(12) BẢN MÔ TẢ SÁNG CHẾ THUỘC BẰNG ĐỘC QUYỀN SÁNG CHẾ

(19) **Cộng hòa xã hội chủ nghĩa Việt Nam (VN)**
CỤC SỞ HỮU TRÍ TUỆ

(11) **1-0037332**

(51)[2020.01] **H04L 12/00**; H04L 12/54

(13) **B**

(21) 1-2021-04138

(22) 06/07/2021

(45) 25/10/2023 427

(43) 27/09/2021 402

(73) Công ty cổ phần công nghệ Mạng LANCS Việt Nam (VN)
Tầng 2 số 236 Âu Cơ, phường Quảng An, quận Tây Hồ, thành phố Hà Nội

(72) Trịnh Quang Kiên (VN); Nguyễn Văn Thành (VN); Dương Quang Mạnh (VN).

(54) BỘ NHỚ CHIA SẺ ĐƠN LUỒNG VÀ ĐA LUỒNG CHO DỮ LIỆU DẠNG GÓI ỨNG DỤNG TRONG CÁC THIẾT BỊ CHUYỂN MẠCH GÓI TỐC ĐỘ CAO

(57)

Sáng chế đề cập tới kiến trúc, cách thức tổ chức và thuật toán quản lý bộ nhớ chia sẻ cho luồng thông tin dạng gói với định hướng ứng dụng cho các thiết bị chuyển mạch số có băng thông lớn. Cụ thể, sáng chế đề cập tới bộ nhớ chia sẻ đơn luồng và bộ nhớ chia sẻ đa luồng cho dữ liệu dạng gói ứng dụng trong các thiết bị chuyển mạch gói tốc độ cao. Bộ nhớ chia sẻ đơn luồng có cách tổ chức lưu trữ dữ liệu theo danh sách liên kết giúp tiết kiệm không gian nhớ cũng như hiệu suất sử dụng không gian trong vùng nhớ chính, tuy nhiên khi sử dụng cần có các khối thực hiện gom và phân phối các luồng thông tin ở đầu vào và đầu ra. Bộ nhớ chia sẻ đa luồng có khả năng phục vụ nhiều luồng thông tin độc lập trực tiếp do đó không cần phải có bộ nhớ đệm cục bộ. Khả năng phục vụ đa luồng cũng giúp giảm thiểu độ trễ gói và tỷ lệ bỏ gói khi ứng dụng trong các hệ thống băng thông lớn. Cả hai kiến trúc bộ nhớ chia sẻ nêu trong sáng chế đều định hướng cho thiết kế chuyển mạch tích hợp với mức yêu cầu về tài nguyên nhớ rất nhỏ và có ưu thế rõ ràng so với một số thiết kế truyền thống về hiệu suất sử dụng phần cứng, giúp giảm độ phức tạp của hệ thống, đặc biệt so với các thiết kế dùng bộ đệm ngoài. Cả hai kiến trúc thiết kế đều có thể ứng dụng trong các bộ chuyển mạch lớp 2 (Layer 2), bộ chuyển mạch lớp 3 (Layer 3) trên nền tảng công nghệ FPGA.

Lĩnh vực kỹ thuật được đề cập

Sáng chế đề cập tới kiến trúc thiết kế và tổ chức bộ nhớ chia sẻ cho dữ liệu dạng gói ứng dụng trong thiết kế các thiết bị mạng băng thông lớn.

Tình trạng kỹ thuật của sáng chế

Hiện nay, nhu cầu về Internet tốc độ cao tăng lên đáng kể, để đáp ứng được các yêu cầu truyền dữ liệu lớn, truyền thông thời gian thực và truyền thông đa phương tiện Độ nét cao (HD) qua giao thức chuyển mạch IP (Internet Protocol), kiến trúc các sản phẩm mạng dựa trên IP cần phải có sự phát triển và thay đổi tương ứng [1]-[2]. Phần cứng xử lý chuyên dụng cho ứng dụng mạng cần có hiệu suất làm việc và hiệu quả sử dụng năng lượng cao, khả năng tùy biến hoặc tái lập trình (tái cấu trúc) linh hoạt để phù hợp với nhu cầu đa dạng trong thực tế. Nội dung của mô tả này tập trung vào bộ đệm dữ liệu gói IP có khả năng đáp ứng băng thông lớn, ứng dụng trong các thiết bị mạng thế hệ tiếp theo (NGN) [3]. Thiết kế của bộ đệm chia sẻ có thể dùng cho dữ liệu Ethernet nói riêng và dữ liệu dạng gói nói chung, cũng như cấu hình cho nhiều ứng dụng đặc thù khác nhau. Thiết kế đề xuất sẽ được đánh giá dựa vào kết quả thực thi trên nền công nghệ FPGA, tuy nhiên bản chất về kiến trúc và thuật toán cũng như các kỹ thuật được đề cập hoàn toàn không phụ thuộc vào công nghệ thực thi.

Bộ nhớ chia sẻ được hiểu là một bộ đệm trung tâm cho tất cả các luồng dữ liệu trung chuyển qua các thiết bị chuyển mạch gói. Các gói dữ liệu từ các giao tiếp vật lý đầu vào qua khối chuyển mạch sẽ được lưu trữ cho tới khi có đủ thông tin để gửi tới địa chỉ đích yêu cầu. Bộ nhớ đệm cần có các tiêu chí kỹ thuật như sau: có tốc độ truy xuất rất nhanh; có khả năng lưu trữ gói tin kích thước bất kỳ; có khả năng hỗ trợ truy cập ngẫu nhiên. Tiêu chí cuối cùng là cần thiết do thứ tự gửi ra của dữ liệu trong các thiết bị mạng thực tế không chỉ phụ thuộc vào thứ tự gửi vào mà còn phụ thuộc các quy định liên quan tới chất lượng của dịch vụ (Quality of Service). Bộ nhớ chia sẻ có khả năng tận dụng khả năng truy xuất rất nhanh để không cần sử dụng kích thước đệm quá lớn nhưng vẫn đáp ứng được yêu cầu về độ trễ và băng thông. Ví dụ, một bộ nhớ chia sẻ kích thước 1Mb có thể phục vụ lưu lượng vài chục tới hàng trăm Gbps. So với bộ nhớ đệm cục bộ phân tán, ưu điểm của bộ nhớ chia sẻ là hiệu suất sử dụng bộ nhớ rất cao. Cụ thể, trong bộ nhớ đệm cục bộ phân tán, trong cùng thời điểm không gian nhớ cục bộ của cổng này không thể chia sẻ cho các cổng khác; ngược lại trong bộ nhớ chia sẻ, cơ chế dùng chung cho phép các cổng có thể tận dụng toàn bộ không gian nhớ còn trống. Tuy vậy thiết kế bộ nhớ chia sẻ có độ phức tạp cao hơn so với mô hình bộ nhớ cục bộ phân tán ở cơ chế điều khiển luồng đọc ghi.

Kiến trúc, thiết kế và thuật toán sử dụng cho các bộ nhớ chia sẻ thường không được công bố đầy đủ vì đây là thiết kế mang tính cốt lõi của các sản phẩm được một số công ty lớn như Cisco, Huawei, Marvel phát triển trên nền công nghệ ASIC kết hợp bộ nhớ ngoài. Dòng Catalyst 5000 [4] của Cisco dùng cơ chế *port buffered,* nghĩa là đệm luồng cục bộ dùng bộ nhớ ngoài DRAM, tuy đơn giản hơn cho phần điều khiển nhưng sẽ chi phí cao hơn. Dòng Catalyst 4000 [5] dùng bộ nhớ chia sẻ nhưng cũng sử dụng bộ nhớ SRAM tĩnh bên ngoài với dung lượng khá lớn từ 8-16 Mb.

Mô tả một số sáng chế liên quan

Trong các hệ thống mạng truyền thông, phương án đơn giản và được áp dụng rộng rãi nhất là sử dụng các bộ đệm trong (*internal buffer*) tại vị trí nhận dữ liệu để lưu trữ tạm thời các gói tin nhận được, tiếp đó kiểm tra tính toàn vẹn của chúng trước khi định hướng chúng tới các đích đến dự kiến. Cơ cấu chuyển mạch như vậy được gọi là "lưu trữ và chuyển tiếp" (*store and forward*). Hình 1 trình bày một ví dụ về một thiết bị chuyển mạch thông thường dựa trên cơ cấu chuyển mạch lưu trữ và chuyển tiếp. Thiết bị có nhiệm vụ chuyển tiếp các gói tin từ các mạng (*network*) khác nhau qua các giao diện thu (*receive interface*) tới các bộ đệm thu (*receive buffer*). Tiếp đó, các gói tin trong các bộ đệm thu được gửi tới mạch chuyển dữ liệu (*data transfer circuit*), nơi có nhiệm vụ định tuyến chúng tới các bộ đệm phát (*transmit buffer*) tương ứng với địa chỉ cổng đích của mỗi gói. Nếu có yêu cầu phát gửi đến, các gói tin lưu trữ trong các bộ đệm phát được đọc ra và qua giao diện phát (*transmit interface*) gửi tới các mạng tương ứng. Để đơn giản vấn đề, các giao diện thu và giao diện phát được thể hiện riêng rẽ, còn trên thực tế chúng thường được thống nhất với nhau trong một giao diện duy nhất.

Nhược điểm của các cơ cấu chuyển mạch gói trong Hình 1 là thiết bị yêu cầu một dung lượng bộ nhớ lớn do nó phải đảm bảo một số lượng vùng nhớ đệm chuyên dụng cho mỗi cổng nhận và cổng truyền. Để khắc phục nhược điểm đó, có thể sử dụng một bộ nhớ chia sẻ duy nhất cung cấp cho tất cả các gói tin đến và đi. Hình 2 thể hiện một phương án thiết bị chuyển mạch thông thường sử dụng bộ nhớ chia sẻ. Tương tự như thiết bị trình bày trên Hình 1, thiết bị trên Hình 2 có chức năng chuyển tiếp các gói tin giữa các mạng khác nhau. Các thành phần của thiết bị chuyển mạch trong phương án này bao gồm: các giao diện thu, mạch chuyển dữ liệu thu, bộ nhớ chia sẻ nằm ở vị trí trung tâm, mạch chuyển dữ liệu phát, các giao diện phát. Để đơn giản vấn đề, các giao diện thu và giao diện phát được thể hiện riêng rẽ, còn trên thực tế chúng thường được thống nhất với nhau trong một giao diện duy nhất. Các giao diện thu tiếp nhận các gói tin từ các mạng tương ứng, sau đó qua mạch chuyển dữ liệu thu chúng được lưu vào trong bộ nhớ chia sẻ. Khi có yêu cầu phát từ mỗi giao diện phát, mạch chuyển dữ liệu phát phân phát các gói tin từ bộ nhớ chia sẻ tới giao diện phát thích hợp tương ứng với đích đến của mỗi gói tin.

Các thiết bị chuyển mạch trên Hình 1 và Hình 2 thường được thiết kế để quản lý các gói tin gửi đi cho mỗi cổng đích cụ thể bằng cách sử dụng hàng đợi có cấu trúc danh sách liên kết

trong đó các gói đang chờ được liên kết lần lượt. Để lưu trữ các gói có độ dài thay đổi, nhiều bộ đệm như vậy được cấu hình để xử lý dữ liệu theo đơn vị độ dài tối đa của gói và các gói được quản lý riêng lẻ bằng con trỏ. Một lợi thế của kiến trúc này là dễ kiểm soát, chẳng hạn, gói nhận được có lỗi có thể loại bỏ một cách dễ dàng bằng cách xóa con trỏ tới gói đó. Tuy nhiên, các bộ đệm kiểu này tiêu thụ một lượng lớn không gian bộ nhớ, bằng độ dài gói tối đa nhân với số gói tối đa được lưu trữ, trong trường hợp đa phần các gói nhận được đều nhỏ hơn đáng kể so với chiều dài gói tối đa giả định, một phần lớn bộ đệm chưa được sử dụng dẫn tới suy giảm hiệu suất của bộ nhớ.

Để giải quyết vấn đề này, một bộ nhớ chia sẻ thường được cấu hình như một khối nhiều bộ đệm dữ liệu không liên tục với độ dài cố định là 256 bytes để lưu trữ các phân đoạn của gói (*packet segment*) nhận từ các cổng vào. Bộ nhớ chia sẻ sau đó sẽ gán các ô nhớ dữ liệu còn trống cho các phân đoạn gói này theo các địa chỉ liên kết (*link addresses*) và danh sách liên kết của các địa chỉ khối (*linked list of the block addresses*) tương ứng với các phân đoạn gói. Địa chỉ liên kết và danh sách liên kết được lưu trữ trong một bộ nhớ liên kết (có thể là bộ nhớ RAM liên kết - *link RAM*). Các phân đoạn gói được chèn vào hàng đợi đầu ra theo các địa chỉ khối, địa chỉ liên kết tương ứng và các cổng ra được chỉ định.

Dựa theo nguyên lý trên, đơn sáng chế của Mĩ số US 2003/0147410 A1 [6] về kiến trúc chuyển mạch Ethernet và cấp phát bộ nhớ động có đề cập tới một cấu trúc bộ nhớ chia sẻ được trình bày như trên Hình 3. Theo đó, bộ nhớ chia sẻ 1 bao gồm các khối nhớ dữ liệu (*block data buffer*) 2 có cùng kích thước nằm trong vùng bộ đệm trống (*free-buffer pool*) 3 và vùng bộ đệm chiếm (*assigned buffer*) 4 để lưu trữ các phân đoạn gói. Vùng bộ đệm trống được tạo ra nhờ việc giải phóng vùng bộ đệm chiếm khỏi các phân đoạn gói đã đọc ra và vùng nhớ này được dùng để lưu trữ các phân đoạn gói đang chờ nhận 5, trong khi đó vùng bộ đệm chiếm lưu trữ các phân đoạn gói được nhận từ cổng vào 6. Kích thước của không gian địa chỉ liên kết 7 trong bộ nhớ RAM liên kết 8 (sau đây gọi là RAM liên kết) được tạo cấu hình bằng số lượng khối nhớ trong bộ nhớ chia sẻ 1. Các địa chỉ liên kết 10 trong RAM liên kết và các địa chỉ khối 11 trong bộ nhớ chia sẻ liên hệ với nhau bởi quan hệ ánh xạ 12.

Trong một ví dụ cụ thể, các địa chỉ liên kết của danh sách liên kết (*linked list*) 13 cho sáu phân đoạn gói của một gói đã nhận bao gồm: #4, #6, #8, #9, #13 và #14. Các phân đoạn gói này được lưu trữ trong vùng bộ đệm chiếm của bộ nhớ chia sẻ tại các địa chỉ khối tương ứng: &50, &70, &90, &100, &140 và &150. Địa chỉ liên kết cuối cùng #14 sẽ được liên kết tới địa chỉ liên kết #10 - phân đoạn gói đầu tiên của gói được nhận kế tiếp. Một cờ (*flag*) 14 thể hiện địa chỉ liên kết #14 trỏ tới địa chỉ liên kết #10 sẽ được chỉ định, đồng thời nó cũng chỉ ra rằng khối nhớ tương ứng của địa chỉ liên kết #14 lưu trữ phân đoạn gói cuối cùng của gói được nhận. Các địa chỉ #0, #7, #11, #12, #15, #3 và #5 của danh sách liên kết 15 trong RAM liên kết được biểu diễn bởi các đường đứt đoạn thể hiện vùng không gian địa chỉ liên kết được giải phóng của một gói đã được đọc ra, tương ứng với các khối nhớ trống &5, &15, &25, &35, &45, &55 và &65 trong vùng bộ đệm

trống của bộ nhớ chia sẻ. Không gian địa chỉ liên kết được giải phóng này cũng như các khối trống tương ứng trong vùng bộ đệm trống sẵn sàng đáp ứng cho các phân đoạn gói của gói nhận tiếp theo. Trong một tình huống *unicast* (truyền một gói dữ liệu đến một địa chỉ nhận duy nhất được chỉ định), mỗi phân đoạn gói được chèn vào hàng đợi đầu ra 15 tới cổng ra được chỉ định 16 theo tham chiếu của danh sách liên kết.

Quá trình từ khi nhận đến khi truyền các phân đoạn gói của một gói được tiến hành theo bốn bước tuần tự. Bước đầu tiên là lấy liên kết (*get link*), là quá trình gán các khối nhớ dữ liệu chưa sử dụng trong bộ nhớ chia sẻ cho các phân đoạn gói và xác định không gian địa chỉ liên kết tương ứng. Bước thứ hai là tạo liên kết (*make link*), là quá trình tạo các địa chỉ liên kết của các phân đoạn gói được liên kết để hình thành danh sách liên kết. Bước thứ ba là đọc liên kết (*read link*), là quá trình đọc danh sách liên kết. Bước cuối cùng là bước giải phóng liên kết (*release link*), là quá trình giải phóng không gian địa chỉ liên kết sau khi danh sách liên kết đã được đọc ra.

Đề xuất trong đơn sáng chế của Mĩ số US 2003/0147410 A1 đưa ra nhằm giải quyết vấn đề giới hạn về băng thông cũng như số lượng cổng truyền trong kiến trúc chuyển mạch gói Ethernet, qua đó giúp làm tăng băng thông phát khi cấp phát bộ nhớ động so với đề xuất đã được mô tả. Sơ đồ khối nguyên lý một phương án của đề xuất được trình bày trên Hình 4 với các số thứ tự tương ứng như các số thứ tự ghi trên Hình 3. So với sơ đồ trên Hình 3, sơ đồ trên Hình 4 đề xuất một RAM liên kết trống (*free-link RAM*) 17 đóng vai trò là bộ nhớ FIFO (*First In First Out*) và một danh sách liên kết "đơn nhất" (*"single" linked list*) 13*. Kích thước của không gian địa chỉ liên kết trống (*free-link address space*) 18 trong RAM liên kết trống và kích thước của không gian địa chỉ liên kết trong RAM liên kết được tạo cấu hình bằng nhau và bằng số lượng khối nhớ trong bộ nhớ chia sẻ. Độ rộng của địa chỉ liên kết trống 19 của RAM liên kết trống nhỏ hơn độ rộng của địa chỉ liên kết 20 của RAM liên kết. Kết quả là quá trình hình thành không gian địa chỉ liên kết trước khi tạo ra danh sách liên kết cho các phân đoạn gói của một gói tin và việc giải phóng không gian địa chỉ liên kết sau khi danh sách liên kết được đọc ra có thể được đồng xử lý.

Trong một ví dụ cụ thể, sáu phân đoạn gói của một gói tin nhận được lưu trữ trong các khối nhớ dữ liệu của bộ nhớ chia sẻ tại các địa chỉ &60, &80, &100, &110, &120 và &140 tương ứng với các địa chỉ liên kết #4, #6, #8, #9, #13 và #14. Địa chỉ liên kết phân đoạn gói đầu tiên #4 của gói tin nhận hiện tại sẽ được liên kết "một lần" ("for once") với địa chỉ liên kết của phân đoạn gói đầu tiên #10 của gói tin nhận kế tiếp. Các địa chỉ liên kết của sáu phân đoạn gói đã cho chỉ được tạo và đọc trong một "quá trình đơn nhất" ("single process") và do vậy hình thành một danh sách liên kết "đơn nhất". Mặt khác, khi tất cả các phân đoạn gói đã đọc xong, các địa chỉ liên kết được giải phóng #0, #7, #11, #12, #15, #3 và #5 sẽ được quản lý theo chuỗi FIFO thông qua RAM liên kết trống để nhận gói tin tiếp theo. Tương tự như các địa chỉ liên kết, các khối nhớ dữ liệu

cũng được giải phóng trở về vùng bộ đệm trống 3 trên bộ nhớ chia sẻ để phục vụ lưu trữ cho gói tin nhận tiếp theo.

Một số nghiên cứu điển hình khác về bộ đệm cho các thiết bị mạng có thể kể đến như đề xuất được mô tả trong [7], tuy nhiên đề xuất này không mang tính tổng quát cho các gói có kích thước thay đổi và các luồng vào không đồng nhất, đặc biệt với các gói dữ liệu rất lớn (Ví dụ, gói Jumbo 9600 Bytes) và băng thông chỉ đạt 40 Gbps trên Xilinx Virtex 4. Trong nghiên cứu [8], tác giả đề xuất sử dụng bộ nhớ ngoài RLDRAM và FPGA để cấu trúc bộ nhớ chia sẻ nhưng băng thông bị hạn chế do giới hạn băng thông giữa FPGA và chip nhớ ngoài, chỉ đạt tối đa 12.8 - 20 Gbps.

Mô tả trong đơn sáng chế của Mỹ số US 2005/0163141 A1 [9] đề xuất một giải pháp bộ nhớ chia sẻ trong đó các con trỏ địa chỉ và khối nhớ chia sẻ nằm xen kẽ nhau, không có sự tách biệt vùng nhớ. Giải pháp sử dụng hệ thống con trỏ liên kết bao gồm con trỏ đơn và con trỏ kép, trong đó con trỏ đơn sẽ trỏ tới địa chỉ của khối nhớ chia sẻ kế tiếp dùng để lưu trữ các phân đoạn gói, khối nhớ này xếp ngay sau con trỏ phân đoạn, còn con trỏ kép trỏ tới khối nhớ chia sẻ đầu tiên của gói dữ liệu tiếp theo. Cách thức tổ chức như vậy tương đối phức tạp trong khâu quản lý, đồng thời tiêu tốn nhiều dung lượng nhớ cho việc lưu trữ con trỏ liên kết so với các giải pháp bộ nhớ chia sẻ khác có sử dụng con trỏ liên kết.

Mô tả trong đơn sáng chế của Mỹ số US 006160814A [10] có đề cập về kiến trúc bộ nhớ chia sẻ nhưng chỉ là đệm tạm thời ở đầu vào trước khi đưa vào các vùng nhớ chính bên trong theo kiến trúc phân tán. Thiết kế này giúp hạn chế hiện tượng nghẽn băng thông cục bộ bên trong khi nhiều gói tin cùng địa chỉ đích được ghi vào cùng một vùng đệm cục bộ nhưng bản chất không phải kiến trúc bộ nhớ chia sẻ.

Mô tả trong đơn sáng chế của Mỹ số US 2010/0325370 A1 [11] về một bộ nhớ chia sẻ dùng bốn loại danh sách liên kết và có cấu trúc con trỏ chứa nhiều trường thông tin sẽ có độ phức tạp cao khi thực thi trên phần cứng và cần phải phân loại gói tin theo bản chất (Ví dụ, gói đơn địa chỉ - unicast, đa địa chỉ - multicast, quảng bá - broadcast).

Mô tả trong đơn sáng chế của Mĩ số US 2012/0236789 A1 [12] đề xuất mô hình và giải pháp quản lý bộ nhớ cho kiểm soát truy cập phương tiện (Media Access Control - MAC) tốc độ cao, sử dụng một bộ đệm dữ liệu lưu trữ các gói tin dưới dạng cấu trúc hỗn hợp, trong đó cấu trúc đầu tiên có phần đầu chứa các thông tin quản lý như độ dài gói, thứ tự của chuỗi và phần sau là một con trỏ. Con trỏ của cấu trúc thứ nhất trỏ tới cấu trúc thứ hai và cấu trúc này có dạng một hoặc một vài khối có thiết kế giống nhau: phần đầu của khối là nơi lưu trữ dữ liệu của gói, phần sau của khối là một con trỏ và nó lại trỏ tới khối tiếp theo nếu như dữ liệu của gói có kích thước vượt quá khả năng chứa của một khối, từ đó hình thành nên một danh sách liên kết (theo chiều dọc) các con trỏ tới các phân đoạn dữ liệu của một gói. Cấu trúc đầu tiên của mỗi gói lại chứa con trỏ tới cấu trúc đầu tiên của gói kế tiếp, tạo thành một chuỗi các nút (node) dữ liệu được liên kết với nhau bằng chuỗi con trỏ theo hàng ngang. Như vậy, có thể coi cấu trúc con trỏ liên kết của các nút dữ liệu cho mỗi gói có

dạng hai chiều, tạo thành một chuỗi liên kết dữ liệu trong bộ đệm khá chặt chẽ. Tuy nhiên, nếu xét về tiêu chí tiết kiệm dung lượng bộ nhớ, giải pháp đề xuất trong sáng chế sẽ không đảm bảo, mặt khác cấu trúc liên kết con trỏ hai chiều như vậy tương đối phức tạp trong việc quản lý.

Mô tả trong đơn sáng chế của Mỹ số 6,363,075 B1 [13] đề xuất giải pháp chia nhỏ không gian nhớ của bộ nhớ SRAM kết hợp cơ chế cho phép các bộ phận chuyển mạch đầu vào và đầu ra có thể truy cập tới từng vị trí nhưng không sử dụng danh sách liên kết và dữ liệu được phân phối động thành các gói liên tục. Thiết kế này cơ bản phù hợp cho băng thông hạn chế vì việc ghép nối cơ học các bộ nhớ rời rạc dẫn đến sự phức tạp trong xử lý của khối chuyển mạch đầu vào.

Như vậy, nhu cầu về một giải pháp hiệu quả về mặt chi phí, đơn giản về mặt cấu trúc phần cứng chuyển mạch lõi trong các thiết bị mạng băng thông lớn ngày càng trở nên cấp thiết. Xuất phát từ các đặc điểm như trên, cùng với các ưu nhược điểm của các sáng chế có liên quan đã được trình bày, chúng tôi đề xuất giải pháp thiết kế bộ nhớ chia sẻ đơn luồng và đa luồng cho dữ liệu dạng gói ứng dụng trong các thiết chuyển mạch gói tốc độ cao đảm bảo các yêu cầu về tính mới, tính sáng tạo cũng như tính hiệu quả. Các nội dung trình bày cụ thể của giải pháp sẽ lần lượt được trình bày trong những phần tiếp theo của tài liệu này.

Danh mục tài liệu trích dẫn

[1] R. Seifert and J. Edwards, The All-New Switch Book: The Complete Guide to LAN Switching Technology, vol. 10. 2008.

[2] Buffering Data on LAN Switch Architecture [Online] at on eTutorials.org. http://etutorials.org/Networking/Lan+switching+fundamentals/Chapter+2.+LAN+Switch+Architecture/Buffering+Data/.

[3] Nayaka, R. J. (2012). High Performance Ethernet Packet Processor Core for Next Generation Networks. International Journal of Next-Generation Networks, 4(3), 89–99. https://doi.org/10.5121/ijngn.2012.4307

[4] Cisco Industrial Ethernet 5000 Series Switches Datasheet [Online] available at on Cisco.com https://andovercg.com/datasheets/cisco-5000-5500-series-switches.pdf

[5] Cisco Industrial Ethernet 4000 Series Switches Datasheet [Online] available at Cisco.com https://www.cisco.com/c/en/us/products/collateral/switches/industrial-ethernet-4000-series-switches/datasheet-c78-733058.pdf

[6] Hsu et al. US Patent Application ETHERNET SWITCHING ARCHITECTURE AND DYNAMIC MEMORY ALLOCATION - US 2003/0147410 A1, 7 Aug. 2003

[7] M. Ejlali, M. A. Montazeri, H. Saidi, and A. Ghiasian, “Design and implementation of a shared memory switch fabric,” 2012 6th Int. Symp. Telecommun. IST 2012, pp. 721-727, 2012.

[8] Burns, D., Toall, C., Mclaughlini, K., Sezeri, S., Hutton, M., & Cackovic, K. (2007). AN FPGA BASED MEMORY EFFICIENT SHARED BUFFER IMPLEMENTATION. 661-664.

[9] Katayama et al. US Patent Application NETWORK SWITCHING DEVICE AND METHOD USING SHARED BUFFER - US 2005/0163141 A1, 28 Jul. 2005

[10] Jing-Fei Ren et al. US Patent Application 6,160,814 B1 DISTRIBUTED SHARED-MEMORY PACKET, 2002

[11] Cummings et al. US Patent Application SHARED-MEMORY SWITCH FABRIC ARCHITECTURE - US 2010/0325370 A1, 31 Dec. 2010
[12] Dravida et al. US Patent Application MEMORY MANAGEMENT FOR HIGH SPEED MEDIA ACCESS CONTROL - US 2012/0236789 A1, 20 Sep. 2012
[13] Wang, Y. T., & Nguyen, P. E. (2002). US Patent Application 6,363,075 B1 SHARED BUFFER MANAGEMENT MECHANISMAND METHOD USING MULTIPLE LINKED LISTS IN A HIGH SPEED PACKET SWITCHING SYSTEM, 2002
[14] Xilinx AXI 4 STREAM reference guide [Online] Available at https://www.xilinx.com/support/documentation/ip_documentation/ug761_axi_reference_guide.pdf.

Bản chất kỹ thuật của sáng chế

Mục tiêu của sáng chế là đề xuất giải pháp về bộ nhớ chia sẻ dùng trong thiết bị mạng băng thông lớn giúp khắc phục các nhược điểm của các giải pháp bộ nhớ tương tự nêu trong tình trạng kỹ thuật của thiết kế. Các mục tiêu cụ thể của sáng chế bao gồm:

- Đề xuất một số cấu trúc bộ nhớ chia sẻ có khả năng ứng dụng trong các lõi chuyển mạch số với băng thông lớn cỡ hàng chục tới hàng vài trăm Gbps.
- Đề xuất thuật toán quản lý dữ liệu vào ra trong các bộ nhớ cho phép truy cập ngẫu nhiên với độ trễ thấp (cỡ µs).
- Đề xuất phương án thiết kế bộ nhớ chia sẻ trên phần cứng có tính tối ưu về tài nguyên sử dụng.
- Đề xuất phương án thiết kế để có thể tận dụng tối đa không gian nhớ (tới 100%) trong trường hợp lý tưởng.
- Đề xuất phương án thiết kế bộ nhớ chia sẻ trong đó nội dung của một gói tin được lưu trữ phân tán nhưng thứ tự truy xuất đảm bảo chính xác.
- Đề xuất phương án thiết kế bộ nhớ sử dụng sẽ là bộ nhớ trong có dung lượng cấp độ vài Mb, và không cần dùng đến bộ nhớ DRAM hay SRAM bên ngoài.
- Đề xuất các mô hình ứng dụng thực tế của bộ nhớ chia sẻ trong các lõi chuyển mạch số.

Trong bản mô tả này sẽ trình bày về hai kiến trúc thiết kế của bộ nhớ chia sẻ: (i) bộ nhớ chia sẻ đơn luồng và (ii) bộ nhớ chia sẻ đa luồng. Kiến trúc thứ nhất tập trung tối ưu cho phần điều khiển và tài nguyên phần cứng sử dụng, cho phép lưu trữ và truy cập các gói tin với thứ tự bất kì. Kiến trúc thứ hai có thể phục vụ đồng thời nhiều luồng dữ liệu vào ra với độ trễ gói trung bình và băng thông hệ thống tốt hơn so với kiến trúc thứ nhất, tuy nhiên độ phức tạp trong tổ chức bộ nhớ cũng trở nên lớn hơn. Cả hai kiến trúc đều tận dụng được tối đa không gian lưu trữ trong trường hợp lý tưởng. Bộ nhớ chia sẻ sẽ được nghiên cứu tích hợp trên một chíp đơn cùng với các phần logic điều khiển sẽ là giải pháp có hiệu quả về mặt chi phí, đơn giản hóa về mặt cấu trúc cho phần cứng chuyển mạch lõi trong các thiết bị mạng.

Theo phương án thứ nhất, sáng chế đề xuất bộ nhớ chia sẻ đơn luồng. Bộ nhớ chia sẻ đơn luồng dùng cho một luồng dữ liệu, có khả năng lưu trữ các gói tin với kích thước bất kỳ (trong trường hợp gói tin Ethernet, kích thước gói tin tối đa là 9600 byte). Thiết kế bộ nhớ

được tổ chức để không gian lưu trữ trong trường hợp lý tưởng sẽ đạt hiệu suất 100% và các gói tin có thể được truy cập ngẫu nhiên. Mặc dù thiết kế bộ nhớ đơn luồng chỉ hỗ trợ một luồng dữ liệu duy nhất nhưng không giới hạn số luồng vật lý từ bên ngoài với điều kiện các luồng này đã được gom lại thành một luồng thống nhất. Các gói tin thông qua giao diện AMBA AXI4-Stream sẽ được chia thành các khối dữ liệu kích thước điều chỉnh được (thường từ 256 - 2024 bit). Giao tiếp AMBA AXI4-Stream cho phép băng thông tối đa BW = B x F bps trong đó B là kích thước gói hay độ rộng dữ liệu, F là tần số làm việc (Ví dụ, bộ nhớ sẽ đạt băng thông 100 Gbps với N = 512, F = 200 MHz).

Theo phương án thứ hai, sáng chế đề xuất bộ nhớ chia sẻ đa luồng. Bộ nhớ chia sẻ đa luồng dùng cho nhiều luồng dữ liệu, có khả năng lưu trữ các gói tin với kích thước bất kỳ và được tổ chức để không gian lưu trữ trong trường hợp tốt nhất sẽ có hiệu suất 100%, các gói tin có thể được truy cập ngẫu nhiên. Điểm khác biệt là thiết kế đa luồng cho phép bộ nhớ có thể phục vụ đồng thời nhiều luồng dữ liệu khác nhau mà không gây ra hiện tượng nghẽn đầu vào (*Head-of-the-line*). Cách tính băng thông một chiều của thiết kế cũng giống như trường hợp bộ nhớ đơn luồng: BW = B x F bps trong đó B là độ rộng dữ liệu, F là tần số làm việc (Ví dụ, bộ nhớ sẽ đạt băng thông 100 Gbps với B = 512, F = 200 MHz). Bộ nhớ chia sẻ đa luồng có thể tổ chức phân cấp và ghép tầng để hỗ trợ băng thông và tốc độ chuyển mạch cao hơn.

Ngoài ra, sáng chế cũng đề xuất lõi chuyển mạch thông tin số dạng gói có sử dụng các bộ nhớ chia sẻ đơn luồng và đa luồng như nêu trên.

Mô tả vắn tắt các hình vẽ

Hình 1 là hình vẽ minh họa một phương án thiết bị chuyển mạch thông thường dựa trên cơ cấu chuyển mạch lưu trữ và chuyển tiếp.

Hình 2 là hình vẽ minh họa một phương án thiết bị chuyển mạch thông thường sử dụng bộ nhớ chia sẻ.

Hình 3 là hình vẽ minh họa một cấu trúc bộ nhớ chia sẻ sử dụng địa chỉ liên kết và danh sách liên kết đầy đủ được mô tả trong sáng chế của Mỹ số US 2003/0147410 A1.

Hình 4 là hình vẽ minh họa một cấu trúc bộ nhớ chia sẻ sử dụng địa chỉ liên kết và danh sách liên kết "đơn nhất" được mô tả trong sáng chế của Mỹ số US 2003/0147410 A1.

Hình 5 là hình vẽ minh họa cấu trúc các khối phần cứng cơ bản trong bộ nhớ chia sẻ đơn luồng theo sáng chế.

Hình 6 là hình vẽ minh họa mô hình tổ chức con trỏ đọc ghi bộ đệm trong bộ nhớ chia sẻ đơn luồng theo sáng chế.

Hình 7 là hình vẽ minh họa chi tiết thuật toán của bộ nhớ chia sẻ đơn luồng (Thuật toán 1) theo sáng chế.

Hình 8 là hình vẽ minh họa việc cập nhật bộ nhớ con trỏ khi mới khởi tạo và trong quá trình ghi dữ liệu cho bộ nhớ đơn luồng theo sáng chế.

Hình 9 là hình vẽ minh họa việc cập nhật bộ nhớ con trỏ khi mới khởi tạo và trong quá trình đọc dữ liệu cho bộ nhớ đơn luồng theo sáng chế.

Hình 10 là hình vẽ minh họa cấu trúc các khối phần cứng cơ bản trong bộ nhớ chia sẻ đa luồng theo sáng chế.

Hình 11 là hình vẽ minh họa tổ chức con trỏ trong bộ nhớ chia sẻ đa luồng theo sáng chế.

Hình 12 là hình vẽ minh họa chi tiết thuật toán của bộ nhớ chia sẻ đa luồng (Thuật toán 2) theo sáng chế.

Hình 13 là hình vẽ minh họa việc cập nhật bộ nhớ con trỏ ở trạng thái khởi tạo (a), trước ghi dữ liệu (b) và sau ghi dữ liệu (c) cho bộ nhớ đa luồng theo sáng chế.

Hình 14 là hình vẽ minh họa việc cập nhật bộ nhớ con trỏ ở trạng thái khởi tạo (a), trước đọc dữ liệu (b) và sau đọc dữ liệu (c) cho bộ nhớ đa luồng theo sáng chế.

Hình 15 là hình vẽ minh họa mô hình ứng dụng bộ nhớ đơn luồng theo sáng chế trong chuyển mạch gói.

Hình 16 là hình vẽ minh họa mô hình ứng dụng bộ nhớ đa luồng theo sáng chế trong chuyển mạch gói.

Hình 17 là hình vẽ minh họa quá trình thử nghiệm bộ nhớ chia sẻ theo sáng chế trong một thiết bị chuyển mạch gói thực tế.

Mô tả vắn tắt các bảng biểu

Bảng 1: Tài nguyên sử dụng trong bộ nhớ chia sẻ đơn luồng được tổng hợp trên Xilinx Series 7 FPGA.

Bảng 2: Tài nguyên sử dụng trong bộ nhớ chia sẻ đa luồng được tổng hợp trên Xilinx Series 7 FPGA.

Mô tả chi tiết sáng chế

Sau đây là phần trình bày chi tiết của các giải pháp bộ nhớ chia sẻ. Các hình vẽ và các nội dung mô tả đi kèm nhằm mục đích minh họa, giúp giải pháp trở nên dễ hiểu, không có mục đích giới hạn nguyên lý kỹ thuật của các giải pháp.

Các đề xuất nêu trong Hình 3 và Hình 4 đã đưa ra được các giải pháp khả thi cho quá trình cấp phát bộ nhớ động, giúp tiết kiệm bộ nhớ và sử dụng hiệu quả không gian nhớ còn trống và đây cũng là hướng giải pháp của chúng tôi khi đề xuất giải pháp tương tự cho thiết kế bộ nhớ chia sẻ dữ liệu dạng gói ứng dụng trong các thiết bị chuyển mạch gói tốc độ cao. Tuy nhiên, các giải pháp đã mô tả vẫn có thể được cải tiến và khắc phục điểm hạn chế để mang lại hiệu quả tối ưu nhất, thể hiện trên các điểm như sau:

1. Việc liên kết giữa các địa chỉ được thực hiện nhờ một bảng tra (giải pháp sử dụng RAM chia sẻ trên Hình 3) gồm hai cột nhớ, do vậy tổng dung lượng nhớ để đáp ứng cho việc lưu địa chỉ các ô nhớ trống trong bộ nhớ chia sẻ sẽ tương đối lớn. Trong trường hợp sử dụng

danh sách liên kết đơn nhất như trên Hình 4, việc quản lý các địa chỉ liên kết tuy trở nên đơn giản hơn, nhưng phải tiêu tốn tới hai bảng tra và hoàn toàn không có lợi về mặt dung lượng bộ nhớ. Để khắc phục điều này, có thể sử dụng một vùng nhớ duy nhất để phục vụ cho việc lưu trữ các con trỏ tới các địa chỉ liên kết, trong đó nội dung của con trỏ này chính là địa chỉ của con trỏ kế tiếp trong chuỗi các địa chỉ của các ô nhớ còn khả dụng của bộ nhớ chia sẻ. Bộ nhớ sử dụng sẽ là bộ nhớ trong, không cần dùng đến bộ nhớ DRAM hay SRAM bên ngoài.

2. Cơ chế đóng chuỗi kết nối trong giải pháp trên Hình 3 cần sử dụng tới một cờ riêng biệt để đánh dấu phân đoạn gói cuối cùng của gói dữ liệu, tuy đây là một cơ chế hiệu quả và tin cậy, nhưng có thể coi cơ chế sử dụng vùng nhớ hai chiều (bảng RAM hai cột). Trong trường hợp sử dụng vùng nhớ một chiều như đề xuất trong điểm 1, có thể đóng chuỗi liên kết bằng cách cho nội dung của con trỏ cuối cùng của chuỗi trỏ tới chính nó, điều này đủ tạo ra dấu hiệu kết thúc chuỗi nhớ trong khi cơ cấu dữ liệu con trỏ tương đối đơn giản. Trong trường hợp giải pháp như trên Hình 4, cơ chế đóng chuỗi tuy đơn giản, nhưng vùng nhớ sử dụng cho con trỏ địa chỉ ở đây có thể coi như bốn chiều (hai bảng RAM hai cột).

3. Các tham số liên quan của bộ nhớ chia sẻ được đề cập trong hai giải pháp đã nêu bị giới hạn bởi kích thước của khối nhớ dữ liệu trong bộ nhớ chia sẻ và bằng 256 byte; số địa chỉ liên kết trong danh sách liên kết trỏ tới các vùng nhớ đệm của bộ nhớ chia sẻ không được vượt quá một giá trị cố định và giá trị này bằng 6. Các giới hạn này góp phần làm suy giảm khả năng tùy biến và linh hoạt của bộ nhớ chia sẻ, vì vậy nhóm tác giả sáng chế đề xuất giải pháp trong đó các tham số của bộ nhớ có thể tùy biến được ngay từ khi khởi tạo, chẳng hạn kích thước mỗi khối nhớ chia sẻ có thể đặt bằng B, số lượng địa chỉ nhớ được đặt bằng M và hai tham số này có thể điều chỉnh được.

4. Quy trình nhận hoặc truyền dữ liệu gói còn tương đối phức tạp, phải cần tới 4 giai đoạn, bao gồm: lấy liên kết, tạo liên kết, đọc liên kết và giải phóng liên kết. Với quy trình thực hiện khá chặt chẽ nhưng cũng cồng kềnh như vậy, một mặt sẽ gây ra những khó khăn nhất định khi triển khai trên thiết bị phần cứng, mặt khác sẽ ảnh hưởng tới tốc độ truy xuất dữ liệu nói chung, hay nói cách khác sẽ ảnh hưởng tới băng thông tối đa. Nếu có thể đơn giản hóa bớt quy trình, tốc độ trao đổi dữ liệu sẽ được cải thiện, đồng thời cũng giúp tiết kiệm tài nguyên logic, do vậy nhóm nghiên cứu sáng chế đề xuất một cơ chế trao đổi dữ liệu chuyển mạch gói tốc độ cao được đơn giản hóa về mặt quy trình, chỉ bao gồm hai giai đoạn chính: giai đoạn thứ nhất sẽ khởi tạo bộ nhớ chia sẻ theo cách đơn giản nhất có thể; giai đoạn thứ hai thực hiện một trong hai thao tác đọc hoặc ghi dữ liệu, tùy thuộc vào yêu cầu cụ thể từ các khối điều khiển. Chi tiết cụ thể về quy trình thực hiện sẽ được trình bày trong các phần sau.

5. Cả hai giải pháp bộ nhớ chia sẻ (được thể hiện trên Hình 3, Hình 4) không đề cập một cách rõ ràng hoặc tường minh về phương pháp truyền nhận dữ liệu thuộc kiểu đơn luồng hay đa luồng. Việc phân định rõ các trường hợp khác nhau của luồng dữ liệu vào ra tốc độ cao đóng vai trò quan trọng ở chỗ, mỗi kiểu dữ liệu luồng có những đặc điểm khác nhau,

do vậy cần phải có những ưu tiên riêng, chẳng hạn bộ nhớ chia sẻ đơn luồng cần ưu tiên về tính đơn giản, hiệu quả và tiết kiệm tài nguyên, trong khi đó với bộ nhớ chia sẻ đa luồng, ngoài những ưu điểm đã nêu trong tổ chức con trỏ đơn luồng cần quan tâm tới độ trễ tối thiểu. Trong sáng chế đề xuất, chúng tôi sẽ trình bày rõ các giải pháp cụ thể cho mỗi trường hợp này.

Theo phương án thứ nhất, sáng chế đề xuất giải pháp thiết kế bộ nhớ chia sẻ đơn luồng. Các khía cạnh của phương án thứ nhất được mô tả cụ thể dưới đây.

Tổ chức bộ nhớ chia sẻ đơn luồng

Hình 5 minh họa cấu trúc các khối phần cứng cơ bản trong bộ nhớ chia sẻ cho dữ liệu đơn luồng 1. Có ba thành phần chính trong thiết kế này: vùng nhớ chính (*memory pool*) 2 dùng để lưu trữ các gói dữ liệu; bộ nhớ con trỏ (*pointer memory*) 3 dùng để lưu trữ vị trí của các khối dữ liệu bằng một danh sách liên kết; các khối điều khiển cho các thao tác đọc và ghi (máy trạng thái ghi 4 - *WRITE Finite State Machine* (WRITE FSM) và máy trạng thái đọc 5 - *READ Finite State Machine* (READ FSM)). Kích thước của vùng nhớ chính là 2^N x B với N là số bit địa chỉ (tương ứng với 2^N ô nhớ), B là kích thước mỗi ô nhớ trong vùng nhớ chính. Vùng nhớ chính là một bộ nhớ hai cổng để đảm bảo quá trình đọc và ghi dữ liệu được thực thi độc lập. Bộ nhớ con trỏ có kích thước 2^N x N và cũng được tổ chức là một bộ nhớ hai cổng để đảm bảo tốc độ truy cập tối đa cho hai thao tác đọc và ghi dữ liệu. Gói dữ liệu có kích thước tùy ý được chia thành các khối dữ liệu có kích thước cố định B bit. Dữ liệu được ghi vào và đọc ra thông qua giao tiếp chuẩn AMBA AXI4-Stream 6 [14], do đó có thể truyền tải dữ liệu chính xác tới cấp độ byte.

Kiến trúc bộ nhớ lưu trữ dữ liệu dưới dạng chuỗi các con trỏ liên kết

Hình 6 trình bày mô hình tổ chức con trỏ trong bộ nhớ chia sẻ đơn luồng. Một gói dữ liệu được chia thành các khối và được lưu trữ theo thứ tự. Bộ nhớ chia sẻ được thiết kế để vị trí của các khối của một gói dữ liệu có thể được lưu trữ ở bất kỳ vị trí nào trong vùng nhớ chính. Cụ thể, sau khi khối đầu tiên của gói được lưu vào ô nhớ khả dụng đầu tiên trong danh sách ô nhớ liên kết, khối tiếp theo được lưu trong ô nhớ khả dụng (chưa được ghi) kế tiếp và cứ tiếp tục như vậy cho đến khối cuối cùng. Kích thước khối về cơ bản được đặt giống với kích thước kênh dữ liệu hệ thống (B) để giảm độ phức tạp của thiết kế, mặc dù không có giới hạn kỹ thuật nào trong việc chọn kích thước khối ngoại trừ việc tham số này trực tiếp quyết định đến băng thông bộ nhớ.

Vị trí các khối dữ liệu của gói tin ban đầu được tổ chức theo danh sách liên kết, nghĩa là khối thứ i trỏ đến khối thứ (i+1) và khối cuối cùng trỏ đến chính nó như minh họa trong Hình 6 Hình 8 và Hình 9. Với N bit địa chỉ sẽ tạo ra được số lượng địa chỉ bằng 2^N, mỗi địa chỉ có kích thước N bit, do đó tổng số bit nhớ cung cấp cho 2^N địa chỉ bằng Số địa chỉ x Kích thước địa chỉ = 2^N x N. Như vậy, một vùng nhớ chính kích thước 2^N ô nhớ sẽ cần một bộ nhớ con trỏ có

kích thước bằng 2^N x N [bit] tương ứng để quản lý địa chỉ, trong đó giá trị N là tham số cấu hình và có thể thay đổi được. Nội dung của bộ nhớ con trỏ là chuỗi các địa chỉ liên kết, các địa chỉ này trỏ đến các ô nhớ trong vùng nhớ chính nơi lưu trữ gói dữ liệu, trong đó nội dung của ô nhớ con trỏ này là địa chỉ của con trỏ tiếp theo chỉ tới vị trí sẽ ô nhớ dữ liệu sẽ truy cập (đọc hoặc ghi) trong vùng nhớ chính. Đến cuối danh sách liên kết, nội dung của ô nhớ con trỏ và địa chỉ của nó trùng nhau, nghĩa là con trỏ sẽ trỏ đến vị trí của chính ô nhớ đó. Cách tổ chức như vậy sẽ tiết kiệm một bit thông tin cần thiết cho việc báo hiệu kết thúc danh sách.

Độ dài của danh sách liên kết các ô nhớ khả dụng thể hiện trạng thái của bộ nhớ chia sẻ. Do đó, trong bộ đệm hàng đợi con trỏ, khi một gói tin được đọc lại, chỉ duy nhất con trỏ đầu tiên được lưu trữ và kích thước bộ đệm con trỏ thường nhỏ hơn rất nhiều so với bộ nhớ con trỏ. Bộ đệm con trỏ có thể được phân đoạn thành các bộ đệm truy cập theo thứ tự vào trước ra trước (FIFO) phụ thuộc vào tổ chức của phần điều khiển chuyển mạch và các tính năng liên quan đến quản lý ưu tiên (ví dụ, quản lý chất lượng dịch vụ - QoS).

Trong quá trình truy cập đọc và ghi dữ liệu trong vùng nhớ chính, bộ nhớ con trỏ được truy cập và cập nhật tương ứng. Thuật toán chi tiết được mô tả ở như ở dưới đây.

Mô tả chi tiết quá trình khởi tạo và thuật toán đọc ghi bộ nhớ

Quy trình thực thi cụ thể của các thao tác khởi tạo bộ nhớ, đọc và ghi bộ nhớ được mô tả trong Thuật toán 1 (Algorithm 1) trên Hình 7 và minh họa trên Hình 8 và Hình 9.

Khởi tạo bộ nhớ: Ở giai đoạn khởi tạo (Hình 8a), tất cả nội dung trong bộ nhớ con trỏ 1 được ghi lại để ô nhớ con trỏ thứ i 2 trỏ đến địa chỉ thứ (i+1) 3 (i = $1 \div 2^N$-1), ngoại trừ con trỏ cuối cùng 4 sẽ trỏ đến chính nó để kết thúc chuỗi danh sách liên kết 5. Quy trình này yêu cầu 2^N chu kỳ xung nhịp nếu sử dụng truy cập qua một cổng và giảm còn 2^{N-1} nếu truy cập qua hai cổng của bộ nhớ con trỏ. Vì quá trình khởi tạo này không quá quan trọng về thời gian, nên thiết kế sẽ sử dụng truy cập một cổng nhằm giảm độ phức tạp và tiết kiệm tài nguyên logic. Việc khởi tạo về bản chất là tạo một danh sách liên kết ảo gồm các ô nhớ khả dụng (chưa được ghi) từ vị trí số 0 đầu tiên 6 đến ô nhớ vị trí số 2^N-1 cuối cùng 7. Trong cấu trúc điều khiển sử dụng hai con trỏ hệ thống WPT 8 và WPT_last 9 trỏ tới vị trí khả dụng đầu tiên và vị trí khả dụng cuối cùng của danh sách liên kết các ô nhớ khả dụng.

Quá trình ghi dữ liệu: Theo minh họa trên Hình 8b, khi một gói dữ liệu mới đến, khối đầu tiên của gói được cấp phát tại địa chỉ WPT, con trỏ của khối dữ liệu đầu tiên của gói sẽ được lưu vào thanh ghi WPT_first_reg để gửi ra ngoài sau khi thao tác ghi hoàn tất. Tiếp đó, địa chỉ ô nhớ trống tiếp theo trong vùng nhớ chính được truy xuất bằng cách đọc bộ nhớ con trỏ tại địa chỉ WPT. Quá trình được lặp lại cho đến khi gói dữ liệu được lưu hoàn toàn và có tín hiệu từ giao tiếp AXI báo last_data (khối dữ liệu cuối cùng), lúc này một chu kỳ xung nhịp bổ sung được yêu cầu để ghi giá trị cho ô nhớ con trỏ cuối cùng bằng địa chỉ của chính nó và cập nhật danh sách các ô nhớ khả dụng từ vị trí WPT hiện tại 10 đến vị trí WPT_last cuối cùng 11. Đồng thời,

vị trí ghi khối dữ liệu đầu tiên của gói (WPT_first_reg) 12 được gửi ra ngoài cho khối điều khiển. Theo đó, danh sách liên kết trỏ tới các ô nhớ khả dụng được cập nhật động.

Ví dụ minh họa cho quá trình ghi dữ liệu thể hiện trên Hình 8c, một gói dữ liệu cần ghi gồm ba khối dữ liệu sẽ lần lượt được ghi vào các ô nhớ 12, 13, 14 có địa chỉ WPT_0, WPT_1 và WPT_2 được liên kết với nhau, trong đó WPT_2 trỏ tới chính nó để kết thúc chuỗi 15 được ghi. Con trỏ WPT_0 16 sau khi kết thúc ghi sẽ gửi ra ngoài tới bộ đệm con trỏ đọc đầu tiên để truy cập đọc về sau này. Cũng như minh họa, con trỏ ghi WPT sẽ tự động được chuyển tới vị trí được trỏ bởi con trỏ WPT_2 17 trước khi vị trí này 18 được ghi lại giá trị WPT_2 để đánh dấu sự kết thúc chuỗi.

Quá trình đọc dữ liệu: Quá trình đọc từ bộ nhớ cũng yêu cầu truy cập đồng thời vào vùng nhớ chính và bộ nhớ con trỏ và quá trình này sẽ sử dụng phần còn lại của những bộ nhớ trên để truy cập và không xảy ra xung đột giữa đọc và ghi. Như mô tả trong thuật toán trên Hình 7, nếu có một con trỏ đọc hợp lệ nhận được yêu cầu xóa hay không xóa dữ liệu sau khi đọc thì chu kỳ đọc sẽ được kích hoạt. Trong chu kỳ đọc, nội dung ô nhớ đầu tiên trong vùng nhớ chính được đọc và gửi ra, vị trí ô nhớ đọc ra tiếp theo được xác định từ bộ nhớ con trỏ và quá trình này tiếp tục cho đến khi nội dung con trỏ đọc ra bằng chính địa chỉ của nó, báo hiệu kết thúc chuỗi con trỏ liên kết của gói dữ liệu cần đọc. Sau khi kết thúc chu trình đọc, nếu có yêu cầu xóa gói dữ liệu vừa đọc thì nội dung ô nhớ tại vị trí WPT_last trước đây trong bộ nhớ con trỏ sẽ được ghi giá trị con trỏ đầu tiên RPT_0 của chuỗi con trỏ đọc có xóa để nối vùng nhớ khả thi hiện tại với vùng nhớ vừa được đọc ra, kết quả cuối cùng WPT_last sẽ được gán bằng con trỏ cuối cùng của chuỗi con trỏ đọc có xóa.

Hình 9 minh họa một trường hợp đọc có xóa dữ liệu. Gói dữ liệu cần đọc ra được lưu vào các ô nhớ 19, 20, 21 có địa chỉ RPT_0, RPT_1, RPT_2. Khối điều khiển bên ngoài gửi con trỏ RPT_0 22 vào để kích hoạt quá trình đọc có xóa dữ liệu. Các ô nhớ liên quan đến các con trỏ RPT_0, RPT_1, RPT_2 sẽ được đọc ra theo thứ tự như trên. Khi đọc tới địa chỉ RPT_2 23 thì nội dung của ô nhớ tương ứng bằng chính địa chỉ của nó (hay nói cách khác, ô nhớ trỏ tới chính nó) báo hiệu quá trình đọc kết thúc. Vì quá trình đọc có yêu cầu xóa nên sau khi đọc xong, con trỏ WPT_last sẽ trỏ tới ô nhớ chứa địa chỉ RPT_0 24 và sau đó WPT_last 25 sẽ được đặt lại bằng RPT_2, tức là vị trí cuối cùng của chuỗi mới được đọc ra và giải phóng bộ nhớ.

Đánh giá tài nguyên và băng thông của thiết kế bộ nhớ chia sẻ đơn luồng

Thiết kế được mô tả trên ngôn ngữ mô tả phần cứng VHDL (Very High Speed Hardware Description Language), sau đó được mô phỏng kiểm tra trước khi tổng hợp và được hiện thực hóa trên phần cứng khả trình FPGA (Field-Programmable Gate Array). Kết quả tổng hợp cho các cấu hình khác nhau được thể hiện trong Bảng 2. Theo kết quả tổng hợp bộ nhớ chia sẻ trên các nền tảng công nghệ FPGA phổ dụng, thiết kế đề xuất có thể đạt được băng thông dao động từ 50 -138 Gbps tùy theo cấu hình của bộ nhớ (các tham số B và N) và

nền tảng phần cứng (loại chíp FPGA). Bộ nhớ chia sẻ đơn luồng đã được chạy thử nghiệm trên thiết bị phần cứng cụ thể để khẳng định tính đúng đắn của thiết kế.

Theo phương án thứ hai, sáng chế đề xuất giải pháp thiết kế bộ nhớ chia sẻ đa luồng. Các khía cạnh của phương án thứ hai được mô tả cụ thể dưới đây.

Tổ chức bộ nhớ chia sẻ đa luồng

Hình 10 minh họa kiến trúc bộ nhớ chia sẻ dữ liệu dạng gói hỗ trợ đa luồng đồng thời 1. Các thành phần chính trong thiết kế này bao gồm: vùng nhớ chính (*memory pool*) 2 dùng để lưu trữ các gói dữ liệu; bộ nhớ con trỏ (*pointer memory*) 3 dùng để tổ chức con trỏ theo kiểu danh sách liên kết, gồm bộ nhớ con trỏ đọc và bộ nhớ con trỏ ghi; các khối điều khiển cho các trạng thái đọc và ghi 4 (WRITE *Finite State Machine* - WRITE FSM, READ *Finite State Machine* - READ FSM); các thanh ghi hệ thống 5 WPT_reg, WPT_first_reg, RPT_reg và RPT_first_reg dùng để quản lý các con trỏ trong quá trình đọc ghi, đây là điểm khác biệt so với bộ nhớ chia sẻ đơn luồng khi không có các thanh ghi hệ thống này. Bộ nhớ chia sẻ đơn luồng cần có các khối thực hiện gom và phân phối các luồng thông tin ở đầu vào và đầu ra, còn bộ nhớ chia sẻ đa luồng thì không. Ngoài ra, bộ nhớ con trỏ trong bộ nhớ chia sẻ đa luồng được cung cấp riêng biệt thành bộ nhớ con trỏ đọc và bộ nhớ con trỏ ghi, trong khi đó trong bộ nhớ chia sẻ đơn luồng chỉ có một bộ nhớ con trỏ dùng chung cho cả hai quá trình đọc và ghi dữ liệu. Từ những khác biệt như vậy, bộ nhớ chia sẻ đa luồng có khả năng phục vụ nhiều luồng thông tin độc lập trực tiếp 6 so với khả năng phục vụ chỉ đơn luồng dữ liệu dồn ghép ở đầu vào của bộ nhớ chia sẻ đơn luồng.

Kích thước của vùng nhớ chính là 2^N x B với N là số bit địa chỉ của bộ nhớ, nghĩa là sẽ có 2^N ô nhớ, mỗi ô nhớ có kích thước B bit. Vùng nhớ chính được thiết kế là một bộ nhớ hai cổng để đảm bảo quá trình đọc dữ liệu và ghi dữ liệu được thực thi trên hai cổng độc lập với nhau Tương tự như vậy, bộ nhớ con trỏ cho phần đọc và phần ghi đều có kích thước 2^N x N và cũng được tổ chức là một bộ nhớ hai cổng để đảm bảo tốc độ truy cập tối đa. Gói dữ liệu kích thước tùy ý được chia thành các khối dữ liệu có kích thước cố định B bit. Dữ liệu được ghi vào và đọc ra thông qua M luồng theo giao tiếp chuẩn AMBA AXI4-Stream, do đó có thể truyền tải chính xác tới cấp độ byte.

Kiến trúc bộ nhớ lưu trữ dữ liệu dưới dạng các chuỗi con trỏ liên kết

Hình 11 trình bày mô hình tổ chức con trỏ trong bộ nhớ chia sẻ đa luồng. Một gói dữ liệu được chia thành các khối và được lưu trữ theo thứ tự. Bộ nhớ chia sẻ được thiết kế để vị trí của các khối của một gói dữ liệu có thể được lưu trữ ở bất kỳ vị trí nào trong vùng nhớ chính. Cụ thể, sau khi khối đầu tiên của gói được lưu vào ô nhớ khả dụng đầu tiên trong danh sách ô nhớ liên kết, khối tiếp theo được lưu trong ô nhớ khả dụng (chưa được ghi) kế tiếp và cứ tiếp tục như vậy cho đến khối cuối cùng. Kích thước khối về cơ bản được đặt giống với

kích thước kênh dữ liệu hệ thống (B) để giảm độ phức tạp của thiết kế, mặc dù không có giới hạn kỹ thuật nào trong việc chọn kích thước khối ngoại trừ việc tham số này trực tiếp quyết định đến băng thông bộ nhớ.

Vị trí các khối dữ liệu của gói tin ban đầu được tổ chức theo danh sách liên kết, nghĩa là khối thứ i trỏ đến khối thứ (i+1) và khối cuối cùng trỏ đến chính nó như minh họa trong Hình 6. Một vùng nhớ có 2^N ô nhớ sẽ cần một vùng nhớ con trỏ có kích thước là 2^N x N tương ứng. Bộ nhớ con trỏ đọc lưu danh sách liên kết cho mỗi gói dữ liệu mà nội dung của mỗi ô nhớ của nó là địa chỉ của con trỏ tiếp theo chỉ tới vị trí sẽ được truy cập trong vùng nhớ chính. Đến cuối danh sách được liên kết, nội dung của ô nhớ con trỏ đọc và địa chỉ của nó trùng nhau, tức là con trỏ sẽ trỏ đến vị trí của chính ô nhớ đó. Cách thức tổ chức như vậy sẽ tiết kiệm một bit thông tin cần thiết cho việc báo hiệu kết thúc danh sách, cũng như đảm bảo việc truy xuất gói tin từ bộ nhớ chỉ cần thông qua con trỏ đầu tiên của chuỗi liên kết. Cụ thể, trong bộ đệm con trỏ, khi một gói tin được đọc lại, chỉ duy nhất con trỏ đầu tiên được lưu trữ và bộ đệm thường nhỏ hơn rất nhiều so với bộ nhớ con trỏ. Bộ đệm con trỏ có thể được phân đoạn thành các bộ đệm có thứ tự (FIFO) phụ thuộc tổ chức của phần điều khiển chuyển mạch và các tính năng liên quan đến quản lý ưu tiên (ví dụ, quản lý chất lượng dịch vụ QoS).

Mô tả chi tiết quá trình khởi tạo và thuật toán đọc ghi bộ nhớ

Quy trình thực thi cụ thể của các thao tác khởi tạo bộ nhớ, đọc và ghi bộ nhớ được mô tả trong Thuật toán 2 (Algorithm 2) trên Hình 12 và các ví dụ về thay đổi trạng thái con trỏ đọc khi khởi tạo, đọc và ghi thể hiện trên Hình 13 và Hình14.

Khởi tạo bộ nhớ: Mô tả thuật toán của thao tác khởi tạo bộ nhớ được trình bày trên Hình 12 và hình ảnh minh họa cho thuật toán này thể hiện trên Hình 13a. Ở giai đoạn khởi tạo, nội dung trong bộ nhớ con trỏ 1 được ghi lại để ô nhớ con trỏ thứ i 2 trỏ đến địa chỉ thứ (i+1) 3 (i = $1 \div 2^N$-1), ngoại trừ con trỏ cuối cùng 4 sẽ trỏ đến chính nó để kết thúc chuỗi danh sách liên kết 5. Thiết kế đề xuất sử dụng một cổng nhằm giảm độ phức tạp và tiết kiệm tài nguyên logic. Quá trình khởi tạo về bản chất tạo thành một danh sách liên kết ảo trên bộ nhớ con trỏ đọc gồm các ô nhớ khả dụng (chưa được ghi) từ vị trí số 0 đầu tiên 6 đến ô nhớ vị trí số 2^N-1 cuối cùng 7. Trong cấu trúc điều khiển sử dụng hai con trỏ hệ thống WPT 8 và WPT_last 9 trỏ tới vị trí khả dụng đầu tiên và vị trí khả dụng cuối cùng của danh sách liên kết các ô nhớ khả dụng.

Quá trình ghi dữ liệu: Mô tả thuật toán của thao tác khởi tạo bộ nhớ được trình bày trên Hình 12. Khi một gói dữ liệu thuộc luồng thứ k mới đến, khối đầu tiên của gói được cấp phát trong vùng nhớ chính tại vị trí khả dụng hiện tại quy định trong bộ nhớ con trỏ ghi, vị trí trống tiếp theo được truy xuất bằng cách đọc bộ nhớ con trỏ ghi, đồng thời con trỏ vừa được ghi sẽ được gửi sang bộ nhớ con trỏ đọc để cập nhật trạng thái con trỏ cho luồng thứ k.

Có thể xảy ra các trường hợp như sau khi kết thúc quá trình ghi một ô nhớ:

- Nếu đây là ô nhớ đầu tiên của gói thì con trỏ của ô dữ liệu này sẽ được lưu vào thanh ghi WPT_first_reg(k) đồng thời cập nhật vào thanh ghi WPT_reg(k).
- Nếu đây không phải là ô nhớ dữ liệu đầu tiên cũng như cuối cùng thì địa chỉ của ô nhớ vừa được ghi trong vùng nhớ chính sẽ được ghi vào ô nhớ trong bộ nhớ con trỏ đọc tại địa chỉ WPT_reg(k). Còn thanh ghi WPT_reg(k) sau đó sẽ được cập nhật giá trị bằng địa chỉ ô nhớ vừa ghi.
- Nếu đây là ô nhớ dữ liệu cuối cùng thì giá trị WPT_reg(k) sẽ được đẩy ra ngoài cho bộ đệm con trỏ, trường hợp này trùng với trường hợp thứ nhất nếu gói dữ liệu được ghi chỉ bao gồm một khối dữ liệu duy nhất.

Sau quy trình này, khối điều khiển ghi sẽ chuyển sang phục vụ gói dữ liệu của luồng kế tiếp, luồng hiện tại dù vẫn còn dữ liệu cũng sẽ phải chờ cho tới khi tất cả các cổng khác được phục vụ mới quay trở lại đến lượt mình. Quá trình ghi sẽ tiếp tục như vậy và tất cả các luồng vào sẽ được phục vụ gần như đồng thời với độ trễ tối thiểu.

Hình 13b và Hình 13c minh họa quá trình ghi phục vụ cho hai luồng dữ liệu. Trước khi ghi các ô mầu trắng nối từ WPT 26 đến WPT_last 27 là các ô nhớ khả dụng. Khi khối dữ liệu đầu tiên của luồng số 0 (Khối_0_0) tới, nó sẽ được ghi vào vị trí ô nhớ hiện tại 28 được trỏ bởi con trỏ WPT, đồng thời vị trí này cũng được gửi vào thanh ghi WPT_first_reg(0) và WPT_reg(0). Sau đó khối điều khiển chuyển sang phục vụ luồng số 1, hiện cũng đang có một khối dữ liệu đầu tiên chờ được ghi (Khối_1_0), khối này sẽ được ghi vào ô nhớ khả dụng tiếp theo 31, đồng thời vị trí ô nhớ đó sẽ được gửi vào thanh ghi WPT_first_reg(1) và WPT_reg (1). Tiếp theo, luồng số 0 lại được phục vụ trở lại, Khối_0_1 được ghi vào ô nhớ khả dụng tiếp theo và đây là ô nhớ cuối cùng 29 của gói dữ liệu. Khi kết thúc ghi sẽ có hai thao tác được thực hiện: ô nhớ có địa chỉ WPT_reg(0) trong bộ nhớ con trỏ đọc sẽ được ghi giá trị bằng địa chỉ ô nhớ Khối_0_1 (ghi đồng thời trong chu kỳ thực hiện ghi dữ liệu). Sau đó ô nhớ có địa chỉ của Khối_0_1 trong bộ nhớ con trỏ đọc sẽ được cập nhật bằng chính nó 30 để báo hiệu kết thúc, đồng thời giá trị ghi trong WPT_first_reg(0) được gửi ra ngoài bộ đệm con trỏ. Với luồng số 1, mọi thao tác được thực hiện tương tự, chỉ khác là gói này kết thúc ở khối dữ liệu thứ 3 tức là Khối_1_2 32.

Quá trình đọc dữ liệu: Mô tả thuật toán của thao tác khởi tạo bộ nhớ được trình bày trên Hình 12 và hình ảnh minh họa cho thuật toán này thể hiện trên Hình 14. Tương tự như quá trình ghi, quá trình đọc từ bộ nhớ sẽ được thực thi đồng thời, nghĩa là ở cùng một thời điểm sẽ có tối đa M con trỏ đọc được gửi vào khối điều khiển đọc.

Có hai thao tác cơ bản trong quá trình đọc:

Thao tác nhận con trỏ đọc đầu tiên: Giả sử có một con trỏ đọc RPT_first(k) được gửi vào khối điều khiển đọc, con trỏ này sẽ được lưu vào thanh ghi RPT_first_reg(k) tương ứng trong danh sách M con trỏ đọc và khối điều khiển đọc sẽ có một cờ ghi nhận yêu cầu đọc cho luồng thứ k. Thao tác này sẽ thực hiện liên tục và độc lập so với thao tác đọc dữ liệu.

Thao tác đọc dữ liệu: Tại thời điểm luồng thứ k được phục vụ đọc sẽ có thể xảy ra các tình huống sau như trong mô tả chi tiết thuật toán (Algorithm 2) trên Hình 12.

- Nếu là đọc lần đầu RPT_first_reg (k) thì địa chỉ này được gửi ra khối điều khiển đọc ô dữ liệu tương ứng trong vùng nhớ chính, đồng thời con trỏ này được ghi vào thanh ghi RPT_reg(k) của bộ nhớ con trỏ ghi để đọc ra vị trí tiếp theo.
- Nếu là đọc lần cuối (nội dung của con trỏ đọc trùng với địa chỉ của nó) và có lệnh xóa dữ liệu thì nội dung ô nhớ tại địa chỉ WPT_last trong bộ nhớ con trỏ ghi sẽ được cập nhật bằng RPT_first_reg(k), đồng nghĩa với không gian nhớ khả dụng sẽ được mở rộng thêm bằng kích thước của gói vừa được đọc ra. Nếu không có yêu cầu xóa thì quy trình đọc cho luồng thứ k sẽ hoàn tất mà không có sự thay đổi bên trong bộ nhớ con trỏ ghi.
- Nếu không phải đọc lần cuối thì giá trị địa chỉ ô cần đọc được lấy từ thanh ghi RPT_reg(k) lúc này đang trỏ tới vị trí cần đọc còn giá trị thanh ghi RPT_reg(k) sẽ được cập nhật bằng giá trị của ô nhớ đọc ra từ bộ nhớ con trỏ đọc chính từ địa chỉ RPT_reg(k).

Sau khi hoàn tất đọc một ô dữ liệu cho luồng, khối điều khiển đọc sẽ xác định luồng tiếp theo cần được phục vụ và ngầm định theo thứ tự lần lượt (Round-Robin) với các thao tác giống như mô tả ở trên.

Hình 14 mô tả một ví dụ của quá trình đọc đồng thời hai luồng dữ liệu từ bộ nhớ chia sẻ. Hai con trỏ đọc RPT_0 33 và RPT_1 34 được gửi vào khối điều khiển đọc tại các vị trí tương ứng. Quá trình đọc được kích hoạt và sau 4 thao tác đọc, các ô nhớ Khối_0_0 35, Khối_0_1 36 của luồng số 0 được đọc ra và kết thúc có xóa, bộ nhớ con trỏ ghi khả dụng sẽ được cập nhật tương ứng để nối vào các vị trí này. Cụ thể, con trỏ WPT_last trước đây 37 được nối tới vị trí được trỏ bởi RPT_0 38, cuối cùng con trỏ WPT_last hiện tại 39 được cập nhật trỏ tới địa chỉ của ô nhớ Khối_0_1. Với luồng số 1 cũng có hai dữ liệu ra nhưng quá trình đọc chưa kết thúc nên mặc dù các ô nhớ 40, 41, 42 chứa Khối_1_0, Khối_1_1 và Khối_1_2 đã được đọc nhưng vẫn chưa được nối vào danh sách con trỏ khả dụng.

Cơ chế phục vụ đa luồng đồng thời

Điểm khác biệt của bộ nhớ chia sẻ đa luồng so với bộ nhớ chia sẻ đơn luồng là khả năng phục vụ đồng thời cả luồng vào và luồng ra. Trong bộ nhớ chia sẻ đơn luồng, một gói dữ liệu sẽ được lưu trữ hoàn toàn trước khi gói dữ liệu mới được phục vụ, nghĩa là các gói dữ liệu được phục vụ luân phiên theo đơn vị gói. Trong bộ nhớ chia sẻ đa luồng, các luồng được phục vụ luân phiên theo đơn vị khối dữ liệu và có thể hiểu theo cách khác, các gói dữ liệu của các luồng được phục vụ theo cách song song.

Đánh giá tài nguyên và băng thông của thiết kế bộ nhớ chia sẻ đa luồng

Thiết kế của bộ nhớ chia sẻ đa luồng được mô tả bằng ngôn ngữ mô tả phần cứng VHDL (Very High Speed Hardware Description Language), được mô phỏng kiểm tra trước khi

tổng hợp và hiện thực hóa trên phần cứng khả trình FPGA (Field-Programmable Gate Array). Kết quả tổng hợp cho các cấu hình khác nhau được thể hiện trong Bảng 2. Thiết kế có thể đạt băng thông định danh một chiều 240 Gbps (cho cấu hình B = 1024 bit, bộ nhớ 1 Mb, chip FPGA Virtex-7 XC7VX550T) và trong trường hợp thấp nhất đạt 85 Gbps (cho cấu hình B = 1024 bit, bộ nhớ 1 Mb, chip FPGA Artix-7 XC7A200T).

Mô hình ứng dụng của bộ nhớ chia sẻ

Bộ nhớ chia sẻ đơn luồng và bộ nhớ chia sẻ đa luồng được ứng dụng trong mô hình chuyển mạch như minh họa trên Hình 15 và Hình 16.

Hình 15 trình bày mô hình ứng dụng cho bộ nhớ chia sẻ đơn luồng. Trong mô hình này, cần một khối tập trung luồng (STREAM CONCENTRATOR) để gom các luồng từ các giao diện vật lý khác nhau đi tới các khối điều khiển truy cập đường truyền RX MAC theo giao thức AXI4-Stream thành một luồng có kênh dữ liệu lớn (Ví dụ, từ các luồng 8 bit thành một luồng 512 bit), sau đó đẩy vào bộ nhớ chia sẻ đơn luồng. Các thông tin mào đầu (header) của các gói tin sẽ được đưa vào khối phân tích gói (Packet Parser) để phân tích. Bộ nhớ chia sẻ sẽ giao tiếp với bộ đệm gói và chịu sự điều khiển của phần điều khiển đọc ghi dữ liệu. Dữ liệu ra khỏi bộ nhớ sẽ đi qua khối phân phối luồng (STREAM DISTRIBUTOR) có chức năng phân tách các gói tin có đính kèm địa chỉ cổng đầu ra và gửi tới khối điều khiển truy cập đường truyền TX MAC theo giao thức AXI4-Stream. Trong kiến trúc bộ nhớ chia sẻ, cần phải có các bộ đệm cục bộ cho các luồng vào ra để hạn chế mất gói tin.

Hình 16 trình bày mô hình ứng dụng cho bộ nhớ chia sẻ đa luồng. Dữ liệu từ khối điều khiển truy cập đường truyền RX MAC chỉ cần được đẩy lên kênh dữ liệu lớn hơn (Ví dụ, từ 8 bit lên 512 bit) trước khi được đẩy song song vào trong bộ nhớ chia sẻ. Tuy nhiên, việc chuyển đổi kích thước kênh dữ liệu không đòi hỏi phải có bộ đệm cục bộ kích thước lớn mà chỉ cần bộ đệm kích thước tối thiểu bằng 2xB bit. Quy trình phân tích gói tin cũng được thực thi qua khối phân tích gói hoàn toàn tương tự như trong trường hợp của bộ nhớ chia sẻ đơn luồng. Tại đầu ra, các luồng từ bộ nhớ chia sẻ được gửi thẳng tới khối điều khiển truy cập đường truyền TX MAC.

Các thiết kế bộ nhớ chia sẻ đơn luồng và bộ nhớ chia sẻ đa luồng có thể được ứng dụng trong các bộ định tuyến cũng như một số thiết bị chuyển mạch thuộc Lớp 2, Lớp 3 và một trong số đó được ứng dụng trong thiết bị được thể hiện trên Hình 17.

Theo phương án khác, sáng chế còn đề xuất lõi chuyển mạch thông tin số dạng gói có sử dụng các bộ nhớ chia sẻ đơn luồng và đa luồng như nêu trên. Ở đây, lõi chuyển mạch số được hiểu là khối xử lý dữ liệu với đầu vào là một luồng hoặc nhiều luồng thông tin dạng gói có băng thông lớn. Khối xử lý này căn cứ vào dữ liệu trong phần mào đầu (*header*) của gói tin, nơi chứa các thông tin quản lý như địa chỉ MAC, địa chỉ IP để gửi gói tin ra ngoài với độ trễ chấp nhận được (tùy theo ứng dụng cụ thể). Để đảm bảo băng thông và hiệu suất của hệ thống,

tốc độ xử lý (ghi và đọc) gói tin trong bộ nhớ chia sẻ cũng như tốc độ tính toán chuyển mạch đóng vai trò quan trọng.

Hiệu quả đạt được bởi sáng chế

- Việc tổ chức con trỏ đơn luồng đảm bảo tính chặt chẽ về mặt tổ chức, tính chính xác về mặt truyền tải dữ liệu và tính đơn giản cho việc thực thi thiết kế trên phần cứng. Bộ nhớ chia sẻ cho dữ liệu gói đơn luồng là một thiết kế đơn giản nhưng hiệu quả và tiết kiệm tài nguyên cho các thiết kế có số lượng luồng giao tiếp không lớn.

- Việc tổ chức hệ thống con trỏ đa luồng cho phép một bộ nhớ chia sẻ có thể phục vụ đồng thời nhiều luồng dữ liệu với độ trễ tối thiểu bên cạnh những ưu điểm đã nêu trong tổ chức con trỏ đơn luồng. Việc tổ chức lưu trữ đa luồng cho phép bỏ qua các bộ đệm đầu vào và đầu ra cho các luồng dữ liệu khác nhau. Bộ nhớ chia sẻ cho dữ liệu gói đa luồng là một thiết kế đơn giản nhưng hiệu quả và tiết kiệm tài nguyên cho các thiết kế có nhiều luồng giao tiếp.

- Một ưu điểm nổi bật của sáng chế là khả năng cho phép tích hợp các thiết kế chuyển mạch trên một chip đơn mà không cần phải sử dụng tới bộ nhớ ngoài. Với kiến trúc hiện tại, bộ nhớ chia sẻ đơn luồng và bộ nhớ chia sẻ đa luồng có thể ứng dụng trong các thiết bị chuyển mạch có băng thông từ hàng chục đến hàng trăm Gbps mà chỉ cần một dung lượng bộ nhớ chia sẻ kích thước rất nhỏ cỡ vài Mb (1-4 Mb). Ngoài ra, một ưu điểm nữa của bộ nhớ chia sẻ đơn luồng và đa luồng là không giới hạn kích thước của gói tin ghi vào và đọc ra.

YÊU CẦU BẢO HỘ

1. Bộ nhớ chia sẻ đơn luồng cho dữ liệu dạng gói ứng dụng trong các thiết bị chuyển mạch gói tốc độ cao, bộ nhớ có cấu trúc bao gồm:

vùng nhớ chính (*memory pool*) để lưu trữ các gói dữ liệu, vùng nhớ chính có kích thước 2^N x B trong đó N là số bit địa chỉ của bộ nhớ, 2^N là số lượng ô nhớ, B là kích thước mỗi ô nhớ theo đơn vị bit, vùng nhớ chính được thiết kế là một bộ nhớ 02 cổng để đảm bảo quá trình đọc dữ liệu và ghi dữ liệu được thực thi độc lập;

bộ nhớ con trỏ (*pointer memory*) để lưu trữ vị trí các khối (phân đoạn gói) của các gói dữ liệu trong vùng nhớ chính, bộ nhớ con trỏ có kích thước 2^N x N, bộ nhớ con trỏ được thiết kế là một bộ nhớ 02 cổng để đảm bảo tốc độ truy cập tối đa cho phần con trỏ đọc và ghi;

các khối điều khiển cho thao tác đọc và thao tác ghi, gồm máy trạng thái ghi (*write finite state machine*) và máy trạng thái đọc (*read finite state machine*); trong đó:

vùng nhớ chính của bộ nhớ chia sẻ đơn luồng được tổ chức lưu trữ dữ liệu dưới dạng chuỗi các con trỏ liên kết trong bộ nhớ con trỏ, trong đó:

mỗi gói dữ liệu có kích thước tùy ý được chia thành các khối (phân đoạn gói) có kích thước cố định B bit và được lưu trữ ở bất kỳ vị trí nào trong vùng nhớ chính, dữ liệu được ghi vào và đọc ra thông qua một luồng theo giao tiếp chuẩn AMBA AXI4-Stream, do đó có thể truyền tải chính xác tới cấp độ byte,

sau khi khối đầu tiên của một gói được lưu vào ô nhớ trống đầu tiên của chuỗi ô nhớ khả dụng, khối tiếp theo sẽ được lưu vào ô nhớ khả dụng (chưa được ghi) kế tiếp trong vùng nhớ chính và cứ như vậy cho đến khối cuối cùng,

chuỗi ô nhớ khả dụng trong vùng nhớ chính có địa chỉ được trỏ bởi chuỗi con trỏ liên kết trong bộ nhớ chia sẻ, nội dung của mỗi ô nhớ trong bộ nhớ chia sẻ là địa chỉ của con trỏ kế tiếp trỏ tới vị trí sẽ được truy cập (đọc hoặc ghi) trong vùng nhớ chính,

dấu hiệu kết thúc chuỗi con trỏ liên kết là khi nội dung của ô nhớ trong bộ nhớ con trỏ và địa chỉ của ô nhớ này trùng nhau và cách tổ chức này giúp tiết kiệm bộ nhớ cho việc báo hiệu kết thúc danh sách con trỏ liên kết cũng như đảm bảo việc truy xuất gói tin từ bộ nhớ chỉ cần thông qua con trỏ đầu tiên của chuỗi,

kích thước khối được đặt giống với kích thước kênh dữ liệu hệ thống (B) để giảm bớt độ phức tạp của thiết kế,

trong bộ đệm con trỏ, khi một gói tin được đọc lại, chỉ duy nhất con trỏ đầu tiên được lưu trữ, bộ đệm con trỏ có thể được phân đoạn thành các bộ đệm truy cập theo thứ tự vào trước ra trước (first in first out (FIFO)) phụ thuộc vào tổ chức của phần điều khiển chuyển mạch và các tính năng liên quan đến quản lý ưu tiên (quản lý chất lượng dịch vụ QoS);

chuyển mạch gói tốc độ cao trong bộ nhớ chia sẻ đơn luồng có cơ chế trao đổi dữ liệu bao gồm quá trình khởi tạo bộ nhớ, quá trình đọc dữ liệu và quá trình ghi dữ liệu, trong đó:

quá trình khởi tạo bộ nhớ tạo một danh sách liên kết ban đầu gồm các ô nhớ khả dụng (chưa được ghi) từ vị trí 0 (đầu tiên) đến vị trí 2^N-1 (cuối cùng), sử dụng hai con trỏ hệ thống WPT và WPT_last trỏ tới vị trí khả dụng đầu tiên và vị trí khả dụng cuối cùng của danh sách liên kết các ô nhớ khả dụng,

quá trình ghi dữ liệu lưu các gói dữ liệu vào các ô nhớ khả dụng trong danh sách liên kết từ vị trí khả dụng đầu tiên tới vị trí khả dụng cuối cùng đồng thời cập nhật động danh sách các ô nhớ khả dụng từ vị trí đầu tiên WPT đến vị trí cuối cùng WPT_last, sau khi hoàn tất con trỏ của khối dữ liệu đầu tiên của gói sẽ được lưu vào thanh ghi WPT_first_reg để gửi ra ngoài,

quá trình đọc dữ liệu đọc và gửi nội dung của ô nhớ đầu tiên của gói dữ liệu trong vùng nhớ chính ra ngoài và lặp lại quá trình này cho các ô nhớ kế tiếp cho đến khi nội dung con trỏ đọc ra bằng chính địa chỉ của nó báo hiệu kết thúc chuỗi con trỏ liên kết, nếu có yêu cầu xóa gói dữ liệu vừa đọc thì nội dung ô nhớ tại vị trí WPT_last trước đây trong bộ nhớ con trỏ sẽ được ghi giá trị con trỏ đầu tiên RPT_0 của chuỗi con trỏ đọc có xóa để nối vùng nhớ khả thi hiện tại với vùng nhớ vừa được đọc ra, kết quả cuối cùng WPT_last sẽ được gán bằng con trỏ cuối cùng của chuỗi con trỏ đọc có xóa;

cấu trúc bộ nhớ chia sẻ được mô tả bằng ngôn ngữ mô tả phần cứng VHDL (Very High Speed Hardware Description Language) và được thực thi trên phần cứng khả trình FPGA (Field-Programmable Gate Array) trong đó băng thông của thiết kế đề xuất có thể đạt 50 Gbps - 138 Gbps tùy theo cấu hình của bộ nhớ (B và N) và nền tảng phần cứng (loại chíp FPGA).

2. Bộ nhớ chia sẻ đa luồng cho dữ liệu dạng gói ứng dụng trong các thiết bị chuyển mạch gói tốc độ cao, bộ nhớ có cấu trúc bao gồm:

vùng nhớ chính (*memory pool*) để lưu trữ các gói dữ liệu, vùng nhớ chính có kích thước 2^N x B trong đó N là số bit địa chỉ của bộ nhớ, 2^N là số lượng ô nhớ, B là kích thước mỗi ô nhớ theo đơn vị bit, vùng nhớ chính được thiết kế là một bộ nhớ 02 cổng để đảm bảo quá trình đọc dữ liệu và ghi dữ liệu được thực thi độc lập;

bộ nhớ con trỏ ghi (*write pointer memory*) phục vụ cho việc tổ chức con trỏ ghi như một danh sách liên kết độc lập, bộ nhớ con trỏ ghi có kích thước 2^N x N và được tổ chức là một bộ nhớ 2 cổng để đảm bảo tốc độ truy cập tối đa cho phần con trỏ ghi, khác biệt so với bộ nhớ chia sẻ đơn luồng khi chỉ có một con trỏ dùng chung cho cả hai quá trình đọc và ghi dữ liệu;

bộ nhớ con trỏ đọc (*read pointer memmory)* phục vụ cho việc tổ chức con trỏ đọc như một danh sách liên kết độc lập, bộ nhớ con trỏ đọc có kích thước 2^N x N và được tổ chức là một bộ nhớ 2 cổng để đảm bảo tốc độ truy cập tối đa cho phần con trỏ đọc, khác biệt so với

bộ nhớ chia sẻ đơn luồng khi chỉ có một con trỏ dùng chung cho cả hai quá trình đọc và ghi dữ liệu;

các khối điều khiển cho thao tác đọc và cho thao tác ghi, gồm máy trạng thái ghi (*write finite state machine*) và máy trạng thái đọc (*read finite state machine*);

hệ thống các thanh ghi trạng thái WPT_reg, RPT_reg, WPT_first_reg, RPT_first_reg để quản lý các con trỏ trong quá trình đọc ghi, khác biệt so với bộ nhớ chia sẻ đơn luồng khi không có các thanh ghi hệ thống này; trong đó:

vùng nhớ chính của bộ nhớ chia sẻ đa luồng được tổ chức lưu trữ dữ liệu dưới dạng chuỗi các con trỏ liên kết trong bộ nhớ con trỏ, trong đó:

mỗi gói dữ liệu có kích thước tùy ý được chia thành các khối (phân đoạn gói) có kích thước cố định B bit và được lưu trữ ở bất kỳ vị trí nào trong vùng nhớ chính, dữ liệu được ghi vào và đọc ra thông qua M luồng theo giao tiếp chuẩn AMBA AXI4-Stream, do đó có thể truyền tải chính xác tới cấp độ byte,

sau khi khối đầu tiên của một gói được lưu vào ô nhớ trống đầu tiên của chuỗi ô nhớ khả dụng, khối tiếp theo sẽ được lưu vào ô nhớ khả dụng (chưa được ghi) kế tiếp trong vùng nhớ chính và cứ như vậy cho đến khối cuối cùng,

chuỗi ô nhớ khả dụng trong vùng nhớ chính có địa chỉ được trỏ bởi chuỗi con trỏ liên kết trong bộ nhớ con trỏ đọc, nội dung của mỗi ô nhớ trong bộ nhớ con trỏ đọc là địa chỉ của con trỏ kế tiếp trỏ tới vị trí sẽ được truy cập trong vùng nhớ chính,

dấu hiệu kết thúc chuỗi con trỏ liên kết là khi nội dung của ô nhớ trong bộ nhớ con trỏ đọc và địa chỉ của ô nhớ này trùng nhau và cách tổ chức này giúp tiết kiệm bộ nhớ cho việc báo hiệu kết thúc danh sách con trỏ liên kết cũng như đảm bảo việc truy xuất gói tin từ bộ nhớ chỉ cần thông qua con trỏ đầu tiên của chuỗi,

kích thước khối được đặt giống với kích thước kênh dữ liệu hệ thống (B) để giảm bớt độ phức tạp của thiết kế,

trong bộ đệm con trỏ, khi một gói tin được đọc lại, chỉ duy nhất con trỏ đầu tiên được lưu trữ, bộ đệm con trỏ có thể được phân đoạn thành các bộ đệm truy cập theo thứ tự vào trước ra trước (first in first out (FIFO)) phụ thuộc vào tổ chức của phần điều khiển chuyển mạch và các tính năng liên quan đến quản lý ưu tiên (quản lý chất lượng dịch vụ QoS);

chuyển mạch gói tốc độ cao trong bộ nhớ chia sẻ đa luồng có cơ chế trao đổi dữ liệu bao gồm quá trình khởi tạo bộ nhớ, quá trình đọc dữ liệu và quá trình ghi dữ liệu, trong đó:

quá trình khởi tạo bộ nhớ tạo một danh sách liên kết ban đầu gồm các ô nhớ khả dụng (chưa được ghi) từ vị trí 0 (đầu tiên) đến vị trí 2^N-1 (cuối cùng), sử dụng hai con trỏ hệ thống WPT và WPT_last trỏ tới vị trí khả dụng đầu tiên và vị trí khả dụng cuối cùng của danh sách liên kết các ô nhớ khả dụng,

quá trình ghi dữ liệu cho gói dữ liệu thuộc luồng dữ liệu thứ k mới đến được bắt đầu bằng việc cấp phát ô nhớ cho khối đầu tiên của gói này trong vùng nhớ chính tại vị trí khả dụng hiện tại quy định trong bộ nhớ con trỏ ghi, vị trí trống tiếp theo được truy xuất bằng cách đọc bộ nhớ con trỏ ghi, đồng thời con trỏ vừa được ghi sẽ được gửi sang bộ nhớ con trỏ đọc để cập nhật trạng thái con trỏ cho luồng thứ k, sau đó khối điều khiển ghi sẽ chuyển sang phục vụ khối đầu tiên trong gói dữ liệu của luồng kế tiếp, luồng thứ k hiện tại dù vẫn còn dữ liệu cũng phải chờ cho tới khi tất cả các luồng khác được phục vụ mới quay trở lại đến lượt mình, khác biệt ở chỗ, quá trình ghi dữ liệu phục vụ các luồng luân phiên theo đơn vị khối dữ liệu, nghĩa là các gói dữ liệu của các luồng phục vụ song song, khác với bộ nhớ chia sẻ đơn luồng phục vụ luồng thông tin đơn theo đơn vị gói dữ liệu, nghĩa là một gói dữ liệu sẽ được lưu trữ hoàn toàn trước khi gói dữ liệu mới được phục vụ,

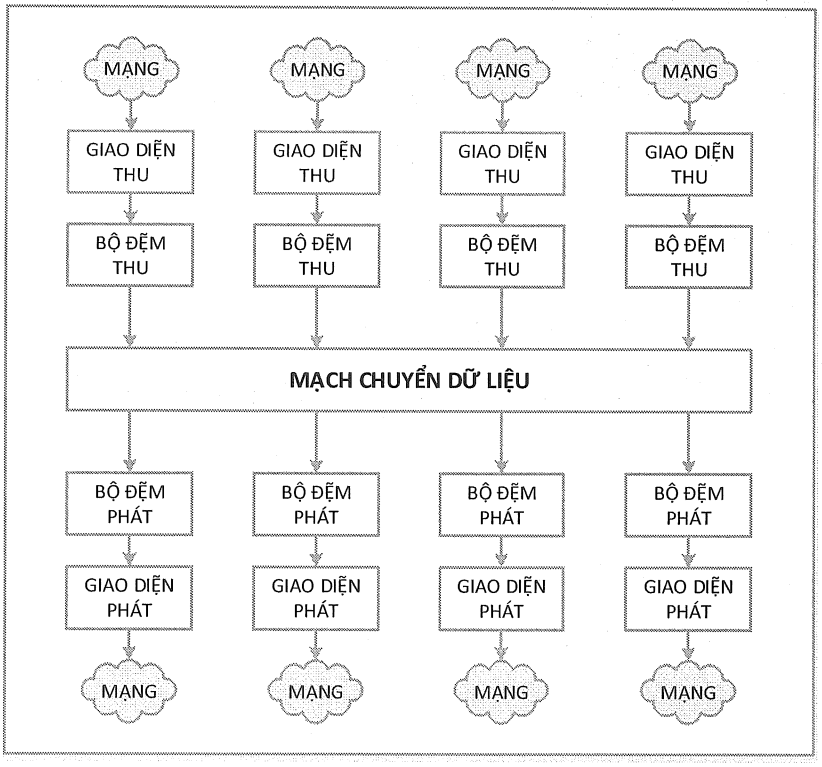
quá trình đọc dữ liệu cũng tiến hành đồng thời trên các luồng theo thứ tự lần lượt (round-robin) tương tự như quá trình ghi, tại một thời điểm có tối đa M con trỏ đọc được gửi vào khối điều khiển đọc, quá trình đọc bắt đầu bằng theo tác nhận con trỏ đọc đầu tiên RPT_first(k) để xác nhận yêu cầu đọc cho luồng thứ k, tiếp đó là thao tác đọc dữ liệu cho một khối dữ liệu của gói trong luồng thứ k này, có phân biệt khối này là khối đầu, khối giữa hay khối cuối của gói đang được đọc cũng như phân biệt thao tác đọc có xóa dữ liệu hay không xóa dữ liệu, khác biệt ở chỗ, quá trình đọc dữ liệu phục vụ các luồng luân phiên theo đơn vị khối dữ liệu, nghĩa là các gói dữ liệu của các luồng phục vụ song song, khác với bộ nhớ chia sẻ đơn luồng phục vụ luồng thông tin đơn theo đơn vị gói dữ liệu, nghĩa là một gói dữ liệu sẽ được đọc hoàn toàn trước khi gói dữ liệu mới được phục vụ;

bộ nhớ chia sẻ đa luồng không cần có các khối thực hiện gom và phân phối các luồng thông tin ở đầu vào và đầu ra như bộ nhớ chia sẻ đơn luồng;

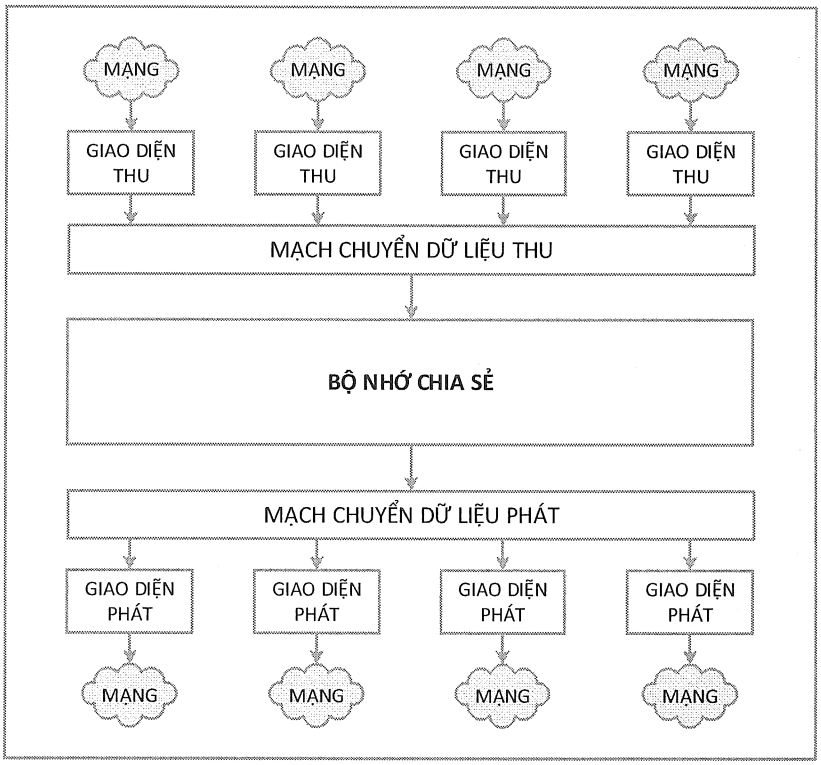
cấu trúc bộ nhớ chia sẻ đa luồng được mô tả bằng ngôn ngữ mô tả phần cứng VHDL (Very High Speed Hardware Description Language) và được thực thi trên phần cứng khả trình FPGA (Field-Programmable Gate Array) trong đó băng thông danh định một chiều của thiết kế đề xuất có thể đạt 240 Gbps (cho cấu hình B = 1024 bit, 1Mb memory) và trong trường hợp thấp nhất có thể đạt 85 Gbps (cho cấu hình B = 1024 bit, 1Mb memory).

3. Lõi chuyển mạch thông tin số dạng gói có sử dụng bộ nhớ chia sẻ đơn luồng theo điểm 1.

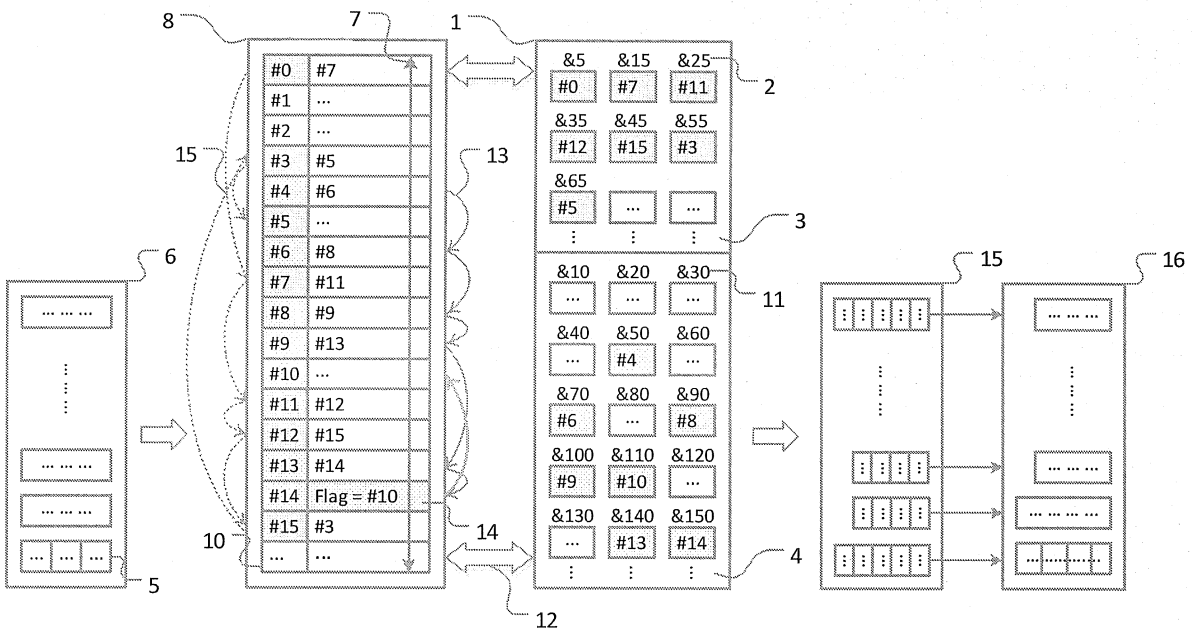
4. Lõi chuyển mạch thông tin số dạng gói sử dụng bộ nhớ chia sẻ đa luồng theo điểm 2.


MẠNG
MẠNG
MẠNG
MẠNG
GIAO DIỆN THU
GIAO DIỆN THU
GIAO DIỆN THU
GIAO DIỆN THU
BỘ ĐỆM THU
BỘ ĐỆM THU
BỘ ĐỆM THU
BỘ ĐỆM THU
MẠCH CHUYỂN DỮ LIỆU
BỘ ĐỆM PHÁT
BỘ ĐỆM PHÁT
BỘ ĐỆM PHÁT
BỘ ĐỆM PHÁT
GIAO DIỆN PHÁT
GIAO DIỆN PHÁT
GIAO DIỆN PHÁT
GIAO DIỆN PHÁT
MẠNG
MẠNG
MẠNG
MẠNG

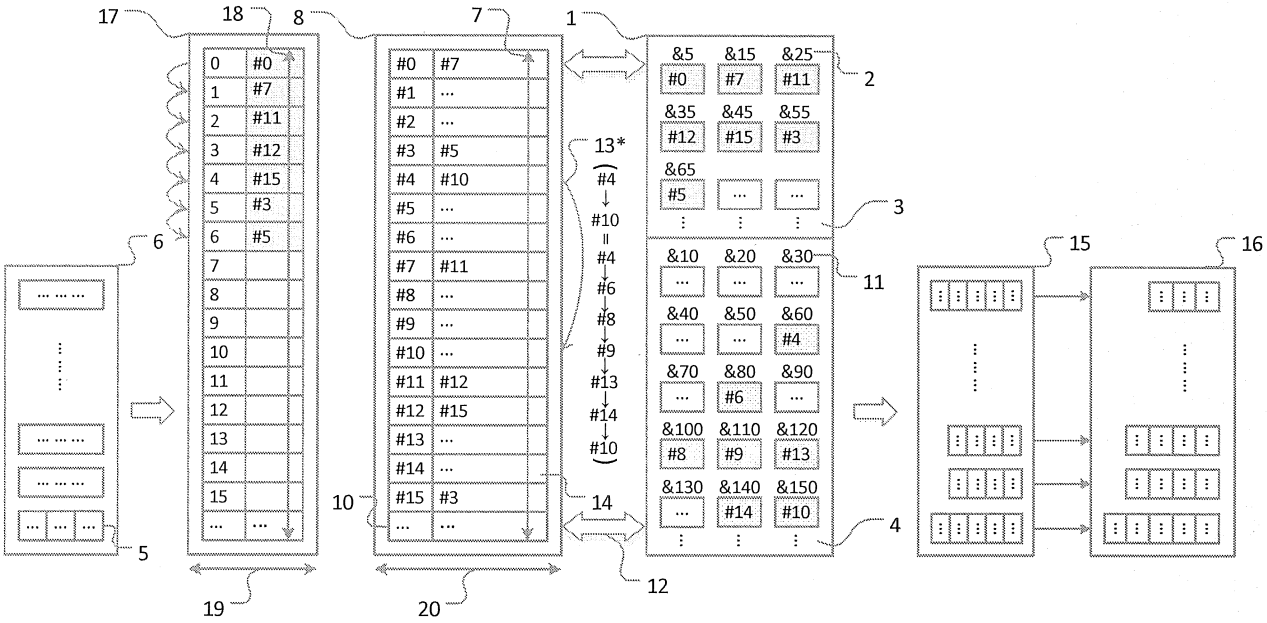
Hình 1


MẠNG
MẠNG
MẠNG
MẠNG
GIAO DIỆN THU
GIAO DIỆN THU
GIAO DIỆN THU
GIAO DIỆN THU
MẠCH CHUYỂN DỮ LIỆU THU
BỘ NHỚ CHIA SẺ
MẠCH CHUYỂN DỮ LIỆU PHÁT
GIAO DIỆN PHÁT
GIAO DIỆN PHÁT
GIAO DIỆN PHÁT
GIAO DIỆN PHÁT
MẠNG
MẠNG
MẠNG
MẠNG

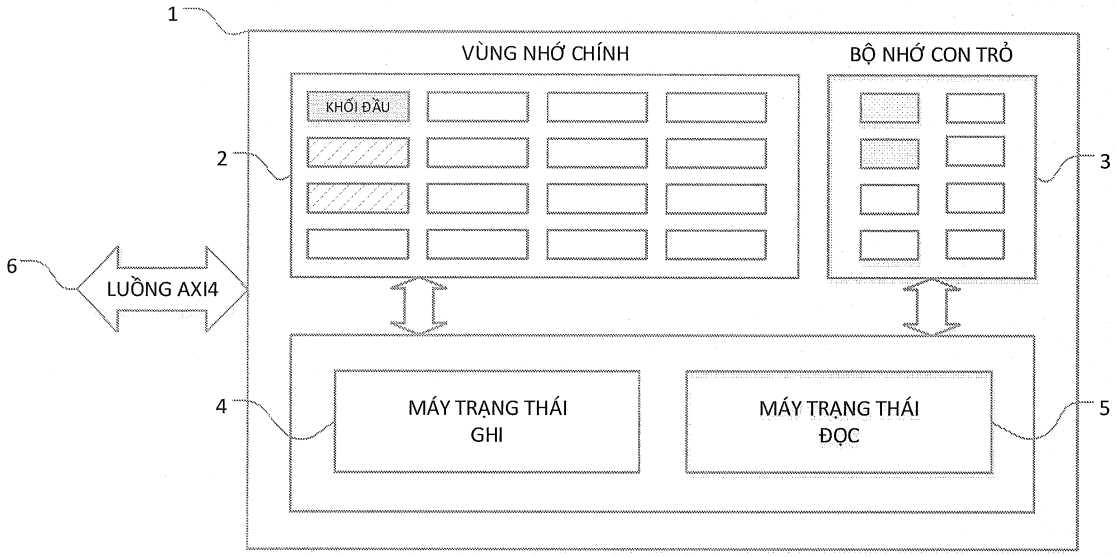
Hình 2


8
7
1
15
13
6
10
14
5
12
2
3
11
4
15
16
Flag = #10

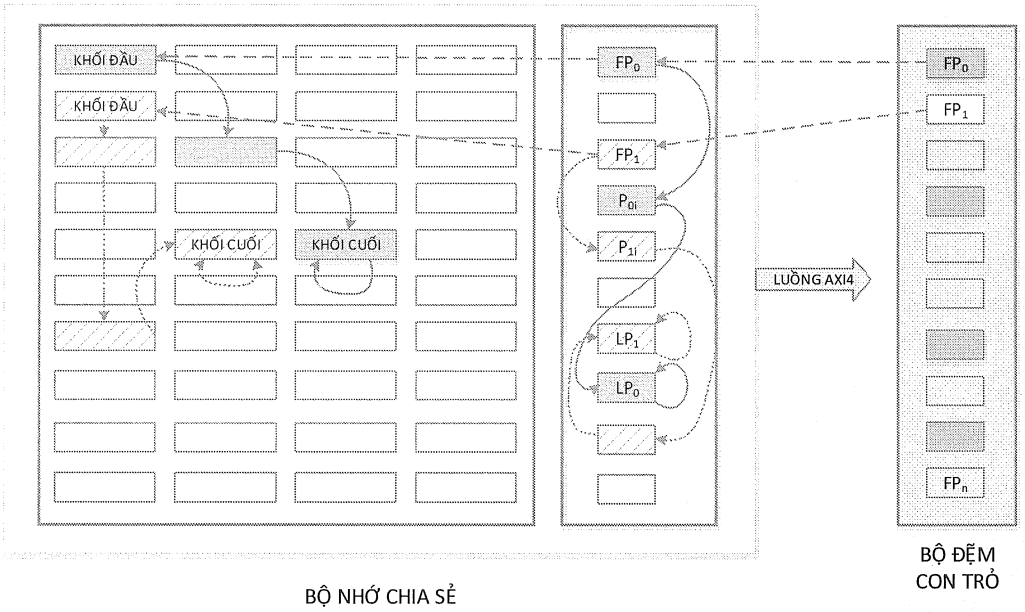
Hình 3


17
18
8
7
1
13*
6
10
14
5
19
20
12
2
3
11
4
15
16

Hình 4


1
VÙNG NHỚ CHÍNH
BỘ NHỚ CON TRỎ
KHỐI ĐẦU
2
3
6
LUỒNG AXI4
4
MÁY TRẠNG THÁI GHI
MÁY TRẠNG THÁI ĐỌC
5

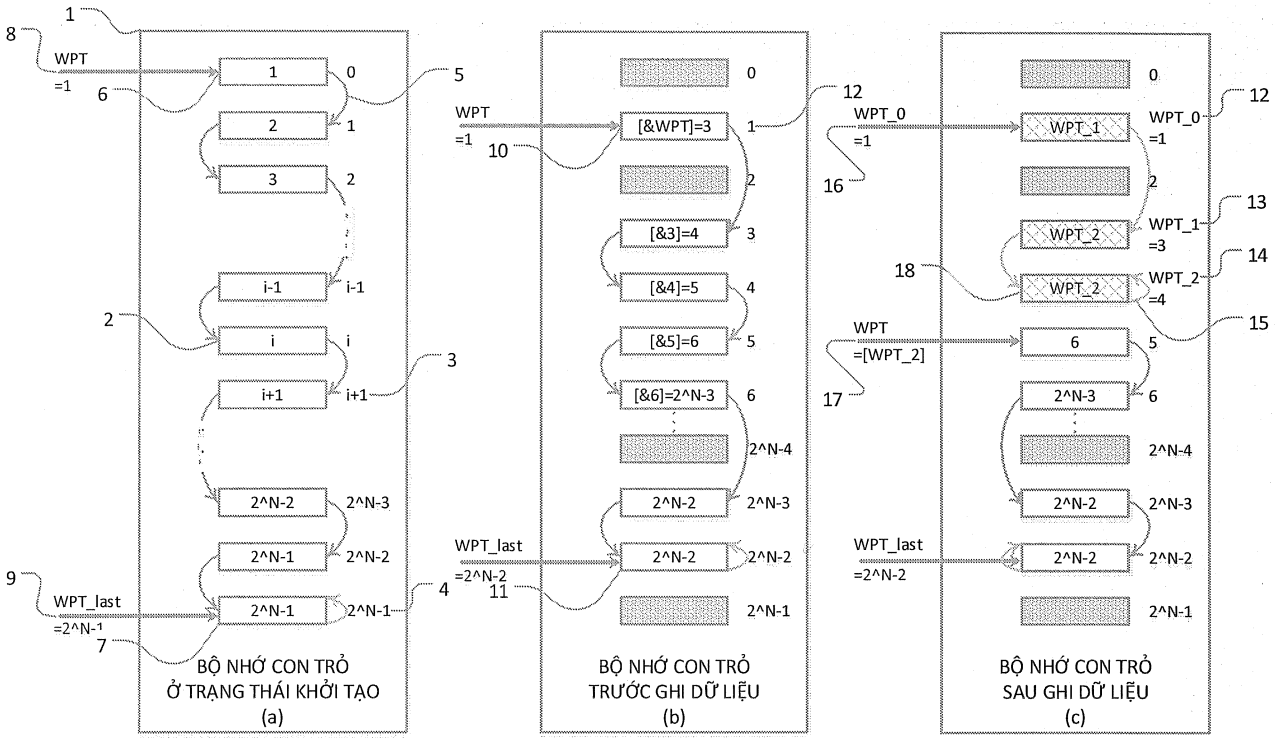
Hình 5


KHỐI ĐẦU
KHỐI ĐẦU
KHỐI CUỐI
KHỐI CUỐI
FP0
FP1
P0i
P1i
LP1
LP0
LUỒNG AXI4
FP0
FP1
FPn
BỘ NHỚ CHIA SẺ
BỘ ĐỆM CON TRỎ

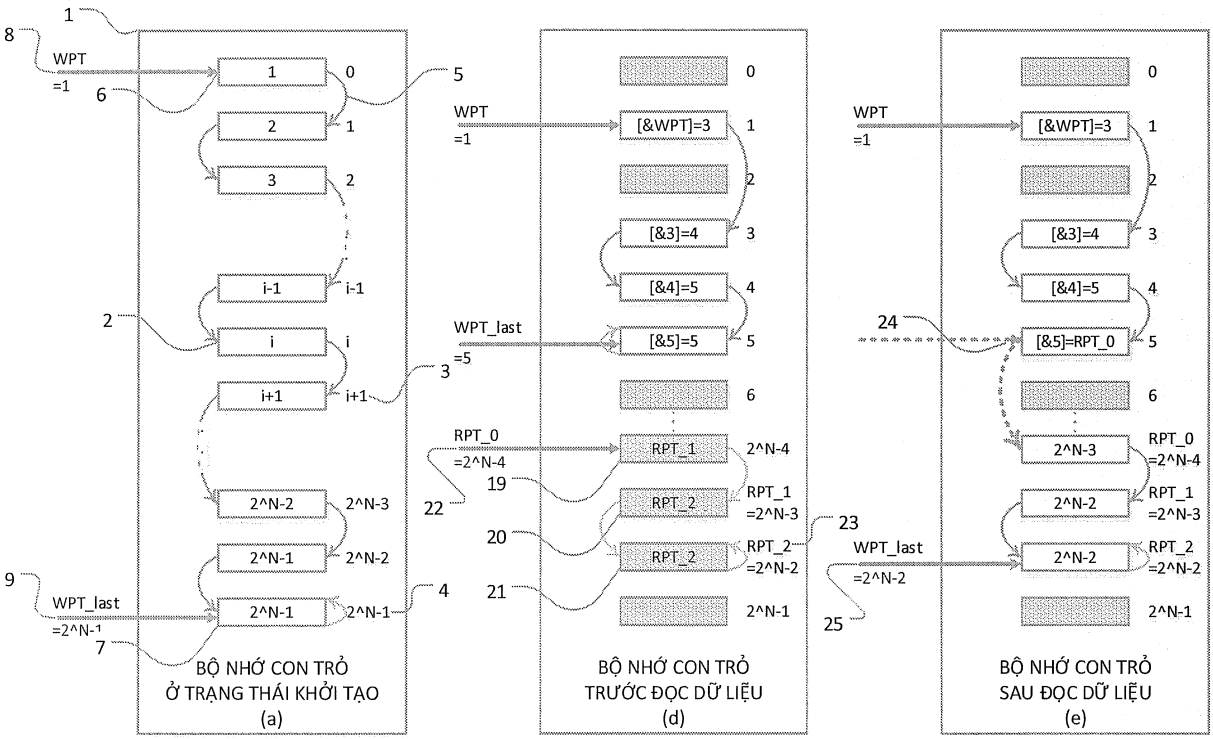
Hình 6

```
     Thuật toán 1: Thuật toán khởi tạo và đọc ghi bộ nhớ chia sẻ đơn luồng
1    # B: Kích thước khối dữ liệu/Độ rộng kênh dữ liệu hệ thống
     # N: Số bít địa chỉ cho vùng nhớ chính và bộ nhớ con trỏ
2    # Khởi tạo bộ nhớ
     Function: Pointer_memory_initialization
        WPT = 0;
        WPT_last = 2^N-1;
        For(i = 0; i < 2^N-1; i = i+1)
            Pointer_memory(i) = i+1;
        End For;
     End Function;
3    # Ghi dữ liệu vào vùng nhớ chính
     Function: Write_data(data)
         # WPT: Vị trí ô nhớ khả dụng đầu tiên
         While((data is valid) and (not last_data))
             Memory_pool(WPT) = data; # Ghi dữ liệu vào ô nhớ WPT
             WPT = Pointer_memory(WPT); # Đọc vị trí ô nhớ khả dụng tiếp theo
             If(first_data)
                 WPT_first = WPT;
             End If;
         End While;
         If((data is valid) and (last_data))
             First_pointer_out = WPT_first; # Gửi con trỏ đầu tiên ra ngoài
             Pointer_memory(WPT) = WPT; # Khởi tạo chuỗi liên kết mới
         End If;
     End Function;
4    # Đọc dữ liệu từ vùng nhớ chính
     Function: Read_data(first_pointer)
         RPT = first_pointer;
         While(RPT <> Pointer_memory(RPT))
             Read_data = Memory_pool(RPT) # Đọc dữ liệu từ bộ nhớ
             RPT = Pointer_memory(RPT) # Đọc con trỏ tiếp theo
         End While;
         # Cập nhật danh sách các ô nhớ khả dụng
         If(delete recent_read_packet)
             WPT_last = first_pointer;
         End If;
     End Function;
     # Kết thúc Thuật toán 1
```

Hình 7


WPT
=1
WPT_last
=2^N-1
BỘ NHỚ CON TRỎ
Ở TRẠNG THÁI KHỞI TẠO
(a)
[&WPT]=3
[&3]=4
[&4]=5
[&5]=6
[&6]=2^N-3
WPT_last
=2^N-2
BỘ NHỚ CON TRỎ
TRƯỚC GHI DỮ LIỆU
(b)
WPT_0
=1
WPT_1
WPT_2
=3
=4
WPT
=[WPT_2]
BỘ NHỚ CON TRỎ
SAU GHI DỮ LIỆU
(c)

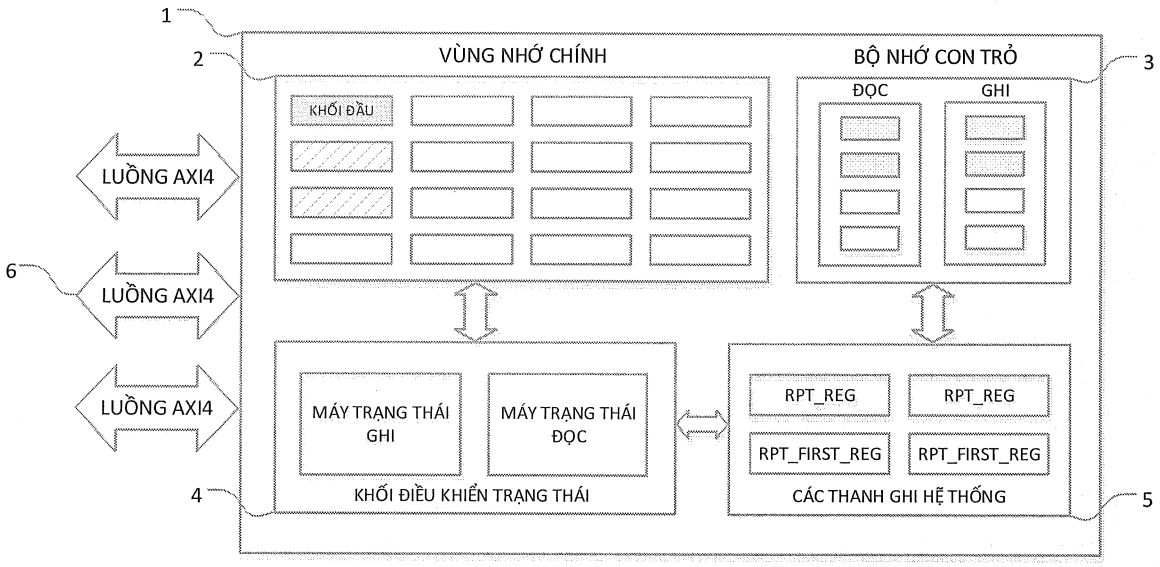
Hình 8


WPT
=1
WPT_last
=2^N-1
BỘ NHỚ CON TRỎ
Ở TRẠNG THÁI KHỞI TẠO
(a)
[&WPT]=3
[&3]=4
[&4]=5
[&5]=5
WPT_last
=5
RPT_0
=2^N-4
RPT_1
RPT_2
=2^N-3
=2^N-2
BỘ NHỚ CON TRỎ
TRƯỚC ĐỌC DỮ LIỆU
(d)
[&5]=RPT_0
WPT_last
=2^N-2
BỘ NHỚ CON TRỎ
SAU ĐỌC DỮ LIỆU
(e)

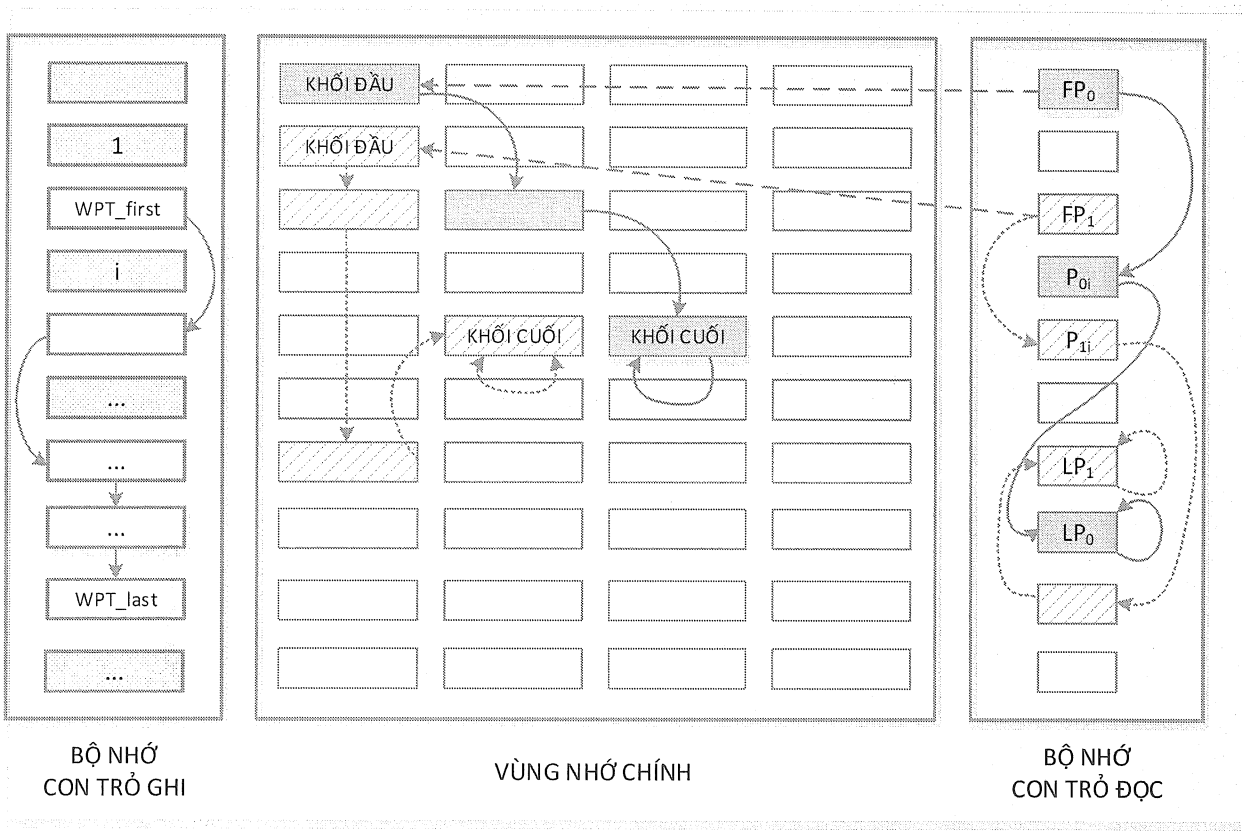
Hình 9

Bảng 1

Family	Part	Fmax (MHz)	LUT	FF	BRAM	Memory config
Artix-7	XC7A200T	181.0	215	252	59	512 bit, 256 KB
Kintex-7	XC7K325T	226.0	215	252	59	512 bit, 256 KB
Virtex-7	XC7VX550T	238.0	215	252	59	512 bit, 256 KB
Artix-7	XC7A200T	118.0	330	1088	131.5	512 bit, 512 KB
Kintex-7	XC7K325T	135.0	330	1088	131.5	512 bit, 512 KB
Virtex-7	XC7VX550T	137.9	330	1088	131.5	512 bit, 512 KB
Artix-7	XC7A200T	118.0	435	1984	259.5	1024 bit, 1024 KB
Kintex-7	XC7K325T	137.0	435	1984	259.5	1024 bit, 1024 KB
Virtex-7	XC7VX550T	138.0	435	1984	259.5	1024 bit, 1024 KB



Hình 10



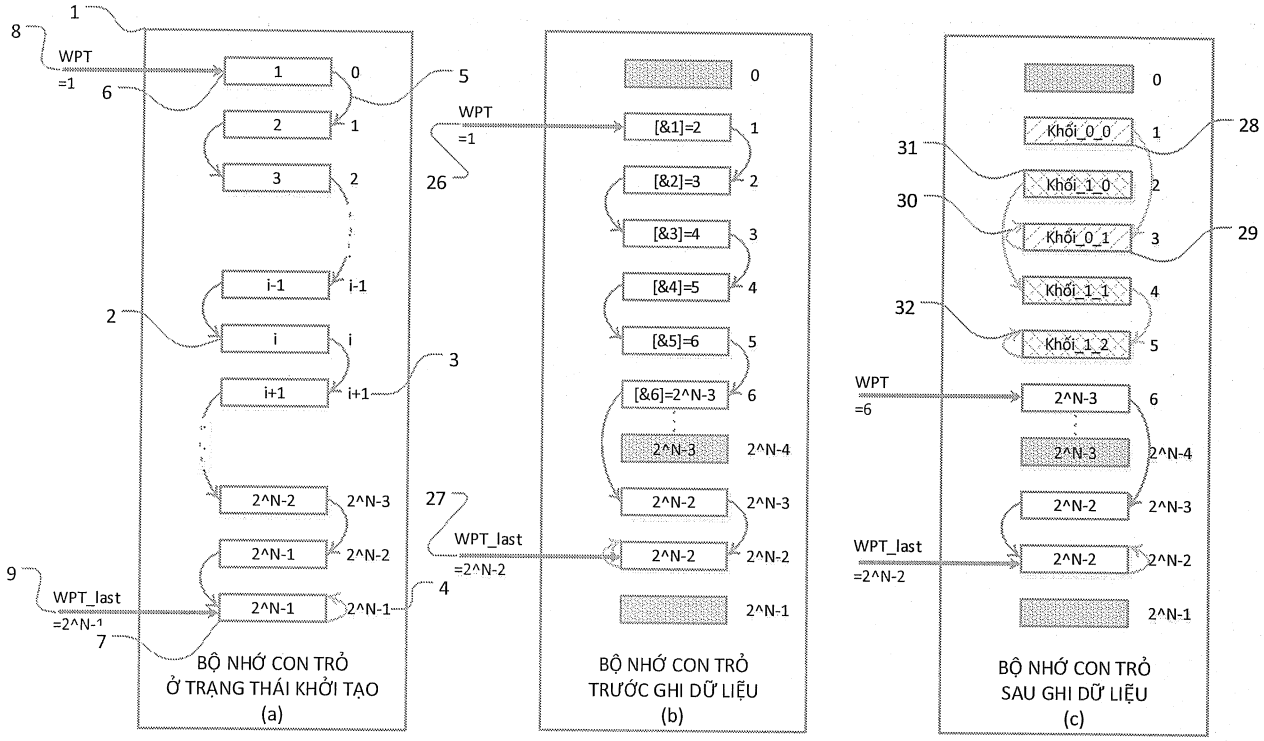
Hình 11

Thuật toán 2: Thuật toán khởi tạo và đọc ghi bộ nhớ đa luồng

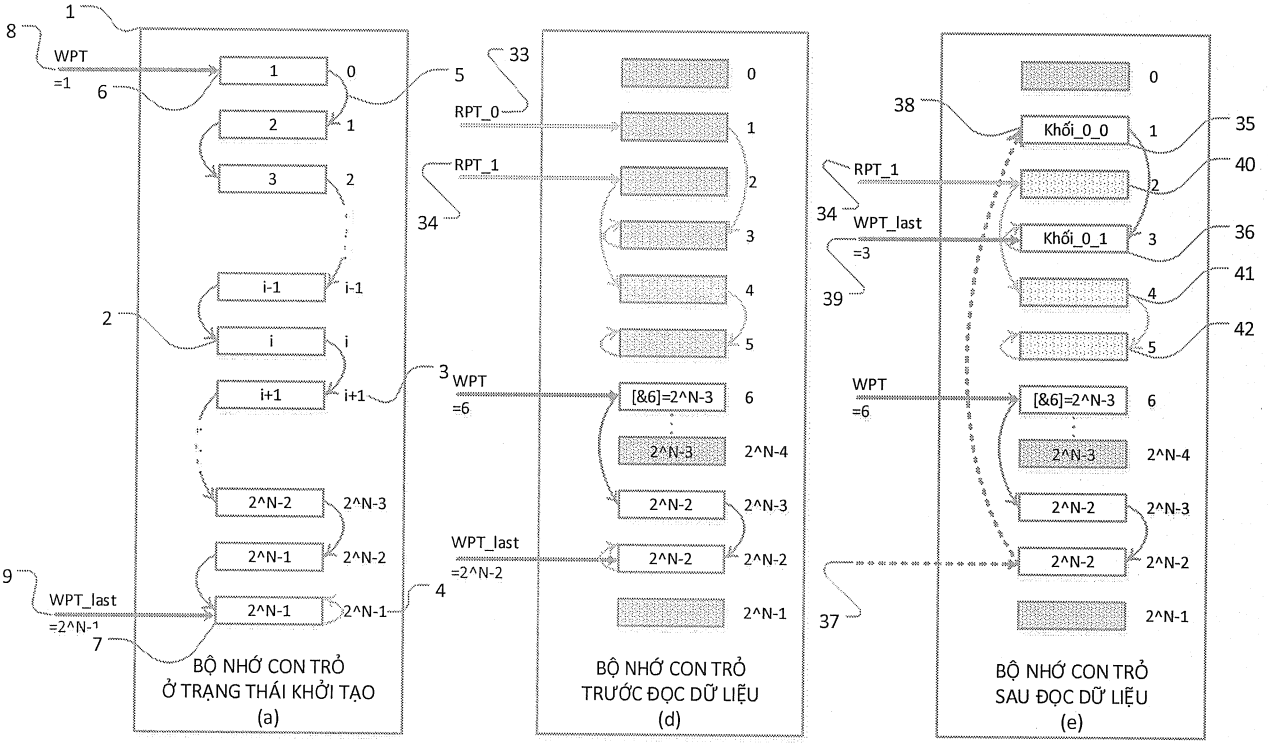
```
1   # B: Kích thước khối dữ liệu/Độ rộng kênh dữ liệu hệ thống
    # N: Số bít địa chỉ cho bộ vùng nhớ chính và các bộ nhớ con trỏ
    # M: Số luồng vào ra của bộ nhớ
2   # Khởi tạo bộ nhớ
    Function: Pointer_memory_initialization
        WPT_first = 0;
        WPT_last = 2^N-1;
        WPT = WPT_first;
        For(i = 0; i < 2^N-1; i = i+1)
            Write_memory_pointer(i) = i+1;
        End For;
    End Function;
3   # Ghi dữ liệu vào vùng nhớ chính
    Function: Write_data(data)
        While(new_data valid)
            k = current_write_stream_index; # Chỉ số luồng hiện tại
            Memory_pool(WPT) = data; # Ghi dữ liệu vào ô nhớ WPT
            WPT =  Write_pointer_memory(WPT); # Vị trí ô nhớ khả dụng tiếp theo
            # Cập nhật bộ nhớ con trỏ đọc
            If(first_data)
                WPT_first_reg(k) = WPT;
                WPT_reg(k) = WPT;
```

```
            Else If(not last_data)
                Read_pointer_memory(RPT_reg(k)) = WPT;
                WPT_reg(k) = WPT;
            Else If((data is valid) and (last_data))
                Read_pointer_memory(WPT_reg(k)) = WPT;
                Read_pointer memory(WPT) = WPT; # Kết thúc ghi gói dữ liệu
                First_pointer_out = WPT_first_reg(k); # Gửi con trỏ ra ngoài
            End If;
        End While;
    End Function;
4   # Đọc dữ liệu từ vùng nhớ chính
    Function: Read_data() # Đọc gói dữ liệu từ con trỏ vị trí đầu tiên
        k = current_write_stream_index; # Chỉ số luồng đang được phục vụ
        For(i = 0; i < M; i < i+1)
            If(first_pointer(i) valid)
                RPT_first_reg(i) = first_pointer(i);
            End If;
        End For;
        k = next_read_pointer_stream_index; # Chỉ số luồng cần đọc
        # Lấy địa chỉ tiếp theo
        If(first_pointer(k) valid)
            RPT = first_pointer(k);
            RPT_reg(k) = first_pointer(k);
        Else
            RPT = Read_pointer_memory(RPT_reg(k));
        End If;
        Read_data = Memory_pool(RPT); # Đọc dữ liệu hiện tại và gửi ra
        RPT = Read_pointer_memory(RPT); # Đọc con trỏ kế tiếp
        End While;
        # Nếu đọc hết và có yêu cầu xóa
        If(RPT <> pre_read_pointer(k))
            RPT_reg(k) = RPT;
        Else If(delete requested)
            WPT_last = RPT_first_reg(k);
        End If;
    End Function;
    # Kết thúc Thuật toán 2
```

Hình 12



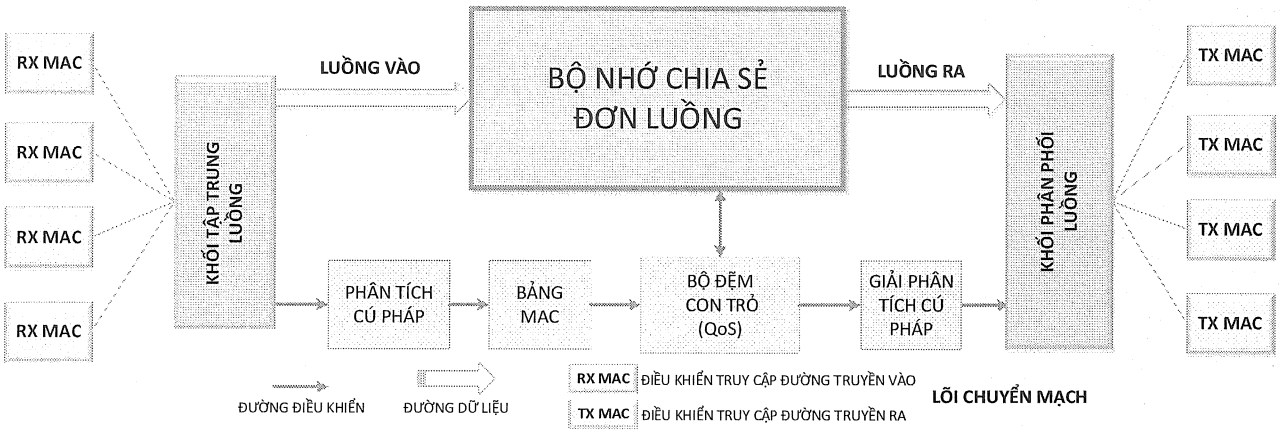
Hình 13



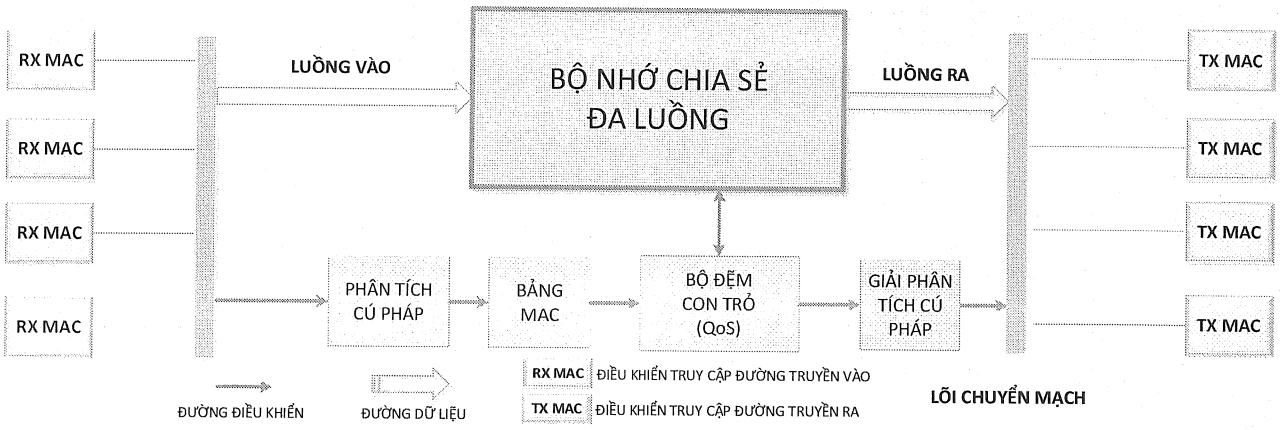
Hình 14

Bảng 2

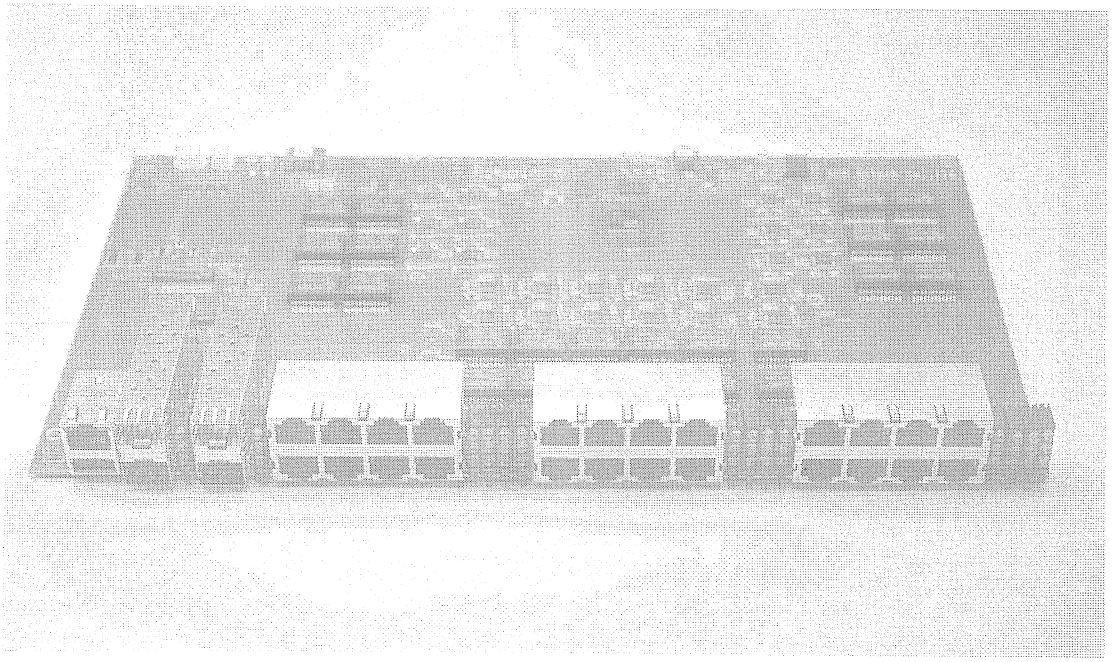
Family	Part	Fmax (MHz)	LUT	LUT RAM	FF	BRAM	Memory config
Artix-7	XC7A200T	170	1288	0	2291	61	512 bit, 256 KB
Kintex-7	XC7K325T	238	1288	0	2291	61	512 bit, 256 KB
Virtex-7	XC7VX550T	282	1288	0	2291	61	512 bit, 256 KB
Artix-7	XC7A200T	172	1359	65	3222	136	512 bit, 512 KB
Kintex-7	XC7K325T	236.1	1359	65	3222	136	512 bit, 512 KB
Virtex-7	XC7VX550T	248.5	1359	65	3222	136	512 bit, 512 KB
Artix-7	XC7A200T	172	1308	65	4546	264	1024 bit, 1024 KB
Kintex-7	XC7K325T	240.3	1308	65	4546	264	1024 bit, 1024 KB
Virtex-7	XC7VX550T	249.8	1308	65	4546	264	1024 bit, 1024 KB



Hình 15



Hình 16



Hình 17