TẠP CHÍ KHOA HỌC & CÔNG NGHỆ ĐẠI HỌC DUY TÂN
DTU Journal of Science and Technology 02(33) (2019) 28-35

TRƯỜNG ĐẠI HỌC DUY TÂN
DUYTAN UNIVERSITY

Thiết kế bộ lọc Gauss trên nền FPGA dùng trong hệ thống xử lý ảnh thời gian thực

Designing 2D Gaussian Filter based on Field Programmable Gate Array (FPGA) used in realtime image-processing system

Lương Hà Quế Yên[a], Ngô Lê Minh Tâm[a], Hà Đắc Bình[a], Nguyễn Trọng Tuấn[b]
Que Yen Luong Ha, Minh Tam Ngo Le, Dac Binh Ha, Trong Tuan Nguyen

[a] *Khoa Điện - Điện tử, Đại học Duy Tân, 03 Quang Trung, Đà Nẵng, Việt Nam*
Faculty of Electrical & Electronics Engineering, Duy Tan University, 03 Quang Trung, Danang, Vietnam
[b] *Công ty Global CyberSoft VietNam, Đà Nẵng, Việt Nam*
Global CyberSoft (Vietnam) JSC, Da Nang, Viet Nam

(Ngày nhận bài: 13/03/2019, ngày phản biện xong: 23/03/2019, ngày chấp nhận đăng: 01/04/2019)

Tóm tắt

Lọc ảnh là một trong những kỹ thuật tiền xử lý quan trọng trong xử lý ảnh và thị giác máy tính, được sử dụng để loại bỏ các chi tiết (không mong muốn) và nhiễu từ một hình ảnh. Bài báo này trình bày phương pháp thiết kế bộ lọc Gauss 2D trên nền FPGA nhằm đáp ứng yêu cầu tài nguyên, tốc độ xử lý thời gian thực. Kiến trúc bộ lọc Gauss sẽ được mô tả hoàn toàn bằng ngôn ngữ mô tả phần cứng Verilog theo chuẩn giao tiếp AXI-Stream Bus tốc độ cao; khả năng mở rộng ma trận và tăng năng lực tính toán phép tích chập được độc lập trong từng mô-đun. Kết quả thực nghiệm được mô phỏng trên công cụ ModelSim và được tính toán tổng hợp bằng trình ISE 14.7- Xilinx cho thấy thông số tài nguyên của chúng tôi sử dụng tốt hơn của tác giả trước, đồng thời ảnh lọc (.txt) được so khớp kết quả giống như chạy trên OpenCV.

Từ khóa: Bộ lọc Gauss, Verilog, FPGA, AXI-Stream, Xilinx.

Abstract

Image filtering is one of the very useful techniques in image processing and computer vision, used to eliminate useless details and noise from an image. This paper *presents methods of designing* 2D Gaussian Filter based on Field Programmable Gate Array (FPGA) to meet the requirements of hardware resources and speed of real-time processor. The Gaussian filter architecture will be described by Verilog HDL completely through a high speed AXI-Stream Bus; thus, extension matrix and operations of convolution are separated in modules. The experimental results simulated by ModelSim tool and logically synthesized by ISE 14.7 tool of Xilinx show that our performance is better than previous author's finding, concurrently, the filtered image (.txt file) is matched to the result run on OpenCV.

Keywords: Gaussian Filter, Verilog, FPGA, AXI-Stream, Xilinx.

1. Giới thiệu

Ngày nay, ứng dụng của thị giác máy đã trở nên rất rộng lớn và đa dạng, len lỏi vào mọi lĩnh vực từ quân sự, khoa học, vũ trụ, cho đến y học, sản xuất và tự động hóa tòa nhà. Để giải các bài toán xử lý bằng hình ảnh, cũng cần nhiều giải pháp khác nhau. Từ phần cứng cho tới các công cụ hỗ trợ phần mềm.

Lọc nhiễu là một công đoạn tiền xử lý trong xử lý ảnh số, nhằm nâng cao chất lượng ảnh cho

Email: luonghaqueyen@gmail.com

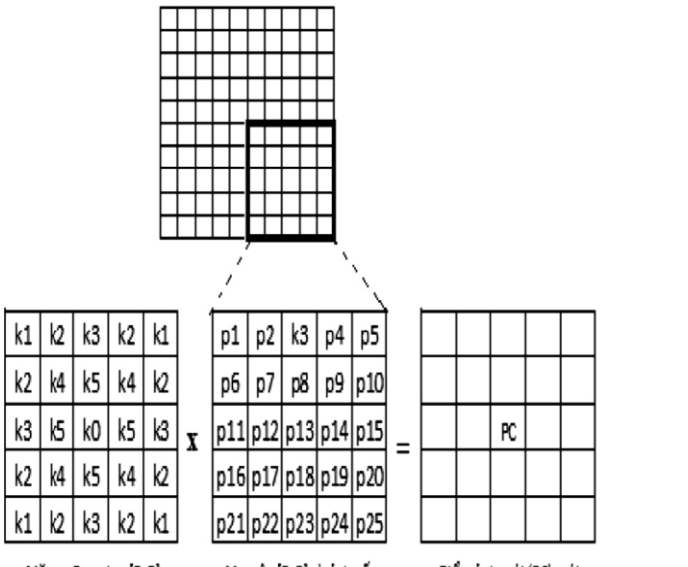
mắt con người hoặc để phục vụ cho các công đoạn sau, xử lý tốt công đoạn này sẽ giúp cho các công đoạn sau tiến hành được dễ dàng hơn [9] và bộ lọc làm mờ ảnh được sử dụng trong nhiều ứng dụng như nhận dạng đối tượng, đối chiếu, phân loại [1, 5, 10] là bước tiền xử lý để giảm nhiễu và mức độ chi tiết (không mong muốn) của hình ảnh [14, 4]. Bài báo này tập trung thiết kế bộ lọc nhiễu theo Gauss.

Bộ lọc Gauss là một trong những thuật toán lọc ảnh quan trọng nhất và tính ứng dụng rộng rãi trong xử lý ảnh [2, 7, 8]. Phương trình hàm Gauss (G) dùng trong không gian hai chiều được xác định bởi công thức (1).

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \qquad (1)$$

Trong đó G là mặt nạ Gauss tại tọa độ điểm (x, y) theo hai trục ngang và đứng, σ là giá trị quyết định độ lệch chuẩn giữa các điểm trên bề mặt Gauss. Nếu giá trị σ lớn, thì tác dụng làm mờ ảnh sẽ càng tăng.

Trong toán học, việc ứng dụng Gauss Blur cho một hình cũng chính là tính tích chập hình đó với hàm Gauss (G). Nghĩa là trong không gian hai chiều, hàm (G) sẽ tạo ra những đường viền là những đường tròn đồng tâm - mặt nạ Gauss, tuân theo logic phân tán Gauss từ điểm trung tâm. Giá trị từ mặt nạ Gauss này sẽ được sử dụng để xây dựng một ma trận kernel dùng tính toán phép tích chập với hình ảnh gốc có kích thước [h x w] như được mô tả ở phương trình (2). Kết quả điểm ảnh mới (PC) thu được bằng cách tính tổng các cặp phần tử tương ứng từ ảnh gốc tích với ma trận Gauss kích thước [5x5]. Minh họa tích chập ma trận Gauss [5x5] và tổ hợp ảnh đầu vào [5x5] ở Hình 1.



$$f(x,y) = \sum_{i=0}^{h-1} \sum_{j=0}^{w-1} G(i,j) I(x-i, y-j) \qquad (2)$$

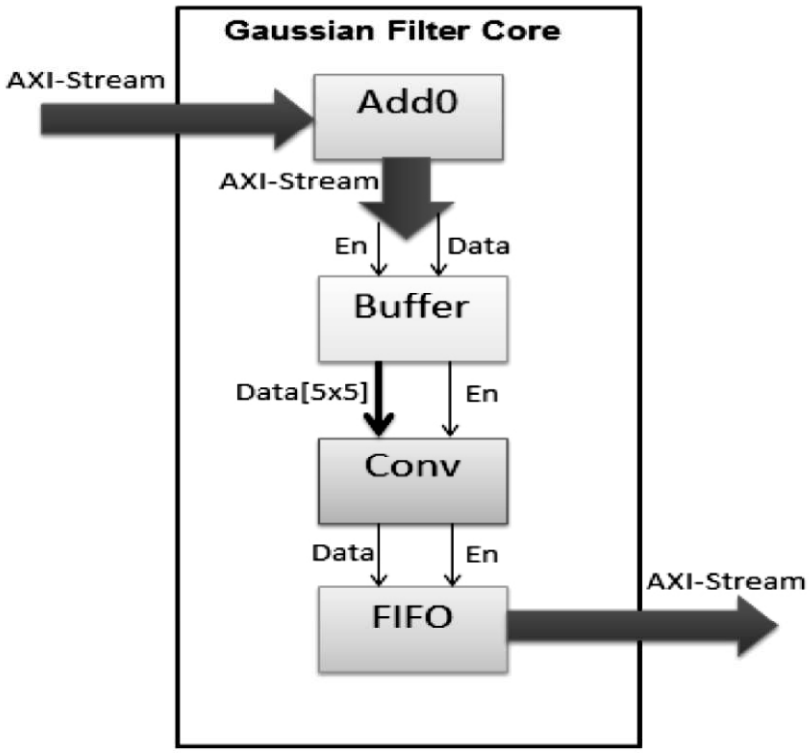
Hình 1. Minh họa tích chập hai ma trận

Tận dụng đặc tính của FPGA là linh hoạt và tốc độ xử lý nhanh, các nhà khoa học trên thế giới đã đưa ra nhiều kỹ thuật được áp dụng cho bộ lọc Gauss [11, 12]. Lọc ảnh là bước tiền xử lý để loại

bỏ các chi tiết (không mong muốn) và nhiễu khỏi ảnh. Ảnh lọc được tạo ra bởi sự tích chập giữa ảnh gốc và một mặt nạ Gauss; tích chập được sử dụng rộng rãi trong thị giác máy và xử lý hình ảnh, bao gồm nhận dạng đối tượng và cạnh hình ảnh [13]; tuy nhiên, quy trình tích chập thường chiếm một lượng tài nguyên máy tính đáng kể [14]. Do đó, [3, 6] đề xuất phương pháp tích chập dựa trên kiến trúc phần cứng FPGA/ASIC thực thi chức năng lọc Gauss cho ứng dụng xử lý ảnh giúp tiết kiệm bộ nhớ; [1] đã thực hiện các thao tác làm mịn Gauss bằng cách kết hợp module hỗ trợ trong các phần mềm MATLAB, Xilinx System Gennerator (XSG) để tạo ra file HDL thực thi phần cứng trên kit Spartan-3E tài nguyên thấp, vẫn đảm bảo quá trình phân tích và tốc độ tính toán theo thời gian thực; [15] phân tách các phép tính toán Gauss thành các module riêng được thực hiện trên nền tảng VirtexV ML605 FPGA, đồng thời [2] xây dựng mô hình bộ lọc dấu phẩy động với hiệu suất tăng.

Mục tiêu của bài báo này hướng đến thiết kế bộ lọc nhiễu Gauss được thực thi trên nền tảng Mảng cổng có khả năng lập trình (FPGA). Chúng tôi sẽ phân tích các chức năng trong thuật toán thành các mô-đun thực thi riêng biệt. Do đó, để tăng tốc độ tính toán và giảm thiểu sử dụng tài nguyên, các tính năng của FPGA sẽ được sử dụng là cơ chế DMA (truy cập bộ nhớ trực tiếp) và DDR3. Cuối cùng, để lõi bộ lọc Gauss hoạt động thì một file user_logic bọc các mô-đun chức năng bên trong để đồng bộ toàn bộ kiến trúc với Bus AXI-Stream của hệ thống. Đây là một cách mới để mô tả lọc nhiễu Gauss hoàn toàn bằng phần cứng. Ứng dụng của chúng tôi được triển khai bởi hai công cụ như ISE và Verilog và được mô phỏng trên trình giả lập ModelSim.

2. Phân tích và thiết kế lõi bộ lọc Gauss



Hình 2. Kiến trúc Top-Down bên trong lõi bộ lọc Gauss

Thiết kế lõi bộ lọc Gauss dựa trên hai bước chính là mở rộng kích thước ảnh gốc và thực hiện tích chập với ma trận Gauss; vấn đề đồng bộ trong từng giai đoạn hoạt động, các tín hiệu truyền nhận đúng thời điểm xác lập được thực hiện theo cơ chế DMA đảm bảo yêu cầu tốc độ cao thì cần thiết kế thêm các khối đệm và khối FIFO để đảm bảo dữ liệu vào/ra được liên tục và không bị mất các pixel của ảnh.

Kiến trúc lõi bộ lọc Gauss ở hình 2 mô tả sơ đồ khối các mô-đun chức năng, việc thiết kế tuân thủ theo quy trình Top - Down. Thiết kế từ mức cao nhất, tổng quát nhất đến mức kế tiếp theo và đến mức nhỏ nhất. Việc phân cấp cấu trúc nhỏ dễ dàng kiểm tra, sửa đổi hoặc phát triển sau này:

Khối Add0: Khối này thực hiện việc mở rộng kích thước ảnh gốc, thêm các bit '0' xung quanh khung ảnh; nhằm bảo đảm kích thước ảnh được giữ nguyên sau khi thực hiện tích chập.

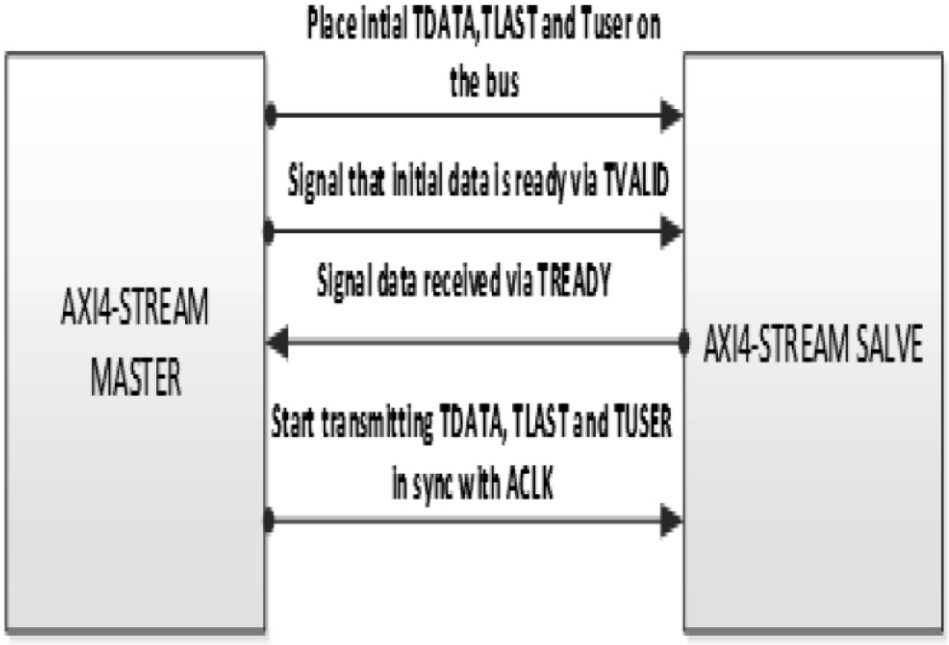
Khối Buffer: Trong module này sẽ sử dụng những thanh ghi để chốt dữ liệu ra và đưa lại đầu vào phù hợp cho khối sau.

Khối Conv: Thực hiện phép tích chập giữa ma trận ảnh [5x5] từ khối Buffer với ma trận Gauss Blur [5x5]. Mặt nạ Gauss Blur được tính xấp xĩ như mô tả dưới đây:

Khối FIFO: Khối bộ nhớ, lưu dữ liệu sau lọc nhiễu. Vùng nhớ FIFO này làm gia tăng sự chủ động giữa PC và hệ thống.

Khối user_logic: Lõi bộ lọc Gauss qua user_ logic được xem là một thiết bị ngoại vi mở rộng, ghép nối với block DMA. Đường bus kết nối giữa DMA và bộ lọc Gauss là đường bus AXI4-Stream (Hình 3).

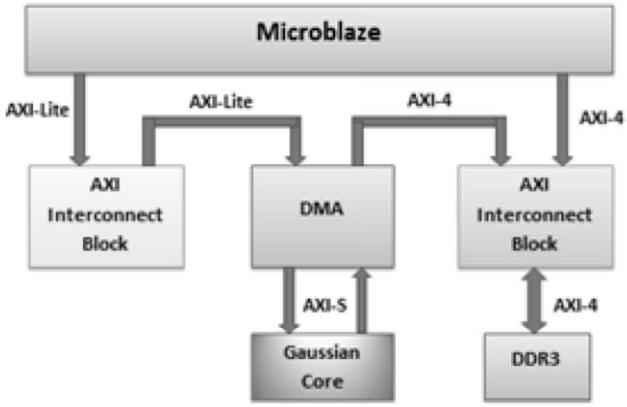
Hệ thống lõi bộ lọc Gauss được nhúng trên bộ Vi xử lý Microblare. Việc ghép nối giữa Microblaze với các thiết bị ngoại vi thông qua Bus AXI-4. Đường bus kết nối giữa DMA và Lõi Gauss là đường bus AXI-Stream (Hình 3), với 3 tín hiệu chính là AXI_DATA, AXI_VALID, AXI_READY, dữ liệu sẽ được truyền trong AXI_DATA, khi có dữ liệu thì tín hiệu AXI_ VALID sẽ được bật lên 1, và nếu nhận dữ liệu thì AXI_READY sẽ được bật lên 1, nếu không dữ liệu vẫn chưa được ghi vào Lõi chương trình.



Hình 3. Kết nối chế độ Master Salve theo đường bus AXI4-Stream

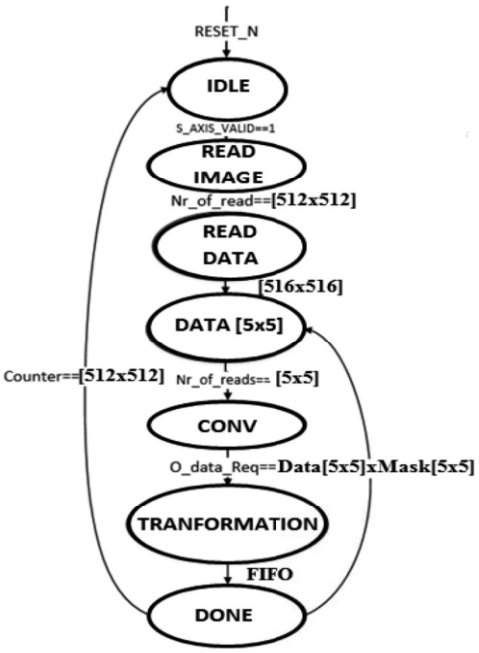
Để thực hiện hoàn chỉnh mô hình hoạt động bộ lọc Gauss trên một hệ thống nhúng (Hình 4), ngoài bộ vi xử lý đóng vai trò điều phối mọi hoạt động, cần có các thiết bị ngoại vi hỗ trợ, các thiết bị ngoại vi hỗ trợ tối thiểu bao gồm: bộ nhớ Bram là nơi lưu trữ các trình điều khiển hệ thống,vi xử lý Microblaze sẽ truy cập các mã lệnh trong Bram để điều khiển hoạt động của hệ thống. Bộ nhớ DDRAM lưu dữ liệu ảnh gốc và ảnh sau khi thực hiện lọc nhiễu. Quá trình hoạt động theo cơ chế DMA – truyền nhận trực tiếp, nhiệm vụ của khối DMA sẽ vừa nhận dữ liệu, vừa thực thiện quá trình

điều khiển hoạt động của Lõi và đẩy tín hiệu vào DDR3. Truyền nhận dữ liệu tốc độ cao, độ rộng dữ liệu theo Bus AXI-Stream từ 16-1024 bit.



Hình 4. Sơ đồ kết nối Lõi Gauss vào hệ thống nhúng FPGA DMA Bus

Sơ đồ FSM mô tả việc giao tiếp giữa 4 khối (Hình 5). Trạng thái IDLE nhằm thiết lập các thông số khởi tạo cho Lõi, chờ tín hiệu S_VALID báo có dữ liệu, đọc đầy đủ các dữ liệu ảnh vào Lõi, sau khi đệm các bit 0 bên ngoài ảnh tăng kích thước dữ liệu đầu vào; trạng thái CONV là lọc các tổ hợp ảnh [5x5] thực hiện phép tích chập với mặt nạ Gauss Blur [5x5]; kết quả thu được là các điểm ảnh pixels sau khi lọc được lưu vào khối FIFO chờ đầy đủ dữ liệu như kích thước ảnh gốc để tiếp tục nhận frame mới tại IDLE.



Hình 5. FSM của Lõi Gauss

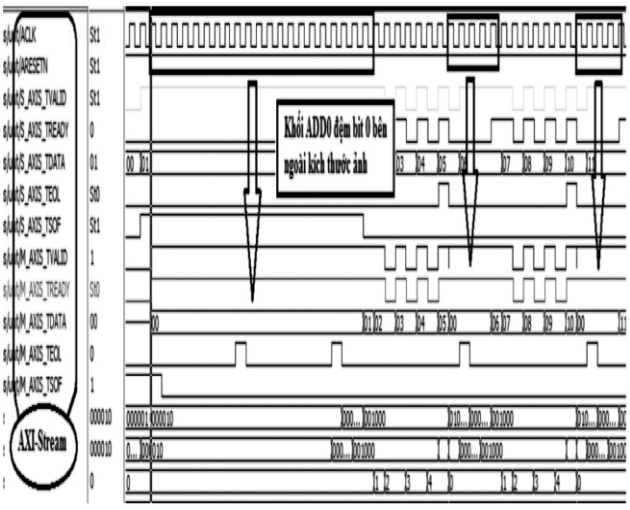
Tóm lại, hệ thống Lõi Gauss tốc độ cao được thực hiện trên công nghệ FPGA theo kiến trúc đường ống (pipeline), nhưng cải tiến Lõi thêm khối ADD0 để đảm bảo ảnh đầu ra đúng kích thước ảnh gốc. Đồng thời triển khai về mặt hệ thống theo cơ chế DMA truyền nhận theo Bus AXI-Stream.

3. Phân tích và đánh giá kết quả

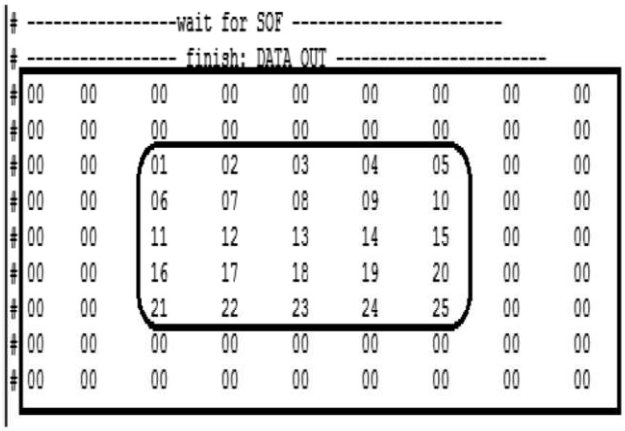
3.1. Phân tích kết quả thực hiện

Các hình dưới đây sẽ tiến hành mô phỏng cho từng khối chức năng thực hiện bộ lọc nhiễu Gauss. Khối ADD0 (Hình 6) thực hiện xuất dãy bit '0' làm đường viền ảnh trước khi nhận và xuất dữ liệu ảnh gốc theo chuẩn AXI-Stream. Dữ liệu ảnh gốc và ảnh sau khi thực hiện mỗi kKhối được lưu vào file. Hình 7 hiển thị kết quả dữ liệu đầu ra của khối ADD0 được đọc từ file.

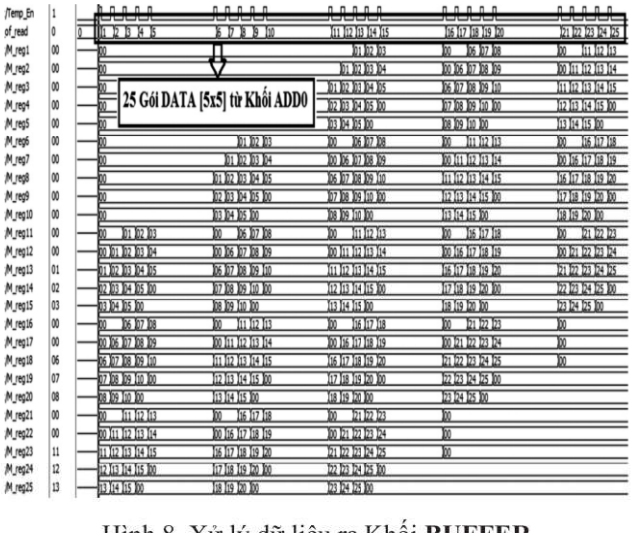
Khối BUFFER mô tả tiến trình đọc ảnh đầu vào từ khối ADD0: lưu dữ liệu vào các thanh ghi, kết hợp giữa S_READY và S_VALID để trạng thái READ DATA [5x5] thực hiện phân luồng đúng theo thứ tự từng cửa sổ ảnh. Hình 8 mô tả 25 lần cửa sổ lệnh [5x5] được đọc lần lượt theo ảnh từ khối ADD0.



Hình 6. Xử lý dữ liệu vào khi **ADD0** thực hiện

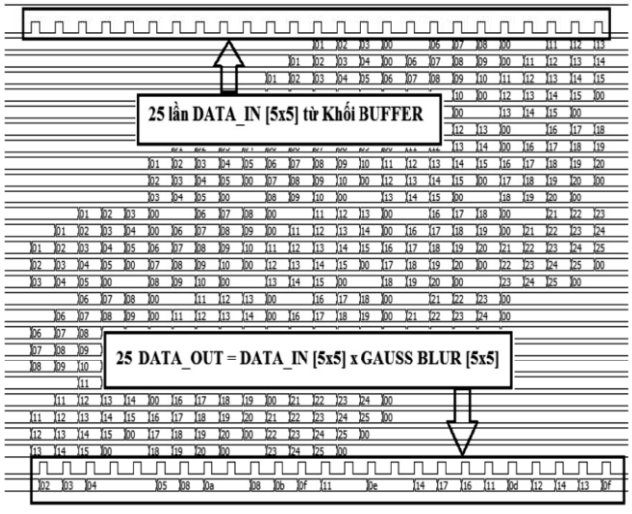


Hình 7. Kết quả đầu ra của **Khối ADD0** trên ModelSim.



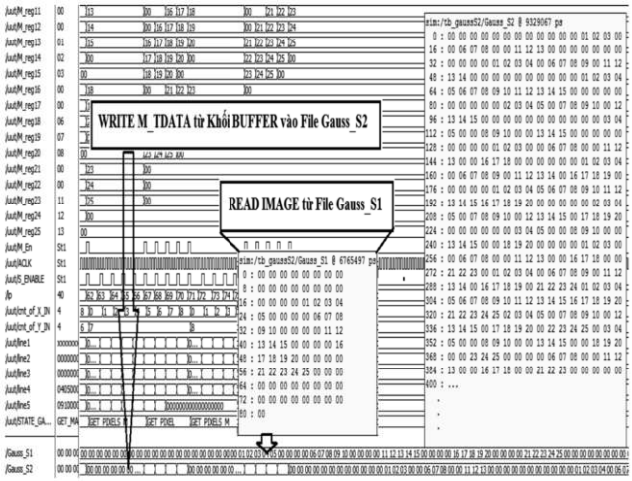
Hình 8. Xử lý dữ liệu ra Khối **BUFFER**

Khối CONV đọc dữ liệu từ READ DATA [5x5] thực hiện tích chập với mặt nạ Gauss Blur [5x5] để tạo ra dữ liệu đầu ra là pixels sau khi lọc nhiễu Gauss (H**ình** 9).

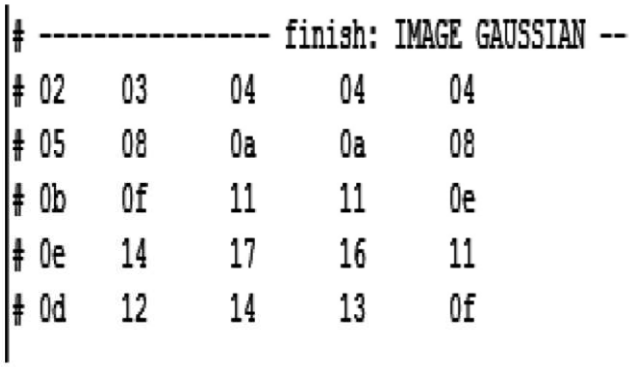


Hình 9. Khối **CONV** thực hiện tích chập hai ma trận

Khối FIFO nhận các pixels sau khi lọc nhiễu, ghi dữ liệu vào bộ nhớ FIFO thực hiện biến đổi theo chuẩn AXI-Stream. Dữ liệu vào ra các Khối ADD0, BUFFER và CONV lần lượt l**à các file Gauss_S1, Gauss_S2 (**Hình 10) và Gauss_S3 (Hình 11).



Hình 10. Kết quả xử lý dữ liệu ở Gauss_S1 và Gauss_S2



Hình 11. Kết quả xử lý dữ liệu Gauss_S3

3.2. Đánh giá kết quả

Tổng hợp là kết quả quá trình biên dịch hoặc chuyển đổi ngôn ngữ mô tả phần cứng Verilog vào trong mức cổng. Căn cứ vào cấu trúc và quy mô phần cứng được mô tả trong chương trình Verilog, trình tổng hợp ISE 14.7 sẽ cho các kết quả sau khi tổng hợp. Các thông số quan trọng đánh giá kết quả quá trình tổng hợp được so sánh với một số tác giả khác ở Bảng 1 và tần số cực đại của Lõi (Hình 12).

```
Speed Grade: -3

   Minimum period: 2.747ns (Maximum Frequency: 364.000MHz)
   Minimum input arrival time before clock: 3.571ns
   Maximum output required time after clock: 5.905ns
   Maximum combinational path delay: 5.749ns

=========================================================================

Process "Synthesize - XST" completed successfully
```

Hình 12. Tần số cực đại từ ISE 14.7

Bảng 1. Tài nguyên phần cứng được tổng hợp trên trình ISE

	Tác giả	[16]	[6]	[2]
Kit	ML605	Virtex5	XC7C	ML605
Registers	210	127	-	-
LUTs	53	176	4750	8985
FF	42	-	-	8464
Delay[ns]	5,75	-	4,7	-
Tần số [Mhz]	364	-	100	200

Được kiểm chứng từ trình biên dịch ISE-Xilinx, với [16] có số lượng thanh ghi kết quả 127 và trong bài báo này cần 210 cho Lõi Gauss, nhưng khi so sánh LUTs với [16] là 176 kết quả của bài báo này là giảm hơn 50% là 53% cho phép tích chập. Kết quả độ trễ cực đại (Delay) 5,75 ns nhiều hơn so với [6] 4,7 ns nhưng với mục tiêu ban đầu nhằm hiệu quả về tài nguyên phần cứng và xét việc ứng dụng trong hệ thống thiết kế nhỏ gọn, giá thành KIT thấp đạt yêu cầu. Lõi Gauss tốc độ cao và linh hoạt khi tần số cực đại Clock của hệ thống cung cấp cho Lõi hoạt động là 364 MHz khi áp dụng cơ chế truyền nhận liên tục DMA.

4. Kết luận

Bài báo này đã trình bày thiết kế lõi bộ lọc Gauss tiết kiệm tài nguyên, tốc độ cao khi Lõi kết nối theo Bus DMA với các thông số tổng hợp đạt được. Các thông số thiết kế IPCore được tính toán dựa trên trình tổng hợp ISE-Xilinx kết hợp với phần mềm mô phỏng ModelSim-Mentor Graphics. Kết quả thực nghiệm cho thấy phương pháp thiết kế của chúng tôi đã làm tăng tính hiệu quả và ứng dụng so với kết quả nghiên cứu [2] mà chúng tôi tham khảo.

Tài liệu tham khảo

[1] G. Bharadwaja Reddy and K. Anusudha. Implementation of Image Edge Detection on FPGA using XSG. 2016 International Conference on Circuit, Power and Computing Technologies (ICCPCT), pp. 1 – 5.

[2] Cuong Pham-Quoc and Tran Ngoc Thinh. An Efficient Runtime Adaptable Floating-Point Gaussian Filtering Core. 2017 4th NAFOSTED Conference on Information and Computer Science (NCICS). pp. 183 – 188.

[3] Gian Domenico Licciardo, Carmine Cappetta, Luigi Di Benedetto. FPGA Optimization of Convolution-based 2D Filtering Processor for Image Processing. 2016 8th Computer Science and Electronic Engineering Conference (CEEC). pp. 180 – 185.

[4] Tejashwini N C and Karthik P. Design and Implementation of Adaptive Gaussian Filters for the Removal of Salt and Pepper Noise on FPGA. 2016 International Conference on Electrical, Electronics, Communication, Computer and Optimization Techniques (ICEECCOT). pp. 53 – 59.

[5] Dionis A. Padilla, Joseph Aaron B. Adriano, Jessie R. Balbin, Ivan G. Matala, Jan Julien R. Nicolas and Steven Rey R. Villadelgado. Implementation of Eye Gaze Tracking Technique on FPGA-based On-screen Keyboard System Using Verilog and MATLAB. Proc. of the 2017 IEEE Region 10 Conference (TENCON), pp. 2771 – 2776.

[6] Carmine Cappetta, Gian Domenico Licciardo and Luigi Di Benedetto. Hardware Architecture for 2D Gaussian Filtering of HD Images on Resource Constrained Platforms. 2017 International Symposium on Signals, Circuits and Systems (ISSCS), pp. 1 – 4.

[7] Manu K.S, Dr. Rekha K.R and Dr . Nataraj K.R. FPGA Implementation of Image Block Generation and Color Space Conversion for the Gaussian Mixture Model. 2017 International Conference on Recent Advances in Electronics and Communication Technology (ICRAECT). pp. 24 – 28.

[8] Geoffrey Brooks. Gaussian-based Filters for Elementary Motion Detector Delay Element: Modeling Spatio-Temporal Information Pathways with Elementary Motion Detection. 2018 IEEE Research and Applications of Photonics In Defense Conference (RAPID). pp. 1 – 3.

[9] Manoj Kumar Jakkula, A.Sai Pavan and Gautam Narayan. Optimization of Image Filtering Architectures for Real Time Systems. 2018 International Conference on Communication and Signal Processing (ICCSP). pp. 1000 – 1004.

[10] Ke Liu, Keming Long, Baozhen Ma and Jing Yang. Gaussian Noise Filtering Using Pulse-Coupled Neural Networks. 2018 IEEE 3rd International Conference on Image, Vision and Computing (ICIVC). pp. 807 – 811.

[11] Vatsala Anand and Amanpreet Kaur. Implementation of energy efficient FIR Gaussian filter on FPGA. 2017 4th International Conference on Signal Processing, Computing and Control (ISPCC), pp. 431 – 435.

[12] Zhanyu Meng, Wei Zou and Liang Shan. The implementation of a fast Gaussian noise generator based on FPGA. 2017 Chinese Automation Congress (CAC). pp. 7487 – 7492.

[13] L. Kabbai, M. Abdellaoui and A. Douik. New robust descriptor for image matching. 2016 Journal of Theoretical and Applied Information Technology, 87(3), pp. 451 – 460.

[14] L. Rao, B. Zhang and J. Zhao. Hardware Implementation of Reconfigurable 1D Convolution, 2016 Journal of Signal Processing Systems, 82(1), pp. 1 – 16.

[15] F.Talbi, F.Alim, S. Seddiki, I. Mezzah and B. Hachemi, Separable Convolution Gaussian Smoothing Filters on a Xilinx FPGA platform. 2015 International conference on innovative computing technology (INTECH), pp.112 – 117.

[16] Leila kabbai, Anissa Sghaier, Ali Douik, and Mohsen Machhout, FPGA implementation of filtered image using 2D Gaussian filter. 2016 International Journal of Advanced Computer Science and Applications (IJACSA), pp.514 – 520.