**REPORT**

**Thiết kế và thực hiện mạch tính Pooling 2-D dùng cho mạng nơ-ron nhân chập CNN**

Ver 1.0

20/10/2022

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Full name** | **Function** | **Date** |
| Written by |  |  |  |
| Verified by |  |  |  |
| Approved by |  |  |  |

|  |
| --- |
| **Abstract (from 5 to 10 lines)** |
|  |

|  |
| --- |
| **Keywords** |
|  |

|  |
| --- |
| **Work context** |
|  |

Document History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Time** | **Revised by** | **Description** |
| V0.1 | 20/10/2022 | Nguyễn Kiêm Hùng | Problem definition |
| V0.2 | 13/12/2022 | Phạm Văn Hệ-18020468  Nguyễn Diệp Yến-18021455 | 1. Version |
|  |  |  |  |

MỤC LỤC

[Document History 3](#_Toc117778198)

[MỤC LỤC 4](#_Toc117778199)

[1. Giới thiệu 5](#_Toc117778200)

[2. Yêu cầu 5](#_Toc117778201)

[2.1. Yêu cầu đối với thiết kế: 5](#_Toc117778202)

[2.2. Định nghĩa giao diện vào/ra 6](#_Toc117778203)

[3. Thuật toán 7](#_Toc117778204)

[4. Thiết kế mức RTL 7](#_Toc117778205)

[4.1. Mô hình máy FSMD 7](#_Toc117778206)

[4.2. Đơn vị xử lý dữ liệu (Datapath) 7](#_Toc117778207)

[4.3. Đơn vị điều khiển (Control Unit) 7](#_Toc117778208)

[4.4. Sơ đồ khối tổng thể 7](#_Toc117778209)

[5. Mô hình hóa bằng VHDL 8](#_Toc117778210)

[6. Mô phỏng/thực thi và đánh giá 8](#_Toc117778211)

[7. Kết luận 8](#_Toc117778212)

[Appendix A: 9](#_Toc117778213)

[Appendix B: VHDL Code 10](#_Toc117778214)

[(đóng gói thành tệp nén và gửi kèm báo cáo) 10](#_Toc117778215)

[Appendix C: 11](#_Toc117778216)

[List of Figures 12](#_Toc117778217)

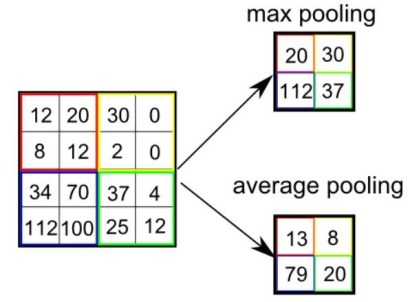
[List of Tables 13](#_Toc117778218)

[References 14](#_Toc117778219)

# Giới thiệu

*(Introduction to the motivation, Objectives, and main Contents of the project)*

**Mục tiêu:** Vận dụng các kiến thức, kỹ năng đã được học đểthiết kế, mô phỏng và thực thi một mô-đun phần cứng thực hiện tính Pooling J = MaxPool2D(I, size, stride) hoặc J = AvgPool2D(I, size, stride) cho hình ảnh lối vào I với kích thước pooling size = (2,2), và bước nhảy stride = 2, padding = 0. Trong đó, mỗi pixel trong hình ảnh đầu ra J đại diện cho giá trị lớn nhất (Max) hoặc giá trị trung bình (Average) của một ma trận có kích thước size được trích xuất từ ma trận đầu vào I. Phép tính max pooling và average pooling được minh họa bằng ví dụ trong Hình 1. Trong đó hình ảnh lối vào I có kích thước 4×4, size = (2,2), stride = 2, padding = 0.



Hình 1‑1. Ví dụ minh họa phép tính Pooling 2D.

# Yêu cầu

## Yêu cầu đối với thiết kế:

* SV chọn một trong hai loại pooling để thực hiện thiết kế phần cứng
* Hình ảnh đầu vào I có kích thước 32×32, size = (2,2), stride = 2 (padding = 0).
* Khối Pool2D có giao diện ghép nối tới CPU sao cho CPU kích hoạt quá trình tính toán của khối Pool2D bằng các đặt tín hiệu Start = ‘1’.
* Sau khi quá trình tính hình ảnh tích phân hoàn thành, khối Pool2D sẽ báo cho CPU biết bằng cách đặt tín hiệu Done = ‘1’;
* Khối Pool2D có 1 giao diện ghép nối tới tới bộ nhớ để đọc hình ảnh đầu vào
* Khối Pool2D có 1 giao diện ghép nối tới tới bộ nhớ để ghi dữ liệu đầu ra



Hình 2‑1. Giao diện ghép nối I/O.

## Định nghĩa giao diện vào/ra

Bảng 1: Mô tả các tín hiệu vào ra.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TT** | **Port** | **Direction** | **Width** | **Meaning** |
| 1 | Start | In | 1 | Tín hiệu bắt đầu quá trình tính toán |
| 2 | CLK | IN | 1 | Tín hiệu xung clock hệ thống |
| 3 | RAddr | IN | N | Địa chỉ bộ nhớ ma trận |
| 4 | REn | OUT | 1 | Tín hiệu đọc từ bộ nhớ |
| 5 | WAddr | OUT | N | Địa chỉ bộ nhớ ghi ma trận |
| 6 | Data\_in | IN | N | Dữ liệu ma trận lấy từ bộ nhớ |
| 7 | Data\_out | OUT | N | Dữ liệu ma trận ghi vào bộ nhớ |

# Thuật toán

Do thuật toán maxpooling xử lí trên dữ liệu 2 chiều sẽ làm phức tạp trong quá trình thiết kế mạch RTL. Vì vậy để tốt ưu, thuật toán nhân ma trận sẽ được xử lý dạng mảng 1 chiều. Dưới đây là lưu đồ thuật toán :

Mã giả (Pseudo code) mô tả thuật toán.

[initialize input data]

Beginning: wait for start = ‘1’

Done =’0’

Index=0

For m=0 to rowA – stride;m=m+2 do

For n= 0 to colA – stride;n=n+2 do

C[index]=0

For i=0 to stride do

For j = 0 to stride do

If(C[index]<A[(i+m)\*rowA+(j+n)]

C[index] = A[(i+m)\*rowA+(j+n)]

End for

End for

Index=index+1

End for

End for

Done=’1’

[write output data]

Wait for start =’0’

Go to beginning

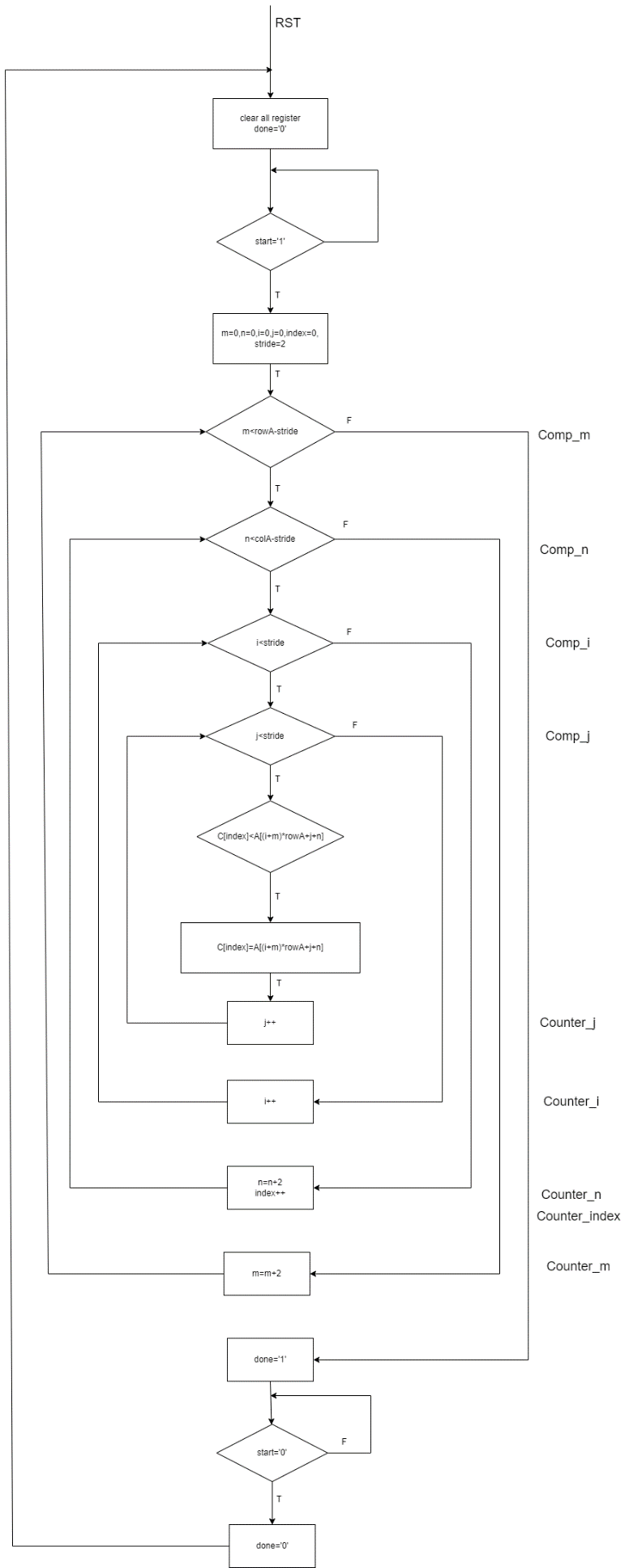
End.

Các phần tử của ma trận A được đưa về ma trận 1 chiều từ đó cho qua 4 vòng lặp for với m chạy trên ma trận cột, n chạy trên ma trận hàng, i j chạy mục đích để tạo một kernel trượt trên ma trận A và lấy giá trị lớn nhất trên ma trận kernel đó sau đó được lưu vào giá trị mảng C[index] với index là phần tử của mảng C và sẽ được tăng khi n thay đổi.

# Thiết kế mức RTL

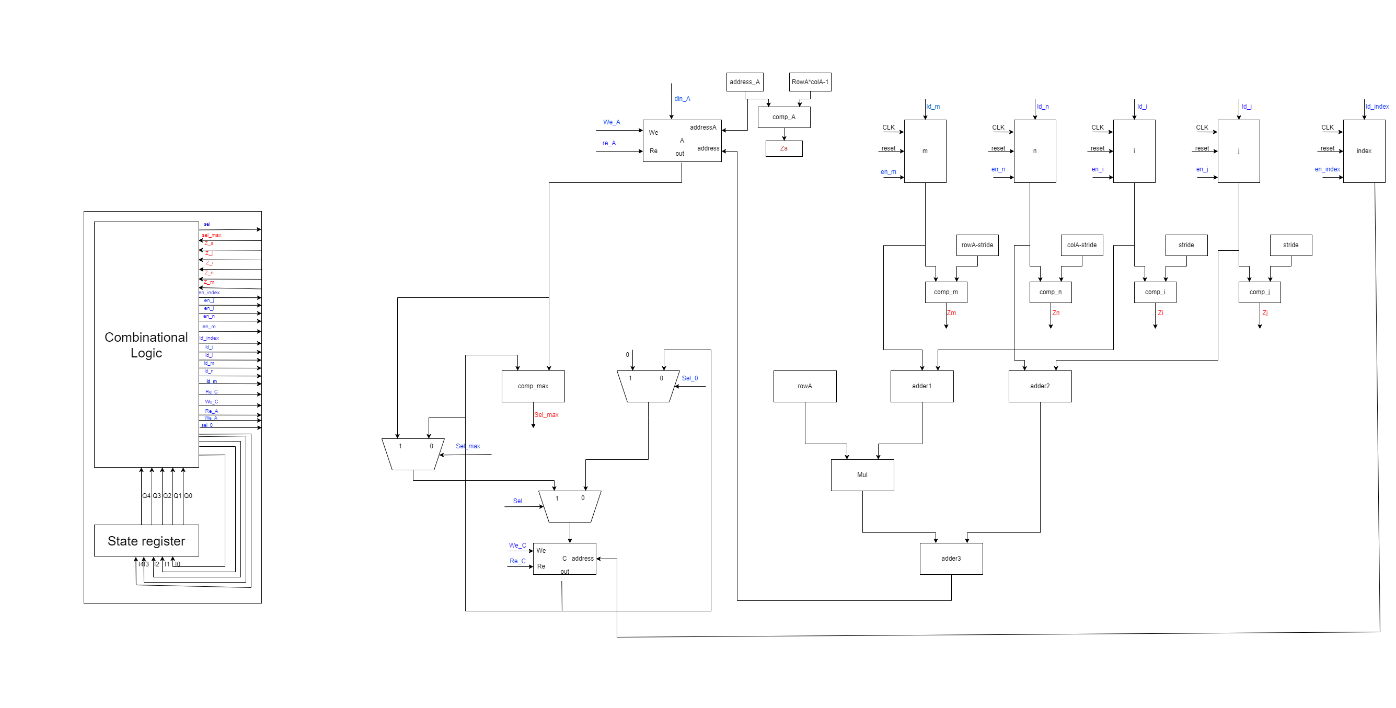
*Gợi ý : Tham khảo Lecture 3*

## Mô hình máy FSMD



Hình 4‑1: Mô hình máy FSMD.

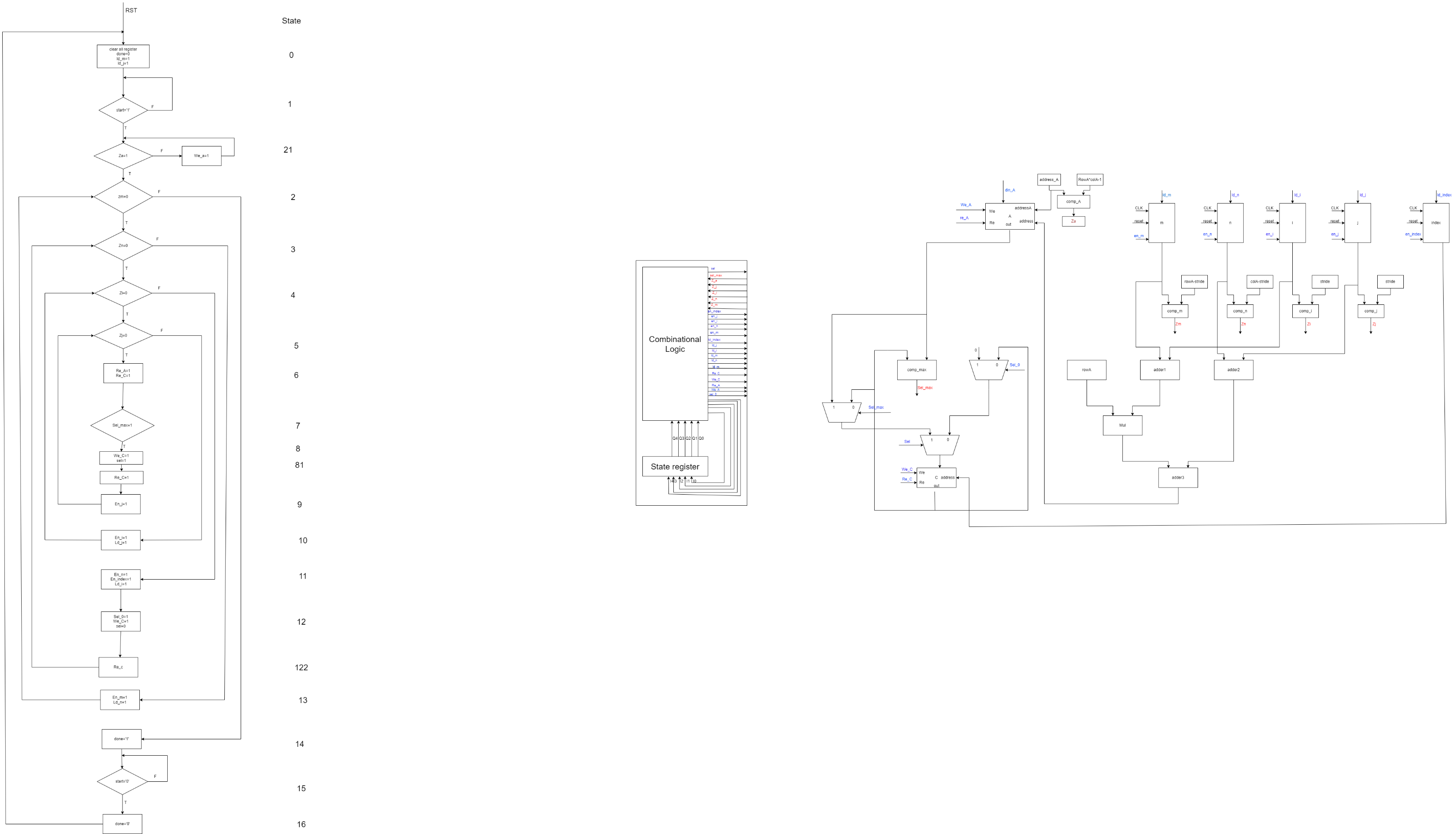
## Đơn vị xử lý dữ liệu (Datapath)



Hình 4‑2 Cấu trúc đơn vị xử lý dữ liệu Datapath

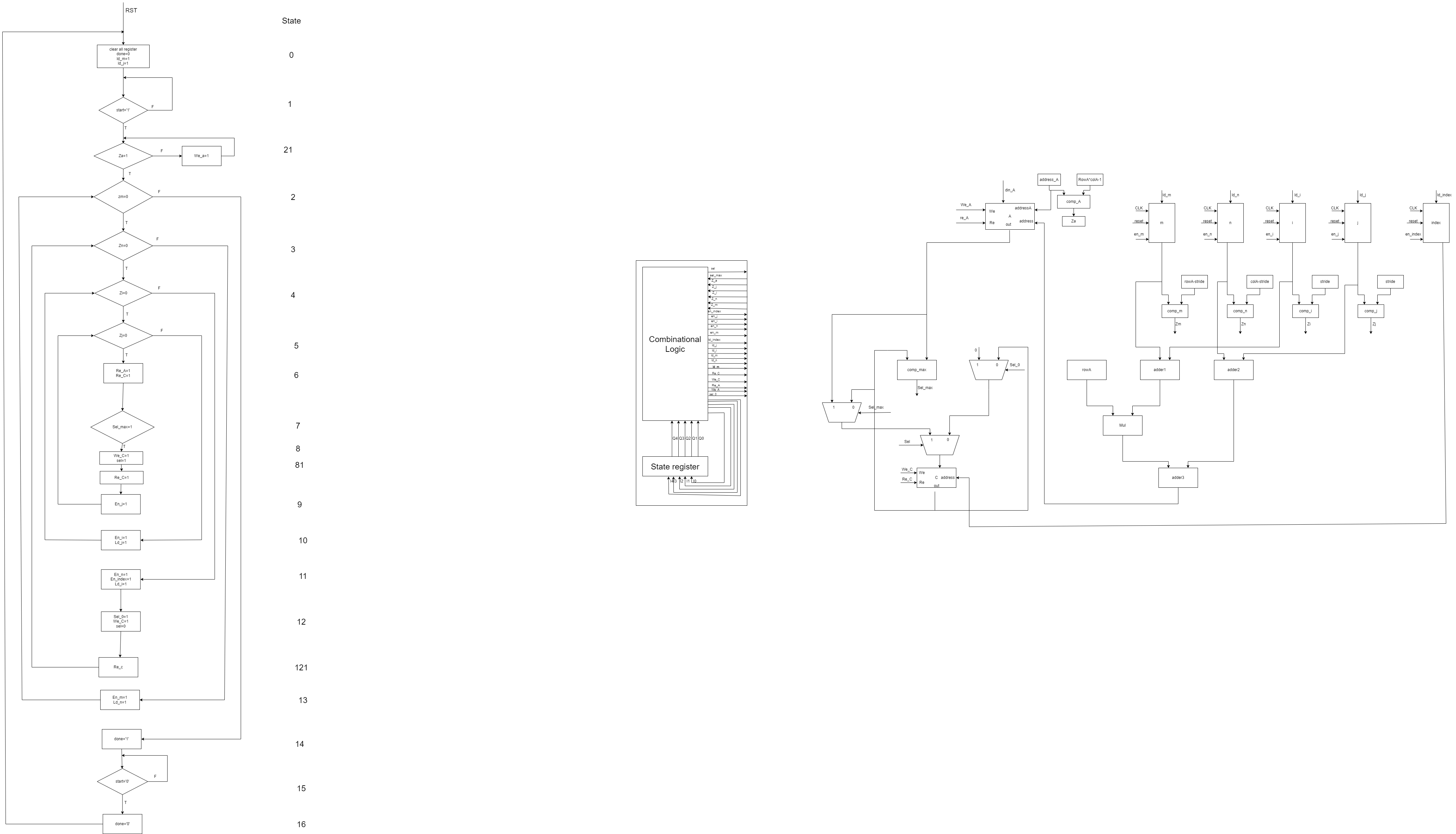
## Đơn vị điều khiển (Control Unit)

Sinh viên/Học viên chỉ ra sơ đồ máy trạng thái FSM của bộ điều khiển controller của thiết kế ở đây!



Hình 4‑3: Máy FSM của đơn vị điều khiển.

## Sơ đồ khối tổng thể



Hình 4‑4: Sơ đồ khối tổng thể của thiết kế.

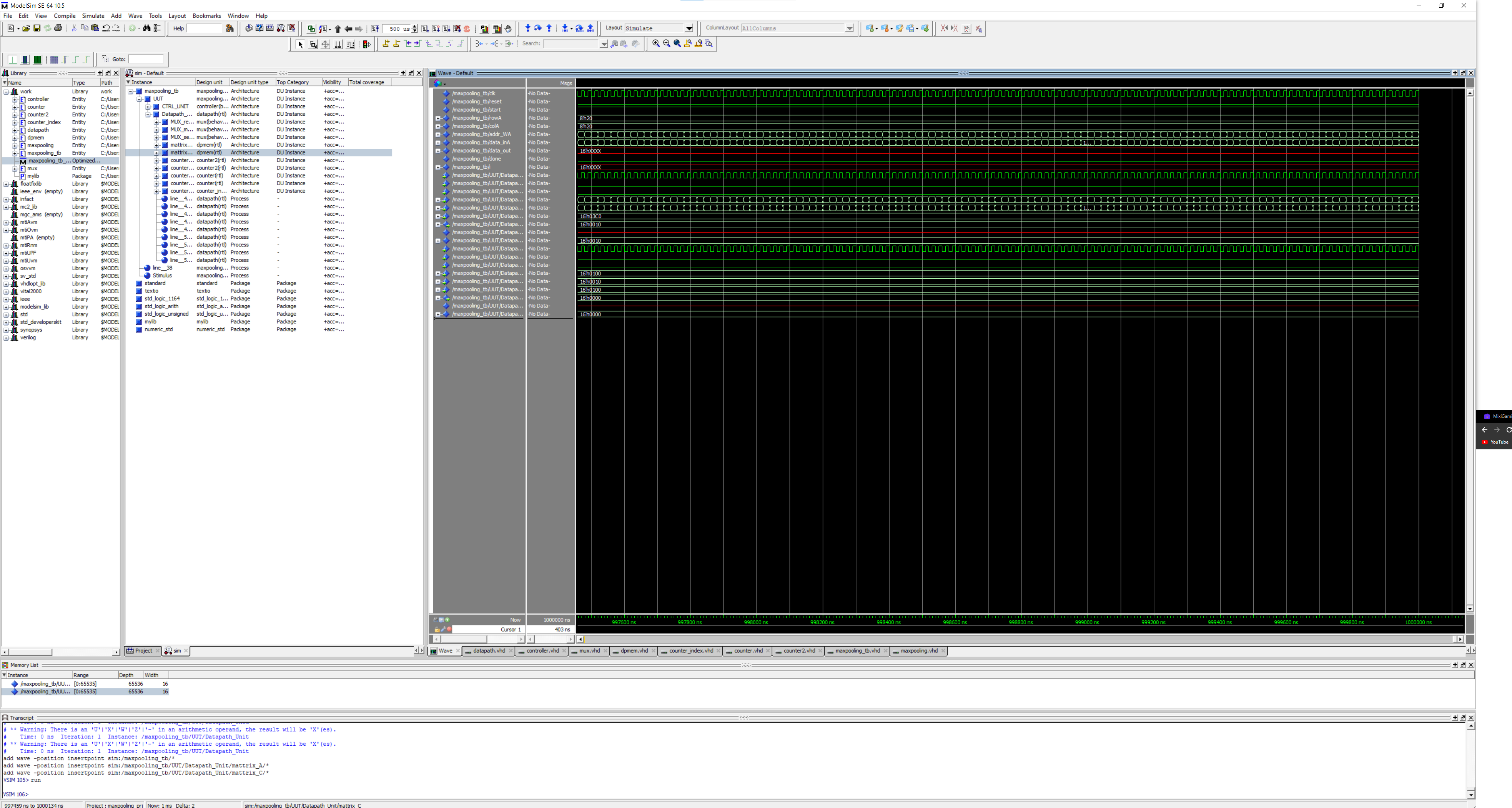
# Mô hình hóa bằng VHDL

# C:\Users\phamh\Downloads\Untitled Diagram.drawio (1).png

Hình 5‑0‑1. Mô hình hóa

# Mô phỏng/thực thi và đánh giá

Kết quả:



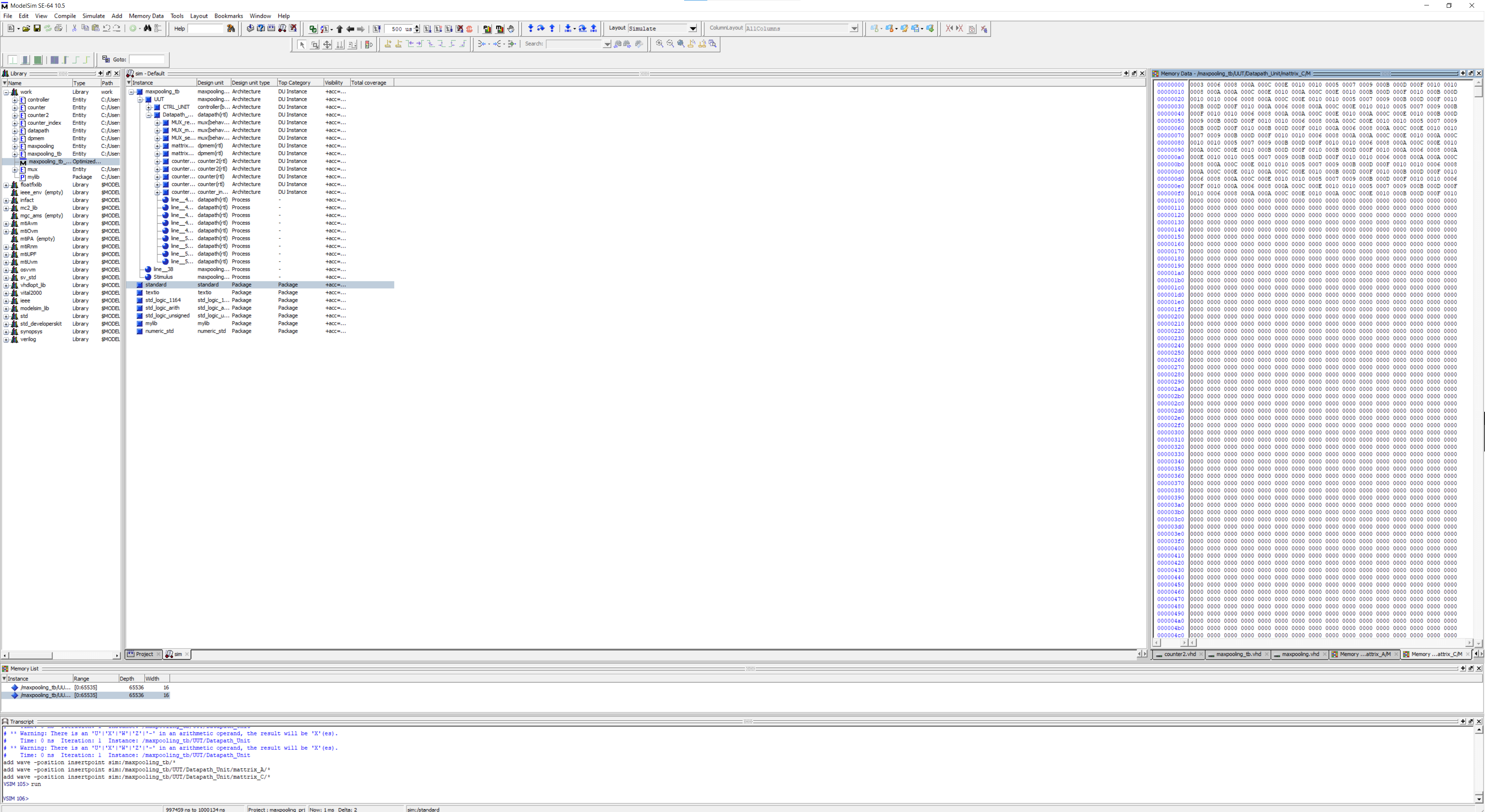
Hình 6‑1 Kết quả chạy thực tế

Ma trận đầu vào A:



Hình 6‑2 Ma trận đầu vào

Ma trận đầu ra C:



Hình 6‑3 Ma trận đầu ra

# Kết luận:

Thuật toán đã được xây dựng và cài đặt thành công trên phần mềm modelSim. Trong bài tập lớn của mình, để thuận tiện cho việc theo dõi kiểm tra kết quả em đã xây dựng thuật toán bằng ngôn ngữ java. Kết quả thu được và file source code chúng em update trên github: <https://github.com/hepham/DMA>

Folder: DMA-file chạy java minh họa thuật toán maxpooling

Folder: maxpooling chạy VHDL minh họa thuật toán maxpooling

Appendix A:

Appendix B: VHDL Code

(đóng gói thành tệp nén và gửi kèm báo cáo)

Appendix C:

Compress and email to hungnvnu@gmail.com

List of Figures

[Hình 1‑1. Ví dụ minh họa phép tính Pooling 2D. 5](#_Toc122014208)

[Hình 2‑1. Giao diện ghép nối I/O. 6](#_Toc122014209)

[Hình 4‑1: Mô hình máy FSMD. 8](#_Toc122014210)

[Hình 4‑2 Cấu trúc đơn vị xử lý dữ liệu Datapath 9](#_Toc122014211)

[Hình 4‑3: Máy FSM của đơn vị điều khiển. 11](#_Toc122014212)

[Hình 4‑4: Sơ đồ khối tổng thể của thiết kế. 11](#_Toc122014213)

[Hình 5‑1. Mô hình hóa 12](#_Toc122014214)

[Hình 6‑1 Kết quả chạy thực tế 13](#_Toc122014215)

[Hình 6‑2 Ma trận đầu vào 13](#_Toc122014216)

[Hình 6‑3 Ma trận đầu ra 14](#_Toc122014217)

List of Tables

[Bảng 1: Mô tả các tín hiệu vào ra. 7](#_Toc496109879)

References

1. https://towardsdatascience.com/intuitively-understanding-convolutions-for-deep-learning-1f6f42faee1