# **Đại học Bách khoa TP.HCM**

**Khoa Khoa học và Kỹ thuật Máy tính**

**So sánh kiến trúc RISC và CISC**

**Nhóm 3: Lê Trương Trọng Duy, Bùi Đức Hiếu, Đoàn Dũ, Nguyễn Thanh Phương Ngọc, Lê Nhựt Trường , Võ Minh Khải**

**Tóm lược** Bài viết này giới thiệu tổng quát, so sánh RISC và CISC trên các khía cạnh kiến trúc, ứng dụng nhằm tiến hành đánh giá được ưu, nhược điểm của RISC và CISC. Từ đó, hỗ trợ cả hai kiến trúc xác định hướng chuyên môn nên tập trung phát triển trong lĩnh vực máy tính.

**Từ khóa** CISC, RISC, CISC vs RISC, CISC RISC attributes, CISC RISC differences.

# **Giới thiệu**

Hiệu suất là mối quan tâm hàng đầu trong mọi lĩnh vực. Lĩnh vực máy tính không nằm ngoài xu thế đó. Các công ty sản xuất luôn chú trọng đến việc cải thiện hiệu suất của máy tính để máy tính ngày càng nhanh và hoạt động hiệu quả hơn. CPU (bộ xử lý trung tâm) là phần quan trọng nhất của một máy tính. Vì vậy, phần lớn nổ lực được tập trung vào cải tiến hiệu suất thực thi của CPU. Thời gian thực thi của một chương trình trên CPU phụ thuộc vào nhiều yếu tố như tập lệnh của chương trình, số chu kỳ thực thi trung bình trên một lệnh và thời gian của một chu kỳ. Trên thực tế, nhiều ý tưởng cho việc thiết kế bộ vi xử lý đã được hiện thực…Dựa vào những ý tưởng trên, nhiều kiến trúc đã được tạo ra. Bài viết này trình bày và so sánh hai kiến trúc phổ biến trong số đó là CISC và RISC. Bài viết sẽ giới thiệu về hai kiến trúc, so sánh và đi tới việc đánh giá sự phù hợp của mỗi kiến trúc với các yêu cầu cụ thể khi hiện thực phần cứng.

# **Định nghĩa**

## **CISC**

Kiến trúc CISC [1] (Complex Instruction Set Computer) là kiến trúc của bộ vi xử lý với một tập các lệnh phức tạp. Mỗi một lệnh có thể thực hiện được nhiều hơn một tác vụ như đọc dữ liệu từ bộ nhớ, tính một phép toán và ghi lại kết quả trở lại bộ nhớ. Mục tiêu chính của kiến trúc CISC là thực hiện một công việc với càng ít lệnh càng tốt. Việc này có thể đạt được bằng cách thiết kế bộ vi xử lý để có thể hiểu được và thực thi một loạt các phép toán. Như vậy, bộ vi xử lý với kiến trúc CISC sẽ có một tập bao gồm nhiều lệnh được hỗ trợ. Với một công việc cụ thể, bộ vi xử lý sẽ chuẩn bị sẵn một lệnh tương ứng để có thể thực thi được công việc này.

## **RISC**

Kiến trúc RISC [2] (Reduced Instruction Set Computer) là kiến trúc của bộ xứ lý với một tập các lệnh được rút gọn. Điều đó có nghĩa là kiến trúc này có ít lệnh hơn và một công việc sẽ phải được thực hiện với một số lệnh đơn giản hơn. Bù lại, mỗi lệnh sẽ được thực thi trong một số ít chu kỳ hơn, dẫn đến thời gian thực hiện toàn bộ chương trình không bị tăng lên với chương trình chạy trên kiến trúc CISC. Việc giảm không gian lưu trữ các tập lệnh sẽ tạo được nhiều không gian hơn dành cho các thanh ghi và góp phần làm tăng hiệu suất của bộ vi xử lý.

# **So sánh**

Trong bài viết này, hai kiến trúc RISC và CISC sẽ được so sánh [3] trên hai khía cạnh: kiến trúc [4], ứng dụng.

## **Kiến trúc**

Về kích thước tập lệnh, tập lệnh CISC rất phức tạp, tập lệnh RISC chỉ chứa ít các lệnh cơ bản. Tập lệnh của CISC được thiết kế nhằm phục vụ các chức năng chuyên biệt nên bên trong chứa nhiều lệnh phức, trong khi với RISC, một phép tính phức tạp có thể được dẫn xuất từ các phép tính đơn nguyên tử nhất. Ví dụ: CISC có lệnh nhân. RISC chỉ có lệnh cộng và phép nhân trên kiến trúc RISC sẽ được tính bằng cách thực hiện các lệnh cộng liên tiếp.

Về độ dài lệnh, các lệnh trong kiến trúc CISC có độ dài không giống nhau. Các lệnh trong kiến trúc RISC có độ dài bằng nhau: 32bit. CISC, điển hình là VAX x86, có cấu trúc như sau:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Opcode | Addr Specifier | Addr Field | … | Addr Specifier | Addr Field |

Và RISC, điển hình là ARM, MIPS, PowerPC, có cấu trúc như sau:

|  |  |  |  |
| --- | --- | --- | --- |
| Opcode | Addr Field 1 | Addr Field 2 | Addr Field 3 |

Do tính cố định chiều dài lệnh này trong RISC nên RISC không thể truy cập bộ nhớ không sắp xếp. Các vi xử lý RISC đã được thiết kế cho các trạm, nơi mà dữ liệu luôn được sắp xếp theo từ (word) bởi vì trình biên dịch đặt chúng nằm như thế. Lượng dữ liệu nhỏ hơn (như byte) hoặc là không tồn tại hoặc là được chèn thêm zero để vừa với một từ 32-bit. Trong các hệ thống nhúng hiếm khi có sự ngăn nắp như vậy. Tương tự như vậy, lượng dữ liệu lẻ như giá trị 24-bit phải được mở rộng zero- hoặc sign- khi lưu trữ chúng, và như thế là lãng phí RAM.

Về thời gian thực thi lệnh, các vi xử lý CISC có tốc độ xung nhịp thấp, dưới 100MHz. Trong khi tốc độ xung nhịp của chip RISC có thể tính bằng GHz. Một lệnh phức trong CISC để hoàn thành phải tốn nhiều chu kỳ hơn là một câu lệnh đơn. Vì thế, một lệnh của RISC thường hoàn thành trong thời gian ngắn hơn một lệnh trong CISC. Do cấu trúc lệnh RISC có chiều dài bằng nhau nên dễ dàng áp dụng pipeline cho kiến trúc này, dẫn đến một chương trình chạy trên kiến trúc RISC có thể sẽ kết thúc nhanh hơn CISC.

Về phương thức hoạt động, trong kiến trúc CISC, transistors dùng để lưu trữ các lệnh phức. Còn kiến trúc RISC huy động nhiều transistors cho thanh ghi bộ nhớ. Do đó, RISC đòi hỏi ít transistor của không gian phần cứng hơn, nên không gian cho các thanh ghi với mục đích chung là nhiều hơn trong kiến trúc CISC. Phương thức hoạt động của CISC phức tạp hơn RISC. CISC thao tác truy xuất bộ nhớ và Ram, còn RISC chỉ truy xuất RAM. CISC sử dụng nhiều cơ chế đánh địa chỉ, trong khi RISC chỉ dùng cơ bản địa chỉ thanh ghi hoặc địa chỉ trực tiếp hoặc địa chỉ tức thời. Kiến trúc cồng kềnh, phức tạp cộng cơ chế phức tạp không kém, dẫn đến CISC thường có tốc độ hoạt động chậm hơn RISC.

Về diện tích phần cứng, CISC nhằm thực hiện các lệnh phức tạp, nên phần cứng CISC cồng kềnh, chiếm nhiều diện tích hơn hẳn RISC. RISC có thiết kế nhỏ gọn hơn nên có thể tích hợp thêm vào bên trong bộ xử lý các thanh ghi, các cổng vào ra và bộ nhớ cache, …

|  |  |  |  |
| --- | --- | --- | --- |
| Tiêu chí | CISC | RISC | Ghi chú |
| Tập lệnh | Nhiều loại lệnh.  Phức tạp.  Bao gồm lệnh đơn và lệnh phức. | Ít, chỉ gồm những loại cơ bản như cộng trừ, …  Đơn giản.  Chỉ gồm lệnh đơn. |  |
| Độ dài lệnh | Mang tính tùy chọn. Vi xử lý 32-bit vẫn có thể có lệnh 8-bit, 16-bit, thậm chí là dài hơn. | Cố định. 32-bit thì độ dài luôn là 32-bit. | RISC có thể gây lãng phí RAM.  RISC thuận tiện cơ chế pipelines. |
| Thời gian thực thi lệnh | Chậm. Một câu lệnh có thể thực thi nhiều tác vụ. | Nhanh. Một câu lệnh chỉ thực thi duy nhất một tác vụ | Một lệnh phức của CISC tốn nhiều chu kỳ, RISC thì một lệnh/một chu kỳ  CISC khó áp dụng pipeline còn RISC thì ngược lại. |
| Sử dụng Transistors | Để lưu trữ lệnh phức. | Để lưu trữ lệnh và dùng cho thanh ghi truy suất bộ nhớ. | RISC dùng ít transistors hơn CISC. |
| Thao tác | Truy xuất nhiều loại bộ nhớ. | Chủ yếu truy xuất RAM. | RISC nhanh hơn và có lệnh truy xuất bộ nhớ riêng biệt. |
| Đánh địa chỉ | Nhiều phương thức dùng để đánh địa chỉ | Thường dùng địa chỉ thanh ghi, địa chỉ trực tiếp và địa chỉ tức thời. |  |
| Diện tích phần cứng | Lớn hơn RISC | Nhỏ hơn CISC. |  |

## **Ứng dụng**

Về kích thước chương trình, để viết một chương trình, kiến trúc CISC hỗ trợ đa dạng lệnh hơn RISC. Do đó, nếu không sử dụng compiler có sẵn, viết chương trình trên CISC nhẹ nhàng hơn so với RISC, đồng thời kích thước chương trình cũng gọn hơn.

Về khả năng mở rộng và rủi ro khi phát triển, mục đích của CISC là tối giản kích thước chương trình nên việc thực hiện các tính toán phức tạp được thực hiện hoàn toàn trên phần cứng. Do đó, để mở rộng thêm tập lệnh, với CISC phải thiết kế lại hoặc thêm phần cứng. Điều này là rất khó, dễ lỗi và khi lỗi thì rất khó phát hiện nơi xảy ra vấn đề cũng như khó sửa chữa. Lí do là trong tương quan so sánh với việc mở rộng chỉ cần thực hiện trên phần mềm, từ thiết kế cho tới phát hiện lỗi và sửa lỗi đều thực hiện trên phần mềm. Tác động lên phần cứng thì cũng tốn nhiều thời gian và chi phí hơn. Vì vậy, về khả năng mở rộng và tránh rủi ro, kiến trúc RISC lợi thế hơn hẳn CISC.

Về năng lượng, kiến trúc RISC đơn giản hơn và thường chỉ truy cập bộ nhớ RAM nên tiêu tốn ít năng lượng hơn CISC.

Về hiệu suất, tốc độ của RISC nhanh hơn nhưng trong chương trình, giả sử cần một phép nhân nhưng vì RISC không hỗ trợ nên người dùng sẽ phải viết mã cho phép nhân đó trong khi CISC đã hỗ trợ sẵn lệnh này. Bù trừ giữa hai khoảng thời gian đó nên chưa hẳn RISC hay CISC đã nhanh hơn. Tuy nhiên, CISC có nhược điểm đó là có những lệnh chuyên biệt mà người dùng không bao giờ dùng đến. Lựa chọn CISC lúc này cũng là một sự lãng phí. Tóm lại, hiệu suất của hai kiến trúc này được đánh giá tùy theo nhu cầu người sử dụng, bởi vì chúng đều có ưu và nhược điểm riêng.

Về phạm vi ứng dụng, như đã đề cập, kiến trúc CISC ít gặp khó khăn hơn khi viết chương trình Assembly. Chương trình càng lớn, CISC càng thể hiện ưu thế này của nó. Kiến trúc CISC thường được sử dụng trong các máy tính PC thông thường hoặc máy tính lớn. Kiến trúc RISC thì thường được ứng dụng trong các thiết bị nhỏ gọn như các thiết bị cầm tay, smartphones, tablets, … Các thiết bị nhỏ gọn này thường không đòi hỏi tính toán phức tạp, và chúng cũng hạn chế về năng lượng cũng như tốc độ, nên RISC là lựa chọn tốt cho các thiết bị này.

|  |  |  |  |
| --- | --- | --- | --- |
| Tiêu chí | CISC | RISC | Ghi chú |
| Kích thước chương trình | Nhỏ. Lệnh trong CISC là lệnh phức. | Lớn. Lệnh trong RISC là lệnh đơn. | Lý do: Gọi lệnh phức thì ngắn gọn hơn một nhóm nhiều lệnh đơn thực hiện cùng chức năng. |
| Khả năng mở rộng | Thấp hơn, chậm hơn.  Chi phí cao hơn.  Thay đổi phải dựa trên sự thay đổi phần cứng. | Cao hơn, nhanh hơn.  Chi phí thấp hơn.  Không cần thay đổi phần cứng, chỉ cần phát triển phần mềm để mở rộng khả năng đáp ứng. | Lý do: kiến trúc CISC có độ phức tạp cao hơn cho nên làm việc với phần cứng khó khăn và không dễ tùy biến như với phần mềm. |
| Rủi ro | Cao hơn | Thấp hơn | Lý do: kiến trúc CISC có độ phức tạp cao hơn nên làm việc với phần cứng khó khăn và không dễ tùy biến như với phần mềm. |
| Năng lượng | Cao hơn | Thấp hơn | Thiết kế của vi xử lý RISC đơn giản hơn CISC.  CISC phải truy cập bộ nhớ nhiều hơn RISC. |
| Hiệu suất | Cao hơn nếu chạy chương trình phức tạp cần tính tái sử dụng và đơn giản hóa lập trình. | Cao hơn nếu chạy chương trình đơn giản, cần tốc độ nhanh. | Được đánh giá tùy theo nhu cầu. |
| Ứng dụng | Thích hợp cho lập trình Assembly.  Chủ yếu trên máy tính, máy tính hớn. | Thích hợp cho lập trình cấp cao.  Các thiết bị nhỏ gọn như tablets, smartphones, ... | Các thiết bị nhỏ gọn thường có sự hỗ trợ về tôc độ chip và năng lượng thấp hơn. |

# **Đánh giá**

## **RISC**

### **Ưu điểm**

* Diện tích dành cho vi xử lý RISC là 10%, ít hơn nhiều so với CISC là 60%. Do đó, các thành phần như thanh ghi, các cổng ra vào và bộ nhớ cache, … có thể tích hợp dễ dàng hơn.
* Việc sử dụng thanh ghi hạn chế truy xuất nhiều vào bộ nhớ, việc thực hiện kỹ thuật ống dẫn liên tục và hiệu quả vì các lệnh đề có thời gian thực hiện giống nhau, cùng dạng cùng với việc giải mã lệnh đơn giản nên tốc độ tính toán sẽ nhanh hơn.
* Bộ điều khiển của RISC trở nên đơn giản hơn. Việc đơn giản hơn thì rủi ro khi thiết kế cũng giảm đi. Đồng thời, thời gian thiết kế cũng sẽ ngắn hơn nhiều.

### **Nhược điểm**

Nhược điểm lớn nhất của RISC là sẽ làm chương trình dài ra hơn so với chương trình viết cho vi xử lý CISC, vì:

* RISC chỉ cho phép lệnh đọc và ghi truy xuất vào bộ nhớ. Do đó, để thực hiện một thao tác nhất định buộc phải dùng nhiều lệnh trung gian.
* Tập lệnh có ít lệnh nên các lệnh không có sẵn phải được thay thế bằng một chuỗi lệnh của bộ xử lý RISC.
* Các chương trình dịch gặp nhiều khó khăn vì có ít lệnh làm cho có ít lựa chọn để diễn dịch các cấu trúc của chương trình gốc. Sự cứng nhắc của kỹ thuật ống dẫn cũng gây khó khăn.
* Việc tính các địa chỉ hiệu dụng là cần thiết vì không có nhiều cách định vị.
* Có ít lệnh trợ giúp cho ngôn ngữ cấp cao.

## **CISC**

### **Ưu điểm**

* Chương trình trên máy CISC ngắn hơn so với máy RISC.
* Số lệnh để thực hiện chương trình ít hơn.
* Khả năng thâm nhập bộ nhớ dễ dàng hơn.
* Vi xử lý CISC trợ giúp mạnh hơn các ngôn ngữ cao cấp nhờ có tập lệnh phức tạp.

### **Nhược điểm**

* Thời gian xây dựng xong bộ vi xử lý trở nên dài hơn do các câu lệnh phức tạp. Khi đó, độ rủi ro cũng sẽ cao hơn và năng lượng tiêu tốn cũng sẽ nhiều hơn.
* Diện tích của vi xử lý dùng cho bộ điều khiển lớn. Điều đó làm giảm khả năng tích hợp thêm vào vi xử lý.
* Tốc độ tính toán chậm so với RISC.

# **Kết luận**

Ngày nay, cả CISC và RISC đều vẫn còn đang phát triển và phổ biến rộng rãi. Tuy nhiên, RISC nhanh chóng vượt qua CISC như hệ thống ưa thích dành cho các máy tính mới. Hướng nghiên cứu chuyển dần sang việc thiết kế tập lệnh cho RISC và tối ưu hóa trình biên dịch. CISC phức tạp, mất nhiều thời gian để thực hiện, nhưng có lợi thế lớn: không đòi hỏi nhiều RAM như RISC. Tuy nhiên, ngày nay giá cả bộ nhớ cũng như công nghệ phát triển giúp cho việc mở rộng bộ nhớ on-board dễ dàng hơn, điều này khiến lợi thế về giá cả của CISC dần bị xóa bỏ. CISC với Mật độ mã và khả năng tích hợp tốt hơn, hiệu suất tốt với các lệnh hỗ trợ bit manipulation, memory accesses, looping, decision trees vẫn là công cụ đắc lực cho các nhà lập trình hệ thống nhúng.

# **Tham khảo**

|  |  |
| --- | --- |
| [1] | Wikipedia. [Online]. Available: http://en.wikipedia.org/wiki/Complex\_instruction\_set\_computing. |
| [2] | Wikipedia. [Online]. Available: http://en.wikipedia.org/wiki/Reduced\_instruction\_set\_computing. |
| [3] | Differencebetween.net. [Online]. Available: http://www.differencebetween.net/technology/protocols-formats/difference-between-risc-and-cisc/. |
| [4] | members.ii.net. [Online]. Available: http://members.ii.net/~lawley/is/12/ca/risc/. |