ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

PHẠM ĐỨC HIẾU

CÔNG NGHỆ LẬP TRÌNH FPGA VÀ ỨNG DỤNG XỬ LÝ DỮ LIỆU ĐA PHƯƠNG TIỆN

Ngành: Công nghệ thông tin
Chuyên ngành: Kỹ thuật phần mềm
Mã số: 6048013

NGƯỜI HƯỚNG DẪN KHOA HỌC: TS. LÊ QUANG MINH

Hà Nội 2016

LỜI MỞ ĐẦU

Trong những năm gần đây với sự phát triển của công nghệ bán dẫn trong việc tạo ra những bộ vi xử lý và vi điều khiển, các hệ thống đo lường và điều khiển ngày càng thông minh hơn, giải quyết được nhiều bài toán phức tạp hơn. Tuy nhiên các hệ thống ngày càng hoàn thiện hơn, thông minh hơn thì vi xử lý và vi điều khiển chưa thể đáp ứng hết nhu cầu. Sự xuất hiện các thiết bị có thể lập trình như FPGA (Filed Progammable Gate Array đã tạo ra bước đột phá. Với công nghệ FPGA đã có rất nhiều công trình nghiên cứu giải quyết các bài toán lớn như bài toán xử lý tín hiện số, mật mã và nhận dạng. Các nghiên cứu hiện nay chủ yếu đi theo hướng kết hợp các bộ FPGA với những bộ xử lý thông thường trong một chip tạo ra hệ thống mới RCS (reconfigurable Computing System). Đây là một mô hình mới trong thiết kế các hệ thống có khả năng tính toán rất mạnh, thay thế được cho những máy tính lớn. Không những thế, các thiết bị này có khả năng tái lập trình nên các hệ thống này có độ linh hoạt cao, có thể thay đổi lại cấu hình để đáp ứng nhiều thuật toán hay các yêu cầu khác nhau về phần cứng trong quá trình hệ thống đang hoạt động.

Cùng với sự bùng nổ của các mạng internet, mạng di động là các nhu cầu giải trí, truyền thông đa phương tiện. Để tăng chất lượng dịch vụ của các dịch vụ đa phương tiện thì việc xử lý các dữ liệu đa phương tiện là rất cần thiết. Hiện nay các giải pháp xử lý dữ liệu đa phương tiện đều có chi phí rất cao. Do vậy mục đích của nghiên cứu này là ứng dụng công nghệ FPGA vào việc xử lý dữ liệu đa phương tiện một cách hiệu quả.

Đối tượng và phạm vi nghiên cứu:

1. Công nghệ FPGA: các lĩnh vực ứng dụng, các công cụ phát triển

2. Kỹ thuật xử lý nhanh dữ liệu pipeline.

3. Ứng dụng kỹ thuật pipeline thiết kế lõi IP xử lý hình ảnh

Những nội dung chính: Nội dung của luận văn gồm phần đặt vấn đề, 3 chương, kết luận và tài liệu tham khảo.

Chương I của luận văn trình bày tổng quan về công nghệ FPGA, các lĩnh vực ứng dụng của công nghệ này và các công cụ phát triển, hỗ trợ lập trình trên FPGA.

Chương II của luận văn trình bày về kỹ thuật xử lý dữ liệu pipeline, cách thức tổ chức pipeline trong công nghệ FPGA và đánh giá hiệu quả của kỹ thuật này.

Chương III thực hiện thiết kế lõi IP xử lý hình ảnh cụ thể lõi IP sẽ xác định các điểm ảnh bị lỗi và sửa chúng, xác định màu bằng phương pháp nội suy, sửa ma trận màu.

Phần kết luận của luận văn trình bày các kết quả đạt được và những hạn chế của luận văn, hướng phát triển của luận văn trong các nghiên cứu tiếp theo.

1. TÔNG QUAN VỀ CÔNG NGHỆ FPGA

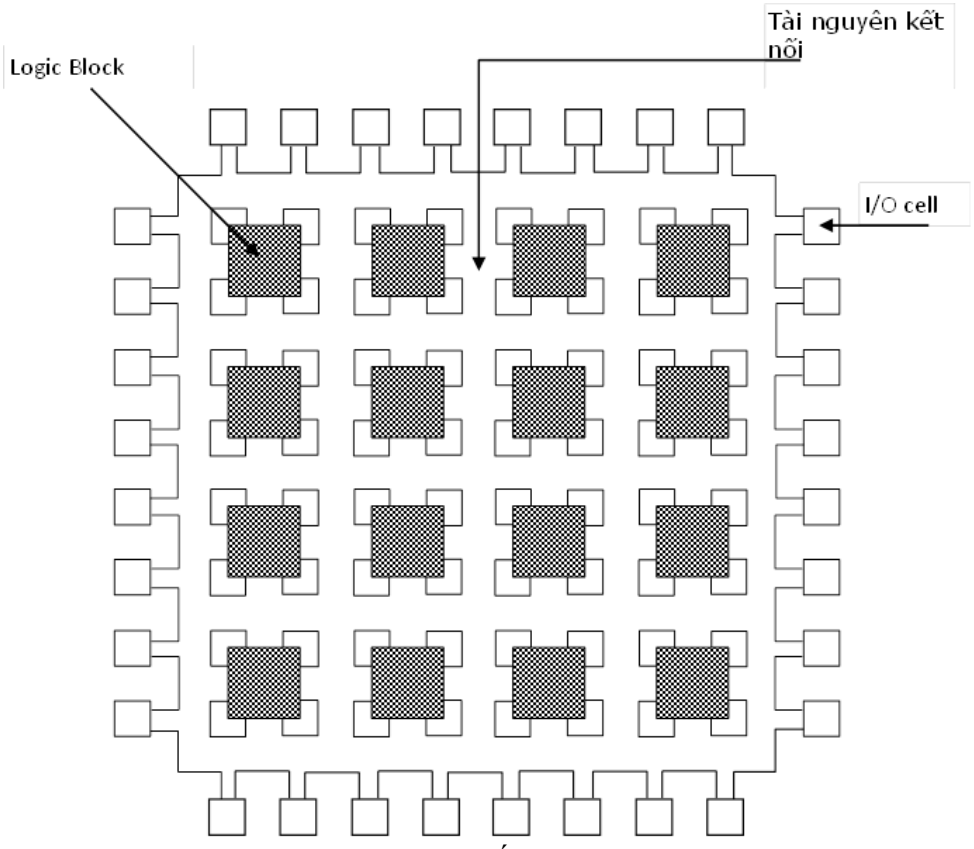
1.1. Lịch sử ra đời FPGA

Năm 1984 Ross Freeman là người đầu tiên thiết kế PFGA và cũng là người sáng lập công ty Xilinx. Kiến trúc mới của FPGA cho phép tích hợp số lượng lớn các phần tử bán dẫn vào một vi mạch so với kiến trúc trước đó là CPLD. FPGA có khả năng chứa từ 100.000 đến vài tỷ cổng logic, trong khi CPLD chỉ chứa từ 10.000 đến 100.000 cổng logic, con số này đối với PAL vá PLA còn thấp hơn rất nhiều chỉ đạt vài nghìn đến 10.000 cổng logic.

Kiến trúc của FPGA là kiến trúc mảng các khối logic, khối logic nhỏ hơn nhiều nếu đem so sánh với một khối SPLD, ưu điểm này giúp FPGA có thể chứa nhiều hơn các phần tử logic và phát huy tối đa khả năng lập trình của các phần tử logic và hệ thống mạch kết nối, để đạt được mục đích này thì kiến trúc của FPGA phức tạp hơn nhiều so với CPLD. Một điểm khác biệt với CPLD là trong những FPGA hiện đại được tích hợp nhiều những bộ logic số học đã sơ bộ tối ưu hóa hỗ trợ RAM, ROM tốc độ cao, hay các bộ nhân cộng (*multication anh accumulation, MAC*), thuật ngữ tiếng anh là DSP slice dùng cho những ứng dụng xử lý tín hiệu số DSP.

Ngoài khả năng tái cấu trúc vi mạch toàn cục, FPGA hiện tại còn hỗ trợ tái cấu trúc một bộ phận riêng lẻ trong khi vẫn đảm bảo hoạt động bình thường cho các bộ phận khác.

1.2. Cấu trúc FPGA



Hình 1-1. Cấu trúc FPGA

FPGA là mạch tích hợp chứa nhiều (64 đến hơn 10.000) ô logic (logic cell) giống nhau có thể xem là các thành phần chuẩn. Mỗi ô logic giữ một hay một số chức năng độc lập. Các ô giống nhau được kết nối bởi một ma trận đường dẫn và các chuyển mạch khả trình. Người thực hiện thiết kế bằng các đặc trưng lôgic đơn của mỗi ô và lựa chọn đóng các chuyển

mạch trong ma trận kết nối. Mảng của các ô lôgic và kiểu kết nối là kết cấu xây dựng khối cơ bản trong mạch lôgic. Các thiết kế phức tạp được tạo ra bằng cách kết hợp các

khối cơ bản để tạo ra các mạch được mô tả.

Mô hình tổng quát của FPGA gồm một dãy hai chiều các khối lôgic (*logic block*) có thể được kết nối bằng các nguồn kết nối chung. Các nguồn kết nối gồm các đoạn kết nối (*segment*) có thể có chiều dài khác nhau. Bên trong các kết nối là các chuyển mạch lập trình được dùng để nối các khối lôgic với các đoạn dây, các khối vào/ra hay các đoạn dây với nhau. Mạch lôgic cài đặt trong FPGA bằng cách ánh xạ lôgic vào các khối lôgic riêng rẽ và sau đó nối các khối lôgic cấu hình (*Configurable logic Block*) cần thiết qua các chuyển mạch. Các khối CLB cung cấp các phần tử chức năng với cấu trúc sử dụng logic. Các khối vào/ra (*I/O Block*) cung cấp giao diện giữa các gói chân và các đường tín hiệu bên trong. Tài nguyên kết nối khả trình cung cấp các bộ phận truyền dẫn tới kết nối đầu vào và đầu ra của các CLB và các IOB trong mạng riêng.

Vậy cấu trúc FPGA gồm ba phần tử chính: Các khối lôgic cấu hình (CLB), các khối vào/ra (IOB) và các kết nối.

a. Các khối lôgic cấu hình (*Configurable logic Block*)

Cấu trúc và nội dung của logic block được gọi theo kiến trúc của nó. Kiến trúc của khối lôgic có thể thiết kế theo nhiều cách khác nhau, có thể là các cổng AND 2 ngõ nhập, các bộ dồn kênh (*Multiplexer*) hay các bảng tìm kiếm (*Lock-up Table*). Ngoài ra có thể chứa các Flip-Flop để hỗ trợ cho việc thực hiện một cách tuần tự.

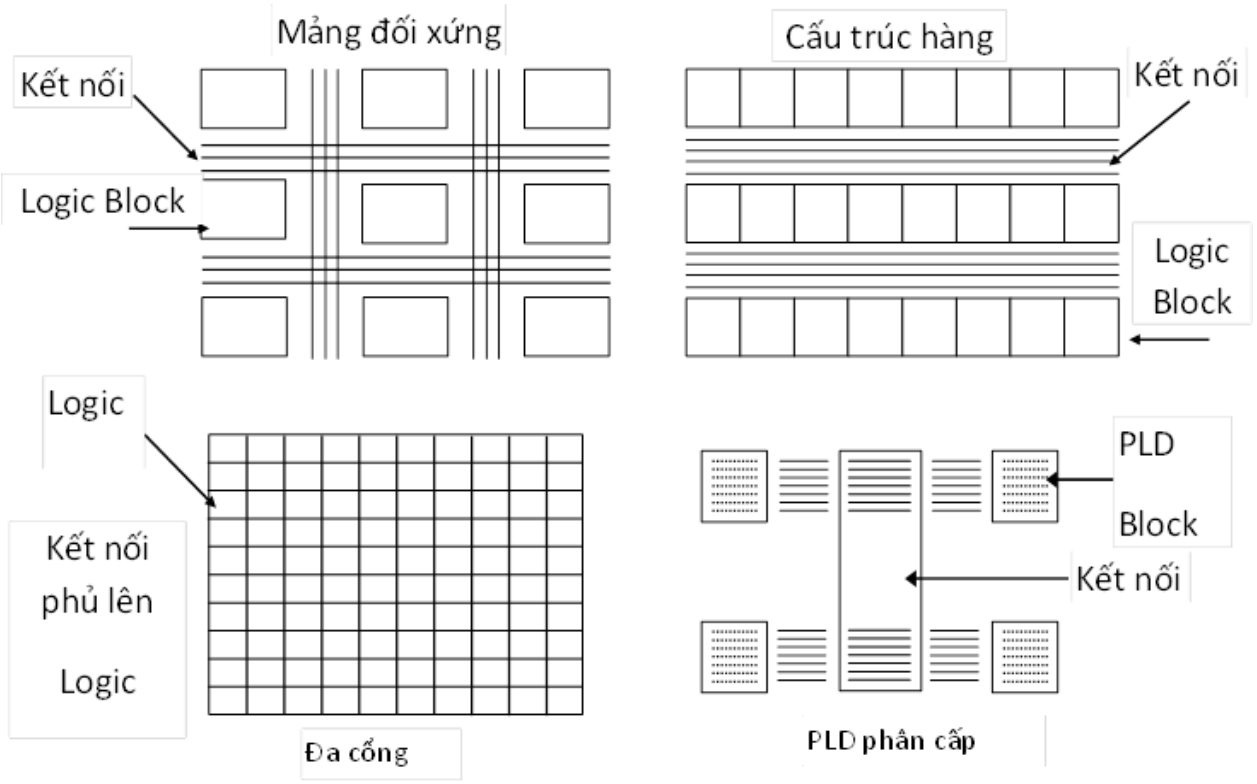
b. Các nguồn kết nối (Routes)

Các nguồn kết nối có cấu trúc và nội dung được gọi là kiến trúc đường

(*Routing Architecture*). Kiến trúc Routing gồm các đoạn đây nối và các chuyển mạch khả trình. Các chuyển mạch khả trình có cấu tạo khác nhau như pass-transistor, được điều khiển bởi các cell SRAM, các phần tử cầu chì nghịch, EPROM transistor và EEROM transitor. Giống như các khối lôgic có nhiều cách khác nhau để thiết kế các kiến trúc routing. Một số FPGA cung cấp nhiều kết nối đơn giản giữa các khối logic, một số khác cung cấp ít kết nối hơn nên routing phức tạp hơn.

1.2.1. Phân loại FPGA

FPGA có nhiều loại khác nhau có cấu trúc và đặc tính riêng tuỳ theo từng hãng sản xuất, tuy nhiên chúng có bốn loại chính sau: cấu trúc mảng đối xứng (*Symmetrical Array*), cấu trúc PLD phân cấp *(hierachircal PLD)*, cấu trúc hàng *(Row base)* và cấu trúc đa cổng *(Sea of Gate)* mô tả dưới đây.



Hình 1.2 Các loại cấu trúc FPGA

1.2.2. Các công nghệ lập trình FPGA

Có nhiều cách thực hiện các phần tử lập trình, các công nghệ lập trình hiện đang sử dụng là: RAM tĩnh, cầu chì nghịch (anti-fuse), EPROM transistor và EEROM transistor. Mặc dù công nghệ lập trình khác nhau, tất cả các phần tử lập trình đều có chung tính chất là có thể cấu hình được trong một trong hai trạng thái ON hoặc OFF. Các phần tử lập trình được dùng để thực hiện các kết nối lập trình được giữa các khối lôgic của các FPGA, còn FPGA thông thường có thể hơn 100.000 phần tử lập trình. Vì vậy các phần tử lập trình phải có những tính chất sau:

- Chiếm càng ít diện tích của chíp càng tốt.
- Có trở kháng thấp khi ở trạng thái ON và trở kháng cao khi ở trạng thái OFF.
- Có điện dung ký sinh thấp khi kết nối các đoạn dây.
- Có thể chế tạo một cách tin cậy số lượng lớn phần tử lập trình trên một chíp.

Có thể tuỳ thuộc vào ứng dụng cụ thể và có các số lượng phần tử lập trình có thể có các đặc tính khác. Về mặt chế tạo, các phần tử lập trình nếu có thể chế tạo theo công nghệ CMOS chuẩn là tốt nhất. Dưới đây sẽ trình bày chi tiết các công nghệ lập trình FPGA.

a. Công nghệ lập trình dùng RAM tĩnh

Công nghệ lập trình dùng RAM tĩnh (SRAM) sử dụng công nghệ CMOS tiêu chuẩn. Các kết nối lập trình được điều khiển bằng các transistor khác trên chíp hoặc bật (On) các transistor truyền dẫn cũng như các cổng transistor để tạo một kết nối hay tắt (Off) để ngắt kết nối.

Trong các FPGA sử dụng công nghệ lập trình SRAM, các khối logic có thể được kết hợp với nhau qua cách kết hợp cả bộ dồn kênh (*Multiplexer*) và cổng truyền dẫn (*pass-gate*). Vì SRAM là bộ nhớ bay hơi, các FPGA này phải được tái cấu hình mỗi khi cấp nguồn cho chíp. Điều này có nghĩa là hệ thống sử dụng các chíp này phải có một số cơ chế lưu trữ thường trực cho các bit của RAM Cell, chẳng hạn ROM hay đĩa từ. Các bit của RAM Cell có

thể được nạp vào FPGA một cách tuần tự hay định địa chỉ như một phần tử của mảng (theo cách thông thường của một RAM).

Các chíp được thực hiện theo công nghệ SRAM có diện tích khá lớn, bởi vì cần ít nhất 5 transistor cho mỗi RAM Cell cũng như các transistor cần thêm cho cổng truyền dẫn hay bộ dồn kênh. Ưu điểm của kỹ thuật này là cho phép FPGA có thể được tái cấu hình ngay trên mạch rất nhanh và nó có thể được chế tạo bằng công nghệ CMOS chuẩn.

b. Các thiết bị lập trình cầu chì nghịch (Anti-fuse)

Công nghệ lập trình anti-fuse được sử dụng trong các FPGA của Actel-Corp, Quick Logic và Cross Point Solution. Tuy anti-fuse được sử dụng trong các loại FPGA này có cấu tạo khác nhau, nhưng chức năng của chúng là như nhau. Một anti-fuse bình thường sẽ ở trạng thái cao, nhưng có thể bị "nóng chảy" thành trạng thái điện trở thấp khi được lập trình ở điện thế cao. Dưới đây sẽ giới thiệu cấu tạo của các anti-fuse của Actell và Quick Logic.

Anti-fuse của Actell được gọi là PLICE. Nó cấu trúc hình chữ nhật gồm 3 lớp: Lớp dưới cùng chứa các silic mang nhiều điện tích dương (n+diffusion), lớp giữa là một lớp điện môi (Oxy-Nitơ-Oxy cách điện), và lớp trên cùng là Poly-Silic.

Anti-fuse PLICE được lập trình bằng cách đặt một điện thế cao thích hợp (18V) giữa hai đầu của anti-fuse và dòng điều khiển khoảng 5mA qua thiết bị. Dòng và áp này tạo ra một nhiệt lượng đủ nóng bên trong lớp điện môi làm nó nóng chảy và tạo ra một liên kết dẫn điện giữa các điện cực. Các transistor chịu được các điện thế cao được chế tạo bên trong FPGA để đáp ứng cho dòng và điện áp đủ lớn. Cả hai lớp dưới cùng và trên cùng của cầu chì nghịch được nối với các dây kim loại để khi được lập trình cầu chì nghịch sẽ tạo ra một kết nối có trở kháng thấp (300Ω đến 500Ω) giữa hai dây kim loại.

Anti-fuse của Quick-Logic được gọi là ViaLink. Nó tương tự như PLICE cũng có ba lớp kim loại. Tuy nhiên, ViaLink sử dụng kim loại mức 1 cho lớp dưới cùng, một hợp chất vô dịnh hình cho lớp giữa và kim loại mức 2 cho lớp trên cùng. Khi ở trạng thái không được lập trình, anti-fuse có trở kháng hàng gigaôm, nhưng khi được lập trình nó sẽ tạo ra một kết nối giữa hai lớp kim loại trở kháng khoảng 80Ω. Anti-fuse được chế tạo bằng cách thêm 3 mặt nạ đặc biệt trong quy trình chế tạo CMOS thông thường.

ViaLink anti-fuse được lập trình bằng cách đặt một điện thế 10V giữa các đầu của nó, dòng được cấp đủ, trạng thái của Silic vô định hình sẽ thay đổi và tạo ra một liên kết điện giữa hai lớp kim loại. Diện tích các chíp sử dụng kỹ thuật anti-fuse rất nhỏ so với công nghệ khác. Tuy nhiên, bù lại cần phải có không gian lớn cho các transistor điện thế cao cần để giữ cho dòng và áp cao lúc lập trình. Nhược điểm của anti-fuse là quy trình chế tạo chúng phải thay đổi so với quy trình chế tạo SMOS.

c. Công nghệ lập trình dùng EPROM và EEROM

Công nghệ được dùng trong các FPGA của Altera Corp, và Plus Logic. Công nghệ này giống như sử dụng trong bộ nhớ EPROM. Không giống CMOS transistor đơn giản, một EPROM transistor gồm hai cổng, một cổng treo (*floating-gate*) và một cổng chọn (*select-gate*). Cổng treo được đặt giữa cổng chọn và kênh dẫn của transitor, cổng này được gọi như thế vì nó không có kết nối điện đến bất kỳ mạch nào.

Các công nghệ lập trình FPGA được tóm tắt trong bảng dưới đây:

Bảng 1.1 Các đặc tính của công nghệ lập trình

Công nghệ lập trình	**Tính bay hơi**	**Có thể lập trình**	**Diện tích chíp**	**R(Kohm)**	**C(pf)**

Static RAM Cell	Có	Trong mạch	Lớn	1-2	10-20
PLICE Anti-fuse	Không	Không	Anti-fuse nhỏ Số transistor lớn	300-500	3-5
ViaLink Anti-fuse	Không	Không	Anti-fuse nhỏ Số transistor lớn	50-80	1-3
EPROM	Không	Ngoài mạch	Nhỏ	2-4	10-20
EEPROM	Không	Trong mạch	2xEPROM	2-4	10-20

1.3. Ứng dụng của công nghệ FPGA

FPGA là thế hệ sau của IC khả trình nên chúng có thể ứng dụng trong hầu hét các ứng dụng của hiện đang dùng MPGA, PLD và các mạch tích hợp loại nhỏ (SSI).

a. Các mạch tích hợp là ứng dụng đặc biệt

FPGA là thiết bị tổng quát nhất để thực hiện các mạch lôgic số. Chúng đặc biệt thích hợp cho các mạch tích hợp chuyên dụng đặc biệt (ASIC) như bộ cộng, bộ điều khiển lôgic Flip-Flop...

b. Thiết kế mạch ngẫu nhiên

Mạch lôgic ngẫu nhiên thường được thực hiện bằng PAL. Nếu tốc độ của mạch không đòi hỏi khắt khe (các PAL nhanh hơn hầu hết các FPGA) thì mạch có thể thực hiện bằng FPGA. Hiện nay một FPGA cần từ 10 đến 20 PAL.

c. Thay thế các chíp SSI cho mạch ngẫu nhiên

Các mạch hiện tại trong các sản phẩm thương mại thường chứa nhiều chíp SSI. Trong nhiều trường hợp có thể thay thế bằng FPGA để giảm diện tích bo mạch.

d. Chế tạo mẫu

FPGA rất lý tưởng cho việc tạo mẫu các sản phẩm. Giá thành thực hiện thấp và thời gian thực hiện thiết kế vật lý ngắn, cung cấp các ưu điểm hơn nhiều so với các phương tiện truyền thống khác để chế tạo mẫu phần cứng. Các mẫu ban đầu có thể thực hiện rất nhanh và những thay đổi sau đó được thực hiện rất nhanh và ít tốn kém.

e. Máy tính dựa trên FPGA

Một loại máy tính dựa trên FPGA có thể tái lập trình ngay trên FPGA. Các máy này có một bo mạch chứa các FPGA với các chân nối với các chíp lân cận giống như thông thường. Ý tưởng là là một chương trình phần mềm có thể được "biên dịch" (sử dụng kỹ thuật tổng hợp mức cao, mức lôgic và mức sơ đồ bằng tay) vào ngay phần cứng. Phần cứng này sẽ được thực hiện bằng cách lập trình bo mạch FPGA. Phương pháp này có hai ưu điểm chính: một là không cần quá trình lấy lệnh như các bộ xử lý truyền thống vì phần cứng đã gộp cả lệnh. Kết quả là tốc độ có thể tăng lên hàng trăm lần. Hai là, môi trường tính toán có thể thực hiện song song mức cao, làm tăng tốc thêm nữa.

f. Tái cấu hình thành phần trực tiếp

FPGA cho phép có thể thay đổi theo mong muốn cấu trúc của một máy đang hoạt động. Một ví dụ là các thiết bị máy tính từ xa có thể thay đổi trực tiếp để khắc phục sự cố hay

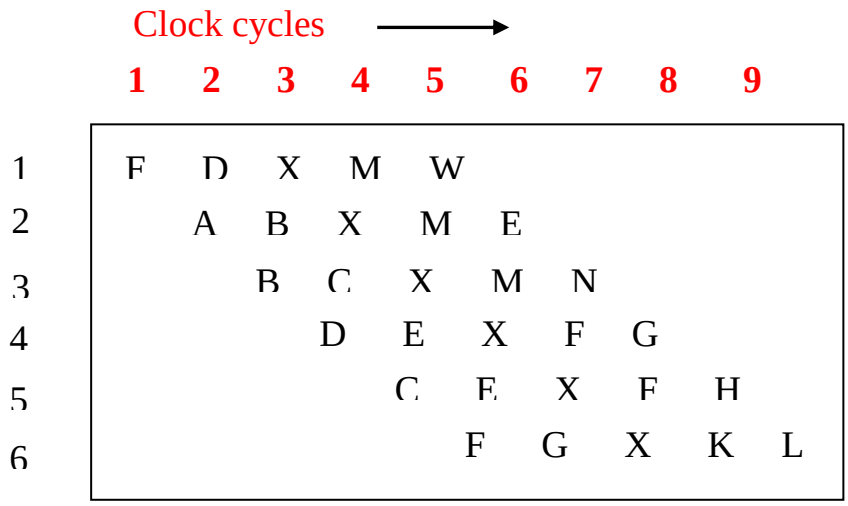
có lỗi thiết kế. Kiểu FPGA thích hợp nhất cho ứng dụng này là những FPGA có các chuyển mạch lập trình được.

2. KỸ THUẬT PIPELINE TRONG CÔNG NGHỆ FPGA

2. 1. Kỹ thuật pipeline trong FPGA

Đối với lập trình FPGA, việc có thể lập trình và điều khiển khả năng chia thành các nhánh xử lý song song, sử dụng hiệu quả tài nguyên của phần cứng là một trong những khả năng có thể mang lại hiệu quả xử lý nhanh, đặc biệt đối với các loại dữ liệu đồng nhất như dữ liệu text, hay đối với các loại giá trị số, các bit dữ liệu hình ảnh. [15]

Kỹ thuật đường ống (Pipeline hay Pipelining) là kỹ thuật thực hiện lệnh trong đó cac lệnh được thực hiện theo kiểu gối đầu nhằm tận dụng những khoảng thời gian rỗi (stalls) giữa các công đoạn (stages), qua đó làm tăng tốc độ thực hiện lệnh của vi xử lý.



Hình 2.1: Kỹ thuật Pipeline

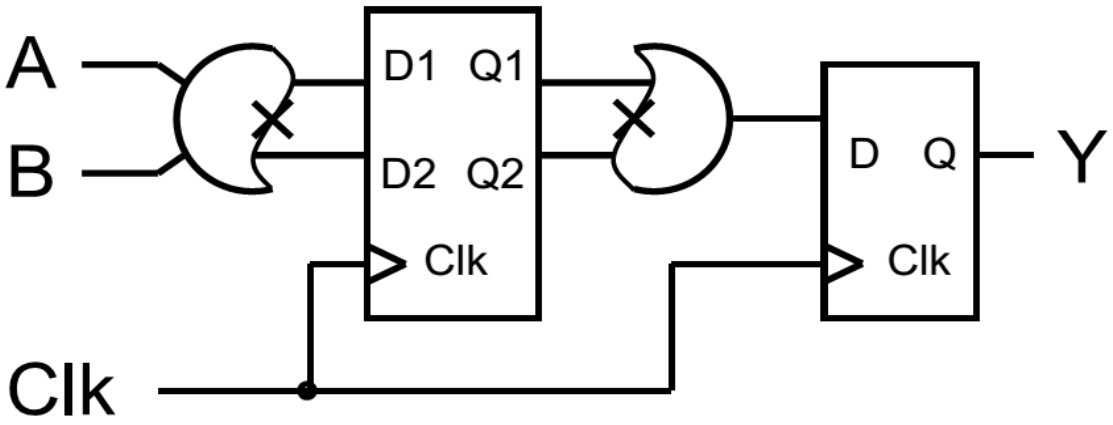
Trên Hình 2.1 mô tả cách thực hiện đối với Kỹ thuật Pipeline, qua đó cho thấy để thực hiện được 6 câu lệnh xử lý text, có thể chỉ mất 10 xung nhịp đồng hồ, trong khi nếu thực hiện tuần tự từng câu lệnh có thể sẽ tốn kém 5 xung nhịp đồng hồ đối với 1 câu lệnh, hay sẽ tốn mất 30 xung nhịp đồng hồ cho cả 6 câu lệnh đó.

Việc sử dụng kỹ thuật Pipeline được thực hiện dễ dàng nhờ kiến trúc đặc biệt của các vi mạch thuộc họ FPGA do khả năng thay đổi cấu trúc và sử dụng tài nguyên tính toán của chip FPGA.

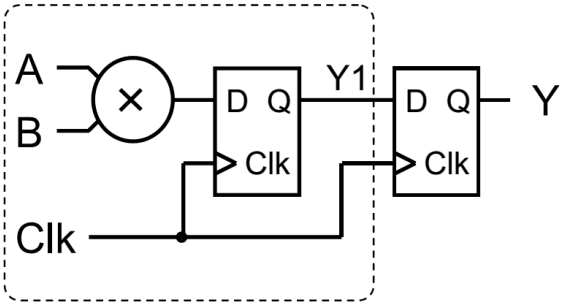
Với kỹ thuật Pipeline trên, rõ ràng kết quả thực hiện được sẽ diễn ra nhanh hơn so với quá trình tuần tự khi không sử dụng Pipeline.

2.2. Tổ chức pipeline trong Lập trình FPGA

Việc sử dụng kỹ thuật Pipeline có thể là cách tiếp cận đề xử lý nhanh dữ liệu, đặc biệt khi thực hiện các phép toán logic trên ALU như phép nhân, phép cộng …, mục tiêu của mục này là tìm hiểu cách thức tổ chức Pipeline khi lập trình cho FPGA, với các code chương trình cho lập trình FPGA như VHDL.



2.2a




```
PipeMultProc : process
begin
  wait until Clk = '1' ;
  Y1 <= A * B ;
  Y <= Y1 ;
end process ;
```

2.2b

3. ỨNG DỤNG KỸ THUẬT PIPELINE TRONG CÔNG NGHỆ FPGA XỬ LÝ DỮ LIỆU ĐA PHƯƠNG TIỆN

3.1. Các kỹ thuật xử lý dữ liệu đa phương tiện

Đối với bài toán xử lý dữ liệu lớn, với các khối dữ liệu text hay voice, các dữ liệu đa phương tiện, một trong những vấn đề cấp bách đặt ra là bài toán xử lý nhanh. Bài toán xử lý dữ liệu nhanh đã được đề cập đến trong nhiều nghiên cứu với những thuật toán tính toán lưới, vấn đề xây dựng các hệ thống hiệu năng cao, tính toán song song, lập trình trên các chip họ FPGA, sử dụng card đồ họa GPU.

3.1.1. Tính toán song song

Tính toán song song là một hình thức tính toán trong đó nhiều phép tính được thực hiện đồng thời, hoạt động trên nguyên tắc là những vấn đề lớn đều có thể chia thành nhiều phần nhỏ hơn, sau đó được giải quyết tương tranh ("trong lĩnh vực tính toán"). Có nhiều hình thức khác nhau của tính toán song song: song song cấp bit, song song cấp lệnh, song song dữ liệu, và song song tác vụ. Song song đã được sử dụng từ nhiều năm qua, chủ yếu là trong lĩnh vực tính toán hiệu năng cao. Gần đây hình thức tính toán này được quan tâm nhiều hơn, do những hạn chế vật lý ngăn chặn việc tăng hiệu năng tính toán chỉ bằng cách tăng tần số, Vì việc tiêu hao điện năng (dẫn đến sinh nhiệt) từ máy tính đã trở thành một mối lo ngại trong những năm gần đây, tính toán song song đã trở thành mô hình thống trị trong lĩnh vực kiến trúc máy tính, phần lớn là dưới dạng bộ xử lý đa nhân.

Thuật toán song song khó viết hơn so với những thuật toán tuần tự, vì sự tương tranh tạo ra nhiều lớp mới tiềm tàng các lỗi phần mềm, trong đó lỗi điều kiện ganh đua là phổ biến nhất. Quản lý việc Giao tiếp và đồng bộ giữa các luồng xử lý là một trong những trở ngại lớn nhất để tạo ra một chương trình song song tốt.

3.1.2. Sử dụng GPU

Bộ xử lý đồ họa (Graphics Processing Unit) [9] [10] hay gọi tắt là GPU là bộ xử lý chuyên dụng cho biểu diễn hình ảnh 3D từ bộ vi xử lý của máy tính. Nó được sử dụng trong các hệ thống nhúng, điện thoại di động, máy tính cá nhân, máy trạm, và điều khiển game. Bộ xử lý đồ họa ngày nay rất hiệu quả trong các thao tác đồ họa máy tính, và cấu trúc song song cao cấp làm cho chúng có năng lực xử lý tốt hơn nhiều so với bộ vi xử lý thông thường trong các thuật toán phức tạp. Trong máy tính cá nhân, một GPU được biết tới như một card màn hình (video card) hoặc được tích hợp luôn trên bảng mạch chủ. Hơn 90% các máy tính cá nhân hoặc máy tính xách tay hiện đại đã có tích hợp GPU nhưng thường yếu hơn nhiều so với GPU tích hợp trên các card màn hình chuyên dụng.[12],[14]

GPU luôn luôn là một bộ xử lý với dư thừa tài nguyên tính toán. Tuy nhiên xu hướng quan trọng nhất gần đây đó là trưng bày khả năng tính toán đó cho các lập trình viên. Những năm gần đây, GPU đã phát triển từ một hàm cố định, bộ xử lý chuyên dụng tới bộ xử lý lập trình song song, đầy đủ tính năng độc lập với việc bổ sung thêm các chức năng cố định, và các chức năng chuyên biệt. Hơn bao giờ hết các khía cạnh về khả năng lập trình của bộ xử lý chiếm vị trí trung tâm. Tôi bắt đầu bằng cách ghi chép lại sự tiến triển này, bắt đầu từ cấu trúc của đường ống dẫn đồ họa GPU và làm thế nào GPU trở thành kiến trúc, công cụ giành cho các mục đích thông dụng, sau đó đi xem xét kỹ hơn các kiến trúc của GPU hiện đại.

3.1.3. Tính toán lưới

Tính toán lưới là một mô hình tính toán đang được ứng dụng rất nhiều, đây là một mô hình có khả năng thực hiện tính toán với tốc độ cao bằng cách tận dụng nhiều máy tính nối mạng để tạo ra một kiến trúc máy tính ảo có thể phân phối việc tính toán trên một cơ sở hạ tầng song song. Lưới tính toán sử dụng tài nguyên của nhiều máy tính riêng lẻ kết nối với nhau thông qua một mạng máy tính(thường là mạng máy Internet) để giải quyết các yêu cầu tính toán lớn. Lưới có khả năng thực hiện việc tính toán trên những tập dữ liệu lớn, bằng cách chia nhỏ các tập dữ liệu này thành các tập hợp nhỏ hơn hoặc thực hiện nhiều qui trình tính toán cùng lúc như trên một mánh tính đơn thông qua mô hình phân phối công việc giữa các tiến trình song song. Ngày nay, việc phân phối tài nguyên trên lưới tuân theo chuẩn SLA(service level agreement)

Tính toán lưới tạo ra một mô hình để giải quyết các bài toán tính toán lớn bằng cách sử dụng những tài nguyên rỗi(CPU, và thiết bị lưu trữ) của một loạt các máy tính riêng rẽ, thường là máy để bàn, hệ thống này được coi là một một cụm máy ảo nhúng trong một môi trường liên lạc phân tán. Tính toán lưới tập trung vào khả năng hỗ trợ tính toán giữa các khu vực hành chính, điều này làm cho mô hình này tách biệt so với mô hình cụm tính toán và tính toán phân tán truyền thống.

Về mặt chức năng có thể chia lưới thành những loại sau:

- Lưới tính toán(bao gồm cả các lưới tận dụng tài nguyên CPU) đây là loại lưới hướng tới các hoạt động tính toán lớn và tập trung
- Lưới dữ liệu tập trung vào việc quản lý việc chia sẻ của một lượng lớn các dữ liệu phân tán.
- Lưới thiết bị thường bao gồm một thiết bị chính như các kính viễn vọng và một lưới xung quanh dùng để điều khiển thiết bị từ xa và phân tích dữ liệu.

3.2. Thiết kế lõi IP xử lý dữ liệu đa phương tiện

3.3.1. Khái niệm vi mạch và lõi IP

Chip vi mạch IC (integrated Circuit) là mạch điện gồm nhiều phần tử tích hợp lên trên một phiến bán dẫn. Dựa trên mật độ tích hợp, người ta phân loại như sau:

- SSI (Small-Scale Integration): Độ tích hợp cỡ nhỏ gồm khoảng 100 linh kiện điện tử trên một chip
- MSI (Medium-Scale Integration): Gồm từ 100 đến 3000 linh kiện điện tử trên một chip
- LSI (Large-Scale Integration): Gồm từ 3000 đến 100000 linh kiện điện tử trên một chip
- VLSI (Very Large-Scale Integration): Gồm từ 100000 đến một triệu linh kiện điện tử trên một chip
- ULSI (Ultra Large-Scale Integration): Hơn một triệu linh kiện điện tử trên một chip

Có hai dạng lõi IP:

- **Lõi IP mềm (soft IP):** có tính linh hoạt nhất, cho phép người dùng sửa đổi chức năng khi cần thiết. Nhưng để đưa vào thiết kế toàn bộ, khách hàng phải thực hiện các bước tổng hợp, đặt và nối dây... Như vậy, không đảm bảo chắc chắn sẽ đạt được các luật thiết kế của nhà máy, tối ưu về thời gian, diện tích...
- **Lõi IP cứng (hard IP):** được thiết kế như là một "hộp đen", khách hàng chỉ việc sử dụng. Không linh hoạt vì khách hàng không thể sửa đổi. Chỉ ứng dụng với một qui trình sản xuất của một nhà máy chế tạo cụ thể. Người cung cấp IP đảm bảo tuân thủ các luật thiết kế của nhà máy, thông báo rõ tần số làm việc, diện tích chiếm chỗ... của IP do mình tạo ra.

3.2.2. Qui trình thiết kế trên ASIC và FPGA

Error! Reference source not found. trình bày qui trình thiết kế một lõi/chip IP trên hai công nghệ FPGA và ASIC. Trong lưu đồ này, qui trình thiết kế như sau : Từ yêu cầu thiết kế (**Specifications**) ➔ Phân tích thiết kế, xây dựng kiến trúc, viết code mô tả phần cứng cho thiết kế (dùng Verilog hoặc VHDL) và tiến hành mô phỏng kiểm tra chức năng và sửa lỗi thiết kế (**RTL code, simulation & debug**) ➔ Tổng hợp thiết kế thành dạng netlist (**logic synthesis**) và kiểm tra chức năng ở mức cổng - gate-level ➔ Tiến hành làm **layout** và phân tích bản thiết kế layout về mặt định thời (timing), công suất, tần số và diện tích chip.

Về mặt phần cứng, có hai hướng để làm chip :

(1) **Hướng FPGA :** dùng chip FPGA chuyên dụng và các phần mềm tương ứng. Ví dụ Quartus II của Altera hay ISE của Xilinx,…

(2) **Hướng ASIC :** khi tổng hợp và làm layout sẽ dựa trên thư viện công nghệ cụ thể và các phần mềm chuyên dụng, ví dụ như IBM, TSMC, …

3.3.3. Một số kỹ thuật tối ưu thiết kế

3.3.3.1. Kỹ thuật tối ưu tài nguyên

- **Kỹ thuật tối giản logic bằng đại số Boole**

Các hàm logic thường có thể được tối giản bằng cách sử dụng đại số Boole và bìa Karnaugh trước khi thiết kế.

- **Kỹ thuật Resource Sharing**

Resource Sharing là kỹ thuật chia sẻ tài nguyên phần cứng cho các xử lý khác nhau. Để áp dụng lỹ thuật này, người thiết kế phải am hiểu về hệ thống và tính toán định thời hợp lý.

Kỹ thuật Reource Sharing nếu được áp dụng đúng lúc và hợp lý sẽ cho kết quả giảm thiểu được nhiều tài nguyên thiết kế.

- **Kỹ thuật Register Packing**

Kỹ thuật Register Packing là một kỹ thuật tối ưu tài nguyền dùng cho thiết kế trên FPGA. Ý tưởng cơ bản là một mạch tổ hợp sẽ sử dụng thanh ghi còn trống trong một Logic Element khác.

- **Kỹ thuật tối ưu phép nhân hằng số**

Đối với các phép toán nhân hằng số, có thể lược giản bộ nhân nhờ vào phép dịch bít. Do mỗi đơn vị dịch về phía trái tướng đương với việc nhân 2, nên các phép toán nhân với 2^N sẽ tương đương với phép dịch trái N bit.

Ví dụ: 8 * 4 = 32, do $4 = 2^2$ nên phép nhân sẽ được thay bằng phép dịch 2 bit cho giá trị 8. Tức là 1000 (8) << 2 = 100000 (32).

Đối với các khác 2^N, có thể quy về phép toán dịch và công như vi dụ sau:

Ví dụ: 9 * 5 = 45, do $5 = 4 + 1 = 2^2 + 1$ nên có thể quy về phép toán dịch và cộng. Tức là 1001 (9) << 2 + 1001 (9) = 101101 (45).

- **Kỹ thuật Retiming theo hướng tối ưu tài nguyên**

Kỹ thuật Retiming theo hướng tối ưu tài nguyên giảm bớt số thanh ghi bằng cách dịch chuyển thanh ghi trong mạch, qua đó giảm bớt số thanh ghi mà khong làm thay đổi chức năng mạch. Tuy nhiên, tốc độ có thể bị ảnh hưởng

3.2.3. Kỹ thuật tối ưu tốc độ xử lý

- **Kỹ thuật Resource Rebalance**

Kỹ thuật resource rebalance cân đối lại sự cân bằng trong việc nhóm các phép tính, phân chia các phép tính có thể tính toán cùng lúc thành các mạch tính song song, từ đó tăng được tốc độ xử lý của mạch.

- **Kỹ thuật Retiming theo hướng tối ưu tốc độ**

Kỹ thuật Retiming cân bằng lại thanh ghi giữa hai mạch tổ hợp để cân bằng delay cho toàn bộ mạch

Bằng cách retiming lại, delay của mạch sẽ bằng delay của một mức logic. Tuy nhiên, số thanh ghi sẽ tốn một thanh cho mục đích retiming này.

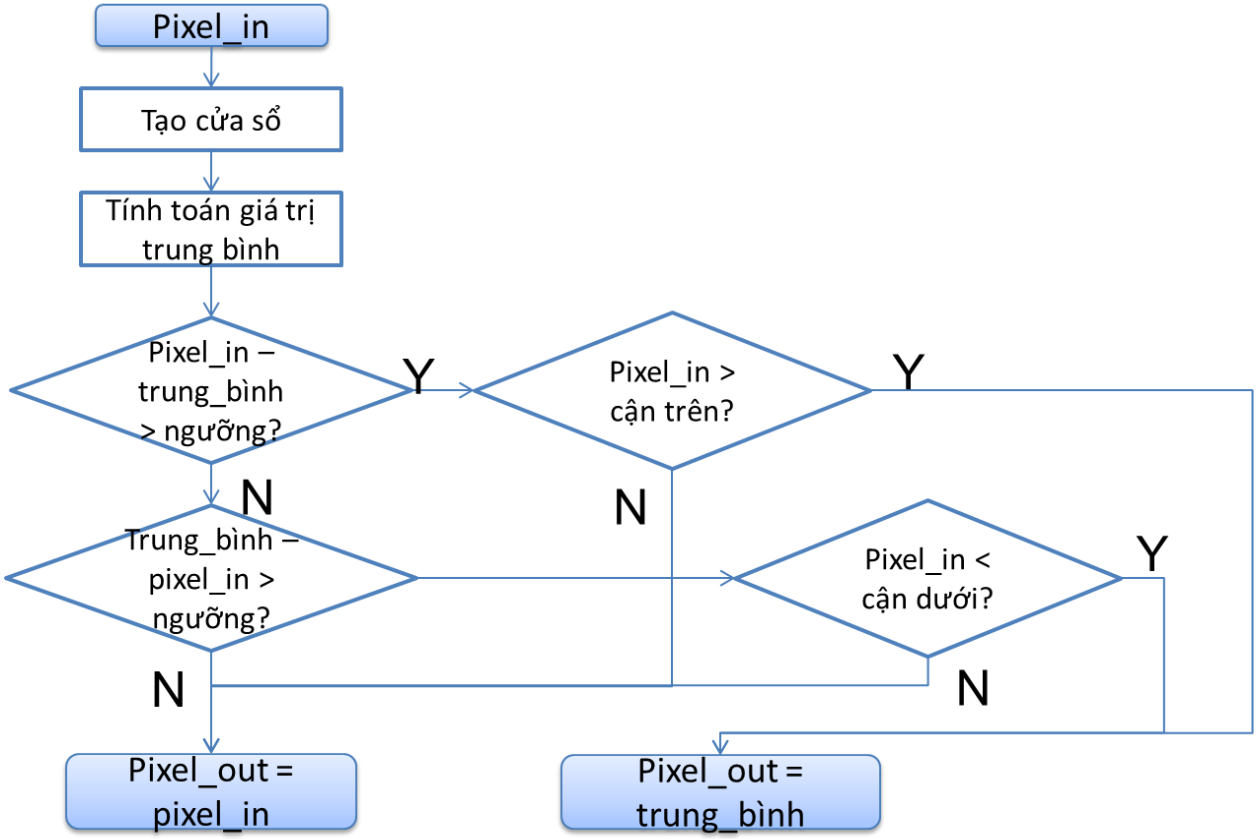
- **Kỹ thuật chèn thanh ghi pipeline**

Kỹ thuật chèn thanh ghi pipeline là một kỹ thuật hiệu quả trong trường hợp cần nâng cao tầng số hoạt động của mạch. Tuy nhiên phương pháp này đồng thời cũng tăng thêm tài nguyên do yêu cầu thêm các thanh ghi pipleline. Ngoài ra, để bảo đảm chức năng của mạch không bị thay đổi, định thời của mạch cũng cần phải được tính toán lại.

3.4. Thiết kế lõi IP xử lý ảnh

3.4.1. Thiết kế lõi IP defect pixel correction

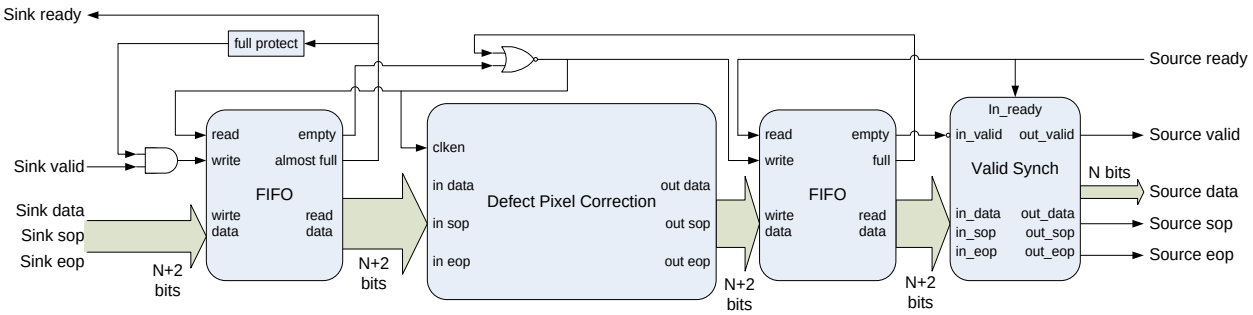
- Giải thuật:
- Sử dụng cửa sổ trung bình
- So sánh ngưỡng
- Kết hợp sánh với cận trên, cận dưới

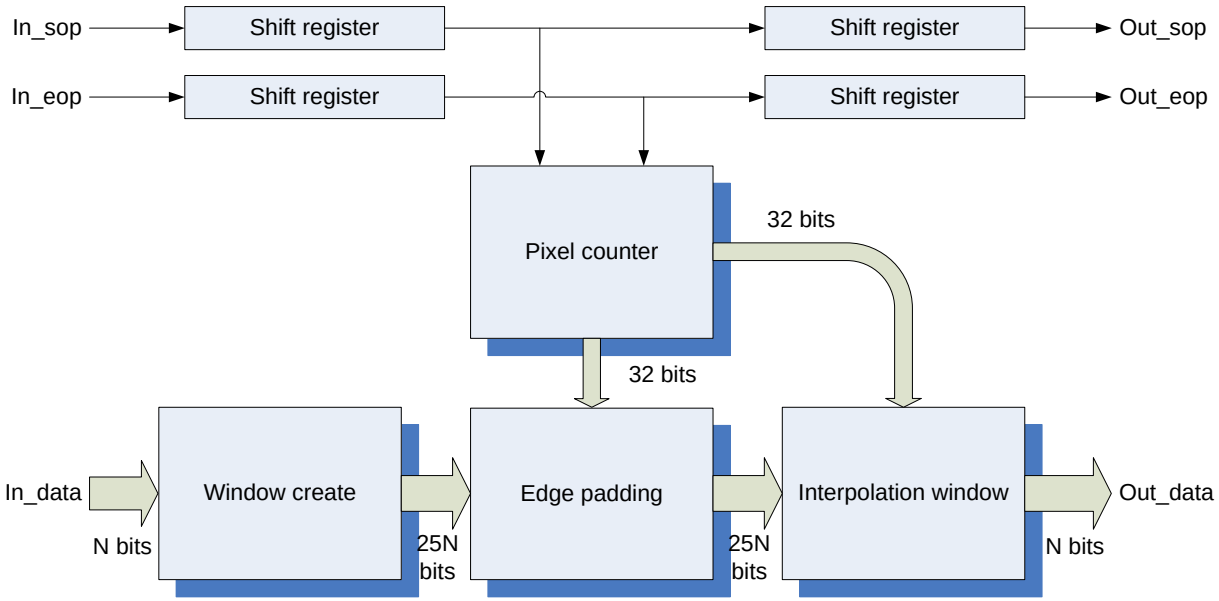


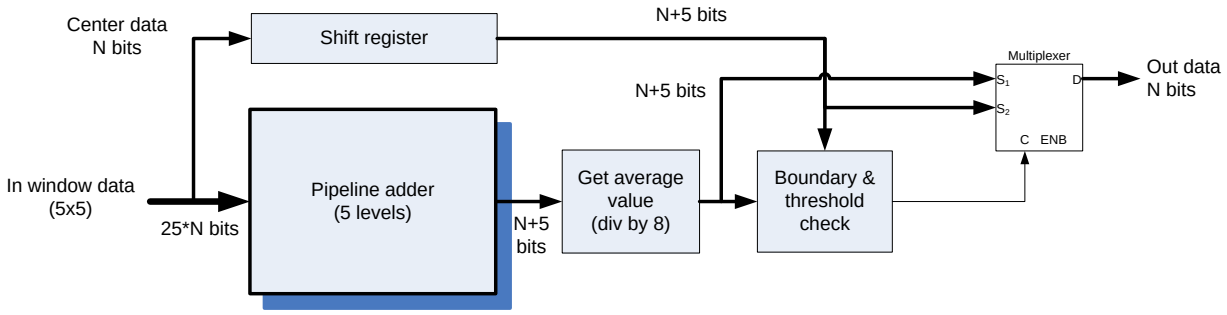
Các ngõ vào ra của IP Defect pixel correction

Tên tín hiệu	Độ rộng (bit)	Hướng	Mô tả
clock	1	in	Avalon Clock and Reset
resetn	1	in	Avalon Clock and Reset
enable	1	In	Avalon Conduit
Sink_data_in	DW	in	Avalon ST – Sink - data
Sink_sop_in	1	in	Avalon ST – Sink – start of packet
Sink_eop_in	1	in	Avalon ST – Sink – end of packet
Sink_valid_in	1	in	Avalon ST – Sink – data valid
Sink_ready_out	1	out	Avalon ST – Sink – data ready
Source_data_out	DW	out	Avalon ST – Source – data
Source_sop_out	1	out	Avalon ST – Source – start of packet
Source_eop_out	1	out	Avalon ST – Source – end of packet
Source_valid_out	1	out	Avalon ST – Source – data valid
Source_ready_in	1	in	Avalon ST – Source – data ready

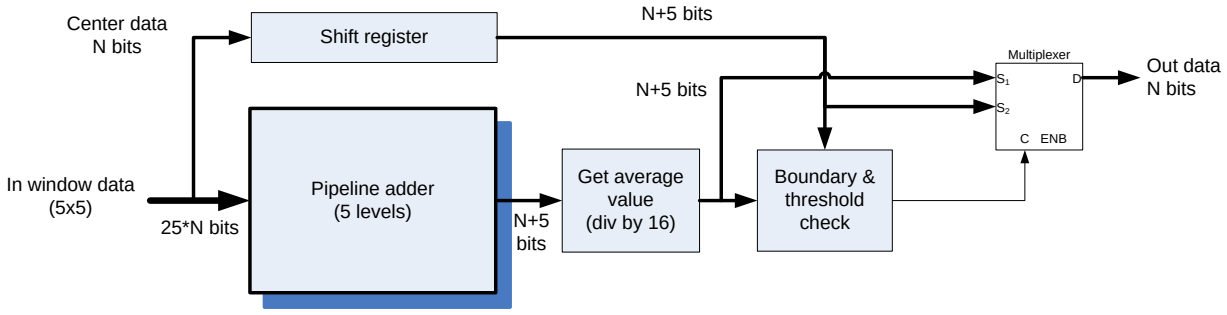
Thiết kế chi tiết



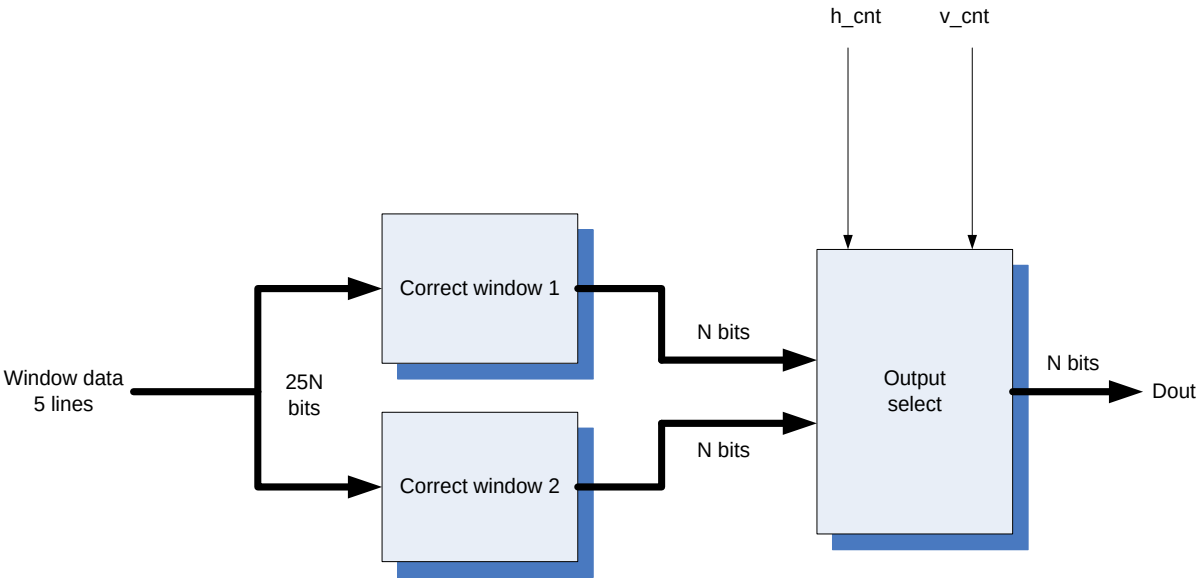




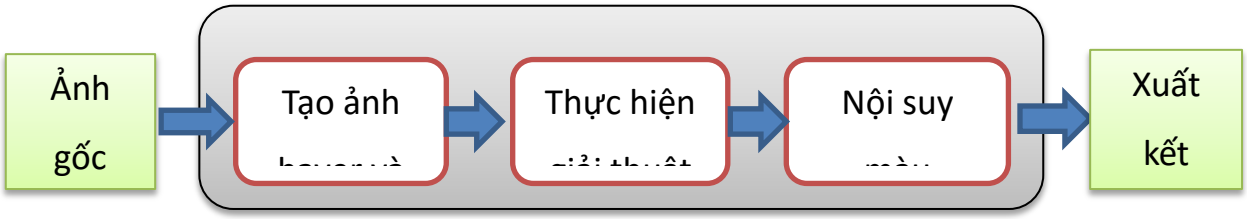
Correct window for R and B pixels



Correct window for G pixels



- Mô phỏng giải thuật bằng Matlab



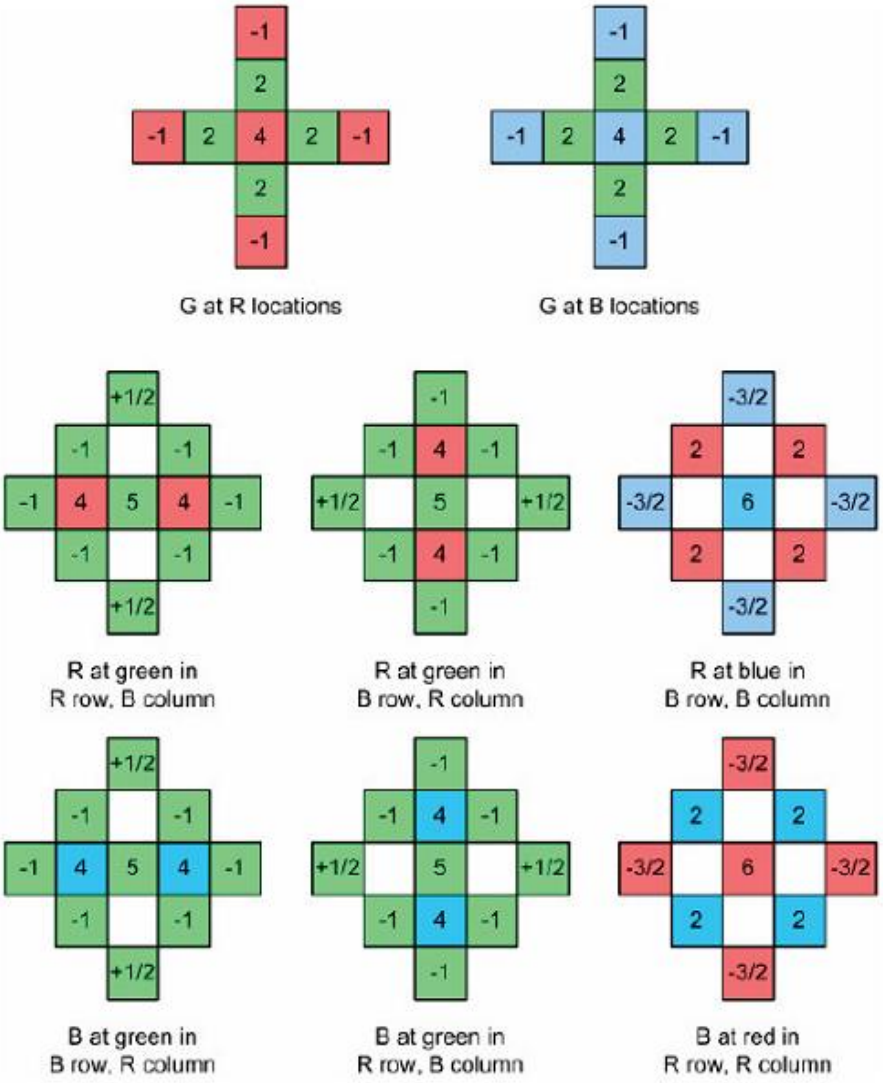
Kết quả chạy mô phỏng:

3.4.2. Thiết kế lõi IP color filter interpolation

- Yêu cầu thiết kế:
  - Hỗ trợ không gian màu RGB và CMY
  - Độ mở ống kính nội suy 5x5
  - Cho phép tùy chọn độ rộng dữ liệu ngõ vào: 8 bit, 10 bit và 12 bit
  - Tương thích chuẩn Avalon Streaming (ready_latency = 1).
  - Tốc độ xung clock >100 MHz trên FPGA Cyclone II của Altera.

- Tốc độ dữ liệu > 30 Msamples/s (1 sample gồm 3 thành phần màu tối thiểu 8bit).
- Thiết kế đồng bộ clock cạnh lên
- Reset bất đồng bộ

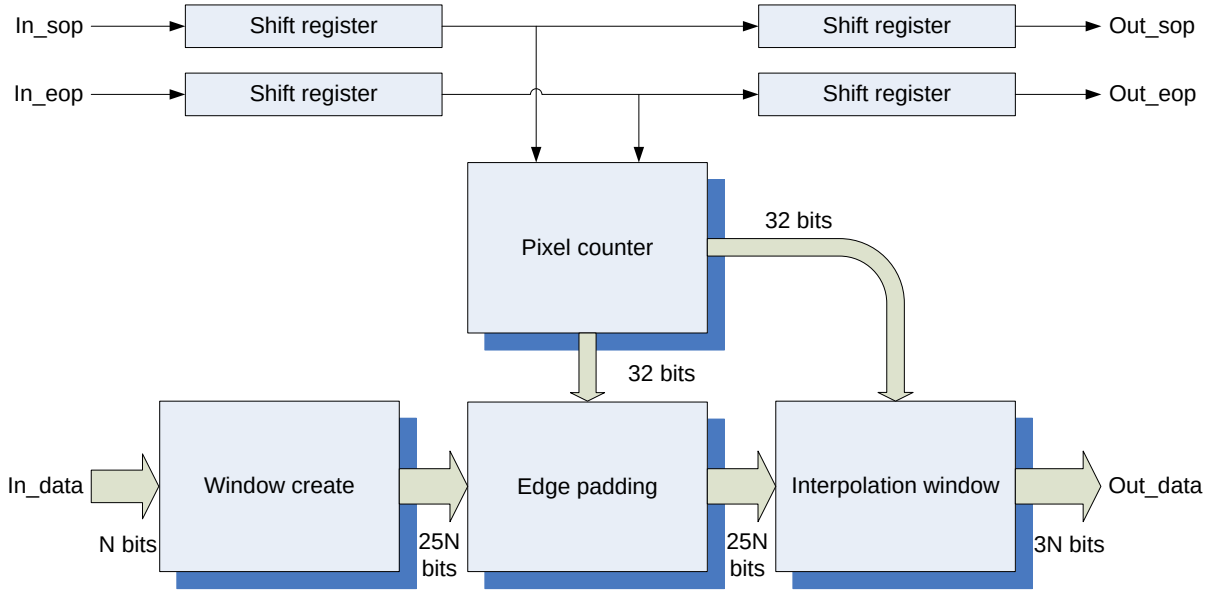
- **Giải thuật đề nghị:** *nội suy tuyến tính cải tiến* - dựa vào 8 nhân cửa sổ với các hệ số được tính toán dựa vào tài liệu "High-quality linear interpolation for demosaicking of Bayer-patterned color image"

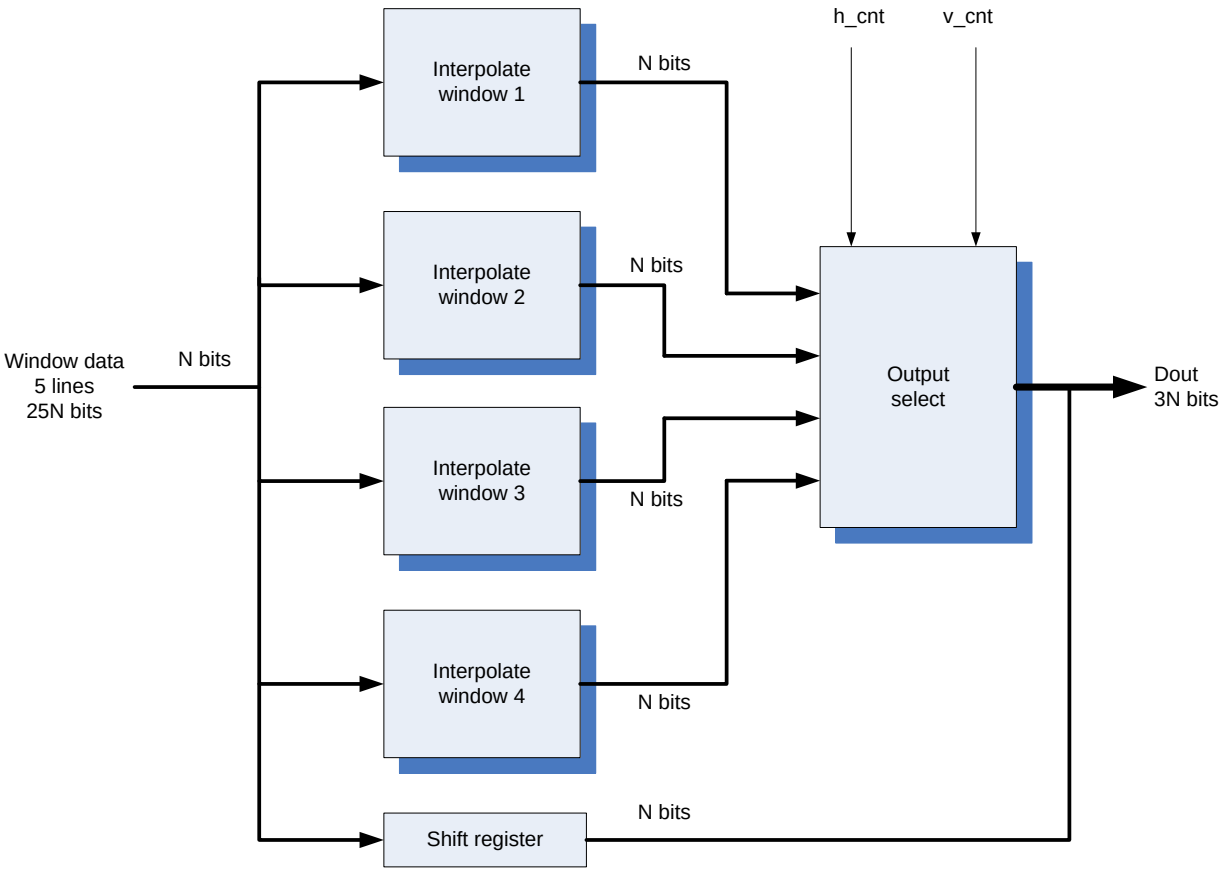


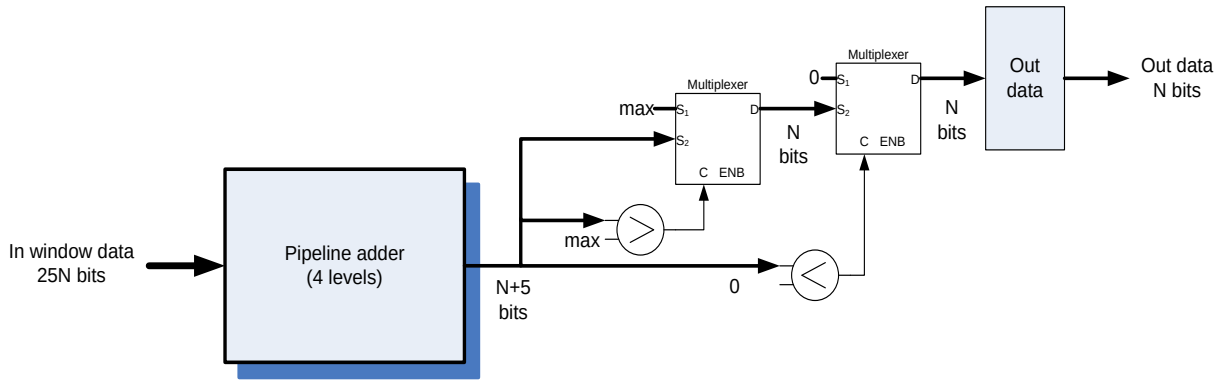
Bảng 3.4 Các ngõ vào ra của IP Color Filter Interpolation

Tên tín hiệu	Độ rộng (bit)	Hướng	Mô tả
clock	1	in	Avalon Clock and Reset
resetn	1	in	Avalon Clock and Reset
enable	1	In	Avalon Conduit
Sink_data_in	DW	in	Avalon ST – Sink - data
Sink_sop_in	1	in	Avalon ST – Sink – start of packet
Sink_eop_in	1	in	Avalon ST – Sink – end of packet
Sink_valid_in	1	in	Avalon ST – Sink – data valid
Sink_ready_out	1	out	Avalon ST – Sink – data ready
Source_data_out	3*DW	out	Avalon ST – Source – data
Source_sop_out	1	out	Avalon ST – Source – start of packet
Source_eop_out	1	out	Avalon ST – Source – end of packet
Source_valid_out	1	out	Avalon ST – Source – data valid
Source_ready_in	1	in	Avalon ST – Source – data ready

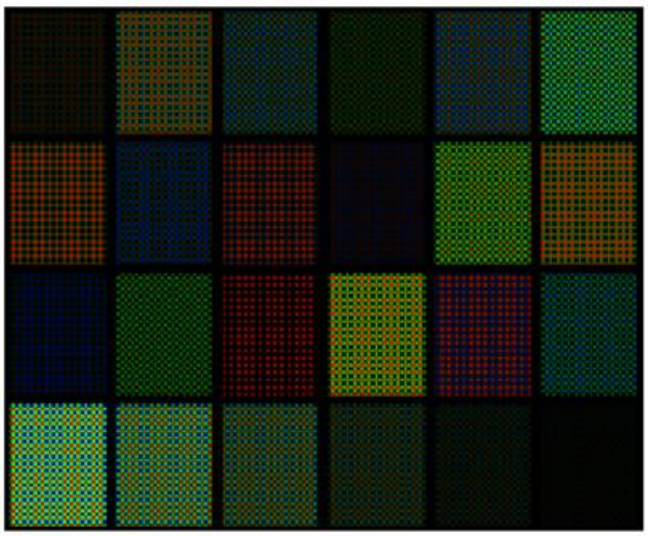
- Thiết kế chi tiết







- Kiểm tra thiết kế trên ModelSim



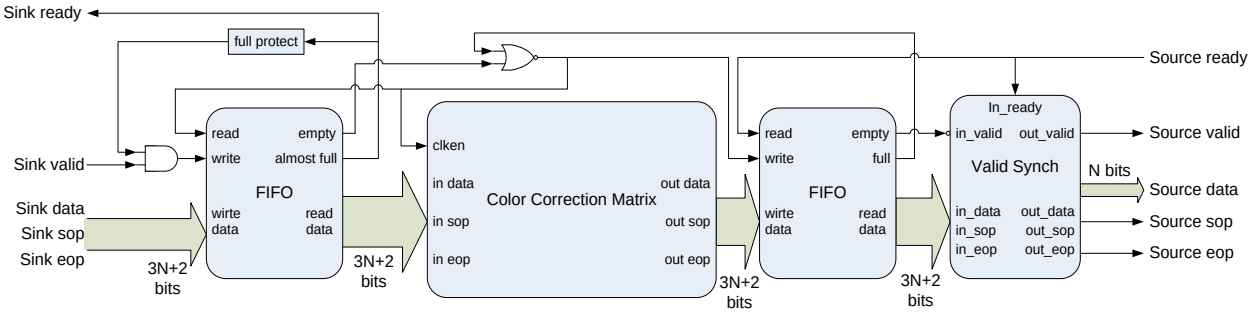
3.4.4. Thiết kế lõi IP color matrix correction

- Yêu cầu thiết kế:
  - Cho phép tùy chọn hệ số ma trận
  - Chuyển đổi màu CMY sang RGB
  - Cho phép tùy chọn độ rộng dữ liệu ngõ vào: 8 bit, 10 bit và 12 bit
  - Tương thích chuẩn Avalon Streaming (ready_latency = 1).
  - Tốc độ xung clock >100 MHz trên FPGA Cyclone II của Altera.
  - Tốc độ dữ liệu > 30 Msamples/s (1 sample gồm 3 thành phần màu tối thiểu 8bit).
  - Thiết kế đồng bộ clock cạnh lên
  - Reset bất đồng bộ

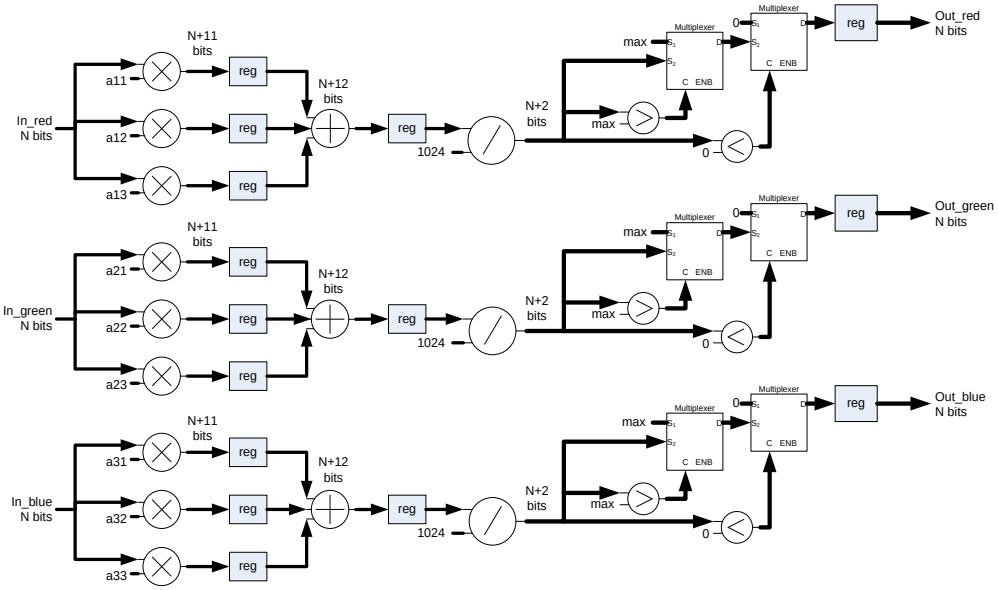
Các ngõ vào ra của IP Color Correction Matrix

Tên tín hiệu	Độ rộng (bit)	Hướng	Mô tả
clock	1	in	Avalon Clock and Reset
resetn	1	in	Avalon Clock and Reset
enable	1	In	Avalon Conduit
Sink_data_in	3*DW	in	Avalon ST – Sink - data
Sink_sop_in	1	in	Avalon ST – Sink – start of packet
Sink_eop_in	1	in	Avalon ST – Sink – end of packet
Sink_valid_in	1	in	Avalon ST – Sink – data valid
Sink_ready_out	1	out	Avalon ST – Sink – data ready
Source_data_out	3*DW	out	Avalon ST – Source – data
Source_sop_out	1	out	Avalon ST – Source – start of packet
Source_eop_out	1	out	Avalon ST – Source – end of packet
Source_valid_out	1	out	Avalon ST – Source – data valid
Source_ready_in	1	in	Avalon ST – Source – data ready

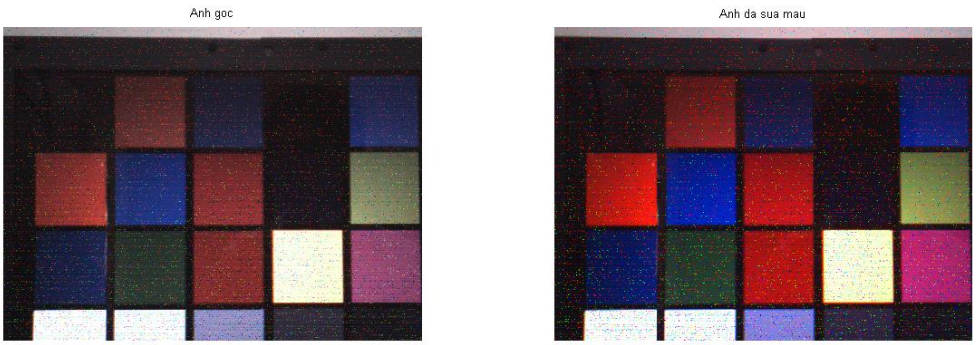
- Thiết kế top-block



- Thiết kế chi tiết

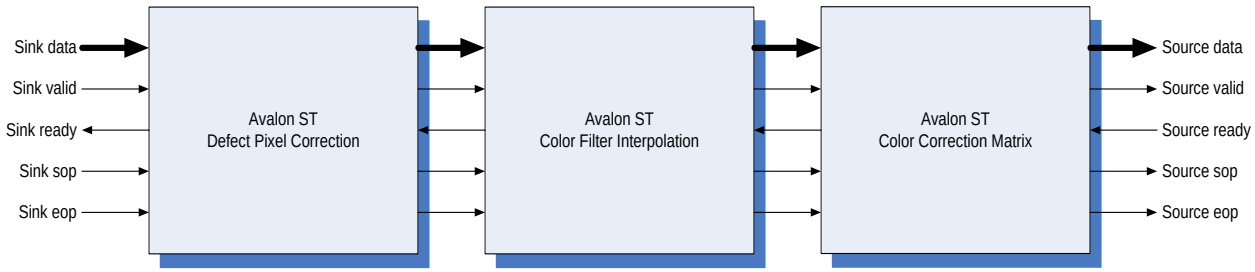


- Kết quả trên ModelSim

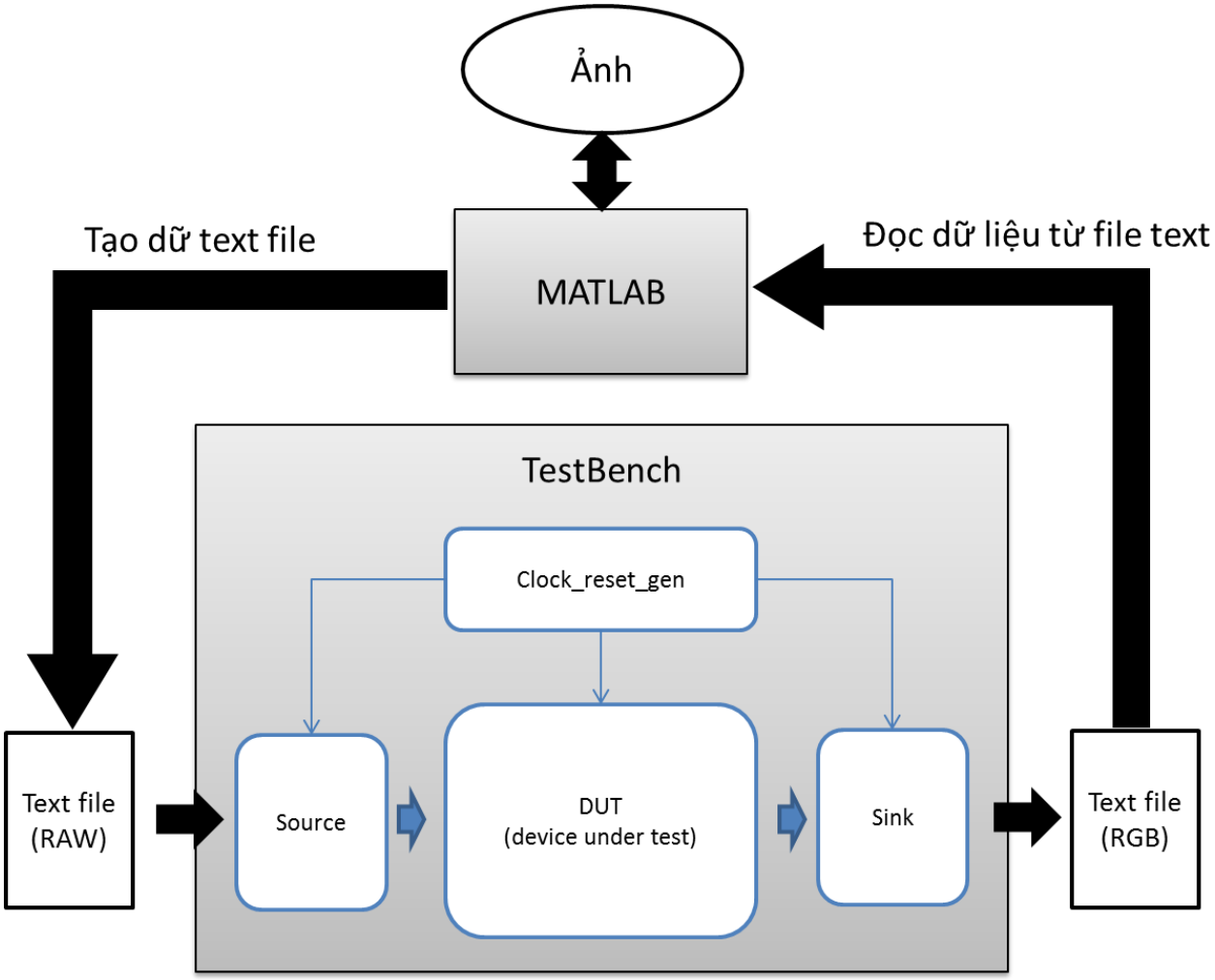


3.4.3. Tổng hợp 3 lõi IP

- Sơ đồ khối



Mô phỏng trên ModelSim để bảo đảm chức năng của lõi IP đúng với thiết kế giải thuật đã đề ra. Mô hình kiểm tra lõi IP trên ModelSim được mô tả trong hình .



Hình 0-1: Mô hình thẩm tra trên Model Sim

Hình ảnh cần kiểm tra được tạo ra dưới dạng text file nhờ công cụ trên Matlab. Text file này sẽ là ngõ vào cho bộ testbench thực hiện mô phỏng thẩm tra lõi IP. Kết quả mô phỏng sẽ được lưu lại dưới dạng text file và được đưa vào Matlab để đánh giá và hiển thị kết quả.

Các Verification IP (VIP) được thiết kế trong môi trường Testbench bao gồm:

- Source: đọc dữ liệu hình ảnh và chuyển thành dữ liệu theo chuẩn Avalon-ST cho lõi IP

- Sink: nhận luồng dữ liệu đã xử lý từ lõi IP theo chuẩn Avalon-ST và ghi lại kết quả vào text file
- Clock_reset_gen: Tạo ra tín hiệu clock và reset cho bộ source, sink và lõi IP.

  Các bước để chạy mô phỏng được mô tả như sau:
- Mở chương trình Matlab, chạy "im2txt.m" để biến đổi ảnh gốc sang text
- Mở chương trình ModelSim, chạy mô phỏng file "top.v" cho đến khi có thông báo "finish".
- Mở chương trình Matlab, chạy "txt2im.m" để biến đổi text file chứa ảnh kết quả thành file ảnh
- Xem file ảnh kết quả và so sánh với ảnh gốc.

Để thực hiện quá trình trên một cách tự động, các file script được thiết kế bao gồm:

- Clear.bat: xóa những file kết quả cũ
- Run.bat: thực hiện toàn bộ quá trình mô phỏng kiểm tra trên Matlab. Kết quả mô phỏng sẽ được hiển thị ra hình ảnh đã xử lý và so sánh với hình ảnh chưa xử lý điểm ảnh lỗi. Chú ý để chạy file run.bat máy tính cần cài đặt ModelSim và Matlab Compiler Runtime.
- Waveform.bat: xem dạng sóng mô phỏng trên ModelSim

Các file được thiết kế cho quá trình mô phỏng kiểm tra lõi IP Pre-Image Processing được liệt kê trong bảng sau:

STT.	**Tên file**	**Mô tả**
1	clock_reset_gen.v	Mã Verilog tạo tín hiệu clock và reset cho bộ source, sink, và lõi IP
2	source.v	Mã Verilog cho bộ đọc dữ liệu hình ảnh từ text file và chuyển thành dữ liệu theo chuẩn Avalon-ST cho lõi IP
3	sink.v	Mã Verilog cho bộ nhận dữ liệu hình ảnh theo chuẩn Avalon-ST và ghi thành text file
4	top.v	Mã Verilog tổng hợp môi trường kiểm tra lõi IP
5	im2txt.m	Mã script Matlab thực hiện việc biến đổi ảnh sang text file chứa ảnh Bayer và chèn thêm điểm ảnh lỗi
6	txt2im.m	Mã script Matlab thực hiện việc chuyển đổi text file RGB thành file hình ảnh
7	clear.bat	Script file thực hiện xóa các kết quả mô phỏng cũ
8	run.bat	Script file thực hiện toàn bộ quá trình mô phỏng sử dụng ModelSim và Matlab

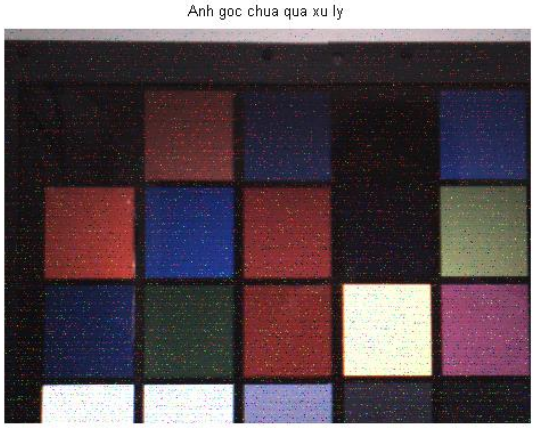
9	waveform.bat	Script file thực hiện hiển thị dạng sóng mô phỏng trên ModelSim

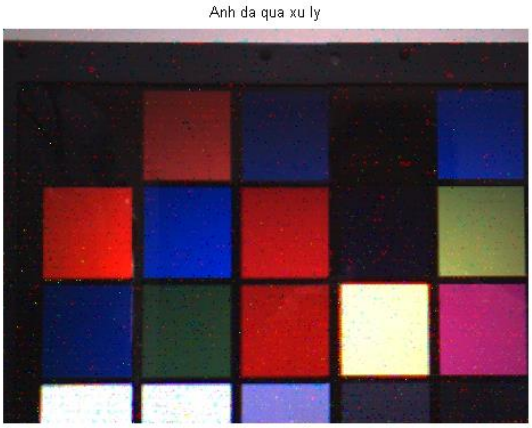
Các dạng sóng kết quả của chương trình ModelSim được kiểm tra để bảo đảm tín hiệu giao tiếp chuẩn Avalon-ST thực hiện đúng chức năng.

Các bước kiểm tra bao gồm:

- Sau khi reset hệ thống và tín hiệu ready đạt được mức tích cực (nghĩa là sink sẵn sàng nhận dữ liệu), tín hiệu valid và startofpacket đạt tích cực, dữ liệu lúc này bắt đầu được truyền đi (hình minh họa 4-3)
- Khi ready mất tích cực, nghĩa là sink không sẵn sàng nhận dữ liệu), tín hiệu valid và data cũng mất sau 1 chu kỳ clock (vì ready latency = 1). Khi ready tích cực trở lại, tín hiệu valid lên mức tích cực và data tiếp tục truyền (hình minh họa 4-4).
- Nếu dữ liệu từ lõi IP không hợp lệ (hoặc dữ liệu chưa có), thì tín hiệu valid mất mức tích cực (hình 4-5).
- Khi lõi IP xử lý dữ liệu xong (có dữ liệu hợp lệ), thì tín hiệu valid đạt tích cực đồng thời với dữ liệu truyền đi, tín hiệu startofpacket tích cực với dữ liệu đầu tiên (hình 4-6).
- Khi lõi IP truyền dữ liệu cuối cùng thì tín hiệu endofpacket đạt tích cực trong 1 chu kỳ clock.

- Kiểm tra trên ModelSim





KẾT LUẬN

Trong luận văn này, em đã nghiên cứu các kỹ thuật xử lý dữ liệu đa phương tiện, công nghệ lập trình FPGA và tìm hiểu kỹ thuật pipeline trong công nghệ FPGA, ngôn ngữ mô tả phần cứng verilog. Trong quá trình thực hiện luận văn em đã thu được những kết quả sau:

- Bước đầu đã nắm được kiến thức cơ bản về công nghệ PPGA, hiểu được tư tưởng luồng thiết kế trên công nghệ FPGA, khả năng xử lý dữ liệu của công nghệ FPGA.
- Hiểu và lập trình được bằng ngôn ngữ mô tả phần cứng Verilog, sử dụng được ngôn ngữ verilog để thiết kế lõi IP xử lý một trong nhiều loại dữ liệu đa phương tiện là xử lý hình ảnh.
- Nắm được cách sử dụng các công cụ lập trình như phần mềm Quartus II, phần mềm mô phỏng ModelSim

Những hạn chế và hướng phát triển của đề tài:

- Do thời gian thực hiện đề tài có hạn nên em mới chỉ thực hiện được xử lý ảnh trên các công cụ mô phỏng chưa thiết kế được vi mạch, chưa hoàn toàn hiểu hết các chức năng của các công cụ.
- Trong thời gian tới em sẽ tiếp tục hoàn thiện đề tài của mình để có được một sản phẩm hoàn chỉnh được thiết kế trên công nghệ FPGA.