DESIGN AND IMPLEMENTATION OF TWO-DIMENSIONAL CONVOLUTION ON PYNQ-Z2 FPGA DEVELOPMENT BOARD

Huynh Viet Thang
Danang University of Science and Technology – The University of Danang

ARTICLE INFO

Received: 22/11/2020

Revised: 25/12/2020

Published: 11/01/2021

KEYWORDS

FPGA
Hardware implementation image processing
2D convolution
PYNQ
Python

ABSTRACT

Two-dimensional (2D) convolution is a very important operation commonly used in the fields of image processing and convolution neural networks. In this paper, we designed and implemented a hardware module that performs two-dimensional convolution for use in high-speed image processing. The convolution module was developed using hardware description language VHDL, synthesized on Xilinx's PYNQ-Z2 development board, and packed into a hardware library for use in Python development environments for related applications. Evaluation results showed that using the designed two-dimensional convolution module could improve the performance of the convolution operation by a factor of up to 9 times compared with the performance of the software implementation. The design has shown its potential in implementing FPGA-based hardware designs for image processing, pattern recognition, and deep learning applications.

THIẾT KẾ VÀ THỰC THI TÍCH CHẬP HAI CHIỀU TRÊN BOARD PHÁT TRIỂN FPGA PYNQ-Z2

Huỳnh Việt Thắng
Trường Đại học Bách khoa – ĐH Đà Nẵng

THÔNG TIN BÀI BÁO

Ngày nhận bài: 22/11/2020

Ngày hoàn thiện: 25/12/2020

Ngày đăng: 11/01/2021

TỪ KHÓA

FPGA
Thiết kế phần cứng
Xử lý ảnh
Tích chập hai chiều
PYNQ
Python

TÓM TẮT

Tích chập hai chiều là phép toán rất quan trọng đang được sử dụng phổ biến trong các lĩnh vực xử lý hình ảnh và mạng nơ-ron tích chập. Trong bài báo này, chúng tôi thiết kế và thực thi một mô-đun phần cứng thực hiện tính tích chập hai chiều để ứng dụng trong xử lý hình ảnh tốc độ cao. Mô-đun tích chập hai chiều được phát triển bằng ngôn ngữ mô tả phần cứng VHDL, được tổng hợp trên board phát triển PYNQ-Z2 của hãng Xilinx, và được đóng gói thành thư viện phần cứng để sử dụng trong môi trường phát triển ứng dụng Python cho các ứng dụng liên quan. Các kết quả đánh giá thực tế trên phần cứng cho thấy, sử dụng mô-đun tích chập hai chiều được thiết kế giúp cải thiện tốc độ thực thi lên đến 9 lần so với thực thi bằng phần mềm, và vì vậy có tiềm năng ứng dụng trong triển khai các thiết kế phần cứng dựa trên FPGA cho các ứng dụng xử lý hình ảnh, nhận dạng mẫu và học sâu.

Email: thanghv@dut.udn.vn

1. Giới thiệu

Tích chập hai chiều (2D convolution) là một phép toán rất quan trọng được sử dụng phổ biến trong các lĩnh vực xử lý hình ảnh [1], [2], và gần đây đã được sử dụng rất phổ biến trong các mạng nơ-ron tích chập [3] trong các ứng dụng nhận dạng mẫu và trí tuệ nhân tạo. Các ứng dụng xử lý hình ảnh tốc độ cao thường đòi hỏi sử dụng các phần cứng chuyên dụng cho việc thực thi tích chập hai chiều nhằm đảm bảo các yêu cầu khắt khe về tốc độ cao, khả năng cấu hình linh hoạt, thời gian và chi phí thiết kế và thực thi ứng dụng ngắn. Trong các trường hợp ứng dụng này, mảng cổng lập trình được dạng trường FPGA (Field Programmable Gate Array) là một nền tảng phần cứng cho phép thực thi các module gia tốc phần cứng (hardware accelerator) với hiệu năng cao và chi phí thấp, rất phù hợp để thực thi các khối tính toán tích chập hai chiều trong các ứng dụng xử lý ảnh tốc độ cao.

Trong những năm gần đây, các thiết bị phần cứng FPGA mới đã được giới thiệu, một trong số đó là board phát triển PYNQ [4]. PYNQ hỗ trợ phát triển toàn bộ hệ thống trên chip SoC (System-on-Chip) với giao diện điều khiển giữa hệ thống phần mềm và phần cứng FPGA có thể được thực hiện toàn bộ trong môi trường phát triển dựa trên ngôn ngữ Python, vì vậy cho phép triển khai dễ dàng các ứng dụng xử lý ảnh cũng như trí tuệ nhân tạo hoàn toàn trong môi trường Python với độ linh hoạt cao.

Trong bài báo này, chúng tôi thiết kế và thực thi một mô-đun phần cứng thực hiện tính tích chập hai chiều trên board phát triển PYNQ-Z2 của hãng Xilinx để ứng dụng trong xử lý ảnh và trong thực thi mạng nơ-ron tích chập, biểu diễn như sau:

$$\mathbf{F}(m,n) = \sum_{i=0}^{S-1}\sum_{j=0}^{S-1} \mathbf{W}[i,j] \cdot \mathbf{I}[m-i, n-j] \quad (1)$$

Bài báo có những đóng góp khoa học sau:

- Thiết kế kiến trúc mô-đun tích chập hai chiều dựa trên ngôn ngữ mô tả phần cứng VHDL, thiết kế có thể tổng hợp được trên FPGA;
- Thực hiện đánh giá hiệu năng của mô-đun tính tích chập hai chiều trên board FPGA PYNQ-Z2 của Xilinx, việc đánh giá được thực hiện trong môi trường Python.

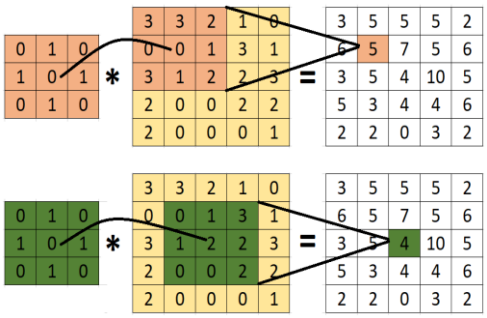
Bố cục của bài báo gồm các phần sau đây. Phần 2 giới thiệu cơ sở lý thuyết liên quan phép toán tích chập hai chiều và board phát triển PYNQ-Z2. Phần 3 trình bày thiết kế kiến trúc phần cứng mô-đun tích chập hai chiều. Các kết quả đánh giá thiết kế trên thiết bị Xilinx FPGA PYNQ-Z2 sẽ được trình bày chi tiết trong Phần 4. Trong phần 5, chúng tôi tóm tắt các kết quả đạt được và giới thiệu các hướng nghiên cứu dự định trong tương lai.

2. Cơ sở lý thuyết

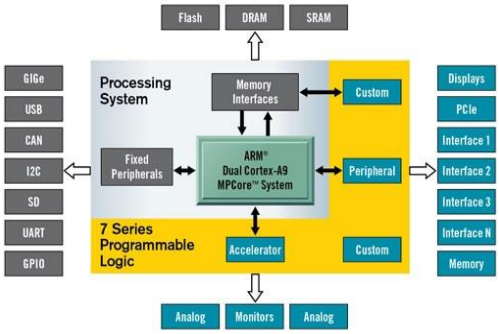
2.1.Tích chập hai chiều

Tích chập hai chiều (2D convolution) là một phép toán cơ sở trong xử lý ảnh, được sử dụng để thực hiện các phép toán lọc hai chiều. Ngày nay, với sự ra đời của các mạng nơ-ron tích chập, phép toán tích chập hai chiều là một thành phần quan trọng của các mạng nơ-ron tích chập, và đóng góp đến trên 90% khối lượng tính toán của các mô hình thực thi mạng nơ-ron tích chập.

Tích chập hai chiều được định nghĩa như sau. Cho ảnh đầu vào I kích thước MxN điểm ảnh và mặt nạ lọc W kích thước SxS phần tử, ảnh đầu ra F có cùng kích thước MxN là kết quả của phép toán tích chập hai chiều giữa ảnh đầu vào I và mặt nạ W được xác định theo công thức (1).



Hình 1. *Minh họa phép toán tích chập hai chiều*



Hình 2. *Sơ đồ khối board mạch PYNQ-Z2 [5]*

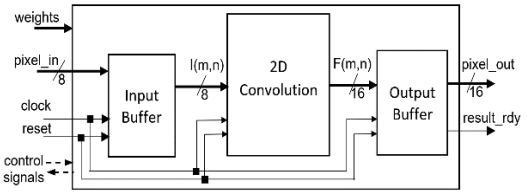
Hình 1 trình bày một ví dụ minh họa về tính tích chập hai chiều giữa một ảnh đầu vào kích thước 5x5 với một mặt nạ lọc 3x3. Để tính giá trị tích chập cho mỗi điểm ảnh, một cửa sổ trượt (*sliding window*) kích thước 3x3 được dùng để lựa chọn các điểm ảnh cần thiết và sau đó thực hiện phép toán nhân cộng tích lũy MAC (Multiply-Accumulate) giữa mặt nạ và các điểm ảnh tương ứng trong vùng cửa sổ trượt. Trong trường hợp tổng quát, đối với ảnh đầu vào kích thước MxN và mặt nạ lọc kích thước SxS thì số lượng phép toán MAC cần thực hiện là MxNxSxS phép toán.

2.2. Board phát triển PYNQ-Z2 FPGA

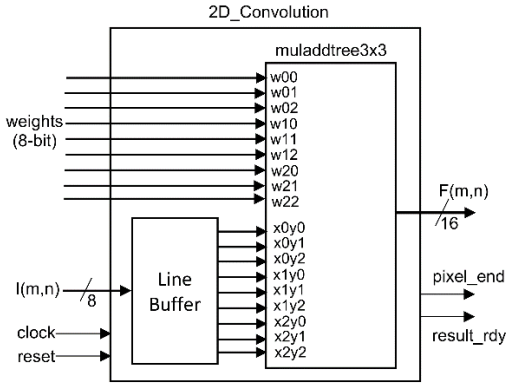
Chúng tôi sử dụng board phát triển FPGA PYNQ-Z2 [5] của hãng Xilinx để thiết kế và thực thi mô-đun tích chập hai chiều. Board phát triển PYNQ-Z2 là một sản phẩm trong dòng sản phẩm chuyên dụng cho phát triển hệ thống trên chip SoC của hãng Xilinx. Sơ đồ khối chức năng của PYNQ-Z2 được trình bày trong Hình 2. Hệ thống gồm có bộ vi xử lý ARM-Cortex A9 lõi kép kết hợp cùng cấu trúc FPGA có thể cấu hình được để thực hiện các mô-đun phần cứng. Bộ xử lý ARM Cortex-A9 lõi kép được gọi là hệ thống xử lý PS (*Processing System*), và cấu trúc FPGA được gọi là mạch logic khả trình PL (*Programmable Logic*). Các bộ tăng tốc phần cứng và các mô-đun phần cứng thường được thực hiện trong PL (cấu trúc FPGA) và có thể điều khiển từ hệ thống xử lý PS. Để thực hiện điều này, các thiết kế phần cứng trên FPGA ở PL được xem như là các thư viện phần cứng và được định nghĩa là các ***overlay*** [5]. Một ***overlay*** có thể được gọi thực thi như một hàm (thư viện phần cứng) từ chương trình chạy trong bộ xử lý ARM-Cortex A9 của khối PS để tăng tốc ứng dụng phần mềm cho một ứng dụng cụ thể. Một tính năng ưu việt của board phát triển PYNQ-Z2 là hỗ trợ các ứng dụng với giao diện Python cho phép các ***overlay*** trong khối PL được điều khiển hoàn toàn bằng chương trình Python đang chạy trong hệ thống xử lý PS, vì vậy giúp FPGA dễ dàng sử dụng hơn trong các ứng dụng thị giác máy tính và học máy ngày nay.

2.3. Thiết lập thử nghiệm

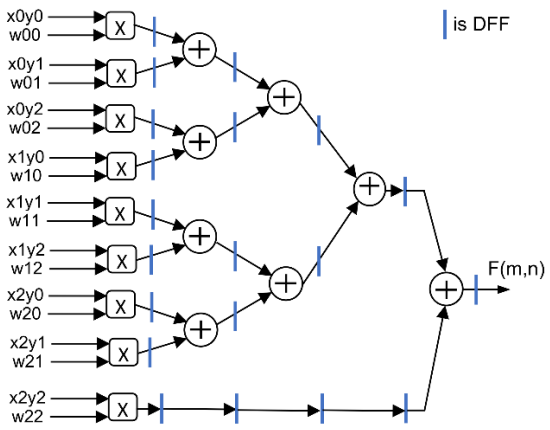
Trong nghiên cứu này, chúng tôi khảo sát ba cấu hình phần cứng khác nhau của mô-đun tích chập hai chiều tương ứng với các kích thước ảnh đầu vào là 32x32, 64x64 và 128x128 điểm ảnh. Mặt nạ lọc có kích thước cố định 3x3 phần tử. Chúng tôi tạo 3 overlay Python khác nhau tương ứng với các cấu hình phần cứng đã chọn của mô-đun được thiết kế. Phần mềm được sử dụng là Vivado Design Suite 2018.3 Webpack Edition của hãng Xilinx.



Hình 3. *Sơ đồ khối mô-đun tích chập hai chiều*



Hình 4. *Mô-đun con tích chập hai chiều*



Hình 5. *Kiến trúc khối muladdtree 3x3*

Định dạng số thực dấu phẩy tĩnh (fixed-point number) được lựa chọn để thực hiện các phép tính số học trong mô-đun tích chập hai chiều bởi định dạng số này có thể đảm bảo độ chính xác trong tính toán cũng như giúp tiết kiệm tài nguyên phần cứng [6]. Chúng tôi sử dụng bộ thư viện thực hiện phép toán dấu phẩy tĩnh (fixed-point number) bằng ngôn ngữ VHDL của David Bishop [7]. Định dạng số dấu phẩy động Q1.7 với các toán hạng có độ dài 8 bit được sử dụng trong bài báo này.

3. Thiết kế mô-đun tích chập hai chiều

Trong phần này, chúng tôi thiết kế và thực hiện mô-đun tích hợp hai chiều cho các FPGA của hãng Xilinx. Hình 3 trình bày sơ đồ khối chung của mô-đun được thiết kế, gồm ba mô-đun con: bộ đệm đầu vào (Input Buffer), tích chập hai chiều (2D Convolution) và bộ đệm đầu ra (Output Buffer).

3.1. Bộ đệm đầu vào và bộ đệm đầu ra

Bộ đệm đầu vào và bộ đệm đầu ra được thiết kế để hỗ trợ giao tiếp giữa mô-đun và hệ thống xử lý. Bộ đệm đầu vào được sử dụng để lưu trữ tất cả các điểm ảnh đến trước khi gửi chúng đến mô-đun tích chập. Tương tự, bộ đệm đầu ra sẽ lưu trữ tất cả các kết quả được tính toán và thông báo cho hệ thống xử lý bằng tín hiệu "result_rdy".

3.2. Tích chập hai chiều

Mô-đun con tích chập - trung tâm của thiết kế - bao gồm hai khối: một bộ đệm hàng (Line Buffer) và một khối muladdtree 3x3 thực hiện chức năng nhân cộng tích lũy MAC, như trình bày trong Hình 4.

Line Buffer đọc dòng hình ảnh đầu vào I(m, n) để trích xuất chín điểm ảnh lân cận cần thiết cho việc tính toán tích chập của mỗi điểm ảnh đầu vào. Chín điểm ảnh đầu ra từ Line Buffer được ký hiệu là $x_m y_n$ (trong đó m, n = 0, 1, 2) cho phép tính tích chập ở mỗi chu kỳ xung nhịp.

Khối muladdtree 3x3 (xem Hình 5) thực hiện phép nhân cộng tích lũy giữa các giá trị trọng số w_{mn} của mặt nạ và 9 điểm ảnh lân cận $x_m y_n$ từ Line Buffer, do đó cung cấp kết quả tính tích chập cho mỗi điểm ảnh đầu vào I(m,n) tại mỗi chu kỳ xung nhịp.

3.3. Tổng hợp phần cứng cho mô-đun tích chập hai chiều trên FPGA

Bảng 1 trình bày kết quả tổng hợp phần cứng của mô-đun tích chập hai chiều trên board FPGA PYNQ-Z2 tương ứng với các kích thước ảnh đầu vào là 32x32, 64x64 và 128x128 điểm ảnh.

Bảng 1. *Kết quả tổng hợp phần cứng mô-đun tích chập hai chiều trên board PYNQ-Z2*

Tài nguyên	Tổng số	Sử dụng		
		32x32	64x64	128x128
LUT	53200	1463	3517	11766
LUTRAM	17400	570	2218	8786
FF	106400	486	559	795

3.4. Đóng gói mô-đun thành Python Overlay

Sau khi tổng hợp mô-đun tích chập hai chiều, chúng tôi thực hiện việc đóng gói mô-đun đã thiết kế thành một Python Overlay để có thể sử dụng mô-đun này như một thư viện phần cứng trong môi trường Python cho các ứng dụng liên quan. Giao diện AXI (Advanced eXtensible Interface) được sử dụng để giao tiếp giữa mô-đun trên FPGA và bộ vi xử lý ARM-Cortex A9.

4. Kết quả và bàn luận

Trong mục này, chúng tôi trình bày kết quả đánh giá hiệu năng mô-đun tích chập hai chiều được thiết kế và thực thi trên board phát triển PYNQ-Z2.

Hiệu năng mô-đun được đánh giá thông qua hai thông số: tốc độ đỉnh trên lý thuyết và tốc độ hoạt động thực tế của mô-đun. Tốc độ đỉnh (*peak performance*) có thể được xác định thông qua mô phỏng hoạt động của mô-đun bằng phần mềm Vivado với giả thiết việc truyền dữ liệu giữa mô-đun và bộ nhớ ngoài của hệ thống vi xử lý hoàn toàn không có trễ. Trong khi đó, tốc độ hoạt động (*sustained performance*) của mô-đun cung cấp một số đo thực tế hơn về giá trị hiệu năng cho toàn hệ thống vì có tính đến thời gian trễ do truyền dữ liệu giữa mô-đun và bộ nhớ ngoài.

Bảng 2 trình bày tốc độ đỉnh của mô-đun được thiết kế. Mô-đun được thiết kế kết nối hoàn toàn theo cấu trúc đường ống (fully-pipeline) giữa các mô-đun con bên trong nên mô-đun có thể cung cấp kết quả tính toán tích chập của mỗi điểm ảnh tại mỗi chu kỳ máy. Số chu kỳ cần thiết để tính tích chập cho một ảnh đầu vào đối với ba cấu hình mô-đun lần lượt là 1034, 4106 và 16394 chu kỳ xung clock, tương ứng với các cấu hình 32x32, 64x64 và 128x128 điểm ảnh. Có thể thấy cả ba cấu hình đều có cùng số chu kỳ trễ (overhead latency) là 10 chu kỳ xung clock. Tốc độ khung hình tối đa ở tần số xung nhịp 100 MHz lần lượt là 96759, 24358 và 6100 ảnh/giây cho các kích thước ảnh đầu vào lần lượt là 32x32, 64x64 và 128x128 điểm ảnh.

Bảng 2. *Đánh giá tốc độ đỉnh (peak performance) của mô-đun tích chập hai chiều được thiết kế*

Thông số	Cấu hình		
	32x32	64x64	128x128
Số điểm ảnh	1024	4096	16384
Số chu kỳ thực thi (chu kỳ)	1034	4106	16394
Thời gian thực thi tại tần số 100MHz (micro-giây)	10.34	41.06	163.94
Tốc độ khung hình tối đa mỗi giây tại tần số 100MHz (ảnh/giây)	96759	24358	6100

Bảng 3 so sánh thời gian thực thi tích chập khi sử dụng mô-đun phần cứng trên FPGA với phiên bản thực thi tích chập bằng phần mềm chạy trên bộ xử lý ARM-Cortex A9 trên cùng một board phát triển PYNQ-Z2. Có thể thấy, tốc độ hoạt động thực tế của mô-đun thấp hơn tốc độ đỉnh lý tưởng, sự suy giảm hiệu năng này là do trễ truyền dữ liệu giữa mô-đun và bộ nhớ ngoài. Kết quả thử nghiệm ở Bảng 3 cho thấy tốc độ thực thi phép toán tích chập hai chiều bằng mô-đun phần cứng được cải thiện đáng kể so với tốc độ thực thi bằng phần mềm, với các hệ số tăng tốc lần lượt là 7,8 lần, 8,6 lần và 9,0 lần tương ứng cho ba cấu hình các kích thước ảnh đầu vào.

Bảng 3. *So sánh thời gian thực thi tích chập dùng mô-đun phần cứng với dùng phần mềm*

Thông số	Cấu hình		
	32x32	64x64	128x128
Thời gian thực thi trên mô-đun phần cứng (giây)	0,033	0,124	0,487
Thời gian thực thi trên phần mềm (giây)	0,260	1,061	4,364
Độ tăng tốc (lần)	7,8	8,6	9,0

5. Kết luận

Trong bài báo này, chúng tôi đã trình bày quá trình thiết kế, thực thi và đánh giá mô-đun tính tích chập hai chiều trên board phát triển FPGA PYNQ-Z2 của hãng Xilinx. Ba cấu hình khác nhau đã được phát triển và đóng gói thành các Python overlay tương ứng với kích thước hình ảnh đầu vào là 32x32, 64x64 và 128x128. Các overlay này có thể được gọi thực thi như một hàm từ chương trình Python chạy trên board phát triển PYNQ-Z2. Các kết quả thực thi trên phần cứng cho thấy sử dụng mô-đun được thiết kế cải thiện tốc độ thực thi của phép toán tích chập đến 9 lần so với thực thi bằng phần mềm. Qua nghiên cứu này, chúng tôi tin rằng mô-đun phần cứng được thiết kế sẽ có tiềm năng ứng dụng trong triển khai các thiết kế phần cứng dựa trên FPGA cho các ứng dụng xử lý ảnh và học sâu.

Để nâng cao hiệu quả của ứng dụng tính tích chập trên board phát triển PYNQ-Z2, giao diện truyền thông giữa mô-đun và hệ thống xử lý dựa trên ARM cần được cải thiện bằng cách sử dụng giao diện AXI-Stream với điều khiển truy cập bộ nhớ trực tiếp. Một hướng nghiên cứu khác là sử dụng mô-đun tích chập hai chiều để thực thi mạng nơ-ron tích chập trên nền tảng PYNQ. Những hướng nghiên cứu này sẽ được dành cho tương lai.

Lời cảm ơn

Nghiên cứu này được tài trợ bởi Quỹ Phát triển khoa học và công nghệ Đại học Đà Nẵng trong đề tài có mã số B2019-DN02-61.

TÀI LIỆU THAM KHẢO/ REFERENCES

[1] C. Ben, *Implementation of 2D Convolution on FPGA, GPU and CPU*, Imperial College Report, 2006.

[2] P. Stefania, M. Lanuzza, P. Corsonello, and G. Cocorullo, "A high-performance fully reconfigurable FPGA-based 2D convolution processor," *Microprocessors and Microsystems*, vol. 29, no. 8-9, pp. 381-391, 2005

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

[4] TUL Technology Unlimited, "TUL PYNQ™-Z2 board". [Online]. Available: http://www.tul.com.tw/ProductsPYNQ-Z2.html. [Accessed Nov. 21, 2020].

[5] PYNQ Overlay Tutorials. [Online]. Available: https://pynq.readthedocs.io/en/v2.5.1/pynq_overlays.html. [Accessed Nov. 21, 2020].

[6] Bishop, and W. David, "VHDL-2008 support library", 2011. [Online]. Available: https://github.com/FPHDL/fphdl. [Accessed Nov. 21, 2020].

[7] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, "Deep learning with limited numerical precision," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning*, 2015, vol. 37 (ICML'15). JMLR.org, pp. 1737-1746.