**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**ĐỒ ÁN TỐT NGHIỆP**

**Tìm hiểm và triển khai thuật toán xử lý ảnh Sift trên FPGA**

**NGUYỄN VĂN A**

nguyenvanabc@sis.hust.edu.vn

**Ngành Công nghệ Thông tin**

**Chuyên ngành Kỹ thuật máy tính**

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn:** | PGS. TS. Phạm Văn ABC  Chữ ký của GVHD |
| **Bộ môn:** | Kỹ thuật máy tính |
| **Viện:** | Công nghệ Thông tin và Truyền thông |
| **HÀ NỘI, 12/2020** | |

**PHIẾU GIAO NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP**

**1. Thông tin về sinh viên**

Họ và tên sinh viên: Nguyễn Văn Hùng

Điện thoại liên lạc: 0326.540.044 Email:hung.nv161972@sis.hust.edu.vn

Lớp: CNTT1.01 K61 Hệ đào tạo: Đại học chính quy

Họ và tên sinh viên: Nguyễn Hữu Tráng

Điện thoại liên lạc: 0352408617 Email:trang.nh164196@sis.hust.edu.vn

Lớp: CNTT1.02 K61 Hệ đào tạo: Đại học chính quy

Đồ án tốt nghiệp được thực hiện tại: Đại học Bách khoa Hà Nội

Thời gian làm ĐATN: Từ ngày 20/09/2020 đến 30/12/2020

**2. Mục đích nội dung của ĐATN**

* Tìm hiểu về các IP Core được cung cấp sẵn trong Qsys.
* Tìm hiểu về cách truyền thông tin giữa FPGA và PC.
* Tìm hiểu quá trình thiết kế IP Core sử dụng công cụ DSP Builder, Simulink.
* Xây dựng, thiết kế IP Core thực hiện thuật toán Sift để phát hiện một logo nào đó trong một video.

**3. Các nhiệm vụ cụ thể của ĐATN**

* Tìm hiểu về giao thức PCIe sử dụng trong truyền thông tin giữa FPGA và PC.
* Tìm hiểu về các IP Core, Interface dùng cho xử lý ảnh/video được cung cấp sẵn trong Qsys.
* Tìm hiểu và học sử dụng các công cụ DSP Builder, Simulink, MATLAB.
* Tìm hiểu về các thuật toán Sift.
* Triển khai thuật toán Sift trên FPGA để phát hiện logo trong video, so sánh kết quả khi thực hiện thuật toán bằng OpenCV.
* Tổng kết và đánh giá.

**4. Lời cam đoan của sinh viên:**

Tôi – *Nguyễn Văn Hùng* và Tôi – *Nguyễn Hữu Tráng* – cam kết ĐATN là công trình nghiên cứu của bản than chúng tôi dưới sự hướng dẫn của *TS Đỗ Công Thuần* và *ThS Nguyễn Đức Tiến*.

Các kết quả nêu trong ĐATN là trung thực, không phải là sao chép toàn văn của bất kỳ công trình nào khác.

|  |  |
| --- | --- |
| *Hà Nội, ngày 30 tháng 12 năm 2020*  Tác giả ĐATN  *Nguyễn Văn Hùng* | *Hà Nội, ngày 30 tháng 12 năm 2020*  Tác giả ĐATN  *Lê Thanh Tùng* |

**5. Xác nhận của giáo viên hướng dẫn về mức độ hoàn thành của ĐATN và cho phép bảo vệ:**

|  |  |
| --- | --- |
| *Hà Nội, ngày 30 tháng 12 năm 2020*  Giảng viên hướng dẫn  *TS Đỗ Công Thuần* | *Hà Nội, ngày 30 tháng 12 năm 2020*  Giảng viên hướng dẫn  *ThS Nguyễn Đức Tiến* |

**Lời cảm ơn**

Chúng em xin chân thành gửi lời cảm ơn tới:

Toàn thể các thầy cô trường Đại học Bách khoa Hà Nội đã truyền thụ cho chúng em những kiến thức, kinh nghiệm quý báu trong suốt thời gian chúng em học tập và rèn luyện tại trường.

Thầy Nguyễn Đức Tiến và thầy Đỗ Công Thuần đã nhiệt tình hướng dẫn chúng em hoàn thành đồ án tốt nghiệp.

Anh Nguyễn xx Dũng – kỹ sư tại … đã nhiệt tình giúp đỡ chúng em ngay từ những ngày đầu thực hiện đồ án tốt nghiệp.

Ngoài ra, chúng em xin được gửi lời cảm ơn đến gia đình, bạn bè đã luôn ở bên động viên, tạo điều kiện cho chúng em có thể hoàn thành đồ án tốt nghiệp này.

Do thời gian và kiến thức còn hạn chế nên đồ án này khó tránh khỏi có nhiều thiếu sót, chúng em rất mong nhận được sự đóng góp ý kiến của các thầy cô và các bạn.

**Tóm tắt nội dung đồ án**

Mục tiêu chính của đồ án là triển khai thuật toán Sift trên nền tảng FPGA để giải quyết bài toán phát hiện vị trí logo trong một video.

Nội dung chính của đồ án gồm:

**Chương 1: AVALON INTERFACE**

Chương này trình bày về Avalon Interface được sử dụng phổ biến trong các IP Core được cung cấp sẵn bởi Altera.

**Chương 2: BOARD DE2i-150**

Chương này tập trung khảo sát kiến trúc, ngoại vi cũng như cách truyền thông tin giữa phần FPGA và phần PC của board DE2i-150 dùng để chạy thử nghiệm.

**Chương 3: KIẾN TRÚC HỆ THỐNG**

Chương này sẽ trình bày kiến trúc tổng quan của hệ thống, cho thấy luồng di chuyển của hình ảnh từ đầu vào tới đầu ra.

**Chương 4: THUẬT TOÁN SIFT**

Chương này trình bày lý thuyết cơ sở, các bước thực thi và các công thức của thuật toán Sift.

**Chương 5: TRIỂN KHAI THUẬT TOÁN TRÊN FPGA**

Chương này trình bày về công cụ DSP Builder, các sơ đồ thiết kế của thuật toán từ lớn (tổng quan) đến nhỏ (chi tiết), xây dựng testbench trên MATLAB để kiểm tra tính đúng đắn.

**Chương 6: KẾT QUẢ THỬ NGHIỆM**

Chương này trình bày các đánh giá, so sánh kết quả, tài nguyên đã sử dụng trên board DE2i-150.

Sinh viên thực hiện

Ký và ghi rõ họ tên

**MỤC LỤC**

[PHIẾU GIAO NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP ii](#_Toc59200183)

[CHƯƠNG 1. GIAO DIỆN AVALON 1](#_Toc59200184)

[1.1 Tổng quan về giao diện Avalon 1](#_Toc59200185)

[1.2 Avalon Streaming Interface 2](#_Toc59200186)

[1.3 Sử dụng Avalon – ST cho truyền nhận dữ liệu ảnh/video 3](#_Toc59200187)

[CHƯƠNG 2. BOARD DE2i-150 6](#_Toc59200188)

[2.1 Kiến trúc và ngoại vi của board DE2i-150 6](#_Toc59200189)

[2.2 Trao đổi thông tin giữa FPGA (Cyclone IV GX) và PC (Atom) 6](#_Toc59200190)

[CHƯƠNG 3. KIẾN TRÚC HỆ THỐNG 7](#_Toc59200191)

[3.1 Sơ đồ kiến trúc tổng quan hệ thống 7](#_Toc59200192)

[3.2 Các IP Core trên Qsys cho board DE2i-150 7](#_Toc59200193)

[3.2.1 IP Frame Reader 7](#_Toc59200194)

[3.2.2 IP Clocked Video Output 7](#_Toc59200195)

[3.3 Luồng di chuyển của dữ liệu trên hệ thống 8](#_Toc59200196)

[CHƯƠNG 4. THUẬT TOÁN SIFT 9](#_Toc59200197)

[4.1 Phép toán Convolution(tên này ông có thể đặt lại theo ý ông nhá) 9](#_Toc59200198)

[4.2 Xây dựng Gaussian scale-space (tên này ông có thể đặt lại theo ý ông nhá) 9](#_Toc59200199)

[4.3 DoG – Difference of Gaussians 9](#_Toc59200200)

[4.4 Tìm các điểm đặc trưng (Keypoint) 9](#_Toc59200201)

[4.5 Tính hướng tham chiếu cho mỗi điểm đặc trưng(cái này ông chọn sift hay brief thì ghi rõ nhá) 9](#_Toc59200202)

[4.6 Xây dựng bộ mô tả đặc trưng(cái này ông chọn sift hay brief thì ghi rõ nhá) 9](#_Toc59200203)

[4.7 So khớp keypoint 9](#_Toc59200204)

[CHƯƠNG 5. TRIỂN KHAI THUẬT TOÁN TRÊN FPGA 10](#_Toc59200205)

[5.1 Công cụ Altera DSP Builder 10](#_Toc59200206)

[5.1.1 Giới thiệu công cụ Altera DSP Builder 10](#_Toc59200207)

[5.1.2 Một vài IP Core trong Altera DSP Builder được sử dụng 10](#_Toc59200208)

[5.2 Triển khai thuật toán trên FPGA 10](#_Toc59200209)

[5.2.1 IP Core Convolution 10](#_Toc59200210)

[5.2.2 IP Core Gaussion scale-spcae 10](#_Toc59200211)

[5.2.3 IP Core Sift extraction (cái này chính là cái làm xong ra được keypiont đó ông) 10](#_Toc59200212)

[5.2.4 IP Core … description 10](#_Toc59200213)

[5.2.5 IP Core matching 10](#_Toc59200214)

[5.3 Kiểm thử với Testbench trên MATLAB 10](#_Toc59200215)

[CHƯƠNG 6. KẾT QUẢ THỬ NGHIỆM 11](#_Toc59200216)

[6.1 Kết quả thử nghiệm 11](#_Toc59200217)

[6.2 Thông số tài nguyên FPGA, so sánh với thực hiện bằng CPU 11](#_Toc59200218)

[TÀI LIỆU THAM KHẢO 12](#_Toc59200219)

[PHỤ LỤC 13](#_Toc59200220)

**DANH MỤC HÌNH VẼ**

[Hình 2.1 Đồ thị kiểu bánh 20](#_Toc20580104)

[Hình 2.2 Đồ thị kiểu thanh ngang 20](#_Toc20580105)

[Hình 2.3 Đồ thị kiểu cột đứng 21](#_Toc20580106)

[Hình 2.4 Đồ thị kiểu đường 21](#_Toc20580107)

[Hình 2.5 Đồ thị kiểu diện tích 22](#_Toc20580108)

**DANH MỤC HÌNH VẼ**

[Bảng 1.1 Thống kê các thiết bị và giá thành 8](#_Toc20580109)

# GIAO DIỆN AVALON

## Tổng quan về giao diện Avalon

Giao diện Avalon đơn giản hóa việc thiết kế hệ thống bằng cách cho phép ghép nối các components trong FPGA của Intel. Họ các giao diện Avalon định nghĩa các giao diện thích hợp truyền dữ liệu tốc độ cao, đọc ghi các thanh ghi và bộ nhớ cũng như điều khiển các thiết bị ngoại vi,…

Bảng . Các giao diện trong họ giao diện Avalon

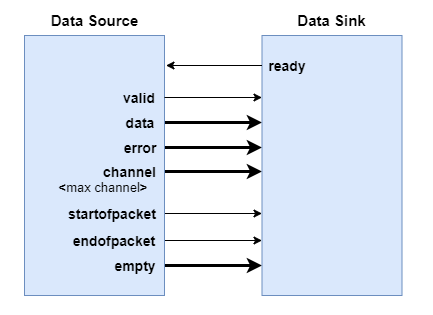
|  |  |
| --- | --- |
| **Tên giao diện** | **Đặc điểm** |
| Avalon Streaming Interface (Avalon-ST) | Hỗ trợ truyền dữ liệu với băng thông cao, độ trễ thấp, một chiều, không quan tâm đến địa chỉ, bao gồm các luồng, các gói và các tín hiệu số được ghép kênh với nhau. Thích hợp truyền dữ liệu ảnh, video, các tín hiệu số,… |
| Avalon Memory Mapped Interface (Avalon-MM) | Sử dụng để đọc / ghi dữ liệu dựa trên địa chỉ trong mô hình kết nối master – slave điển hình như vi điều khiển (Microprocessors), bộ nhớ (Memories), thanh ghi, UARTs, DMAs, các bộ định thời (Timers),… |
| Avalon Conduit Interface | Giao diện chứa các tín hiệu riêng lẻ hoặc các nhóm tín hiệu không tương thích với với các giao diện Avalon nào khác. Chúng có thể được kết nối với các giao diện Conduit khác trong cùng Platform Designer, ngoài ra cũng có thể xuất và kết nối chúng với các mô-đun khác hoặc các ngoại vi như switch, button,… của FPGA. |
| Avalon Tri-State Conduit Interface (Avalon-TC) | Hỗ trợ kết nối với các thiết bị ngoại vi bên ngoài FPGA. Thích hợp sử dụng ở đầu vào hoặc đầu ra. |
| Avalon Interrupt Interface | Cho phép gửi tín hiệu ngắt đến các components khác. |
| Avalon Clock Interface | Điều khiển hoặc nhận tín hiệu clock. |
| Avalon Reset Interface | Cung cấp kết nối cho tín hiệu reset. |

Chi tiết về các giao diện trên được mô tả trong tài liệu ***Avalon® Interface Specifications*** được cung cấp bởi Intel.

## Avalon Streaming Interface

Như đã đề cập ở phần trên, Avalon Streaming Interface (Avalon-ST) cho phép truyền dữ liệu với băng thông cao, độ trễ thấp, truyền theo một chiều,…, thích hợp sử dụng truyền tín hiệu hình ảnh, video, tín hiệu số. Trên thực tế, hầu hết các IP Core xử lý dữ liệu dạng ảnh/video được cung cấp bởi Altera (Intel) đều sử dụng giao diện này. Để sử dụng cũng như thiết kế các IP Core xử lý ảnh/video hay xử lý tín hiệu số trên FPGA của Altera (Intel), cần thiết phải tìm hiểu rõ về giao diện này.

Vì đồ án tốt nghiệp xử lý tín hiệu ảnh/video nên sẽ tập trung chủ yếu vào tìm hiểu chi tiết giao diện Avalon – ST cho trường hợp truyền các gói tin (packets). Avalon – ST truyền dữ liệu một chiều từ Source sang Sink theo sơ đồ sau:

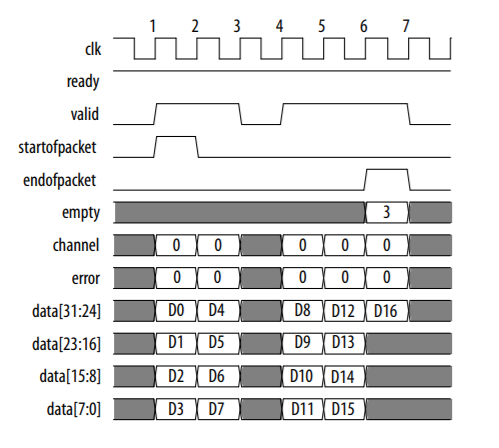


Hình . Tín hiệu khi truyền gói (packet) bằng giao diện Avalon – ST

Bảng . Chi tiết các tín hiệu truyền gói của Avalon - ST

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên tín hiệu** | **Độ rộng bus** | **Chiều truyền** | **Mô tả đặc điểm** |
| channel | 1 – 128 | Source → Sink | Số kênh dữ liệu được truyền trong chu kì hiện tại. |
| data | 1 – 8192 | Source → Sink | Dữ liệu mang phần lớn thông tin được truyền từ source đến sink |
| error | 1 – 256 | Source → Sink | Mặt nạ bit dùng đánh dấu các lỗi ảnh hưởng đến dữ liệu đang truyền. |
| ready | 1 | *Sink → Source* | Báo hiệu phía sink có sẵn sàng để nhận dữ liệu hay không. |
| valid | 1 | Source → Sink | Báo hiệu các tín hiệu đang truyền từ source đến sink là hợp lệ. |
| empty | 1 – 10 | Source → Sink | Cho biết số lượng symbols là trống, không đại diện cho dữ liệu hợp lệ. |
| End of packet | 1 | Source → Sink | Đánh dấu kết thúc một gói tin. |
| Start of packet | 1 | Source → Sink | Đánh dấu bắt đầu một gói tin. |

Để hiểu rõ hơn về cách hoạt động của giao diện Avalon – ST, chúng ta xem xét biểu đồ thời gian minh họa quá trình truyền dữ liệu song song:

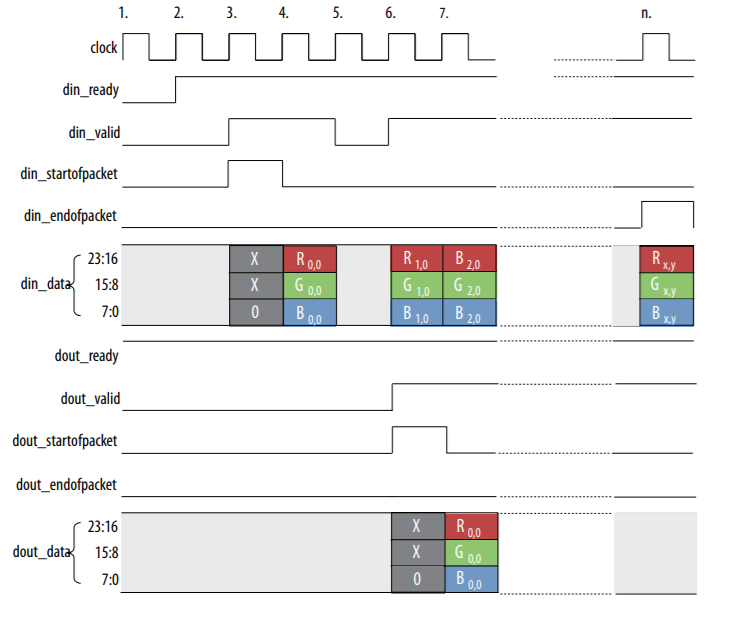


Hình . Biểu đồ thời gian minh họa quá trình truyền nhận dữ liệu Avalon - ST

Tại các chu kỳ 1, 2, 4, 5, 6 cả tín hiệu *ready* và *valid* đều ở mức cao (logic ‘1’) nên dữ liệu được truyền từ ***source*** sang ***sink***. Tại chu kỳ 3, tín hiệu *valid* ở mức thấp (logic ‘0’) báo hiệu dữ liệu đang truyền không hợp lệ nên tại chu kỳ này sẽ được coi như dữ liệu không được truyền đi. Tại chu kỳ 1, tín hiệu *startofpacket* ở mức cao (logic ‘1’) báo hiệu một gói mới bắt đầu và 4 byte đầu tiên của gói tin này được chuyển song song thông qua tín hiệu *data*. Ở chu kỳ 6, tín hiệu *endofpacket* ở mức cao (logic ‘1’) báo hiệu gói tin đang truyền kết thúc, đồng thời, tín hiệu *empty* đạt giá trị 3 thể hiện đây là cuối gói và 3 trong 4 symbols của tín hiệu *data* là trống (chỉ có D16 mà không có D17, D18 và D19).

## Sử dụng Avalon – ST cho truyền nhận dữ liệu ảnh/video

Trong đồ án tốt nghiệp này, dữ liệu cần xử lý là dữ liệu ảnh/video với định dạng màu RGB và dùng mỗi 8 bit để biểu diễn giá trị một kênh màu. Việc sử dụng một số IP Core có sẵn do Altera (Intel) cung cấp hay thiết kế các IP Core xử lý cho các thuật toán đều sử dụng giao diện Avalon – ST để truyền nhận dữ liệu theo gói, truyền song song dữ liệu 3 kênh màu cũng lúc, có nghĩa là với mỗi chu kỳ mà 2 tín hiệu ready và valid đều ở mức cao (logic ‘1’) thì dữ liệu của một pixel được truyền thành công. Ta xét một ví dụ về một IP Core có một cổng Avalon – ST tên là ***din*** và một cổng Avalon – ST tên là ***dout***. Dữ liệu sẽ đi vào IP Core từ cổng ***din*** và đi ra khỏi IP Core thông qua cổng ***dout***.

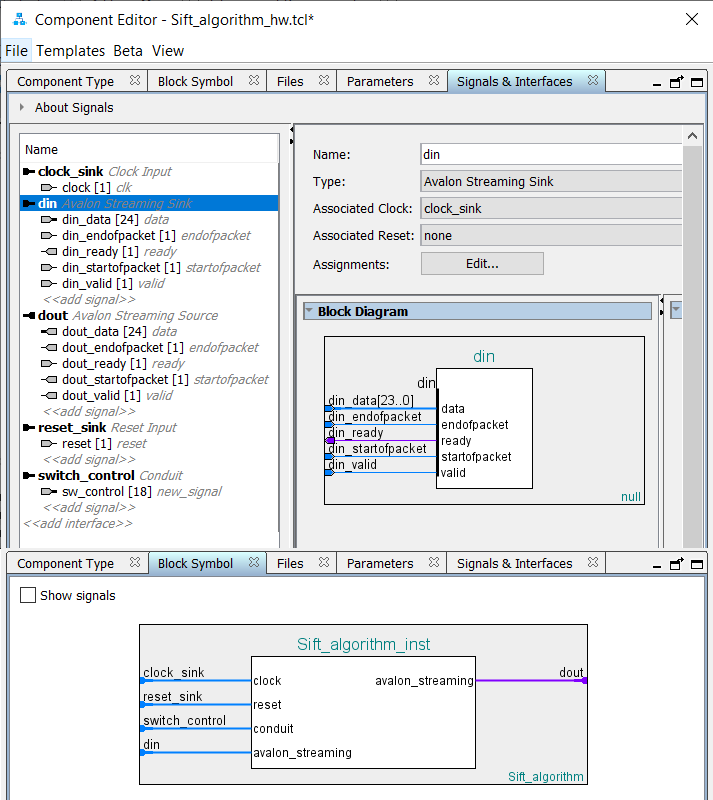


Hình . Biểu đồ thời gian minh họa truyền nhận dữ liệu ảnh / video bằng Avalon-ST

Mỗi cổng ***din*** và ***dout*** đều chứa 5 tín hiệu: *ready*, *valid*, *data*, *startofpacket*, *endofpacket*. Tín hiệu *din\_ready* báo hiệu rằng IP Core sẵn sàng nhận dữ liệu, nó là một đầu ra (output) của IP Core được gắn vào cổng ***din***. Các tín hiệu còn lại của cổng ***din*** đều là đầu vào (input) của IP Core. Tín hiệu *din\_data* truyền dữ liệu các pixel, 24 bit mang thông tin các kênh màu RGB của từng pixel sẽ được truyền song song tại mỗi chu kỳ clock nếu cả *din\_ready* và *din\_valid* đều ở mức cao (logic ‘1’). Tín hiệu *din\_ startofpacket* báo hiệu bắt đầu một packet mới và tín hiệu *din\_ endofpacket* báo hiệu kết thúc một packet. Các tín hiệu của cổng ***dout*** tương tự như cổng ***din*** nhưng ngữ nghĩa ngược lại.

Xem xét biểu đồ trên theo thời gian, ta thấy ban đầu *din\_ready* ở mức logic ‘0', báo hiệu rằng IP Core không sẵn sàng để nhận dữ liệu từ cổng ***din*** và không có dữ liệu các pixel được truyền vào. Đến chu kỳ 2, *din\_ready* được nâng lên mức logic ‘1’, IP Core sẵn sàng nhận dữ liệu sau một chu kỳ clock. Một chu kỳ trễ này được quyết định bởi tham số tên *ready latency*, có thể cài đặt khi thiết kế và tất cả các IP Core trong bộ *Video and Image Processing* được cung cấp bởi Altera đều đặt giá trị tham số này là 1. Tại chu kỳ 3, *din\_valid* được nâng lên logic ‘1’ để báo hiệu dữ liệu hình ảnh/video được gửi qua *din\_data*. Đồng thời, *din\_ startofpacket* được nâng lên logic ‘1’ để báo hiệu giá trị được truyền là giá trị đầu tiên của một gói (pixel ở góc trên bên trái). Ở đây giá trị này bằng 0, có nghĩa là gói này là dữ liệu video. Chu kỳ 4, *din\_valid* vẫn giữ ở mức logic ‘1’, *din\_ startofpacket* thì bị kéo xuống logic ‘0’, báo hiệu rằng đang gửi phần thân gói vào IP Core. Chu kỳ 5, *din\_ready* vẫn ở logic ‘1’ báo hiệu IP Core vẫn sẵn sàng nhận dữ liệu, nhưng có thể do không có dữ liệu để gửi nên *din\_valid* bị kéo xuống logic ‘0’ để báo hiệu dữ liệu này không hợp lệ. Ở chu kỳ 6, việc truyền dữ liệu tiếp tục diễn ra bình thường và *din\_valid* được nâng lên logic ‘1’. Đồng thời, IP Core bắt đầu truyền dữ liệu qua cổng ***dout*** (trong ví dụ này giả định IP Core có độ trễ bên trong là 3 chu kỳ clock). Đến chu kỳ n, *din\_endofpacket* được đặt thành logic ‘1’ trong 1 chu kỳ báo hiệu đã truyền xong pixel cuối cùng của gói (pixel ở góc dưới bên phải).

Trong đồ án này, để thuận tiện cho việc ghép nối giữa các IP Core, IP Core thuật toán sẽ được thiết kế tương tự ví dụ trên với các tín hiệu vào ra như hình sau:



Hình . Thiết kế giao diện Avalon – ST cho IP Core thuật toán trên Qsys

IP Core cho thuật toán Sift sẽ được thiết kế với các đầu vào và đầu ra như hình trên. Về cơ bản nó khá giống như ví dụ vừa phân tích phía trên với đầu vào ***din*** và đầu ra ***dout*** đều sử dụng giao diện Avalon – ST, ngoài ra còn có thêm cổng ***clock\_sink*** sử dụng giao diện Avalon Clock, cổng ***reset\_sink*** sử dụng giao diện Avalon Reset để truyền các tín hiệu clock và reset cho IP Core. Bên cạnh đó, thiết kế còn có thêm cổng ***switch\_control*** với giao diện Avalon Conduit để truyền tín hiệu ngoại vi từ các switch trên board DE2i-150 dùng điều khiển, cấu hình trong IP Core.

# BOARD DE2i-150

## Kiến trúc và ngoại vi của board DE2i-150

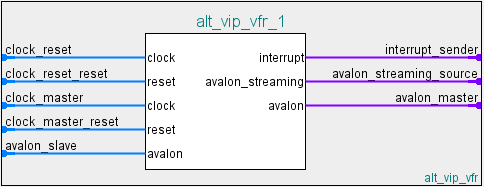
## Trao đổi thông tin giữa FPGA (Cyclone IV GX) và PC (Atom)

# KIẾN TRÚC HỆ THỐNG

## Sơ đồ kiến trúc tổng quan hệ thống

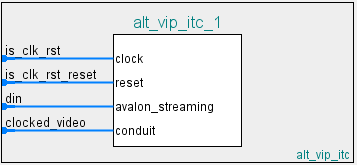
## Các IP Core trên Qsys cho board DE2i-150

### IP Frame Reader



IP Frame Reader đọc các khung hình (frames video) từ bộ nhớ ngoài qua cỏng avalon\_master theo giao diện Avalon-MM và xuất chúng dưới dạng một luồng video qua cổng avalon\_streaming\_source theo giao diện Avalon – ST. Cổng avlon\_slave cung cấp các dữ liệu cấu hình cho IP Core. Các khung hình (frames video) được lưu trữ trong bộ nhớ ngoài dưới dạng dữ liệu video thô (chỉ bao gồm giá trị các pixel), ngay trước khi Frame Reader IP Core đọc những khung hình từ bộ nhớ ngoài, nó tạo ra một gói (packer) điều khiển và tiêu đề cho gói dữ liệu video được đẩy ra giao diện Avalon – ST. Sau đó, dữ liệu video từ bộ nhớ ngoài sẽ được truyền trực tiếp qua IP Core. Nội dung của khối điều khiển dữ liệu được thiết lập thông qua cổng avalon\_slave. Quá trình này được lặp lại với mọi khung hình được đọc từ bộ nhớ ngoài.

### IP Clocked Video Output



IP Clocked Video Output chuyển đổi video ở định dạng Avalon – ST sang các định dạng video tiêu chuẩn như BT656 hay VGA. Đầu vào của IP Core (din) là một giao diện Avalon – ST, video ở định dạng Avalon – ST sẽ đi vào IP Core từ cổng din, sau khi chuyển đổi hoàn thành, video định dạng tiêu chuẩn như BT656 hay VGA sẽ được xuất ra cổng clocked video theo giao diện Avalon Conduit. Video qua chuyển đổi này có thể đưa vào các bộ giải mã tương ứng như VGA để giải mã và hiển thị ra màn hình.

## Luồng di chuyển của dữ liệu trên hệ thống

# THUẬT TOÁN SIFT

## Phép toán Convolution(tên này ông có thể đặt lại theo ý ông nhá)

## Xây dựng Gaussian scale-space (tên này ông có thể đặt lại theo ý ông nhá)

## DoG – Difference of Gaussians

## Tìm các điểm đặc trưng (Keypoint)

## Tính hướng tham chiếu cho mỗi điểm đặc trưng(cái này ông chọn sift hay brief thì ghi rõ nhá)

## Xây dựng bộ mô tả đặc trưng(cái này ông chọn sift hay brief thì ghi rõ nhá)

## So khớp keypoint

# TRIỂN KHAI THUẬT TOÁN TRÊN FPGA

## Công cụ Altera DSP Builder

### Giới thiệu công cụ Altera DSP Builder

### Một vài IP Core trong Altera DSP Builder được sử dụng

## Triển khai thuật toán trên FPGA

### IP Core Convolution

### IP Core Gaussion scale-spcae

### IP Core Sift extraction (cái này chính là cái làm xong ra được keypiont đó ông)

### IP Core … description

### IP Core matching

## Kiểm thử với Testbench trên MATLAB

# KẾT QUẢ THỬ NGHIỆM

## Kết quả thử nghiệm

## Thông số tài nguyên FPGA, so sánh với thực hiện bằng CPU

# TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| [1] | Trần Bách, Lưới điện và hệ thống điện, Nhà xuất bản Khoa học Kỹ thuật, 2004. |
| [2] | Abe Masayuki, “A Practical Approach to Accurate Fault Location on Extra High Voltage Teed Feeders,” *IEEE Transaction on Power Delivery,* pp. 159-168, 1995. |
| [3] | Microsoft, "Add citations in a Word document," 2017. |

# PHỤ LỤC

1. **Chi tiết số liệu thí nghiệm**

Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có).

1. **Chi tiết các bước tính toán**

Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có).

1. **Chi tiết sơ đồ mô phỏng**

Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trìn