ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT
KHOA ĐIỆN – ĐIỆN TỬ

ĐỒ ÁN TỐT NGHIỆP
ĐẠI HỌC

NGÀNH: ĐIỆN - ĐIỆN TỬ

CHUYÊN NGÀNH: KỸ THUẬT ĐIỆN TỬ

ĐỀ TÀI:

NGHIÊN CỨU, THIẾT KẾ IP CORE NETFLOW V5 TRÊN NỀN TẢNG FPGA ĐỂ PHÂN TÍCH VÀ GIÁM SÁT MẠNG

Người hướng dẫn : **TS. Trần Hoàng Vũ**
Sinh viên thực hiện : **Nguyễn Duy Hà Sơn**
Đặng Sỹ Phi Hùng
Mã Sinh viên : 1711505110126
1711505210110
Lớp : **17KTDT**

Đà Nẵng, 8/2021

Nguyễn Duy Hà Sơn & Đặng Sỹ Phi Hùng TÊN ĐỀ TÀI : NGHIÊN CỨU, THIẾT KẾ IP CORE NETFLOW V5 TRÊN NỀN TẢNG FPGA ĐỂ PHÂN TÍCH VÀ GIÁM SÁT MẠNG Năm: 2021

ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT
KHOA ĐIỆN – ĐIỆN TỬ

ĐỒ ÁN TỐT NGHIỆP
ĐẠI HỌC

NGÀNH: ĐIỆN – ĐIỆN TỬ

CHUYÊN NGÀNH: KỸ THUẬT ĐIỆN TỬ

ĐỀ TÀI:

NGHIÊN CỨU, THIẾT KẾ IP CORE NETFLOW V5 TRÊN NỀN TẢNG FPGA ĐỂ PHÂN TÍCH VÀ GIÁM SÁT MẠNG

Người hướng dẫn : **TS. Trần Hoàng Vũ**
Sinh viên thực hiện : **Nguyễn Duy Hà Sơn**
Đặng Sỹ Phi Hùng
Mã Sinh viên : 1711505110126
1711505210110
Lớp : **17KTDT**

Đà Nẵng, 8/2021

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT
KHOA ĐIỆN – ĐIỆN TỬ

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

NHẬN XÉT ĐỒ ÁN TỐT NGHIỆP
(*Dành cho người hướng dẫn*)

I. Thông tin chung:

1. Họ và tên sinh viên: NGUYỄN DUY HÀ SƠN
2. Lớp: 17KTDT1 Mã SV: 1711505110126
3. Tên đề tài: NGHIÊN CỨU, THIẾT KẾ IP CORE NETFLOW V5 TRÊN NỀN TẢNG FPGA ĐỂ PHÂN TÍCH VÀ GIÁM SÁT MẠNG
4. Người hướng dẫn: TS. Trần Hoàng Vũ Học hàm/ học vị: Tiến sĩ

II. Nhận xét, đánh giá đồ án tốt nghiệp:

1. Về tính cấp thiết, tính mới, mục tiêu của đề tài: (0,9đ)

a. Tính cấp thiết:

Việc giám sát, vận hành, điều phối hệ thống mạng Thông tin – Dữ liệu lớn là vấn đề luôn luôn mới và cấp bách đối Chính Phủ điện tử, các doanh nghiệp, tổ chức có hệ thống CNTT lớn. Với sự tiến bộ khoa học kỹ thuật như Mạng băng thông rộng 5G, đường truyền Quang tốc độ lớn, kỹ thuật vi mạch bán dẫn tích hợp mật độ các bóng bán dẫn cực lớn, công nghệ IoT, dẫn đến lượng dữ liệu khổng lồ và tốc độ dữ liệu cao, điều đó đặt ra vấn đề cấp bách là nhà vận hành hệ thống mạng phải có công cụ dùng để phân tích nhanh và chính xác các diễn biến trên hệ thống, phân tích các lưu lượng bất thường, phân tích các kết nối bất thường, hay phân tích các hành vi bất thường để có các giải pháp ngăn chặn kịp thời và chính xác. Hiện nay có khá nhiều phần mềm thương mại có chức năng phân tích, giám sát các vấn đề bất thường nêu trên. Tuy nhiên một số vấn đề đặt ra liên quan đến kỹ thuật, bảo mật và kinh phí vận hành. Dữ liệu lớn, gồm nhiều giao thức và phức tạp thì đòi hỏi phải mua và sử dụng các phần mềm bản quyền, đi kèm với nó là chi phí lớn để duy trì hằng năm. Dữ liệu lớn, bằng thông cao, theo dõi cho nhiều Node mạng, các Sever tầm trung hầu như không xử lý kịp nên phải cần các thiết bị mạng đặc chủng và Sever mạnh. Và đây là bài toán kinh tế hóc búa, vừa phải tốn chi phí quyền để sử dụng và duy trì (Splunk; Syslog-Ng; ArcSight Logger) vừa phải đầu tư hệ thốn Sever tầm cao để đáp ứng tốc độ truyền tải của băng thông.

Với những bất cập được nêu trên, giải pháp của đề tài ***"Nghiên cứu, thiết kế IP Core NetFlow V5 trên nền tảng FPGA để phân tích và giám sát mạng"*** *mang tính cấp thiết và thực tế.*

b. Tính mới:

Đề tài đề cập đến việc sử dụng kết hợp giữa phần cứng FPGA và phần mềm Splunk để xây dựng một hệ thống giám sát mạng tại các hệ thống dữ liệu lớn nhằm mục tiêu giảm chi phí đầu tư duy trì vừa làm chủ công nghệ.

Trên nền tảng FPGA, việc thiết kế IP Core NetFLow có thể xử lý song song đa luồng, tùy chỉnh - tái cấu trúc có thể cập nhật, thay đổi về các chức năng theo mục đích của khách hàng.

c. Mục tiêu:

Thiết kế IP Core NetFlow V5 trên nền tảng FPGA thực hiện việc phân tích nguồn dữ liệu thô internet sau đó trích xuất các trường thông tin cần thiết như IP Nguồn, IP Đích, Port Nguồn, Port Đích, Protocol và các các thông tin kèm theo (lưu lượng bytes, thời gian) đưa về cho người quản trị sử dụng để giám sát và thống kê.

Sau khi trích xuất các trường, hệ thống sẽ liên kết với phần mềm giám sát mạng (Splunk,..), ở đây phần mềm sẽ sử dụng các trường thông tin này để thống kê, vẽ biểu đồ giám sát. Phục vụ cho việc giám sát và quản lý mạng của người quản trị trong hệ thống mạng.

2. Về kết quả giải quyết các nội dung nhiệm vụ yêu cầu của đồ án: (3,8đ)

Nắm được cơ sở lý thuyết về giao thức NetFlow, về các giao thức – gói tin mạng, về phương pháp thiết kế IP Core trên nền tảng FPGA và thực hiện lập trình mô tả phần cứng với ngôn ngữ Verilog.

Hoàn thành việc mô tả thiết kế NetFlow V5 bằng ngôn ngữ Verilog và kiểm tra chức năng các modules: Rx_queus, Classification, Export_Expired.

Hoàn thành tích hợp hệ thống IP Core NetFlow V5 với các ngoại vi (FMC 4 Port, RTC, IIC, UART), tạo file .bit để nhúng hệ thống xuống board ML605.

Hoàn thành kiểm tra và testing hệ thống NetFLow V5 trên board ML605.

Xây dựng được chương trình đọc các dữ liệu thu thập được từ NetFlow lên Splunk Software để thu thập và vẽ biểu đồ.

Hoàn thiện báo cáo tổng kết, đánh giá và hoàn thiện sản phẩm.

3. Về hình thức, cấu trúc, bố cục của đồ án tốt nghiệp: (2đ)

Nội dung của đồ án tốt nghiệp: gồm 05 chương đáp ứng các mục tiêu của đề tài. Chương I: Tổng quan về NetFlow và công nghệ FPGA.

Chương II: Cơ sở lý thuyết và kiến trúc NetFlow V5.

Chương III: Thiết kế IP Core NetFlow và thực hiện nhúng trên FPGA của Xilinx. Chương IV: Xây dựng chương trình điều khiển và giao diện hệ thống.

Chương V: Kiểm tra, đánh giá hệ thống và thực hiện giám sát hệ thống mạng tập trung Splunk cho các Data center.

4. Kết quả đạt được, giá trị khoa học, khả năng ứng dụng của đề tài: (0,9đ)

Kết quả đạt được: IP Core NetFlow V5 thiết kế trên nền tảng FPGA phân tích và trích xuất các giá trị một cách chính, đầy đủ từ nguồn dữ liệu thô internet. Sau đó các trường dữ liệu đó được đưa lên phần mềm Splunk để người quản trị mạng có thể thống kê và giám sát.

Giá trị khoa học: Cung cấp giải pháp hoàn chỉnh phân tích đầy đủ giao thức thông dụng và phức tạp NetFlow v5 – Thực hiện thiết kế IP Core trên nền tảng FPGA theo chức năng của giao thức NetFLow để thực hiện việc phân tích và trích xuất thông tin.

Khả năng ứng dụng: Sản phẩm của đề tài giám sát và quản lý mạng mang lại các mục tiêu cho hiệu năng cao và chi phí thấp có thể ứng dụng trong các hệ thống công nghệ thông tin, các trung tâm dữ liệu của tổ chức hay công ty, …

5. Các tồn tại, thiếu sót cần bổ sung, chỉnh sửa:

Cần chỉnh sửa một số lỗi chính tả trong đồ án

III. Tinh thần, thái độ làm việc của sinh viên: (2đ)

Có tinh thần và thái độ nghiên cứu nghiêm túc, làm việc theo kế hoạch đã được quy định. Chủ động gặp gỡ và trao đổi với giảng viên hướng dẫn về đề cương và báo cáo kết quả đạt được hàng tuần.

IV. Đánh giá:

1. Điểm đánh giá: 9,6/10 (lấy đến 1 số lẻ thập phân)
2. Đề nghị: ☒ Được bảo vệ đồ án ☐ Bổ sung để bảo vệ ☐ Không được bảo vệ

Đà Nẵng, ngày 14 tháng 8 năm 2021

Người hướng dẫn

TS. Trần Hoàng Vũ

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT
KHOA ĐIỆN – ĐIỆN TỬ

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

NHẬN XÉT ĐỒ ÁN TỐT NGHIỆP
(*Dành cho người hướng dẫn*)

I. Thông tin chung:

1. Họ và tên sinh viên: ĐẶNG SỸ PHI HÙNG
2. Lớp: 17KTDT1 Mã SV: 1711505210110
3. Tên đề tài: NGHIÊN CỨU, THIẾT KẾ IP CORE NETFLOW V5 TRÊN NỀN TẢNG FPGA ĐỂ PHÂN TÍCH VÀ GIÁM SÁT MẠNG
4. Người hướng dẫn: TS. Trần Hoàng Vũ Học hàm/ học vị: Tiến sĩ

II. Nhận xét, đánh giá đồ án tốt nghiệp:

1. Về tính cấp thiết, tính mới, mục tiêu của đề tài: (0,9)

a. Tính cấp thiết:

Việc giám sát, vận hành, điều phối hệ thống mạng Thông tin – Dữ liệu lớn là vấn đề luôn luôn mới và cấp bách đối Chính Phủ điện tử, các doanh nghiệp, tổ chức có hệ thống CNTT lớn. Với sự tiến bộ khoa học kỹ thuật như Mạng băng thông rộng 5G, đường truyền Quang tốc độ lớn, kỹ thuật vi mạch bán dẫn tích hợp mật độ các bóng bán dẫn cực lớn, công nghệ IoT, dẫn đến lượng dữ liệu khổng lồ và tốc độ dữ liệu cao, điều đó đặt ra vấn đề cấp bách là nhà vận hành hệ thống mạng phải có công cụ dùng để phân tích nhanh và chính xác các diễn biến trên hệ thống, phân tích các lưu lượng bất thường, phân tích các kết nối bất thường, hay phân tích các hành vi bất thường để có các giải pháp ngăn chặn kịp thời và chính xác. Hiện nay có khá nhiều phần mềm thương mại có chức năng phân tích, giám sát các vấn đề bất thường nêu trên. Tuy nhiên một số vấn đề đặt ra liên quan đến kỹ thuật, bảo mật và kinh phí vận hành. Dữ liệu lớn, gồm nhiều giao thức và phức tạp thì đòi hỏi phải mua và sử dụng các phần mềm bản quyền, đi kèm với nó là chi phí lớn để duy trì hằng năm. Dữ liệu lớn, bằng thông cao, theo dõi cho nhiều Node mạng, các Sever tầm trung hầu như không xử lý kịp nên phải cần các thiết bị mạng đặc chủng và Sever mạnh. Và đây là bài toán kinh tế hóc búa, vừa phải tốn chi phí quyền để sử dụng và duy trì (Splunk; Syslog-Ng; ArcSight Logger) vừa phải đầu tư hệ thốn Sever tầm cao để đáp ứng tốc độ truyền tải của băng thông.

Với những bất cập được nêu trên, giải pháp ***"Nghiên cứu, thiết kế IP Core NetFlow V5 trên nền tảng FPGA để phân tích và giám sát mạng"*** *mang tính cấp thiết và thực tiễn cao.*

b. Tính mới:

Đề tài đề cập đến việc sử dụng kết hợp giữa phần cứng FPGA và phần mềm Splunk để xây dựng một hệ thống giám sát mạng tại các hệ thống dữ liệu lớn nhằm mục tiêu giảm chi phí đầu tư duy trì vừa làm chủ công nghệ.

Trên nền tảng FPGA, việc thiết kế IP Core NetFLow có thể xử lý song song đa luồng, tùy chỉnh - tái cấu trúc có thể cập nhật, thay đổi về các chức năng theo mục đích của khách hàng.

c. Mục tiêu:

Thiết kế IP Core NetFlow V5 trên nền tảng FPGA thực hiện việc phân tích nguồn dữ liệu thô internet sau đó trích xuất các trường thông tin cần thiết như IP Nguồn, IP Đích, Port Nguồn,

Port Đích, Protocol và các các thông tin kèm theo (lưu lượng bytes, thời gian) đưa về cho người quản trị sử dụng để giám sát và thống kê.

Sau khi trích xuất các trường, hệ thống sẽ liên kết với phần mềm giám sát mạng (Splunk,..), ở đây phần mềm sẽ sử dụng các trường thông tin này để thống kê, vẽ biểu đồ giám sát. Phục vụ cho việc giám sát và quản lý mạng của người quản trị trong hệ thống mạng.

2. Về kết quả giải quyết các nội dung nhiệm vụ yêu cầu của đồ án: (3,8đ)

Nắm được cơ sở lý thuyết về giao thức NetFlow, về các giao thức – gói tin mạng, về phương pháp thiết kế IP Core trên nền tảng FPGA và thực hiện lập trình mô tả phần cứng với ngôn ngữ Verilog.

Hoàn thành việc mô tả thiết kế NetFlow V5 bằng ngôn ngữ Verilog và kiểm tra chức năng các modules: Input Arbiter, Create/Update & Hashing modules.

Hoàn thành xây dựng chương trình điều khiển và giao diện hệ thống cho Netflow V5 IP Core.

Hoàn thành việc testing các chức năng của NetFlow trên board ML605 FPGA. - Xây dựng được chương trình đọc các dữ liệu thu thập từ NetFlow lên Splunk Software để thu thập và vẽ biểu đồ.

Hoàn thiện báo cáo tổng kết, đánh giá và hoàn thiện sản phẩm.

3. Về hình thức, cấu trúc, bố cục của đồ án tốt nghiệp: (2đ)

Nội dung của đồ án tốt nghiệp: gồm 05 chương đáp ứng các mục tiêu của đề tài:

Chương I: Tổng quan về NetFlow và công nghệ FPGA.

Chương II: Cơ sở lý thuyết và kiến trúc NetFlow V5.

Chương III: Thiết kế IP Core NetFlow và thực hiện nhúng trên FPGA của Xilinx.

Chương IV: Xây dựng chương trình điều khiển và giao diện hệ thống.

Chương V: Kiểm tra, đánh giá hệ thống và thực hiện giám sát hệ thống mạng tập trung Splunk cho các data center.

4. Kết quả đạt được, giá trị khoa học, khả năng ứng dụng của đề tài: (0,9đ)

Kết quả đạt được: IP Core NetFlow V5 thiết kế trên nền tảng FPGA phân tích và trích xuất các giá trị một cách chính, đầy đủ từ nguồn dữ liệu thô internet. Sau đó các trường dữ liệu đó được đưa lên phần mềm Splunk để người quản trị mạng có thể thống kê và giám sát.

Giá trị khoa học: Cung cấp giải pháp hoàn chỉnh phân tích đầy đủ giao thức thông dụng và phức tạp như NetFlow v5 – Thực hiện thiết kế IP Core trên nền tảng FPGA theo chức năng của giao thức NetFLow để thực hiện việc phân tích và trích xuất thông tin.

Khả năng ứng dụng: Sản phẩm của đề tài giám sát và quản lý mạng mang lại các mục tiêu cho hiệu năng cao và chi phí thấp có thể ứng dụng trong các hệ thống công nghệ thông tin, các trung tâm dữ liệu của tổ chức hay công ty, …

5. Các tồn tại, thiếu sót cần bổ sung, chỉnh sửa:

Cần chỉnh sửa một số lỗi chính tả trong đồ án.

III. Tinh thần, thái độ làm việc của sinh viên: (điểm tối đa 2đ)

Thái độ làm việc: chăm chỉ, cần cù, nghiêm túc và tinh thần tự lập trong nghiên cứu. làm việc theo kế hoạch đã được quy định. Chủ động gặp gỡ và trao đổi với giảng viên hướng dẫn về đề cương và báo cáo kết quả đạt của các chương theo quy định hàng tuần.

IV. Đánh giá:

1. Điểm đánh giá: 9,6/10 (lấy đến 1 số lẻ thập phân)
2. Đề nghị: ☒ Được bảo vệ đồ án ☐ Bổ sung để bảo vệ ☐ Không được bảo vệ

Đà Nẵng, ngày 14 tháng 8 năm 2021

Người hướng dẫn

TS. Trần Hoàng Vũ

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT
KHOA ĐIỆN – ĐIỆN TỬ

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

NHẬN XÉT PHẢN BIỆN ĐỒ ÁN TỐT NGHIỆP
(*Dành cho người phản biện*)

I. Thông tin chung:

1. Họ và tên sinh viên: NGUYỄN DUY HÀ SƠN

2. Lớp: 17KTDT1 Mã SV: 1711505110126

3. Tên đề tài: NGHIÊN CỨU, THIẾT KẾ IP COR NETFLOW V5 TRÊN NỀN TẢNG FPGA ĐỂ PHÂN TÍCH VÀ GIÁT SÁT MẠNG

4. Người phản biện: Phạm Văn Phát Học hàm/ học vị: Thạc sĩ

II. Nhận xét, đánh giá đồ án tốt nghiệp:

1. Về tính cấp thiết, tính mới, mục tiêu của đề tài:

+ Dựa trên nền tảng FPGA để sử dụng nghiên cứu, thiết kế tạo nên một IP Core NetFLow là một tiến trình, công cụ hoàn toàn phù hợp với yêu cầu và xu hướng công nghệ của ngành Kỹ thuật điện tử trong nước và thế giới.

+ Việc thiết kế IP Core NetFlow có thể xử lý song song đa luồng, tái cấu trúc và tuy chỉnh, cập nhật trên nền tảng phần cứng FPGA và phần mềm Splunk nhằm tạo ra một Core giám sát mạng trong các hệ thống theo dõi, giám sát mạng dữ liệu lớn có ý nghĩa thực tiễn, đáp ứng các yêu cầu cấp thiết, dảm bảo tính mới và khả năng ứng dụng cao.

+ Thiết kế IP Core NetFlow V5 trên nền tảng FPGA thực hiện việc phân tích nguồn dữ liệu thô từ internet và trích xuất các trường như IP nguồn, IP đích .v.v. dựa trên các thông tin, các thuộc tính đợợc trích xuất để thống kê, phân tích, đánh giá trên các phần mềm hệ thống giám sát mạng Internets.

2. Về kết quả giải quyết các nội dung nhiệm vụ yêu cầu của đồ án:

+ Cơ sở lý thuyết về giao thức NetFlow, về các giao thức mạng. Phương pháp thiết kế IP Core trên nền tảng FPGA, phương pháp lập trình mô tả phần cứng với ngôn ngữ Verilog.

+ Đồ án đã thiết kế, thực thi, xây dựng công cụ kiểm tra đánh giá IP Core NetFlow V5 trên nền tảng FPGA.

+ Đồ án đã thực hiện trên cơ sở sự hỗ trợ của Lab tại Cty Công nghệ Acronics, với môi trường làm việc chuyên nghiệp, chuyên sâu trong lĩnh vực thiết kế chip.

3. ***Về hình thức, cấu trúc, bố cục của đồ án tốt nghiệp:***

+ Bố cục ĐATN hài hòa, hợp lý. Thể hiện chi tiết các cơ sở lý thuyết và thực tiễn, các phương pháp tiếp cận và công cụ thực thi, quá trình thực hiện, các kết quả kiểm thử, tổng hợp, phân tích đánh giá hệ thống.

4. ***Kết quả đạt được, giá trị khoa học, khả năng ứng dụng của đề tài:***

+ Nhóm SV đã tiếp cận, sử dụng một số phần mềm, công cụ chuyên nghiệp trong quá trình thực hiện ĐATN. Các công cụ bao gồm phần mềm ISE của Xilinx, công cụ hỗ trợ thiết kế Xilinx XPS, công cụ soạn thảo lập trình Xilinx SDK, KIT lập trình FPGA Virtex-6-ML605 Development System.

+ Đồ án đã thực hiện được thiết kế Core NetFlow V5, thực thi nhúng IP Core vào hệ thống trên board Virtex-6 ML605 và kiểm tra hoạt động của Core trên ModulSim và trên môi trường thực tế.

+ Các kết quả thể hiện qua các Sơ đồi khối chức năng của hệ thống, thiết kế máy trạng thái FSM(*Finite State Machine*) và các module chức năng, các kết quả mô phỏng, phân tích Testbench, đánh giá các khối chức năng bằng công cụ ISIM.

+ Thực thi hệ thống nhúng trên FPGA VIRTEX -6 ML605 của Xilinx với công cụ sử dụng là phần mềm Xilinx Studio Platform (XPS).

+ Đánh giá kiểm tra các thông số của IP Core kết hợp Sử dụng công cụ Splunk software. Đánh giá khả năng tối ưu, chiếm dụng tài nguyên phần cứng của IP Core NetFlow V5 trên KIT lập trình FPGA Virtex-6-ML605.

+ Sản phẩm của đề tài giám sát và quản l mạng mang lại các mục tiêu cho hiệu năng cao và chi phí thấp có thể ứng dụng trong các hệ thống công nghệ thông tin, các trung tâm dữ liệu của tổ chức hay công ty.

5. ***Các tồn tại, thiếu sót cần bổ sung, chỉnh sửa:***

+ Một số ít lỗi chính tả, định dạng văn bản còn tồn tại trong cuốn báo cáo ĐATN, ví dụ như các đề mục con tại các chương chưa đúng với quy định.

+ Các khối chức năng(Functions Block) và mô tả tín hiệu trình bày khá rõ, chi tiết. Tuy nhiên ở trong các khối này chưa thấy đề cập, hay các mô tả chi tiết, chuyên sâu mức logic, các khối hàm(thực thi con) chức năng trong các khối này.

TT	Các tiêu chí đánh giá	Điểm tối đa	Điểm đánh giá
1	**Sinh viên có phương pháp nghiên cứu phù hợp, giải quyết các nhiệm vụ đồ án được giao**	**8,0**	**7,8**
1a	- Tính cấp thiết, tính mới (nội dung chính của ĐATN có những phần mới so với các ĐATN trước đây); - Đề tài có giá trị khoa học, công nghệ; giá trị ứng dụng thực tiễn;	1,0	1,0
1b	- Kỹ năng giải quyết vấn đề; hiểu, vận dụng được kiến thức cơ bản, cơ sở, chuyên ngành trong vấn đề nghiên cứu; - Khả năng thực hiện/phân tích/tổng hợp/đánh giá; - Khả năng thiết kế, chế tạo một hệ thống, thành phần, hoặc quy trình đáp ứng yêu cầu đặt ra;	3,0	3,0
1c	- Chất lượng sản phẩm ĐATN về nội dung báo cáo, bản vẽ, chương trình, mô hình, hệ thống,…;	3,0	2,8
1d	- Có kỹ năng sử dụng phần mềm ứng dụng trong vấn đề nghiên cứu (thể hiện qua kết quả tính toán bằng phần mềm); - Có kỹ năng sử dụng tài liệu liên quan vấn đề nghiên cứu (thể hiện qua các tài liệu tham khảo).	1,0	1,0

2	**Kỹ năng trình bày báo cáo đồ án tốt nghiệp**	**2,0**	**1,8**
2a	- Bố cục hợp lý, lập luận rõ ràng, chặt chẽ, lời văn súc tích;	1,0	1,0
2b	- Hình thức trình bày.	1,0	0,8
3	**Tổng điểm theo thang 10** (lấy đến 1 số lẻ thập phân)		**9,6**

- Câu hỏi đề nghị sinh viên trả lời trong buổi bảo vệ:

  1. Việc sử dụng các công cụ, thiết kế, thực thi các khối chức năng và toàn hệ thống của IP Core NetFlow V5 là rất tốt. SV cần nói rõ hơn cách thức xây dựng các Functions, thiết kế logic bên trong các Blocks?

  2. Việc thiết kế các chip nói chung và các Core nói riêng cần được tối ưu, ổn định, tiết kiệm tài nguyên hệ thống(phần cứng), công nghệ và chi phí. Tác giả đánh giá như thế nào khi các nghiên cứu, thiết kế thử nghiệm thực hiện trên nền tảng FPGA mà cụ thể là KIT KIT Virtex-6 ML605 với chip Virtex-6 XC6CCX240t1ff156 FPGA của Xilinx, hỗ trợ RAM DDR2 cũng như việc chiếm dụng tài nguyên hệ thống IP Core NetFlow V5?

- Đề nghị: ☒ Được bảo vệ đồ án ☐ Bổ sung để bảo vệ ☐ Không được bảo vệ

Đà Nẵng, ngày 22 tháng 8 năm 2021

Người phản biện

ThS. Phạm Văn Phát

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT
KHOA ĐIỆN – ĐIỆN TỬ

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

NHẬN XÉT PHẢN BIỆN ĐỒ ÁN TỐT NGHIỆP
(*Dành cho người phản biện*)

I. Thông tin chung:

1. Họ và tên sinh viên: ĐẶNG SỸ PHI HÙNG

2. Lớp: 17KTDT1 Mã SV: 1711505210110

3. Tên đề tài: NGHIÊN CỨU, THIẾT KẾ IP COR NETFLOW V5 TRÊN NỀN TẢNG FPGA ĐỂ PHÂN TÍCH VÀ GIÁT SÁT MẠNG

4. Người phản biện: Phạm Văn Phát Học hàm/ học vị: Thạc sĩ

II. Nhận xét, đánh giá đồ án tốt nghiệp:

1. Về tính cấp thiết, tính mới, mục tiêu của đề tài:

+ Dựa trên nền tảng FPGA để sử dụng nghiên cứu, thiết kế tạo nên một IP Core NetFLow là một tiến trình, công cụ hoàn toàn phù hợp với yêu cầu và xu hướng công nghệ của ngành Kỹ thuật điện tử trong nước và thế giới.

+ Việc thiết kế IP Core NetFlow có thể xử lý song song đa luồng, tái cấu trúc và tuy chỉnh, cập nhật trên nền tảng phần cứng FPGA và phần mềm Splunk nhằm tạo ra một Core giám sát mạng trong các hệ thống theo dõi, giám sát mạng dữ liệu lớn có ý nghĩa thực tiễn, đáp ứng các yêu cầu cấp thiết, dảm bảo tính mới và khả năng ứng dụng cao.

+ Thiết kế IP Core NetFlow V5 trên nền tảng FPGA thực hiện việc phân tích nguồn dữ liệu thô từ internet và trích xuất các trường như IP nguồn, IP đích .v.v. dựa trên các thông tin, các thuộc tính đợợc trích xuất để thống kê, phân tích, đánh giá trên các phần mềm hệ thống giám sát mạng Internets.

2. Về kết quả giải quyết các nội dung nhiệm vụ yêu cầu của đồ án:

+ Cơ sở lý thuyết về giao thức NetFlow, về các giao thức mạng. Phương pháp thiết kế IP Core trên nền tảng FPGA, phương pháp lập trình mô tả phần cứng với ngôn ngữ Verilog.

+ Đồ án đã thiết kế, thực thi, xây dựng công cụ kiểm tra đánh giá IP Core NetFlow V5 trên nền tảng FPGA.

+ Đồ án đã thực hiện trên cơ sở sự hỗ trợ của Lab tại Cty Công nghệ Acronics, với môi trường làm việc chuyên nghiệp, chuyên sâu trong lĩnh vực thiết kế chip.

3. ***Về hình thức, cấu trúc, bố cục của đồ án tốt nghiệp:***

+ Bố cục ĐATN hài hòa, hợp lý. Thể hiện chi tiết các cơ sở lý thuyết và thực tiễn, các phương pháp tiếp cận và công cụ thực thi, quá trình thực hiện, các kết quả kiểm thử, tổng hợp, phân tích đánh giá hệ thống.

4. ***Kết quả đạt được, giá trị khoa học, khả năng ứng dụng của đề tài:***

+ Nhóm SV đã tiếp cận, sử dụng một số phần mềm, công cụ chuyên nghiệp trong quá trình thực hiện ĐATN. Các công cụ bao gồm phần mềm ISE của Xilinx, công cụ hỗ trợ thiết kế Xilinx XPS, công cụ soạn thảo lập trình Xilinx SDK, KIT lập trình FPGA Virtex-6-ML605 Development System.

+ Đồ án đã thực hiện được thiết kế Core NetFlow V5, thực thi nhúng IP Core vào hệ thống trên board Virtex-6 ML605 và kiểm tra hoạt động của Core trên ModulSim và trên môi trường thực tế.

+ Các kết quả thể hiện qua các Sơ đồi khối chức năng của hệ thống, thiết kế máy trạng thái FSM(*Finite State Machine*) và các module chức năng, các kết quả mô phỏng, phân tích Testbench, đánh giá các khối chức năng bằng công cụ ISIM.

+ Thực thi hệ thống nhúng trên FPGA VIRTEX -6 ML605 của Xilinx với công cụ sử dụng là phần mềm Xilinx Studio Platform (XPS).

+ Đánh giá kiểm tra các thông số của IP Core kết hợp Sử dụng công cụ Splunk software. Đánh giá khả năng tối ưu, chiếm dụng tài nguyên phần cứng của IP Core NetFlow V5 trên KIT lập trình FPGA Virtex-6-ML605.

+ Sản phẩm của đề tài giám sát và quản l mạng mang lại các mục tiêu cho hiệu năng cao và chi phí thấp có thể ứng dụng trong các hệ thống công nghệ thông tin, các trung tâm dữ liệu của tổ chức hay công ty.

5. ***Các tồn tại, thiếu sót cần bổ sung, chỉnh sửa:***

+ Một số ít lỗi chính tả, định dạng văn bản còn tồn tại trong cuốn báo cáo ĐATN, ví dụ như các đề mục con tại các chương chưa đúng với quy định.

+ Các khối chức năng(Functions Block) và mô tả tín hiệu trình bày khá rõ, chi tiết. Tuy nhiên ở trong các khối này chưa thấy đề cập, hay các mô tả chi tiết, chuyên sâu mức logic, các khối hàm(thực thi con) chức năng trong các khối này.

TT	Các tiêu chí đánh giá	Điểm tối đa	Điểm đánh giá
1	- **Sinh viên có phương pháp nghiên cứu phù hợp, giải quyết các nhiệm vụ đồ án được giao**	**8,0**	**7,8**
1a	- Tính cấp thiết, tính mới (nội dung chính của ĐATN có những phần mới so với các ĐATN trước đây); - Đề tài có giá trị khoa học, công nghệ; giá trị ứng dụng thực tiễn;	1,0	1,0
1b	- Kỹ năng giải quyết vấn đề; hiểu, vận dụng được kiến thức cơ bản, cơ sở, chuyên ngành trong vấn đề nghiên cứu; - Khả năng thực hiện/phân tích/tổng hợp/đánh giá; - Khả năng thiết kế, chế tạo một hệ thống, thành phần, hoặc quy trình đáp ứng yêu cầu đặt ra;	3,0	3,0
1c	- Chất lượng sản phẩm ĐATN về nội dung báo cáo, bản vẽ, chương trình, mô hình, hệ thống,…;	3,0	2,8
1d	- Có kỹ năng sử dụng phần mềm ứng dụng trong vấn đề nghiên cứu (thể hiện qua kết quả tính toán bằng phần mềm); - Có kỹ năng sử dụng tài liệu liên quan vấn đề nghiên cứu (thể hiện qua các tài liệu tham khảo).	1,0	1,0
2	**Kỹ năng trình bày báo cáo đồ án tốt nghiệp**	**2,0**	**1,8**

2a	- Bố cục hợp lý, lập luận rõ ràng, chặt chẽ, lời văn súc tích;	1,0	1,0
2b	- Hình thức trình bày.	1,0	0,8
3	**Tổng điểm theo thang 10** (lấy đến 1 số lẻ thập phân)		**9,6**

- Câu hỏi đề nghị sinh viên trả lời trong buổi bảo vệ:

  1. Các chức năng chính của IP Core NetFlow V5 và phân tích, đánh giá kết quả kiểm thử các Core trên KIT Virtex-6 ML605?

  2. Việc thiết kế, tích hợp các Core NetFlow trong các thiết bị mạng, đặc biệt là các hệ thống định tuyến, giám sát lưu lượng, phân tích các trường dữ liệu, phát hiện các đột nhập là rất quan trọng và cần thiết. Tác giả đánh giá như thế nào về hiệu quả, chi phí cũng như khả năng đáp ứng băng thông rộng, tốc độ cao của Core NetFlow V5? hỗ trợ RAM DDR2 cũng như việc chiếm dụng tài nguyên hệ thống IP Core NetFlow V5?

- Đề nghị: ☒ Được bảo vệ đồ án ☐ Bổ sung để bảo vệ ☐ Không được bảo vệ

Đà Nẵng, ngày 22 tháng 8 năm 2021

Người phản biện

ThS. Phạm Văn Phát

TÓM TẮT

Tên đề tài: Nghiên cứu, thiết kế IP Core NetFlow V5 trên nền tảng công nghệ FPGA để phân tích và giám sát mạng.

Sinh viên thực hiện:

1. Nguyễn Duy Hà Sơn. Mã SV: 1711505110126
2. Đặng Sỹ Phi Hùng. Mã SV: 1711505210110
   Lớp: 17KTDT1.

Đề tài này sử dụng kết hợp giữa phần cứng FPGA và phần mềm Splunk để xây dựng một hệ thống có thể phân tích và giám sát mạng dữ liệu nhằm mục tiêu giảm chi phí duy trì vừa làm chủ được công nghệ và đảm bảo an toàn hơn so với các sản phẩm hiện có trên thị trường.

Đề tài tập trung thực hiện thiết kế IP CORE NetFlow V5 trên nền tảng công nghệ FPGA nhúng trên board Virtex6-ML605 để thực hiện nhiệm vụ phân tích các dữ liệu mạng đầu vào để trích xuất các trường tại header của gói tin là các trường thông tin cần thiết bao gồm: Địa chỉ IP nguồn; địa chỉ IP đích; các giao thức mạng sử dụng của gói tin như UDP, TCP (protocol); kèm theo các thông số tính toán gói tin như số lượng bytes trong một gói hay tổng số bytes trong một Port nhận; Đó là những header cho chúng ta thấy được các thông tin dữ liệu được gửi từ nguồn đến đích và dung lượng là bao nhiêu, theo giao thức nào để người quản trị mạng dễ dàng quản lý, bằng cách gửi các trường thông tin đã phân tích được lên phần mềm Splunk để giám sát và theo dõi.

Splunk là phần mềm để thu thập Log, tìm kiếm, theo dõi và phân tích dữ liệu lớn (big data) do máy tạo ra, thông qua một giao diện web nhằm giải quyết nhiều bài toán khác nhau của các tổ chức trong việc giám sát, vận hành hệ thống và điều tra sự cố,..v.v.. Đề tài sử dụng phần mềm Splunk làm hệ thống thu nhận, trích ra các dữ liệu thời gian thực có liên quan đến nhau từ các trường thông tin được trích xuất từ phần cứng, từ đó có thể tạo ra các đồ thị, các báo cáo, các biểu đồ. Mục đích của phần mềm Splunk là giúp cho việc xác định mô hình dữ liệu và thu thập dữ liệu trên toàn hệ thống dễ dàng hơn. Phần mềm Splunk cung cấp số liệu, chẩn đoán các vấn đề xảy ra, phục vụ cho hoạt động giám sát các luồng dữ liệu trong hệ thống mạn

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT
KHOA ĐIỆN – ĐIỆN TỬ

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

Giảng viên hướng dẫn: TS. TRẦN HOÀNG VŨ

Sinh viên thực hiện: NGUYỄN DUY HÀ SƠN……Mã SV: 1711505110126

ĐẶNG SỸ PHI HÙNG…… Mã SV: 1711505210110

1. Tên đề tài:

NGHIÊN CỨU, THIẾT KẾ IP CORE NETFLOW V5 TRÊN NỀN TẢNG FPGA ĐỂ PHÂN TÍCH VÀ GIÁM SÁT MẠNG.

2. Các số liệu, tài liệu ban đầu:

- Tài liệu, thuật toán NetFlow V5.

- Tài liệu về ngôn ngữ lập trình phần cứng Verilog; Quy trình thiết kế IP Core trên nền tảng FPGA; Các phần mềm Xilinx trong thiết kế nhúng.

3. Nội dung chính của đồ án:

Chương I: Tổng quan về NetFlow và công nghệ FPGA.

Chương II: Cơ sở lý thuyết và kiến trúc NetFlow V5.

Chương III: Thiết kế IP Core NetFlow và thực hiện nhúng trên FPGA của Xilinx.

Chương IV: Xây dựng chương trình điều khiển và giao diện người dùng cho hệ thống

Chương V: Kiểm tra, đánh giá hệ thống và thực hiện giám sát hệ thống mạng tập trung Splunk cho các Data center.

4. Các sản phẩm dự kiến

IP Core NetFlow V5 thiết kế trên nền tảng FPGA phân tích và trích xuất các giá trị một cách chính xác, đầy đủ từ nguồn dữ liệu thô internet. Sau đó các trường dữ liệu đó được đưa lên phần mềm Splunk để người quản trị mạng có thể phân tích thống kê và giám sát.

5. Ngày giao đồ án: 10/03/2021

6. Ngày nộp đồ án: 10/08/2021

Đà Nẵng, ngày 10 tháng 03 năm 2021

P. Trưởng Bộ môn	**Người hướng dẫn**
ThS. Phạm Văn Phát	**TS. Trần Hoàng Vũ**

LỜI NÓI ĐẦU

Đầu tiên chúng em xin gửi lời cảm ơn sâu sắc đến **TS. Trần Hoàng Vũ** đã trực tiếp hướng dẫn, định hướng, dành nhiều thời gian và tâm huyết giúp đỡ chúng em về mọi mặt để hoàn thiện đồ án tốt nghiệp.

Chúng em chân thân thành cảm ơn quý thầy cô trong **Khoa Điện – Điện tử trường Đại Học Sư Phạm Kỹ thuật – Đại Học Đà Nẵng** đã tạo điều kiện, tạo môi trường thuận lợi cho chúng em học tập và nghiên cứu, đã tận tình truyền đạt các kiến thức nền tảng, kiến thức nâng cao trong những năm vừa qua, với cơ sở lý thuyết và hệ thống thực hành tại các phòng thí nghiệm, phòng Labs, tất cả là hành trang quý báu, là nền tảng để chúng em đạt được kết quả tốt trong đồ án của mình.

Chúng em cũng mong muốn gửi lời cảm ơn đến những những người bạn đã gắn bó trong suốt thời gian ngồi trên ghế nhà trường những năm đại học qua, cùng nhau phát triển để có thể hoàn thiện; Và cuối cùng chúng em cũng bày tỏ lòng biết ơn sâu sắc đến gia đình: Ba mẹ, anh chị đã giúp đỡ và hy sinh rất nhiều để lo chúng em ăn học, học tập trong thời gian qua. Đây cũng chính là động lực to lớn để chúng em vượt qua mọi khó khăn để hoàn thiện đồ án tốt nghiệp này.

CAM ĐOAN

Nhóm chúng em gồm có 2 thành viên:

Nguyễn Duy Hà Sơn , sinh viên lớp 17KTDT1

Đặng Sỹ Phi Hùng, sinh viên lớp 17KTDT1

Chúng em xin cam đoan các kết quả được trình bày trong đồ án này là thành quả nghiên cứu của chúng em trong suốt thời gian qua và chưa từng xuất hiện trong công bố hay sao chép của tác giả khác dưới sự định hướng và hướng dẫn của TS. Trần Hoàng Vũ. Các thông tin trích dẫn trong đồ án được chỉ rõ, nguồn gốc rõ ràng và được phép công bố. Các kết quả đạt được chính xác và trung thực. Nếu có bất kỳ vi phạm nào, nhóm xin chịu hoàn toàn trách nhiệm và chịu mọi sự kỷ luật của khoa và nhà trường.

Sinh viên thực hiện

{Chữ ký, họ và tên sinh viên}

Nguyễn Duy Hà Sơn Đặng Sỹ Phi Hùng

MỤC LỤC

DANH SÁCH CÁC BẢNG

DANH SÁCH CÁC HÌNH ẢNH

DANH SÁCH CHỮ VIẾT TẮT

EDK	:	Embedded Deveplopment Kit.
SDK	:	Software Development Kit
XPS	:	Xilinx Platform Studio
FPGA	:	Field-Progammable Gate Array.
HDL	:	Hardware Description Language.
MPGA	:	Mask Programmable Gate Array.
SoC	:	System on Chip.
CRC	:	Cyclic Redundancy Check.
Tri-EMAC	:	Tri-Mode Ethernet Media Access Controller.
GUI	:	Graphical User Interface.
MDIO	:	Management Data Input/Output.
MII	:	Media Independent Interface
GMII	:	Gigabit Media Independent Interface.
RGMII	:	Reduced Gigabit Media Independent Interface.
TCP	:	Transmission Control Protocol.
UDP	:	User Datagram Protocol.
IPv4	:	Internet Protocol version 4.
FSM	:	Finite Status Machine.
FIFO	:	First In First Out.
ALU	:	Arithmetic logic unit.
FPU	:	Floating Point Unit.
LMB	:	Local Memory Bears.
CPLD	:	Complex Programmable Logic Device.
SPLD	:	System Programmable Logic Device.
PLD	:	Program Logic Device.
VHDL	:	VHSIC Hardware Description Language.
SIEM	:	Security information and event management
HDL	:	Hardware Description Language.
RAM	:	Random Access Memory.
PC	:	Personal Computer
MAC	:	Medium Access Control
FMC	:	FPGA Mezzanine card
PHY	:	Physical Layer
UART	:	Universal Asynchronous Receiver Transmitter
IIC	:	Inter-Integrated Circuit
RTC	:	Real-Time control
SDA	:	Serial Data
SCL	:	Serial Clock
HTTP	:	Hyper Text Transfer Protocol

MỞ ĐẦU

1. Tính cấp thiết của đề tài

Ngày nay, những ứng dụng của hệ thống mạng như điện toán đám mây, mạng xã hội hay dịch vụ đa phương tiện đang trở nên phổ biến. Sự gia tăng về lưu lượng mạng Internet cũng như trong các trung tâm dữ liệu, năng lượng cần thiết để vận hành cơ sở hạ tầng mạng lõi và hệ thống mạng trung tâm cũng tăng đáng kể. Qua đó việc giám sát, vận hành, điều phối hệ thống mạng Thông tin – Dữ liệu lớn là vấn đề luôn luôn mới và cấp bách đối các Chính Phủ điện tử, các doanh nghiệp, tổ chức có hệ thống CNTT lớn. Với sự tiến bộ khoa học kỹ thuật như Mạng băng thông rộng 5G, đường truyền Quang tốc độ lớn, kỹ thuật vi mạch bán dẫn tích hợp mật độ các bóng bán dẫn cực lớn, công nghệ IoT, dẫn đến lượng dữ liệu khổng lồ và tốc độ dữ liệu cao, điều đó đặt ra vấn đề cấp bách là nhà vận hành hệ thống mạng phải có công cụ dùng để phân tích nhanh và chính xác các diễn biến trên hệ thống, ví dụ phân tích các lưu lượng bất thường, phân tích các kết nối bất thường, hay phân tích các hành vi bất thường để có các giải pháp ngăn chặn kịp thời và chính xác. Hiện nay có khá nhiều phần mềm thương mại có chức năng phân tích, giám sát các vấn đề bất thường nêu trên. **Tuy nhiên một số vấn đề đặt ra liên quan đến kỹ thuật, bảo mật và kinh phí vận hành.**

Cụ thể như sau:

- Với các dữ liệu lớn, tốc độ cao và xử lý nhiều kết nối, thì các phần mềm thương mại không phân tích và xử lý kịp các diễn biến luồng thông tin trên mạng, dẫn đến việc phân tích và xử lý không chính xác và kịp thời.
- Hiện các phần mềm mua ở nước ngoài, việc đảm bảo các phần mềm đó không bị cài mã độc hay Back-Door rất khó xác định. Điều đó sẽ có phần gây mất an ninh an toàn thông tin của hệ thống.
- Các phần mềm có giá thành rất cao, một số phần mềm có giá thành cao hơn rất nhiều so với giá thành một hệ thống cơ sở hạ tầng mạng thông tin.
- Dữ liệu lớn, gồm nhiều giao thức và phức tạp thì đòi hỏi phải mua và sử dụng các phần mềm bản quyền, đi kèm với nó là chi phí lớn để duy trì hằng năm. Dữ liệu lớn, bằng thông cao, nhiều giao thức phức tạp và theo dõi cho nhiều Node mạng, các Sever tầm trung hầu như không xử lý kịp nên phải cần các thiết bị mạng đặc chủng và Sever mạnh. Và đây là bài toán kinh tế hóc búa, vừa phải tốn chi phí quyền để sử dụng và duy trì (Splunk; Syslog-Ng; ArcSight Logger) vừa phải đầu tư hệ thốn Sever tầm cao để đáp ứng tốc độ truyền tải của băng thông.

Với những bất cập được nêu trên, giải pháp của đề tài ***"Nghiên cứu, thiết kế IP Core NetFlow V5 trên nền tảng FPGA để phân tích và giám sát mạng"*** sẽ được khắc phục và giải quyết tốt. Cụ thể các giai đoạn phân tích và trích xuất các luồng thông tin mạng phức tạp sẽ thực hiện trên công nghệ vi mạch FPGA và IP core NetFlow V5 sẽ thực hiện điều này, việc giám sát sẽ sử dụng phần mềm Splunk. Hệ thống dùng để phân tích và giám sát sẽ không nối trực tiếp vào Internet hay các thiết bị mạng không rõ nguồn gốc, mà nối qua thiết bị FPGA. Điều đó sẽ tăng mức độ an toàn anh ninh cho hệ thống. Giám sát lưu lượng mạng sử dụng phần mềm Splunk với dung lượng các trường nhỏ, sử dụng hợp lý thì giá thành thiết bị sẽ cạnh tranh tốt so với các phần mềm thương mại ngoại nhập. Hiện nay các hãng công nghệ lớn như Intel, IBM, Cisco điều kết hợp giải pháp Hardware – Software Co_Monitorring để giám sát, phân tích và vận hành các hệ thống mạng.

Do vậy giải pháp của thiết bị sử dụng phần mềm Splunk kết hợp FPGA là một hướng đi đúng các mục tiêu đặt ra trong giai đoạn hiện nay và cấp bách để giải quyết những vấn đề bất thường trong hệ thống mạng hiện nay ở nước ta, chúng ta có thể giám sát để phát hiện những vấn đề không ổn như nghẽn mạng, dung lượng tăng đột ngột… và nhanh chóng đưa ra giải pháp và ngăn chặn kịp thời.

2. Mục tiêu, phạm vi, đối tượng và phương pháp nghiên cứu:

2.1 Mục tiêu nghiên cứu:

- Thiết kế một IP Core NetFlow V5 trên nền tảng công nghệ FPGA, IP core này sẽ thực hiện phân tích nguồn dữ liệu thô Internet, phần cứng sẽ tách ra các thông tin cần thiết như IP-Nguồn, IP-Đích, Port-Nguồn, Port-Đích, Protocol để đưa về Splunk nhận dạng và thống kê.
- Sử dụng phần mềm Splunk để thu nhận, trích ra những dữ liệu theo thời gian thực có liên quan đến nhau từ đó có thể tạo ra các đồ thị, các báo cáo và đồ thị. Mục đích là giúp cho việc xác định mô hình dữ liệu và thu thập dữ liệu mạng trên toàn hệ thống dễ dàng hơn.

2.2 Phạm vi nghiên cứu:

- Phân tích thuật toán NetFlow V5 phân tích lưu lượng mạng tốc độ cao, phục vụ cho việc trích xuất các thông tin mạng.
- Sử dụng ngôn ngữ mô tả phần cứng Verilog HDL để mô tả thuật toán NetFlow V5.
- Nghiên cứu phần mềm thu thập thông tin, giám sát và phân tích lưu lượng mạng Splunk.
- Thực hiện thuật toán NetfLow V5.

2.3 Đối tượng và phương pháp nghiên cứu:

- Tập trung vào thuật toán NetFlow V5, thiết kế IP Core trên nền tảng kit FPGA. Cơ sở dữ liệu, gói tin Internet và các giao thức UDP, TCP.
- Phương pháp nghiên cứu:

  Nghiên cứu lý thuyết:

  - Tìm hiểu NetFlow V5, chuẩn AXI-lite, chuẩn AXI-Stream, vi điều khiển Microblaze, RTC.
  - Phương pháp tích hợp IP Core vào hệ thống nhúng FGPA và chương trình testbench kiểm thử

  Nghiên cứu thực nghiệm:

  - Sử dụng ngôn ngữ mô tả phần cứng Verilog để mô tả thành các IP Core.
  - Thực hiện thuật toán NetFlow V5 để thiết kế IPCore NetFlow V5 trên nền FPGA, mô phỏng và đánh giá kết quả.
  - Thực hiện nhúng IPCore NetFlow V5 trên nền tảng FPGA.

3. Cấu trúc nội dung của đồ án:

Đề tài của đồ án bao gồm 5 chương. Chương giới thiệu tổng quan về đề tài, các vấn đề về dữ liệu, data center và công nghệ FPGA sẽ được trình bày ở chương 1. Toàn bộ cơ sở lý thuyết và kiến trúc của đồ án nằm ở chương 2. Và các phần thực hiện, thiết kế kiểm tra vận hành sẽ được giới thiệu ở chương 3, chương 4, chương 5.

CHƯƠNG I: TỔNG QUAN VỀ NETFLOW VÀ CÔNG NGHỆ FPGA

Trong chương này sẽ giới thiệu về NetFlow, các ứng dụng của Netflow trong thực tế. Giới thiệu về khái niệm, cấu trúc, các ứng dụng của FPGA và ưu điểm khi sử dụng FPGA.

CHƯƠNG II: CƠ SỞ LÝ THUYẾT VÀ KIẾN TRÚC NETFLOW V5

Chương này sẽ bao gồm các nội dung về lý thuyết liên quan đến quá trình thực hiện nghiên cứu. Giới thiệu và mô tả chức năng kiến trúc của Netflow V5.

CHƯƠNG III: THIẾT KẾ IPCORE NETFLOW V5 VÀ THỰC HIỆN NHÚNG TRÊN FPGA CỦA XILINX

Chương này giới thiệu quy trình thiết kế IPCore Netflow V5 sử dụng ngôn ngữ lập trình veriilog, sơ đồ khối tổng quát của hệ thống, chi tiết chức năng từng module. Kết nối giữa các module với nhau. Thực thi nhúng IPCore Netflow V5 vào hệ thống và kiểm tra hoạt động của core trên môi trường thực tế.

CHƯƠNG IV: XÂY DỰNG CHƯƠNG TRÌNH ĐIỀU KHIỂN VÀ GIAO DIỆN HỆ THỐNG

Chương này chủ yếu ta sẽ xây dựng chương trình điều khiển các ngoại vi được tích hợp trong hệ thống trên công cụ SDK, lập trình khởi tạo, truyền nhận dữ liệu bằng UART, IIC của vi điều khiển MicroBlaze.

CHƯƠNG V: KIỂM TRA, ĐÁNH GIÁ HỆ THỐNG VÀ THỰC HIỆN GIÁM SÁT HỆ THỐNG MẠNG TẬP TRUNG SPLUNK CHO CÁC DATA CENTER

Trong chương này ta sẽ xây dựng các Testcase để thực hiện test hệ thống core Netflow trên môi trường thực tế, trên board. Thực hiện kiểm tra và đánh giá kết quả để có hướng khắc phục hay cải thiện. Đồng thời xây dựng giao diện hiển thị bằng Java và liên kết với Splunk locohost để giám sát hệt thống mạng bằng cách đẩy các trường thông tin phân tích được ở NetFlow IP core lên Splunk rồi tạo các biểu đồ giám sát realtime.

Chương 1: TỔNG QUAN VỀ NETFLOW VÀ CÔNG NGHỆ FPGA

1.1 Giới thiệu chương

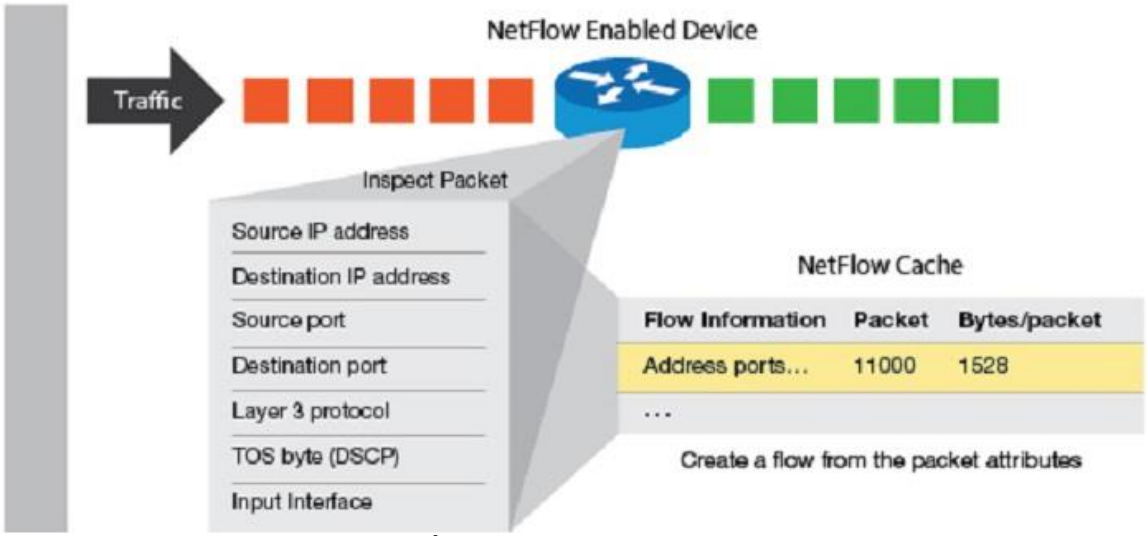
Trong chương đầu tiên này sẽ trình bày khái quát về thuật toán Netflow V5, các phương pháp giám sát mạng, xu hướng sử dụng của các thiết bị giám sát mạng hiện nay. Chương này giới thiệu ngắn gọn về ngôn ngữ mô tả phần cứng Verilog, công nghệ FPGA, lịch sử ra đời, các khái niệm, cấu trúc của FPGA và các ứng dụng của FPGA.

1.2 Tổng quan về NetFlow

1.2.1. Khái niệm và ứng dụng của Netflow V5

Netflow là một giao thực do hãng Cisco phát triển vào những năm 1996, được phát triển thành một công nghệ giám sát lưu lượng mạng [19].

Hiện nay, Netflow được xây dựng thành tiêu chuẩn và sử dụng hầu hết trong các thiết bị mạng Router của Cisco, Juniper, Extreme, Habour,… Netflow đã được phát triển qua nhiều phiên bản: version 1 đến version 10; trong đó thông dụng nhất hiện nay là version 5, version 7 và version 9 *[19]*.



Hình 1.1. Luồng dữ liệu giao thức Netflow V5 [19]

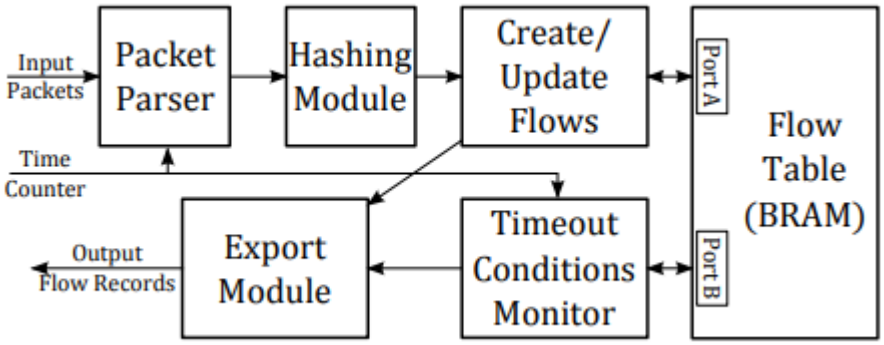
Netflow cho phép thực hiện giám sát, phân tích, tính toán lưu lượng gói. Một trong các ưu điểm của Netflow so với các giao thức khác là nó cho phép định danh và phân loại những loại tấn công như DoS, DdoS, Worm,… theo thời gian thực dựa vào những hành vi thay đổi bất thường trong mạng, đặc biệt trong mạng có lưu lượng lớn.

Giao thức Netflow V5 có định dạng gói tin cố định giúp cho việc giám sát và báo các lưu lượng mạng dễ dàng hơn vì mỗi gói được nhận dạng nhanh chóng. Thuật toán Netflow V5 thực hiện phân tích dữ liệu thô từ Internet để lấy những trường thông tin cần thiết, gồm:

- Địa chỉ IP nguồn.
- Địa chỉ IP đích.
- Cổng nguồn cho UDP hoặc TCP, 0 cho các giao thức khác.

- Cổng đích cho UDP hoặc TCP, nhập và mã ICMP hoặc 0 cho các giao thức khác.
- Giao thức IP.

Những trường thông tin này sẽ cần thiết cho nhà quản trị mạng có thể dễ dàng giám sát và phát hiện các sự cố khi có những hành vi bất thường liên quan đến hệ thống mạng. Quá trình luồng hoạt động của Netflow V5 được thực thể hiện chi tiết theo sơ đồ chức năng sau:



Hình 1.2. Kiến trúc Netflow V5 [10]

1.2.2. Phương pháp và xu hướng giám sát mạng

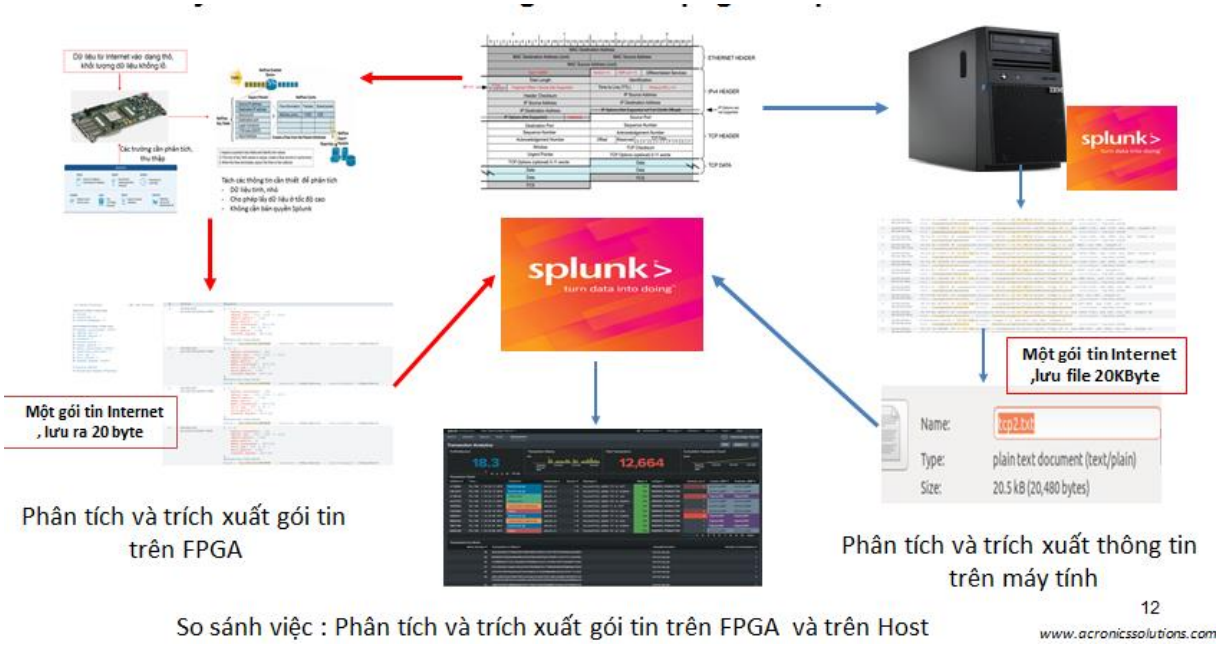
Hệ thống giám sát an ninh mạng viết tắt là SIEM (Security information and event management – SIEM) là các hệ thống được thiết kế nhằm thu thập và phân tích nhật ký, các sự kiện an ninh từ các thiết bị đầu cuối và được lưu trữ tập trung [21].

Hình 1.3. Mô hình SIEM-Security information and event management [21].

Việc giám sát mạng: Cho phép giám sát tình trạng hoạt động của mạng theo thời gian thực. Giám sát mạng dựa vào luồng dữ liệu theo V5/V7/V9 của giao thức NetFlow tập những gói có cùng 7 thông tin : IP nguồn, IP đích, Port nguồn, Port đích, ToS, loại

giao thức lớp 3, cổng vào,) nhằm thực hiện thu thập thông tin theo lưu lượng gói, theo luồng liên quan đến một thiết bị router, switch hoặc sự kết hợp của nhiều lưu lượng từ nhiều thiết bị giúp chủ động nhận diện được vấn đề, hiệu quả trong quá trình xử lý sự cố và đưa ra giải pháp giải quyết vấn đề một cách nhanh chóng.

Hiện nay, giám sát lưu lượng mạng sử dụng các công cụ và kỹ thuật khác nhau để phân tích lưu lượng mạng trên máy tính. Khi các mạng trở nên bận rộn hơn, điều phổ biến là tốc độ chung của các mạng này chậm lại. Rất nhiêu xu hướng khác nhau đang trở nên phổ biến trong cơ sở hạ tầng CNTT như sự gia tăng sử dụng máy chủ đám mây, video, VOIP, v.v… Tất cả những xu hướng này gây áp lực rất lớn cho tài nguyên cơ sở hạ tầng CNTT. Quá trình này không chỉ tốn kém mà hiệu quả trong một khoảng thời gian ngắn. Với giải pháp Splunk Software: một gói tin Internet đi vào thì Splunk Software sẽ phân tích gói tin ấy từ file data, mỗi file này có dung lượng 20Kbyte (20 x 1024 Byte = 4028 Byte) và đưa lên tầng trên để xử lý.



Hình 1.4. So sánh giải pháp phần mềm và phần cứng Splunk

Cách tốt nhất hiện nay được các hãng công nghệ lớn như IBM, Cisco, Intel là kết hợp giải pháp Hardware- Software Co_Monitorring để giám sát, phân tích và vận hành các hệ thống trung tâm dữ liệu mạng. Trong đó, các giải pháp phần cứng về mạng được triển khai trên công nghệ nền tảng NetFPGA luôn được lựa chọn. Một gói tin Internet đi vào thì FPGA sẽ phân tích và trích xuất gói tin ra các thông tin cần thiết với dung lượng 20Byte – 50Byte và gửi về Software Splunk để xử lý. Cùng với một lượng thông tin cần quản lý trên Internet, thì Splunk Software sẽ chiếm dung lượng xử lý gấp 80 – 200 lần so với sự kết hợp FPGA – Splunk. So với các giải pháp phần mềm về giám sát an ninh mạng nói chung và Splunk nói riêng, thì chi phí bản quyền dựa vào dụng lượng

dữ liệu cần phân tích (Indexing) đầu vào. Do vậy với giải pháp Hardware-Software thì chi phí bản quyền sẽ giảm đi ít nhất 50 – 80 lần so với cùng một dung lượng đưa vào.

Dựa vào các đánh giá hiệu suất giữa hai giải pháp Software và Hardware FPGA kết hợp Software Splunk với việc giám sát, bảo mật thông tin mạng của những Data Center lớn như Ngành Vận Tải, Ngân hàng, Trung Tâm dữ liệu Thanh Phố hay Quốc Gia, mỗi ngày xử lý hàng trăm Terabyte sẽ giảm chi phí đầu tư, đồng thời làm chủ công nghệ và an toàn hơn so với các sản phẩm ngoại nhập.

1.3 Ngôn ngữ mô tả phần cứng Verilog và tổng quan về FPGA

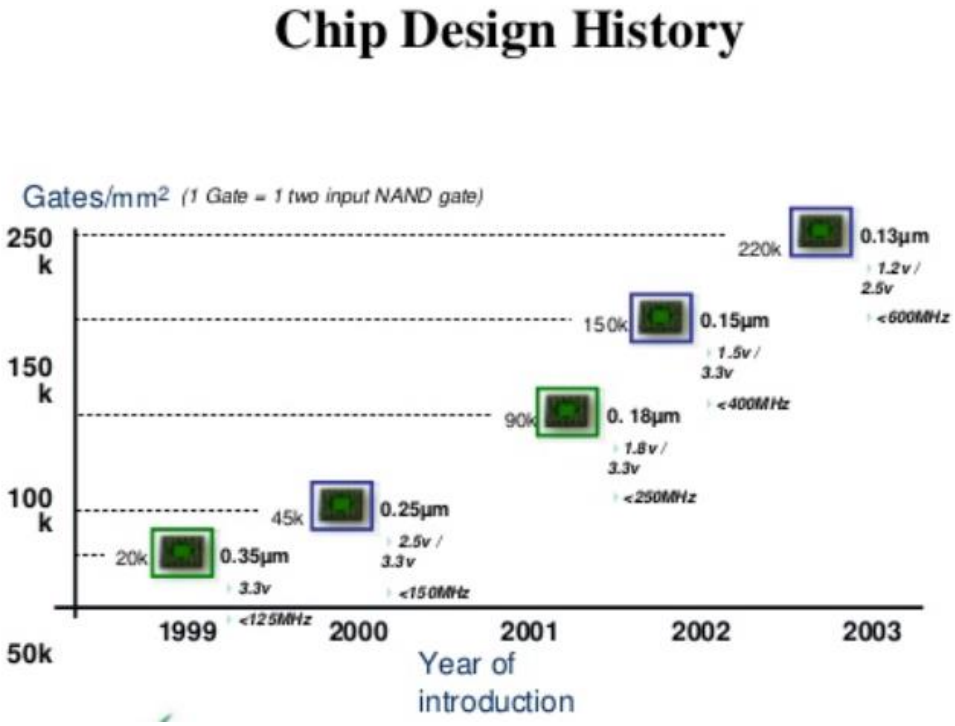
Ngôn ngữ mô tả phần cứng (HDL) là ngôn ngữ lập trình phần mềm dùng để mô hình hoạt động mong muốn của phần cứng. Có nhiều ngôn ngữ mô tả phần cứng khác nhau, tuy nhiên có hai ngôn ngữ mô tả phần cứng theo chuẩn công nghiệp hiện tại được sử dụng phổ biến ưa chuộng hơn cả là VHDL và Verilog. Cả hai ngôn ngữ này được IEEE công nhận.

1.4 Ngôn ngữ Verilog

Công ty Gateway phát hành ngôn ngữ mô tả phần cứng Verilog có tên là Verilog HDL, nâng cấp thành Verilog XL, ngoài mô tả phần cứng có thể tạo các công cụ kiểm tra, mô phỏng. Ngôn ngữ này đã thông dụng trên thế giới. 85% các xưởng chế tạo ASIC trên thế giới sử dụng Verilog trong thiết kế. Năm 1995 – IEEE đã công nhận Verilog thành chuẩn IEEE1364 Ngôn ngữ Veriolg có cấu trúc chương trình, lệnh, kiểu dữ liệu, khai báo biến đơn giản, tương tự như ngôn ngữ C, nên việc tiếp cận, sử dụng dễ dàng . Đối với những người đã sử dụng ngôn ngữ C để lập trình thì dễ dàng nắm bắt ngôn ngữ Verilog [20].

1.5 Lịch sử ra đời của FPGA

FPGA được thiết kế đầu tiên bởi Ross Freeman, người sáng lập công ty Xilinx vào năm 1984, kiến trúc mới của FPGA cho phép tích hợp số lượng tương đối lớn các phần tử bán dẫn vào 1 vi mạch so với kiến trúc trước đó là CPLD. FPGA có khả năng chứa tới từ 100.000 đến hàng vài tỷ cổng logic, trong khi CPLD chỉ chứa từ 10.000 đến 100.000 cổng logic; con số này đối với PAL, PLA còn thấp hơn nữa chỉ đạt vài nghìn đến 10.000 [18].



Hình 1.5. Lịch sử phát triển thế kế chips [18]

CPLD được cấu trúc từ số lượng nhất định các khối SPLD (Simple Programable Logic Device) thuật ngữ chung chỉ PAL, PLA. SPLD thường là một mảng logic AND/OR lập trình được có kích thước xác định và chứa một số lượng hạn chế các phần tử nhớ đồng bộ (clocked register). Cấu trúc này hạn chế khả năng thực hiện những hàm phức tạp và thông thường hiệu suất làm việc của vi mạch phụ thuộc vào cấu trúc cụ thể của vi mạch hơn là vào yêu cầu bài toán.

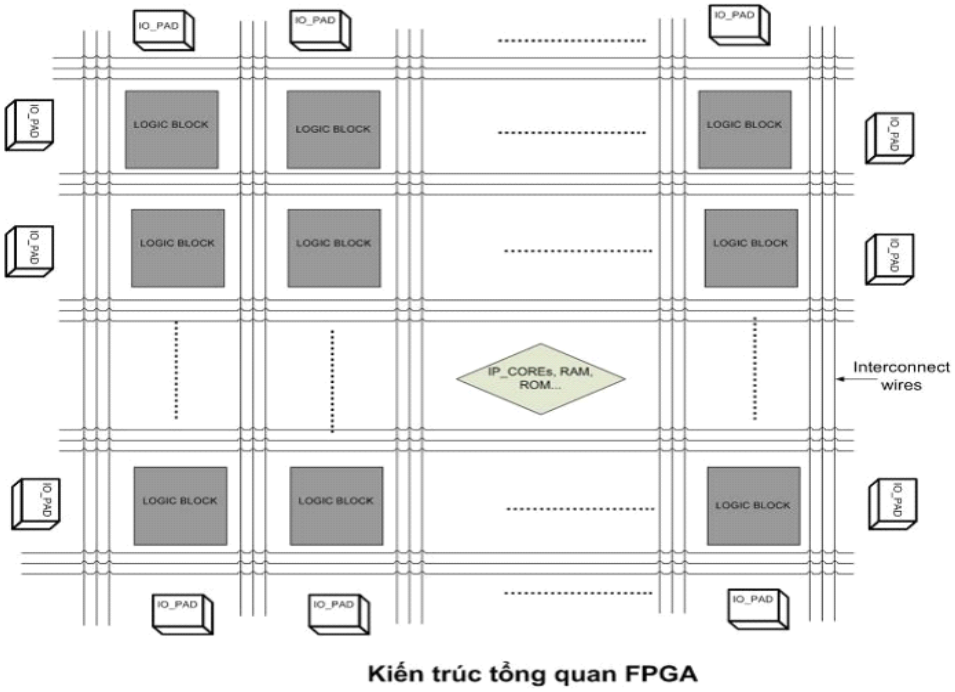
Kiến trúc của FPGA là kiến trúc mảng các khối logic, mỗi khối này nhỏ hơn nhiều nếu đem so sánh với một khối SPLD, ưu điểm này giúp FPGA có thể chứa nhiều hơn các phần tử logic và phát huy tối đa khả năng lập trình của các phần tử logic và hệ thống mạch kết nối, để đạt được mục đích này thì kiến trúc của FPGA phức tạp hơn nhiều so với CPLD.

Một điểm khác biệt nữa với CPLD là trong những FPGA hiện đại được tích hợp nhiều bộ logic số học đã được tối ưu hóa, hỗ trợ RAM, ROM, tốc độ cao, hay các bộnhân, cộng dùng cho những ứng dụng xử lý tín hiệu số. Ngoài khả năng cấu trúc lại vi mạch ở mức toàn cục, một số FPGA hiện đại còn hỗ trợ cấu trúc lại ở mức cục bộ, tức là khả năng cấu trúc lại một bộ phận riêng lẻ trong khi vẫn đảm bảo hoạt động bình thường cho các bộ phận khác [18].

1.6 Khái niệm cơ bản và cấu trúc của FPGA

FPGA (Field-Programmable Gate Array) là vi mạch dùng cấu trúc mảng phần tử logic mà người dùng có thể lập trình được. Chữ field ở đây muốn chỉ đến khả năng tái lập trình "bên ngoài" tuỳ theo mục đích ứng dụng của người sử dụng, không phụ thuộc

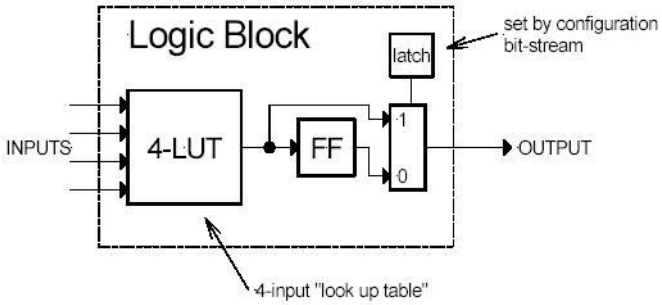
vào dây chuyền sản xuất phức tạp của nhà máy bán dẫn. Kiến trúc tổng quan về FPGA được mô tả như hình dưới đây.



Hình 1.6. Kiến trúc tổng quan FPGA [18]

Các khối logic cơ bản lập trình được (logic block)

- Phần tử chính của FPGA là các khối logic (logic block). Khối logic được cấu thành từ LUT và một phần tử nhớ đồng bộ flip-flop. LUT (Look up table) là khối logic có thể thực hiện bất kì hàm logic nào từ 4 đầu vào, kết quả của hàm này tùy vào mục đích mà gửi ra ngoài khối logic trực tiếp hay thông qua phần tử nhớ flip-flop.



Hình 1.7. Khối logic lập trình được của FPGA [18]

- Trong tài liệu hướng dẫn của các dòng FPGA của Xilinx còn sử dụng khái niệm SLICE,1 Slice gồm 4 khối logic tạo thành, số lượng các Slices thay đổi từ vài nghìn đến vài chục nghìn tùy theo loại FPGA.

Hệ thống mạch liên kết lập trình được

- Mạng liên kết trong FPGA được cấu thành từ các đường kết nối theo hai phương ngang và đứng, tùy theo từng loại FPGA mà các đường kết nối được chia thành các nhóm khác nhau, ví dụ trong XC4000 của Xilinx có 3 loại kết nối: ngắn, dài và rất dài. Các đường kết nối được nối với nhau thông qua các khối chuyển mạch lập trình được (programable switch), trong một khối chuyển mạch chứa một số lượng nút chuyển lập trình được, đảm bảo cho các dạng liên kết phức tạp khác nhau.

Khối vào/ra (IO Pads)

- Khối vào/ra nhiều hay ít là tuỳ thuộc vào từng loại FPGA. Chúng có thể được kết nối với các thiết bị bên ngoài như LED, USB, RS232, RAM....tuỳ theo mục đích sử dụng

Các phần tử tích hợp sẵn

- Ngoài các khối logic, tùy theo các loại FPGA khác nhau mà có các phần tử tích hợp thêm khác nhau, ví dụ để thiết kế những ứng dụng SoC, trong dòng Virtex 4, 5 của Xilinx có chứa nhân xử lý PowerPC, hay cho những ứng dụng xử lý tín hiệu số trong FPGA được tích hợp các DSP Slice là bộ nhân, cộng tốc độ cao, thực hiện hàm A*B+C, ví dụ dòng Virtex của Xilinx chứa từ vài chục đến hàng trăm DSP slices với A, B, C 18-bit.

  Nội dung trên được trích dẫn ở tài liệu tham khảo mục [18].

1.7 Các ứng dụng của FPGA

Ứng dụng của FPGA bao gồm: xử lý tín hiệu số, các hệ thống hàng không, vũ trụ, quốc phòng, tiền thiết kế mẫu ASIC (ASIC prototyping), các hệ thống điều khiển trực quan, phân tích nhận dạng ảnh, nhận dạng tiếng nói, mật mã học, mô hình phần cứng máy tính... Do tính linh động cao trong quá trình thiết kế cho phép FPGA giải quyết lớp những bài toán phức tạp mà trước kia chỉ thực hiện nhờ phần mềm máy tính, ngoài ra nhờ mật độ cổng logic lớn FPGA được ứng dụng cho những bài toán đòi hỏi khối lượng tính toán lớn và dùng trong các hệ thống làm việc theo thời gian thực.

1.8 Tổng kết chương

Trong chương đầu tiên này, đã trình bày khái quát về thuật toán Netflow V5, giới thiệu được bức tranh tổng quan về ngôn ngữ và công nghệ FPGA, trong chương tiếp theo sẽ trình bày cơ sở lý thuyết và kiến trúc của IP core Netflow V5 sử dụng trong hệ thống.

Chương 2: CƠ SỞ LÝ THUYẾT VÀ KIẾN TRÚC NETFLOW V5

2.1 Giới thiệu chương

Nội dung chính trong chương 2 này sẽ trình bày các lý thuyết về FPGA và đối tượng sử dụng trong quá trình thực hiện nghiên cứu, giới thiệu chi tiết các công cụ cần thiết về phần mềm cũng như kiến trúc của thuật toán Netflow V5 sử dụng cho đề tài này.

2.2 Giới thiệu về kit Virtex 6 và vi điều khiển nhúng Microbaze

2.2.1 Giới thiệu về kit FPGA Virtex 6

Hình 2.1. Kit Virtex-6 [11]

Virtex-6-ML605 là bo mạch phát triển nền tảng với bộ nhớ on-board và giao tiếp kết nối tiêu chuẩn công nghiệp.

Virtex-6-ML605 thích hợp cho việc dạy và nghiên cứu về các lĩnh vực [11]:

- Thiết kế số
- Hệ thống nhúng
- Giao tiếp và xử lý tín hiệu số
- Cấu trúc máy tính
- Hệ điều hành
- Mạng
- Xử lý ảnh, video
- Truyền nhận nối tiếp tốc độ cao

Cấu hình và ngoại vi của Virtex-6-ML605 Development System [11]:

- 2 Xilinx XCF32P Platform Flash PROMs (32 Mb) phục vụ lưu trữ cấu hình dung lượng lớn.

- Xilinx Virtex-6 XC6CCX240t-1ff156 FPGA
- Xilinx SystemACE Compact Flash configuration controller
- 64-bit wide 256Mbyte DDR2 small outline DIMM (SODIMM) module compatible with EDK supported IP and software drivers
- On-board 32-bit ZBT đồng bộ SRAM và Intel P30 StrataFlash
- 10/100/1000 tri-speed Ethernet PHY hỗ trợ giao tiếp MII, GMII, RGMII, và SGMII.
- Điều khiển USB host và ngoại vi
- Bộ phát xung clock hệ thống lập trình được.
- Stereo AC97 codec với line in, line out, headphone, microphone, và đầu cắm âm thanh số SPDIF.
- Cổng RS-232, 16x2 LCD, và các thiết bị I/O các cổng khác.

2.2.2 Vi điều khiển nhúng Microblaze

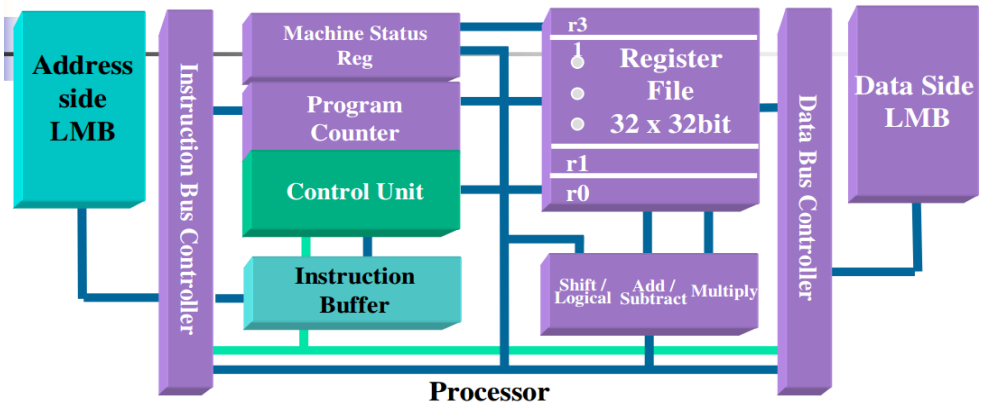
2.2.2.1 Khái niệm:

Microbalze là bộ xử lý mềm nhúng 32-bit của Xilinx. Lõi có cấu trúc Havard, tập lệnh thu gọn RISC (reduced Instruction Set Computer), với các bus riêng biệt để truy xuất dữ liệu và lệnh từ bộ nhớ on-chip và bộ nhớ ngoài cùng một thời điểm [17].

2.2.2.2 Kiến trúc và giao tiếp tín hiệu của Microblaze

Kiến trúc cơ bản của Microblaze [17]:

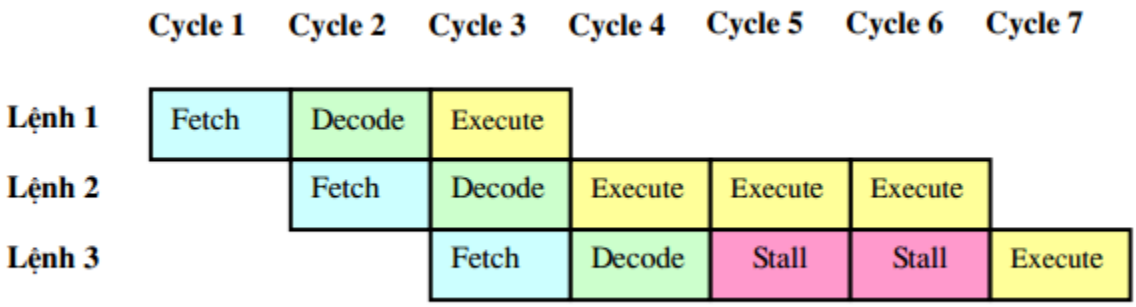
- Tầng xử lý đường ống với 32 thanh ghi mục đích chung 32 bits.
- Từ lệnh 32 bit với 3 toán hạng và 2 chế độ định địa chỉ.
- Đường bus 32 bit địa chỉ.
- Các đường lệnh và dữ liệu riêng biệt nối trực tiếp đến khối RAM trên chip thông qua LMB (Local Memory Bus).
- 1 khối ghi dịch.
- 2 cấp độ ngắt.
- Khối ALU (Arithmetic Logic Unit) : gồm các bộ cộng/trừ, ghi dịch/logic, nhân.
- Khối FPU (Floating Point Unit) : áp dụng các chuẩn dấu phẩy động đơn.



Hình 2.2. Kiến trúc vi điều khiển MicroBlaze [17]

- Trong đó:
  - Instruction Bus Controller: đường bus lệnh
  - Data Bus Controller: Đường bus dữ liệu
  - Instruction buffer: bộ đệm lệnh
  - Instruction decode: bộ giải mã lệnh
  - Program Counter: bộ đếm chương trình
  - Add/Sub: khối cộng/trừ; Shift/Logical: khối ghi dịch/logic; Multiply: khối nhân
  - Register File: thanh ghi dữ liệu 32thanh x 32 bit

Microblaze xử lý lệnh theo kiến trúc pipeline, hầu hết các lệnh của nó, mỗi giai đoạn mất 1 chu kỳ để thực hiện xong. Quá trình xử lý lệnh được chia làm 3 giai đoạn chính: nhận lệnh (Fetch), giải mã lệnh (Decode), thi hành lệnh (Execute).



Hình 2.3. Kiến trúc pipeline của MicroBlaze [17]

Giao tiếp tín hiệu trong Microblaze [17]

- Microblaze cung cấp 3 giao tiếp bộ nhớ:
- LMB: Local Memory Bus.
- PLB: Peripheral Local Bus.

OPB (On-chip Peripheral) và XCL (Xilinx Cache Link).

2.3 Chuẩn giao tiếp AXI

2.3.1 Giới thiệu AXI

AXI là một phần của AMBA (Advanced Microcontroller Bus Architecture), một họ bus vi điều khiển của ARM được giới thiệu lần đầu vào năm 1996.

Phiên bản AXI được được xuất hiện lần đầu trong AMBA 3.0, tung ra thị trường năm 2003. AMBA 4.0 được tung ra thị trường vào năm 2010, gồm trong đó phiên bản thứ hai của AXI và AXI 4.

Có 3 loại AXI:

- AXI4: cho yêu cầu ánh xạ bộ nhớ hiệu suất cao.
- AXI-4Lite: cho giao tiếp ánh xạ bộ nhớ thông lượng thấp, đơn giản.
- AXI-STREAM: Cho truyền dữ liệu tốc độ cao.

2.3.2 Chuẩn giao tiếp AXI-4Lite

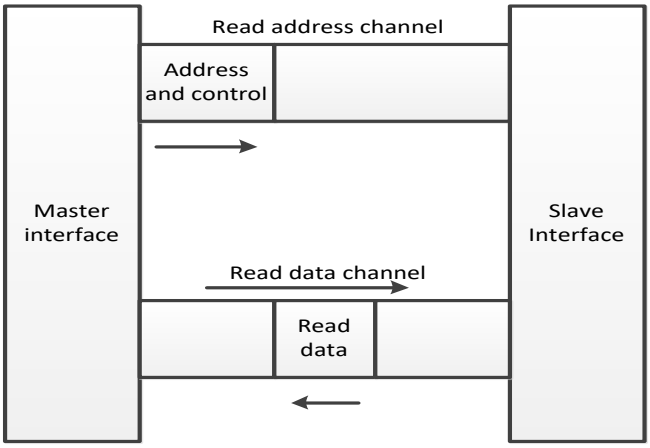
AXI-Lite cho phép giao tiếp với thanh ghi (Control & Status interface)

Giao tiếp AXI-Lite gồm 5 kênh khác nhau:

- Kênh đọc địa chỉ
- Kênh ghi địa chỉ
- Kênh đọc dữ liệu
- Kênh ghi dữ liệu
- Kênh phản hồi ghi

Dữ liệu có thể di chuyển đồng thời theo cả hai hướng giữa master và slave. AXI-Lite chỉ cho phép một dữ liệu trao đổi trong mỗi lần thực hiện.

Hình dưới đây mô tả quá trình đọc của AXI-Lite sử dụng kênh đọc địa chỉ và kênh đọc dữ liệu:



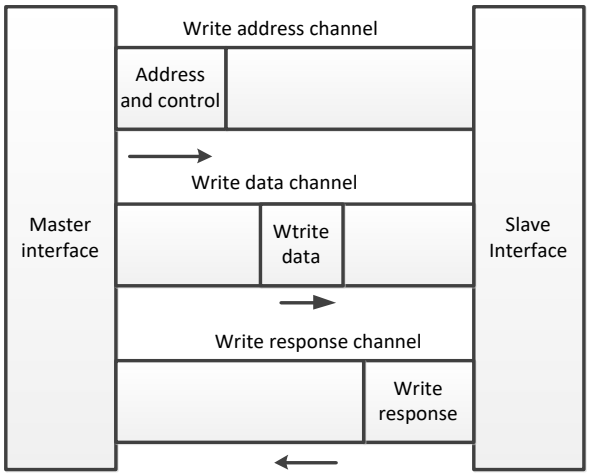
Hình 2.4. Quá trình đọc của bus AXI-Lite [12]

Các tín hiệu AXI-Lite read

Bảng 2.1 Các tín hiệu AXI-Lite read

Kênh	Tín hiệu	Mô tả
Read address	ARADDR	Địa chỉ đọc
	ARVALID	Chỉ ra địa chỉ đọc là valid
	ARREADY	Tín hiệu điều khiển luồng
Read data	RDATA	Dữ liệu đọc
	RRESP	Tín hiệu phản hồi đọc
	RVALID	Chỉ ra dữ liệu đọc được là valid
	RREADY	Tín hiệu điều khiển luồng

Hình dưới đây mô tả quá trình ghi của AXI-Lite sử dụng kênh ghi địa chỉ, kênh ghi dữ liệu và kênh phản hồi ghi.



Hình 2.3. Quá trình ghi của bus AXI-Lite [12]

Các tín hiệu AXI-Lite write:

Bảng 2.2: Các tín hiệu AXI-Lite write

Kênh	Tín hiệu	Mô tả
Write address	AWADDR	Địa chỉ ghi
	AWVALID	Chỉ ra địa chỉ ghi là valid
	AWREADY	Tín hiệu điều khiển luồng
Write data	WDATA	Dữ liệu ghi
	WSTRB	Chỉ ra byte nào của dữ liệu ghi là valid
	WVALID	Chỉ ra dữ liệu ghi là valid

	WREADY	Tín hiệu điều khiển luồng
Write response	BRESP	Tín hiệu phản hồi ghi
	BVALID	Cho biết tín hiệu phản hồi ghi là valid
	BREADY	Tín hiệu điều khiển luồng của kênh phản hồi ghi

Tín hiệu clock/reset của AXI-Lite:

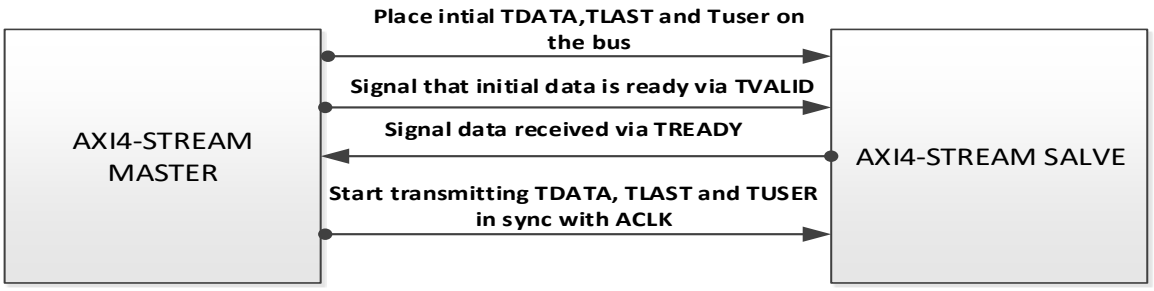
Bảng 2.3: Các tín hiệu clock/reset của AXI-Lite

Kênh	Tín hiệu	Mô tả
Clock/reset	ACLK	Tín hiệu clock AXI-Lite tần số 100Mhz, có thể được sử dụng bởi nhiều interface AXI-Lite từ một nguồn duy nhất
	ARESETN	Tín hiệu reset, tích cực mức thấp, có thể được sử dụng bởi nhiều interface AXI-Lite từ một nguồn duy nhất

AXI-Lite cung cấp các kết nối dữ liệu và địa chỉ riêng lẻ cho đọc và ghi, điều này cho phép việc truyền dữ liệu liên tục, song công.

2.3.3 Chuẩn giao tiếp AXI_STREAM

Giao thức AXI-Stream định rõ một kênh riêng cho truyền luồng dữ liệu. Kênh AXI-Stream được mô hình dựa trên kênh ghi dữ liệu của AXI-Lite.



Hình 2.4. Kết nối chế độ Master Slave theo đường bus AXI4-Stream

Tín hiệu clock/reset

Bảng 2.4: Các tín hiệu clock/reset của axi4-stream

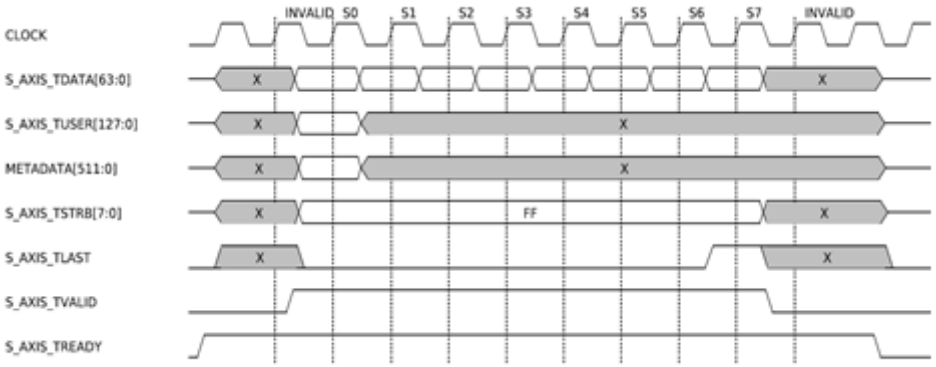
Tín hiệu	Mô tả
ACLK	Tín hiệu clock AXI-Stream tần số 125Mhz

ARESETN	Tín hiệu reset, tích cực mức thấp

Tín hiệu trong bus AXI-Stream

Bảng 2.5: Các tín hiệu của giao thức Axi-stream

Tín hiệu	Độ rộng (bit)	Mô tả
TVALID	1	Chỉ ra dữ liệu là valid
TDATA	64	AXI-Stream data, dữ liệu của packet (bao gồm cả header và payload).
TREADY	1	Tín hiệu cho thấy module sau sẵn sàng nhận gói packet tiếp theo
TSTRB	8	AXI-Stream data có độ dài 8 byte, tín hiệu này cho biết byte nào của AXI-stream data là valid. Tín hiệu tích cực mức cao.
TLAST	1	End of packet, cho biết kết thúc một gói packet.
TUSER	128	Tín hiệu cho biết packet length, port nguồn, port đích.



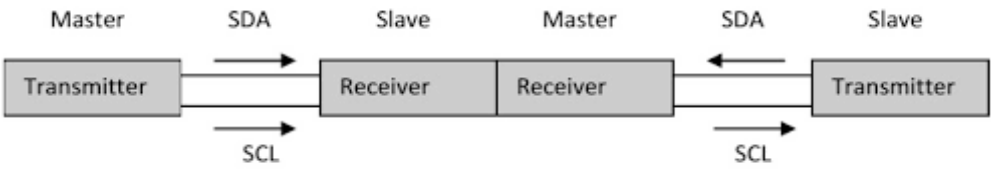
Hình 2.5. Giản đồ dạng sóng của AXI-Stream [13]

2.4 Các chuẩn giao tiếp UART và IIC

2.4.1 Chuẩn giao tiếp IIC

I2C (Inter - Intergrated Circuit) là bus giao tiếp giữa các IC với nhau, là chuẩn kết nối một số ICs với các thiết bị ngoại vi đạt được hiệu quả cho phần cứng tốt nhất với mạch điện đơn giản.

Một giao tiếp I2C gồm có 2 dây: Serial Data (SDA) và Serial Clock (SCL). SDA là đường truyền dữ liệu 2 hướng, còn SCL là đường truyền xung đồng hồ và chỉ theo một hướng. Như hình vẽ trên, khi một thiết bị ngoại vi kết nối vào đường I2C thì chân SDA của nó sẽ nối với dây SDA của bus, chân SCL sẽ nối với dây SCL.[14]



Hình 2.6. Chuẩn giao tiếp I2C [14]

2.4.1.1 Module RTCC DS3231

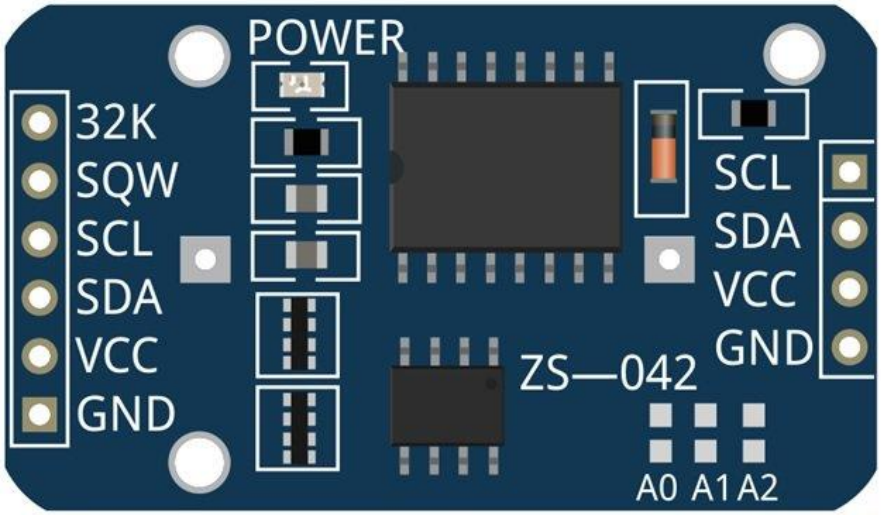
Moudule thời gian thực RTC DS3231 là IC thời gian thực chính xác với thạch anh tích hợp sẵn có khả năng điều chỉnh nhiệt. IC có đầu vào cho pin riêng, tách biệt khỏi nguồn chính đảm bảo chính đảm cho việc giữ thời gian chính xác. Thạch anh tích hợp sẵn giúp tăng độ chính xác trong thời gian dài hoạt động và giảm số lượng linh kiện vần thiết khi làm board.

Thời gian trong trong IC được giữ ở định dạng: giờ, phút, giây, ngày, thứ, tháng, năm. Các tháng ít hơn 31 ngày sẽ tự động được điều chỉnh, các năm Nhuận cũng được chỉnh đúng số ngày. Thời gian có thể hoạt động ở chế độ 24h hoặc AM/PM. Các giao tiếp IC được thực hiện thông qua I2C bus.

Trong chip có mạch điện áp chuẩn dùng để theo dõi trạng thía của nguồn VCC, phát hiện lỗi nguồn, tự động chuyển nguồn khi có vấn đề. Có tín hiệu Reset xuất ra cho mạch ngoài, MCU khi nguồn điện hồi phục trạng thái [14].

Các thông số kỹ thuật

- Size: dài 38mm, rộng 22mm, cao 14mm.
- Khối lượng: 8g.
- Điện thế hoạt động: 3.3V – 5.5V.
- Clock: high-precision clock on chip DS3231.
- Clock Accuracy: 0-40^{o}C range, the accuracy 2ppm, the error was about 1 minute.
- Thông tin thời gian: giờ, phút, giây, ngày, thứ, tháng, năm đến năm 2100.
- I2C bus có tốc độ tối đa 400Khz.
- Kèm thêm pin sạc được CR2032.



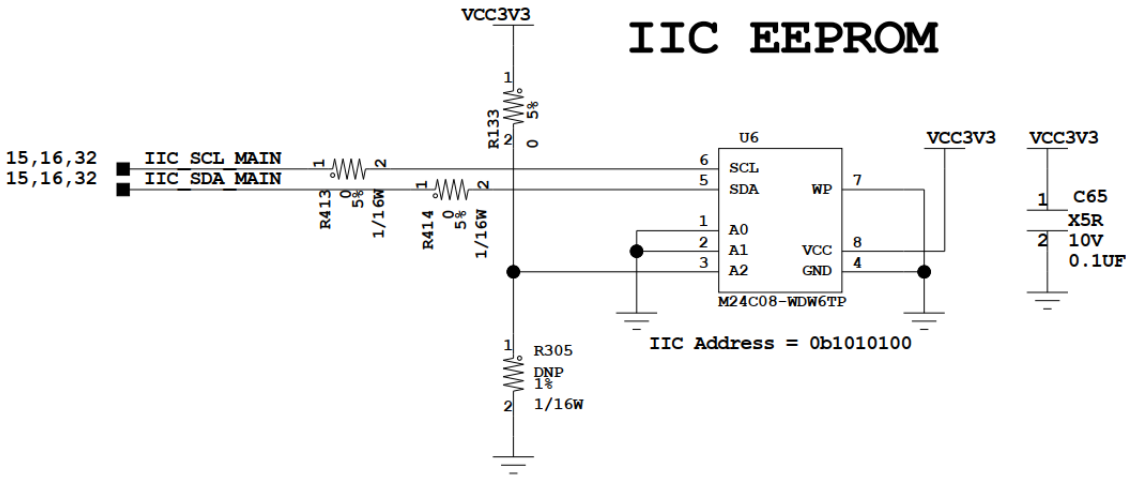
Hình 2.7. Module RTC DS3231 [14]

2.4.1.2 Chip EEPROM on board ML605

Thiết kế layout của board ML605 đã tích hợp sẵn IC EEPROM (Electrically Erasable PROgrammable Memory) của hãng STMicroelectonics với model M24C08-WDW6TP [11].

Các thông số kỹ thuật

- Tổ chức bộ nhớ: 1K x 8 bits.
- Kích thước bộ nhớ: 8Kbit.
- Ngưỡng điện áp cho phép hoạt động từ 2.5V-5.5V.
- Giao tiếp bằng chuẩn I2C bus có tốc độ tối đa 400Khz.
-



Hình 2.8. IIC EEPROM trên FPGA Virtex-6 Kit [11]

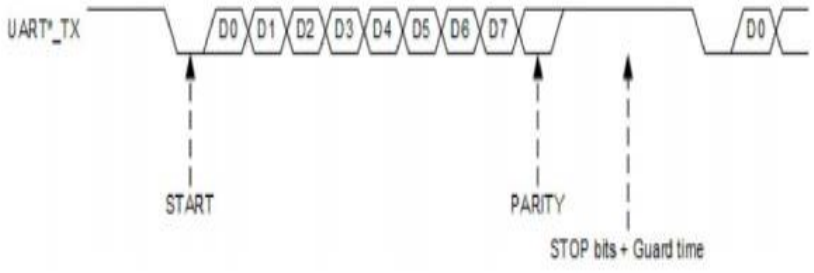
2.4.2 Chuẩn giao tiếp UART:

UART – Universal asynchronous receiver transmitter là bộ truyền nhận nối tiếp bất đồng bộ. UART là một ngoại vi cơ bản trong chip STM32F103C8T6 thường được dùng trong các quá trình giao tiếp với các loại module như: Zigbee, Bluetooth, Wifi…[15]

UART bao gồm hai thành phần là máy phát và máy thu được . Phần máy phát bao gồm ba khối là thanh ghi giữ truyền, thanh ghi dịch chuyển và logic điều khiển. Tương tự, phần máy thu bao gồm một thanh ghi giữ, thanh ghi thay đổi và logic điều khiển. Hai phần này thường được cung cấp bởi một bộ tạo tốc độ baud. Trình tạo này được sử dụng để tạo tốc độ khi phần máy phát và phần máy thu phải truyền hoặc nhận dữ liệu.

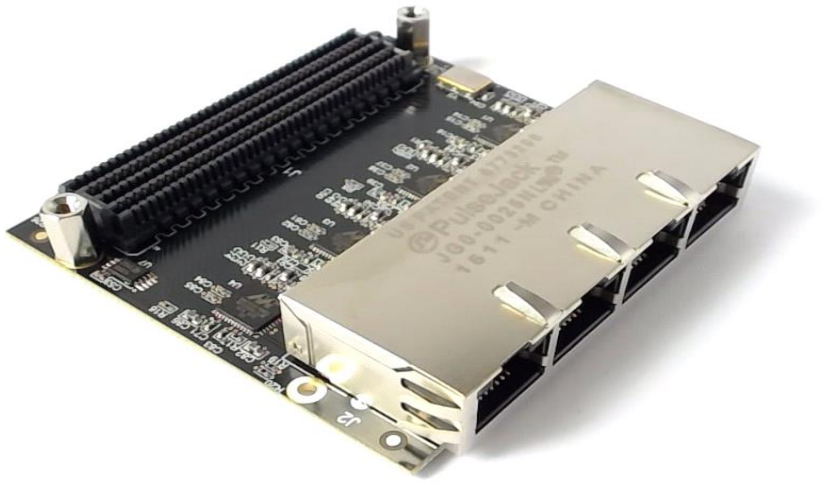
Thanh ghi giữ trong máy phát bao gồm byte dữ liệu được truyền. Các thanh ghi thay đổi trong máy phát và máy thu di chuyển các bit sang phải hoặc trái cho đến khi một byte dữ liệu được truyền hoặc nhận. Một logic điều khiển đọc (hoặc) ghi được sử dụng để biết khi nào nên đọc hoặc viết.

Máy phát tốc độ baud giữa máy phát và máy thu tạo ra tốc độ dao động từ 110 bps đến 230400 bps. Thông thường, tốc độ truyền của vi điều khiển là 9600 đến 115200 [15]



Hình 2.9. Khung dữ liệu truyền/nhận UART [15]

2.5 ETHERNET FMC



Hình 2.10. Hình ảnh Ethernet FMC [17]

Ethernet FMC là một card bổ sung 4 cổng Ethernet Gigabit bằng kết nối FMC.

Một số đặc điểm:

- Marvell Gigabit Ethernet PHY cho khả năng tương thích và hiệu suất mạng tối đa.
- Đầu nối FMC với Low-pin-count Vita 57 để tương thích với tất cả các kết nối với FMC
- Các dao động đầu ra LVDS khác nhau để cung cấp clock cho MAC Ethernet

Một số ứng dụng

- Kết nối các thiết bị với nhau qua mạng máy tính.
- Chuyển tiếp gói tin giữa cá mạng máy tính với độ trễ tối thiểu
- Đo lường, theo dõi và phân tích thời gian phản hồi thiết bị trong thời gian thực
- Kiểm tra các giao thức thử nghiệm trên các mạng hiện có
- Loại bỏ tắc nghẽn mạng bằng các sử dụng kiểm soát lưu lượng tăng tốc phần cứng

Các thông số kỹ thuật

- 4 cổng Ethernet 10/100 / 1000Mbps
- LPC FMC: Pin tương thích với ZedBoard và các cung cấp FMC tương thích khác.
- Clock 125 MHz cung cấp cho MAC

Các thiết kế có sẵn cho ZedBoard và các board phát triển FPGA [17].

2.6 Tổng quan về Ethernet

2.6.1 Giới thiệu về Ethernet

Ethernet là một họ lớn và đa dạng gồm các công nghệ mạng dựa vào khung dữ liệu (frame-based) dành cho mạng LAN, Trên Ethernet được xuất phát từ khái niệm Ête trong ngành vật lý học. Ethenet định nghĩa một loạt các chuẩn nối dây và phát tín hiệu cho tầng vật lý, hai phương tiện để truy nhập mạng tại phần MAC (điều khiển truy cập môi trường truyền dẫn) của tầng liên kết dữ liệu và một định dạng chung cho việc đánh địa chỉ. Ethernet đã được chuẩn hóa thành các tiêu chuaant IEEE 802.3 [22].

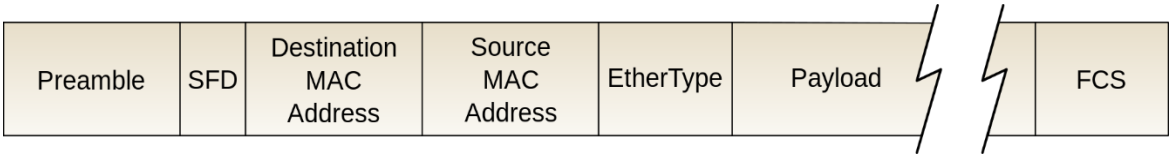
2.6.2 Một số đặc điểm của Ethernet

Cấu trúc mạng hình sao, hình thức nối dây cáp xoắn (twised pair) của Ethernet đã trở thành công nghệ LAN được sử dụng rộng rãi nhất từ thập kỷ 1990 cho tới nau. Trong những năm gần đây, WiFi, dạng LAN không dây đã được chuẩn hóa bởi IEEE 802.11 đã được sử dụng bên cạnh hoặc thay thế cho Ethernet trong nhiều cấu hình mạng. IEEE đã phát triển chuẩn Ethernet trên nhiều công nghệ truyền dẫn khác nhau thay vì có nhiều

loại mạng Ethernet. Mỗi mạng được mô tả dựa vào ba nguyên tố: tốc độ, phương pháp tín hiệu sử dụng và đặc tuyến đường truyền vật lý.

- Các hệ thống Ethernet 10Mb/s.
- Các hệ thống Ethernet 100Mb/s – Ethernet tốc độ cao (Fast Ethernet).
- Các hệ thống Ethernet 1gb/s.

Định dạng khung trong Ethenet: sẽ chia dữ liệu thành nhiều khung (frame). Khung là một gói thông tin được truyền như một đơn vị duy nhất. Khung trong Ehternet có thể dài từ 64 đến 1518 byte, nhưng bản thân khung Ethernet đã sủ dụng ít nhất 18 byte, nên dữ liệt một khung Ethernet có thể dài từ 46 đến 1500 byte. Mỗi khung đều có chứa thông tin điều khiển và tuân theo một tổ chức cơ bản. Ví dụ khung Ethernet (dùng cho TCP/IP) được truyền qua mạng với các thành phần sau:



Hình 2.11. Cấu trúc Frame của Ethernet [22]

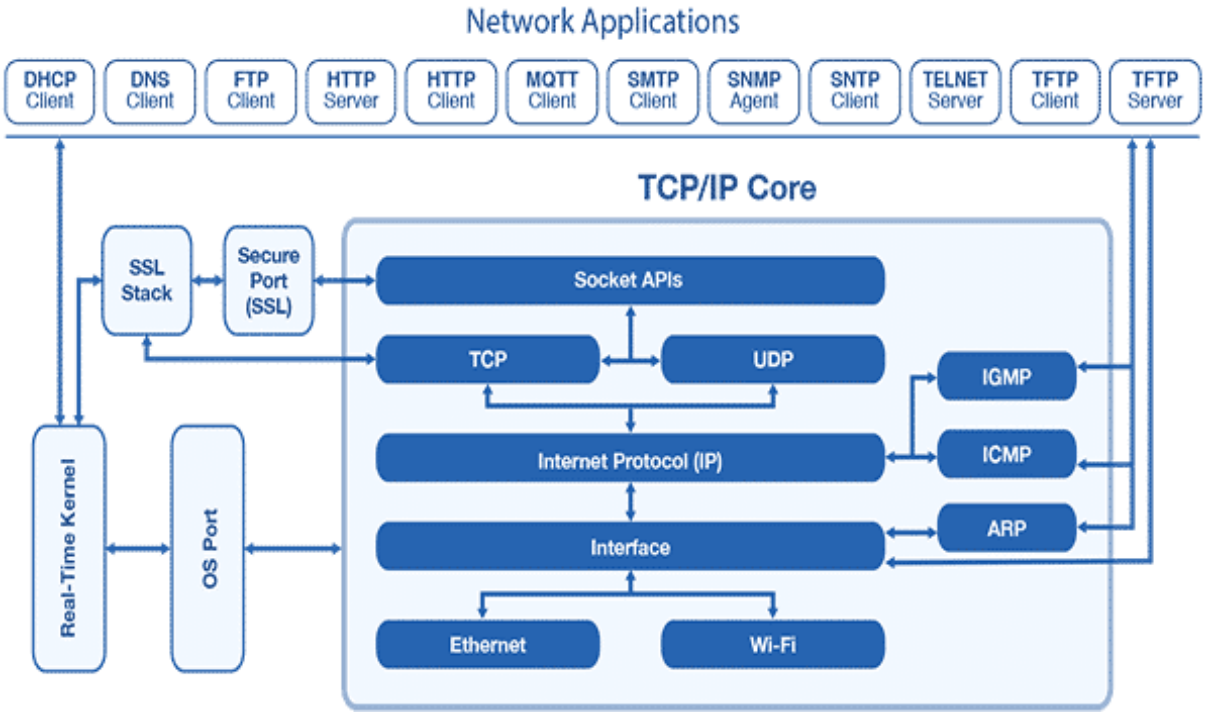
Trong đó:

- Preamable + SFD: 8-byte báo hiệu bắt đầu của một frame.
- Destination MAC Address: 6-byte thể hiện địa chỉ MAC đích.
- Source MAC Address: 6-byte thể hiện địa chỉ MAC nguồn.
- EherType: 2-byte chỉ rõ giao thức lớp mạng.
- Payload: dữ liệu chuyển đi.
- CRC: 4-byte dùng để kiểm tra lỗi của Frame (Cyclic Redundancy Check).

2.6.3 Giao thức TCP/IP

TCP/IP là viết tắt của Transmission Control Protocol / Internet Protocol – Giao thức điều khiển truyền thông/ Giao thức Internet. Các tầng trong mô hình này là:

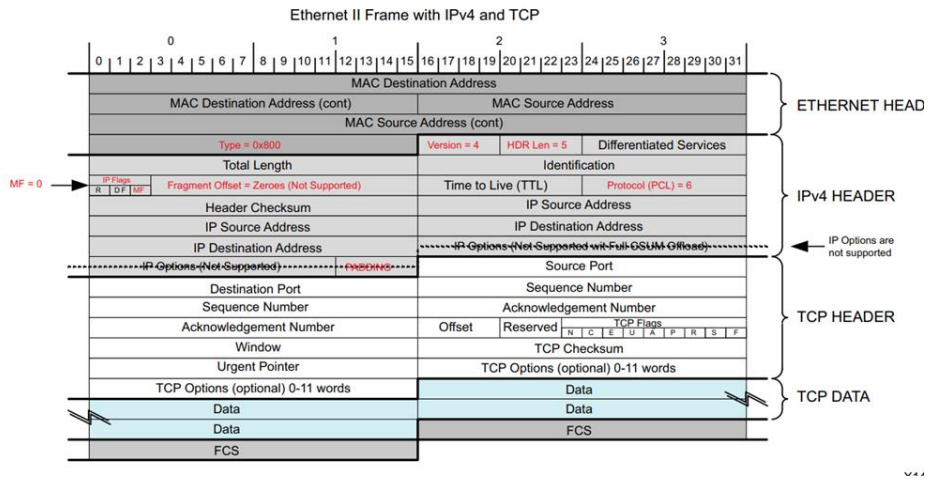
- Lớp ứng dụng (Application Layer).
- Lớp giao vận (Transport Layer).
- Lớp liên kết mạng (Internet Layer).
- Lợp giao tiếp mạng (Network Interface Layer).



Hình 2.12. Mô hình giao thức TCP/IP [22].

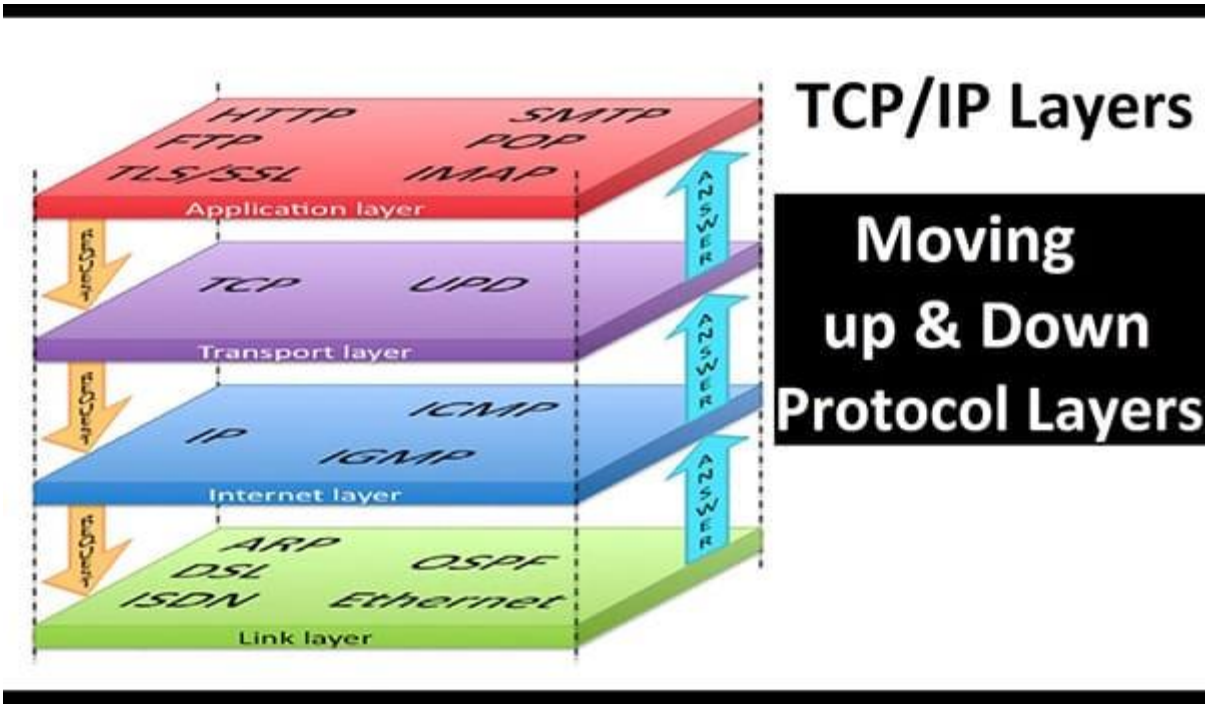
2.6.4 Lớp giao vận (Transport Layer)

Nhiệm vụ của tầng vận chuyển là thiết lập phiên truyền thông giữa các máy tính và quy định cách truyền dữ liệu. Trong mô hình TCP/IP có sự kết hợp giữa các giao thức riêng biệt. FPT, HTTP, HTTPS ba giao thức được sử dùng nhiều nhất của TCP/Ip. FTP là giao thức giúp máy tính có thể gửi dữ liệu không giới hạn đến một hay nhiều máy tính khác. HTTP có chức năng truyền dữ liệu không an toàn giữa người dùng web và máy chủ web. HTTPS là giao thức được sử dụng để truyền dữ liệu an toàn giữa người dùng và máy chủ web.



Hình 2.13. Cấu trúc gói TCP [22]

Sinh viên thực hiện : Nguyễn Duy Hà Sơn – Đặng Sỹ Phi Hùng
Người hướng dẫn : TS. Trần Hoàng Vũ



Hình 2.14. Mô hình tầng TCP/IP [22]

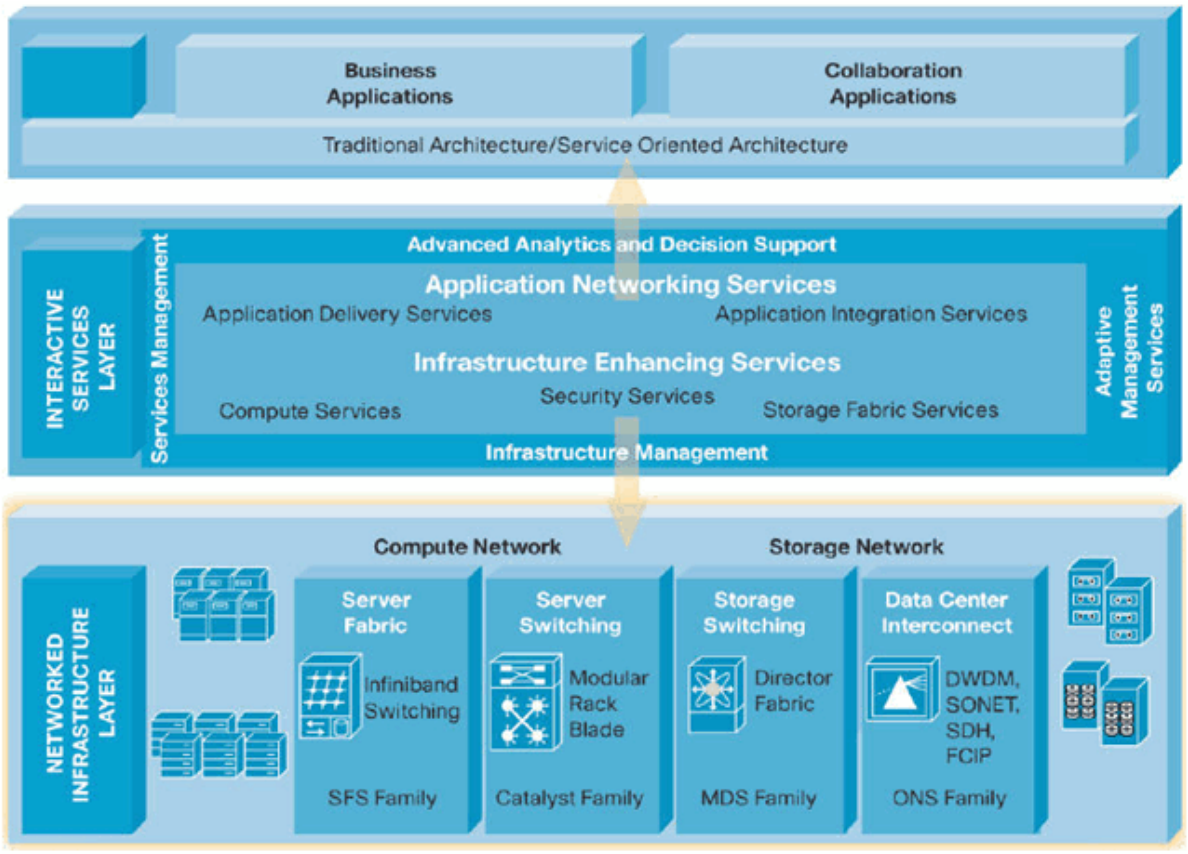
Ưu điểm của TCP/IP là:

- Mô hình TCP/IP không chịu sự kiểm soát của tổ chức nào. Do đó, người dùng có thể tự do sử dụng.
- TCP/IP có khả năng tương thích với các mạng, hệ điều hành và phần cứng máy tính.
- TCP/IP có khả năng định tuyến, mở rộng và nhận được đường dẫn tố nhất thông qua mạng.

2.7 Trung tâm dữ liệu (DATA CENTER)

Sự phát triển của xu hướng tập trung hóa và ảo hóa các nguồn nhân lực tại Trung tâm dữ liệu yêu cầu một nền tảng có khả năng đảm bảo an ninh, an toàn và có khả năng mở rộng cao cho Trung tâm dữ liệu. Trung tâm dữ liệu là nơi chứa các hệ thống quan trọng của network và đóng vai trò vô cùng quan trọng đối với tính liên tục của các hoạt động hàng ngày của một tổ chức. Tại đây cho phép lưu trữ, quản lý và phân phối dữ liệu của tổ chức đó.

Cơ sở dạ tầng là một kết cấu an toàn cho người dùng đầu cuối tới các dịch vụ của Trung tâm dữ liệu và là cơ sở vật chất cho việc phát triển, kết nối, và tập trung các thành phần chia sẽ của Trung tâm dữ liệu khi cần, bao gồm các ứng dụng, các hệ thống máy chủ, các thiết bị, và các hệ thống lưu trữ. Một hệ thống Trung tâm dữ liệu được xây dựng và lập kế hoạch tốt cung cấp khả năng bảo vệ sự toàn vẹn của dữ liệu và dịch vụ, tối ưu hóa các ứng dụng về mặt hiệu năng và độ sẵn sàng, cho phép đáp ứng một cách nhanh chóng những yêu cầu thay đổi về mặt thị trường, mức độ ưu tiên trong kinh doanh và sự phát triển công nghệ.



Hình 2.15. Mô hình System DataCenter [21]

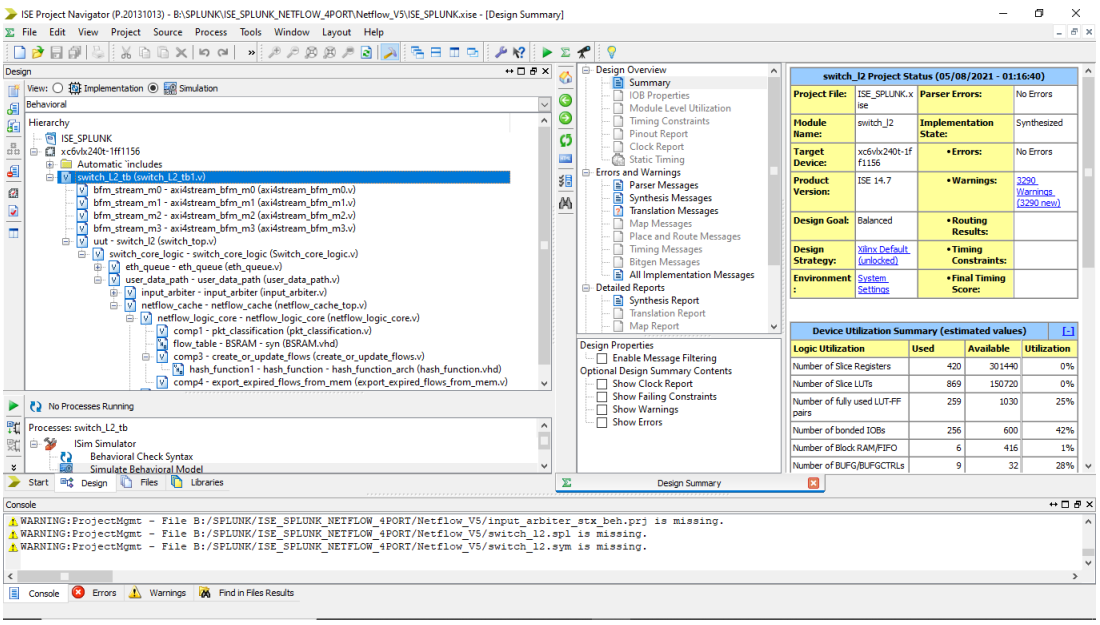
2.8 Một số công cụ phần mềm sử dụng trong dự án

2.8.1 Xilinx ISE

Phần mềm ISE (Intergrated Software Enviroment) là một môi trường thiết kế hoàn hảo củ Xilinx, cung cấp cho người thiết kế hầu hết các công cụ cần thiết để có thể hoàn thành một đề án thiết kế nhanh nhất và hiệu quả nhất. ISE tích hợp các công nghệ tiên tiến nhất, sử dụng thao tác linh hoạt và có giao diện GUI thân thiện với người sử dụng.

Một số ưu điểm của ISE:

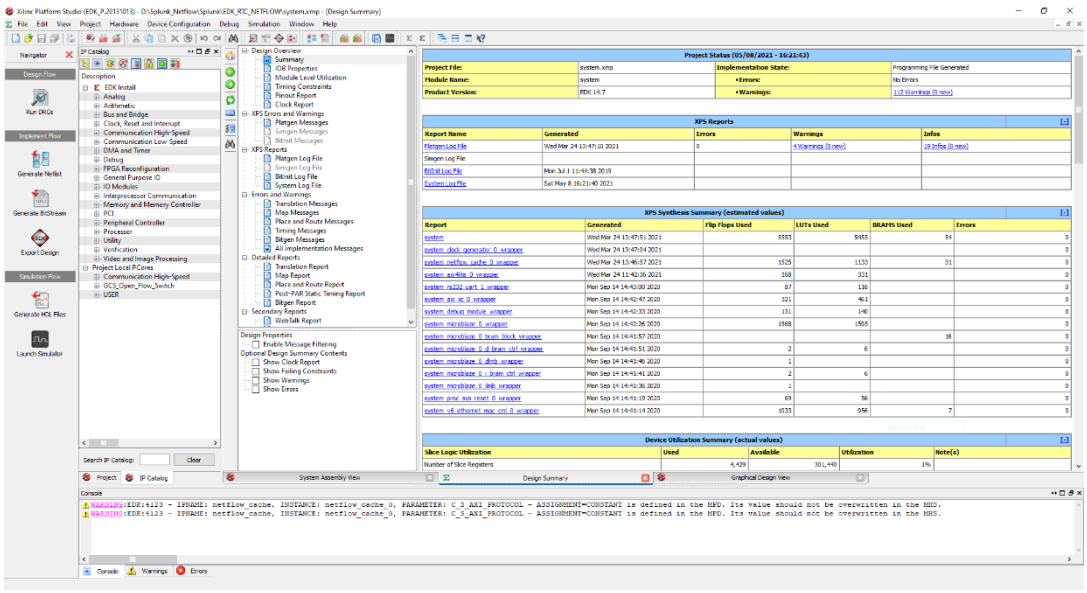
- Tân dụng tối đa tất cả các công nghệ tiên tiến nhất của PLD.
- Tiết kiệm thời gian thiết kế, hỗ trợ nhiều dòng sản phẩm của Xilinx.
- Tăng hiệu quả và giảm giá thành.
- Hỗ trợ tối đa cho việc thiết kế các hệ thống nhúng.



Hình 2.16. Giao diện của Xilinx ISE

2.8.2 Xilinx Platform Studio

XPS (Xilinx Platform Studio) là một công cụ chính của bộ thiết kế phiên bản nhúng ISE, có khả năng cấu hình và tích hợp lõi IP và IP core hỗ trợ sẵn của Xilinx, với các thiết kế Verilog và VHDL hoặc tùy chỉnh. Hỗ trợ xây dựng, kết nối và cấu hình cho các hệ thống dựa trên bộ xử lý nhúng từ các máy trạng thái đơn giản đến các hệ vi xử lý RISC 32 bit.



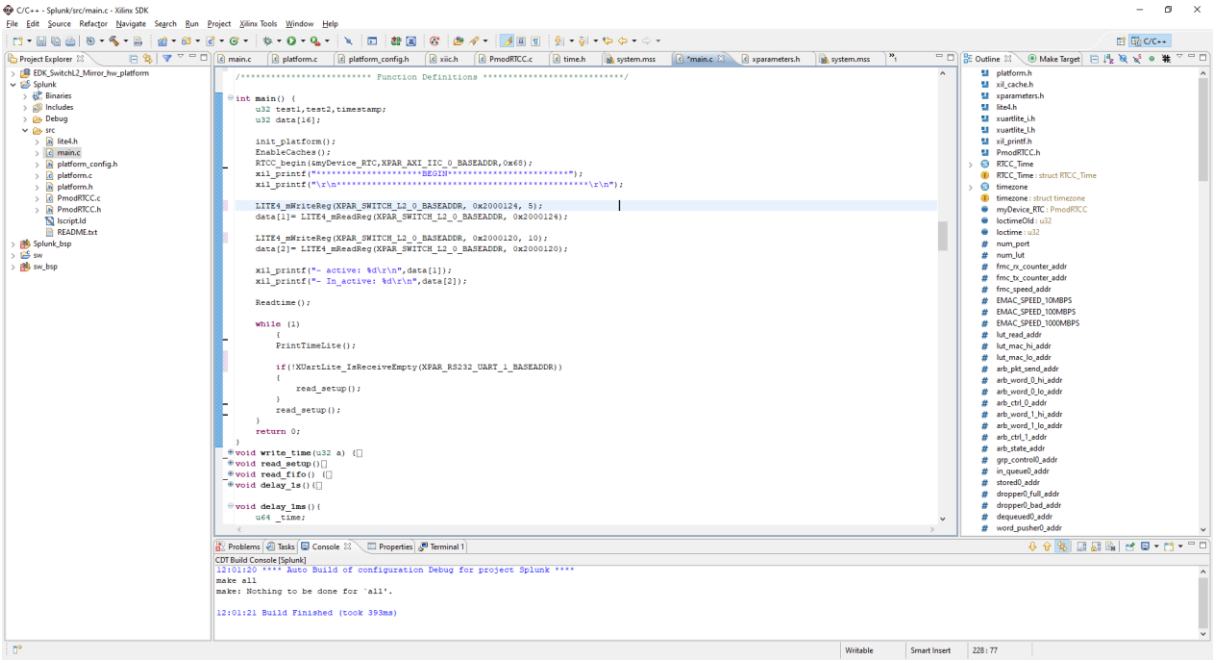
Hình 2.17. Giao diện chính của Xilinx XPS

2.8.3 Xinlinx SDK

SDK (Software Development Kit) là công cụ được phát triển bởi hãng Xilinx để soạn thảo trình điều khiển ngôn ngữ C/C++, debug và nạp mã lệnh xuống Kit FPGA của Xilinx.

Một số tính năng của SDK:

- Trình biên dịch C/C++ và tích hợp sửa lỗi.
- Kiểm soát phiên bản mã nguồn mở, cấu hình xây dựng ứng dụng và tạo Makefile tự động.
- Phân tích hiệu suất của hệ thống và tích hợp các công cụ để cấu hình cho FPGA.



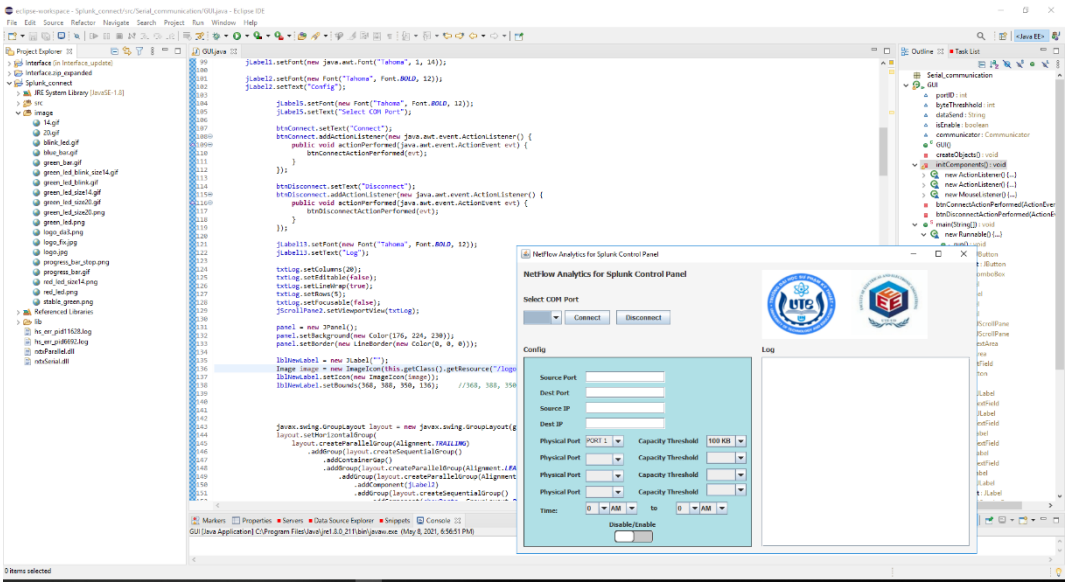
Hình 2.18. Giao diện chính của Xilinx SDK

2.8.4 Eclipse Java Development

Eclipse là một công cụ hỗ trợ lập trình miễn phí, mạnh mẽ. Nó có chứa nhiều công cụ hữu ích để khởi tạo, chạy và tối ưu hóa code Java.

Ưu điểm của Eclipse Java Development là:

- Tạo thuận tiện cho tích hợp liền mạch các công cụ bên trong mỗi một và xuyên qua nhiều nội dung và các nhà cung cấp công cụ khác.
- Hỗ trợ việc xây dựng nhiều công cụ.
- Hỗ trợ các công cụ thao tác các kiểu nổi dung bất kỳ (Java, JSP,..).
- Hỗ trợ cả môi trường phát triển ứng dụng GUI lẫn không dựa trên GUI.
- Chạy trên nhiều hệ điều hành Windows và Linux.



Hình 2.19 Giao diện chính của Eclipse Java Development

2.8.5 Ngôn ngữ lập trình Java

Java được biết tới là một trong những ngôn ngữ lập trình phổ biến nhất trên thế giới ngày nay. Đây là một ngôn ngữ lập trình máy tính đa mục đích và nó đã đạt được sự phổ biến nhờ vào tính định hướng đối tượng, đồng thời và dựa trên class.

Chương trình viết bằng ngôn ngữ lập trình Java có thể chạy trên bất kỳ hệ thống ảo nào có cài máy ảo Java (Jave Virtual Machine).

Một số ưu điểm của ngôn ngữ lập trình java:

- Là ngôn ngữ thuần hướng đối tượng.
- Java được sử dụng trên mọi thiết bị.
- Là ngôn ngữ có mã nguồn mở.

Java dễ thực thi, sử dụng và dễ tiếp cận.

- Được hỗ trợ IDE miễn phí.
- Multi- Threading.

2.8.6 Splunk Enterprise:

Splunk là một phần mềm giám sát mạng dựa trên sức mạng của việc phân tích Log. Splunk thực hiện các công việc tìm kiếm, giám sát và phân tích các dữ liệu được sinh ra từ các ứng dụng, các hệ thống và các thiết bị hạ tầng mạng. Nó có thể thao tác tốt với nhiệu định dạng dữ liệu khác nhau (Syslog, csv, apache-log, access_combined...). Splunk được xây dựng dựa trên nền tảng Lucene and MongoDB. *Nội dung tham khảo tài liệu [16]*:

Hiện tại, Splunk có ba phiên bản, bao gồm:

- Splunk Enterprise cho các khách hàng có nhu cầu xử lý log tại chỗ với khối lượng lớn.
- Splunk Cloud cho các khách hàng tải log lên nền tảng đám mây của Splunk để xử lý.

- Splunk Light cho các khách hàng có nhu cầu xử lý log tại chỗ với khối lượng vừa và nhỏ.
  Hạn chế nhất của Splunk chính là chi phí cài đặt lớn, do khoản đầu từ ban đầu cho hệ thống thiết bị chuyên dụng có độ phức tạp cao. Một vấn đề khác là phí bản quyền hàng năm của Splunk cũng rất đắt đỏ (ước tính có thể lên đến hàng chục ngàn đô-la Mỹ mỗi năm).

Hình 2.20. Giao diện chính của Splunk Enterprise [21]

2.9 TỔNG KẾT CHƯƠNG

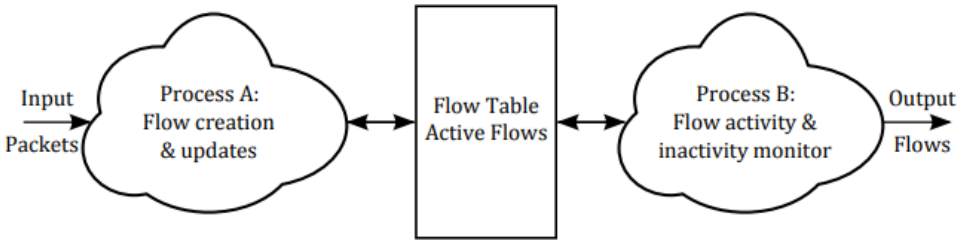
Qua chương 2 đã phần nào giúp chúng ta hiểu được lý thuyết cơ bản về FPGA và thiết kế với FPGA, chuẩn giao tiếp AXI4-Stream và AXI4-Lite, chuẩn giao tiếp I2C, UART, cấu trúc của một gói tin, cũng như các công cụ phần mềm cần thiết để xây dựng hệ thống. Tuy nhiên, đây cũng chỉ mới là các cơ sở lý thuyết nên để có thể hiểu rõ hơn quy trình hoạt động cụ thể của kỹ thuật đã nêu ở trên như thế nào ta sẽ tiếp tục đến chương 3, đề cập đến việc thực thi thiết kế phần cứng sử dụng ngôn ngữ verilog và nhúng trên nền tảng FPGA X

Chương 3: THIẾT KẾ IPCORE NETFLOW V5 VÀ THỰC HIỆN NHÚNG TRÊN FPGA CỦA XILINX

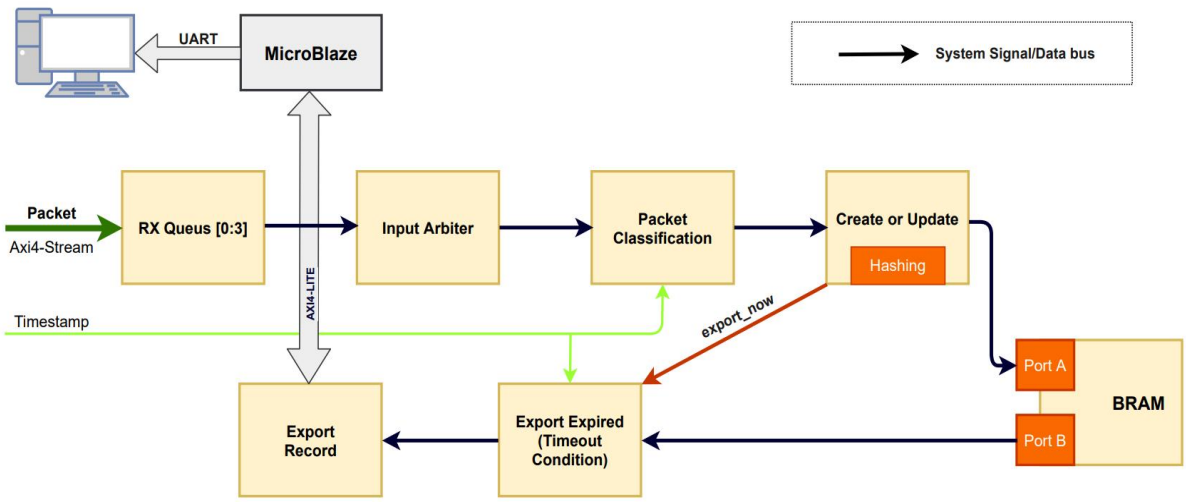
3.1 Giới thiệu chương

Trong chương này sẽ tiến hành việc thực hiện thuật toán NetFlow V5 dựa trên cơ sở lý thuyết và kiến trúc được trình bày ở chương 2. Quá trình thiết kế được xem xét ở mức độ tổng thể như xem xét các khối chức năng, độ dài dữ liệu, các tín hiệu giao tiếp như tín hiệu trạng thái, tín hiệu điều khiển. Mô tả phần cứng các module với ngôn ngữ Verilog, chi tiết chức năng cũng như cách hoạt động của từng module. Thực thi nhúng NetFlow V5 IPCore vào hệ thống trên board Virtex-6 ML605 và kiểm tra hoạt động core trên ModulSim và trên môi trường thực tế.

3.2 Sơ đồ khối tổng quát của hệ thống:



Hình 3.1 Tổng quan của hệ thống



Hình 3.2 Sơ đồ khối chi tiết các khối chức năng của hệ thống

Sinh viên thực hiện : Nguyễn Duy Hà Sơn – Đặng Sỹ Phi Hùng
Người hướng dẫn : TS. Trần Hoàng Vũ

Giải thích luồng hoạt động của sơ đồ:

Sơ đồ khối chức năng của NF-BRAM gồm có 6 khối Packet parser, Hashing Module, Create/ Update Flows, Flow Table (BRAM), Timeout Conditons Monitor, Export Module.

- Gói tin sẽ được đẩy vào trong các port tại 4 module Rx Queus tương ứng với 4 Port đầu vào của IP Core, luồng dữ liệu vào với 32bit data được lưu vào Fifo bên trong Rx Queus.

- Module Input Arbiter có 4 đầu vào sẽ nhận data với 64-bit được lấy ra từ Fifo của tương ứng với 4 module Rx Queues. Tại module Input Arbiter các gói tin sẽ được lưu vào fifo và được lấy ra theo cơ chế Round Robin ra theo 1 port và được đẩy sang module Packet Classification để xử lý.

- Tại module Packet Classification các gói tin được nhận theo 64-bit data sẽ được đưa vào để phân tích và trích xuất ra các trường thông tin cần thiết như: IP nguồn, IP đích, Port Nguồn, Port đích, Protocol (được gọi là 5 tuple) và các thông tin của gói tin như timestamp, bytes counter và packets counter.

- Khi trích xuất được 5 tuple và các info packet thì sẽ được đưa đến module Create/Update, tại module Hashing sẽ tiến hành hash 5 tuple mới nhận được thành 12-bit địa chỉ làm địa chỉ của dữ liệu (5 tuple & info packet) ở bộ lưu trữ BRAM. Dữ liệu khi đưa xuống đây nó sẽ được tạo mới hoặc update trong BRAM. Tại đây khi phát hiện gói tin TCP có chứa cờ FIN hay RST thì sẽ phát thông báo kèm địa chỉ của gói tin đó đến module Export Expired để export.

- BRAM: Bộ nhớ lưu trữ 5 tupe & thông tin của gói tin theo tổ chức bộ nhớ.

- Module Export Expired, module này sẽ thực hiện chức năng export các luồng dữ liệu bên trong BRAM đã hết hạn, được tính bởi thời gian timeout (Active timeout & InActive timout) và gói tin chứa flags TCP (FIN, RST). Các luồng dữ liệu được export ra sẽ lưu vào các thanh ghi tại module Export Record để chờ xử lý.

- Module Export Record: Module này chứa các thanh ghi sẽ lưu trữ các luồng dữ liệu đó và sẽ được đọc ra theo Bus Axi4-Lite bởi MicroBlaze và hiển thị lên Monitor của người giám sát mạng theo đường UART.

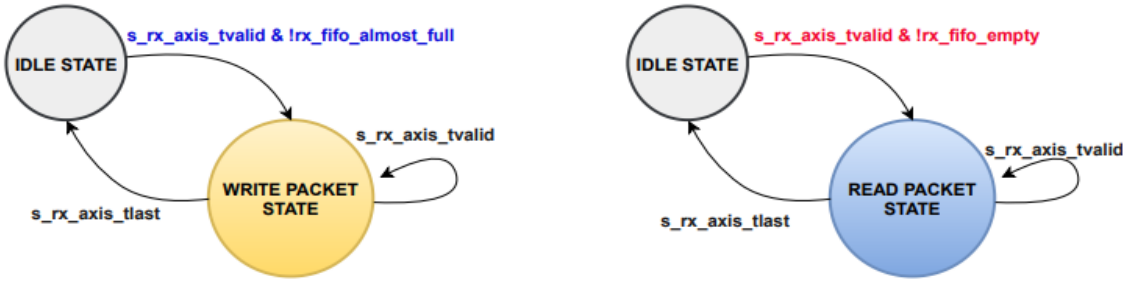
3.3 Mô tả thiết kế các khối chức năng trong hệ thống Core NetFlow V5

3.3.1 Module Rx Queues

3.3.1.1 Mô tả chức năng của module

Module Rx Queues gồm có 4 module tương ứng là hàng đợi nhận các dữ liệu đầu vào tại 4 Port. Với 32-bit data đầu vào lưu trữ tại FIFO Rx queues và lấy ra 64 bit data tại output FIFO.

Sơ đồ trạng thái Write/Read FIFO:



Hình 3.3 Trạng thái quá trình đọc/ghi FiFO tại module Rx Queues

3.3.1.2 Các tín hiệu vào ra của module

Bảng 3.1 Các tín hiệu vào ra của module Rx Queues

Tín hiệu	Độ rộng (bit)	Input/Output	Mô tả chức năng
s_rx_axis_tdata	32	Input	Giao diện AXI_Stream nhận gói tin từ RX từ Tri-Emac [1]
s_rx_axis_tkeep	4	Input	
s_rx_axis_tlast	1	Input	
s_rx_axis_tvalid	1	Input	
s_rx_axis_tready	1	Output	
m_tx_axis_tready	1	Input	Giao diện AXI_Stream Interface gửi gói tin từ RX đến Tri-Emac [1]
m_tx_axis_tdata	32	Output	
m_tx_axis_tkeep	4	Output	
m_tx_axis_tlast	1	Output	
m_tx_axis_tvalid	1	Output	
axis_aclk	1	Input	
reset	1	Input	Reset hệ thống
clk	1	Input	Clock hệ thống
Out_data	64	Output	Dữ liệu đầu ra

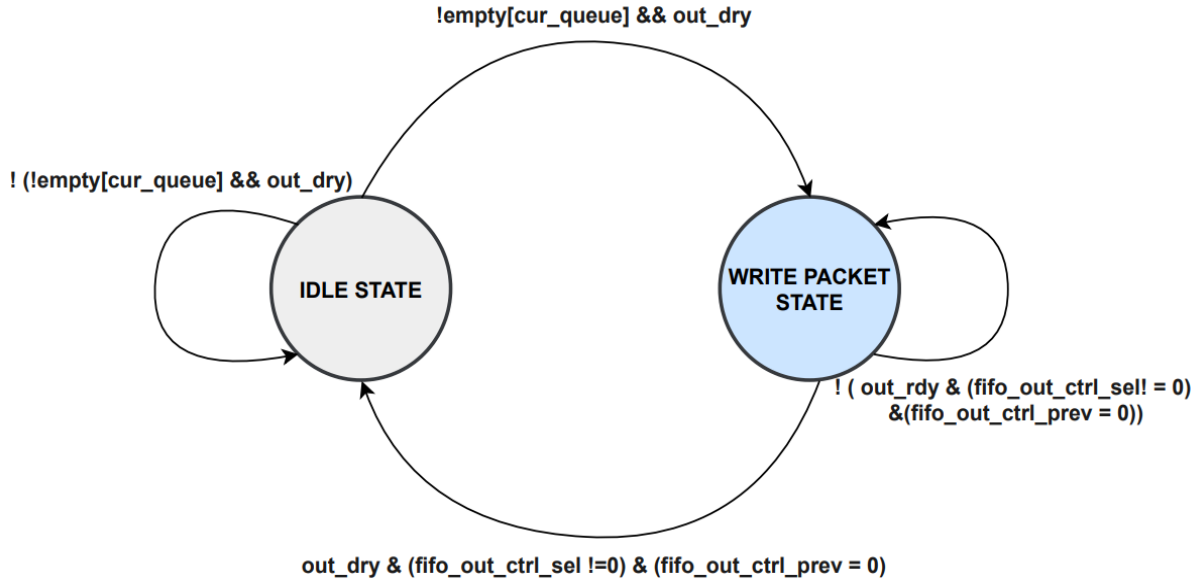
3.3.1.3Module Input Arbiter

Mô tả chức năng của module

Module Input Arbiter sẽ nhận gói lấy từ 1 trong 4 output data của 4 module Rx Queues và lưu trữ vào tại Fifo chứa bên trong module. Sau đó sẽ theo cơ chế Round Robin lấy ra theo thứ tự Port input vào để đưa dữ liệu sang khối tiếp theo để xử lý.

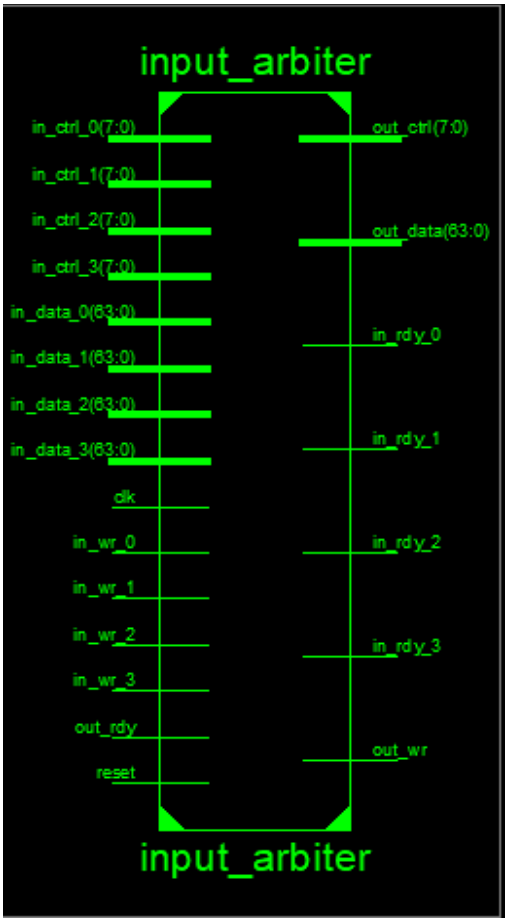
* Thuật toán chọn gói tin của Input Arbiter:

```
For i from 1 to 8
  begin
     If ( FIFO no. i is not empty)
        Read 1 packet at FIFO no. i
     i = i + 1;
  end
```



Hình 3.4 Máy trạng thái của của module Input Arrbiter

3.3.1.4 Kiến trúc của module Input Arbiter



Hình 3.5 Interface của module Input Arbiter

*Mô tả các tín hiệu:

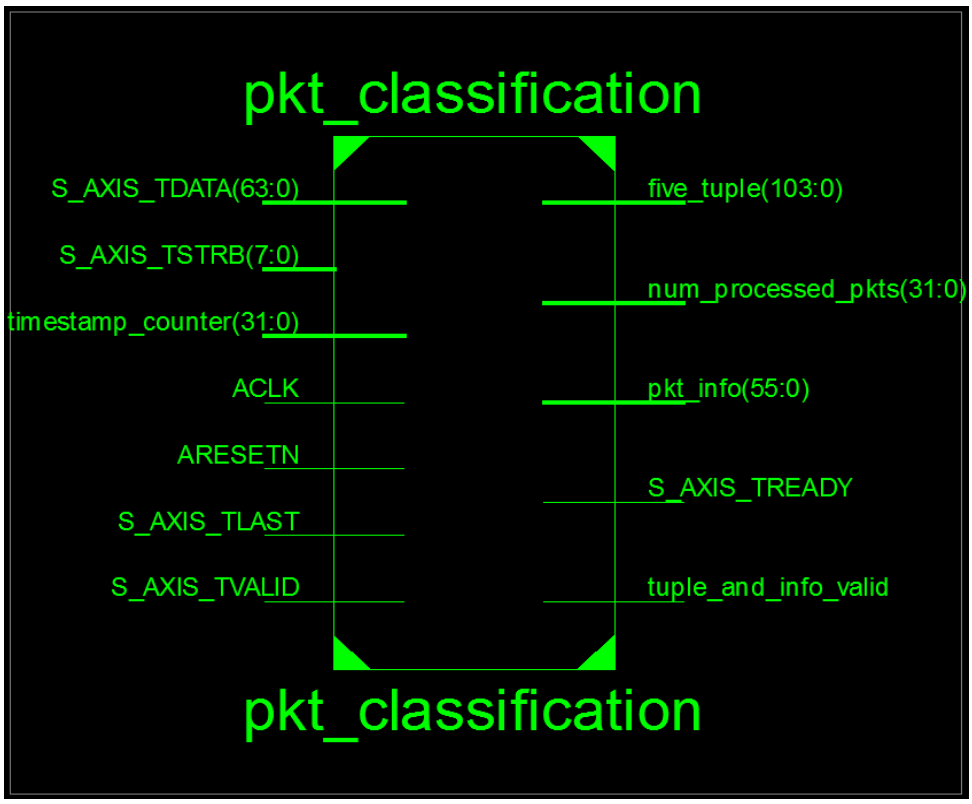
Bảng 3.2 Các tín hiệu vào ra của module Input_Arbiter

Tín hiệu	Input/Output	Độ rộng (bit)	Mô tả chức năng
out_data	Output	64	Dữ liệu đầu ra
in_data_i*	Input	64	Dữ liệu của gói tin vào từ 1 trong 4 port từ module Rx queues (*) 0<= i <= 3
in_wr_i*	Input	1	
in_rdy_i*	Output	1	Báo hiệu module packet classification sẵn sàng nhận gói
reset	Input	1	Reset hệ thống
clk	Input	1	Clock hệ thống

3.3.2 Module Packet Classification

3.3.2.1Mô tả chức năng

Module Packet Classification sẽ nhận gói tin từ output của module Input Arbiter sau đó tiến hành trích xuất các trường thông tin tại header của gói tin gồm 5 tuple (IP nguồn, IP đích, Port Nguồn, Port Đích, Protocol) và các thông tin gói tin như: timestamp, bytes counter, packets counter.



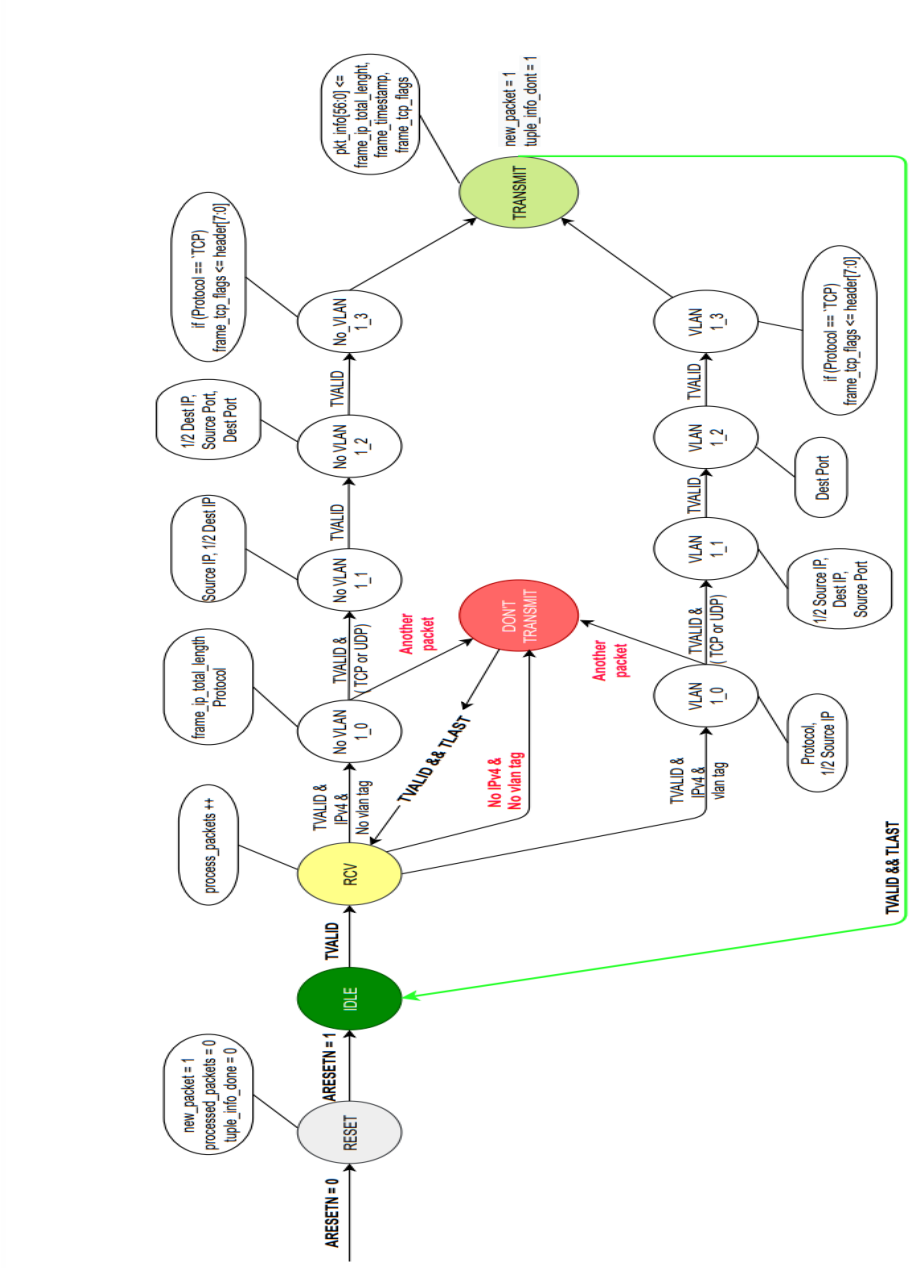
Hình 3.6 Kiến trúc của module packet classification

Mô tả các tín hiệu

Bảng 3.3 Các tín hiệu trong module packet classification

Tín hiệu	Input/Output	Độ rộng (bit)	Mô tả chức năng
S_AXIS_TDATA	Input	64	Dữ liệu đầu vào
TIMESTAM_COUNTER	Input	32	Thời gian thực
ACLK	Input	1	Clock hệ thống
ARESETN	Input	1	Reset hệ thống
S_AXIS_TLAST	Input	1	Thông báo phần còn lại của gói tin.
AXIS_TVALID	Input	1	Thông báo hợp lệ của gói tin đưa vào
Five_tuple	Output	104	Lưu trữ dữ liệu của 5 tuple
Num_process_pkts	Output	32	Báo hiệu có gói tin đang xử lý [1]
Pkt_info	Output	56	Lưu trữ dữ liệu các thông tin của gói tin
S_AXIS_TREADY	Output	1	Báo hiệu sẵn sàng nhận gói tin phía Slave
	Output	1	Thông báo rằng 5 tuple & thông tin gói đã trích xuất xong.

3.3.2.2Mô tả FSM của module



Hình 3.7 Finite State Machine của module packet classification

Mô tả luồng dữ liệu theo FSM của module Classification:

- Khi tín hiệu Reset = 1 và có TVALID báo hiệu có gói tin đi vào, tại trạng thái RCV nó sẽ kiểm tra xem gói tin đi vào hoạt động theo giao thức internet phiên bản 4 (IPv4) và có tag Vlan hay không tag Vlan
- Những gói tin nằm ngoài TCP, UDP, IPv4 sẽ không xử lý và bị loại bỏ.
- Còn lại các gói tin hợp lệ sẽ được module này xử lý, trích xuất các trường thông tin gồm 5 tuple (Port Nguồn, Port Đích, IP nguồn, IP đích, Protocol) và các thông tin

của gói: timestamp, bytes counter, packets counter theo các trạng thái tiếp theo trong FSM.

- Sau khi trích xuất xong thì dữ liệu sẽ được gửi đến module tiếp theo để xử lý.

3.3.3 Module Create or Update

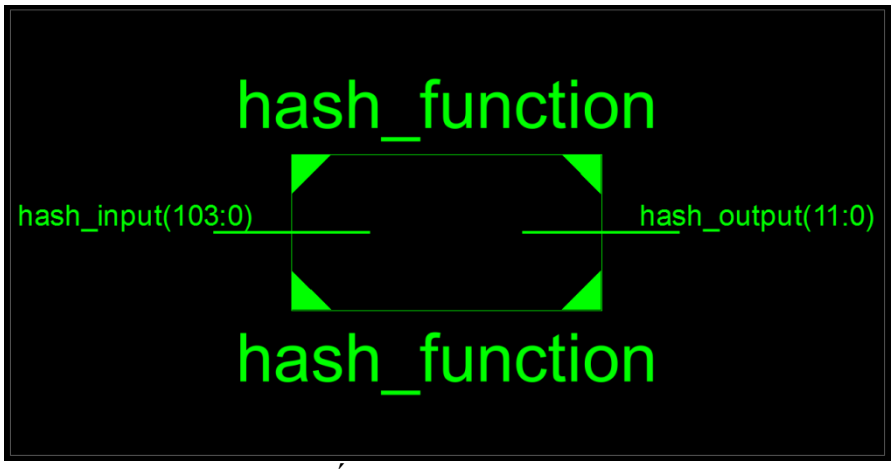
3.3.3.1 Mô tả chức năng

Trong module Create or Update sẽ chứa một module Hashing, module hashing có chức năng tạo một địa chỉ 12 bits. Với mỗi gói tin đi vào trong IP core sẽ có một " Five Tuple " (gồm IP Source, IP Destination, Port Source, Port Destination, Protocol). Module Hashing sử dụng một hàm băm để tạo ra một địa chỉ 12 bits từ 104 bits của " five tuple ". Địa chỉ này là địa chỉ của flow bên trong BRAM.

Module Create or Update, sẽ dựa vào địa chỉ được tạo ra từ module hashing để kiểm tra flow chứa trong Bram tại địa chỉ đó có tồn tại hay không để Update hay Create vào Bram tại địa chỉ đó. Nếu phát hiện gói tin chứa cờ FIN hay RST thì module này sẽ gửi thông báo kèm địa chỉ sang module Export Expired để tiếp tục xử lý.

3.3.3.2 Kiến trúc của module Create or Update

- Module Hashing:
  Khối này nhận được 5 tuple từ khối Packet Classification , sau đó sử dụng hashing function để băm 5 tuple(104 bits_ width) lấy hash code 14bits(có thể cấu hình khác). Hashing nằm trong khối create/update module để phục vụ cho việc truy cập vào các flow trong BRAM bằng địa chỉ hashcode này.



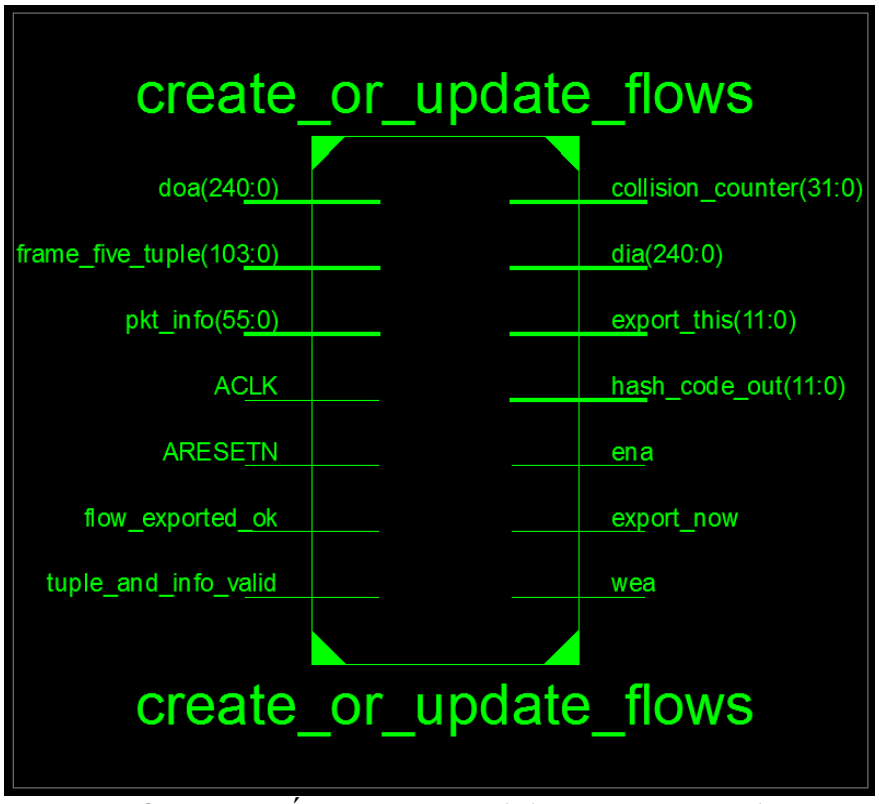
Hình 3.8 Kiến trúc của module Hashing

Mô tả tín hiệu:

Bảng 3.4 Tín hiệu vào ra của module Hashing

Tín hiệu	Input/Output	Độ rộng (bit)	Mô tả chức năng
Hash_input	Input	104	Dữ liệu đầu vào là 104-bit của 5 tuple
Hash_output	Output	12	Địa chỉ sau khi hash từ 104-bit dữ liệu

Module Create or Update: Tạo/cập nhật dữ liệu (gói tin) vào BRAM.



Hình 3.9 Kiến trúc của module Create or Update

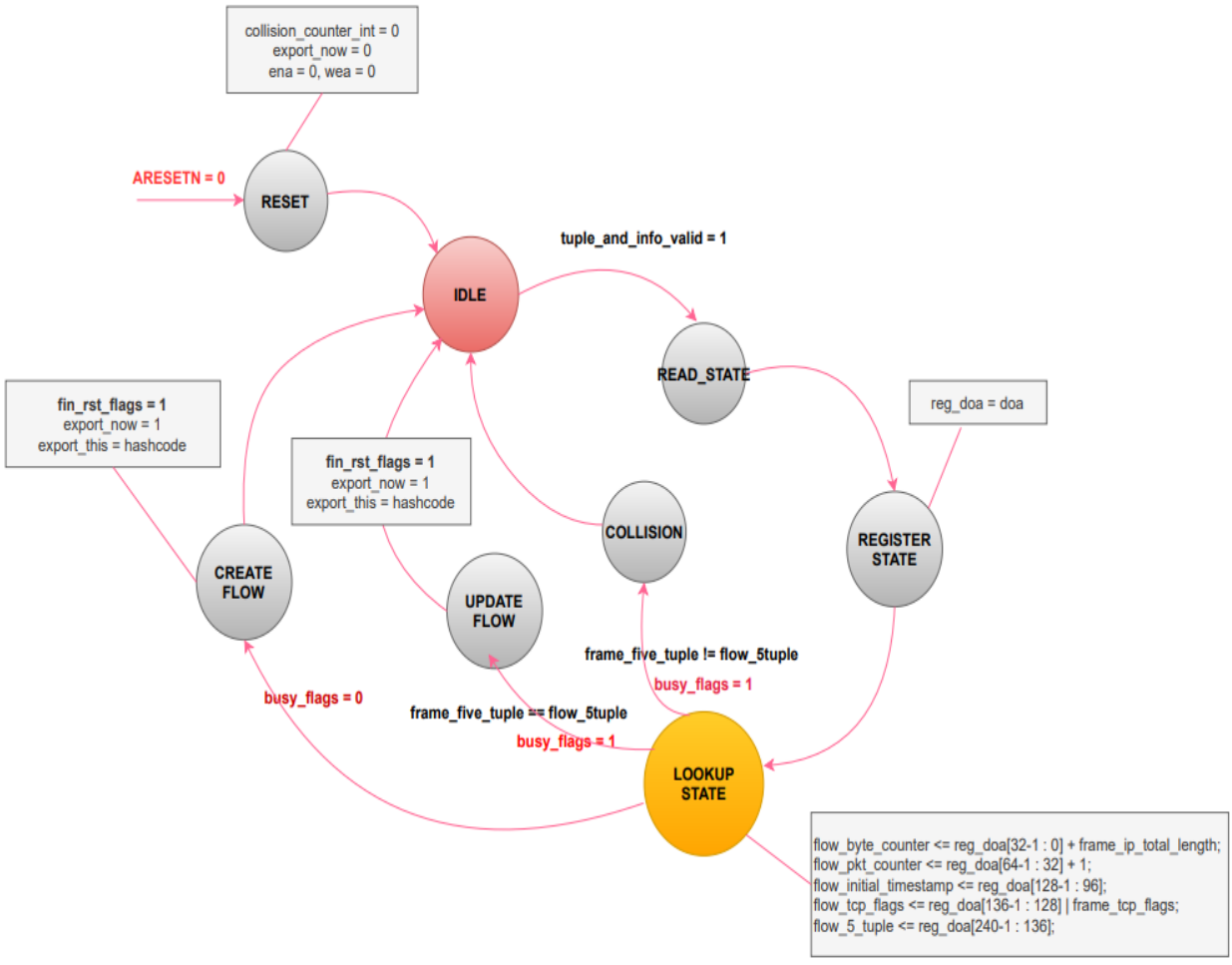
Mô tả các tín hiệu:

Bảng 3.5 Các tín hiệu vào ra của module Create or Update

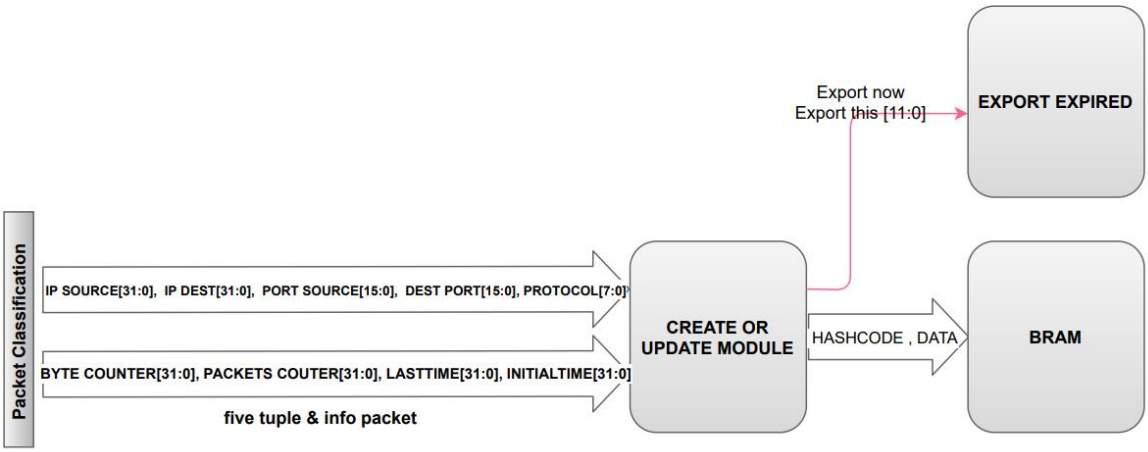
Tín hiệu	Input/Output	Độ rộng (bit)	Mô tả chức năng
doa	Input	241	Tín hiệu rộng 241-bit để chứa dữ liệu là 1 flow entry khi lấy ra từ BRAM
Frame_five_tuple	Input	104	Lưu trữ dữ liệu là 5 tuple
Pkt_info	Input	56	Lưu dữ liệu thông tin gói tin
ACLK	Input	1	Clock hệ thống
ARESETN	Input	1	Reset hệ thống
Flow_export_ok	Input	1	Tín hiệu trả về từ module Export Expired
Tuple_and_info_valid	Input	1	Thông báo 5 tuple và thông tin gói tin là hợp lệ

Collision_counter	Output	32	Bộ đếm số lượng các gói tin bị xung đột
Dia	Output	241	Dữ liệu đầu vào
Export_this	Output	12	Lưu địa chỉ hashcode
Ena	Output	1	Cho phép truy cập Cổng A của BRAM
Export_now	Output	1	Tín hiệu truyền đến module Export Expired báo hiệu cần export flow
Wea	Output	1	Cho phép Read/write vào Port A.

3.3.3.3Mô tả FSM của module Create or Update



Hình 3.10 Máy trạng thảo của module Create or Update



Hình 3.11 Sơ đồ luồng dữ liệu của module Create Update

Mô tả luồng dữ liệu FSM của module Create or Update:

- Khi các 5 tuple (Port Nguồn, Port đích, IP nguồn, IP đích, protocol) và các info packet đã phân tích xong sau tại module Classification thì module này sẽ tiến hành lưu các dữ liệu đó vào biến tạm tại State READ STATE, sau đó dùng hashcode để truy cập vào BRAM để kiểm tra luồng đang tồn tại flow hay không tại State: FLOW LOOKUP, nếu chưa tồn tại Flow thì busy flags = 0 sẽ chuyển sang State Create. Ngược lại nếu đã tồn tại có busy flags = 1 sẽ tiếp tục kiểm tra 5 tuple tại flow đã chứa trong Bram với 5 tuple của gói tin mới lưu ở module Create or Update; Nếu giống nhau sẽ chuyển đến trạng thái Update; Nếu khác sẽ chuyển đến trạng thái Collisison.

- Tại State Create: Tại địa chỉ hashing thì nó sẽ tạo mới 1 flow mới bao gồm 240 bit (5tuple và pkt info; đồng thời sẽ kiểm tra xem gói tin TCP có chứa cờ FIN hay RST hay không, nếu có thì nó sẽ phát thông báo và gửi hashcode của gói tin mới đến module Export Expired để xử lý.

- Tại State Update: Tại địa chỉ hashing thì nó sẽ update các thông tin: lasttimestamp, TCP flags, bytes counter, packets counter; đồng thời nó cũng kiểm tra xem gói tin có chứa FIN hay RST hay không, nếu có sẽ phát tín hiệu và địa chỉ tương tự như tại State Create.

- Tại State Collission: Luồng dữ liệu mới sẽ được xử lý và không được Update hay Create.

- Sau khi thực hiện các quá trình thì trạng thái sẽ trở về ban đầu để xử lý gói tin tiếp theo.

3.3.4 BSRAM

3.3.4.1Mô tả chức năng:

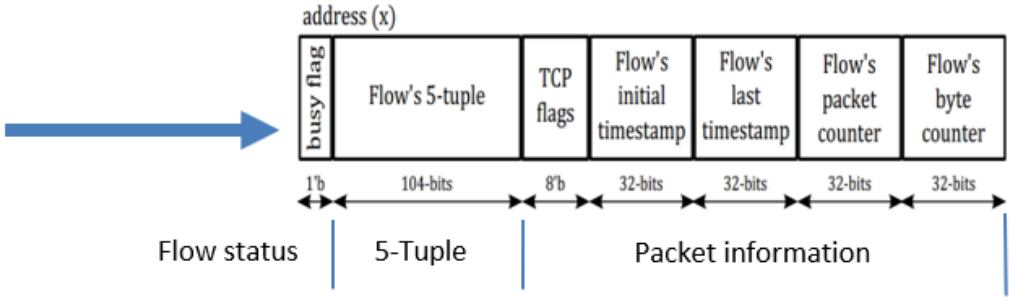
- Là bộ nhớ cache dùng để lưu trữ các flow, gồm có 2 cổng A & B hỗ trợ các hoạt động đọc, ghi đồng thời & hoạt động độc lập với nhau.

- Khối này thực hiện 2 quá trình song song và độc lập nhau đó là tiếp nhận, ghi flow vào flow table thông qua port A và đọc flow từ flow table thông qua port B.

* Thông số:

Độ rộng địa chỉ bit (Width): 241-bit để lưu trữ data gồm flags status, 5 tuple, packet info.

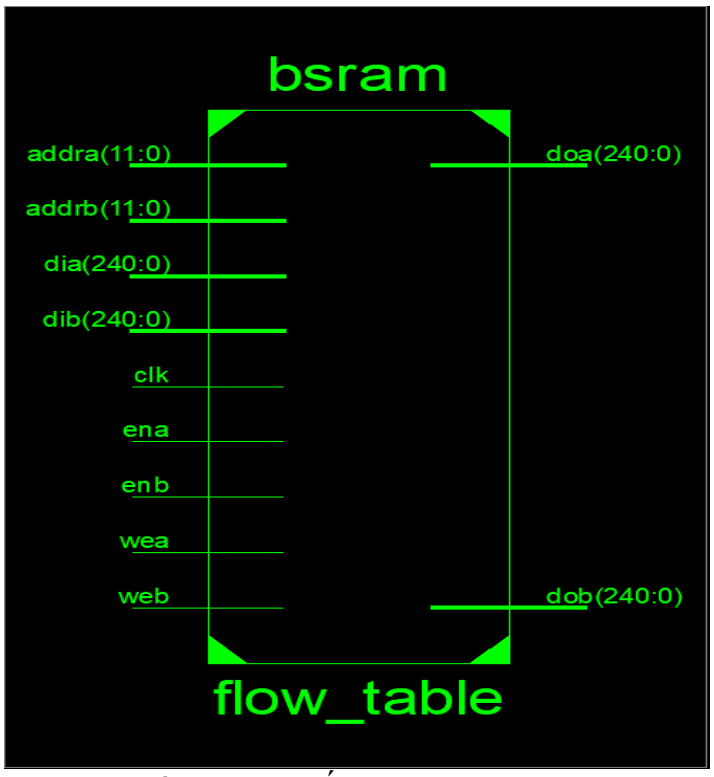
Tổ chức bộ nhớ với address 12-bit, có thể lưu trữ được 2^{12} flow entry.



Hình 3.12 Tổ chức bộ nhớ trong BRAM

Module Create or Update sẽ tạo mới hay update flow vào trong BSram, với tổ chức bộ nhớ, Width 241-bit bao gồm: busy flags (bit trạng thái của flow), 5 tuple & packet info.

3.3.4.2 Kiến trúc BRAM



Hình 3.13 Kiến trúc của BRAM

Mô tả tín hiệu:

Bảng 3.6 Các tín hiệu vào ra của module Bram

Signal	Input/Output	Width (bit)	Decription
Addra	Input	12	Ghi địa chỉ theo cổng A
Addrb	Input	12	Ghi địa chỉ theo cổng B
Dia	Input	241	
Dib	Input	241	Ghi dữ liệu vào cổng B
Clk	Input	1	Clock hệ thống
Ena	input	1	Cho phép truy cập cổng A (mức cao)
Enb	Input	1	Cho phép truy cập cổng B (mức cao)
Wea	Input	1	Read/Write cổng A
Web	Input		Read/Write cổng B
Doa	Output	241	Xuất dữ liệu từ địa chỉ cổng A
Dob	Output	241	Xuất dữ liệu từ địa chỉ cổng B

3.3.5 Module Export Expired

3.3.5.1Mô tả:

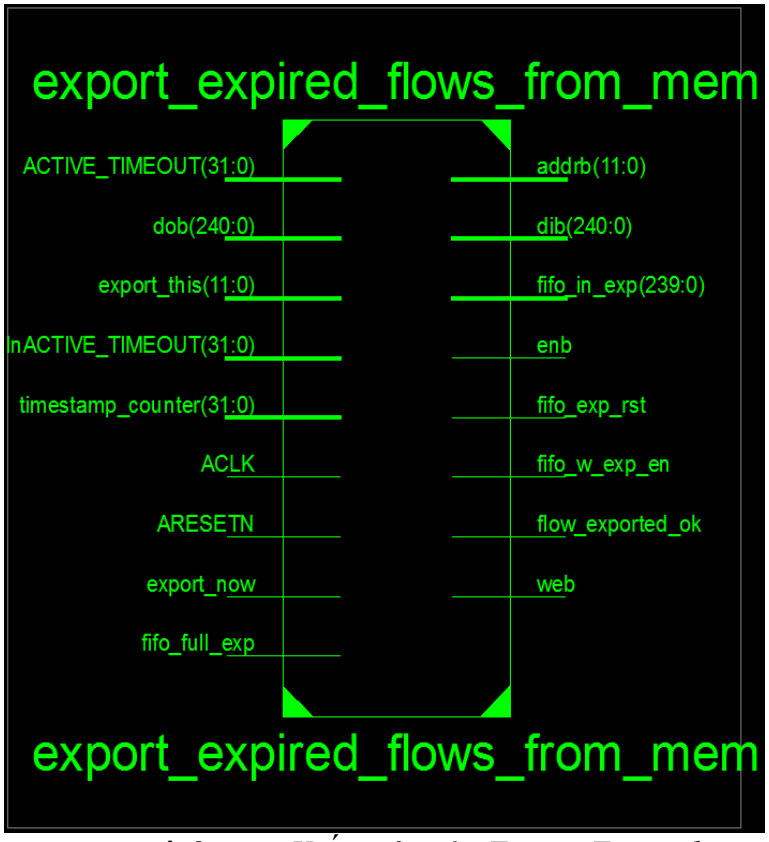
Kiểm tra thời gian hết hạn của Flow được lưu tại từng địa chỉ lưu trong BRAM. Bằng cách, đọc ra lần lượt các Flow tại từng địa chỉ trong BRAM để kiểm tra. Nếu Flow tại địa chỉ đang xét hết hạn thì Module Export Expired sẽ xóa Flow đó khỏi BRAM, Export đến FIFO (xóa trạng thái cờ Busy và viết các bit 0 vào Flow ở địa chỉ này). Nếu Flow tại địa chỉ đang xét chưa hết hạn thì tiếp tục kiểm tra địa chỉ tiếp theo.

Điều kiện hết hạn của 1 flow entry nằm trong BRAM:

Current Time – Last TimeStamp > Inactive TimeOut (mặc định 15s).

Current Time – Initial TimeStamp> Active TimeOut (mặc định 30ph).

3.3.5.2 Kiến trúc module Export Expired



Hình 3.14 Kiến trúc của Export Expired

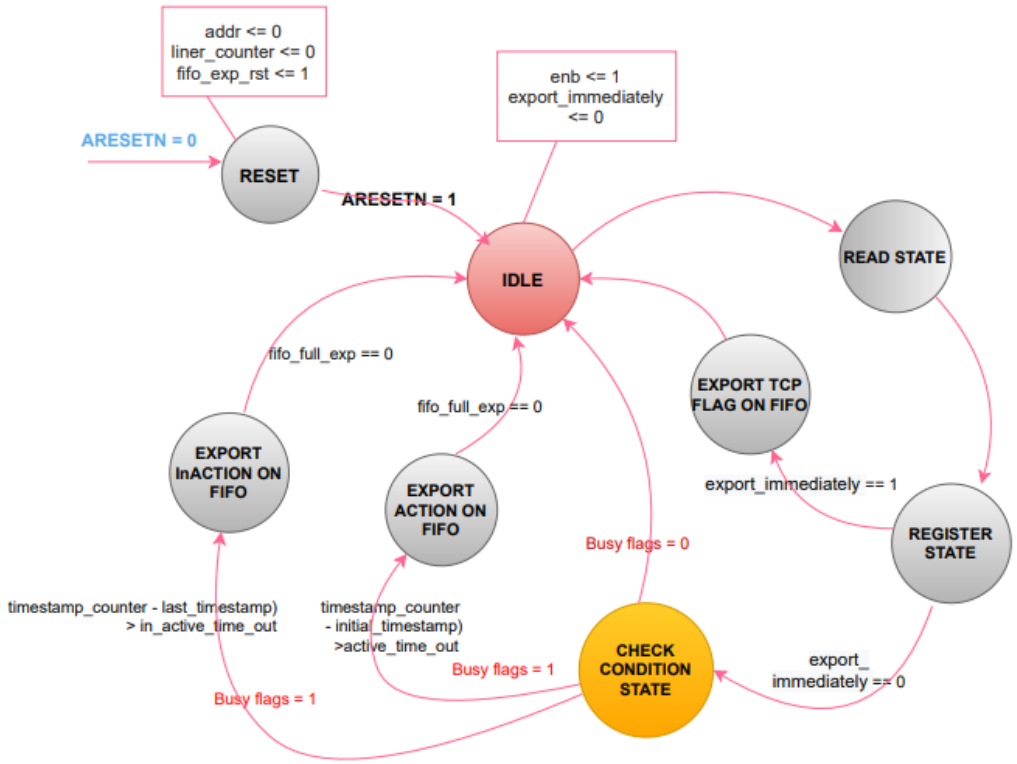
Mô tả các tín hiệu:

Bảng 3.7 Các tín hiệu vào ra của module Export Expired

Tín hiệu	Input/Output	Độ rộng (bit)	Mô tả chức năng
ACTIVE_TIME_OUT	Input	32	Ghi giá trị thời gian active để phục vụ cho tính toán flow expired
Dob	Input	241	Ghi dữ liệu từ BSRAM (flow)

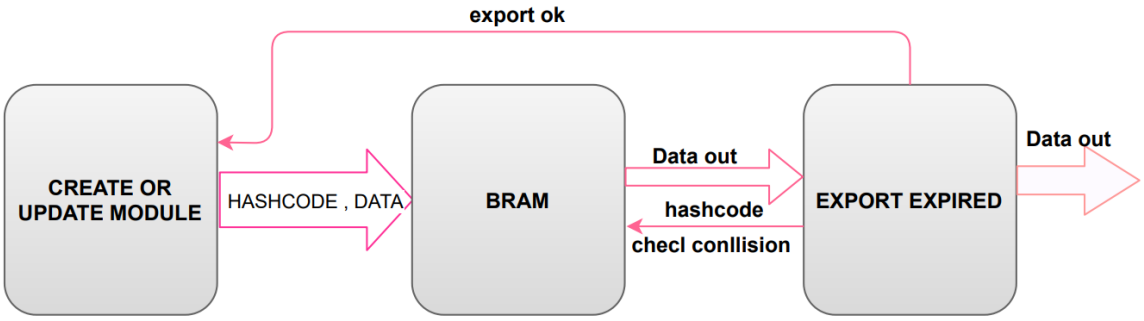
Export_this	Input	12	Ghi địa chỉ của flow chứa cờ Fin/rst từ module Create/Update
InACTIVE_TIMEOUT	Input	32	Ghi giá trị thời gian inactive để phục vụ cho tính toán flow expired
CLK	Input	1	Clock hệ thống
ARESETN	Input	1	Reset hệ thống
Export_now	Input	1	Tín hiệu thông báo có gói tin chứa cờ Fin/rst cần rexport từ module Create Update
Fifo_full_exp	Input	1	Tín hiệu thông báo fifo đầy
Addrb	Output	32	Ghi địa chỉ tại cổng B
dib	Output	241	Ghi dữ liệu vào ra tại cổng B
Flow_in_exp	Output	240	Ghi dữ liệu được export ra khi flow expired
Enb	Output	1	Tín hiệu cho phép truy cập cổng B (mức cao)
Fifo_exp_rst	Output	1	Tín hiệu rst fifo
Fifo_w_exp_en	Output	1	Cho phép ghi vào fifo
Export_exported_ok	Output	1	Tín hiệu đưa về module Create/Update báo hiệu đã export gói chứa cờ Flash xong.
Web	Output	1	Read/Write cổng B (0 | 1)

3.3.5.3 Mô tả FSM của module Create or Update



Hình 3.15 Máy trạng thái của module Export Expired

Mô tả luồng dữ liệu:



Hình 3.16 Sơ đồ khối mô tả luồng dữ liệu của module Export Expired

- Khối Export Expired (Timestamp counter moniter) giám sát điều kiện thời gian để giải quyết flow table sử dụng bộ đếm đánh dấu thời gian và xác định flow expired bị loại khỏi flow table bằng cách sử dụng bộ đếm tuyến tính vào từng địa chỉ của mỗi entry trong flow table từ 0,1…n để kiểm tra busy flag.

+ Nếu busy flag = 0 (không có flow) thì sẽ bỏ qua và kiểm tra địa chỉ tiếp theo
+ Nếu busy flag = 1(có flow) thì sẽ kiểm tra 2 hoạt động song song sau:
    - Kiểm tra thời gian trôi qua kể từ khi last packet đến (current time counter – flow's last timestamp) > 15s (inactive timeout)
    - Kiểm tra thời gian trôi qua kể từ khi flow record được tạo (current time counter – flow's intital timestamp) >30 phút (active flow) , điều kiện này để tránh tràn bộ đếm .
    - ⇨ Nếu một trong 2 điều kiện này thỏa thì sẽ export flow đến Export Module và loại khỏi flow table.

- Trong lúc bộ đếm tuyến tính đang kiểm tra nếu có tín hiệu từ khối create/update flow thông báo TCP new packet có RST flag hoặc Fin flag yêu cầu export flow này. Lúc này hệ thống vẫn hoàn thành xong việc kiểm tra sau đó sẽ thực hiện yêu cầu từ khối create /update flow (tối ưu hơn) hoặc sẽ đi thực hiện yêu cầu ngay lập tức.
    - ⇨ Các điều kiện trên chính là điều kiện để 1flow bị export (active/inactive và Fin flag, RST flag)
- Các cách để xóa flow trong flow table:
  + Đến flow entry đó cho busy flag về mức 0, cách này không tối ưu bởi vì sẽ xảy ra các sự cố nhầm bit, nhầm địa chỉ.
  + Xóa địa chỉ của flow entry về "00..00"
- Khối **Export Module** sẽ nhận các flow bị loại khỏi flow table và export chúng ra khỏi flow cache core.

3.3.6 Thống kê FIFOs

3.3.6.1Kích thước của FIFO

Bảng 3.8 Thống kê FIFOs sử dụng trong IP Core

Module	Tên FIFO	Width (bit)	Depth (bit)
Rx_queue	gmac_rx_fifo	Input: 32 Output: 64	2K
Input_arbiter	in_arb_fifo	64	16
	FIFO_SYNC_MACRO	Input: 72 Output: 72	36K

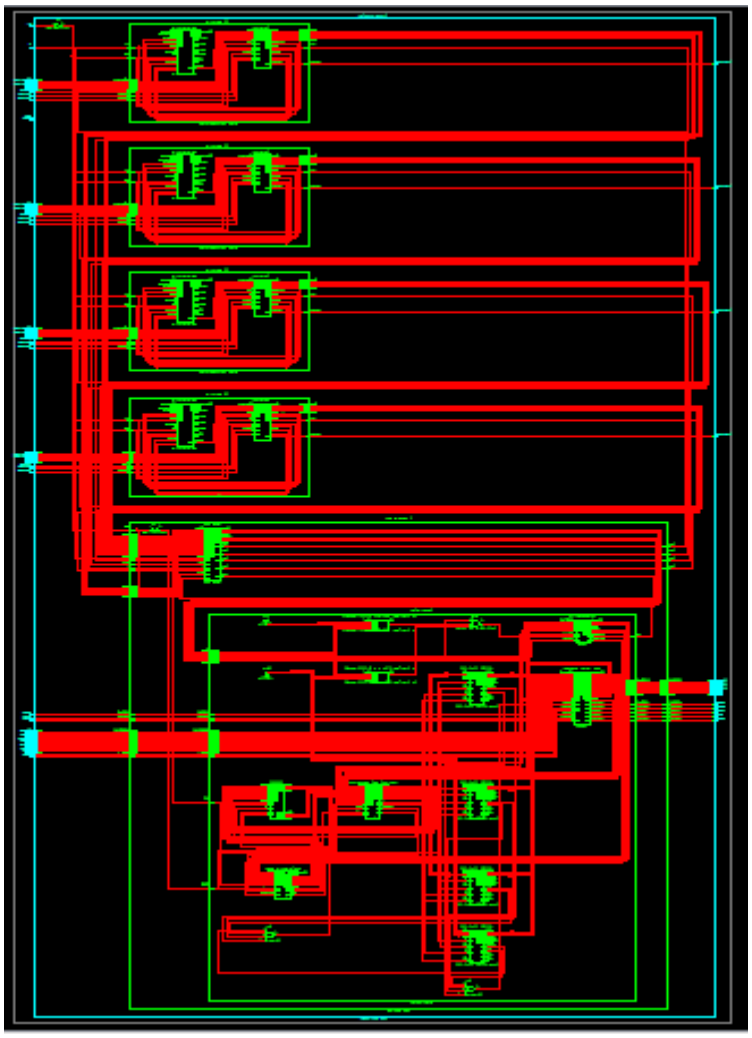
Mô tả chức năng:

- **gmac_rx_fifo**: nhận dữ liệu là các gói tin đầu vào theo 32bit Input và lấy ra 64 bit cung cấp cho module tiếp theo xử lý.

- **in_arb_fifo** : nhận và lưu trữ dữ liệu từ module Rx_queues đẩy sang với 4 port, nó sẽ lưu và đẩy ra theo cơ chế robin.

- **FIFO_SYNC_MACRO**: bao gồm có 4 FIFO, mỗi fifo với đầu vào và đầu ra gồm 72 bit, fifo sẽ lưu trữ các flow bị export ra ngoài BRAM từ module Export Expired để người dùng xử lý sau này.

3.3.3: Sơ đồ tổng hợp hệ thống sử dụng phần mềm ISE 14.7



Hình 3.17 Kiến trúc toàn bộ IP Core NetFlow V5 trên phần mềm ISE

3.4: Mô phỏng bằng ISIM sử dụng phần mềm ISE 14.7

Phân tích Testbench đối với gói tin có chứa VLAN và không chứa VLAN ở khối Packet Classification:

Khi gói tin được đưa vào module phân tích gói để tiến hành phân tích lấy ra 5-tuple và các info packet. Ở 12-byte đầu của gói tin, nó sẽ chứa địa chỉ MAC đích và MAC nguồn, 4-byte tiếp theo để xác định được gói tin này có chứa VLAN hay không.

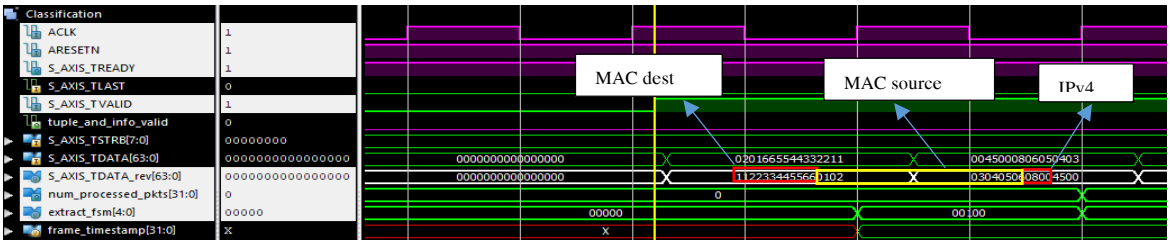
Ở 2-byte đầu trong 4-byte tiếp theo sau địa chỉ MAC, kiểm tra rằng gói tin có chứa VLAN hay không

- Nếu 2byte đầu có giá trị:
  ⇨ 0x8100, cho biết gói tin đưa vào chứa VLAN single tag.

⇨ 0x88a8, cho biết gói tin đưa vào chứa VLAN douple tag.
⇨ 0x0800, cho biết gói tin đưa vào không chứa VLAN.

- 2-byte còn lại chứa VID.

- **Trường hợp gói tin không chứa VLAN:**



Hình 3.18 Testbench no VLAN 1

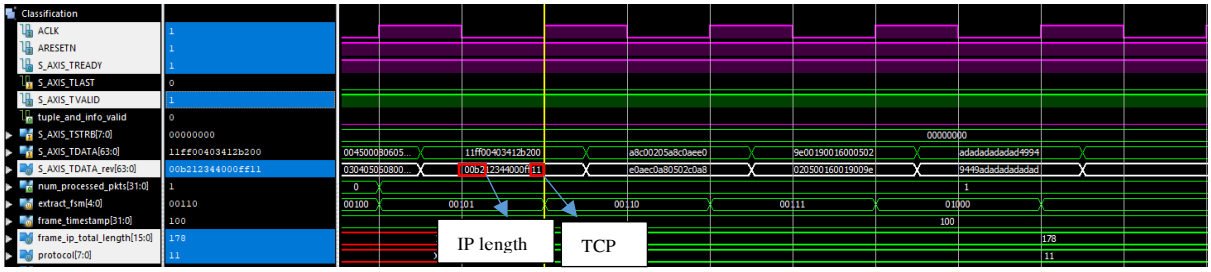
Ở trạng thái đầu: 0000 (idle), 00100 (rcv1):

Khi có tín hiệu của clk và reset = 1, tín hiệu S_AXIS_TVALID = 1 báo rằng có gói tin hợp lệ đưa vào.

Gói tin bắt đầu đưa vào: Ở 6byte đầu chứa MAC dest: 112233445566, 6byte tiếp thep chứa MAC source: 010203040506 và 2-byte tiếp theo: "0800" và 4bit có giá trị "4" cho biết gói tin đưa vào là IPv4 và không chứa VLAN.

Sau khi kiểm tra gói tin không chứa VLAN, chuyển sang các trạng thái tiếp theo để thực hiện phân tích và lấy ra 5-tuple gồm: Source IP, Dest IP, Port Source, Port Dest, protocol và packet info gồm: tcp flags, initial_timestamp, last_timestamp, packet_counter và byte counter. (Trong đó initial_timestamp được lấy từ lúc gói tin được đưa vào).

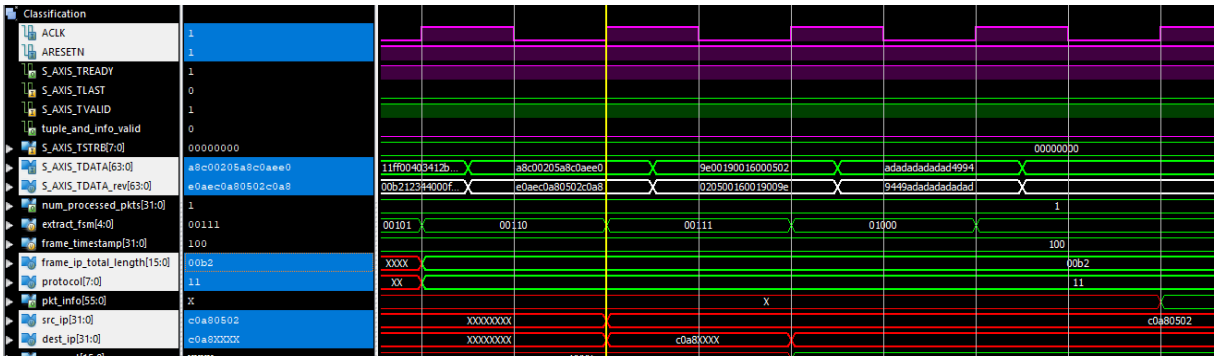
Ở trạng thái 00101 (rvc1_ipv4_no_vlan0):



Hình 3.19 Testbench no VLAN 2

Tiến hành phân tích lấy: protocol[7:0] = 6 (giao thức TCP) hoặc protocol[7:0] =1 (giao thức UDP) IP_total_length = 178 (0x00b2).

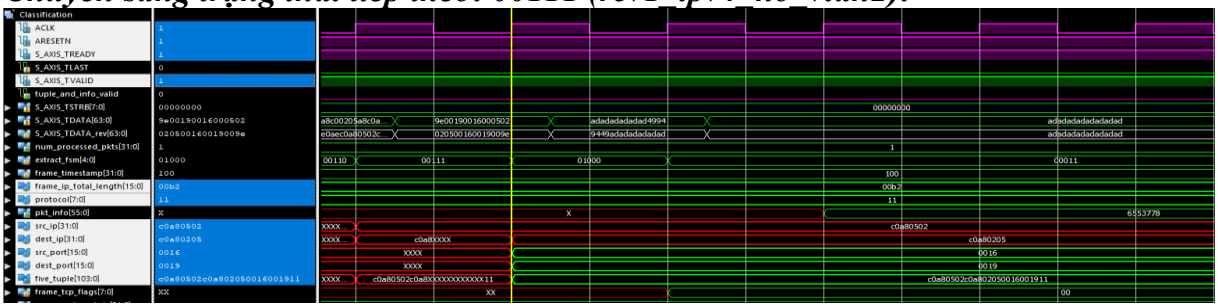
Chuyển sang trạng thái: 00110(rcv1_ipv4_no_vlan1):



Hình 3.20 Testbench no VLAN 3

Tiến hành phân tích và lấy: Source IP: Src_ip[31:0] =0xc0a80502 và ½ Dest IP: Dest_ip[31:0] = 0xc0a8.

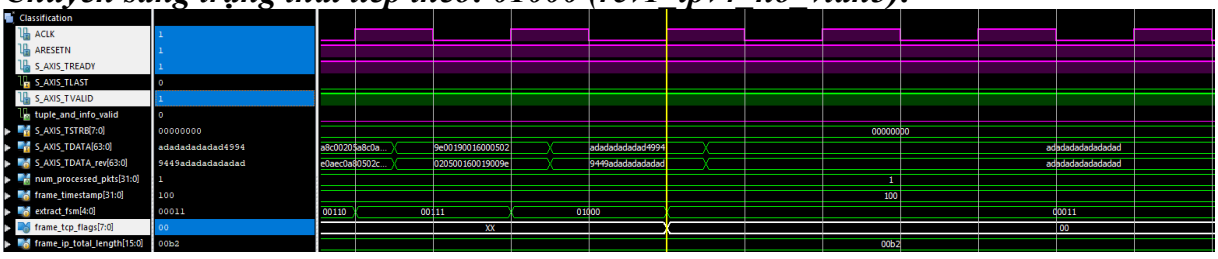
Chuyển sang trạng thái tiếp theo: 00111 (rcv1_ipv4_no_vlan2):



Hình 3.21 Testbench no VLAN 4

Tiến hành lấy: ½ Dest IP còn lại: để có Dest IP: dest_ip [31:0] = 0xc0a80205, Source Port: src_port[15:0] = 0x0016 và Dest Port: dest_Port[15:0] = 0x0019.

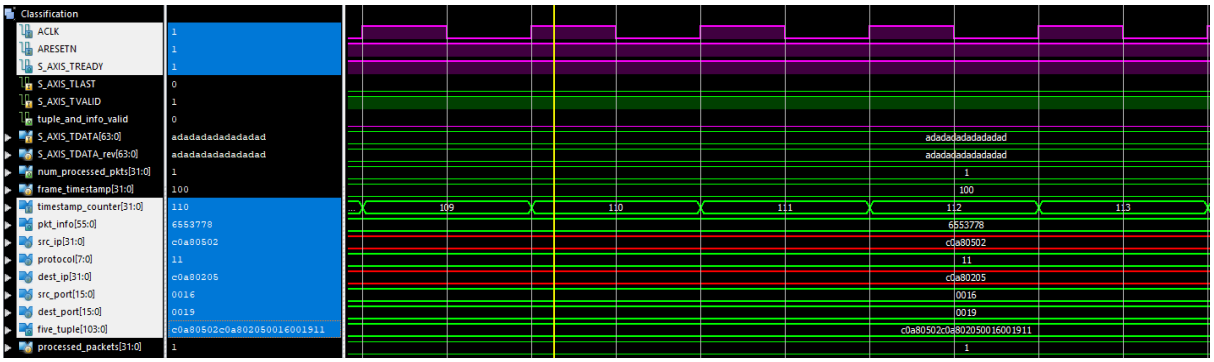
Chuyển sang trạng thái tiếp theo: 01000 (rcv1_ipv4_no_vlan3):



Hình 3.22 Testbench no VLAN 5

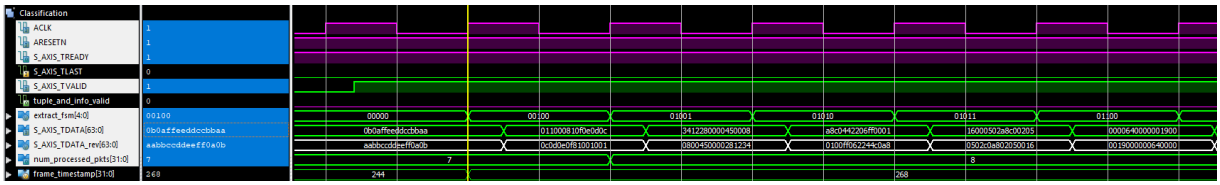
Ở trạng thái này, kiểm tra xem gói tin có chứa TCP flags (FIN/RST). Nếu không chứa TCP flags thì set fram_tcp_flags = 0.

Sau quá trình phân tích gói tin trên, ta lấy ra 5-tuple + packet info và chuyển sang module create/Update.



Hình 3.23 Đóng gói five tuple và các thông tin

- **Trường hợp gói tin chứa VLAN tag:**

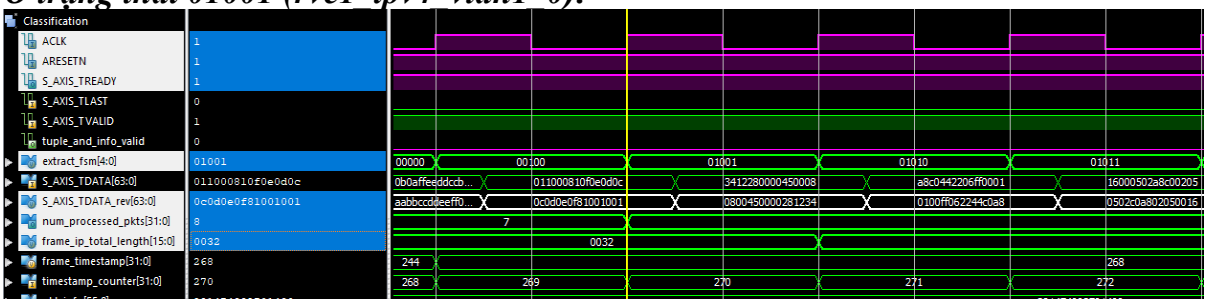


Hình 3.24 Testbench VlanTag 1

Khi có tín hiệu của clk và reset = 1, tín hiệu S_AXIS_TVALID = 1 báo rằng có gói tin hợp lệ đưa vào. Gói tin bắt đầu đưa vào: 6 byte đầu chứa MAC dest: aabbccddeeff, 6 byte tiếp thep chứa MAC source: 0x0a0b0c0d0e0f và 2 byte tiếp theo: "8100" cho biết gói tin chứa VLAN tag.

Sau khi kiểm tra gói tin chứa VLAN tag, chuyển sang các trạng thái tiếp theo để thực hiện phân tích và lấy ra 5-tuple gồm: Source IP, Dest IP, Port Source, Port Dest, protocol và packet info gồm: tcp flags, initial_timestamp, last_timestamp, packet_counter và byte counter.(Trong đó initial_timestamp được lấy từ lúc gói tin được đưa vào).

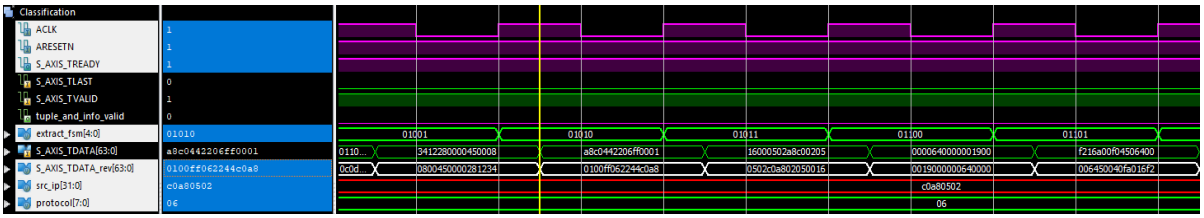
Ở trạng thái 01001 (rvc1_ipv4_vlan1_0):



Hình 3.25 Testbench VlanTag 2

Tiến hành phân tích lấy: Frame_ip_total_length = 0x0032.

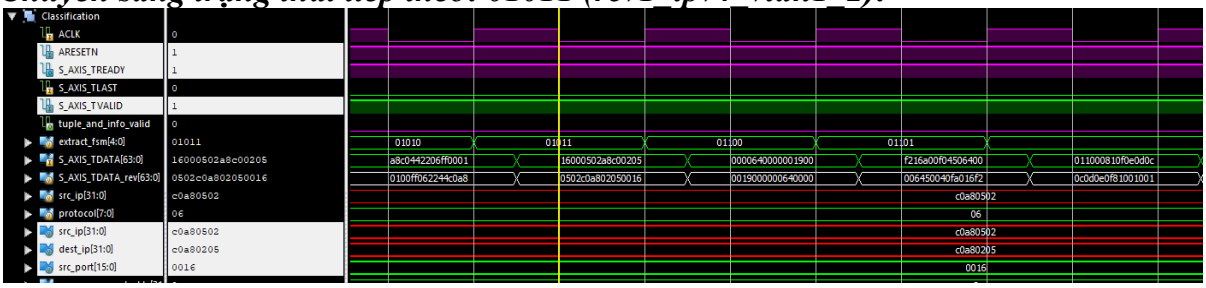
Chuyển sang trạng thái: 01010(rcv1_ipv4_vlan1_1):



Hình 3.26 Testbench VlanTag 3

Tiến hành phân tích và lấy: Protocol[7:0]: 06 (giao thức TCP) và ½ Source IP: Src_ip[31:0] = 0xc0a8.

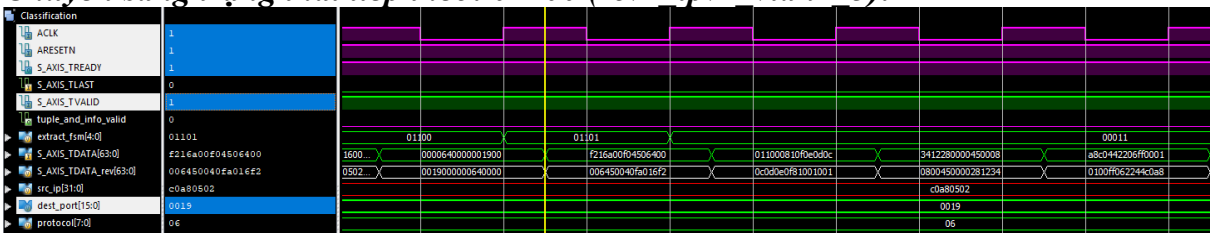
Chuyển sang trạng thái tiếp theo: 01011 (rcv1_ipv4_vlan1_2):



Hình 3.27 Testbench VlanTag 4

Tiến hành lấy: dest_ip [31:0] = c0a80205, Source IP: Src_ip[31:0] =0xc0a80502 và Source Port: src_port[15:0] = 0x0016.

Chuyển sang trạng thái tiếp theo: 01100 (rcv1_ipv4_vlan1_3):



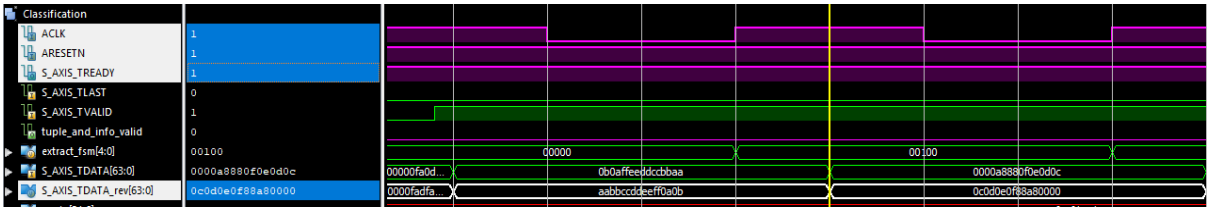
Hình 3.28 Testbench VlanTag 5

Tiến hành lấy Dest Port: dest_port[15:0]: 0x0019.

Chuyển sang trạng thái tiếp theo: 01101 (rcv1_ipv4_vlan1_4):

Ở trạng thái này, kiểm tra xem gói tin có chứa TCP flags (FIN/RST). Nếu không chứa TCP flags thì set fram_tcp_flags = 0. Sau quá trình phân tích gói tin trên, ta lấy ra 5-tuple và packet info và chuyển sang module create/Update.

- **Trường hợp gói tin chứa doubletag VLAN:**



Hình 3.29 Testbench Douple Tag Vlan

Khi có tín hiệu của clk và reset = 1, tín hiệu S_AXIS_TVALID = 1 báo rằng có gói tin hợp lệ đưa vào. Gói tin bắt đầu đưa vào: 6-byte đầu chứa MAC dest: aabbccddeeff, 6-byte tiếp thep chứa MAC source: 0x0a0b0c0d0e0f và 2 byte tiếp theo: "0x88a8": gói tin chứa VLAN doubletag.

Sau khi kiểm tra gói tin chứa VLAN doubletag, chuyển sang các trạng thái tiếp theo để thực hiện phân tích và lấy ra 5-tuple gồm: Source IP, Dest IP, Port Source, Port Dest và protocol + packet info gồm: tcp flags, initial_timestamp, last_timestamp, packet_counter và byte counter (Trong đó initial_timestamp được lấy từ lúc gói tin được đưa vào). Sau quá trình phân tích gói tin trên, ta lấy ra 5-tuple và packet info và chuyển sang module create/Update.

Phân tích testbench create/update.

Ban đầu, các frame packet sẽ được đưa vào module classification packet để phân tích và lấy ra 5-tuple, info-packet. Sau khi phân tích để lấy ra, 5-tuple cùng các info packet đưa vào modul creat/update để tiến hành quá trình tạo mới và cập nhật các flow trong BRAM.

Phân tích được 5 flow, mỗi flow chứa 5-tuple và info-packet. Tiến hành đưa lần lượt các flow vào để phân tích create/update.

Trong quá trình create/update flow, thì module cũng kiểm tra xem packet đưa vào có chứa FIN flag/RST flag hay không.

```
3'b100:  //CREATE
    begin
        ena <= 1'b1;
        wea <= 1'b1;
        dia <= {1'b1,
                frame_five_tuple_reg,
                frame_tcp_flags,
                frame_timestamp,
                frame_timestamp,
                32'b00000000000000000000000000000001,
                16'b0000000000000000,
                frame_ip_total_length};
        if ((protocol == `TCP) & (fin_rst_flags == 1'b1))
        begin  //--condition to export (end of flow)
          export_now    <= 1'b1;
          export_this   <= hash_code;
        end
        create_update_fsm <= 3'b000; //IDLE;
    end
```

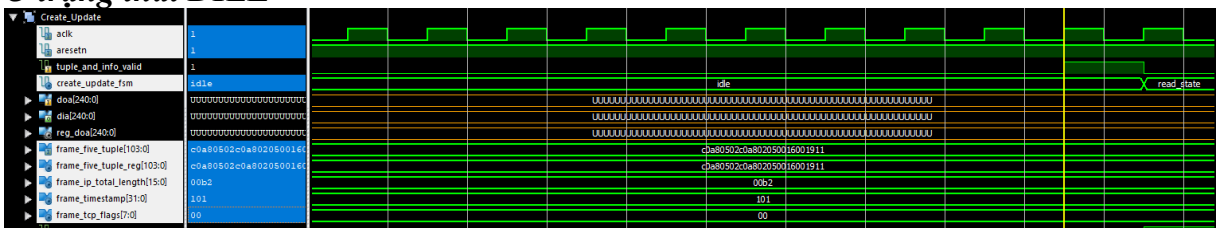
```
3'b101: //UPDATE-- Write back the updated flow-entry to memory
    begin
       ena <= 1;
       wea <= 1;
       dia <= {1'b1,frame_five_tuple_reg,
                    flow_tcp_flags,
                    flow_initial_timestamp,
                    frame_timestamp,
                    flow_pkt_counter,
                    flow_byte_counter};

       if ((protocol == `TCP) & (fin_rst_flags == 1'b1))
         begin
            export_now    <= 1;
            export_this   <= hash_code;
         end
        create_update_fsm <= 3'b000;//IDLE;

    end
```

Hình 3.30 Trạng thái Creat và Update gói tin

Trường hợp: Module Create/update nhận gói tin không chứa FIN flag/ RST flag *Ở trạng thái DILE*



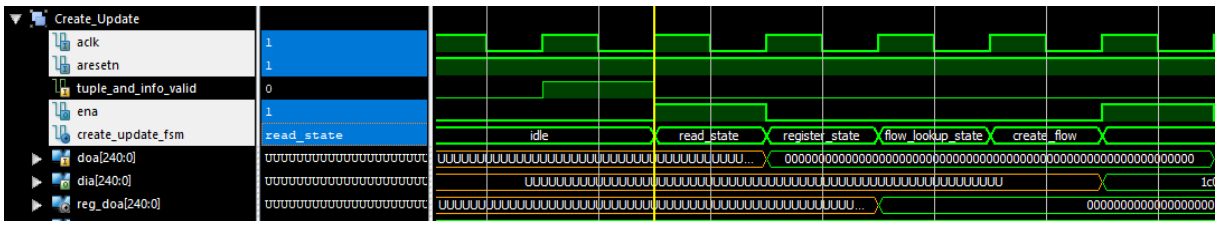
Hình 3.31 Testbench Create và Update 1

Khi có tín hiệu xung CLK và ARESET = '1', các flow lần lượt được đưa vào:

Thực hiện gán các giá trị của gói tin bao gồm 5-tuple và info-packet vào frame chứa trong module Create/update flow: doa[240:0] **: busy flag, 5tuple, TCP flags, Timestamp(initialtime, lasttime), packet counter, byte_counter**vào**:frame_five_tuple_reg[103:0],frame_ip_total_length [15:0], frame_timestamp [31:0] và frame_tcp_flags[7:0].**

⇨ Để tiến hành quá trình: create or update flow

- Khi module packet classification phân tích lấy đủ 5-tuple và info packet đưa vào modul create/update: *stuple_and_info_valid = '1'* cho biết modul vừa nhận được 5-tuple *ena = '1'*: tiến hành chờ xung clk tiếp theo sẽ đặt giá trị flow mới *(doa[240:0])* lên dataout của PortA trong Bram

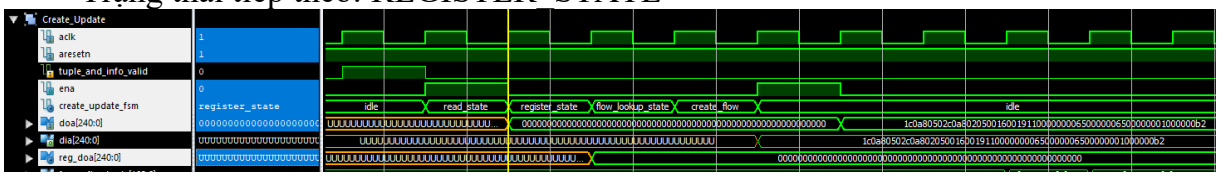


Hình 3.32 Testbench Create và Update 2

Và chuyển sang trạng thái tiếp theo: READ_STATE.

Trạng thái READ_STATE sẽ tiếp tục chuyển sang trạng thái tiếp theo khi ena=0.

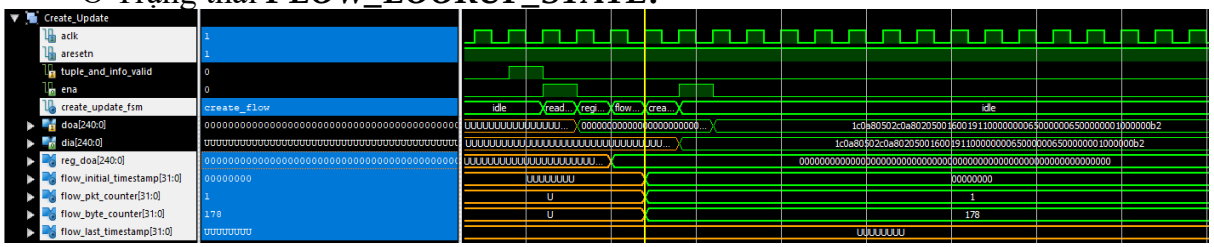
Trạng thái tiếp theo: REGISTER_STATE =>



Hình 3.33 Testbench Create và Update 3

Khi enable hạ xuống 0, tiến hành đọc data flow nằm trong Bram ra và lưu memory's output (register_doa): reg_doa <= doa và chuyển sang trạng thái FLOW_LOOKUP_STATE

Ở Trạng thái ***FLOW_LOOKUP_STATE:***



Hình 3.34 Testbench Create và Update 5

Tiến hành đặt các giá trị của reg_doa[240:0] vào các flow và gán giá trị của 5-tuple của gói tin vào thanh ghi trong modul creat/update: *flow_5_tuple := reg_doa(240-1 downto 136);*

```
when FLOW_LOOKUP_STATE =>
    flow_byte_counter <= reg_doa(32-1 downto 0) + frame_ip_total_length;
    flow_pkt_counter <= reg_doa(64-1 downto 32) + 1;
    flow_initial_timestamp <= reg_doa(128-1 downto 96);
    flow_tcp_flags <= reg_doa(136-1 downto 128) or frame_tcp_flags;
    flow_5_tuple := reg_doa(240-1 downto 136);
```

Hình 3.35 Trạng thái Lookup flow

Sau đó tiến hành kiểm tra busy flags chứa trong flow:

Điều kiện để flow create or update:

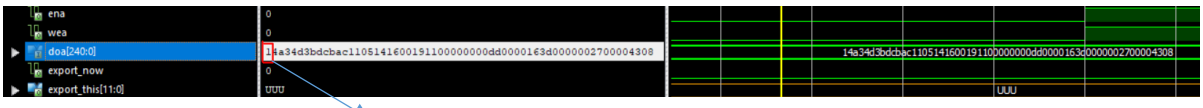
```
3'b011:
    begin
      flow_byte_counter        <= reg_doa[32-1 : 0] + frame_ip_total_length;
      flow_pkt_counter         <= reg_doa[64-1 : 32] + 1;
      flow_initial_timestamp   <= reg_doa[128-1 : 96];
      flow_tcp_flags           <= reg_doa[136-1:128] || frame_tcp_flags;
      flow_5_tuple             = reg_doa[240-1 : 136];  //:=
       if (reg_doa[`MEM_ENTRY_STATUS_INDEX] == 1'b1)                  //if it's '1', then it has a flow on it
          if (flow_5_tuple != frame_five_tuple_reg )                 //check if no collision is present
          |     create_update_fsm <= 3'b110 ;//COLLISION_PRESENT;            // collision
          else
              create_update_fsm <= 3'b101 ;//UPDATE_FLOW;             // frame 5-tuple matched flow-entry 5-tuple
       else                                                                 //No flow on memory
          create_update_fsm       <= 3'b100 ;//CREATE_FLOW;
       protocol    <= frame_five_tuple_reg[8-1 : 0];
       fin_rst_flags <= frame_tcp_flags[0] || frame_tcp_flags[2];
    end
```

Hình 3.36 Điều kiện update hay create

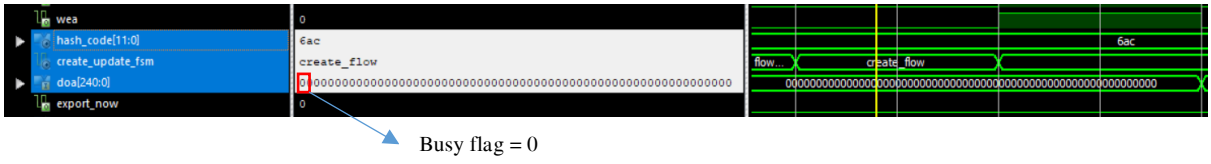
Khi Busy *flag = 1:* (reg_doa[240] == 1'b1) *Module create/update lấy 5-tuple của gói tin mới nhận so sánh với 5- tuple trong gói tin chứa trong Bram dựa vào hash_code.*

- Nếu giống nhau: Tiến hành update flow.
- Nếu khác : Xảy ra xung đột.

Khi Busy *flag = 0 :* Tiến hành create các flow vào trong BRAM.



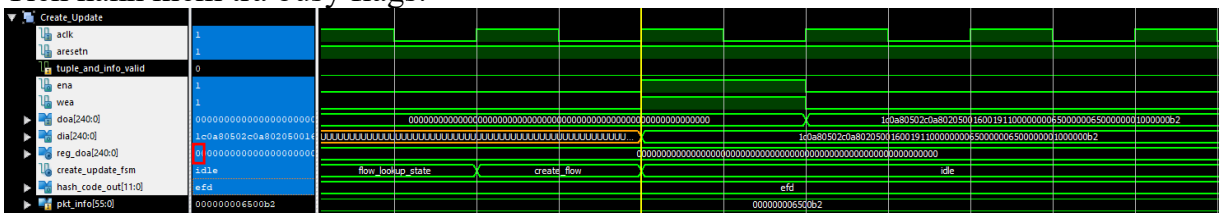
Hình 3.37 Testbench create và Update 6



Hình 3.38 Testbench create và Update 7

CREATE PROCESS: 5 flow đầu tiên khi được đưa vào thì nó sẽ tiến hành create vào BRAM vì nó là frame đầu tiên:

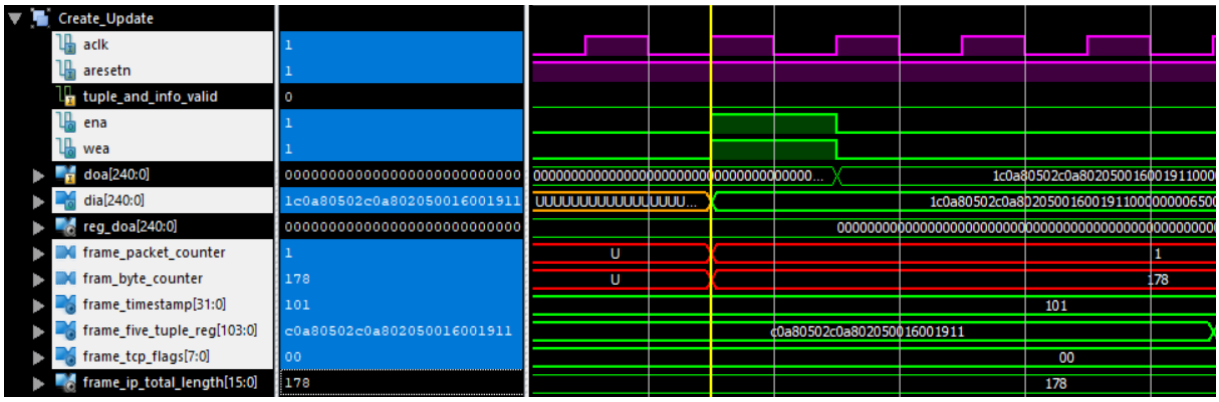
Tiến hành kiểm tra busy flags:



Hình 3.39 Testbench create và Update 7

Trước khi create, ta thấy tại hash_code_out[11:0] = efd, doa[240:0] = 0. Khi chưa create/updtat giá trị byte_counter, packet_counter chưa có giá trị. Busy flag có giá trị '0', có nghĩa là tại địa chỉ *efd* đó trong Bram chưa tồn tại flow, ta tiến hành create flow vào Bram:

- *Enable Bram(ena<= '1') và Enable write in Bram(wea <= '1') tiến hành ghi vào Bram:*
- Create các giá trị của flow trong gói tin vào bram gồm: busy flag, 5-tuple, tcp flags, initial_timestamp, last_timestamp, packet counter, byte counter *(dia[240:1])*



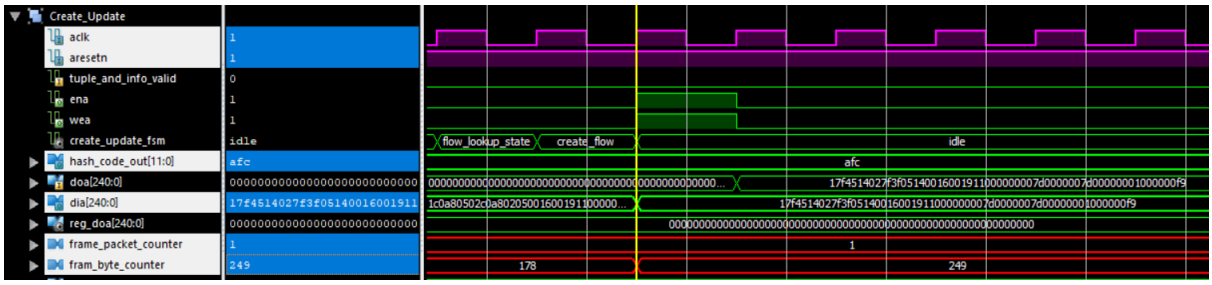
Hình 3.40 Testbench create và Update 8

Hình 3.41 Testbench create và Update 9

Trước khi create, chưa có gói tin nào tồn tại nên các giá trị sẽ là U. Sau khi create gói tin đầu tiên vào thì byte_counter = 178 (0x00b2) và packet_couter = 1.

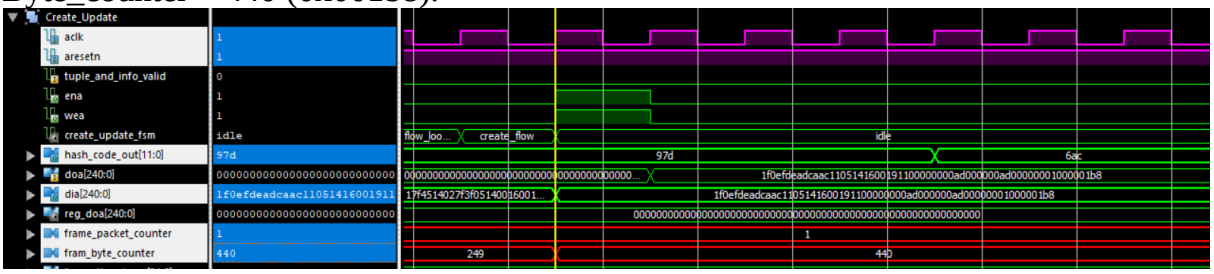
Tương tự: 4 flow còn lại sẽ được create vào BRAM.

Đối với gói tin thứ 2: là gói đầu tiên ở addr: afc cho nên packet_couter =1 và Byte_counter = 249 (0x00f9).



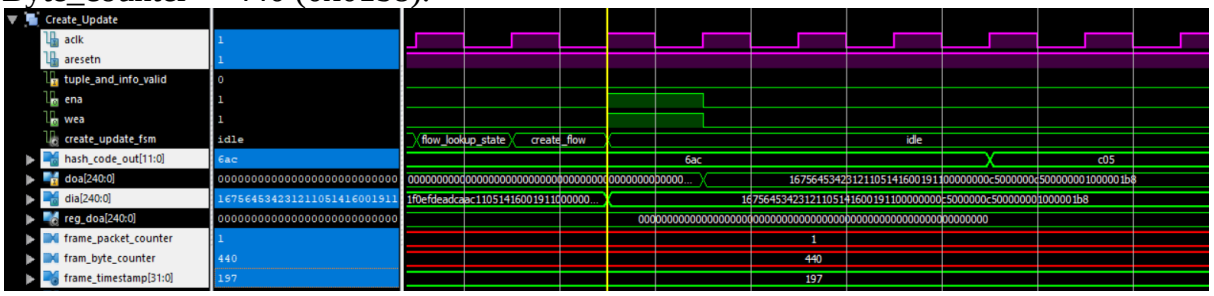
Hình 3.42 Testbench create và Update 10

Đối với gói tin thứ 3: là gói đầu tiên ở addr: 97d cho nên packet_couter =1 và Byte_counter = 440 (0x001b8).



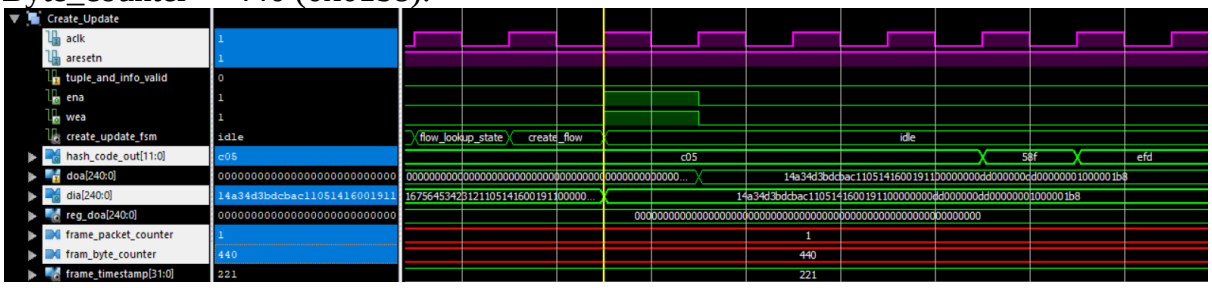
Hình 3.43 Testbench create và Update 11

Đối với gói tin thứ 4: là gói đầu tiên ở addr: 6ac cho nên packet_couter =1 và Byte_counter = 440 (0x01b8).



Hình 3.44 Testbench create và Update 12

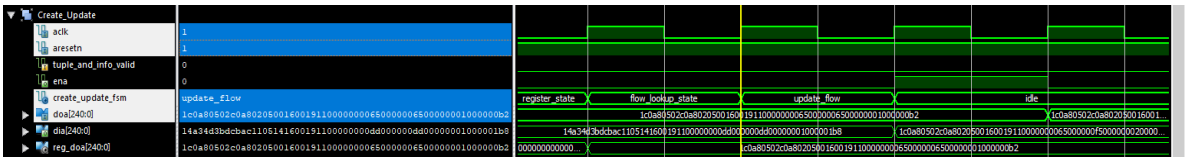
Đối với gói tin thứ 5: là gói đầu tiên ở addr: c05 cho nên packet_couter =1 và Byte_counter = 440 (0x01b8).



Hình 3.45 Testbench create và Update 13

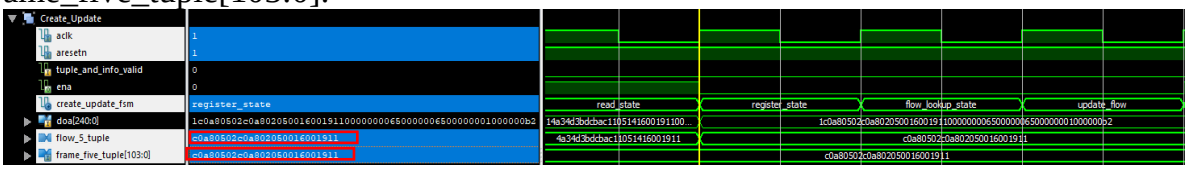
Sau khi kết thúc frame paket gồm 5 gói tin, nó sẽ chạy lại frame đó, lúc này ta tiến hành kiểm tra và update.

UPDATE PROCESS: Kiểm tra busy flag =1:



Hình 3.46 Testbench create và Update 14

Nếu bit busyflags có giá trị '1', ta tiến hành lấy 5-tuple của gói tin so sánh với 5-tuple trong Bram. flow_5_tuple := reg_doa(240-1 downto 136) và frame_five_tuple[103:0].



Hình 3.47 Testbench create và Update 15

Nếu 5-tuple của flow mới và 5-tuple trong Bram giống nhau, tiến hành Update flow vào trong Bram:

- *Enable Bram(ena<= '1') và Enable write in Bram(wea <= '1') tiến hành ghi vào Bram: dia <= '1' & frame_five_tuple_reg & flow_tcp_flags & flow_initial_timestamp & frame_timestamp & flow_pkt_counter & flow_byte_counter;*

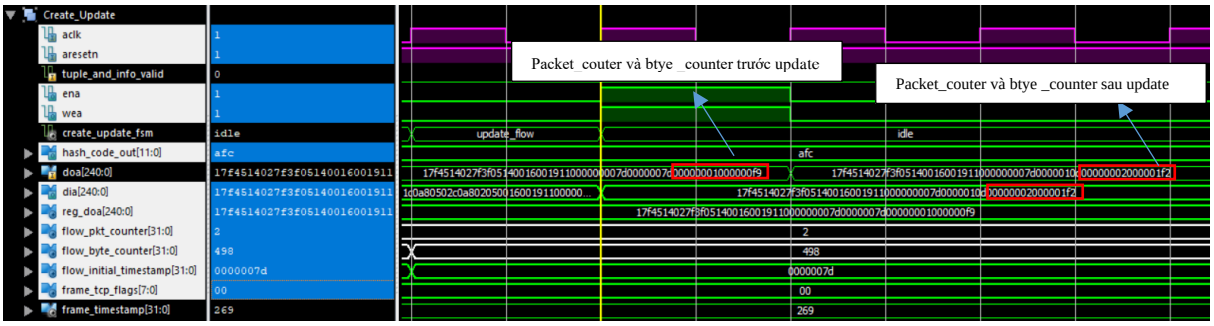
Update các giá trị pkt_info vào flow trong Bram bao gồm: initial_timestamp, packet counter, byte counter vào Bram *(dia[240:0]).*

Update gói tin 1:

Hình 3.48 Testbench create và Update 16

Gói tin 1: Tại addr = efd. Khi mới create, packet_couter =1. Byte_counter = 178 (0x00b2). Sau khi update : packet_counter =2. Byte_counter = 356 (0x0164).

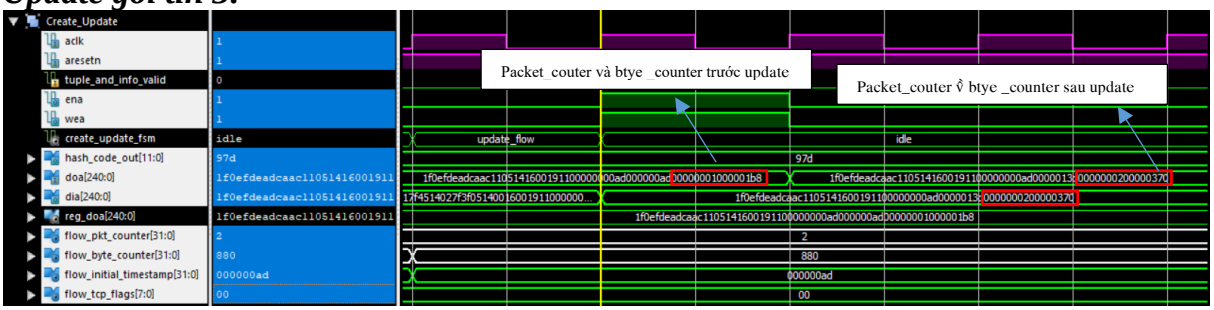
Update gói tin 2:



Hình 3.49 Testbench create và Update 17

Gói tin 2: Tại addr = afc. Khi mới create, packet_couter =1. Byte_counter = 249 (0x00f9). Sau khi update : packet_counter =2. Byte_counter = 498 (0x01f2).

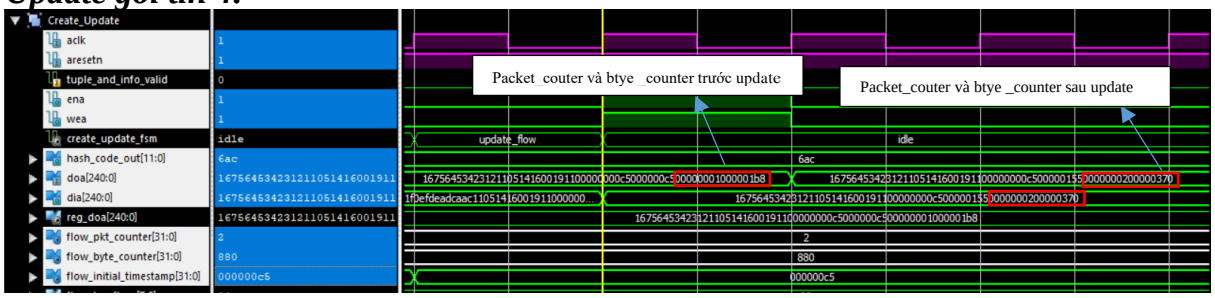
Update gói tin 3:



Hình 3.50 Testbench create và Update 17

Gói tin 3: Tại addr = 97d. Khi mới create, packet_couter =1. Byte_counter = 440 (0x01b8). Sau khi update: packet_counter =2. Byte_counter = 880 (0x0370)

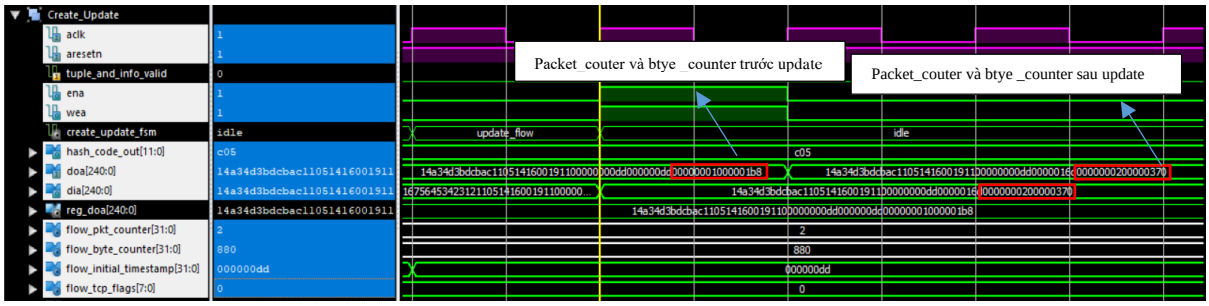
Update gói tin 4:



Hình 3.51 Testbench create và Update 18

Gói tin 4: Tại addr = 6ac. Khi mới create, packet_couter =1. Byte_counter = 440 (0x01b8). Sau khi update: packet_counter =2. Byte_counter = 880 (0x0370).

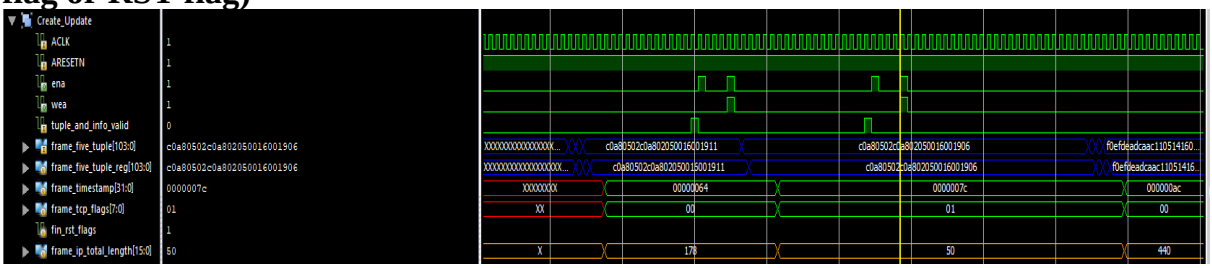
Update gói tin 5:



Hình 3.52 Testbench create và Update 19

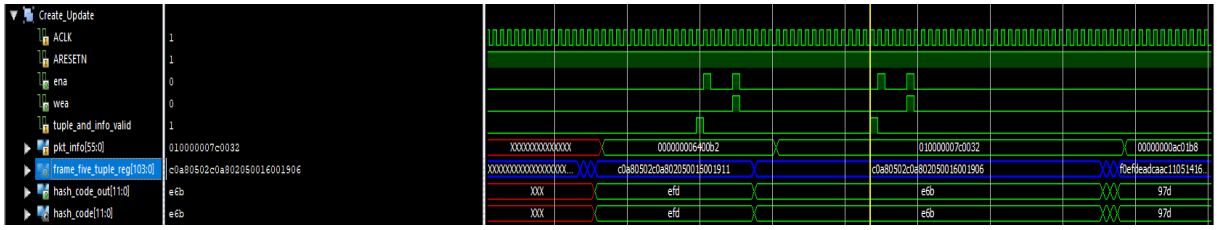
Gói tin 5: Tại addr = c05. Khi mới create, packet_couter =1. Byte_counter = 440 (0x01b8). Sau khi update: packet_counter =2. Byte_counter = 880 (0x0370).*Nếu 5-tuple của flow mới và 5-tuple trong Bram khác nhau, thì xảy ra xung đột: collision_counter_int <= collision_counter_int +1. Thì sẽ tăng biến đếm giá trị xung đột lên ++1, và chuyển sang trạng thái ban đầu để chờ phân tích gói tin tiếp theo.*

Trường hợp: Module Create or update nhận một **packet** có chứa **TCP Flags (Fin flag or RST flag)**



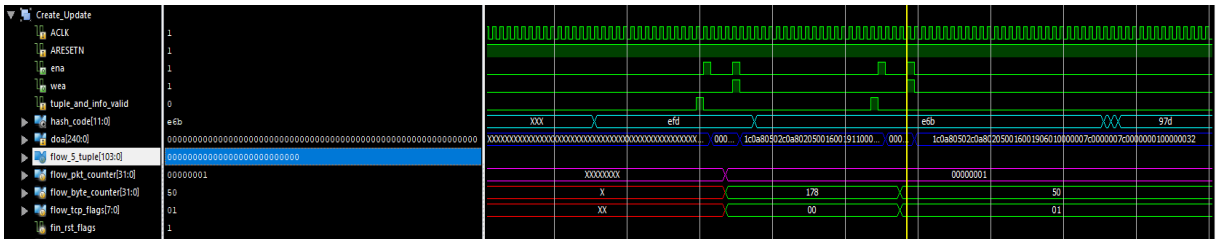
Hình 3.53 Testbench create và Update 20

Hashing function trong **Create or update module** sẽ thực hiện **Hasing frame_five_tuple_reg** *thành* **hash_code**.



Hình 3.54 Testbench create và Update 21

Sau đó, dùng **địa** chỉ hash code này để truy cập vào **Bram**và **read doa** của flow ở vị trí đó.

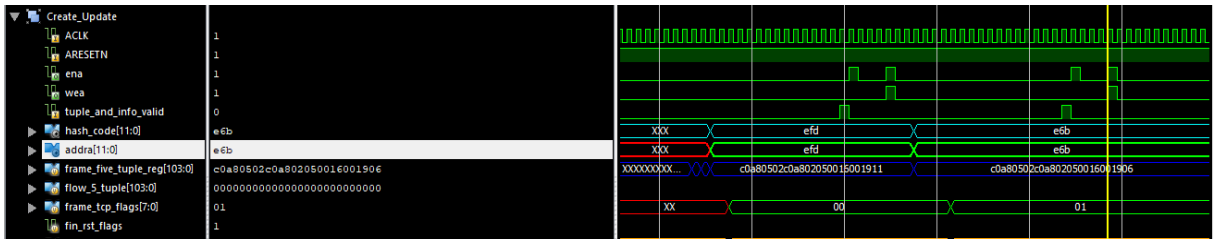


Hình 3.52 Testbench create và Update 22

Module Create or update sẽ kiểm tra **Busy flag** và so sánh **flow_5_tuple** của flow từ BRAM với frame**_five_tuple_reg** của gói tin với nhận vào.

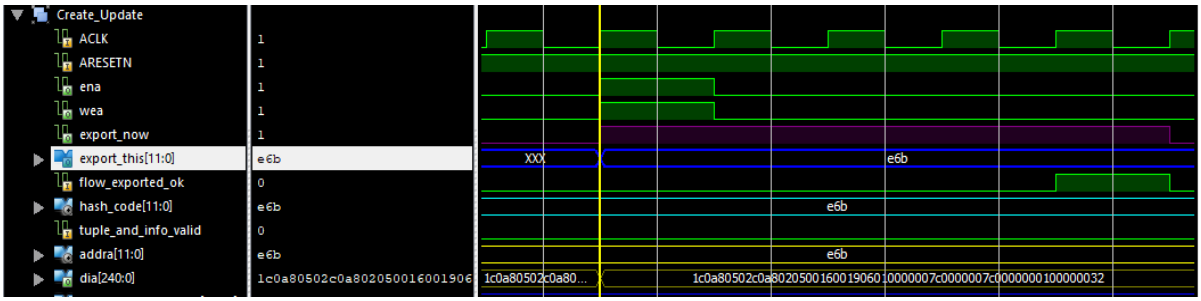
- Khi **Busy flag = 0**, gói tin mới nhận sẽ được **Create** vào **address** của **flow** đang xét trong **Bram**. Đồng thời, kiểm tra **protoco**l và **fin_rst_flags** của gói tin mới nhận vào.

Ở tại địa chỉ **addra[11:0]: e6b:** gói tin mới nhận sẽ được **new create** và có chứa **TCP flags (Fin Flag).**



Hình 3.55 Testbench create và Update 23

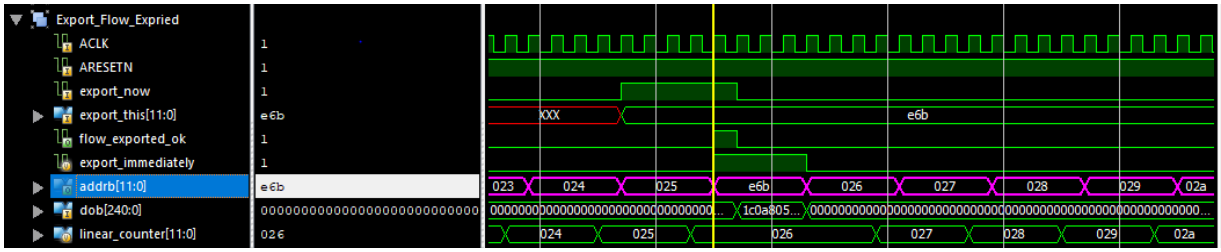
Sau khi gói tin đã được tạo mới trong **Bram**, module **Create or update** sẽ set signal **Export_now** lên 1 và **Export_this** giá trị **hash_code** vị trí của **Flow** vừa **wirte (dia)** gói tin có chưa **Fin flag** vào **Bram**. Nó sẽ chờ tín hiệu **đẩy lên** của signal **Flow_export_ok** khi **module Export expired flow from memory** nhận được thông báo **Export now** và **hash_code** của **export this** và set **Export now** về trạng thái **thấp**.



Hình 3.56 Testbench create và Update 24

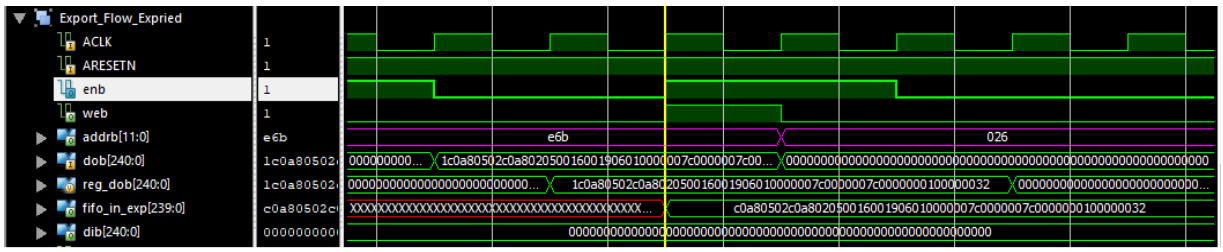
Sinh viên thực hiện : Nguyễn Duy Hà Sơn – Đặng Sỹ Phi Hùng
Người hướng dẫn : TS. Trần Hoàng Vũ

Module Export expired sau khi nhận được thông báo của **Create or update module** sẽ dừng lại việc kiểm tra **Flow expired sequential** mà truy cập vào **Flow** mang địa chỉ **hash_code[11:0] : e6b** và **read flo**w này qua **dob** từ **Bram** ra **Export expired flow from memory module**.



Hình 3.57 Tetsbench Export Expired 1

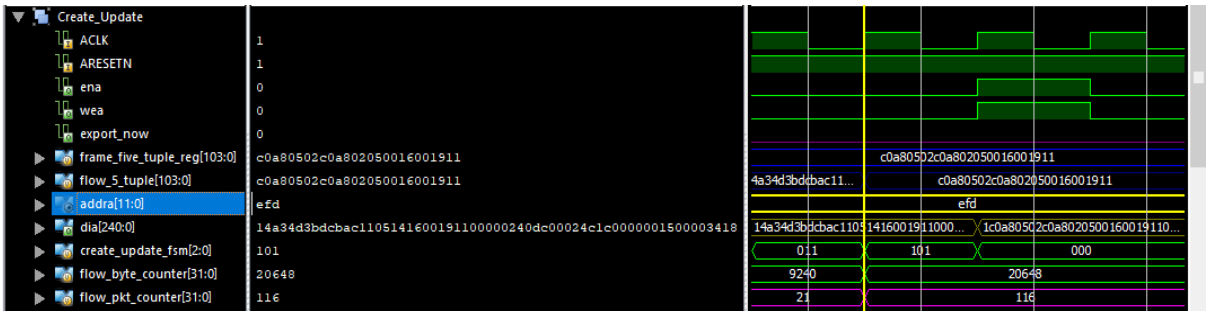
Thông tin read từ dob của **port B (Bram)** sẽ đưa vào **registor reg_doa** của **module Export** và sau đó được **Export.** Khi **Export expired module** set signal **enb** và **web** lên mức cao, **Bram** sẽ **write dib[240:0] = 0** vào vị trí của **Flow** mới được export, xóa dữ liệu củ Flow thành **empty flow**.



Hình 3.58 Tetsbench Export Expired 2

Sau khi kết thúc việc **Export flow expired** có chứa **Fin flag**, thì quá trình kiểm tra **Flow expired sequential** của **Export Module** sẽ được tiếp tục tại địa chỉ tạm dừng trước đó.

Module Create or update sẽ kiểm tra **Busy flag** và so sánh **flow_5_tuple** của gói tin từ BRAM với **frame_five_tuple_reg** của gói tin với nhận vào.
Khi **Busy flag = 1** và **flow_5_tuple** giống với **frame_five_tuple_reg** gói tin mới nhận sẽ được **Update** vào **address** của **flow** đang xét trong **Bram** (Flow hiện tại sẽ được **update last_timestamp, flow_byte_counter**, **flow_pkt_counter**). Đồng thời, kiểm tra **protoco**l và **fin_rst_flags** của gói tin mới nhận vào.



Hình 3.59 Tetsbench Export Expired 3

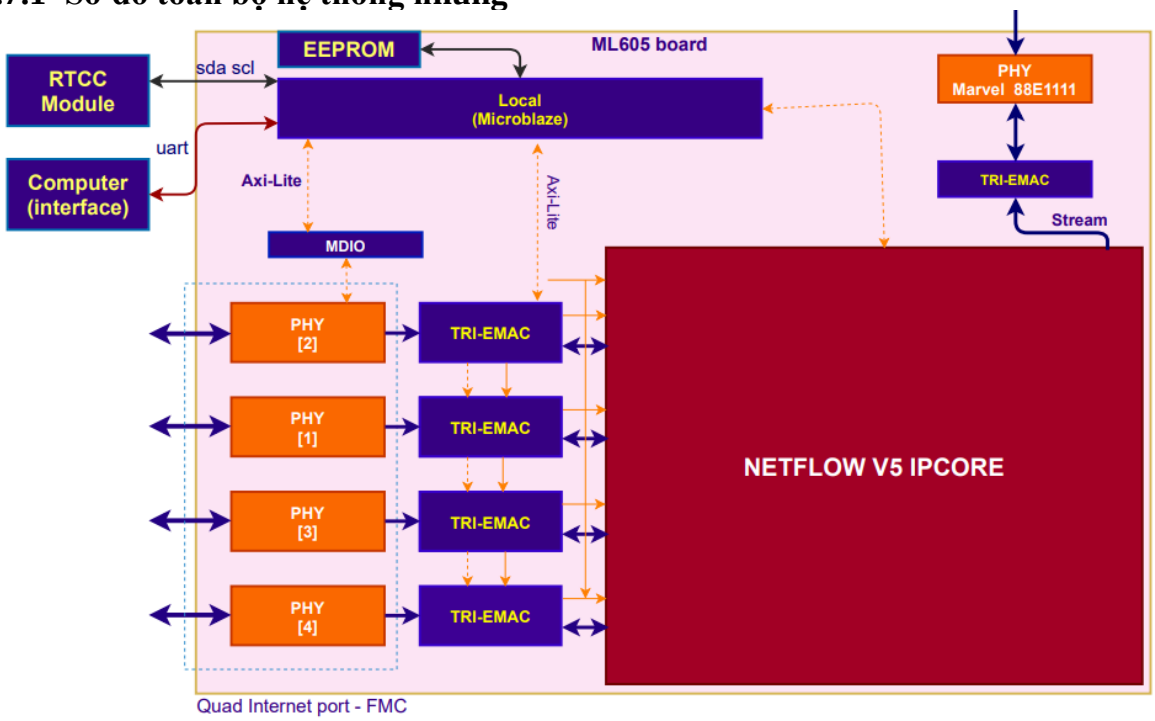
Trường hợp: gói tin mới được Update là một **TCP packet** có chứa **TCP Flags (Fin flag or RST flag)** thì quá trình thông tin giữa **Module Create or update** và **Module Export expired flow from memory**, quá trình **Export Flow** đó cũng sẽ được thực hiện giống với trường hợp **Create** ở trên.

Như vậy ta có thể kết luận việc sử dụng ngôn ngữ Verilog đã mô tả thành công kết quả thuật toán NetFlow V5.

3.3.7 Thực thi hệ thống nhúng trên FPGA VIRTEX -6 ML605 của Xilinx

Với hệ thống đã thiết kế chúng ta thực hiện nhúng trên Kit FPGA Xilinx sử dụng phần mềm *Xilinx Studio Platform (XPS)*, sẽ bao gồm lõi IP core Netflow_V5 đã được viết bằng ngôn ngữ verilog để xử lý tốc độ cao chạy song song với một lõi IP mềm vi điều khiển MicroBlaze trên FPGA Virtex 6 được lập trình được bằng ngôn ngữ C để điều khiển lõi IP core Netflow_V5 hoạt động. Cùng với đó, để quan sát và chuyển dữ liệu từ phần cứng Netflow lên công cụ phần mềm Splunk Enterprise giám sát, phân tích cần có một giao diện để quan sát và điều khiển. Từ những yêu cầu trên, chương 4 trình bày quy trình xây dựng trình điều khiển bằng ngôn ngữ C trên công cụ SDK của Xilin và phát triển GUI bằng ngôn ngữ Java trên môi trường Eclips Java Development.

3.3.7.1 Sơ đồ toàn bộ hệ thống nhúng



Hình 3.60 Sơ đồ hệ thống nhúng trên FPGA

Nhiệm vụ các khối:

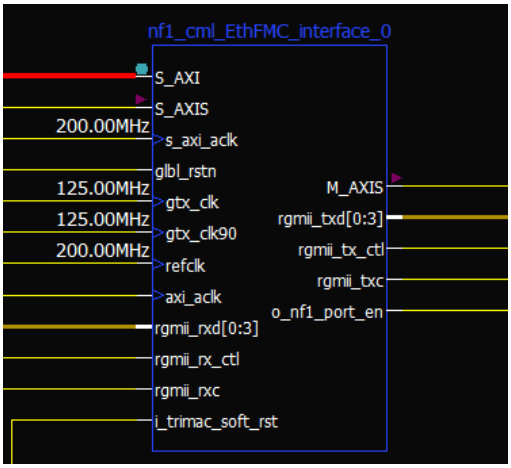
- RTC Moudule: cung cấp thời gian thwucj cho hệ thống.
- Computer: Máy tính sẽ truyền dữ liệu cần hiển thị về vi điều khiển MicroBlaze thông qua đường UART.
- MicroBlaze: Khối vi điều khiển được lập trình C cấu hình UART để truyền nhận dữ liệu từ máy tính và IP core Netflow V5.
- IP core Netflow_V5: triển khai thuật toán Netflow V5 nhận data và trích xuất các trường mạng cần thiết. Lõi IP này nhận dữ liệu từ vi điều khiển MicroBlaze thông qua chuẩn giao tiếp AXI4-Stream và AXI4-Lite.
- EEPROM: lưu trữ thông tin của người dùng và thông tin cấu hình cho hệ thống.
- Ethernet FMC (PHY): Truyền và nhận các gói tin thông qua 4 cổng Ethernet (Port 0, Port1, Port2, Port3).

3.3.7.2 Tích hợp hệ thống nhúng cùng các lõi IP:

- **MDIO core:** Đọc tốc độ automatically của PHY.

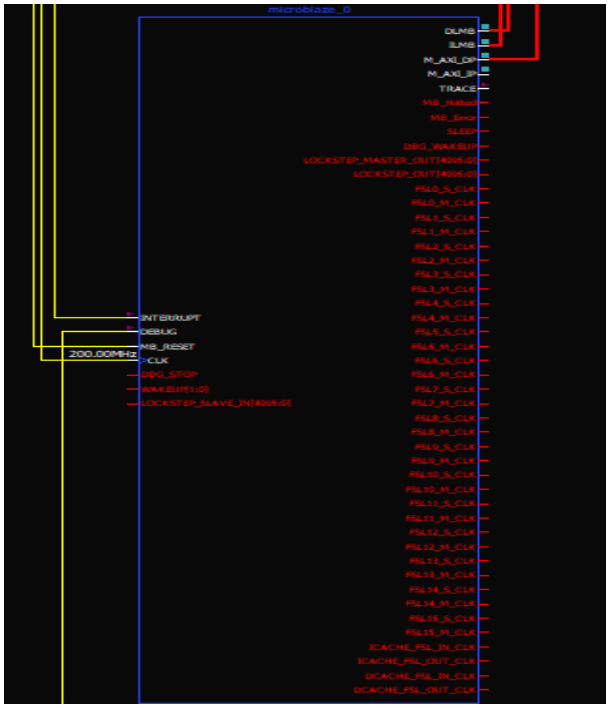
Hình 3.61 MDIO core

- **Tri-EMAC core:** Lõi IP này được sử dụng để đọc điều khiển tốc độ PHY (10/100/1000Mbs), số lần nhận/truyền PHY thông qua kết nối AXI Lite.



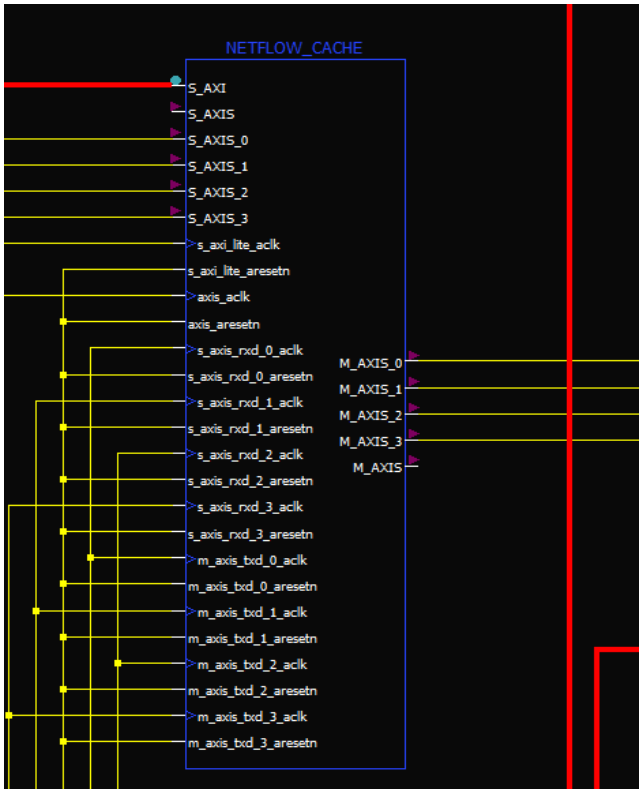
Hình 3.62 Tri-EMAC core

- **Soft Processor Microblaze:** Bộ xử lý này dùng để điều khiển hệ thống bằng cách quản lý một số IP Core như: NetFlow Cache Core, IIC core, MDIO, Tri-EMAC, …



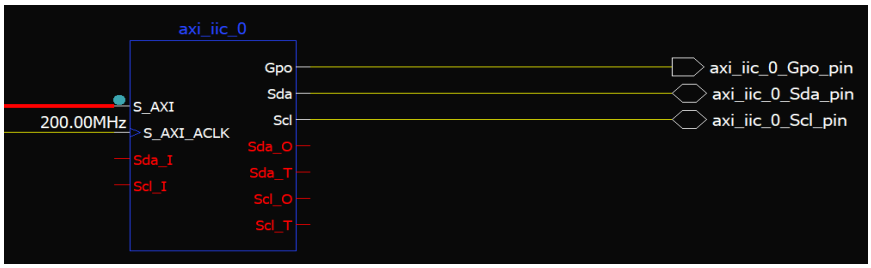
Hình 3.63 Soft processor microblaze

- **NetFlow V5 IP Core:** Đây là lõi thiết kế người dùng được sử dụng để nhận dữ liệu (gói tin) từ lõi Tri-EMAC thông qua Axi4-Stream để thực hiện phân tích và trích xuất các trường.



Hình 3.64 NetFlow V5 IPCore

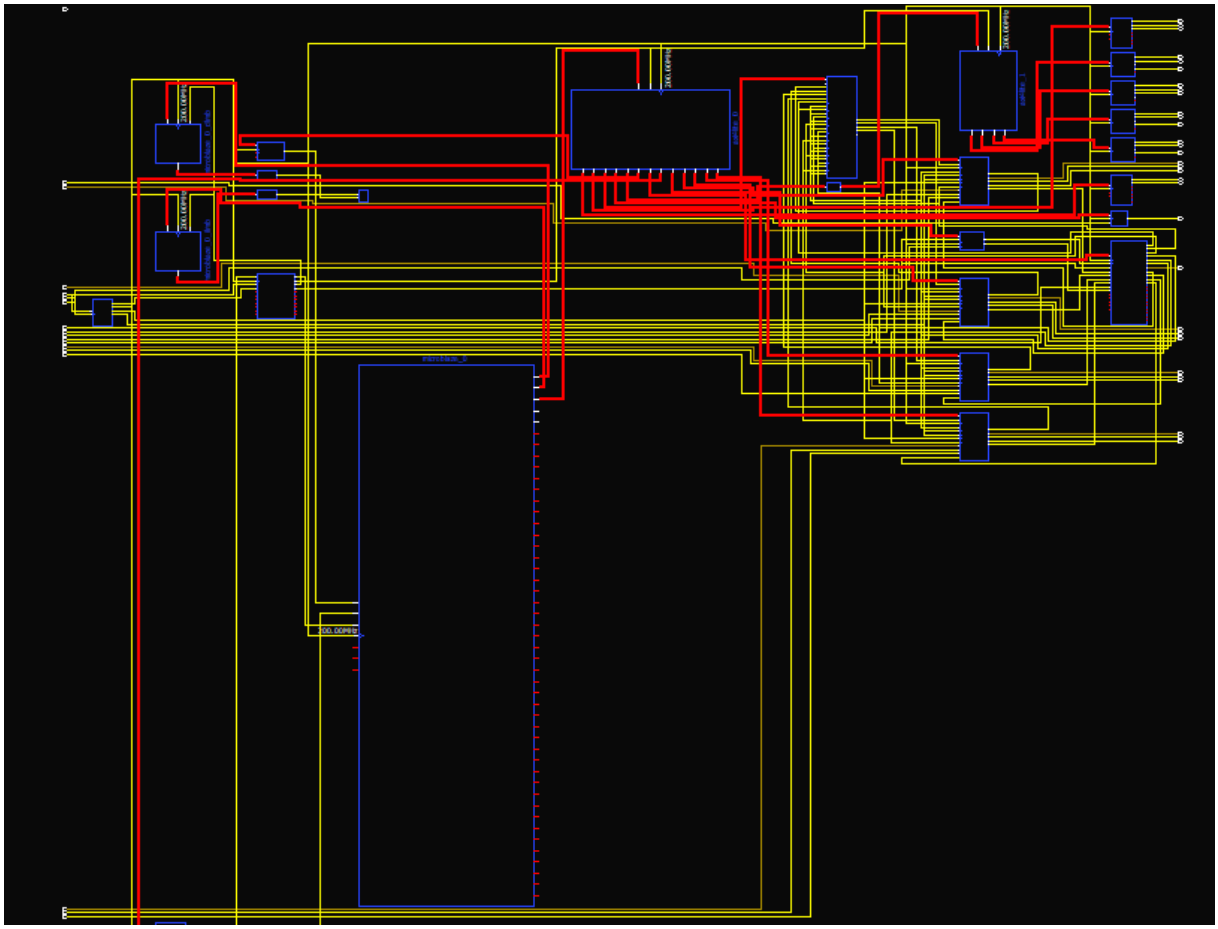
- Ngoài ra còn một số IP core để kết nối bên trong và bên ngoài. Chúng là Axi Interconnect, IIC Core, UART Core. Được sử dụng để điều khiển Microblaze tới lõi IP khác.



Hình 3.65 Axi IIC Core

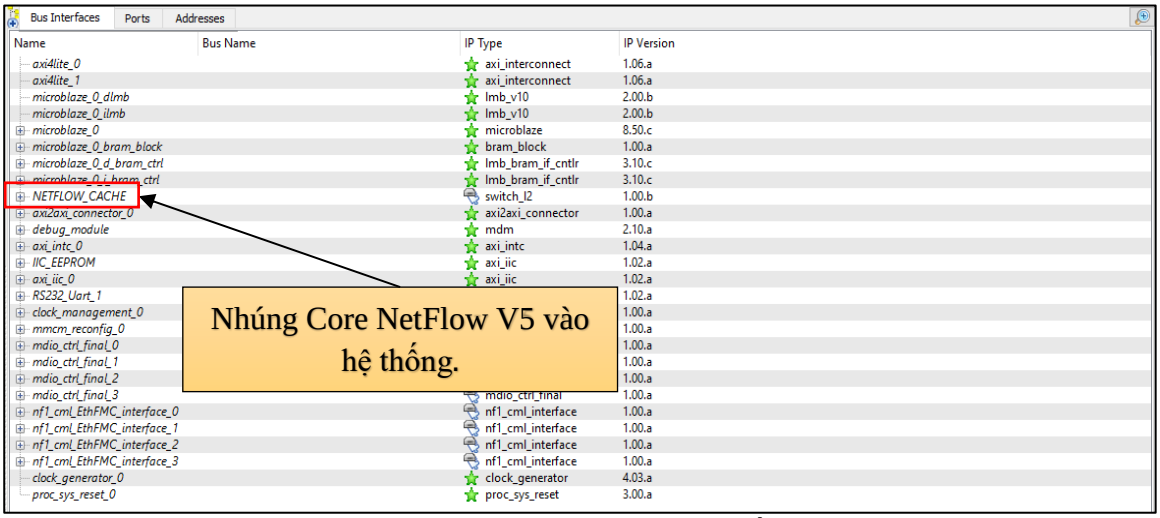
Hình 3.66 UART Core

Dưới đây là toàn bộ thiết kế hệ thống được tạo bởi phần mềm Xilinx Stadium Platform (XPS) 14.7 của Xinlinx:



Hình 3.67 Tổng hệ thống nhúng IPCore NetFlow V5 và các ngoại vi trên XPS

Giao diện Bus Hệ thống



Hình 3.68 Giao diện Bus Hệ thống

Sinh viên thực hiện : Nguyễn Duy Hà Sơn – Đặng Sỹ Phi Hùng
Người hướng dẫn : TS. Trần Hoàng Vũ

3.4 Kết luận chương

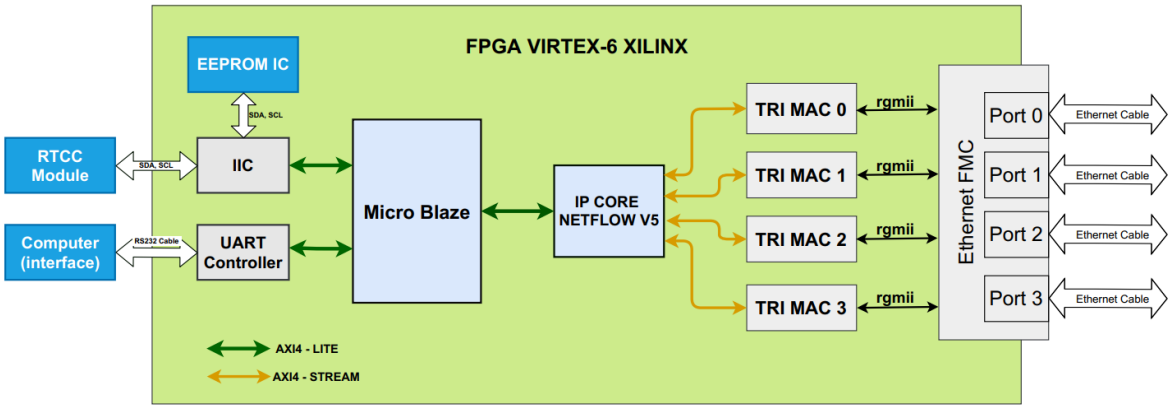
Qua chương 3, nhóm tác giả đã trình bày về thiết kế IP Core NetFlow V5 bằng ngôn ngữ Verilog để mô tả kiến trúc của thuật toán NetFlow. Nhóm tác giả đã thực hiện kiểm tra core bằng mô phỏng ISIM công cụ mô phỏng của Xilinx. Chương này cũng đề cập đến việc thực hiện tích hợp hệ thống nhúng để nạp xuống board ML605 với hệ thống xử lý cùng Microblaze vi xử lý mềm cùng các IP core trên hệ thống như IIC, UART, Tri-Mac. Và để điều khiển được hệ thống thì qua chương 4 nhóm tác giả sẽ xây dựng chương trình điều khiển cùng với giao diện để quản lý.

Chương 4: XÂY DỰNG CHƯƠNG TRÌNH ĐIỀU KHIỂN VÀ GIAO DIỆN NGƯỜI DÙNG CHO HỆ THỐNG

4.1 Giới thiệu chương

Với hệ thống đã thiết kế và tích hợp ở chương 3 khi nhúng trên Kit FPGA Xilinx sẽ bao gồm lõi IP core Netflow_V5 đã được viết bằng ngôn ngữ verilog để xử lý tốc độ cao chạy song song với một lõi IP mềm vi điều khiển MicroBlaze trên FPGA Virtex 6 được lập trình được bằng ngôn ngữ C để điều khiển lõi IP core Netflow_V5 hoạt động. Cùng với đó, để quan sát và chuyển dữ liệu từ phần cứng Netflow lên công cụ phần mềm Splunk Enterprise giám sát, phân tích cần có một giao diện để quan sát và điều khiển. Từ những yêu cầu trên, chương 4 trình bày quy trình xây dựng trình điều khiển bằng ngôn ngữ C trên công cụ SDK của Xilin và phát triển GUI bằng ngôn ngữ Java trên môi trường Eclips Java Development.

4.2 Sơ đồ toàn bộ hệ thống nhúng



Hình 4.1. Sơ đồ hệ thống nhúng

4.3 Bản đồ địa chỉ thanh ghi nền (Base-Address) của các IP core:

Bản đồ dưới đây được tạo ra bởi các công cụ XPS Xilinx. Nó bao gồm Địa chỉ cơ sở và địa chỉ cao của mỗi lõi IP. Để điều khiển đọc/ghi các thanh ghi trong lõi IP netflow_cache, IIC_EEPROM, axi_ICC_0(RTC) và lõi IP Tri EMAC.

Bus Interfaces | Ports | Addresses

Instance	Base Name	Base Address	High Address	Size	Bus Interface(s)	Bus Name	Lock
microblaze_0's Address Map							
microblaze_0_d_bram_ctrl	C_BASEADDR	0x00000000	0x0000FFFF	64K	SLMB	microblaze_0_dlmb	
microblaze_0_i_bram_ctrl	C_BASEADDR	0x00000000	0x0000FFFF	64K	SLMB	microblaze_0_ilmb	
RS232_Uart_1	C_BASEADDR	0x40600000	0x4060FFFF	64K	S_AXI	axi4lite_0	
axi_iic_0	C_BASEADDR	0x40800000	0x4080FFFF	64K	S_AXI	axi4lite_0	
IIC_EEPROM	C_BASEADDR	0x40840000	0x4084FFFF	64K	S_AXI	axi4lite_0	
axi_intc_0	C_BASEADDR	0x41200000	0x4120FFFF	64K	S_AXI	axi4lite_0	
debug_module	C_BASEADDR	0x41400000	0x4140FFFF	64K	S_AXI	axi4lite_0	
NETFLOW_CACHE	C_BASEADDR	0x50000000	0x5FFFFFFF	256M	S_AXI	axi4lite_0	
nf1_cml_EthFMC_interface_3	C_BASEADDR	0x76000000	0x7600FFFF	64K	S_AXI	axi4lite_0	
nf1_cml_EthFMC_interface_2	C_BASEADDR	0x76020000	0x7602FFFF	64K	S_AXI	axi4lite_0	
nf1_cml_EthFMC_interface_1	C_BASEADDR	0x76040000	0x7604FFFF	64K	S_AXI	axi4lite_0	
nf1_cml_EthFMC_interface_0	C_BASEADDR	0x76060000	0x7606FFFF	64K	S_AXI	axi4lite_0	
clock_management_0	C_BASEADDR	0x78E00000	0x78E0FFFF	64K	S_AXI	axi4lite_0	
mdio_ctrl_final_3	C_BASEADDR	0x79A00000	0x79A0FFFF	64K	S_AXI	axi4lite_1	
axi2axi_connector_0	C_S_AXI_RNG00_BASEADDR	0x79A00000	0x79A0FFFF	64K	S_AXI	axi4lite_0	
mdio_ctrl_final_2	C_BASEADDR	0x79A20000	0x79A2FFFF	64K	S_AXI	axi4lite_1	
axi2axi_connector_0	C_S_AXI_RNG01_BASEADDR	0x79A20000	0x79A2FFFF	64K	S_AXI	axi4lite_0	
mdio_ctrl_final_1	C_BASEADDR	0x79A40000	0x79A4FFFF	64K	S_AXI	axi4lite_1	
axi2axi_connector_0	C_S_AXI_RNG02_BASEADDR	0x79A40000	0x79A4FFFF	64K	S_AXI	axi4lite_0	
mdio_ctrl_final_0	C_BASEADDR	0x79A60000	0x79A6FFFF	64K	S_AXI	axi4lite_1	
axi2axi_connector_0	C_S_AXI_RNG03_BASEADDR	0x79A60000	0x79A6FFFF	64K	S_AXI	axi4lite_0	
mmcm_reconfig_0	C_BASEADDR	0x7DE00000	0x7DE0FFFF	64K	S_AXI	axi4lite_0	

Hình 4.2. Giao diện của MicroBlaze Address Map

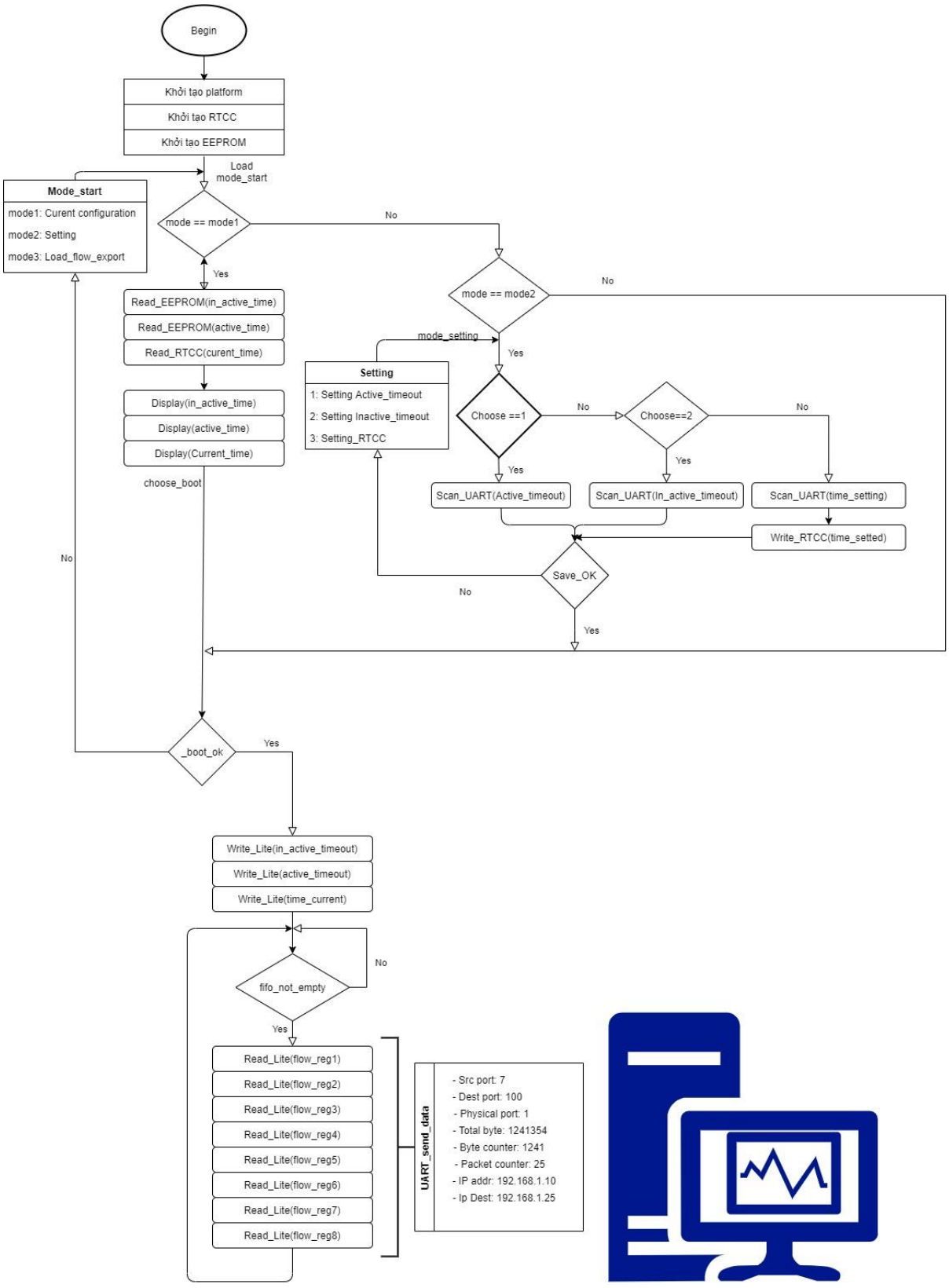
4.4 Xây dựng trình điều khiển cho Microblaze bằng ngôn ngữ C

Chương trình điều khiển của dự án FPGA dựa vào nền tảng – platform đã được xây dựng trên công cụ XPS của Xilinx trước đó, bao gồm: giao diện hệ thống nhúng, bản đồ địa chỉ nền thanh ghi IP và driver, thư viện hỗ trợ của Xilinx và User.

Trình điều khiển xây dựng một chế độ cho phép điều khiển, cấu hình và giao tiếp giữa IP core Netflow V5 với các thiết bị ngoài hệ thống, cụ thể:

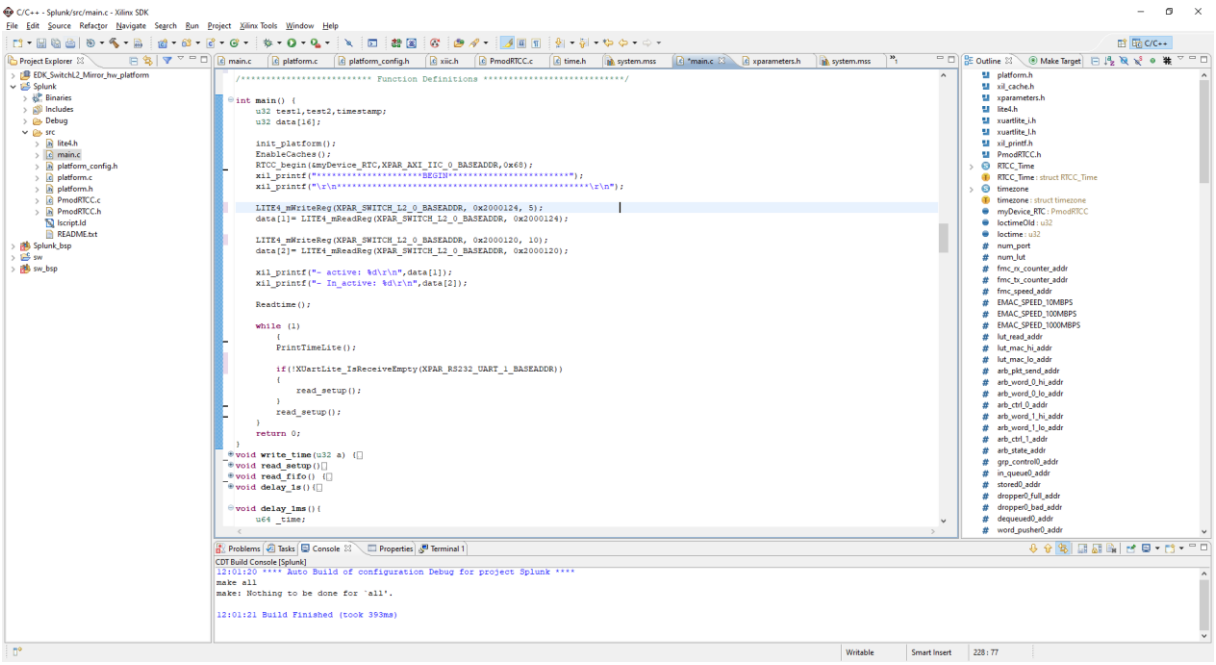
- Giúp hệ thống hoạt động ở thời gian thực bằng việc điều khiển module RTC cung cấp giá trị thời gian liên tục.
- Lưu trữ các thông tin cấu hình hoạt động cần thiết cho hệ thống khi mất nguồn.
- Điều khiển đọc ghi dữ liệu vào ra IP core một cách tuần tự, đảm bảo tính logic cho hệ thống.

Toàn bộ chương trình điều khiển của hệ thống phần cứng Splunk được thể hiện bằng lưu đồ bên dưới dùng để phân tích, thiết kế, phân loại và quản lý công việc.

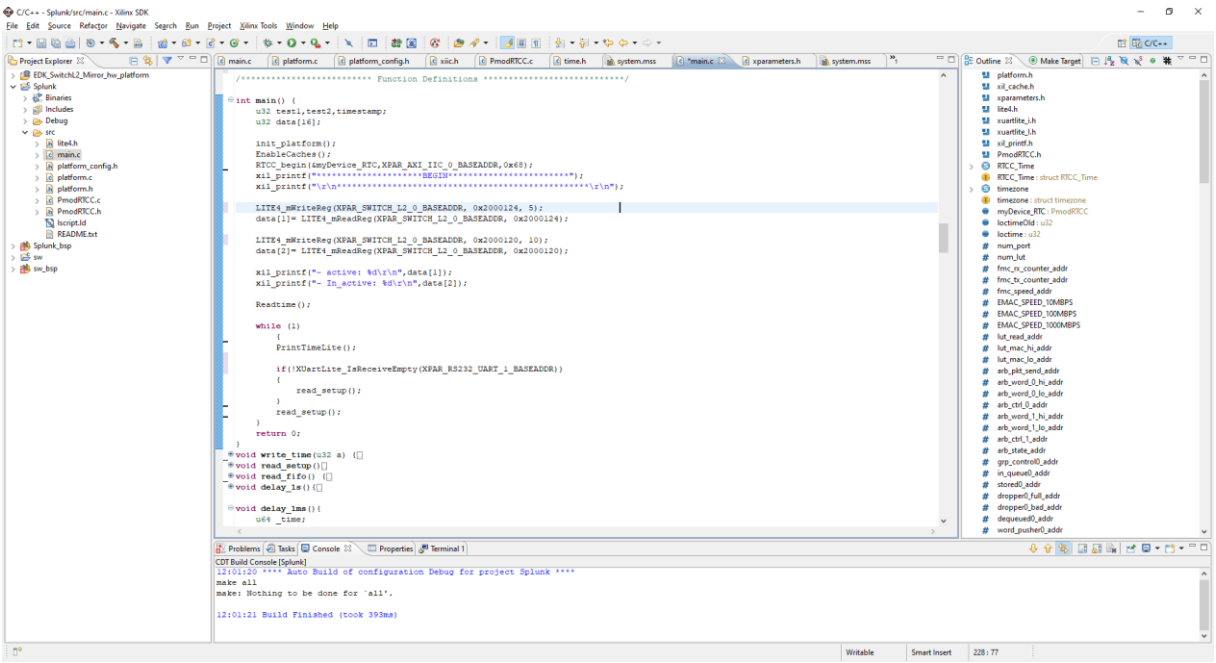


Hình 4.3. Lưu đồ thuật toán trình điều khiển Core NetFlow

Dựa vào lưu đồ thuật toán đã xây dựng, chương trình điều khiển sẽ được thực thi trên công cụ SDK của Xilinx.



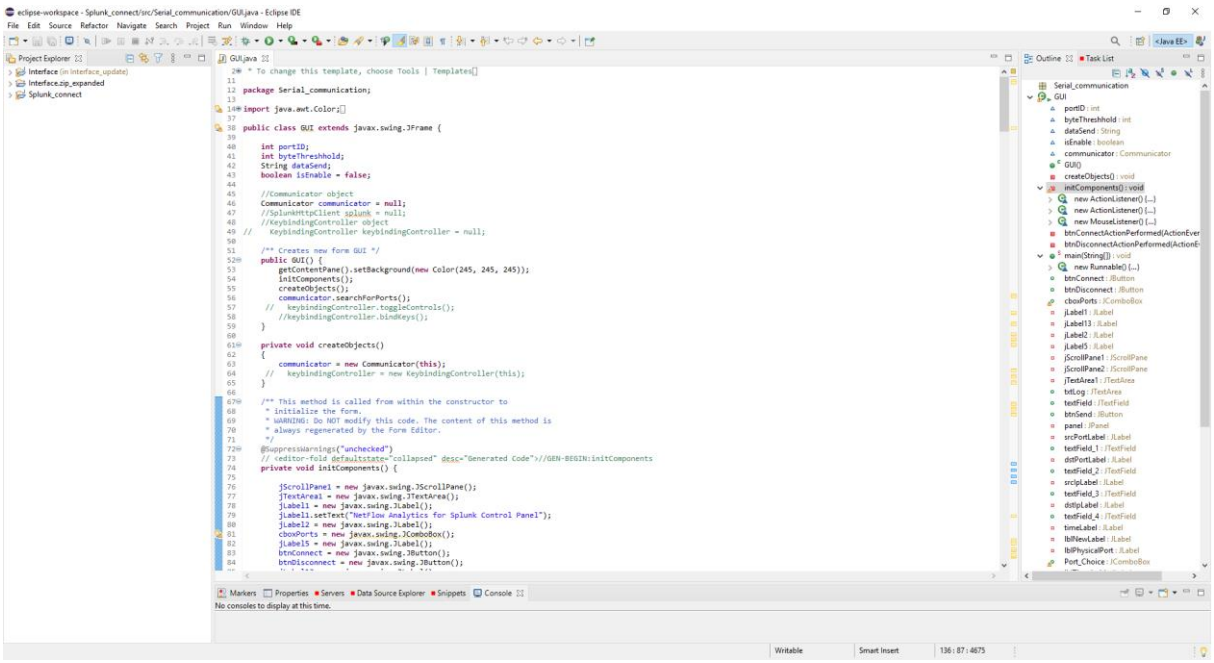
Hình 4.4. Giao diện chính SDK sử dụng ngôn ngữ C để khởi tạo IP core



Hình 4.5. Giao diện chính SDK sử dụng ngôn ngữ C để đọc các thông tin từ IP core

4.5 Thiết kế giao diện người dùng cho hệ thống

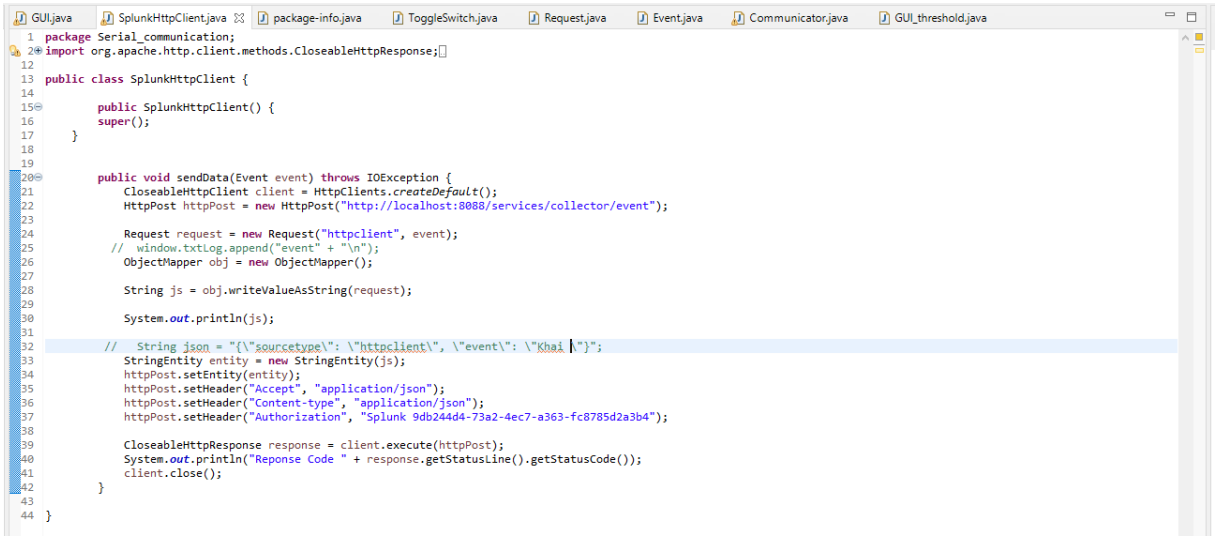
Để thiết kế giao diện với người dùng ta sử dụng phần mềm Eclipse Java Development và ngôn ngữ lập trình Java. Giao diện người dùng là một guide làm việc gồm các class, setting config và hiển thị thông tin dữ liệu. Cho phép nhận, xử lý và truyền data từ các giao thức trao đổi thông tin UART, HTTP.



Hình 4.6. Giao diện chương trình Java guide

Guide Java lấy dữ liệu từ bus uart của kit FPGA, sau đó chuyển kiểu dữ liệu từ dạng string thành mã JSON (Javascript Object Notation). Dữ liệu ở dạng mã JSON sẽ được gửi lên host Splunk theo phương thức Http post:

http://localhost:8088/services/collector/event.

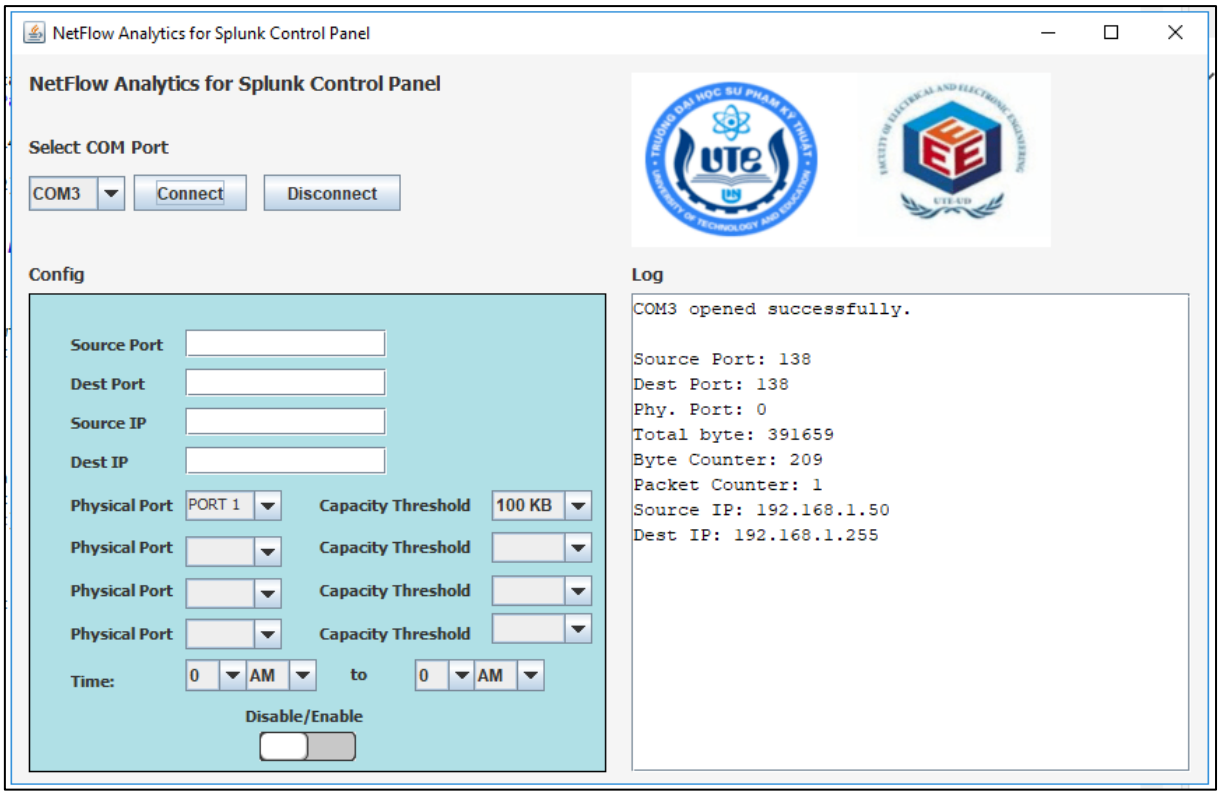


```
GUI.java | SplunkHttpClient.java | package-info.java | ToggleSwitch.java | Request.java | Event.java | Communicator.java | GUI_threshold.java

1  package Serial_communication;
2  import org.apache.http.client.methods.CloseableHttpResponse;
12
13 public class SplunkHttpClient {
14
15     public SplunkHttpClient() {
16     super();
17   }
18
19
20     public void sendData(Event event) throws IOException {
21         CloseableHttpClient client = HttpClients.createDefault();
22         HttpPost httpPost = new HttpPost("http://localhost:8088/services/collector/event");
23
24         Request request = new Request("httpclient", event);
25       //  window.txtLog.append("event" + "\n");
26         ObjectMapper obj = new ObjectMapper();
27
28         String js = obj.writeValueAsString(request);
29
30         System.out.println(js);
31
32      //   String json = "{\"sourcetype\": \"httpclient\", \"event\": \"Khai \"}";
33         StringEntity entity = new StringEntity(js);
34         httpPost.setEntity(entity);
35         httpPost.setHeader("Accept", "application/json");
36         httpPost.setHeader("Content-type", "application/json");
37         httpPost.setHeader("Authorization", "Splunk 9db244d4-73a2-4ec7-a363-fc8785d2a3b4");
38
39         CloseableHttpResponse response = client.execute(httpPost);
40         System.out.println("Reponse Code " + response.getStatusLine().getStatusCode());
41         client.close();
42     }
43
44 }
```

Hình 4.7. Giao diện chính chương trình Java kết nối với host Splunk

Giao diện của guide cho phép người sử dụng lựa chọn và kết nối với cổng COM để đọc dữ liệu từ UART của hệ thống cần giám sát. Ở phía Config bao gồm các chức năng: set ngưỡng hoạt động của từng port vật lý, cài đặt khoảng thời gian hoạt động của ngưỡng port để ngăn chặn các cuộc tấn công mạng.



Hình 4.8. Giao diện của hệ thống giám sát.

Trong hình ở giao diện Log thu thập được các trường thông tin mạng, bao gồm:

- Source Port: 138
- Dest port: 139
- Phy. Port: 0
- Tolal byte: 391659
- Byte Counter: 209
- Packet Counter: 1
- Source IP: 192.168.1.50
- Dest IP: 192.168.1.255

4.6 Kết luận chương

Qua chương này, nhóm tác giả đã thực hiện hai công việc chính đó là:

- Xây dựng chương trình điều khiển C cho MicroBlaze dùng SDK để giao tiếp truyền nhận qua máy tính và dữ liệu đưa lên.
- Thiết kế giao diện người dùng cho hệ thống bằng ngôn ngữ Jav

Chương 5: KIỂM TRA, ĐÁNH GIÁ HỆ THỐNG VÀ THỰC HIỆN GIÁM SÁT HỆ THỐNG MẠNG TẬP TRUNG SPLUNK CHO CÁC DATA CENTER

5.1 Giới thiệu chương:

Trong hầu hết các thiết kế IP Core, nhiệm vụ cuối cùng của người thiết kế là kiểm tra, thẩm định kết quả đã đạt được so với yêu cầu đặt ra ban đầu. Yêu cầu đặt ra ban đầu phải đảm bảo đúng thuật toán ban đầu, kế đến là Core ưu tiên về tốc độ và băng thông xử lý, hoặc yêu cầu về tài nguyên sử dụng. Tuỳ theo ứng dụng mà yêu cầu nào sẽ được đặt ưu tiên trong quá trình thiết kế. Ví dụ trong các thiết bị truyền dẫn, tổng đài thì ưu tiên về tốc độ và băng thông, trong thiết bị cầm tay, ứng dụng cá nhân thì ưu tiên về tài nguyên để giảm giá thành. Do vậy khi đánh giá kết quả đạt được là đánh giá dựa trên mục tiêu mà IP Core mà được ứng dụng.

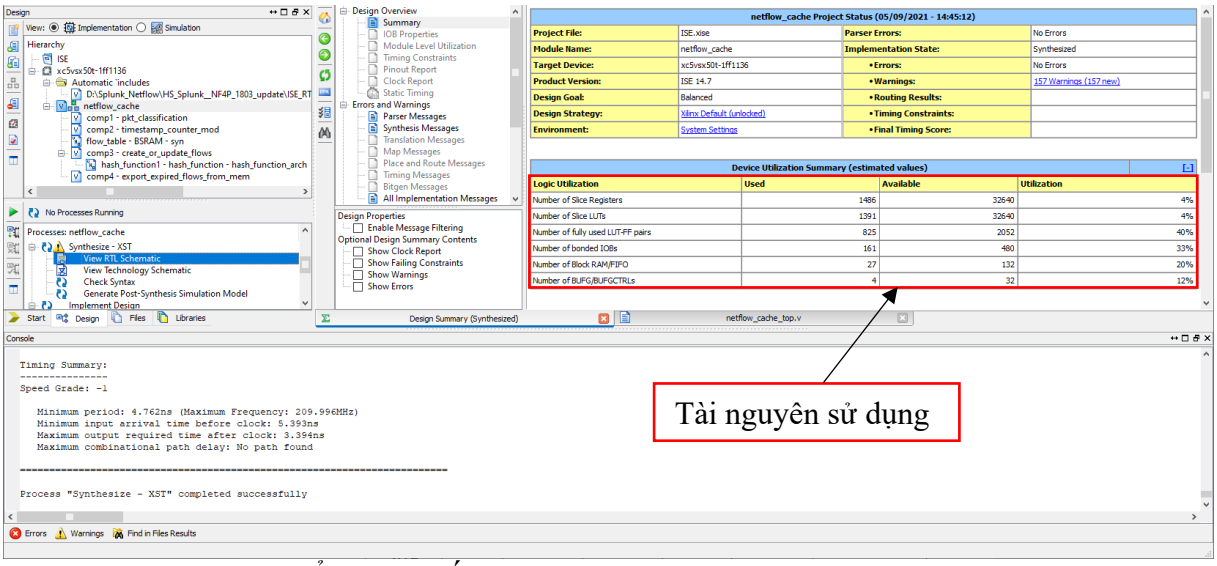
Trong đề tài này, IP Core NetFlow V5 được nhúng vào hệ thống nhúng FPGA, ghép nối với các thiết bị ngoài. Trong chương này, tác giả sẽ phân tích các kết quả đã đạt được dựa trên kết quả tổng hợp của trình tổng hợp ISE 14.7 của Xilinx.

5.2 Phân tích các thông số của kết quả tổng hợp

Tổng hợp là kết quả quá trình biên dịch hoặc chuyển đổi ngôn ngữ mô tả phần cứng Verilog vào trong mức cổng. Căn cứ vào cấu trúc và quy mô phần cứng được mô tả trong chương trình Verilog, trình tổng hợp ISE 14.7 sẽ cho các kết quả sau khi tổng hợp. Các thông số quan trọng đánh giá kết quả quá trình tổng hợp gồm:

- Tài nguyên sử dụng
- Độ trễ cực đại
- Tần số hoạt động
- Bộ nhớ được sử dụng

Tài nguyên sử dụng: là số Logic Block mà IP Core cần sử dụng. Khái niệm Logic Block tuỳ thuộc vào mỗi hãng sản xuất định nghĩa, ví dụ của hãng Altera gọi LE (Logic Element), hãng Xilinx gọi là LC (Logic Cell hoặc Slice). Mỗi Logic Block bao gồm các thành phần chính là LUT và Flip Flop, số lượng LUT và Flip Flop trong Logic Block tuỳ thuộc từng hãng sản xuất và dòng sản phẩm

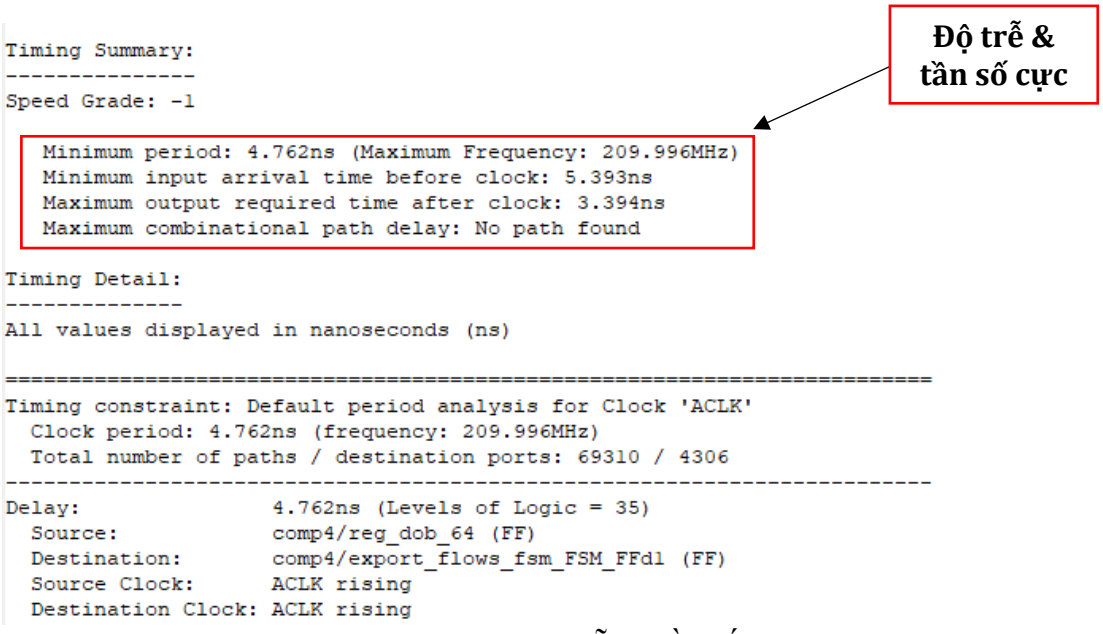


Hình 5.1 Tổng hợp kết quả cuả Core NetFlow ở mức ***Synthesise***

Độ trễ cực đại: là thời gian trễ lớn nhất trong các độ trễ mà một khối logic tính toán từ một tính hiệu vào đến một tín hiệu ra. Độ trễ càng lớn thì tốc độ xử lý của Core càng chậm

Tần số cực đại: là tần số cực đại Clock của hệ thống cung cấp cho Core mà Core hoạt động không bị lỗi

Thông số tài nguyên



Hình 5.2 Báo cáo độ trễ và tần số cực đại

5.3 Kiểm tra hệ thống trên board ML605 tại môi trường thực tế:

5.3.1 Xây dựng kịch bản kiểm thử:

Nhóm tác giả sẽ xây dựng một kịch bản để so sánh giải pháp Splunk Software hiện nay và khi sử dụng giải pháp Splunk bằng phần cứng FPGA nhúng IPCORE NetFlow V5 của nhóm *(Sử dụng virsion: Splunk Enterprise).*

5.3.2 Đối tượng chính:

- Giải pháp Splunk Software: Setup một PC intel core I5 với 4 card mạng. Phần mềm Splunk Enterprise index data từ 4 card mạng của PC.

- Giải pháp tăng tốc Splunk bằng phần cứng FPGA: gồm Ipcore Netflow_V5 4 port tích hợp hệ thống và nhúng trên board Ml605. Sử dụng trình điều khiển để đọc các trường thông tin đã được Netflow trích xuất. Phần mềm Splunk index data từ board ML605 bằng giao tiếp UART.

5.3.3 Mục tiêu:

- Thể hiện các điểm hạn chế của giải pháp Splunk Software: Data index có kích thước lớn, dung lượng free có giới hạn 500Mb/date. Phần mềm Splunk chụp gói tin không có khả năng chụp đủ gói tin khi đẩy gói ở băng thông tốc độ cao (Max speed của tool PackEth).

- Giới thiệu các tính năng của giải pháp tăng tốc Splunk bằng phần cứng FPGA: khắc phục các điểm hạn chế của giải pháp phần mềm; thể hiện được tổng hợp lưu lượng byte từ 4 port đưa lên Sever Plunk, biểu đồ lưu lượng tăng tuyến tính và thể hiện liên tục; cho phép xác định được chính xác lưu lượng byte, lưu lượng gói của data trên biểu đồ giám sát theo từng port và từng địa chỉ ip của thiết bị.

5.3.4 Thiết bị sử dụng:

- 2 PC monitor : đã cài phần mềm Splunk Enterprise.
- 8 PC đẩy gói đã cài tool PackEth cho cả hai hệ thống.
- Board FPGA ML605 đã nạp sẵn Netflow_V5 4 port.

5.3.4.1 Nội dung thực hiện:

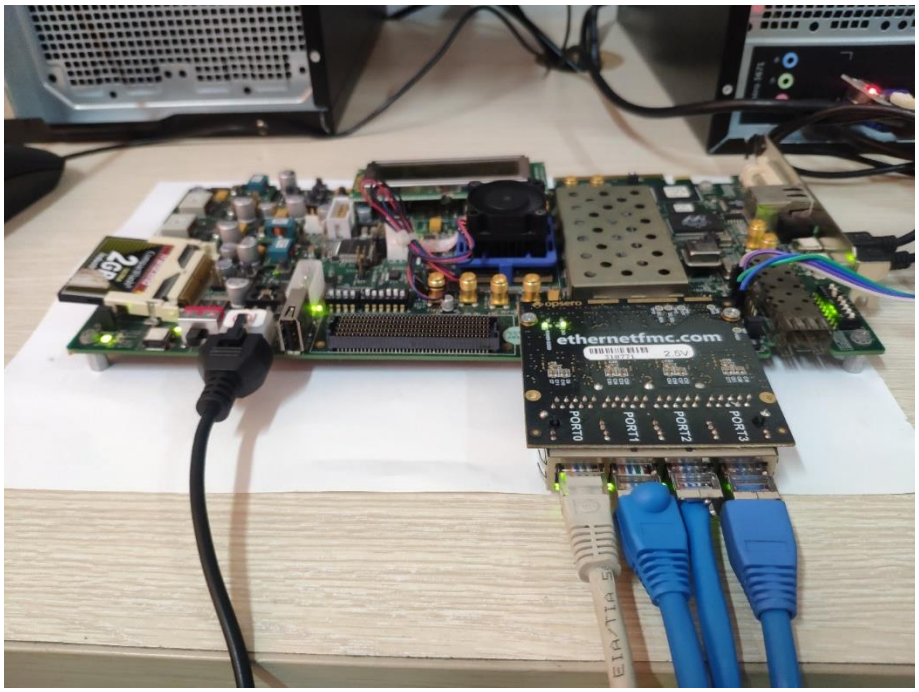
1. Demo hệ thống 1: Gồm 1 PC intel core i5 setup 4 card mạng, sử dụng 4 PC khác có tool đẩy gói PackEth để đẩy gói ở 2 trường hợp, trên sever Splunk dùng hàm vẽ biểu đồ thể hiện tổng lưu lượng data của 4 card mạng. Thực hiện đẩy gói với tốc độ cao.

2- Demo hệ thống 2: Setup Board ML605 đã nhúng Ip core Netflow và 4 PC dùng tool đẩy gói PackEth đẩy gói vào 4 port FMC, chạy Guide Java để chuyển data dạng string từ board FPGA sang dạng mã Json; Splunk index data đã được decode lên sever, thống kê và giám sát. Thực hiện đẩy gói tốc độ cao như demo hệ thống thứ nhất.

5.3.4.2 Tiến hành Demo:

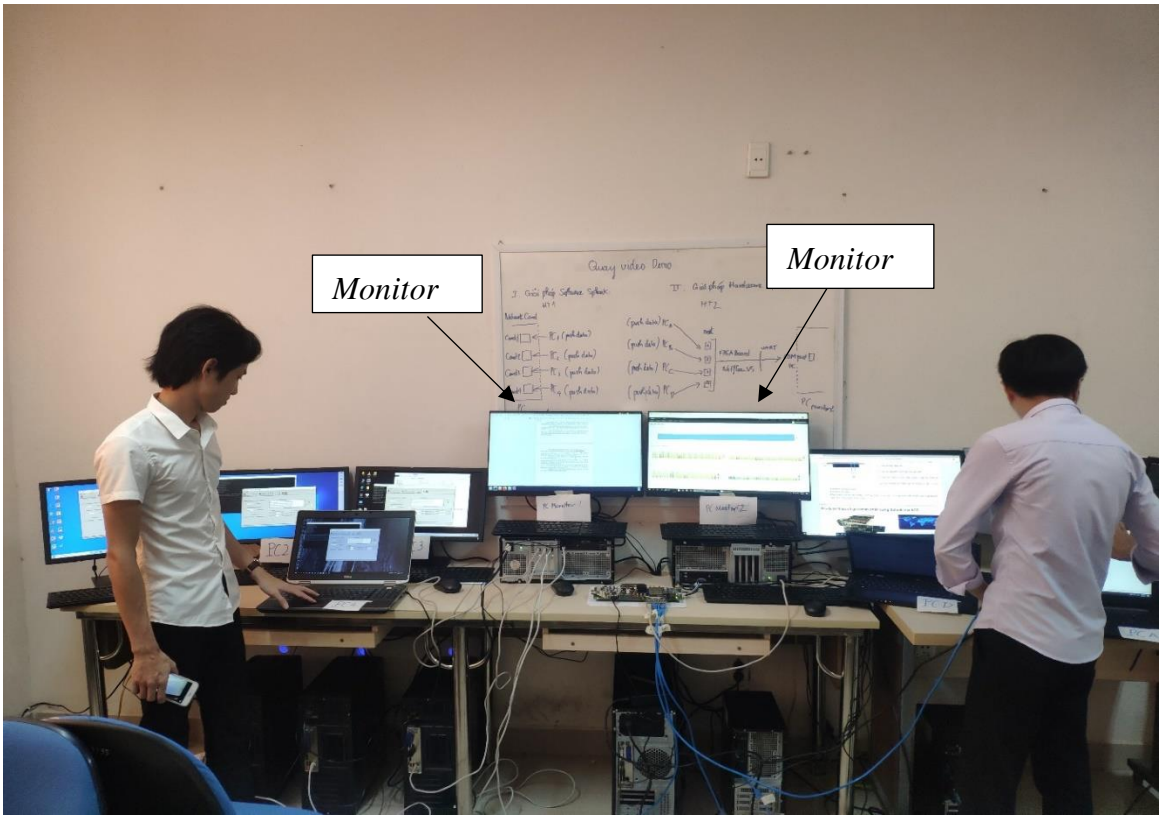
Nạp chương trình cho board ML605

- Kết nối board ML605, sử dụng công cụ SDK phần mềm của Xilinx để nạp chương trình xuống kit.



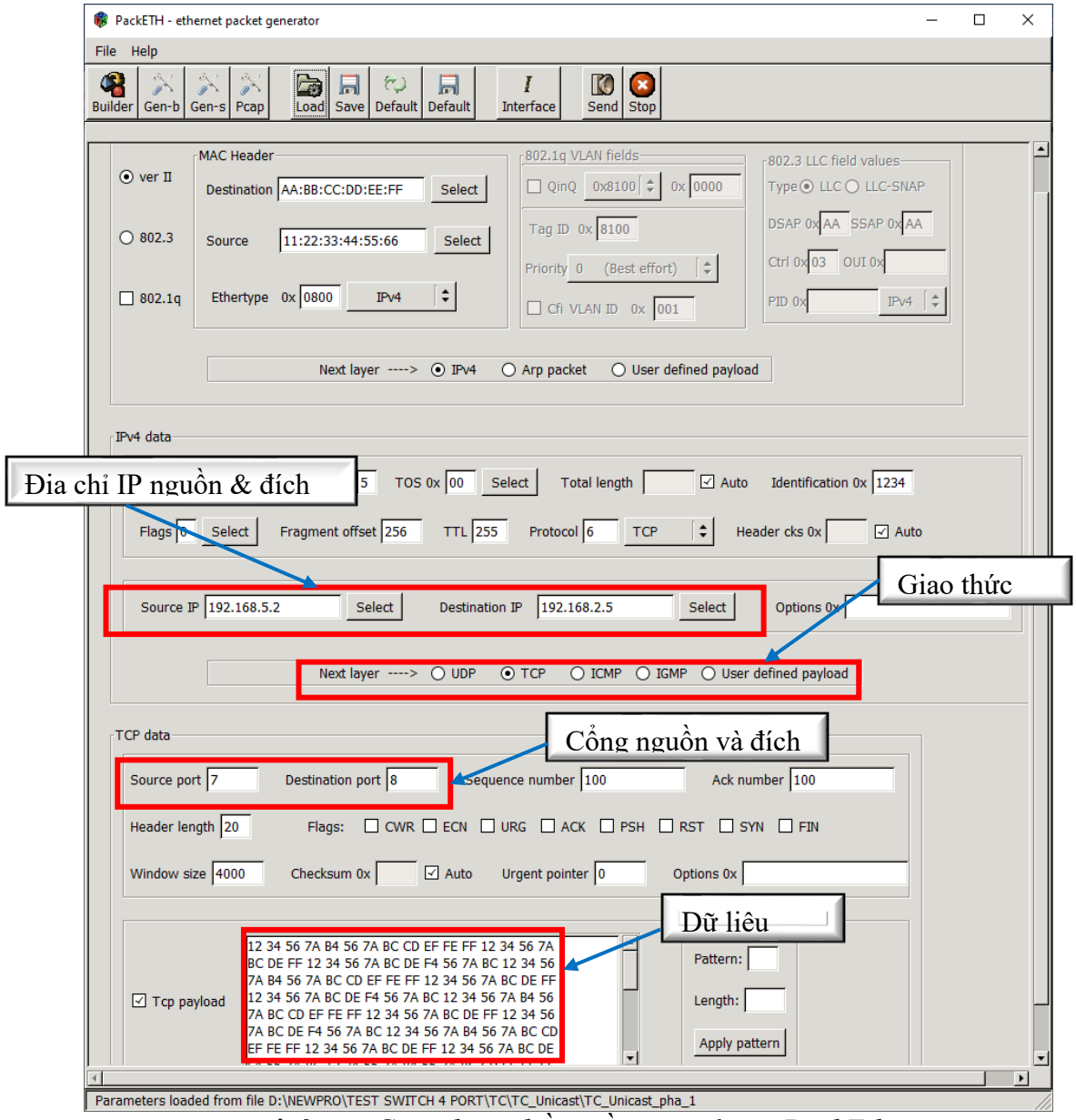
Hình 5.3 Nạp chương trình cho board ML605

Bước 1: Tiến hành setup hệ thống DEMO 1 & 2:



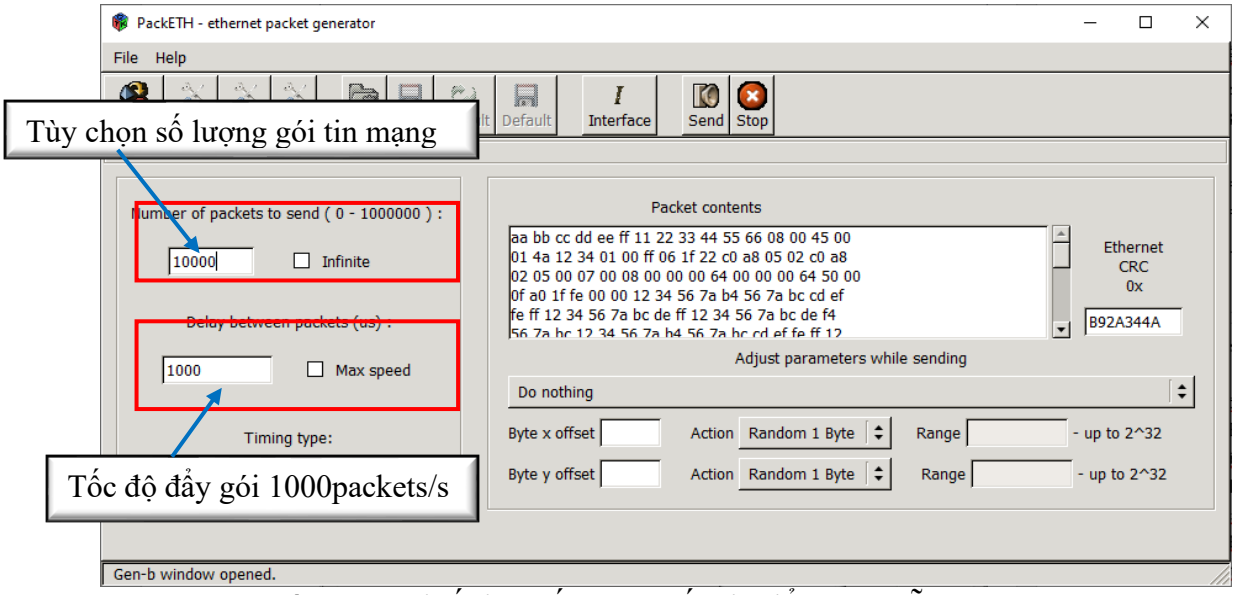
Hình 5.4 Setup hệ thống để kiểm tra

Bước 2: Tạo gói tin internet sử dụng công cụ PackEth trên 4 PC kết nối với 4 card mạng của PC Monitor 1. Và tạo trên 4 PC kết nối với các port Ethernet trên Board ML605.



Hình 5.5 Giao diện phần mềm tạo gói tin PackEth

Sinh viên thực hiện : Nguyễn Duy Hà Sơn – Đặng Sỹ Phi Hùng
Người hướng dẫn : TS. Trần Hoàng Vũ

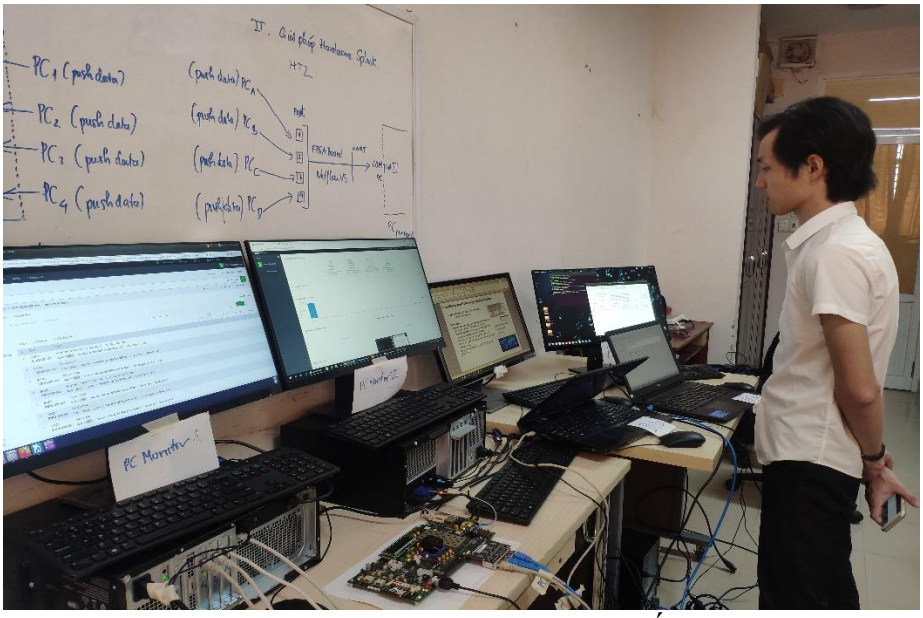


Hình 5.5 Thiết lập số gói và tốc độ đẩy của mỗi gói tin

Bước 3: Tiến hành đẩy gói tin với tốc độ 1000packets/s liên tục vào 2 hệ thống:

Hình 5.6 Thực hiện dùng 4 PC đẩy vào các gói tin liên tục để kiểm tra hệ thống.

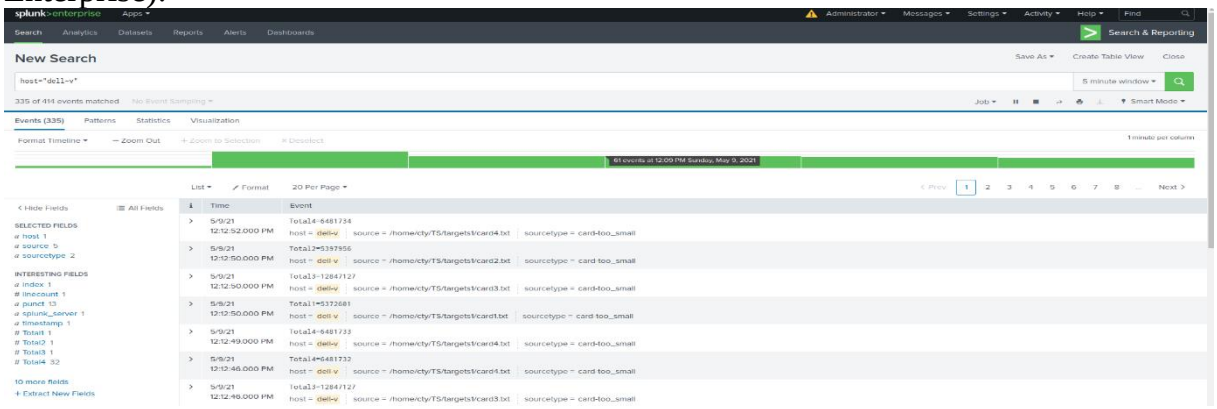
Bước 4: Quan sát và đánh giá kết quả:



Hình 5.7 Giám sát và đánh giá kết quả

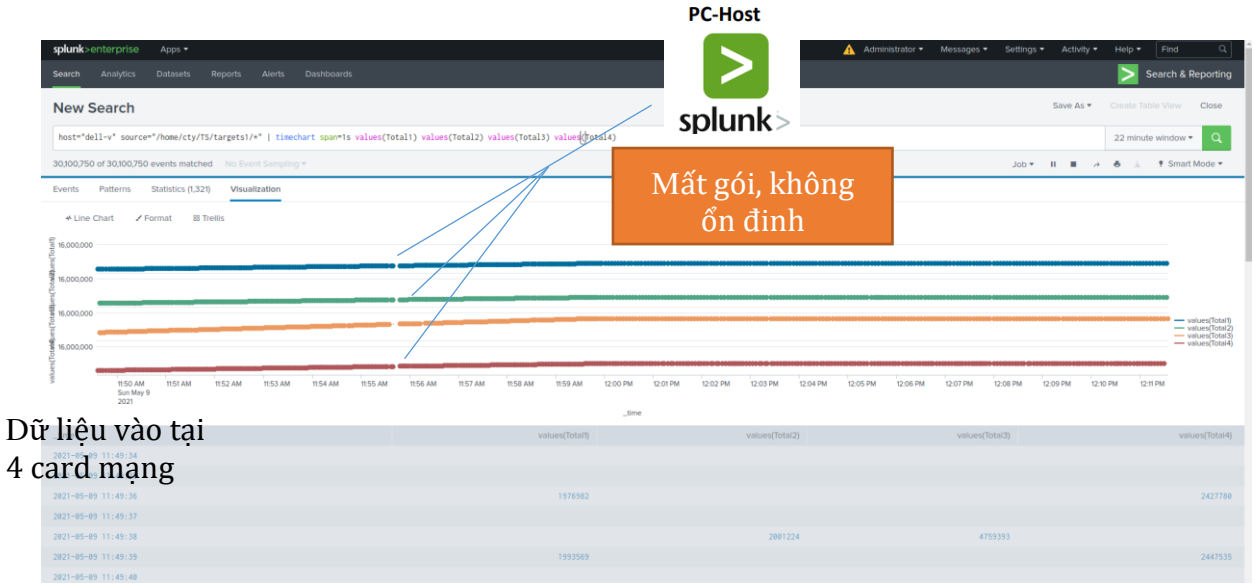
Kết quả:

- Đối với hệ thống 1: Sử dụng trực tiếp phần mềm Splunk Software (Splunk Enterprise):

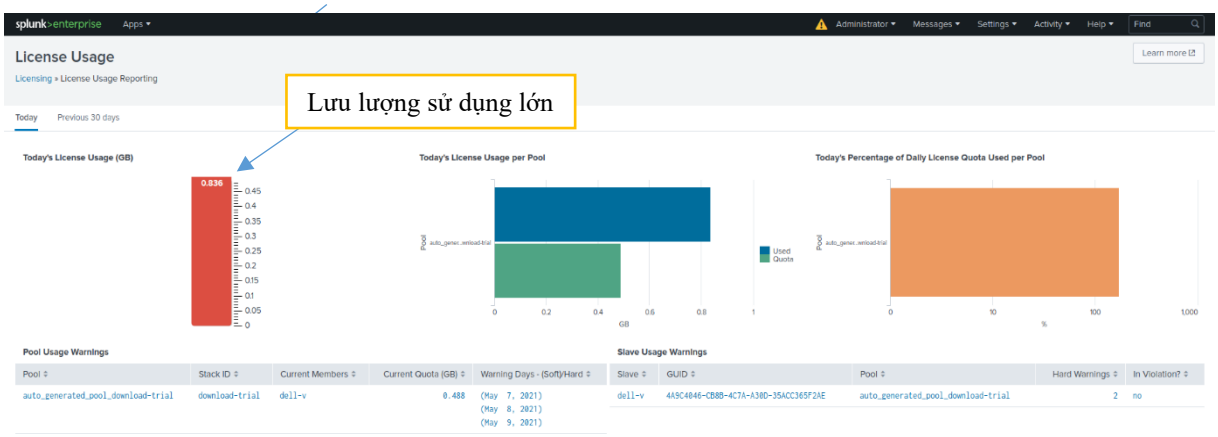


Hình 5.8 Các gói tin từ 4 card mạng đã được đẩy lên hostSplunk

Sinh viên thực hiện : Nguyễn Duy Hà Sơn – Đặng Sỹ Phi Hùng
Người hướng dẫn : TS. Trần Hoàng Vũ



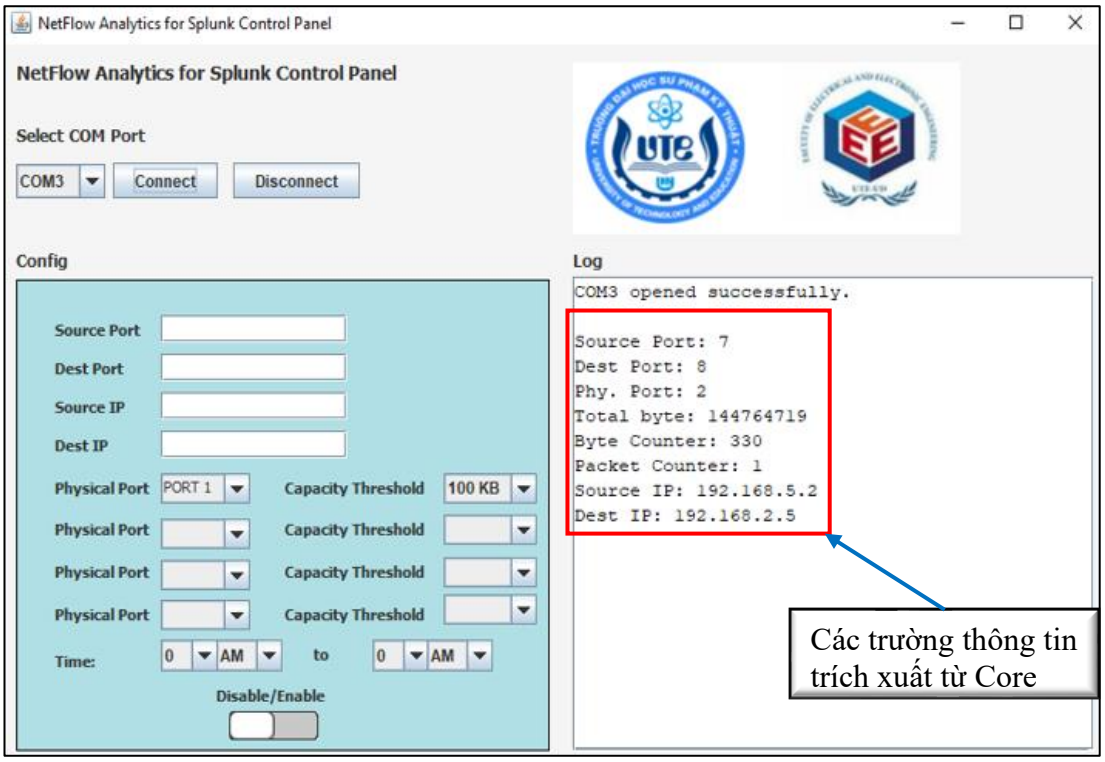
Hình 5.9 Biểu đồ mô tả tổng dung lượng được đẩy lên theo từng card mạng



Hình 5.10 Biểu đồ mô tả lưu lượng sử dụng trong ngày.

- Đối với hệ thống 2: Sử dụng kết hợp Splunk software và phần cứng FPGA nhúng IP Core NetFlow V5.

+ Đã trích xuất đúng các trường thông tin cần thiết của gói tin sử dụng IPCore NetFlow V5 nhúng trên FPGA board ML605.



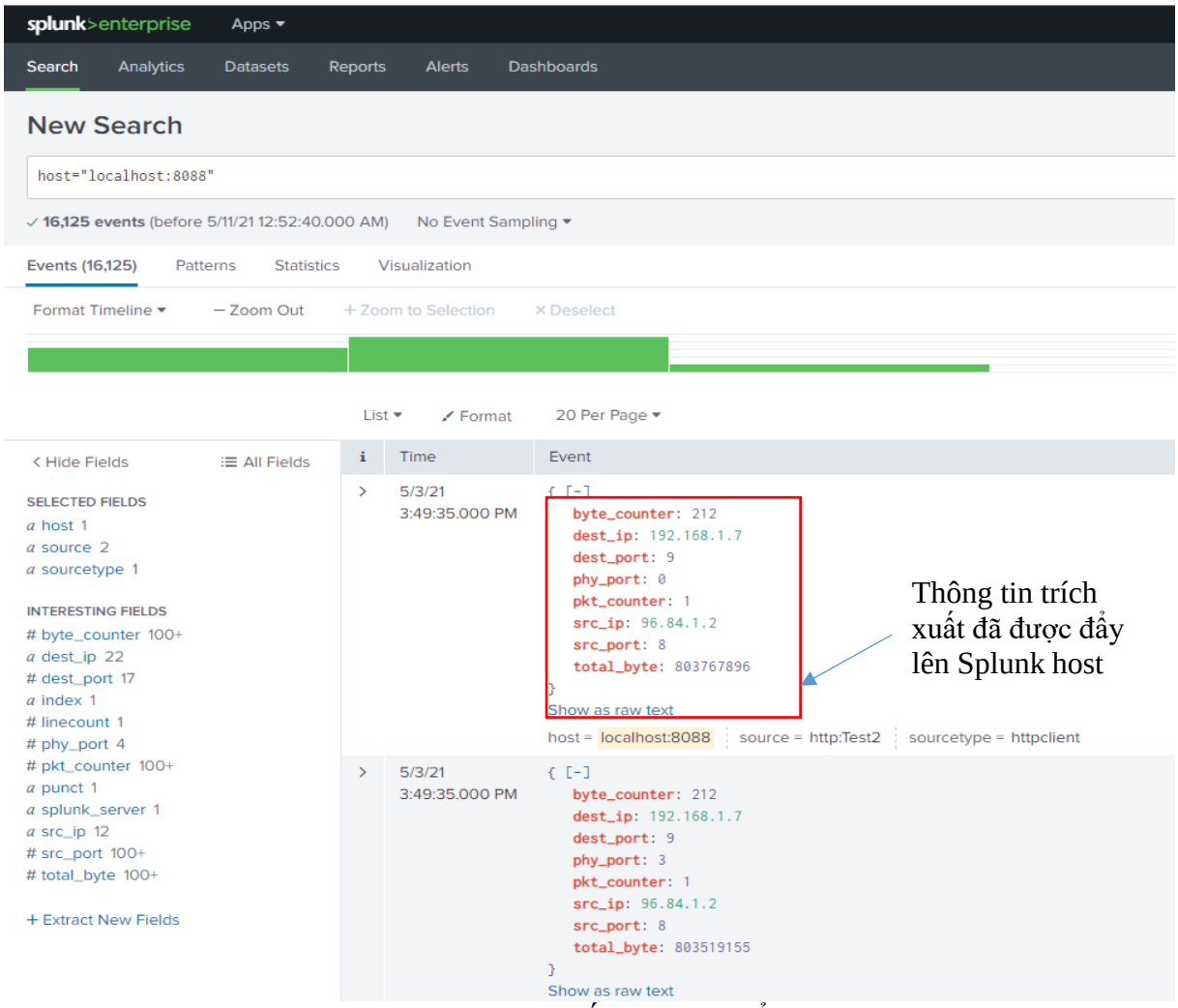
Hình 5.11 Giao diện, dữ liệu đọc lên từ UART hiển thị các thông tin trích xuất

- Truy cập vào **Splunk Host** để tiến hành quan sát các thông tin được đọc lên từ **java** và vẽ **biểu đồ giá trị** theo các thông tin đó.

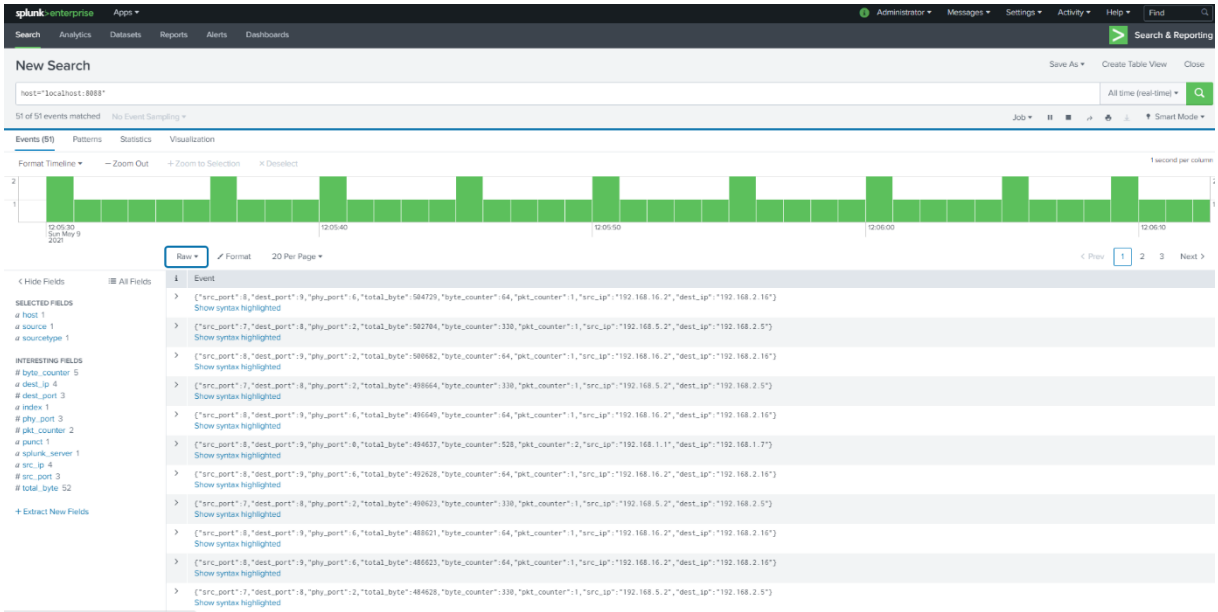
Hình 5.12 Giao diện Splunk host

- Vào Search & Reporting: nhập vào Research:
host="localhost:8088"

Kết quả:



Hình 5.13 Các thông trích xuất đã được đẩy lên Splunk host

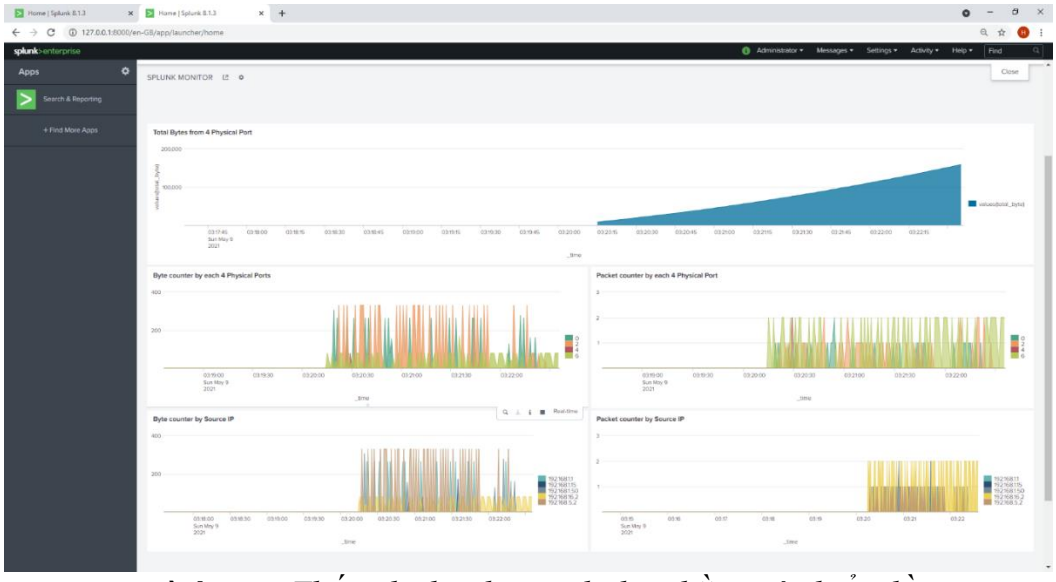


Hình 5.14 Thống kê các gói tin theo thời gian thực trên Splunk host

- Tiến hành dùng các hàm vẽ để hiển thị biểu đồ giá trị theo các thông tin đó bằng cách nhập các hàm vẽ vào **New Reseacrh:**

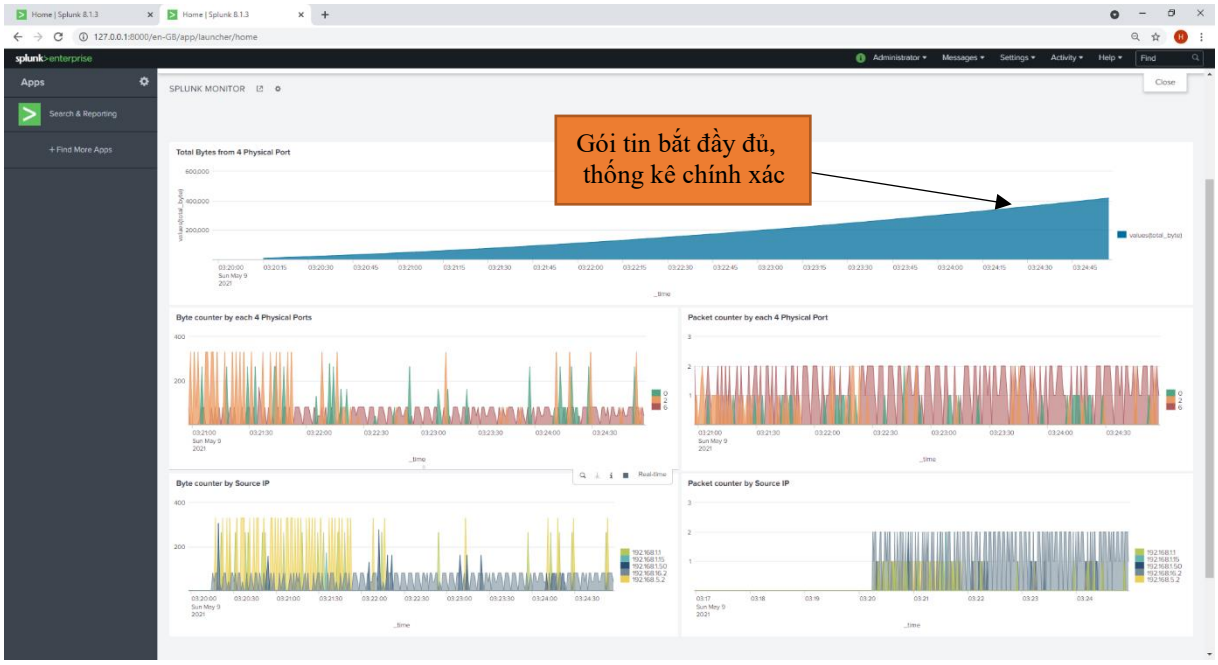
1. Vẽ giá trị **Total byte** (Tổng các byte đầu vào tại 4 port)
   Host= "localhost:8088"|timechart span=1s values(Total_byte)
2. Vẽ **Packet_counter** theo các port **phy_port:**
   Host= "localhost:8088"|timechart span=1s values(Packet_counter) by phy_port
3. Vẽ **Byte_counter** theo các port **phy_port:**
   Host= "localhost:8088"|timechart span=1s values(byte_counter) by phy_port
4. Vẽ **Packet_counter** theo các port **IP_Source:**
   Host= "localhost:8088"|timechart span=1s values(byte_counter) by ip_src
5. Vẽ **Byte_counter** theo các port **IP_Source:**
   Host= "localhost:8088"|timechart span=1s values(byte_counter) by ip_src

Hình 5.15 Thống kê lưu lượng dữ liệu bằng các biểu đồ

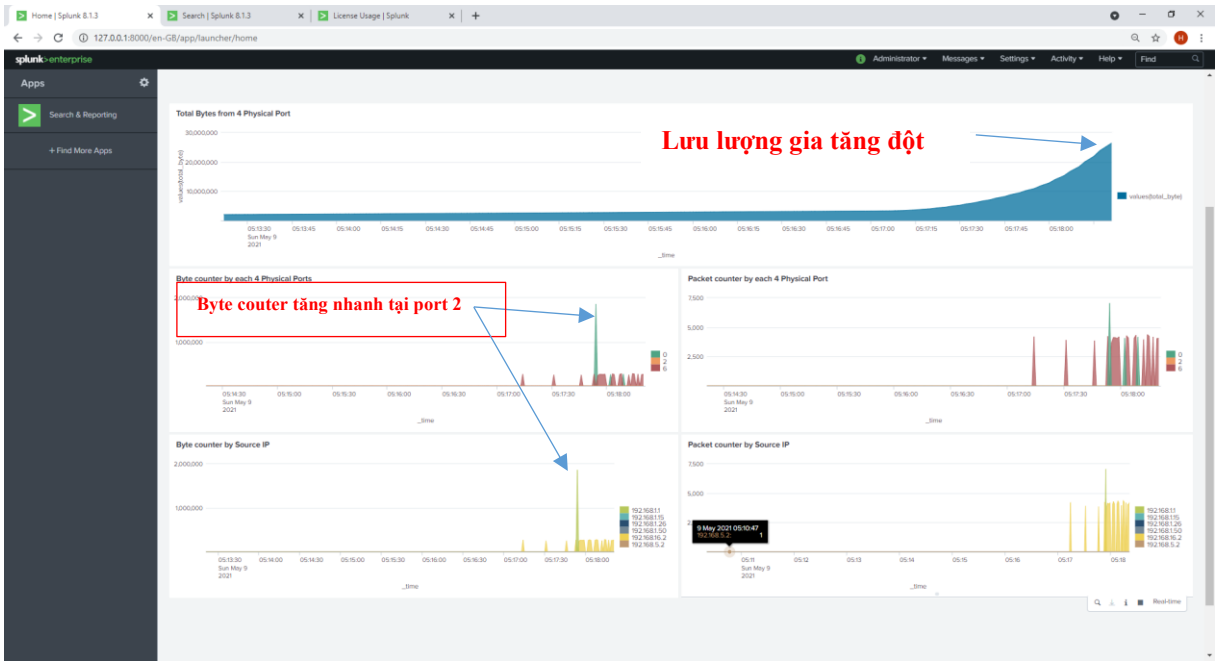


Hình 5.16 Thống kê lưu lượng dữ liệu bằng các biểu đồ

* Quan sát trên biểu đồ ta thấy được lưu lượng tại các port tăng đều nhau.

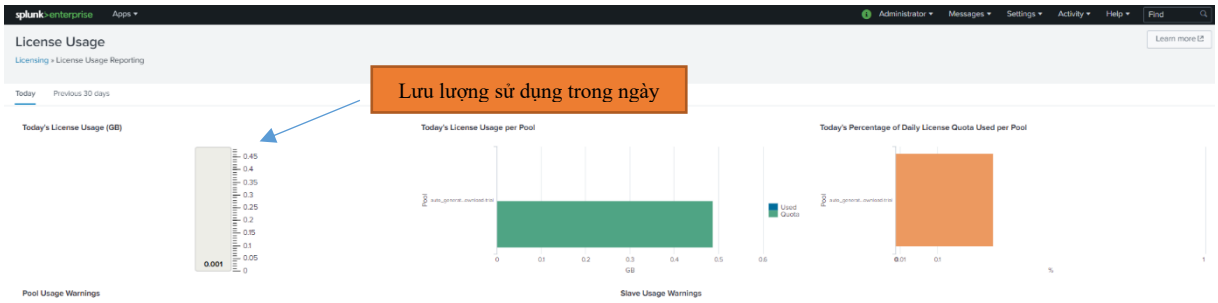


Hình 5.17 Thống kê lưu lượng dữ liệu bằng các biểu đồ



Hình 5.18 Sự gia tăng lưu lượng đột ngột

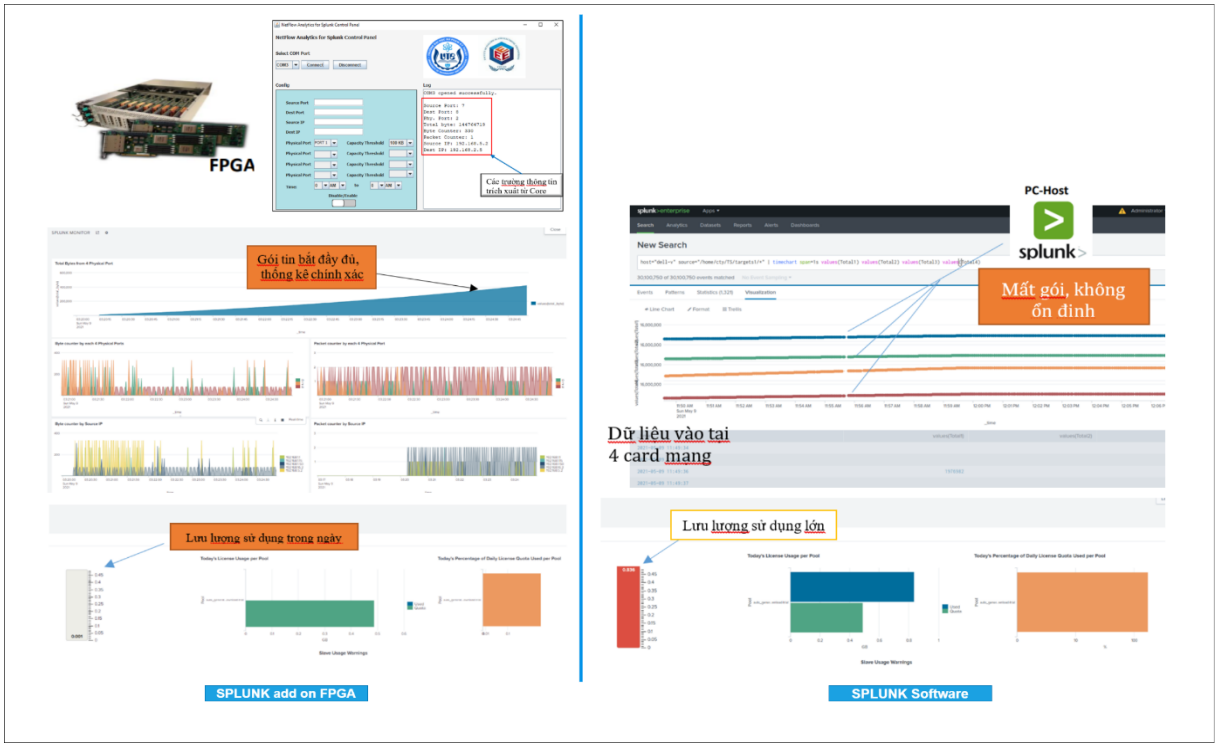
Sinh viên thực hiện : Nguyễn Duy Hà Sơn – Đặng Sỹ Phi Hùng
Người hướng dẫn : TS. Trần Hoàng Vũ



Hình 5.19 Lưu lượng sử dụng trong ngày

Đánh giá và kết luận về kết quả của hai hệ thống:

- Dựa vào kết quả đã test trên chúng tôi xin đưa ra kết luận đánh giá:



Hình 5.20 So sánh kết quả Splunk kết hợp FPGA và Splunk Software

- **Hệ thống sử dụng Splunk Software** trực tiếp để bắt các gói tin từ dữ liệu tại các card mạng trong hệ thống mạng, một gói tin khi chưa được phân tích có thể lên đến **20-50Kbyte**, tốc độ càng tăng thì lưu lượng tải lên nó sẽ càng lớn, 1 ngày có thể lên đến hàng GB. Bản quyền Splunk Software **2000usd/năm/GB mỗi ngày**, về Việt Nam tương đương 72 – 180 triệu đồng, nếu càng sử dụng nhiều lưu lượng tăng lên 100GB thì sẽ tăng 600usd cho mỗi GB tăng (dùng 5G/ngày = 8.100 usd; 20G/ngày = 24.000usd.

- Để giải quyết vấn đề trên hãy xem hệ thống của chúng tôi khi sử dụng **Splunk kết hợp với FPGA với IPCORE NetFlow V5**: bắt các gói tin và đẩy lên Host đầy đủ chính xác,

dữ liệu được cập nhật liên tục theo **realtime**. Đối với những thông tin ta đưa lên chúng ta có thể linh hoạt giám sát theo từng port, từng IP... Có giao diện quản lý dễ dàng giám sát và quản lý để ngăn chặn các tình trạng xấu như ngẽn mạch, lưu lượng tăng vọt,… Với công nghệ FPGA chúng ta có thể **xử lý đồng thời nhiều luồng** kết nối với các tốc độ Ethernet 1G, 2.5G,…Phân tích đồng thời **hang triệu Data Flow** nhưng lưu lượng chúng ta đưa lên để giám sát rất nhỏ (**20byte/packet**), có thể sử dụng gần như vô hạn không cần mua bản quyền – chỉ cần dùng phiên bản miễn phí là đủ. An toàn, làm chủ được công nghệ, giá thành hợp lý có thể cạnh tranh với Splunk Software.

5.4 Đánh giá kết quả:

Đề tài đặt ra nhiệm vụ: **"Nghiên cứu và thiết kế IPCORE NetFlow V5 trên nền tảng công nghệ FPGA để phân tích và giám sát mạng"**, qua thời gian thực hiện với sự nỗ lực bản thân các thành viên trong nhóm và đặc biệt được sự giúp đỡ tận tình của thầy giáo hướng dẫn, kết quả đã đạt so với yêu cầu của đề tài, các kết quả mô phỏng đúng theo tài liệu kỹ thuật NetFlow V5. Các thông số trong kỹ thuật thiết kế như: tốc độ xử lý, tài nguyên sử dụng đều có kết quả đạt yêu cầu để đưa xuống hệ thống nhúng FPGA. Đây là một hướng phát triển mới của các trường Đại học và nhu cầu cấp thiết về nguồn nhân lực trên thị trường công nghệ cao hiện nay. Hệ thống bao gồm vi xử lý MicroBlaze, các thiết bị ngoại vi, việc nhúng này tuân thủ theo chuẩn giao tiếp của PLB, đồng thời phát triển các Firmwave điều khiển hệ thống. Trong quá trình phát triển hệ thống nhúng, bản thân tác giả gặp nhiều khó khăn vì đây là một lĩnh vực hoàn toàn mới mẻ, trước đây chưa từng được làm hay được biết về hệ thống nhúng FPGA và tài liệu hướng dẫn, tham khảo hầu như, chỉ có tài liệu của nhà sản xuất. Tuy nhiên như đã nói trên, với sự nỗ lực bản thân và định hướng tích cực của thầy giáo hướng dẫn nên việc thực hiện NetFlow V5 IPCORE trên hệ thống nhúng FPGA đã hoàn tất.

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Với những kết quả đạt được có thể khẳng định việc thiết kế và thực hiện NetFlow V5 trên nền FPGA đã đáp ứng các yêu cầu của đề tài đặt ra. Cụ thể, đề tài đã đạt những kết quả sau:

- Thực hiện thuật toán NetFlow V5 để thiết kế IP Core NetFlow trên nền tảng FPGA.

- Thực việc việc nhúng NetFlow V5 IPCore và tạo các File Bitstream để nạp hệ thống hoàn chỉnh.

- Kết hợp Splunk Sotfware để thực hiện giám sát mạng.

Chúng tôi sẽ xây dựng hướng phát triển trong thời gian đến như sau:

- Hiện tại bộ vi xử lý mềm trong dự án này là MicroBlaze được hổ trợ trong Virtex 6 ML605 Board FPGA. Một số mới nhất trong Xilinx FPGA sử dụng bộ xử lý cứng như RAM Cortex. Chúng tôi có thể cài đặt Petalinux trên một số FPGA tiên tiến của Xilinx – Zynq 7000 để thực hiện nhúng Linux và Buid Driver cho dự án.

- Phát triển chương trình điều khiển cho những ứng dụng hay nhiệm vụ nâng cao và phát triển thêm giao diện quả lý.

- Phát triển version V5 của NetFlow lên các version 7, version 9. Tối thiểu hoá các chân vào/ra để tương tích các họ FPGA như Xilinx hoặc Altera

- Xây dựng các NetFlow IPCore phục vụ cho các ứng dụng chuyên biệt như trong hệ thống truyền dẫn, phân tích các giao thức khác mới hơn, hoặc phát triển để ít tốn tiêu hao nguồn…

Hiện nay trên thế giới hình thành những công ty chuyên thiết kế các Core theo yêu cầu của khách hàng. Ngay những hãng bán dẫn lớn vẫn không thể bao quát hết (*vì nhiều lý do*) tất cả các thành phần trong sản phẩm của mình, phải mua Core từ các công ty khác để nhúng vào trong sản phẩm. Việc thiết kế sản phẩm Core không đòi hỏi đầu tư ban đầu lớn, chủ yếu là đầu tư chất xám, con người mà cho hiệu quả kinh tế rất cao

Đây là một hướng phát triển ngành công nghệ cao: thiết kế vi mạch và hệ thống nhúng mà thế giới nói chung và Việt Nam nói riêng rất cần đế

TÀI LIỆU THAM KHẢO

Tiếng việt

[1] Nguyễn Trọng Hải (2005). *Tóm tắt bài giảng thiết kế hệ thống số - Phần Verilog,* Đại học kỹ thuật công nghệ- Thành phố Hồ Chí Minh

[2] Tống Văn On (2005)**.** *Nguyên lý mạch tích hợp T1 - ASIC Lập trình được*, NXB Lao động- Xã hội, Hà Nội.

[3] Huỳnh Viết Thắng (2007). *Nghiên cứu Network-on-chip và thực hiện Network-on-chip trên nền FPGA*, Luận văn thạc sĩ, Đại Học Bách Khoa Đà Nẵng.

[4] Ngô Đức Minh, Phạm Quốc Cường, Trần Ngọc Thịnh, *An Efficient High-Throughput and Low-Latency SYN Flood Defender for High-Speed Networks*.

[5] Võ Thành Văn (2017). *Thiết kế và thực thi lõi IP phân luồng dữ liệu trên FPGA*, Luận văn thạc sĩ, Trường Đại Học Bách Khoa Đà Nẵng.

Tiếng Anh

[6] National Institute of Standards and Technology (NIST). *Secure Hash Standard, Federal Information Processing Standards Publication 180- 2 (FIPS PUB 180- 2*, 2002 August 1

[7] "*Creating Your First MicroBlaze Design with the Spartan-6 LX16 Evaluation Kit*"

[8] D. Moore, C. Shannon, D. J. Brown, G. M. Voelker, and S. Savage(2006). *Inferring internet denial-of-service activity, ACM Transactions on Computer Systems*, vol. 24, no. 2, pp. 115–139.

[9] A. G¨unther and C. Hoene. *Measuring round trip times to determine the distance between wlan nodes in Proceedings of the International conference on research in networking, vol. 3462 of Lecture Notes in Computer Science*, pp. 768–779, Springer, Berlin, Germany.

[10] Marco Forconesi (2013). *Accurate and Flexible Flow-Based Monitoring for High-Speed Networks*, vol. 24, pp. 28–32.

Các trang Web

[11] *ML 605 Hardware User Guide :*
https://www.xilinx.com/support/documentation/boards_and_kits/ug534.pdf

[12] Xilinx. *AXI4-Lite IP Interface(IPIF):*
https://www.xilinx.com/products/intellectual-property/axi_lite_ipif.html

[13] Xilinx. *AXI4-Stream Infrastructure IP Suite v3.0:*
https://www.xilinx.com/support/documentation/ip_documentation/axis_infrastructure_ip_suite/v11/pg085-axi4stream-infrastructure.pdf

[14] Xilinx. *AXI IIC Bus Interface v2.0:*
https://www.xilinx.com/support/documentation/ip_documentation/axi_iic/v2_0/pg090-axi-iic.pdf

http://arduino.vn/bai-viet/369-giao-tiep-i2c-va-su-dung-module-realtime-clock-ds1307
[15] Xilinx. *AXI UART Lite v2.0:*
https://www.xilinx.com/support/documentation/ip_documentation/axi_uartlite/v2_0/pg142-axi-uartlite.pdf
[16] Splunk:
https://www.security4vn.com/2016/08/splunk-mot-so-san-pham-ninh-mang.html
https://www.splunk.com/
[17] Tài liệu liên quan đến xilinx:
https://vimach.net/threads/vi-xu-ly-mem-microblaze.442/
https://www.xilinx.com/products/design-tools/microblaze.html#tabAnchor-documentation
https://www.xilinx.com/
https://ethernetfmc.com/
[18] Tài liệu nội dung về FPGA:
http://www.thuvientailieu.vn/tai-lieu/tim-hieu-tong-quan-ve-cong-nghe-fpga-25710/
https://vi.wikipedia.org/wiki/Fieldprogrammable_gate_array
https://www.slideshare.net/ravi4all/fpga-asic-technologiesflow
[19] Tài liệu về NetFlow V5:
https://vietnix.vn/netflow/
https://licensesoft.vn/netflow-la-gi.htm
https://cpc.vn/vi-vn/Tin-tuc-su-kien/Tin-tuc-chi-tiet/articleId/39301
[20] Tài liệu về ngôn ngữ Verilog:
http://www4.hcmut.edu.vn/~npduy/For%20Student_files/Tim%20hieu%20VHDL-Tieng-Viet.pdf
[21] Tài liệu về dữ liệu và giám sát mạng:
https://vietnetco.vn/siem-software-giam-sat-an-toan-mang-trong-tam-tay/1741.html
http://congnghehanoi.edu.vn/siem-la-gi.html
https://www.motadata.com/vi/blog/netflow-traffic-monitoring/
https://viettelidc.com.vn/tin-tuc/data-center-la-gi-cac-yeu-to-tao-nen-mot-trung-tam-du-lieu-chat-luong
[22] Tài liệu về ethernet frame và các giao thức:
https://vi.wikipedia.org/wiki/Frame_Ethernet#:~:text=M%E1%BB%99t%20packet%20data%20(g%C3%B3i%20d%E1%BB%AF,Ethernet%20%E1%BB%9F%20t%E1%BA%A7ng%20v%E1%BA%ADt%20l%C3%BD.
https://cpc.vn/vi-vn/Tin-tuc-su-kien/Tin-tuc-chi-tiet/articleId/34737
https://www.pydev.vn/d/30-giao-thuc-tcp-va-cau-truc-goi-tin-tcp