=1, b=0, and hence E=F=7. To complete the task, the number of operations (one multiplication-accumulation, MAC) is $E\times F\times C\times R\times S$=196. Besides, since C=1, we call our convolution as 2D-convolution.

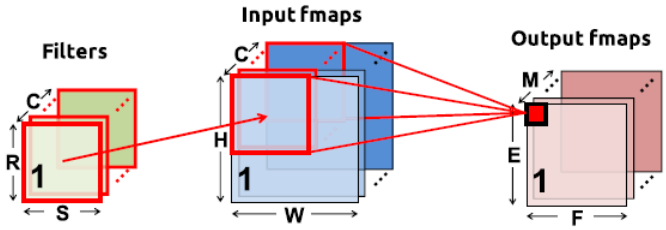

Figure 1. Visualized typical convolution operations.

TABLE I. PORT SPECIFICATIONS OF THE CONVOLUTION PROCESSOR

Name	I/O	Width
clk	I	1
rst_n	I	1
valid	I	1
inpur_addr	O	6
inpur_data	I	8
weight_addr	O	2
weight_data	I	8
WEN	O	1
output_addr	O	6
output_data	O	16
done	O	1

Table I shows the port definitions of our design. The data of ifmap and filter are stored in the external behavioral ROM. Then, the convolution results are written back to the output memory with a control signal, **WEN** (write enable). The input signal, **valid**, tells the design when it can start the operations, and the output signal, **done**, tells the testbench when all operations are done.

The patterns of the filter and ifmap are generated by random numbers. They are then transformed into two behavioral ROMs for both Verilog and Hspice simulation. In Verilog, we verify the result by storing the result to an output memory in advance. When the task is done, the testbench checks if all outputs are equal to theoretical results. Nonetheless, when it comes to Hspice simulation, a behavioral memory array is too computing-resource-consuming. To overcome this issue, we assume that all data of each output pixel are written once. When the design triggers **WEN**, the testbench finds the correct answer in the ROM of the ofmap instead of storing data. The schematic view is shown in Fig. 2. In the beginning, the D flip-flop of **ERR** resets to logic 0. When **WEN** is logic 1 and the answer is wrong,

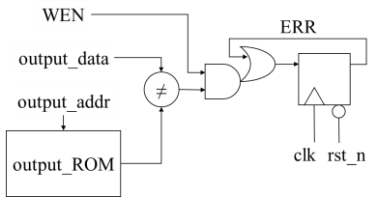

Figure 2. Schematic of output testbench in Hspice.

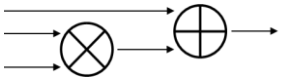

Figure 3. Schematic of one process element.

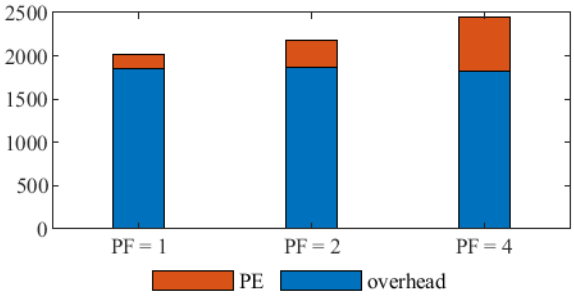

Figure 4. Area distribution of the designs with three different PFs.

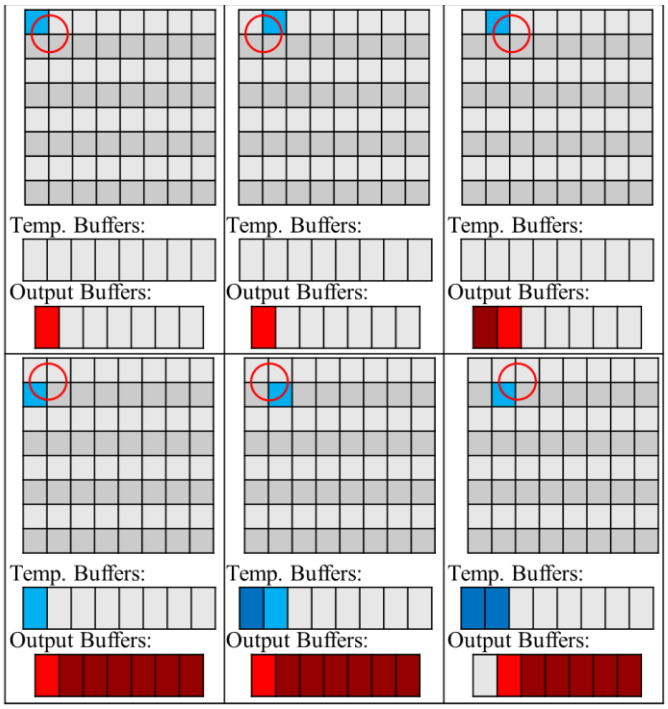

Figure 5. Data flow of PF = 1.

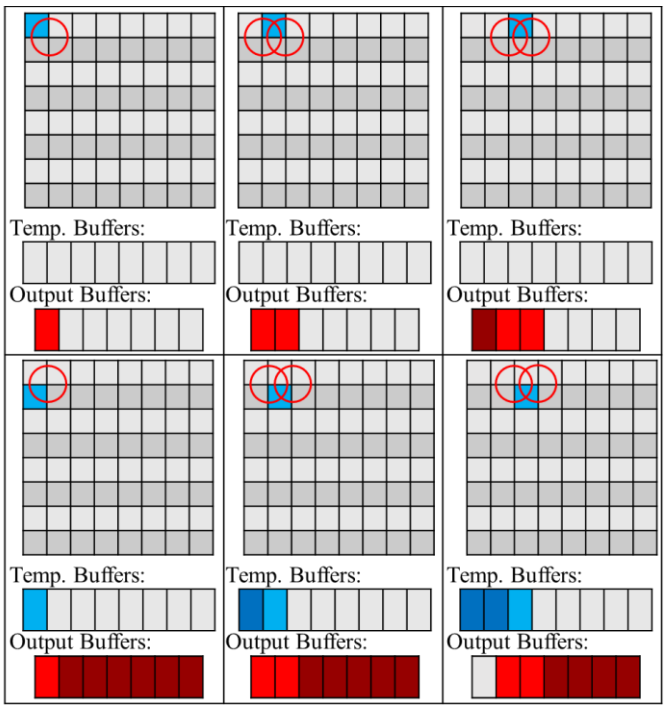

Figure 6. Data flow of PF = 2.

ERR becomes logic 1 at the next rising edge of the clock. On the contrary, if the answer is correct, **ERR** keeps its state.

Furthermore, we define the PF of our design as the number of process elements (PEs). Since the memory only supports one port of data access, the size of the filter (2×2) makes the maximum number of PEs working at the same time limited to 4. After synthesis in tsmcN28 technology, the three designs with PF = 1, 2, and 4 are converted to netlists and tested with different supply voltages and the corresponding minimal clock period via Hspice.

III. Design Architecture

Knowing that memory access is energy-consuming [4], there should be several stationary buffers to ensure that all pixels of both the ifmap data and the filter weights are accessed only once. Besides, each PE executes one MAC [Fig. 3] for the advantage of smaller area and flexibility. To manifest the impact on PF, all of the designs possess the same level of complexity (i.e. area and energy) in both control and stationary buffers. This is accomplished through reading the input data according to an increasing address sequence, and fixing the size of output stationary buffers to the width of the ofmap. Fig. 4 illustrates the synthesized areas of the three designs. After we subtract the area of the PEs (this is accomplished by individual synthesis), we observe that the areas of other overhead are almost the same.

Fig. 5–7 shows the algorithms of data access under different architectures. For PF = 1, when each input pixel, except for the leftmost column, is accessed, it should be calculated for two cycles due to two sharing convolution windows. To deal with cross-row conflict, additional temporary input stationary buffers are required. This case is similar when PF = 2, in which the two PEs operate the windows in the same row. However, for PF = 4, the simultaneous operating windows can extend to two rows, so we do not need input stationary buffers anymore. However, cross-row conflict still exists so that we use an additional temporary output stationary buffer instead.

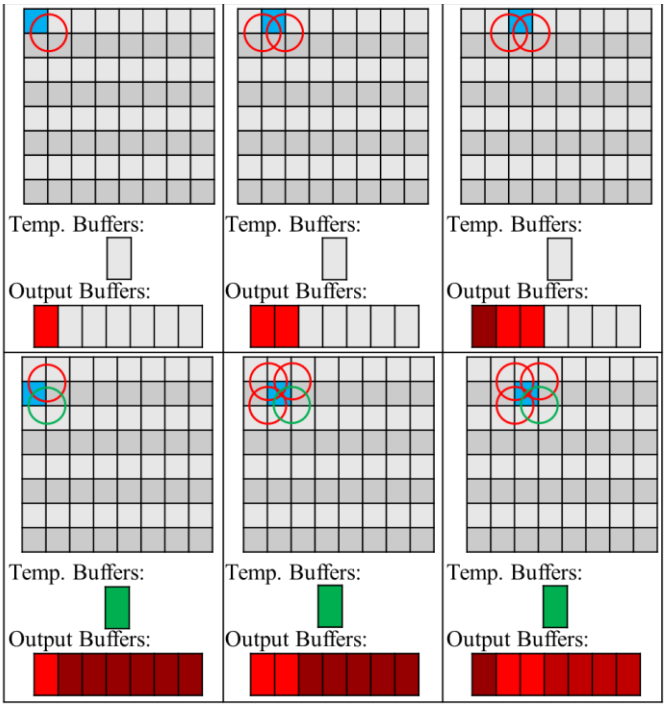

Figure 7. Data flow of PF = 4.

Except for marginal ones, all input pixels are used in four different windows, so activating four PEs, i.e., PF = 4 in the same clock cycle, should be the optimal architecture to our specification. Fig. 8 illustrates the utilities of each PE among the three designs. Since the marginal input pixels do not require four multiplication-accumulations, some PEs sacrifice their utility. However, the adequately high utility makes the design with PF = 4 earn fewer cycles to complete the task [Table II].

IV. Simulation Results and Discussions

Before the simulation of the convolution circuits, we first analyze the results of an inverter standard cell [Fig. 9]. The supply sweeps with 0.1-V increment. We find that the inverter still keeps the function when $V_{DD} \geq 0.2$ V, but the delay at V_{DD} = 0.3 V is about 60× longer than that at V_{DD} = 0.9 V. By individually operating point analysis of the transistor model, it is found that the threshold voltage is about 0.38 V. Since further lowering the supply voltage contributes to subthreshold conduction with exponential grow in delay, and hence potentially high static energy, we decide to analyze the convolution circuits of which $V_{DD} \geq 0.4$ V.

Fig. 10 illustrates the relation between delay and supply, as well as energy and supply. We regard PF = 1 and V_{DD} = 0.9 V as the reference design point. Because of parallelism, we can lower the supply to reach the same original throughput [5]. By applying alpha-power model [3], we find that α is approximately 1.2 in tsmcN28 technology. Since the design with PF = 2 can be 132/216 times slower [Table II], the expected V_{DD} is 0.72 V. Similarly, the design with PF = 4 is expected to operate at V_{DD} = 0.60 V. Compared to the measurement results (black line), the design with PF = 2 can operate at V_{DD} = 0.7 V, which earns 1.71× energy reduction, and that with PF = 4 can further operate

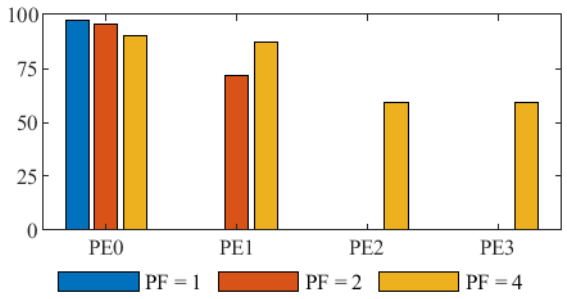

Figure 8. Utitlity of each process element under different PFs.

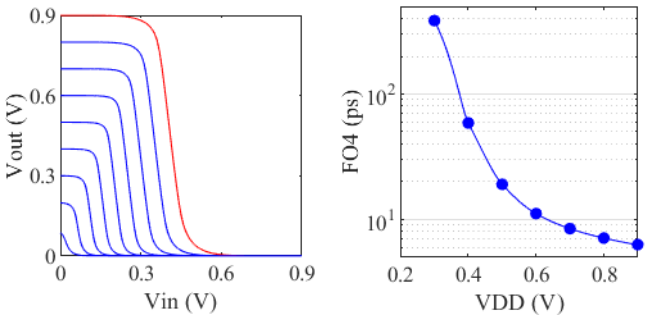

Figure 9. Characteristics of the standard inverter cell. (a) DC sweep. (b) FO4 delay.

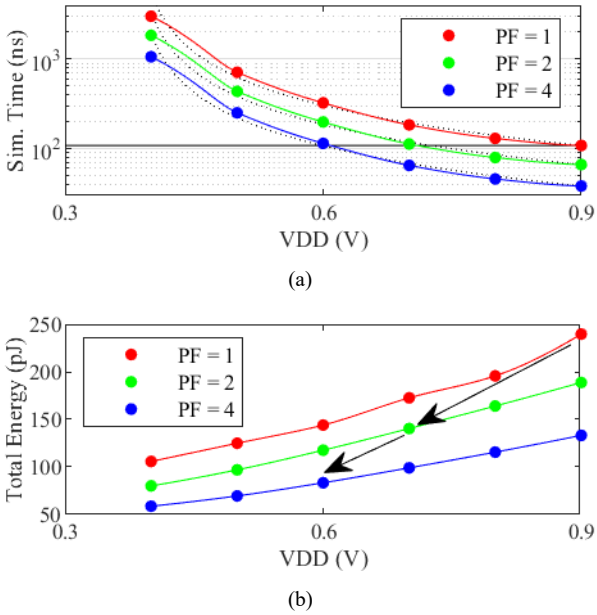

Figure 10. Delay and energy analysis by sweeping V_{DD}. (a) Delay-supply curve. (b) Energy-supply curve.

at V_{DD} = 0.6 V and earn 1.68× more. As a result, the total energy reduction based on the reference design can be 2.87×.

Fig. 11 illustrates the energy-delay curve of the designs. We observe that increasing PF makes the curve move lower left, meaning more optimal design point. In addition, the sensitivities

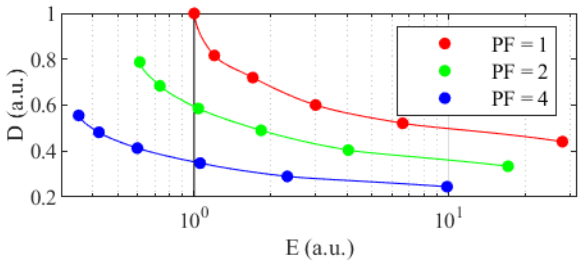

Figure 11. Energy-delay tradeoff of the convolution circuits.

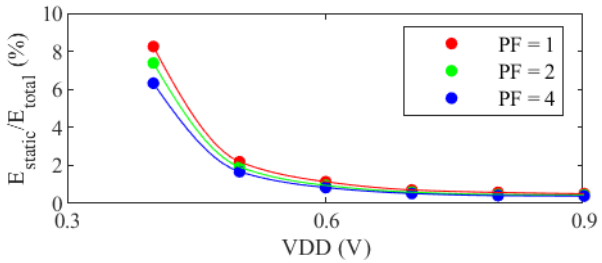

Figure 12. The portion of static energy in the convolution circuits.

of V_{DD} [3] to the circuits with PF = 1, 2, 4 equal to 1 when V_{DD} = 0.8 V, 0.78 V, 0.76 V, respectively. Fig. 12 shows the ratio of static energy to total energy. Although the area of PF = 4 is slightly larger [Fig. 4], lower clock cycles and adequate utility decrease the final ratio. Note that if we regard the synthesized netlists as a set of static CMOS logic gates, the remaining current has to do with leakage. In other words, increasing PF also improves leakage.

Table II summarizes the comparison metrics of different PF. One can find that PF = 4 performs the best in latency, energy efficiency, and area efficiency. From the reference design to the most energy-efficient design (PF = 4, V_{DD} = 0.4 V), we benefit from 4.13× energy reduction.

V. Conclusion

Convolution features in data reusability, which leads to two issues: (1) Stationary buffers are implemented to decrease repeating memory access. In addition to a row of output stationary buffers, to avoid row-crossing conflict, other temporary buffers are introduced to the design. (2) Designs with high PF are suitable for input data flow. We design each PE to support one MAC. The PF can be maximized according to the filter size since its size implies the number of MACs for one input pixel (except for marginal cases). Based on the balanced overhead among the three designs: PF = 1, 2, and 4, we obtain a 2.84× reduction in latency, and 2.87× reduction in energy if we view PF = 1, V_{DD} = 0.9 V as the reference point, and PF = 4 with the same throughput as the target design. Besides, the energy can be even lower if we operate the design with PF = 4 at V_{DD} = 0.4 V. As a result, we can earn totally 4.13× energy efficiency. By simulating these simple convolution circuits, we can verify that parallelism should also take utility into account, or there will be a tradeoff between area and energy.

TABLE II. Comparison Metrics of Architectures with Different Pallelism Factors

PF	**1**	**2**	**4**
Latency (cycles)	216	132	76
Freq (MHz)	2000 @ 0.9 V; 71.4 @ 0.4 V		
Area (μm^2)	2009	2180	2443
Energy (pJ)	240 @ 0.9 V 106 @ 0.4V	189 @ 0.9 V 80 @ 0.4V	133 @ 0.9 V 58 @ 0.4V
Nop (OP/cycle)	0.91	1.48	2.58
Energy Efficiency (GOPs/W)	817 @ 0.9 V 1849 @ 0.4 V	1037 @ 0.9 V 2450 @ 0.4 V	1474 @ 0.9 V 3379 @ 0.4 V
Area Efficiency ($GOPs/mm^2$)	906 @ 0.9 V 32.3 @ 0.4 V	1358 @ 0.9 V 48.5 @ 0.4 V	2112 @ 0.9 V 75.4 @ 0.4 V

Acknowledgment

The authors would like to thank Prof. Chia-Hsiang Yang for topic suggestions and Energy-Efficient Circuit and System Lab in National Taiwan University for their technical instructions.

References

[1] A. Biswas and A. P. Chandrakasan, “CONV-SRAM: An Energy-Efficient SRAM With In-Memory Dot-Product Computation for Low-Power Convolutional Neural Networks,” in *IEEE Journal of Solid-State Circuits*, vol. 54, no. 1, pp. 217–230, Jan. 2019.

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

[3] D. Markovic, V. Stojanovic, B. Nikolic, M. A. Horowitz and R. W. Brodersen, "Methods for true energy-performance optimization," in *IEEE Journal of Solid-State Circuits*, vol. 39, no. 8, pp. 1282–1293, Aug. 2004.

[4] Y. Chen, T. Krishna, J. S. Emer and V. Sze, "Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks," in *IEEE Journal of Solid-State Circuits*, vol. 52, no. 1, pp. 127–138, Jan. 2017.

[5] A. P. Chandrakasan and R. W. Brodersen, "Minimizing power consumption in digital CMOS circuits," in *Proceedings of the IEEE*, vol. 83, no. 4, pp. 498–523, April 1995.