|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**  Description: Description: Description: logo_128**VIỆN ĐIỆN TỬ-VIỄN THÔNG**  ====o0o====  **BÁO CÁO THỰC TẬP KĨ THUẬT**  **NHẬN DIỆN CHỮ CÁI SỬ DỤNG MATLAB VÀ**  **THỰC HIỆN MẠNG NEURAL TRÊN FPGA**  **ESRC Lab - C9 420**   |  |  | | --- | --- | | GVHD : | PGS.TS Phạm Ngọc Nam  ThS Nguyễn Thị Kim Thoa |  |  |  |  |  |  | | --- | --- | --- | --- | --- | | Sinh viên thực hiện :  Mã số sinh viên :  Lớp : | | Nguyễn Minh Hiếu  20151336  Điện tử 03 – K60 | |  | |  |  | | |  |  | |   Hà Nội, tháng 8 năm 2018 |

DANH MỤC HÌNH VẼ

[Hình 2.1 Tế bào thần kinh sinh học 9](file:///D:\TatCaCacMon\ESRC\Báo%20cáo_HLD%20Hường.docx#_Toc523956393)

[Hình 2.2 Cấu trúc mạng neural 10](file:///D:\TatCaCacMon\ESRC\Báo%20cáo_HLD%20Hường.docx#_Toc523956394)

[Hình 2.3 Một neuron riêng lẻ 11](file:///D:\TatCaCacMon\ESRC\Báo%20cáo_HLD%20Hường.docx#_Toc523956395)

[Hình 2.4 Minh họa 6 bước đầu nhận diện chữ cái bằng Matlab 15](file:///D:\TatCaCacMon\ESRC\Báo%20cáo_HLD%20Hường.docx#_Toc523956396)

[Hình 2.5 Kết quả sau khi thực hiện bước 8 16](file:///D:\TatCaCacMon\ESRC\Báo%20cáo_HLD%20Hường.docx#_Toc523956397)

[Hình 2.6 Công cụ nftool trong Matlab 17](file:///D:\TatCaCacMon\ESRC\Báo%20cáo_HLD%20Hường.docx#_Toc523956398)

[Hình 2.7 Kiến trúc một chip FPGA 19](file:///D:\TatCaCacMon\ESRC\Báo%20cáo_HLD%20Hường.docx#_Toc523956399)

[Hình 2.8 Mạng neural 2:3:2 21](file:///D:\TatCaCacMon\ESRC\Báo%20cáo_HLD%20Hường.docx#_Toc523956400)

[Hình 2.9 Mạng neuron riêng lẻ 21](file:///D:\TatCaCacMon\ESRC\Báo%20cáo_HLD%20Hường.docx#_Toc523956401)

[Hình 2.10 Lưu đồ thuật toán giải thuật Booth 22](file:///D:\TatCaCacMon\ESRC\Báo%20cáo_HLD%20Hường.docx#_Toc523956402)

[Hình 2.11 Sơ đồ ASM giải thuật Booth 23](file:///D:\TatCaCacMon\ESRC\Báo%20cáo_HLD%20Hường.docx#_Toc523956403)

[Hình 2.12 Tổng quát mạch nhân 23](file:///D:\TatCaCacMon\ESRC\Báo%20cáo_HLD%20Hường.docx#_Toc523956404)

[Hình 2.13 Hàm sigmoid 24](file:///D:\TatCaCacMon\ESRC\Báo%20cáo_HLD%20Hường.docx#_Toc523956405)

[Hình 2.14 Tổng quát mạch hàm kích hoạt 24](file:///D:\TatCaCacMon\ESRC\Báo%20cáo_HLD%20Hường.docx#_Toc523956406)

[Hình 2.15 Mô phỏng một neuron 25](file:///D:\TatCaCacMon\ESRC\Báo%20cáo_HLD%20Hường.docx#_Toc523956407)

[Hình 2.16 Cấu trúc cụ thể mạng neural 2:3:2 26](file:///D:\TatCaCacMon\ESRC\Báo%20cáo_HLD%20Hường.docx#_Toc523956408)

[Hình 2.17 Mô phỏng mạng neural 2:3:2 26](file:///D:\TatCaCacMon\ESRC\Báo%20cáo_HLD%20Hường.docx#_Toc523956409)

# A. LỜI NÓI ĐẦU

Học phần Thực tập kĩ thuật nằm trong chương trình đào tạo của viện Điện tử - Viễn thông, giúp sinh viên tiếp cận môi trường, có thêm kinh nghiệm làm việc thực tế ở một công ty thuộc lĩnh vực Điện tử - Viễn thông hoặc tham gia vào một phòng nghiên cứu cả trong và ngoài trường

Trong học kì hè 20173, em đã tham gia thực tập tại Phòng thí nghiệm Hệ thống và Tính toán cấu hình lại (Embedded System and Reconfigurable Compute Labroratory - ESRC Lab), thuộc Trung tâm Nghiện cứu và Phát triện, viện Điện tử Viễn Thông, trường Đại học Bách Khoa Hà Nội. Em đã được các anh chị khóa trên hướng dẫn nghiên cứu đề tài “Nhận diện chữ cái sử dụng Matlab và thực hiện mạng neural trên FPGA”, với mục tiêu làm quen với trí thông minh nhân tạo (AI – Artificial Intelligence) và luyện tập ngôn ngữ mô tả phần cứng Verilog. Vì lần đầu biết tới mạng neural nhận tạo (ANNs – Artificial Neural Networks) và ngôn ngữ Verilog nên để tài không được hoàn chỉnh, bọn em dự định sẽ tiếp tục phát triển trong năm học tới.

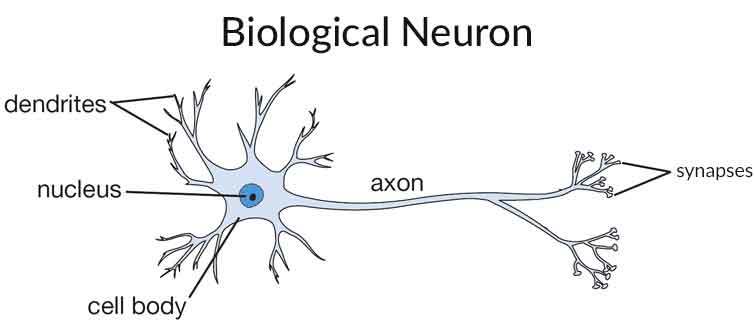
Em xin cảm ơn Ban giám hiệu trường Đại học Bách Khoa Hà Nội và Ban lãnh đạo viện Điện tử - Viễn thông đã tạo điều kiện cho chúng em tham gia thực tập, cũng như thầy Phạm Ngọc Nam, các anh chị, các bạn trong Lab đã đồng hành và hỗ trợ em em hoàn thành tốt học phần thực tập.

# B. NỘI DUNG

# Chương 1. Nội dung thực tập

## 1.1 Giới thiệu mạng neural nhân tạo

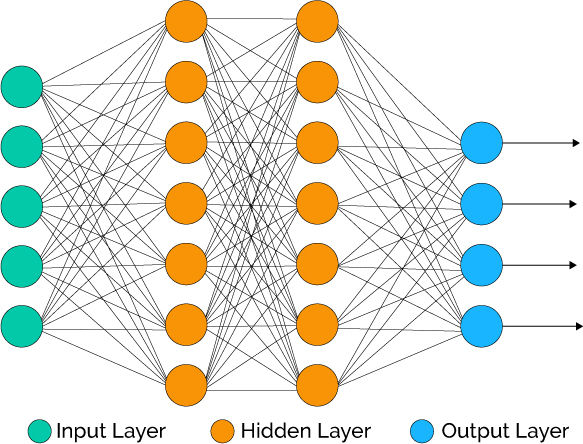
Thuật ngữ “neural” bắt nguồn từ đơn vị chức năng cơ bản của tế bào thần kinh (neuron) trong não người. Mạng neural sinh học là một mạng lưới liên kết hàng tỷ tế bào thần kinh với hàng nghìn tỷ kết nối giữa chúng. Hình 2.1 mô tả một tế bào thần kinh sinh học (biological neuron).



Hình 2.1 Tế bào thần kinh sinh học

Mạng neural nhân tạo là một hệ thống tính toán được xây dựng dựa trên mạng neural sinh học của não người. Mạng neural nhân tạo có rất nhiều ứng dụng trong thực tế: điều khiển hệ thống, chơi trò chơi điện tử, đưa ra quyết định, nhận diễn hình mẫu, khuôn mặt, chẩn đoán y tế, khai thác dữ liệu giao dịch tự động,…

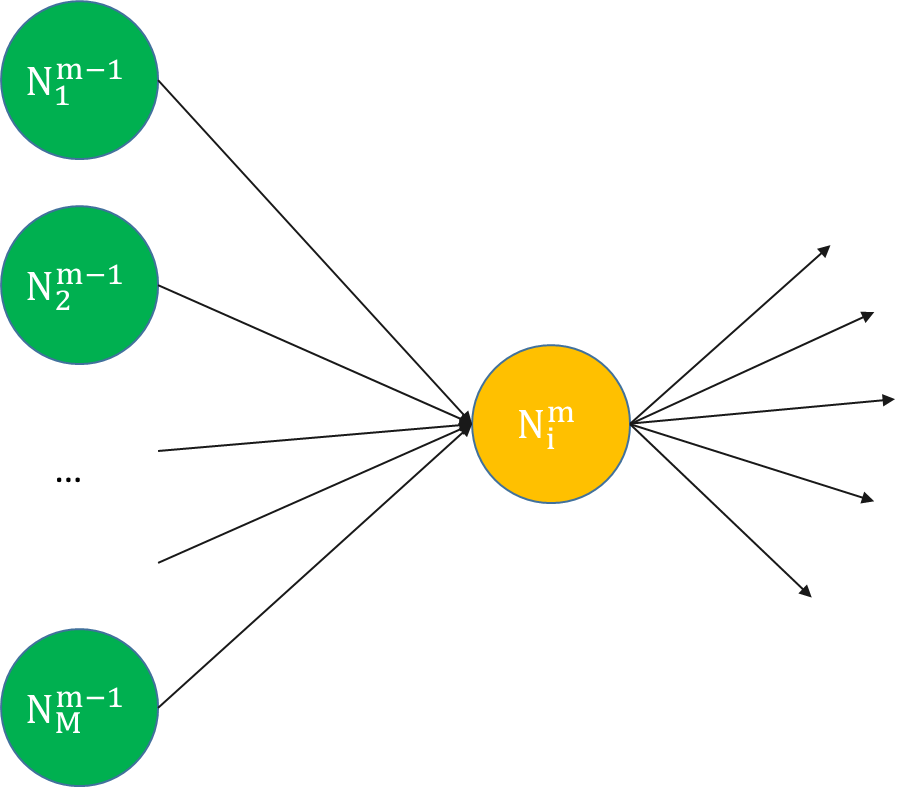
Mạng neural nhân tạo gồm một lớp vào (input layer), một hoặc nhiều lớp ẩn (hidden layer) và một lớp ra (output layer). Một lớp bao gồm nhiều neuron, các neuron ở một lớp sẽ được kết nối đầy đủ với tất cả các neuron ở lớp ngay trước thông qua hàm truyền (propagation function) với trọng số (weight) tương ứng (ngoài ra có thể có thêm phần độ lệch (bias)), sau đó được xử lí qua hàm kích hoạt (activation function) trước khí tiếp tục kết nối với tất cả các neuron lớp sau. Hình 2.2 mô tả một mạng neural.



Hình 2.2 Cấu trúc mạng neural

Mạng neural nhân tạo học bằng các điều chỉnh trong số và độ lệch, lặp đi lặp lại đến khi mang lại kết quả mong muốn. Mạng được huấn luyên trước bằng các sử dụng bộ quy tắc còn gọi là thuật toán học tập. Các thuật toán học tập phổ biến bao gồm: giảm gradient (gradient descent), lan truyền ngược (back propagation), luật Hebb (Hebb rule), luật Hopfield (Hopfield law), thuật toán trung bình tối thiểu (LMS - least mean square), học tập cạnh tranh (competitive learning). Quá trình học gồm nhiều kiểu học khác nhau: học có giám sát (supervised learning), học không giám sát (unsupervised learning), học tăng cường (reinforcement learning), học ngoại tuyến (offline learning), học trực tuyến (online learning).

## 1.2 Cấu trúc một neuron



Hình 2.3 Một neuron riêng lẻ

Hình 2.3 mô tả một neuron thứ i thuộc lớp thứ m, được kết nối với các neuron từ thứ 1 đến thứ M của lớp m-1 bằng hàm truyền như sau:

+ (1)

Trong đó là đầu vào của neuron thứ i thuộc lớp thứ m, là đầu ra của neuron thứ j thuộc lớp thứ m-1, là trọng số kết nối từ neuron thứ j thuộc lớp thứ m-1 đến neuron thứ i thuộc lớp thứ m, là bias của neuron thứ i thuộc lớp thứ m

Mối quan hệ giữa đầu ra và đầu vào của 1 neuron được thể hiện qua hàm kích hoạt. Trong đề tài này sử dụng hàm sigmoid làm hàm kích hoạt:

(2)

Sau đó đầu ra của neuron lại được nối với các neuron lớp kế tiếp.

## 1.3 Thuật toán lan truyền ngược (backpropagation algorithm)

Các giá trị weight và bias sẽ được gán các giá trị ngẫu nhiên khác 0. Sau đó các vector trong tập huấn luyện sẽ lần lượt được đưa vào mạng để tính toán ra output và từ đó tính toán được hàm giá (cost function) theo công thức:

(3)

Trong đó:

(4)

Với i là chỉ số tương ứng với mẫu huấn luyện thứ i.

Sau khi tính toán xong giá trị của cost funtion sẽ là giai đoạn lan truyền ngược (back propagation). Ở giai đoạn này, các đạo hàm riêng của hàm Cost function theo các và sẽ được tính toán.

Ở lớp output, giá trị unit error sẽ được tính theo công thức:

(5)

Trong đó p là thứ tự của neuron output, y là activation của neuron ouput. Qua đó ta tính được:

(6)

(7)

Đối với lớp hidden:

(8)

Qua đó ta cũng tính được:

(9)

(10)

Giá trị đạo hàm riêng phần sẽ được tính theo công thức:

(11)

(12)

Đến đây ta tính:

(13)

(14)

Trong đó là một số thực dương gọi là learning rate. Một điểm cực tiểu của hàm số tính chất, đạo hàm bên trái âm (hàm trong lân cận phía trái điểm này nghịch biến), đạo hàm bên phải dương (hàm trong lân cạn phía phải điểm này đồng biến). Do đó nếu lỡ chọn tham số làm cho đạo hàm âm thì phải tăng tham số lên để nó sát về điểm cực tiểu, ngược lại nếu lỡ chọn tham số cho đạo hàm dương thì ta phải giảm tham số đi để nó sát về điểm cực tiểu, suy ra phải là số dương để đảm bảo ta luôn đi ngược chiều với đạo hàm, có thể chọn càng lặp càng nhỏ để tìm điểm sát với cực tiểu hơn.

Giá trị của các và sẽ được cập nhật theo công thức:

(15)

(16)

Ta lặp đi lặp lại các bước trên đến khi nào các đạo hàm riêng sát với 0 thì dừng.

Thuật toán lan truyền ngược hiểu đơn giản là ban đầu tính unit’s error của các neuron lớp đầu ra trước rồi tính unit’s error của các lớp trước theo các lớp sau. Sau đó tính các đạo hàm theo unit’s error. Giá trị trong biểu thức (13), (14) ảnh hưởng khá nhiều đến quá trình học của thuật toán. Thường giá trị này sẽ chọn khoảng từ 0 đến 1.

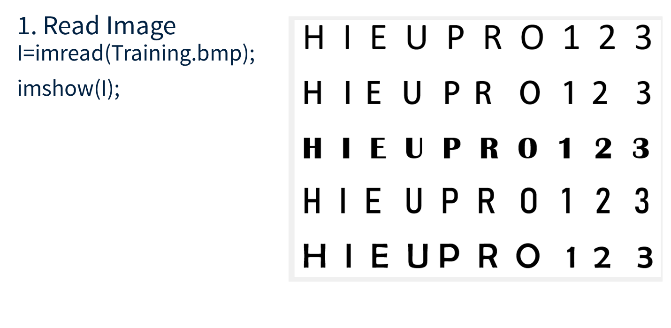
## 1.4 Nhận diện chữ cái sử dụng Matlab

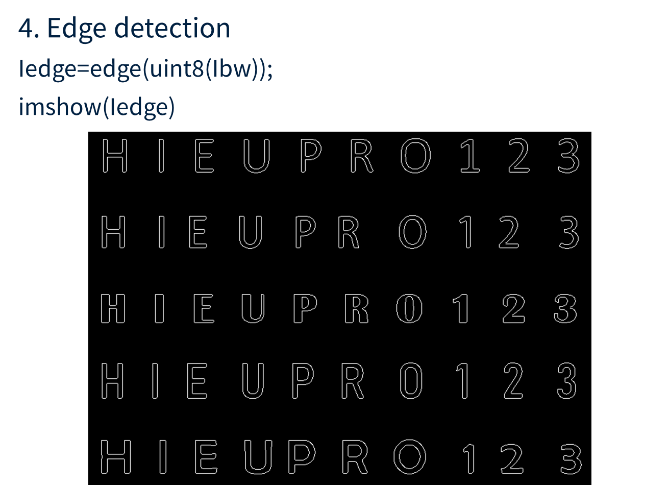
Matlab (Matrix Laboratory), là một bộ phần mềm toán học của hãng Mathworks để lập trình, tính toán số và có tính trực quan rất cao. Matlab làm việc chủ yếu với ma trận, với chuỗi kí tự Matlab cũng xem là một dãy các kí tự hay là dãy mã số của các ký tự. Matlab dùng để giải quyết các bài toán về giải tích số, xử lý tín hiệu số, xử lý đồ họa,… mà không phải lập trình cổ điển. Hiện nay, Matlab có đến hàng ngàn lệnh và hàm tiện ích. Ngoài các hàm cài sẵn trong chính ngôn ngữ, Matlab còn có các lệnh và hàm ứng dụng chuyên biệt trong các Toolbox, để mở rộng môi trường Matlab nhằm giải quyết các bài toán thuộc các phạm trù riêng. Các Toolbox khá quan trọng và tiện ích cho người dùng như toán sơ cấp, xử lý tín hiệu số, [xử lý ảnh](http://vimach.net/tags/xử%20lý%20ảnh/), xử lý âm thanh, ma trận thưa, logic mờ,… Ở phần này ta sử dụng công cụ nftool của Matlab để nhận diện chữ cái.

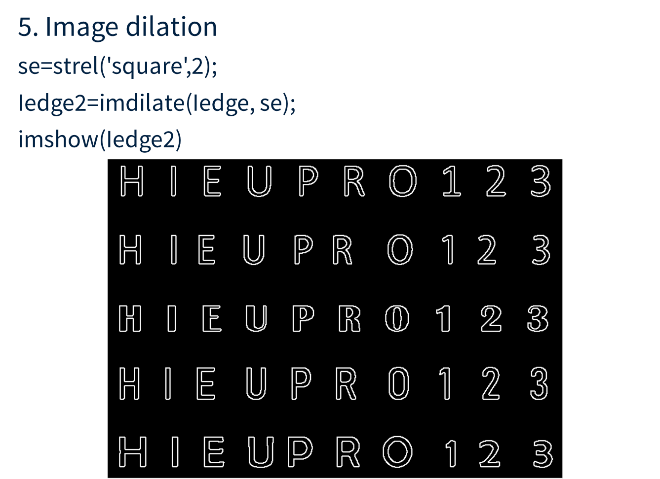
Để thực hiện chương trình trên Matlab, ta đi theo 8 bước tiền xử lí ảnh sau:

1. Tải ảnh cần nhận diện
2. Chuyển ảnh về mức xám
3. Chuyển ảnh về mức nhị phân
4. Xác định cạnh
5. Tăng gấp đôi cạnh
6. Làm đầy các đối tượng

Hình 2.4 minh họa 6 bước trên khi thực hiện bằng Matlab.





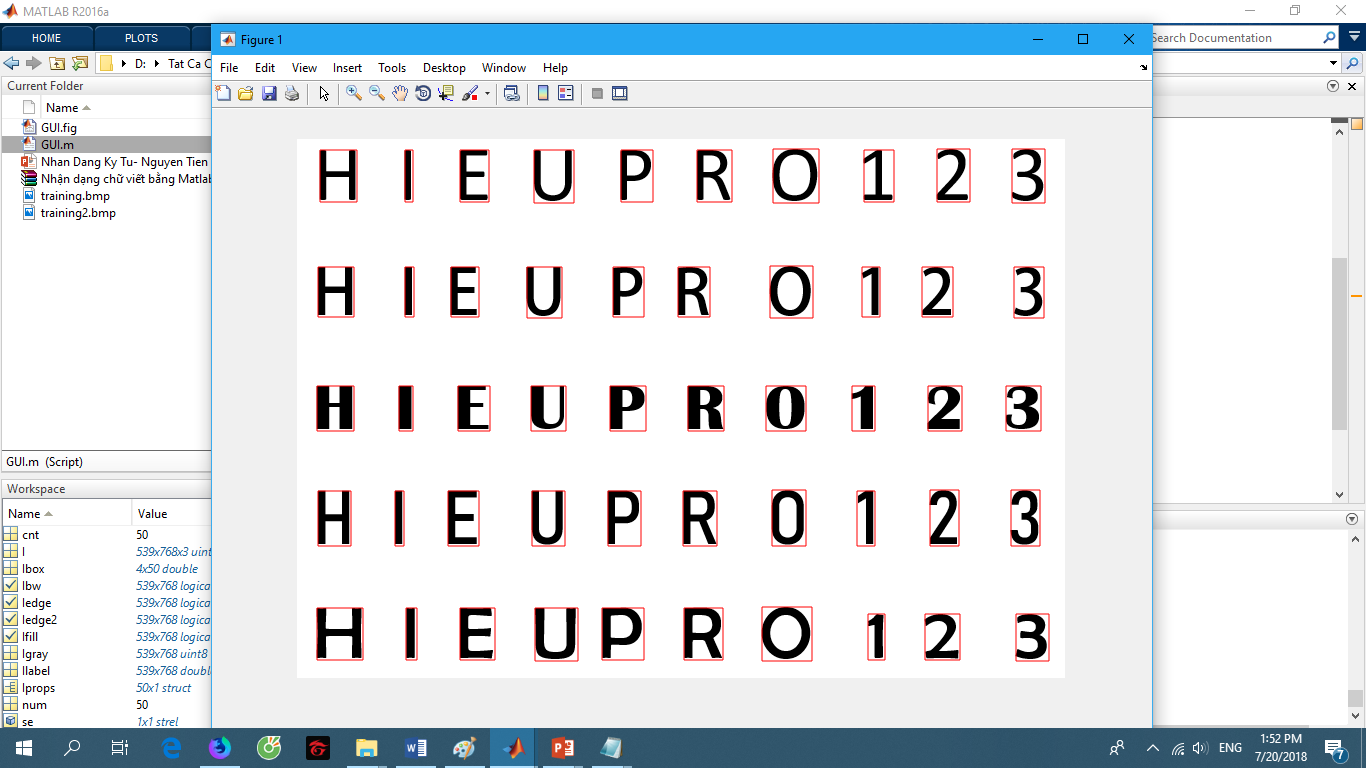


Hình 2.4 Minh họa 6 bước đầu nhận diện chữ cái bằng Matlab

Sau 6 bước trên, ta thực hiện 2 bước quan trọng là

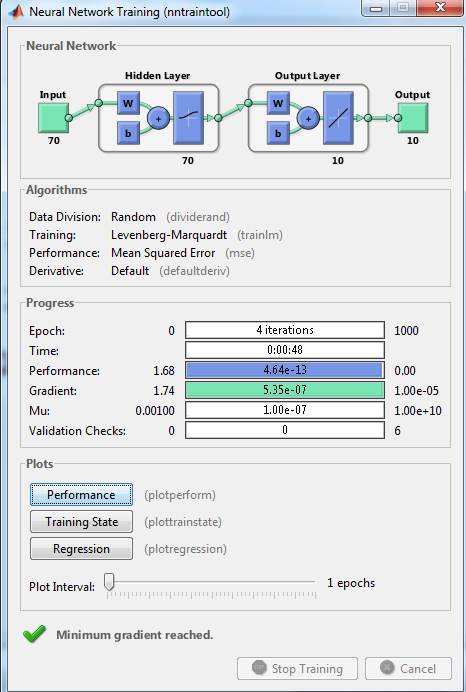
1. Đánh dấu các đối tượng, xác định vị trí các đối tượng
2. Vẽ hình chữ nhật bao quanh các đối tượng dựa trên ảnh gốc

Hình 2.5 là kết quả sau khi thực hiện bước 8.



Hình 2.5 Kết quả sau khi thực hiện bước 8

Trong chương trình này sử dụng dữ liệu đầu vào có 70 mẫu của 40 phần tử từ ảnh trong phần tiền xử lý. Còn 70 mẫu của 10 phần tử còn lại dùng để tạo mẫu test. Đầu tiên ta chọn đối tượng cần nhận diện, tiếp theo xử lí đối tượng và tạo dữ liệu kiểm tra, sau đó tạo mạng neural huấn luyện bằng công cụ nftool (hình 2.6).



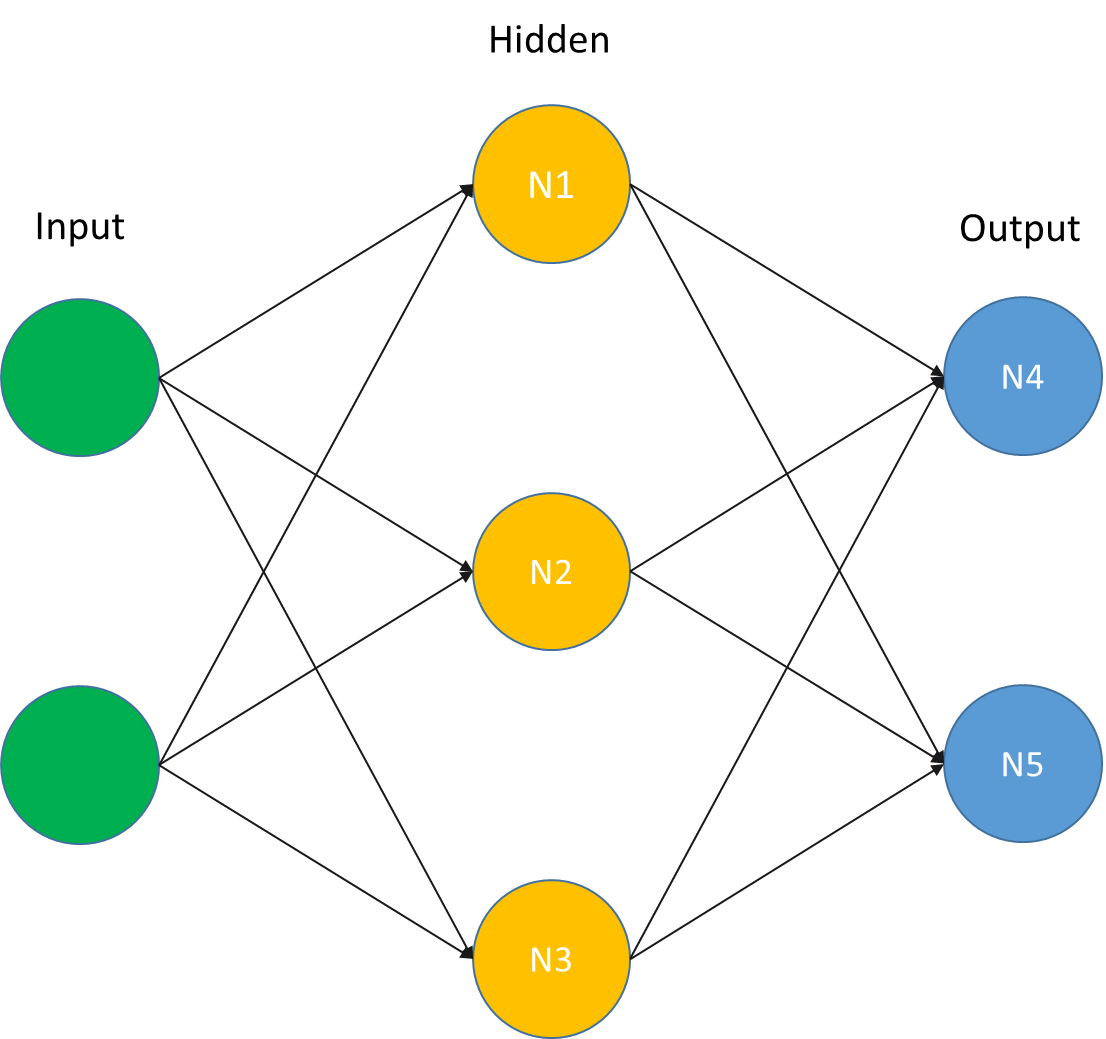
Hình 2.6 Công cụ nftool trong Matlab

Cuối cùng là nhận diện kí tự và cho ra kết quả. Kết quả chạy cho thấy chương trình chưa thực sự nhận diện đúng kí tự, và chỉ nhận diện được các kí tự đã huấn luyện.

Chi tiết code và giao diện, video chạy Matlab có ở trong phần phục lục.

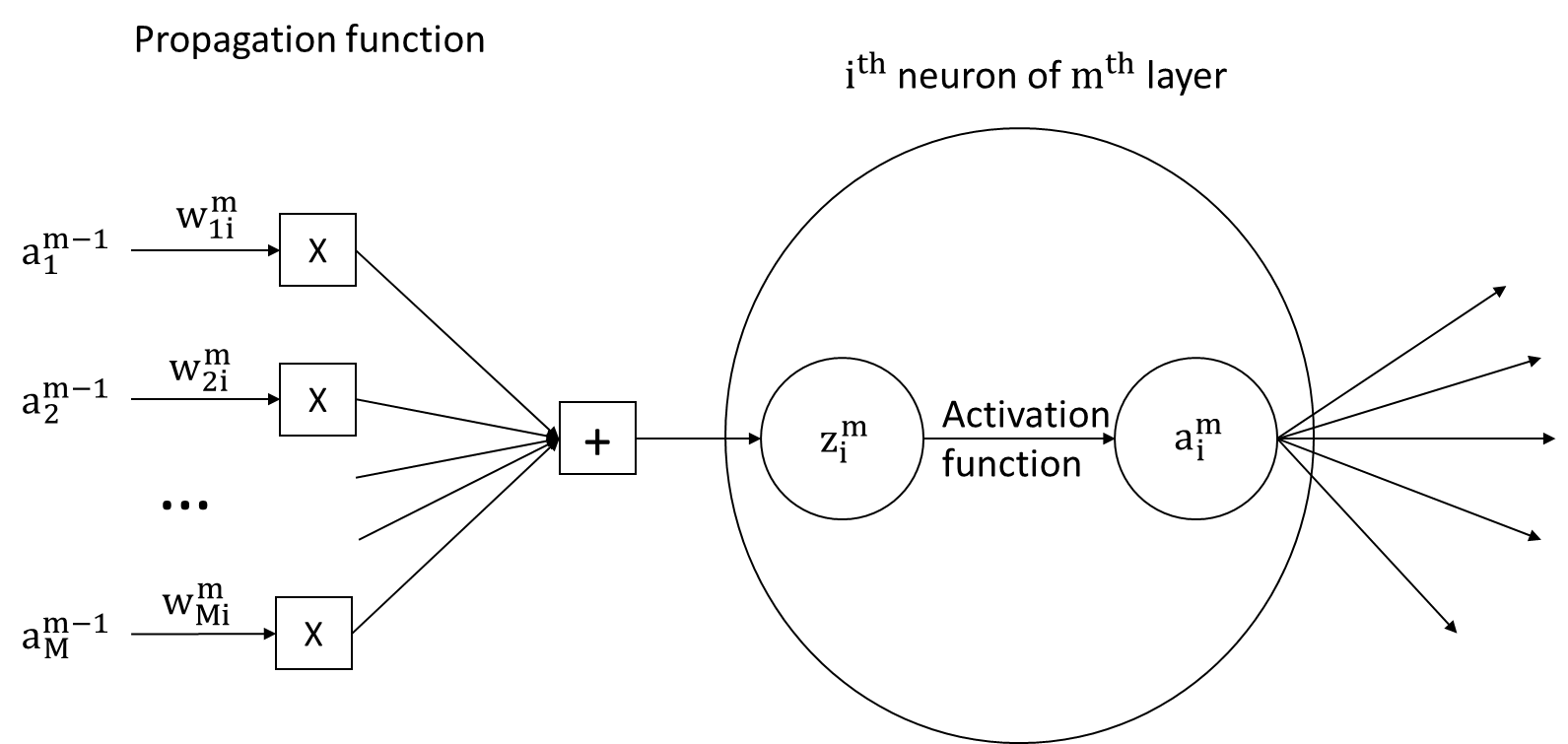
## 1.5 Khái quát về FPGA, ngôn ngữ Verilog và mạng neural 2:3:2

Hình 2.8 là một mạng neural 2:3:2 (2 neuron lớ đầu vào, 3 neuron lớp ẩn, 2 neuron lớp đầu ra). Ta chỉ quan tâm 5 neuron ở lớp ẩn và lớp đầu ra như ghi chú ở trên hình 2.8.



Hình 2.8 Mạng neural 2:3:2

Để thiết kế một mạng neural đầy đủ, ta thiết kế từng neuron riêng lẻ, sau đó nối các neuron với nhau. Trong một neuron, ta phải thiết kế các module mạch nhân và module hàm kích hoạt. Hình 2.9 là một neuron riêng lẻ.



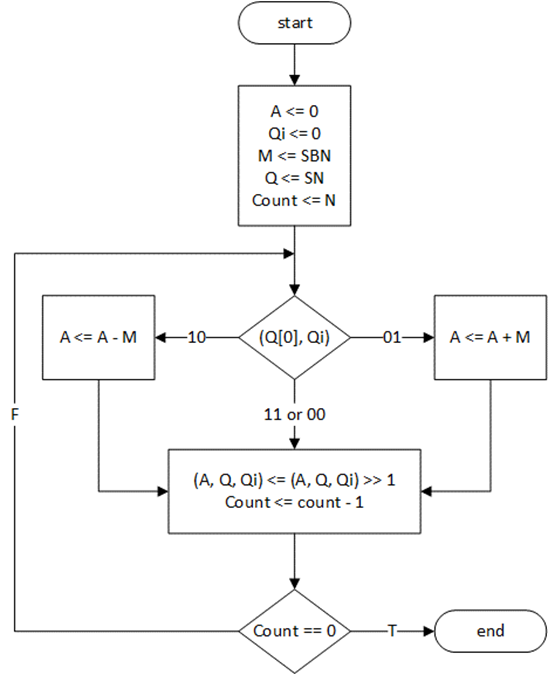
Hình 2.9 Mạng neuron riêng lẻ

## 1.6 Thiết một neuron

### 1.6.1 Module mạch nhân

Như đã thấy ở hình trên, một neuron được kết nối với M neuron ở lớp trước đó, chúng ta cần M bộ nhân 2 số thực, và M/2 bộ cộng 2 số thực (nếu M chẵn), (M+1)/2 bộ cộng 2 số thực (nếu M lẻ). Để thực hiện bộ cộng sử dụng toán tử cộng có sẵn trong Verilog.

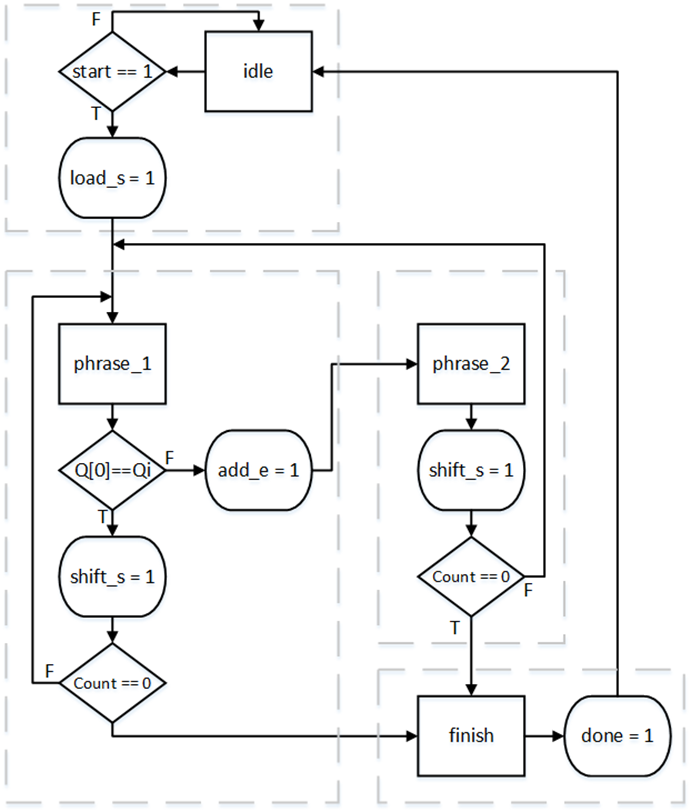
Mạch nhân số thực thực hiện chức năng nhân hai số thực có dấu dấu phẩy động. Quá trình nhân sẽ diễn ra trong nhiều chu kỳ đồng hồ. Cách thực hiện nhân ở đây là sử dụng giải thuật booth. Lưu đồ thuật toán được thể hiện ở hình 2.10.



Hình 2.10 Lưu đồ thuật toán giải thuật Booth

Ban đầu gán A và Qi=0, 2 biến M và Q lưu số bị nhận và số nhân, Qi là bit dấu, count lưu số bit của 2 số cần nhân với nhau. Theo sơ đồ, trong mỗi vòng lặp, nếu giá trị của Q[0]Qi=01 thì cộng thêm M vào A, 10 thì trừ M từ A, còn lại 11 hoặc 00 thì thực hiện quay phải toàn bộ dãy A,Q,Qi. Sau mỗi vòng lặp giảm count đi 1, lặp đến khi count bằng 0 thì {A,Q} chính là kết quả.

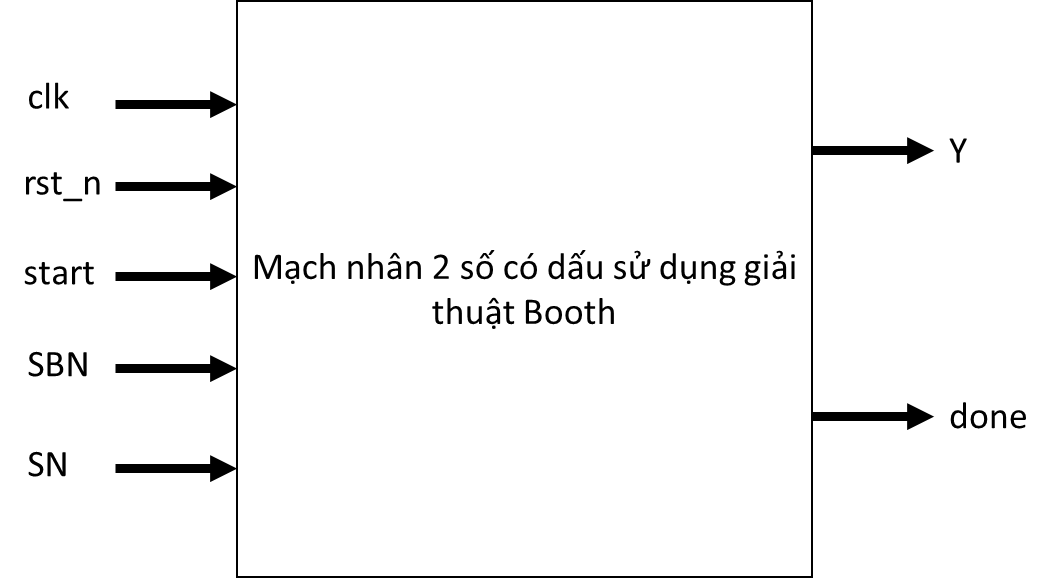
Sau đây ta xây dựng sơ đồ máy trạng thái thuật toán (Algorithmic state machine – ASM) (hình 2.11).



Hình 2.11 Sơ đồ ASM giải thuật Booth

Sơ đồ ASM được suy ra từ lưu đồ thuật toán. Mạch bao gồm các trạng thái: idle (trạng thái ban đầu), phrase\_1,phrase\_2, finish. Tín hiệu add\_s cho phép chuyển trạng thái từ phrase\_1 sang phrase\_2 và cộng. Tín hiệu shift\_s cho phép trừ bộ đếm và dịch các giá trị trong thanh ghi. Cuối cùng ở trạng thái finish, một tín hiệu done được xuất ra thông báo quá trình nhân hoàn tất.

Tổng quát mạch nhân như hình 2.12.



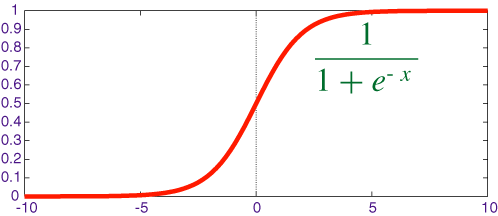
Mạch có 5 tín hiệu vào: xung đồng hồ (clk), reset (rst\_n), tín hiệu cho phép mạch chạy (start), số bị nhân (SBN), số nhân (SN); 2 tín hiệu ra: kết quả phép nhân (Y), tín hiệu báo hiệu quá trình nhân đã hoàn tất (done).

Hình 2.12 Tổng quát mạch nhân

Chi tiết code mạch nhân có ở trong phần phục lục.

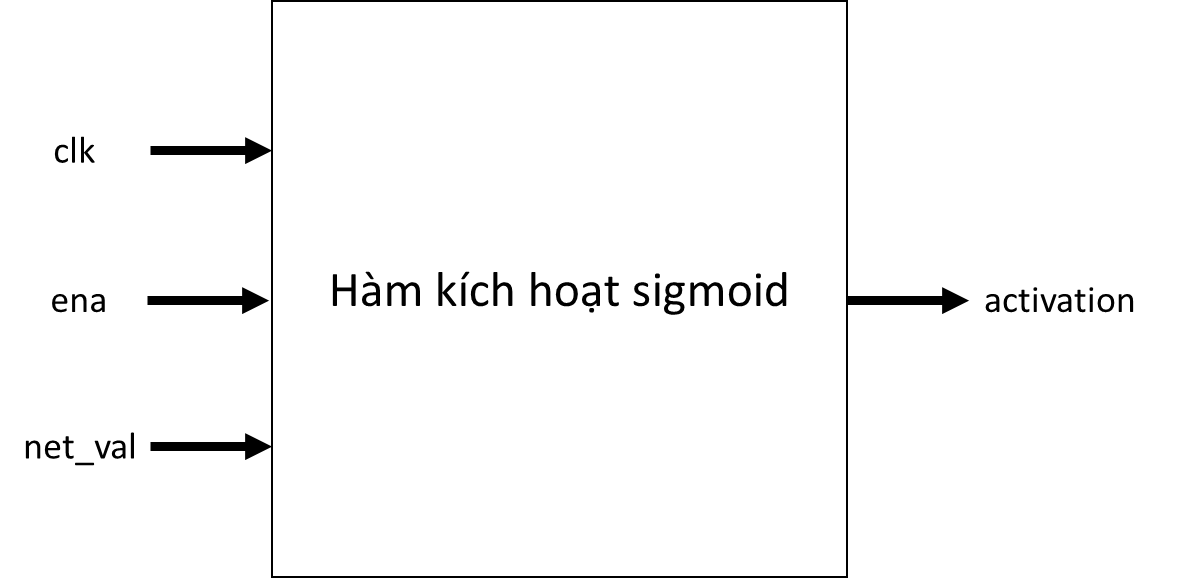
### 1.6.2 Module hàm kích hoạt

Hàm sigmoid là một hàm phi tuyến, nhận vào một giá trị thực bất kì và cho ra giá trị nằm trong khoảng từ 0 đến 1 (hình 2.13).



Hình 2.13 Hàm sigmoid

Hàm kích hoạt sigmoid được gen code bằng matlab, ta dùng công cụ HDL Coder trong Mathlab để thay thế hàm Sigmoid với Lookup Table.

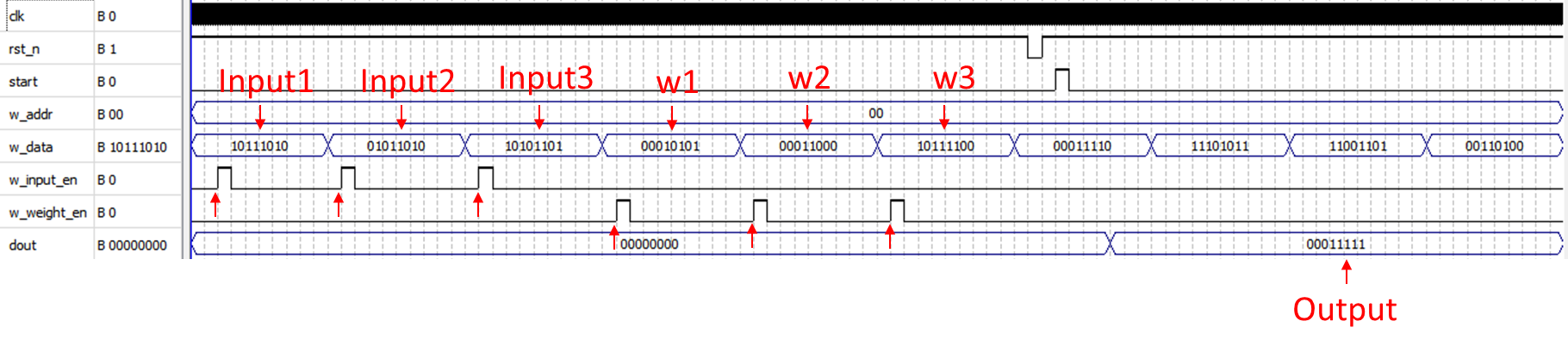


Hình 2.14 Tổng quát mạch hàm kích hoạt

Như hình 2.14, mạch hàm kích hoạt gồm 3 tín hiệu đầu vào: xung đồng hồ (clk), tín hiệu cho phép chạy module, giá trị vào neuron, tín hiệu đầu ra là giá trị cần tính của hàm sigmoid ra khỏi neuron. Code cụ thể đã tạo bằng matlab có trong phần phụ lục.

### 1.6.3 Mô phỏng một neuron

Hình 2.15 là kết quả mô phỏng một neuron thực hiện bằng công cụ Modelsim trong Quartus 13.

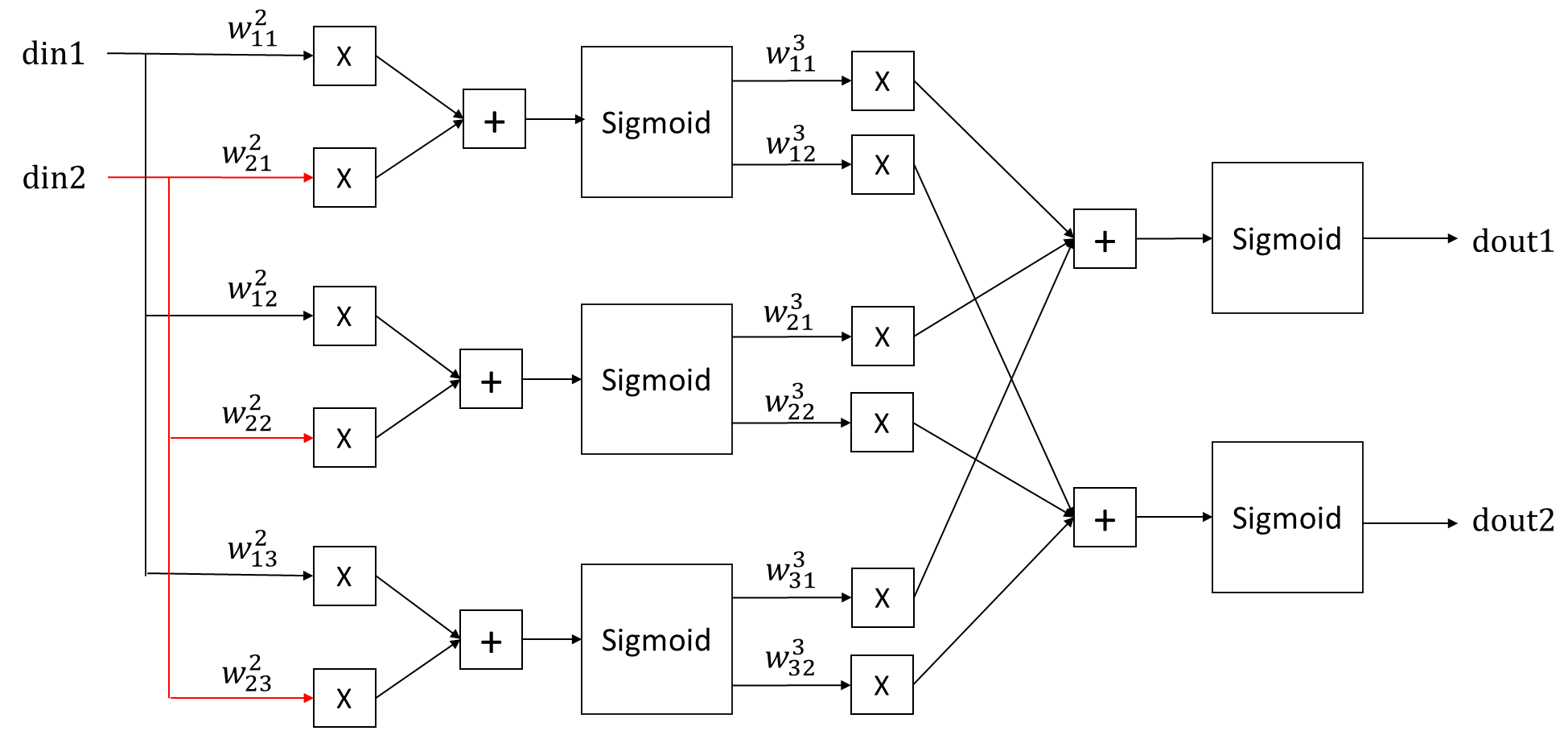


Hình 2.15 Mô phỏng một neuron

3 vector input được đưa vào tuần tự mỗi khi có tín hiệu w\_input\_en, tương tự với 3 trọng số khi có tín hiệu w\_weight\_en, tín hiệu start được kích hoạt, sau một loạt xug clock thì output hiện kết quả đầu ra. Kết quả này đã được kiểm tra là chính xác.

## 1.7 Thực hiện mạng neural 2:3:2

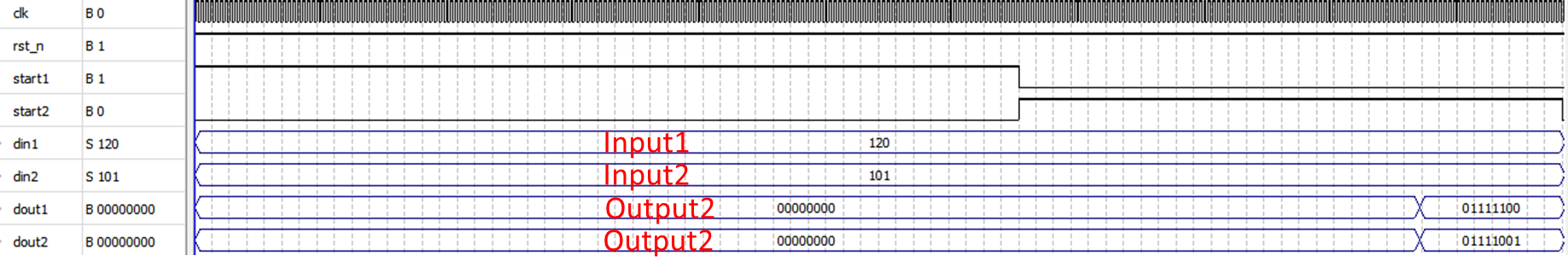
Cấu trúc cụ thể mạng neural 2:3:2 như hình 2.16.



Hình 2.16 Cấu trúc cụ thể mạng neural 2:3:2

Việc kết nối các neuron riêng lẻ với nhau không được kết quả như dự tính, bởi trình độ còn hạn chế, nên ở đây ta thiết kế mạng neural như một mạch tổ hợp với các module mạch nhân và module hàm kích hoạt đã có. Code toàn bộ mạch ó ở trong phần phụ lục.

Kết quả mô phỏng mạng neural 2:3:2 như hình 2.17.



Hình 2.17 Mô phỏng mạng neural 2:3:2

2 vector đầu vào din1 và din2 ở dạng thập phân, tín hiệu start1, start2 cho phép các mạch nhân từ layer1 sang layer2 và từ layer2 sang layer3 hoạt động, dout1 và dout2 là 2 vector đầu ra ở 2 neuron N4 và N5.

# Chương 2. Nhận xét, đề xuất

## 2.1 Ưu điểm

Trí thông minh nhân tạo hiện nay có tính thực tế cao và được ứng dụng rất nhiều trong cuộc sống. Đề tài chúng em thực hiện là đề tài hay, giúp chúng em làm quen với mạng neural, sử dụng cả phần mềm (Matlab) và phần cứng (FPGA) để triển khai mạng neural. Trong thời gian thực tập tại ESRC Lab, em đã được trau dồi và luyện tập nhiều về các kĩ năng:

* Kĩ năng giao tiếp và làm việc nhóm
* Kĩ năng làm slide, thuyết trình, báo cáo

Các thành tựu đạt được sau khóa thực tập là:

* Sử dụng được Matlab thực hiện thành công nhân diện chữ cái bằng nftool
* Luyện tập Verilog, xây dựng được các module nhỏ và ghép lại thành module lớn
* Tìm hiểu về trí thông minh nhân tạo và mạng neural, triển khai mạng neural đơn giản trên FPGA

## 2.2 Nhược điểm

Vì làm việc theo nhóm nhiều người, khả năng lên kế hoạch còn hạn chế nên chúng em chưa có thời gian làm việc nhiều cùng nhau. Bên cạnh đó khả năng tiếng Anh còn hạn chế nên việc nghiên cứu sách vở mất rất nhiều thời gian. Trong đề tài thực hiện vẫn còn thiếu sót:

* Chưa ghép được các neuron riêng lẻ lại với nhau
* Chưa thực hiện được thuật toán lan truyền ngược

## 2.3 Đề xuất

Trong năm học bọn em sẽ tiếp tục nghiên cứu để hoàn thiện đề tài. Em hi vọng các khóa sinh viên sau cũng sẽ tham gia những khóa thực tập bổ ích như thế này.

# C. KẾT LUẬN

Trong thời gian thực tập cùng các anh chị và các bạn, em đã học hỏi được nhiều kiến thức và kĩ năng thực tế, tích lũy thêm kinh nghiệm làm việc, các buổi thảo luận nhóm và lên thuyết trình giúp em tự tin hơn. Chúng em có thêm kiến thức mới về mạng neural, những ứng dung của mạng neural, cách huấn luyện mạng neural, ngoài ra còn được luyện tập về Matlab và ngôn ngữ Verilog. Đề tài tuy còn nhiều thiếu sót nhưng là kết quả làm việc tích cực của cả nhóm. Một lần nữa em xin được chân thành cảm ơn viện Điện tử-viên thông đã giúp cho chúng em có học phần thực tập kĩ thuật, chân thành cảm ơn thầy Phạm Ngọc Nam chủ nhiệm phòng Lab đã tạo điều kiện để chúng có thể thực tập tại ESRC Lab, cảm ơn các anh chị và các bạn đã cùng em hoàn thành đề tài này.

# Tài liệu tham khảo

[1] Pong P, Chu, FPGA Protyping by Verilog Example, A John Wiley & Sons, Inc., Publication, 2008

[2] David Kriesel, A Brief Introduction to Neural Networks, dkriesel.com, 2005

[3] Mentor: prof. Primož Potočnik, Student: Žiga Zadnik, Character Rcognition:

Handwritten character Recognition: Training a Simple NN for classification using MATLAB

[4] Mrs. Rana D. Abdu-Aljabar, Design and Implementation of Neural Network in FPGA, Journal of Engineering and Development, Vol. 16, No.3, Sep. 2012 ISSN 1813- 7822

[5] Internet