ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH
——————o0o——————

BÁO CÁO BÀI TẬP LỚN

Môn : Thiết kế luận lí

Đề tài : Sử dụng màn hình thông qua giao tiếp VGA (Trò chơi Rắn săn mồi)

Giáo viên hướng dẫn : Nguyễn Xuân Quang

Nhóm 6 : Ung Ngô Minh Lăng (2011507)

Trần Văn Kiên (2013552)

Nguyễn Huy Hoàng (2013231)

TP Hồ Chí Minh, tháng 6/2021

Mục lục

1 Giới thiệu

1.1 Giới thiệu về đề tài

Sử dụng các kiến thức đã học trong môn Thiết kế luận lý HDL và các môn học có liên quan để xây dựng trò chơi rắn săn mồi bằng ngôn ngữ verilog trên kit phát triển FDGA DE2i-150 và sử dụng màn hình để hiện thị thông qua giao tiếp VGA

Giới thiệu cơ bản về VGA:

Video Graphics Array (VGA) là một bộ điều khiển màn hình và tiêu chuẩn đồ họa đi cùng với nó, được giới thiệu năm 1987 bởi IBM cùng dòng máy tính cá nhân PS/2, và trở nên phổ biến rộng rãi trong nền công nghiệp máy tính cá nhân trong vòng ba năm. Thuật ngữ này bây giờ dùng để chỉ tiêu chuẩn hiển thị, hay là cổng hiển thị màn hình máy tính, hay là độ phân giải hiển thị đặc trưng 640x480. Ngày nay, các bộ điều khiển VGA đã có thể hỗ trợ xuất tín hiệu với độ phân giải cao hơn, lên đến Full HD.

Giới thiệu về kit FDGA DE2i-150 :

DE2i-150 là một nền tảng đột phá, với sự kết hợp của bộ vi xử lí nhúng Intel N2600 và bộ Altera Cyclone IV GX FPGA của hãng Altera. Chính nhờ sự kết hợp này, DE2i-150 trở thành một hệ thống máy tính đầy đủ tính năng, và có hiệu năng xử lý rất cao. Đặc biệt, bộ Altera Cyclone IV GX FPGA nằm trên board DE2i-150 có thể tăng tốc khả năng đáp ứng của hệ thống mà vẫn giữ nguyên chi phí giải pháp và hiệu quả năng lượng.

DE2i-150 có đến 150.000 phần tử logic, với sự mền dẻo, linh hoạt của khả năng tái cấu trúc phần cứng, nó có thể đáp ứng cho bất cứ nhiệm vụ nào .Bộ vi xử lí của Intel và bộ thiết bị FPGA được liên kết với nhau thông qua 2 luồng PCIe tốc độ cao, đảm bảo cho việc truyền dữ liệu giữa chúng đạt tốc độ cao.Chính nhờ những điều này, DE2i-150 sẽ là một công cụ tuyệt vời để xử lý các tác vụ đặc biệt, cũng như thiết kế phần cứng

Giới thiêu về trò chơi rắn săn mồi :

Trò chơi rắn săn mồi xuất hiện vào năm 1997 trên dòng điện thoại Nokia 6610, tuy tại thời điểm đó game chỉ là những ô vuông xếp liên tiếp nhau di chuyển trên một màn hình màu xanh đơn giản, nhưng nó đã gây nên sứt hút vô cùng lớn và đã xây dựng rất thành công tên tuổi của mình. Với bốn trăm triệu bản được xuất xưởng và đến hiện tại đã là phiên bản thứ tám. Khả năng gây nghiện và sự hấp dẫn của rắn săn mồi cho tới thời điểm hiện tại là không thể bàn cãi.

Cách chơi game vô cùng đơn giản, việc của người chơi là chỉ cần điều khiển bằng 4 nút lên, xuống, trái, phải để điều khiển con rắn đi ăn các thức ăn được hiển thị ngẫu nhiên trên màn hình. Sau mỗi lần ăn được mồi, thân rắn sẽ dài ra và người chơi sẽ nhận được số điểm nhất định. Đồng thời, người chơi phải tránh không để con rắn va phải tường cũng như thân của nó trong quá trình di chuyển, nếu không làm được điều đó thì người chơi sẽ thua. Tuy có lối chơi vô cùng đơn giản, nhưng với sự mới lạ, hấp dẫn, game rắn săn mồi đã làm mưa làm gió trên toàn cầu và có thể coi đây chính là một trong những tựa game huyền thoại của làng game thế giới.

1.2 Công cụ sử dụng

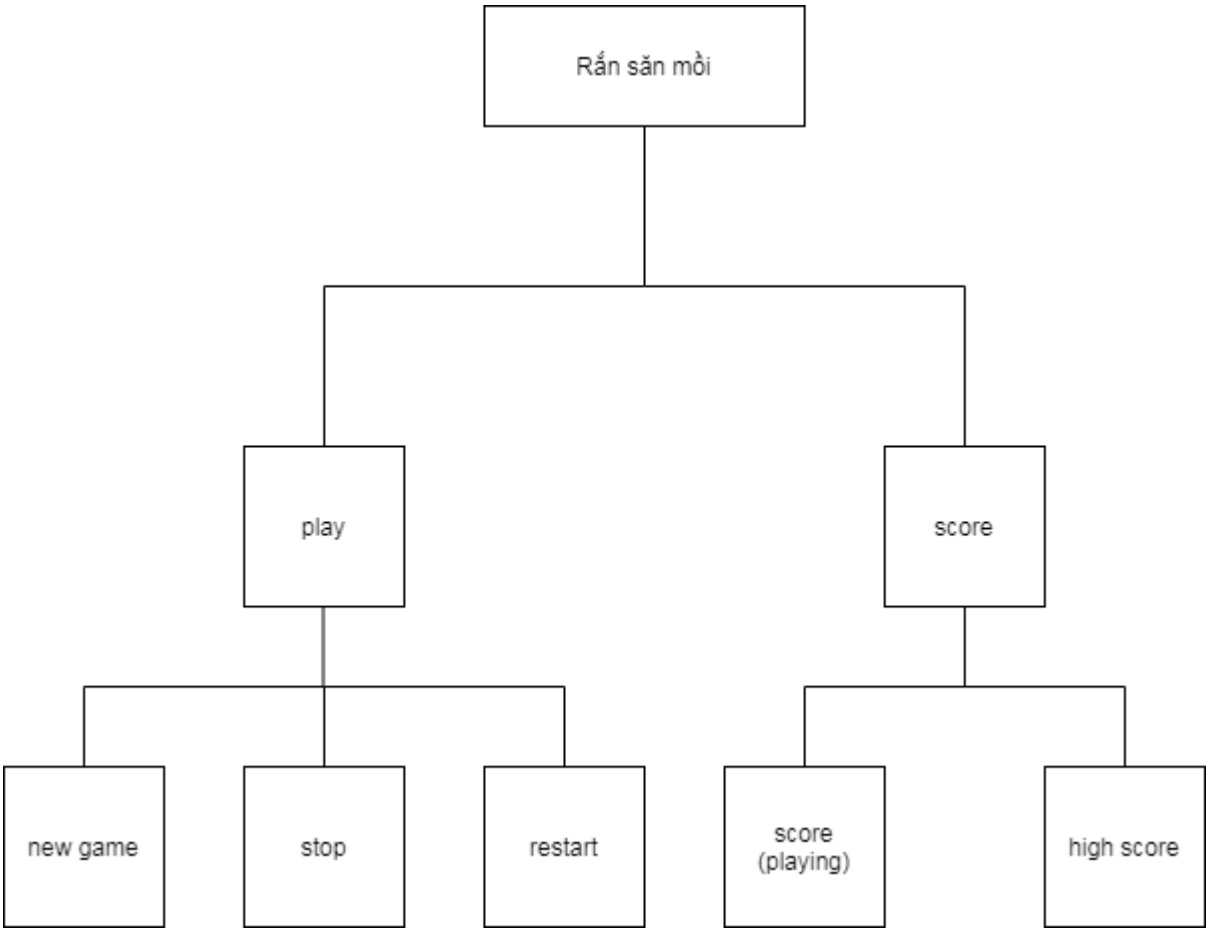
- Phần mềm Quartus II.
- Ngôn ngữ Verilog HDL
- Board FPGA DE2i-150.
- Màn hình VGA độ phân giải 640x480

1.3 Chức năng của sản phẩm

- Lợi ích với người chơi:

Trò chơi rắn săn mồi đã được thiết kế và xây dựng để có thể chơi được trên màn hình VGA.Dùng để giải trí sau những giờ học, giờ làm việc căng thẳng. Bên cạnh đó giúp tăng khả năng tập trung, phản xạ đối với người chơi, giúp tinh thần và trí tuệ của người chơi được tốt hơn.

- Sơ đồ phân rã chức năng (hướng tới khi hoàn thiện):

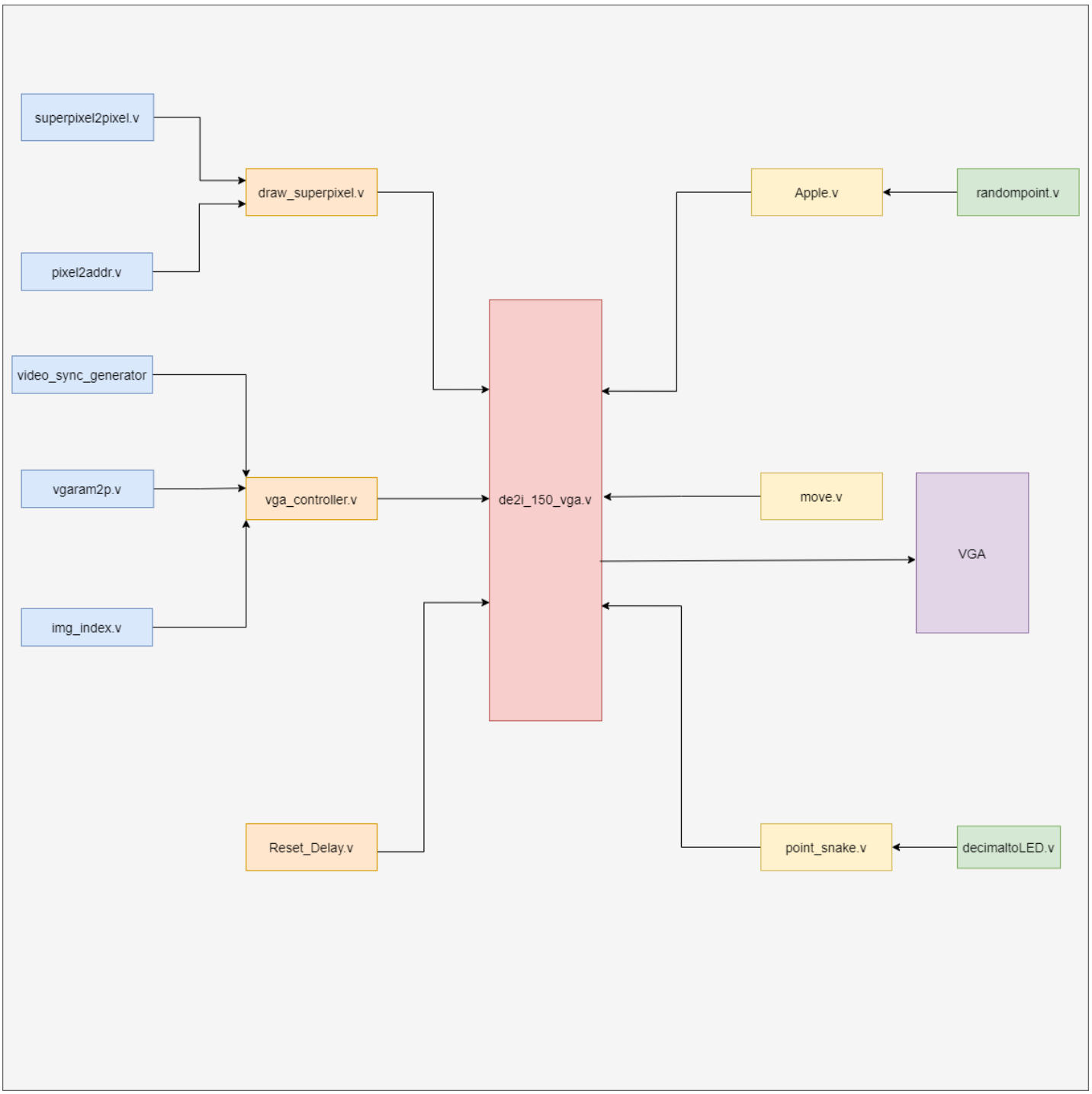


- Mô tả chức năng:
  - play: Đây là chức năng chính của chương trình
    - new game: Bắt đầu một game mới (level 1), điểm về 0 và tính lại từ đầu
    - stop: Cho dừng game tại thời điểm chơi
    - restart: Bắt đầu lại màn mới, điểm giữ nguyên từ màn trước

- score:
  - score(playing): Hiển thị điểm của người chơi hiện có, được hiển thị trên LED
  - high score: Điểm cao nhất người chơi có thể đạt được, cũng được hiển thị trên đèn LED

2 Thiết kế đề tài

2.1 Sơ đồ khối



2.2 Các khối và chức năng chính

Khối	Chức năng
Apple	Tạo mồi cho rắn
Move	Điều khiển cách di chuyển của rắn
randompoint	Chọn vị trí xuất hiện của mồi
de2i_150_vga	Điều khiển việc hiển thị trên VGA
point_snake	Hiển thị điểm của người chơi
Chưa hoàn thành(Bổ sung sau)	Hiển thị người chơi thắng khi độ dài rắn đạt 200
Chưa hoàn thành(Bổ sung sau)	Hiển thị người chơi thua khi rắn va vào thân

3 Hiện thực đề tài

3.1 Ý tưởng chính

- Người chơi điều khiển rắn bằng 4 nút trên board FPGA DE2i-150
- Rắn sẽ chết nếu như rắn đâm vào chính nó, lúc này sẽ hiện thị báo rằng người chơi đã thua
- Tính điểm cho người chơi, điểm được tính dựa vào số lượng mồi mà rắn ăn được (Số điểm của người chơi sẽ được hiện thị trên board)
- Hiển thị người chơi thắng nếu số điểm đạt được là 200
- Luật điều khiển được áp dụng như sau: Rắn không thể quay đầu ngược lại hướng đang di chuyển, tức là nếu đang tiến lên thì khi nhấn nút lùi rắn sẽ đâm vào thân, tương tự khi răng đang sang trái, phải, hoặc đi xuống.
- Mồi ăn sẽ được tạo ra ngẫu nhiên trên bản đồ, mồi sẽ được tạo mới khi rắn ăn được mồi. Trong trường hợp mồi được tạo lại trùng với thân rắn, một mồi mới sẽ được tạo lại ngay sau đó

3.2 Hiện thực các khối chức năng

3.2.1 Khối move

- Input:

- clk: xung clock 50MHz
- rst: tín hiệu reset
- vld: tín hiệu bắt đầu
- way: hướng đi của snake điều khiển bởi KEY
- length: số đốt rắn
- pixel_done: tín hiệu vẽ xong một superpixel

- Output:

- x, y: tọa độ hiện tại của rắn

- is_end: tín hiệu gắn tọa độ old của đốt cuối cùng
- is_queue: tín hiệu rắn đang di chuyển nối đuôi
- bite_self: tín hiệu rắn tự va chạm
- vld_t: tín hiệu gán các đốt kế tiếp

Ngoài ra còn có biến x_logic, y_logic, oldx_logic, oldy_logic là mảng gồm 201 phần tử, mỗi phần từ có độ rộng 5-bit sẽ lưu tọa độ hiện tại và tọa độ cũ của rắn. Đốt rắn được đánh chỉ số từ 0 đến length - 1

- Giải thích ý tưởng:

Khối move sẽ quyết định tọa độ để gán cho x, y. Nếu is_end=0 thì x, y sẽ là tọa độ của các đốt rắn, nếu is_end=0 sẽ là tọa độ old của đốt cuối cùng Rắn có thể di chuyển lên, xuống, trái, phải. Nếu rắn đi chuyển đến biên thì sẽ đi qua biên đối diện. Biến oldway sẽ lưu lại giá trị của way trước đó để đảm bảo người điều khiển rắn không di chuyển rắn ngược với hướng hiện tại
Khi vld HIGH, tọa độ cúa đốt đầu rắn sẽ được cập nhật, và output cũng là tọa độc của đầu rắn
Sau đó, lần lượt vld_t_reg sẽ HIGH sau khi đã vẽ xong đốt trước đó. Tại đây tọa độ hiện tại và tọa độ của của đốt thứ i sẽ được cập nhật. Đồng thời is_end, is_queue và biến đếm i cũng được cập nhật.
Đối với biến bite_self_reg, x_logic[0] và y_logic[0] sẽ được so sánh lần lượt với đốt rắn thứ i, và điều kiện length> 4 và i khác length vld_t và bite_self sẽ được gán riêng để xuất ra output

3.2.2 Khối Apple

- Input:

- clk: xung clock 50MHz
- rst: tín hiệu reset
- x_snake_cur, y_snake_cur: tọa độ của rắn hiện tại
- length: số đốt rắn
- is_end: tín hiệu gắn tọa độ old của đốt cuối cùng
- pixel_done: tín hiệu vẽ xong một superpixel
- vld: tín hiệu bắt đầu
- vld_start: tín hiệu bắt đầu sau vld 1 chu kì clk

- Output:

- is_eat: tínhh hiệu rắn đã ăn mồi
- appleX, appleY: tọa độ của mồi apple
- good: tín hiệu tọa độ của apple hợp lệ

- Giải thích ý tưởng:

Khối Apple sẽ xuất ra tọa độ của mồi apple đồng thời kiểm tra rắn có ăn được mồi hay không
Đầu tiên sẽ tạo 2 xung clk1, clk2 với tần số khác nhau, truyền vào khối randompoint để lấy rand_X và rand_Y là tạo tọa độ ngẫu nhiên. Ban đầu tọa độ của mồi apple được gán giá trị mặc định và nó được làm mới khi rắn ăn được mồi hoặc tọa độ mới của mồi trùng với thân rắn
Trong khối Apple sẽ có 2 biến để lưu tọa độ của rắn tương tự như move là x_snake và y_snake
Ta sẽ gán tọa độ của tất cả các đốt vào x_snake và y_snake. Khi có cạnh lên của tín hiệu bắt đầu vld_start hoặc tín hiệu vld_t, tọa độ của đốt thứ i sẽ được gán tương ứng với tọa độ hiện tại được truyền vào x_snake_cur và y_snake_cur với điều kiện is_end=0.
vld_check là HIGH khi tọa độ old của đốt cuối cùng của rắn được vẽ, khoảng thời gian vld_check HIGH là lúc kiểm tra điều kiện của tọa độ apple

Có 2 điều kiện để kiểm tra:

- is_eat: đầu của rắn có trùng với apple hay không bằng cách so sánh x_snake[0], y_snake[0] và appleX, appleY
- bad_collsion: apple có trùng với thân rắn hay không bằng cách cho vòng lặp for kiểm tra từ đầu đấn đốt cuối của rắn x_snake[i], y_snake[i] có bằng với appleX, appleY mới rồi gán vào từng bit của collision. Gán bad_collision bằng bitwise OR của collision.

Biến good là điều kiện để gán tọa độ của apple vào pixel_x_logic và pixel_y_logic trong module chính de2i_150_vga, good sẽ HIGH khi điểm mới tạo không trùng với thân rắn hoặc khi rắn vẫn chưa ăn được mồi

3.2.3 Khối randompoint

- Input:

Gồm clk1, clk2 là 2 xung clock khác nhau

- clk1: xung clock 50MHZ
- clk2: xung clock có giá trị nhỏ hơn 50MHz

- Output:

Gồm randomX, randomY là các tọa dộ của con rắn

- randomX: tọa độ mồi của rắn theo chiều ngang
- randomY: tọa độ mồi của rắn theo chiều dọc

- Giải thích ý tưởng:

Để tạo một tọa độ ngẫu nhiên cho mồi, ta dùng 2 bộ đếm xoay vòng với một xung clock có giá trị nhỏ, xung clock có giá trị lớn (Bộ đếm 1 với xung clock có giá trị 50MHz và bộ đếm 2 với giá trị xung clock nhỏ hơn 50 MHz)
Bộ đếm 1 được sử dụng để tạo ra giá trị tọa độ x(randomX) theo phương ngang của mồi, giá trị của bộ đếm này sẽ được lấy ra phụ thuộc vào các tín hiệu từ khối Apple.

Bộ đếm 2 sử dụng để tạo ra giá trị tọa độ y(randomY) theo phương thẳng đứng của mồi, giá trị này cũng được lấy ra phụ thuộc vào các tín hiệu của khối Apple
Kết hợp 2 giá trị này ta sẽ có được tọa độ của mồi mới được sinh ra, và có vẻ gần như là ngẫu nhiên.

3.2.4 Khối draw_superpixel

- Input:

  - clk: Xung clock 50 MHZ
  - rst: tín hiệu reset
  - x: tọa đọ x logic
  - y: tọa độ y logic
  - idata: giá trị màu 8 bit của pixel
  - idata_vld: tín hiệu cho phép truyền tọa độ logic của superpixel vào

- Output:

  - odone: tín hiệu đã vẽ xong 1 superpixel
  - oaddr: là tọa độ vật lý trên màn hình
  - odata: giá trị màu 8-bit của pixel
  - owren: tín hiệu cho biết data đang được ghi vào addr

- Giải thích ý tưởng:

Khối draw_superpixel dùng để vẽ 1 superpixel bất kì tại tọa độ logic được truyền vào Tọa độ Top-Left và Bottom-Right được lấy từ khối superpixel2pixel. Tọa độ vật lý được chuyển từ tọa độ logic sẽ lấy từ khối pixel2addr

- Khi vld_start HIG, tọa độ Top-Left được truyền vào đầu tiên
- Sau đó run sẽ HIGH bằng khoảng thời gian 1 superpixel được vẽ
- Cứ mỗi xung lên của clk thì 1 điểm ảnh với tọa độ vật lý trên màn hình được vẽ, từ Top-Left cho đến Bottom-Right.
- Khi điểm ảnh ở Bottom-Right được vẽ xong hay 1 superpixel đã hoàn thành, tính hiệu done sẽ HIGH trong 1 chu kỳ

3.2.5 Khối superpixel2pixel

- Input:

  - x, y: tọa độ logic của superpixel

- Output:

  - tlx, tly: tọa độ TOP-LEFT của superpixel
  - brx. Bry; tọa độ BOTTOM-RIGHT của superpixel

- Giải thích ý tưởng:

SPIXEL_PHY = PIXEL_X_MAX / SPIXEL_X_MAX = 640/32 = 20

-> cứ một ô vuông 20x20 pixel thì ta sẽ có 1 superpixel

Suy ra:

- TL = TĐ * SPIXEL_PHY = 20*TĐ
- BR = TL + (SPIXEL_PHY - 1) = TL+19

3.2.6 Khối pixel2addr

- Input:
  - x: Tọa độ vật lí theo phương ngang
  - y: Tọa độ vật lí theo phương thẳng đứng
- Output:
  - adrr : Địa chỉ của điểm pixel
- Giải thích ý tưởng:

Module sẽ chuyển đổi tọa độ x,y vật lí thành địa chỉ dựa trên công thức:

$$\text{addr} = y * (H\_PHY\_MAX + 10'd1) + x + 19'd2$$

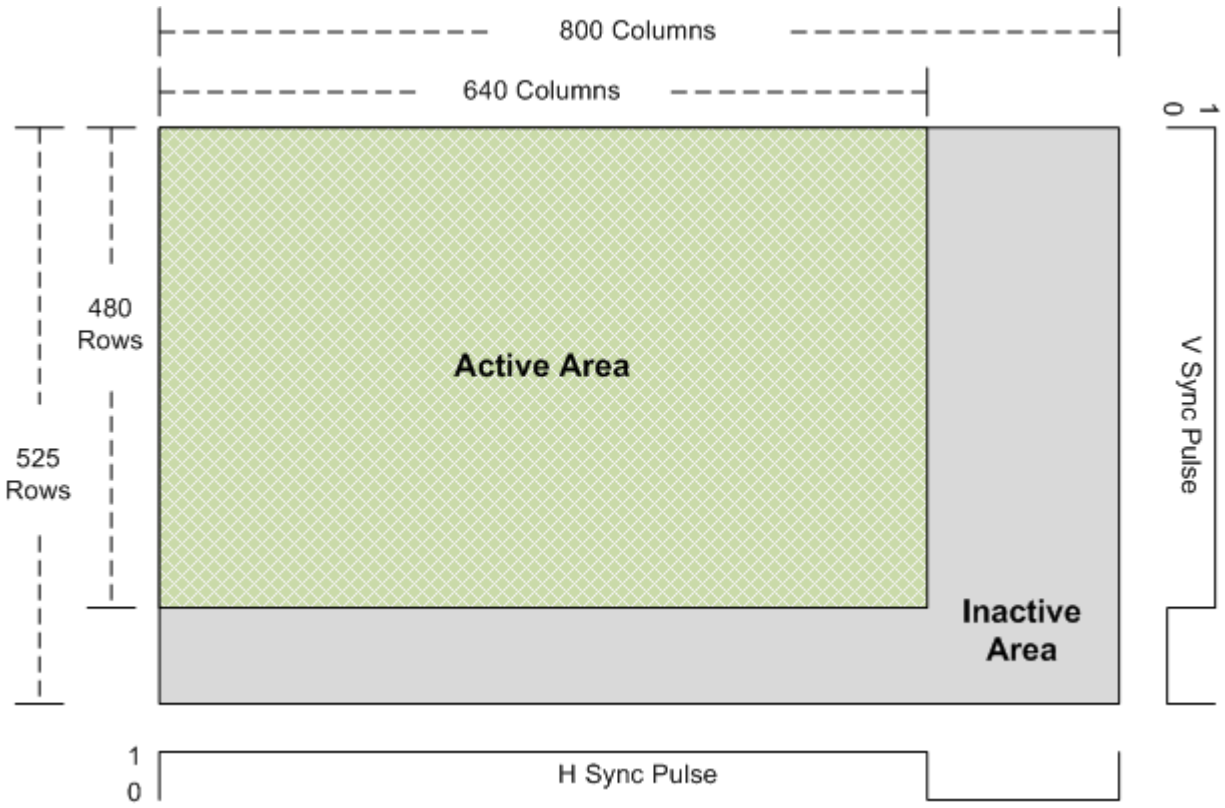
3.2.7 Khối Reset_Delay

- Input:
  - iCLK: xung clock 50MHz
  - iRST_n : tín hiệu reset
- Output:
  - oRESET
- Giải thích ý tưởng:

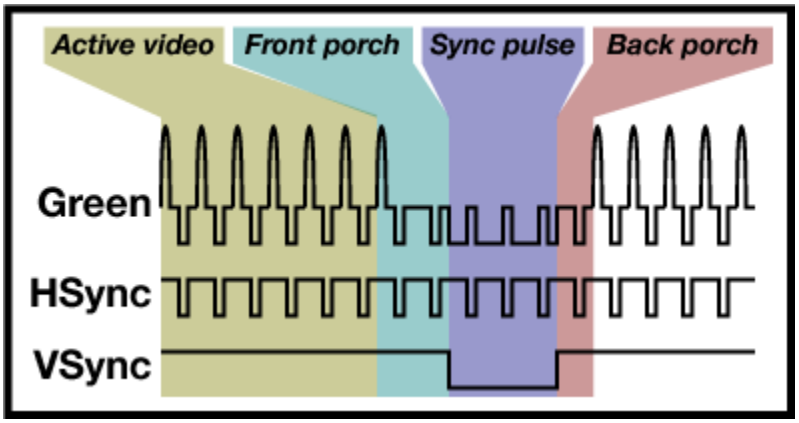
Tạo biến đếm Cont 20-bit, khi Cont khác giá trị lớn nhất của nó là 20'hFFFFF thì oRESET = 0
Chỉ cho phép gán giá trị của oRESET theo giá trị đầu vào của công tắc chỉ khi Cont = 20'hFFFFF

3.2.8 Khối vga_controller_mod

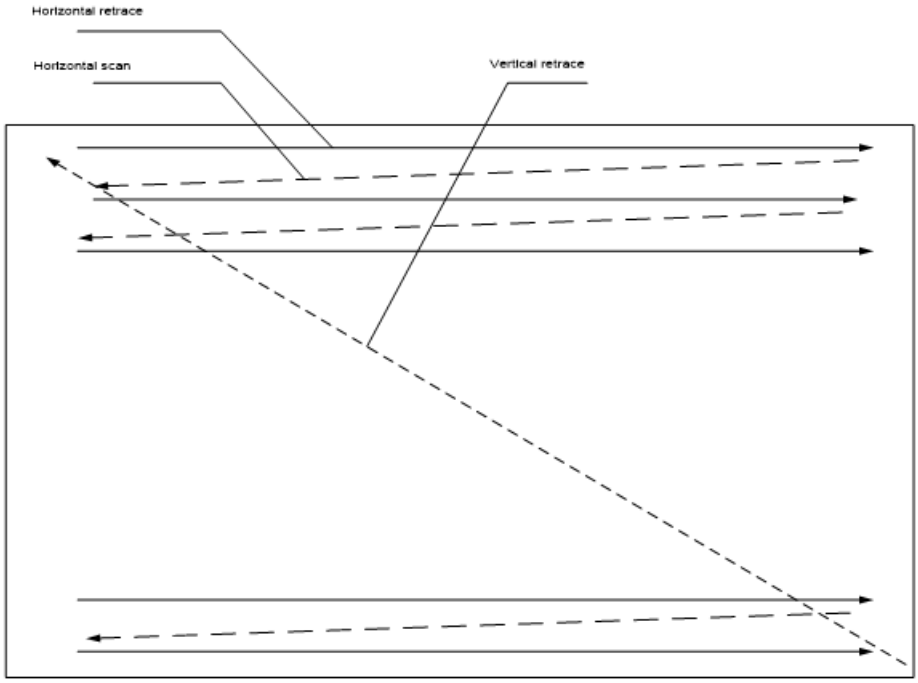
Tóm tắt về cách hoạt động của màn hình VGA



Trên màn hình VGA có kích thước là 800x525 nhưng thực tế chỉ hiển thị được 640x480, do những phần còn lại bị ẩn đi là vùng biên và vùng electron quay trở lại. Vì vậy ta chỉ ta chỉ write dữ liệu khi electron nằm trong vùng hiển thị



Khi màn hình VGA hoạt động, chùm electron sẽ di chuyển từ trái sang phải, từ trên xuống dưới, khi electron tới đáy màn hình thì sẽ quay lại đỉnh màn hình



Trong module vga_controller_mod
- Input:

- iRST_n: tín hiệu reset
- iVGA_CLK: xung clock 25MHz
- iclk: xung clock 50MHz
- iwren: tín hiệu cho biết data đang được ghi vào addr
- idata : là giá trị màu của pixel
- iaddr: giá trị của địa chỉ vật lý trên màn hình

- Output:

- oBLANK_n: tín hiệu cho biết bộ đếm ngang và dọc có nằm trong vùng hiển thị không
- oHS: tín hiệu bộ đếm ngang bắt đầu vào vùng hiển thị
- oVS: tín hiệu bộ đếm dọc bắt đầu vào vùng hiển thị
- b_data, g_data, r_data: dữ liệu màu 24-bit

- Giải thích ý tưởng:

Khối vga_controller_mod sẽ gọi các khối:

- video_sync_generator: để lấy các tín hiệu từ bộ đếm ngang và bộ đếm dọc là CBLANK_n, CHS, CVS
- vgaram2p: truyền vào mã màu 8-bit, và địa chỉ vào RAM để xuất ra địa chỉ đang đọc (ADDR) và index của màu 24-bit (index)

- img_index: sẽ lấy index màu 24-bit sau đó lấy ra mã màu 24-bit để xuất ra màn hình

Các biến khác như: delay_bus, delay_busv, delay_bush dùng để tạo delay bằng phép dịch bit qua trái khi lần lượt cBLANK_n = 1, CHS = 0, CVS = 0

3.2.9 Khối TOP de2i_150_vga

- Input:

- CLOCK_50: xung clock 50MHz lấy từ board de2i_150
- SW: được kết nới với công tắc dùng để reset tín hiệu
- KEY: 4-bit tương ứng 4 button trên board

- Output:

- VGA_B, VGA_G, VGA_R: data màu 24-bit
- VGA_BLANK_N: lấy từ oBLANK_n
- VGA_CLK: xung clock 25MHz
- VGA_HS: lấy từ oHS
- VGA_VS: lấy từ oVS
- LED1: led 7 đoạn hàng đơn vị
- LED2: led 7 đoạn hàng đơn chục
- LED3: led 7 đoạn hàng đơn trăm

- Giải thích ý tưởng: Module de2i_150_vga sẽ là khối chính để điều khiển trò chơi, là khối trung gian để truyền dữ liệu giữa các khối: vga_controller_mod, move, Apple và draw_superpixel Trong khối gồm một số biến quan trọng khác như:

- vld: tín hiệu bắt đầu để vẽ, mỗi 1 giây vld sẽ HIGH đúng 1 chu kì clk 50MHz, tức cách mỗi 1 giây sẽ cập nhật tất cả giá trị
- vld_start: giống với vld nhưng sau vld HIGH 1 chu kì clk,
- rst: tín hiệu reset, reset sẽ HIGH khi đầu rắn va chạm với chính nó, nếu không thì vẫn hoạt động theo công tắc SW
- pixel_vld: là tín hiệu cho phép vẽ lên VGA

Khi người dùng dùng button trên board để điều khiển rắn, move sẽ cập nhật tọa độ của rắn là x_logic và y_logic, đồng thời tọa độ của mồi là appleX, appleY cũng đã được cập nhật

- pixel_x_logic, pixel_y_logic: là tọa độ sẽ được truyền vào khối draw_superpixel để vẽ một superpixel, tọa độ đó là của rắn hay của mồi còn tùy thuộc vào tín hiệu vld_apple
- pixel_color: là màu của superpixel sẽ truyền vào khối draw_superpixel, nếu vld_apple=1 -> màu đỏ, khi vld_apple=0 và (is_end && !is_queue)=1 tức đã vẽ đốt cuối cùng và rắn di chuyển không nối đuôi -> màu trắng là màu nền, nếu không thỏa 2 điều kiện trên -> màu xanh là màu của rắn
- Mỗi khi rắn ăn được táo, length sẽ tăng thêm 1, đồng thời điểm cũng sẽ tăng theo length và xuất ra led 7 đoạn trên board.

3.2.10 Khối point_snake

Trong module point_snake gồm có:
-Input:

- clock: xung clock 50MHz
- reset: Tín hiệu reset
- length: Số đốt rắn

-Output

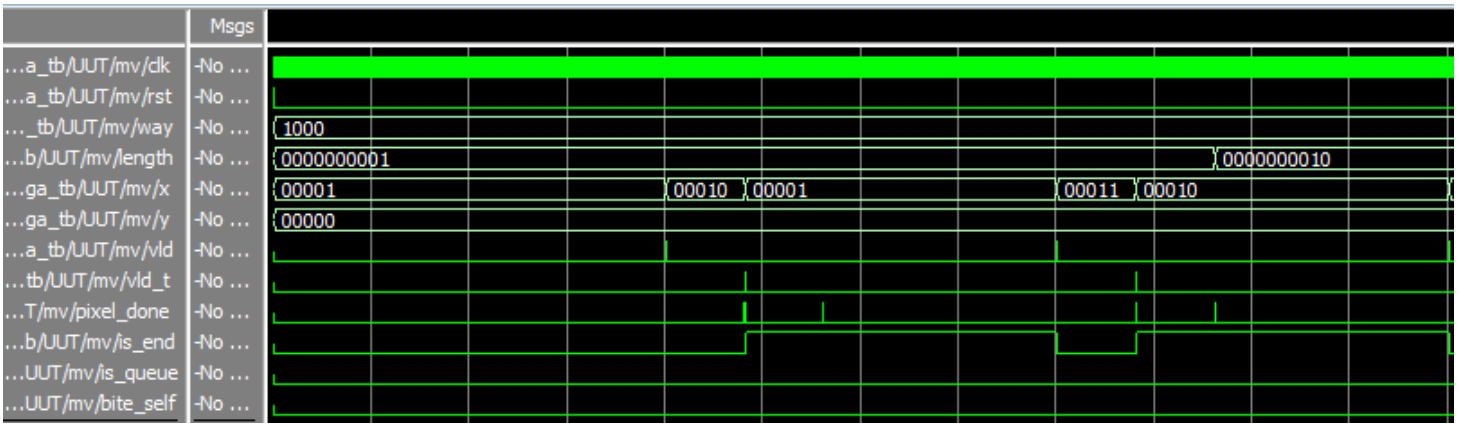
- LED1, LED2, LED3: điểm hiện tại của người chơi và xuất ra LED

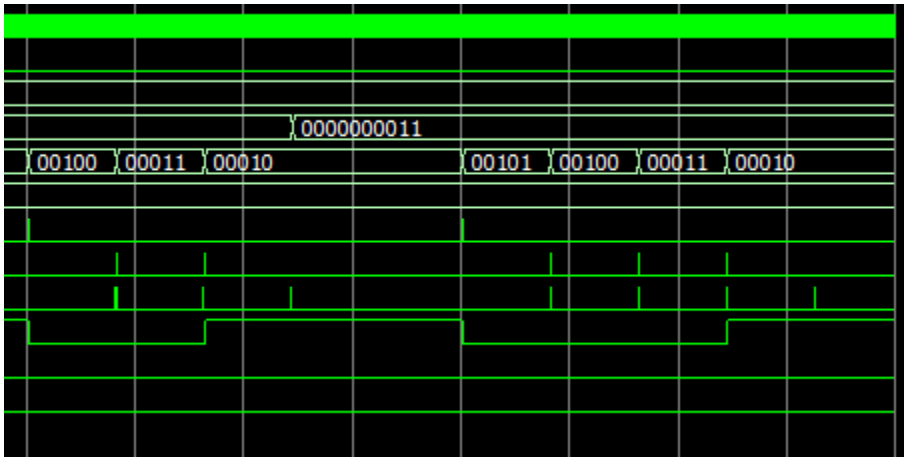
-Giải thích ý tưởng: Khối point_snake sẽ sử dụng length của module chính de2i_150_vga để xuất ra điểm của người chơi.
Khối point_snake gọi 3 reg là first, second, third có nhiệm vụ lưu giá trị của biến length ở hàng đơn vị, hàng chục và hàng trăm và khối point_snake cũng đồng thời gọi module decimaltoLED có nhiệm vụ biến đổi thành tín hiệu LED1, LED2, LED3 tại ba đèn LED của hàng đơn vị, hàng chục, hàng trăm thông qua output LED.

3.3 Mô phỏng một số chức năng

3.3.1 Rắn di chuyển

Mô phỏng có độ dài length của rắn lần lượt là 1, 2, 3

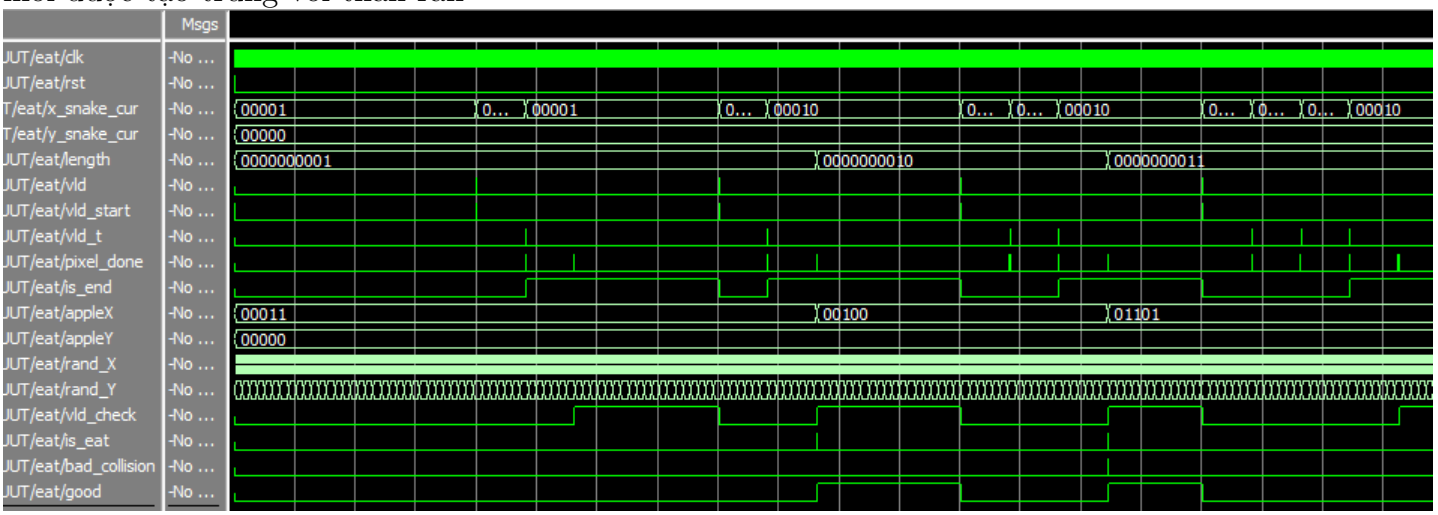




Đầu tiên, để rst lên mức HIGH trong 3 chu kì xung clk để làm mới tín hiệu (khí rst rắn sẽ nằm ở tọa độ (1;0))
Khi vld HIGH 1 chu kì clk thì tọa độ hiện tại của đầu rắn sẽ được gắn lại cho oldx_logic[0] và oldy)logic[0], sau đó đầu rắn là x_logic[0] và y_logic[0] sẽ thay đổi theo input way
Tiếp theo, sẽ gán output x, y tương ứng với x_logic[0] và y_logic[0] Tiếp theo biến vld_t sẽ HIGH 1 chu kì clk để cập nhật tọa độ của các đốt còn lại, vld_t HIGH cuối cùng sẽ gàn cho output x, y tọa độ cũ của đốt cuối cùng. Cùng lúc đó is_end sẽ HIGH

3.3.2 Tạo mồi (apple) cho rắn

Mô phỏng sẽ chia 3 trường hợp: rắn chưa ăn mồi, rắn ăn được mồi, rắn ăn được mồi và mồi mới được tạo trùng với thân rắn

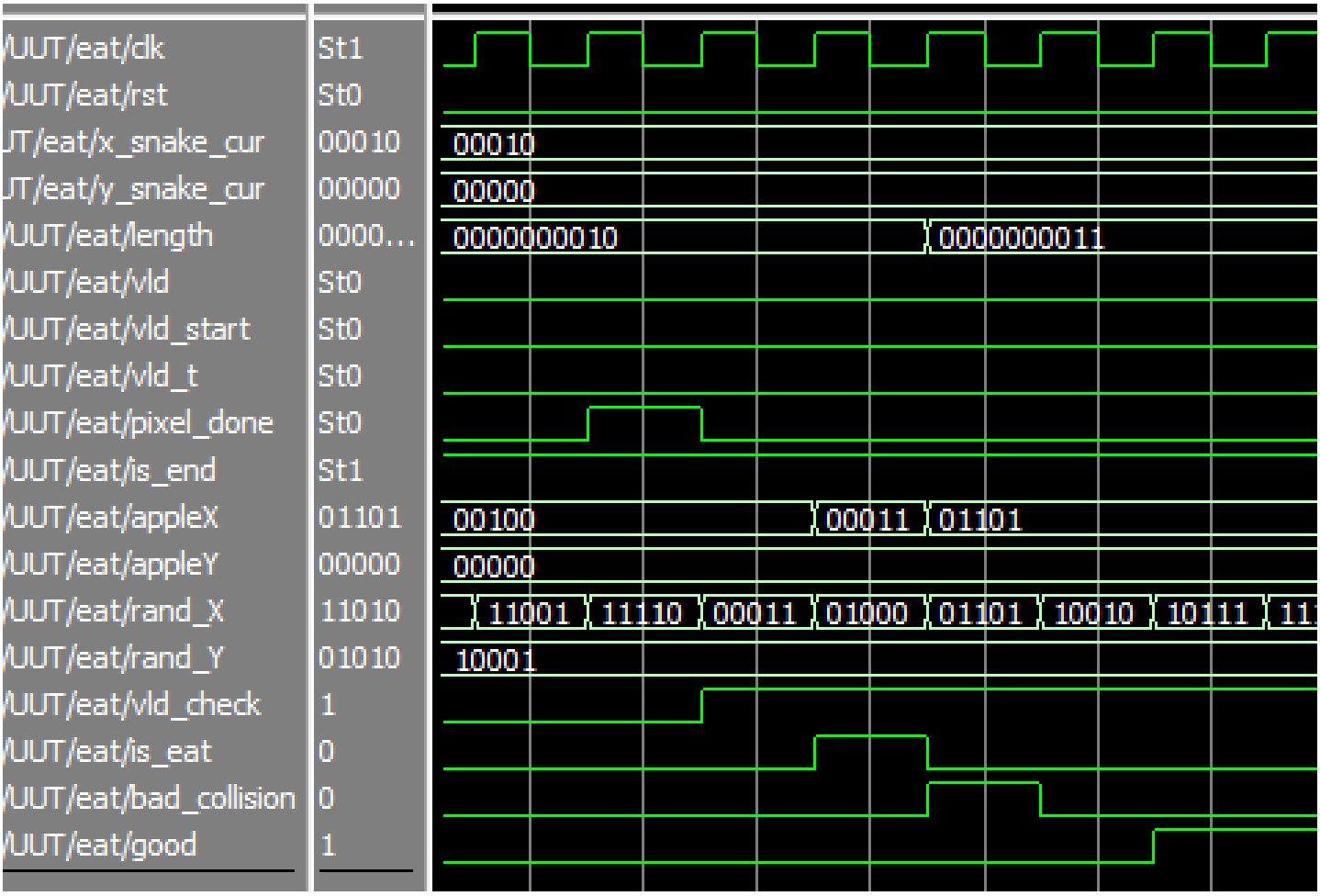


Đầu tiên, để rst lên mức HIGH trong 3 chu kì xung clk để làm mới tín hiệu (khí rst rắn sẽ nằm ở tọa độ (1;0))
Khi xung clk hoạt động, tọa độ logic hiện tại là đốt rắn thứ i (x_snake_cur và y_snake_cur) lấy từ module move, khi có cạnh lên của vld_start hoặc vld thì tọa độ đốt rắn thứ i sẽ được gán cho x_snake[i] và y_snake[i].
Cùng lúc đó, tọa độ của apple là appleX và appleY khóa giá trị ứng với rand_X và rand_Y (trong hình trên tọa độ logic apple được gán lần lượt là (3;0), (4:0), (3;0) để thuận tiện cho

mô phỏng)
Sau đó, khi tín hiệu is_end và pixel_done cùng ở mức HIGH (tọa độ cũ đốt cuối cùng được vẽ), vld_check HIGH để bắt đầu kiểm tra điều kiện is_eat và bad_collision

Khi is_eat HIGH lần 1 như hình trên, tọa độ mới của apple sẽ được cập nhật theo rand_X và rand_Y là (3;0) -> (4,0). Ngay sau đó 1 chu kì clk, bad_collision sẽ được cập nhật theo phép toán bitwise OR của collision là bằng 0. Đồng thời, length cũng sẽ tăng thêm 1

Tương tự như vậy, khi is_eat HIGH lần 2, trường hợp này tọa độ của apple trùng với đốt thứ 1 của rắn, nên sau khi is_eat HIGH 1 chu kì clk thì bad_collision sẽ HIGH. Cùng lúc đó, tọa độ mới của apple được cập nhật và tiếp tục được kiểm tra. Tọa độ mới là (13;0) không trùng với thân rắn nên bad_collision về LOW. Sau đó 1 chu kì clk, good sẽ HIGH và về LOW cho đến khi vld HIGH



4 Kết luận

4.1 Lời kết

Qua quá trình tìm hiểu về bài tập lớn, nhóm chúng em đã có điều kiện được ôn tập lại những kiến thức đã học ở trên lớp đồng thời có cơ hội để tiếp thu những kiến thức mới về VGA, nâng cao khả năng sáng tạo trong việc lên ý tưởng, cải thiện khả năng làm việc nhóm của mỗi cá

nhân.

Thông qua việc lên ý tưởng và hiện thực, mỗi thành viên trong nhóm đã có những hiểu biết cơ bản về cách sử dụng VGA, đã hiểu hơn về cách thức làm việc nhóm , cùng nhau phối hợp để đưa ra sản phẩm cuối cùng ăn ý nhất. Đồng thời bài tập này đã giúp chúng em rèn luyện được kỹ năng nghiên cứu , tự học, nâng cao khả năng tư duy và khả năng sử dụng ngôn ngữ đặc tả phần cứng verilog HDL để thiết kế mạch logic số.

Lời cuối cùng, chúng em xin chân thành cảm ơn thầy cô đã tạo điều kiện cho chúng em có cơ hội được học tập, nghiên cứu và tiếp cận kiến thức mới một cách chủ động và sáng tạo. Chúng em cũng chân thành cảm ơn sự giúp đỡ tận tình của các giáo viên hướng dẫn trong suốt quá trình thực hiện nghiên cứu, nhờ có những sự giúp sức đó mà nhóm chúng em có thể vượt qua được những khó khăn và đưa ra được sản phẩm cuối cùng hoàn chỉnh

4.2 Hướng phát triển trong tương lai

Qua việc tìm hiểu về VGA và phát triển trò chơi rắn săn mồi, nhóm chúng em đã tìm thấy sự hứng thú trong việc ứng dụng những kiến thức được học trên lớp vào thực tế. Vì vậy trong tương lai mỗi thành viên trong nhóm sẽ cố gắng học tập, trau dồi từng bước nâng cao khả năng sử dụng ngôn ngữ verilog HDL của bản thân, mở rộng kiến thức và hiểu rõ về VGA và ứng dụng của nó cũng như về các chức năng khác của board FPGA. Nhóm sẽ cố gắng tìm hiểu thêm về những đề tài khác và nếu có điều kiện thì nhóm sẽ tiến hành nghiên cứu để thực hiện, phát triển và nâng cao khả năng tư duy phần cứng, khả năng hiện thực những ý tưởng và phát triển khả năng sáng tạo của bản thân.

4.3 Khó khăn gặp phải khi thực hiện đề tài

- Khó khăn trong việc hiện thực các ý tưởng thiết kế và các khối chức năng.
- Gặp trở ngại trong việc tìm hiểu và tiếp xúc với VGA.
- Khó khăn trong việc phân bố thời gian lên ý tưởng và hiện thực sản phẩm.
- Gặp lúng túng trong việc tìm kiếm lỗi sai và sửa chữa.

4.4 Phân chia công việc của từng thành viên

Ung Ngô Minh Lăng: khối move, de2i_150_vga, apple, đánh word, mô phỏng move và apple

Trần Văn Kiên: báo cáo latex, khối point_snake, vẽ sơ đồ phân rã chức năng

Nguyễn Huy Hoàng: khối randompoint, khối apple, viết báo cáo bằng latex, vẽ sơ đồ khối