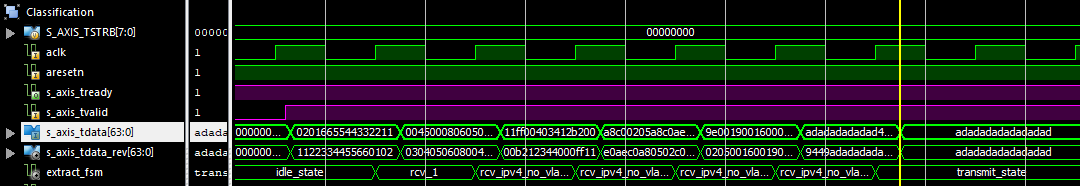
**Phân tích luồng dữ liệu trong NetFLow trong các trường hợp   
( FIN và RST = 0 và FIN hoặc RST = 1)**

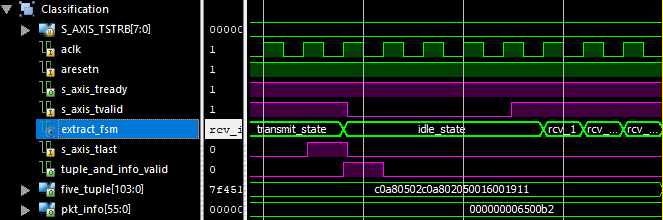
1. **Gói tin khi vào modude pkt classification**

Khi có xung ACLK, tín hiệu ARESETN và tready = 1 .Nếu tvalid bật thì module sẽ bắt đầu nhận dữ liệu. Data đi vào từng gói 64bit và đảo ngược chuỗi theo từng 8bit. Đồng thời thì trạng thái idle\_state kiểm tra data là vlan hay no\_vlan để phân tích data( Hình 1)



*Hình 1*

Sau khi phân tích, module classification sẽ thực hiện phân tích trích xuất ra các thông tin của dữ liệu đó là đó là: 5 tuple (IP nguồn, IP đích, MAC nguồn, Mac đích, protocol) và packet\_info( IP total length, cờ TCP và timestamp). Sau khi trích xuất các thông tin trên thì tlast = “1” báo hiệu cho việc phân tích kết thúc và bật tuple\_and\_info\_valid = “1” để module create\_or\_update bắt đầu nhận dữ liệu từ bram về so sánh.(Hình 2)

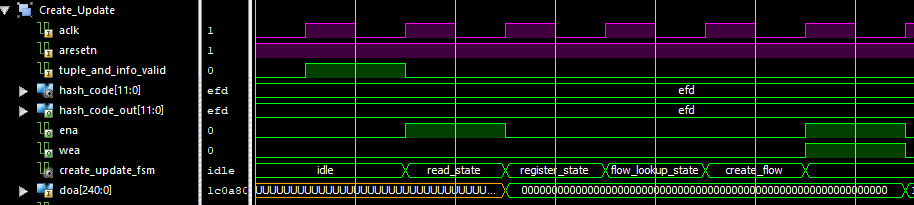


*Hình 2*

1. **Gói tin khi vào module Create or update.**

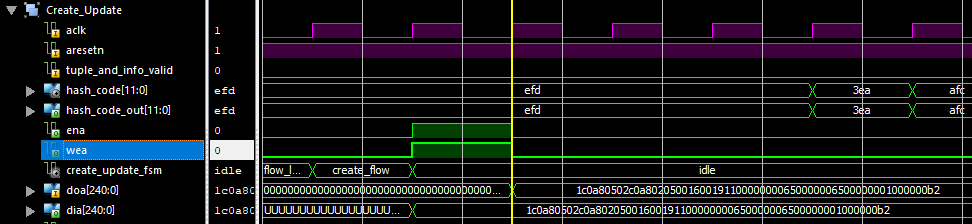
Hàm hash\_function sẽ thực hiện băm 5 tuple từ 104 bit thành 12 bit. Sử dụng để tìm địa chỉ ô nhớ đó trong Bram bằng cách đẩy hash\_code ra hash\_code\_out. Khi có tín hiệu từ tuple\_and\_info\_valic thì ena = “1” dữ liệu từ Bram được read ra cổng doa về lại module create or update để kiểm tra.

**2.1** Nếu read\_state cho ra dãy hex là dãy số 0 ( tức là flow này còn trống vậy là cờ busy của flow này là “ 0” ) ( Hình 3)



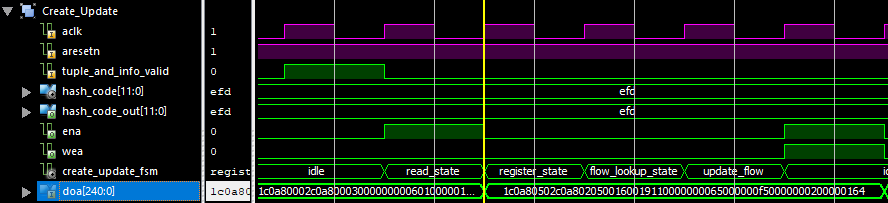
*Hình 3*

Lúc này thì create\_update\_fsm sẽ thực hiện “case” create\_flow để tạo một flow mới bằng cách bật ena = “1” và wea = “1”. Dữ liệu sẽ được đẩy vào Bram qua cổng dia.( Hình 4)



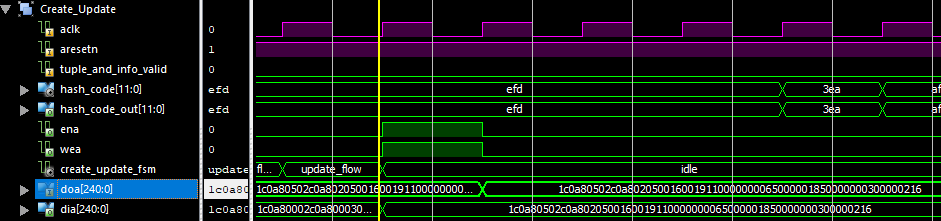
*Hình 4*

**2.2** Nếu read\_state đọc ra 1 dãy hex có bit đầu tiên bằng “1”( tức là đây là flow đang hoạt động trong bram và cờ busy = “1”) thì sẽ xảy ra sự so sánh giữa 5 tuple trong bram vừa đọc ra với 5 tuple vừa được chuyển từ module classification ( Hình 5)



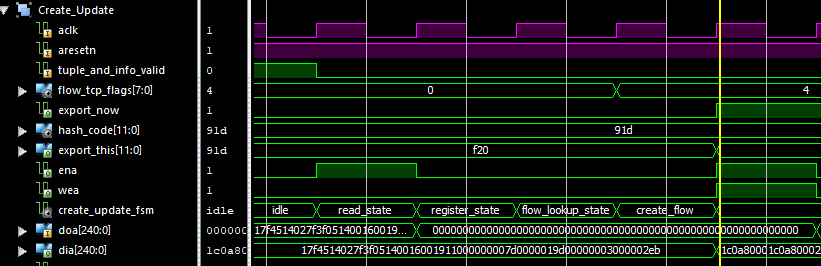
*Hình 5*

Nếu 5 tuple hoàn toàn trùng khớp thì create\_update\_fsm sẽ nhảy sang update\_flow. Lúc này Ena = “1” và wea = “1” data sẽ được đẩy vào bram qua dia.(Hình 6)



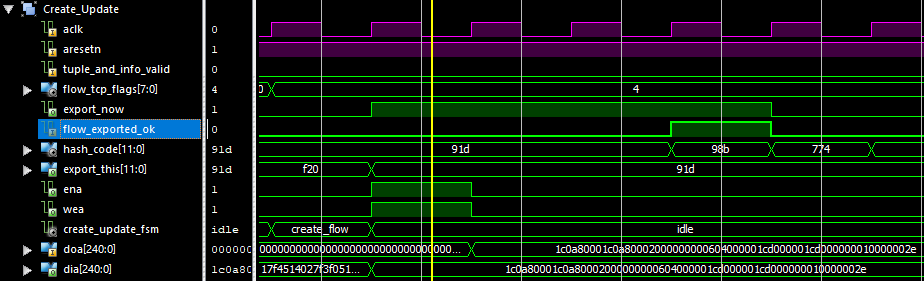
*Hình 6*

**2.3** Trong trường hợp create flow hoặc update flow phát hiện cờ RST hoặc cờ FIN = “ 1”. ( Trong trường hợp này read\_state đọc ra chuỗi Hex “0” như vậy flow này chưa tồn tại. Lúc này create\_update\_fsm sẽ thực hiện creat\_flow và đã phát hiện có cờ RST = “1” tức là flow\_tcp\_flags = “4” )( Hình 7)



*Hình 7*

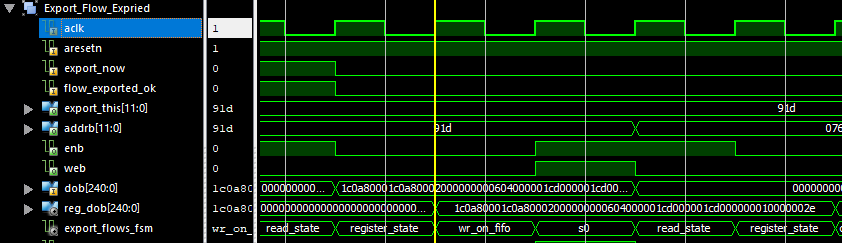
Khi có RST hoặc FIN = “1” ( trường hợp này là RST vì flags = “4”) thì Ena và wae vẫn được set = “1” và dữ liệu vẫn tiếp tục create or update lên Bram ( trong trường hợp này là create). Song song với việc create\_flow thì module create or update phát đi 2 tín hiệu đó là export\_now = “1” và export\_this = hash\_code và gửi đến module export\_expired. Cho đến khi nhận được phản hồi export\_ok từ module export\_expired thì export\_now sẽ tắt. (Hình 8)

**

*Hình 8*

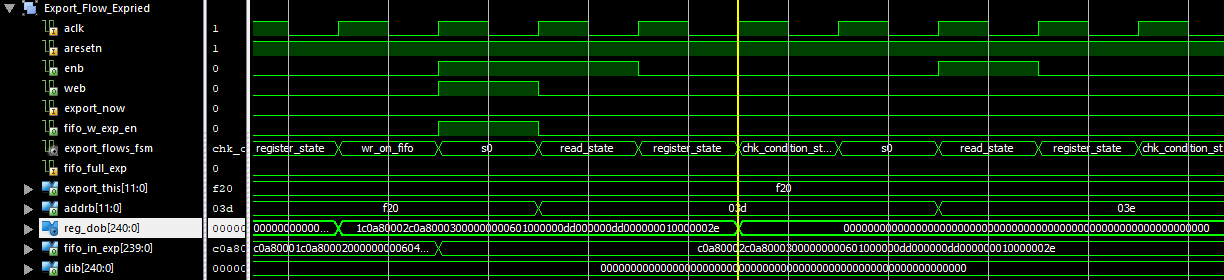
1. **Module Export\_expired.**
   1. Khi nhận được tín hiệu từ module create or update.

Khi export\_now = “1” và export\_this = hash\_code gửi đến thì module export\_expired phát ra tín hiệu export\_oke về module create\_or\_update để tắt export\_now, đồng thời cũng gửi một tín hiệu là addrb từ export\_expired đến bram. Đây chính là hash\_code được gửi từ create\_or\_update đến. Sau khi gửi địa chỉ đến bram. Đến khi enb = “1” thì flow có địa chỉ hash\_code được read về từ cổng dob và chứa trong reg\_dob của module export\_expired. (Hình 9).



*Hình 9*

Lúc này trong module sẽ thực hiện “case” write\_on\_fifo để export bằng cách đẩy tín hiệu reg\_dob( trừ busy flag) sang cổng fifo\_in\_exp và kiểm tra tín hiệu từ fifo\_full. Nếu fifo\_full = “0” thì fifo\_w\_exp\_en bật, dữ liệu từ reg\_dob nạp vào fifo\_in\_exp( lúc này đã mất cờ busy). Sau đó web và enb được set lên “1” module export\_expired thực hiện xóa luồng tại vị trí hash\_code bằng cách write vào bram tất cả bằng “0” qua cổng dib và sau khi nạp xong cho fifo. reg\_dob cũng trở thành dãy hex = “0” ( có xảy ra sự đồng bộ tín hiệu trong 2 module). (Hình 10)



*Hình 10*

**3.2** Khi Active timeout hoặc InActive time\_out hết hạn.

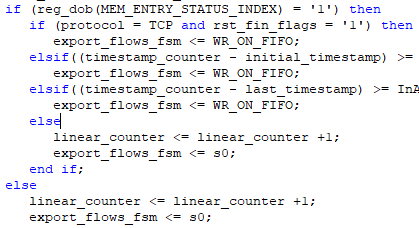
Nếu không nhận được tín hiệu export\_now và export\_this từ create or update thì module export\_expired sẽ đếm lên liên tục các luồng và đẩy vào reg\_dob. Đầu tiên nó sẽ kiểm tra cờ busy. Nếu cờ busy = “0” thì nó sẽ bỏ qua việc kiểm tra. Nếu cờ busy = “1” , lúc này nó kiểm tra 3 điều kiện.

Thứ nhất, RST hoặc FIN trong luồng có bằng “1” hay không.

Thứ hai, Active timeout có tới hạn hay chưa( timestamp\_counter - initial\_timestamp).

Thứ ba, InActive timeout có hết hạn hay chưa( timestamp\_counter - last\_timestamp).

Chỉ cần thỏa mãn một trong 3 điều trên. Luồng hiện tại sẽ lập tức được export.( phần này không có mô phỏng. Phần code thể hiện điều đó bằng các câu lệnh if, ngược lại 3 điều trên không thỏa thì linear counter vẫn đếm lên).



*Code mô tả*

1. **Module exp to 10G interface.**

Khi fifo\_empty\_exp = “0” tức là fifo lúc này đã có chứa data. Module exp\_via\_10G\_interface sẽ set fifo\_rd\_exp\_en = “1” để đọc dữ liệu từ fifo. Lúc này module chờ tín hiệu M\_AXIS\_10GMAC\_tready = “1”. Sau khi nhận được tín hiệu M\_AXIS\_10GMAC\_tready = “1” thì module sẽ xuất từng gói 64 bit (như vậy 5 tuple sẽ được chia thành hai, các gói nhỏ sẽ nhóm lại nhưng vẫn đảm bảo tính nguyên vẹn của data).