Tạp Chí Tin Học và Điều Khiển Học
Tập IX(1993), Số 4, 20-27

Một Số Vấn Đề Cơ Bản Của Hệ Điều Hành Thời Gian Thực

Tiếp theo

Phan Minh Tân
Viện Tin Học, Viện Khoa Học Việt Nam

3. Hệ điều hành thời gian thực trong điều khiển quá trình

Hệ điều hành đơn nhiệm và đa nhiệm thời gian thực

Hệ điều hành thời gian thực đơn giản cung cấp việc sử dụng nguyên lý đơn chương trình. Ở đây các nhiệm vụ đã khởi hành thì đượchạy đến khi hoàn thành. Nó chỉ chờ cổng I/O liên quan. Theo nguyên lý này, tổng thời gian để kết thúc hai nhiệm vụ nêu trong hình (8) đã kéo dài đến (f), (xem vẽ lại trong hình 13).

Hình 14 so sánh sự hoạt động của hệ điều hành thời gian thực đơn nhiệm và đa nhiệm để xử lý 3 nhiệm vụ. Hệ điều hành đa nhiệm vụ sử dụng CPU hữu hiệu hơn gấp 2 lần loại đơn nhiệm. Thực tế phần lớn các hệ điều hành thời gian thực được thiết kế trên cơ sở đa nhiệm.

Các yêu cầu đối với việc thiết kế các hệ điều hành thời gian thực

Các hệ điều hành thời gian thực phải đáp ứng được các yêu cầu sau đây:

1. Danh mục thực hiện các nhiệm vụ theo mức ưu tiên hoặc theo thời gian.
2. Treo các nhiệm vụ ưu tiên thấp hơn khi nhiệm vụ ưu tiên cao hơn cần kéo dài.
3. Điều khiển truyền qua cổng I/O bằng các thủ tục truyền mềm.
4. Phục vụ các ngắt cứng.
5. Có đồng hồ thời gian thực.
6. Phát hiện và hiệu chỉnh thời gian lỗi.
7. Cung cấp việc quản lý bộ nhớ.
8. Cho phép thực hiện chương trình ẩn (background).
9. Cho phép phát triển chương trình online.

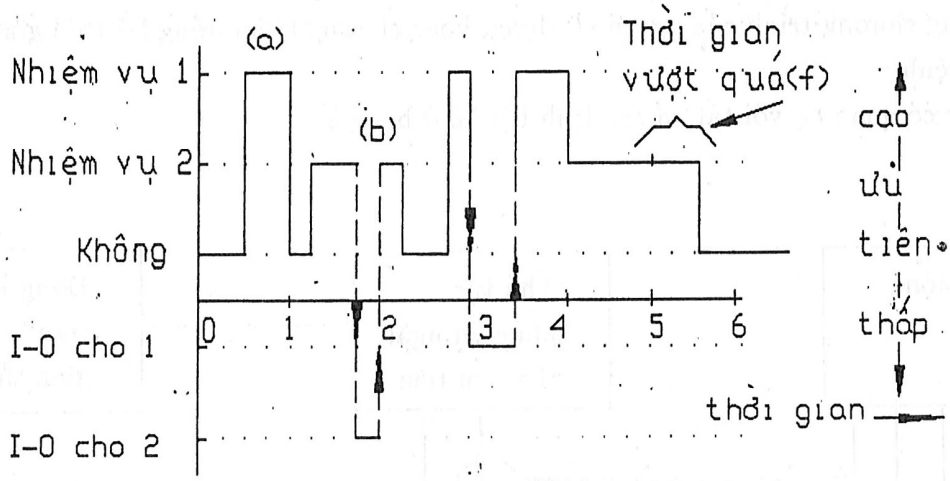
Sơ đồ cấu trúc và khối cơ bản các hệ điều hành thời gian thực được mô tả trong hình 15

Hoạt động của RTOS trong điều khiển quá trình.

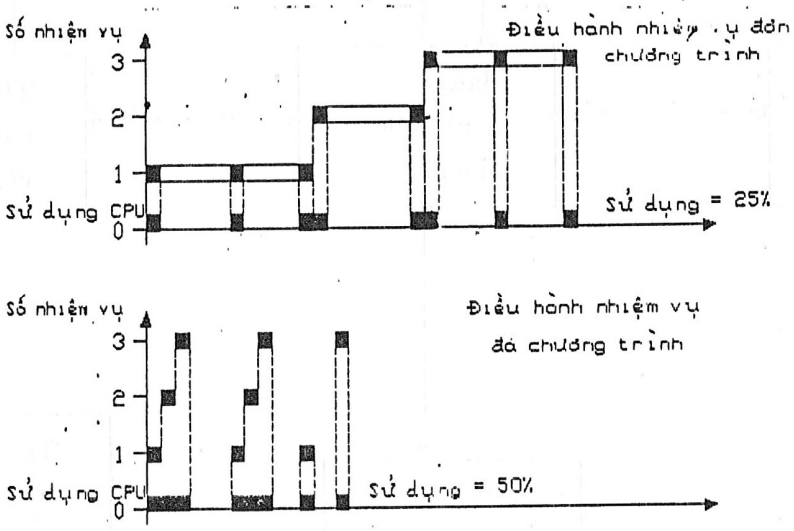
Xây dựng hệ điều hành thời gian thực

Khi xây dựng một hệ điều hành thời gian thực cần chú ý đáp ứng thêm các yêu cầu sau:

1. Các ngắt chỉ bị cấm với thời gian rất ngắn (đủ thời gian cho phát hiện và tiến hành ngắt).



Hình 13: Hệ điều hành đơn nhiệm



Hình 14: Các hệ điều hành thời gian thực

2. Cấu trúc mức ưu tiên điều chỉnh được bằng phần mềm linh động và mềm dẻo. Các ngắt cứng được tổ hợp ở một mức duy nhất.

3. Người sử dụng không cần thiết đi sau vào các chi tiết, cấu trúc của hệ điều hành mà chỉ cần làm quyen với các mô tả về việc hội thoại với nó.

Ta hãy đi sâu vào các chức năng của các modul trong hình 15 của hệ điều hành thời gian thực:

1. Khối khởi động

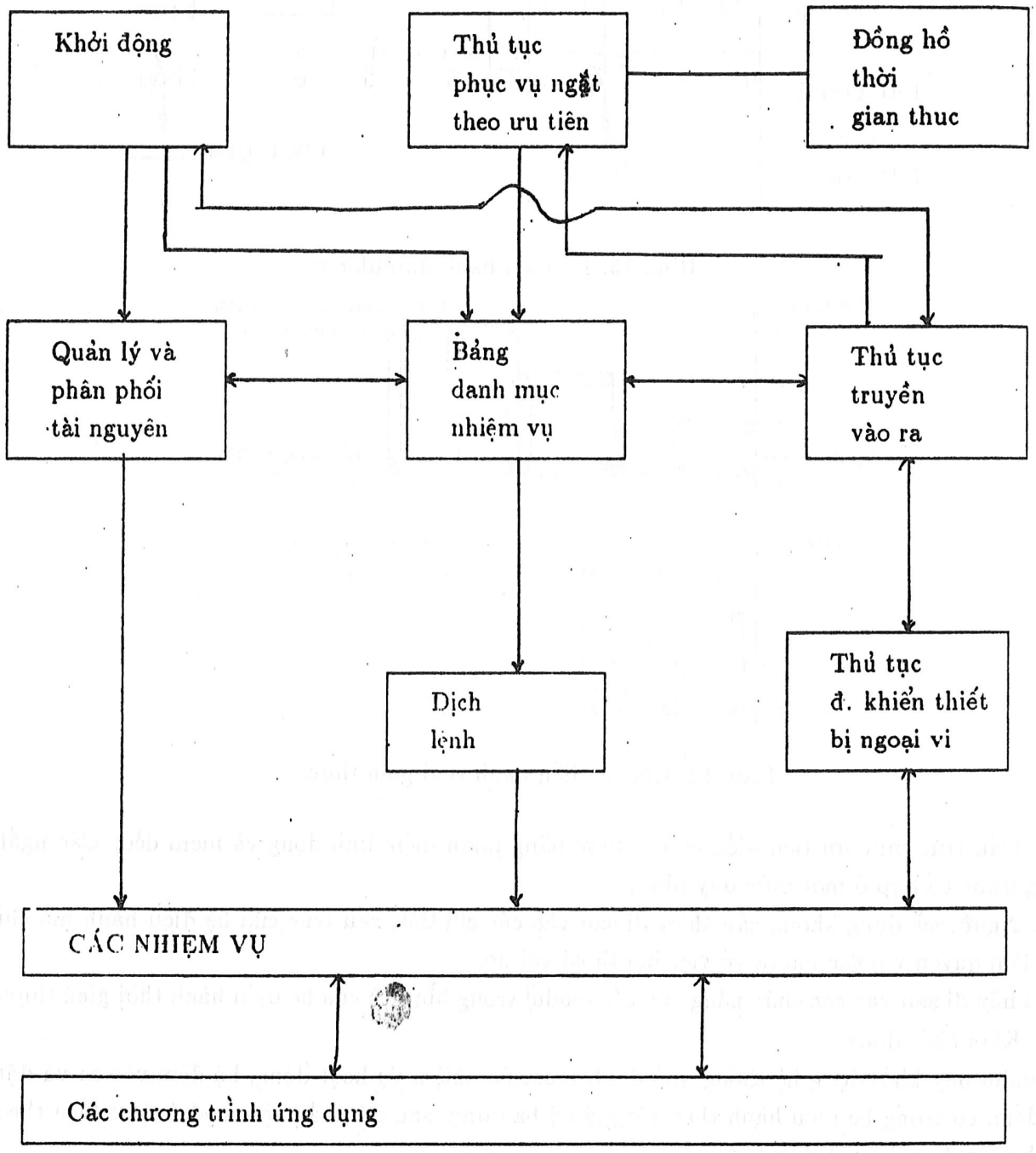
Modul này khởi động hệ thống, mở danh mục các nhiệm vụ hoạt động, bộ đệm vào ra và đặt bọ đếm, cơ trong hệ điều hành theo các giá trị ban đầu, sau đó nhảy vào modul thực hiện theo danh mục để chờ ngắt đầu tiên.

2. Khối trả lời ngắt

Sau khi cất các trạng thái của nhiệm vụ bị ngắt và nhận biết nguồn của ngắt, phần thủ tục n cho phép nguồn ngắt đó đưa vào hoạt động. Nguồn ngắt có thể là một phần ngắn của chươ trình để tiến hành sử dụng thủ tục truyền thiết bị ngoại vi (ví dụ chuyển các kí tự đến máy i làm hoạt động chương trình của người sử dụng, hoặc chương trình đồng hồ thời gian thực.

3. Modul lệnh

Modul này có quan hệ với tất cả các lệnh liệt kê ở bảng 2.



Hình 15: Cấu trúc cơ bản của hệ điều hành thời gian thực

4. Bảng danh mục

Việc xác định nhiệm vụ được trung tâm xử lí CPU thực hiện. Danh sách nhiệm vụ hoạt động lược kiểm tra và nhiệm vụ có mức ưu tiên cao nhất được khởi hành. Các nhiệm vụ có thể không hể thực hiện do không đủ bộ nhớ, có thể phải chờ cổng và-ra, hoặc chờ điều kiện thuận lợi để hoát ra. Mặt khác, các nhiệm vụ cũng có thể bị treo bởi ngắt của các nhiệm vụ ưu tiên cao hơn.

5. Phân phối bộ nhớ

Phần này chứa bản đồ bộ nhớ tức thời và lưu trữ. Bản đồ này thường xuyên được cập nhập ố liệu. Các nhiệm vụ hoạt động được chuyển vào bộ nhớ tức thờ và các nhiệm vụ mới được nạp ào bộ nhớ lưu trữ. Cũng có khi cần thiết phải chuyển các nhiệm vụ ưu tiên thấp vào bộ nhớ dự rữ để cho phép nhiệm vụ ưu tiên cao hơn tiến hành.

6. Modul thủ tục truyền vào ra

Yêu cầu vào ra trong chương trình sử dụng được đưa vào danh sách vào-ra và phát lệnh cho nodul điều khiển các thiết bị ngoại vi. Mỗi loại thiết bị có một thủ tục truyền phù hợp để nối hép với phần cứng.

7. Thủ tục con và trễ

Các thủ tục con này dùng để dừng chương trình trong một khoảng thời gian cho trước hoặc để ích hoạt chương trình tại mọt thời điểm cho trước. Nó lấy các tham số đầu vào đã xếp trong anh sách các nhiệm vụ dưới sự điều khiển của đồng hồ thời gian (Còn gọi là danh sách thời ian).

8. Đồng hồ thời gian thực

Là một thủ tục xử lí thời gian thực được khởi động bằng ngắt thời gian. Khi ngắt này xảy ra, hủ tục kiểm tra danh sách theo thời gian và kích hoạt các nhiệm vụ tại thời điểm đó. Nhịp đồng ồ có thể đặt lại cho phù hợp với quá trình điều khiển.

Ví dụ 2.

Sau đây là một ví dụ mô tả hoạt động của một vòng điều khiển số. Vòng điều khiển DDC được ô tả qua ba modul chương trình sau:

1- Thu nhậ số liệu tương tự (ADA)

2- Thuật toán DDC (DDCA)

3- Đưa kết quả ra tương tự (AO)

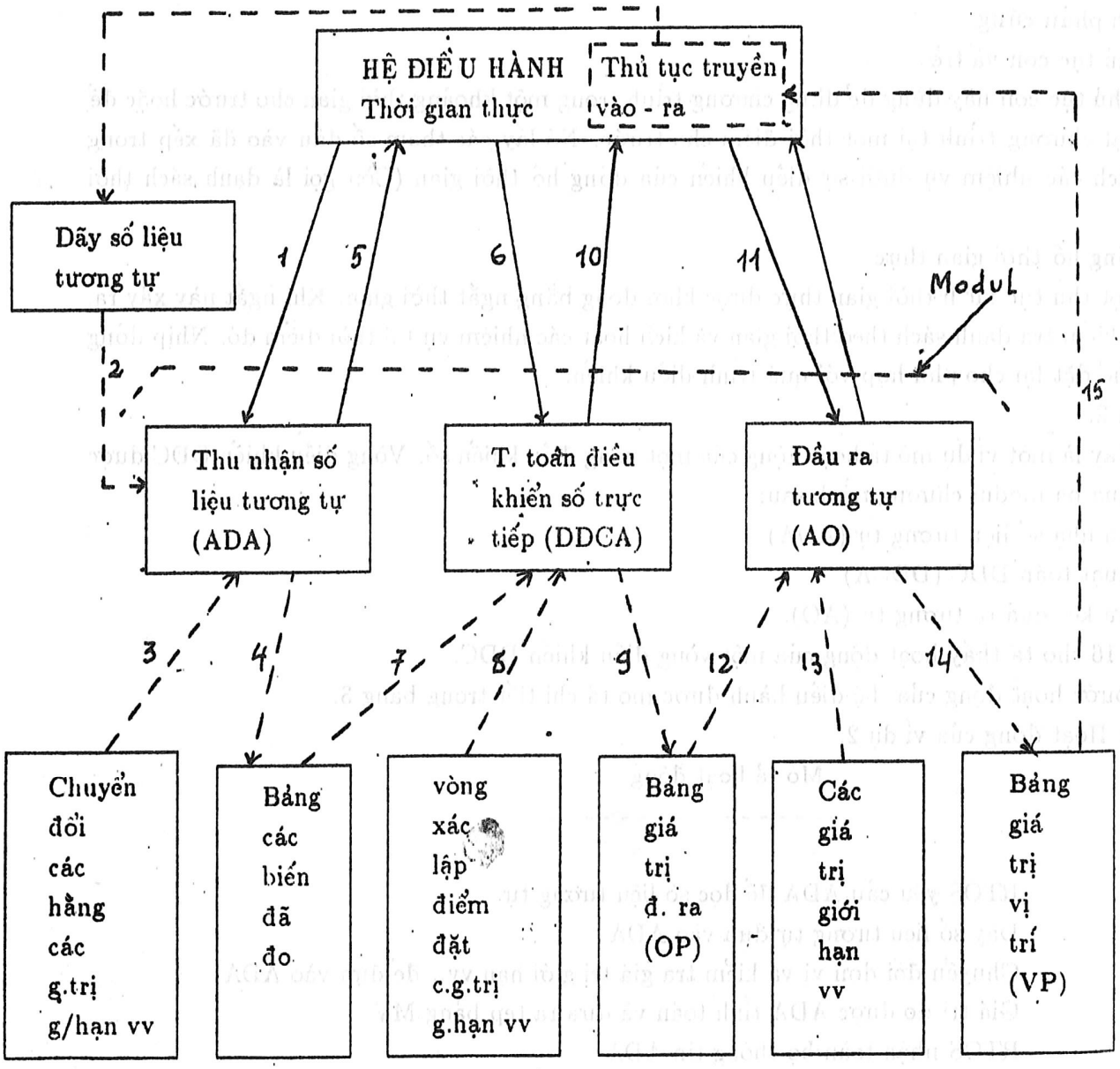
Hình 16 cho ta thấy hoạt động của một vòng điều khiển DDC.

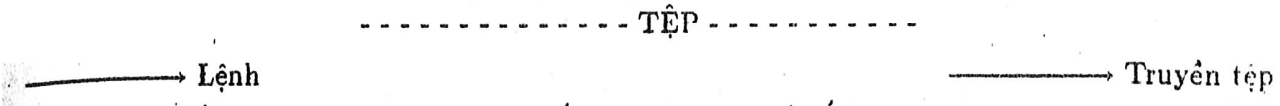
Các bước hoạt động của hệ điều hành được mô tả chi tiết trong bảng 3.

ảng 3: Hoạt động của ví dụ 2.

Số TT	Mô tả hoạt động
	RTOS yêu cầu ADA để đọc số liệu tương tự
	Dãy số liệu tương tự đưa vào ADA
	Chuyển đổi đơn vị và kiểm tra giá trị giới hạn,vv.. để đưa vào ADA
	Giá trị đo được ADA tính toán và đưa ra tệp bảng MV
	RTOS nhận toàn bộ thông tin ADA

Số TT	Mô tả hoạt động
6	TROS yêu cầu DDCA để tính toán thuật toán điều khiển
7	Đưa giá trị đã đo vào DDCA
8	Các thong số điều khiển, giá trị đặt trước, các giá trị giới hạn vv.. đưa vào DDCA
9	Chức năng điều khiển được DDCA tính toán đưa tới tệp giá trị đầu ra
10	RTOS nhận toàn bộ thông tin DDCA
Số TT	Mô tả hoạt động
11	RTOS yêu cầu AO để đưa ra thế hiệu tương tự
12	Giá trị đưa ra chuyển tới AO
13	Giá trị giới hạn vv.. chuyển vào AO
14	Vị trí của van được chuyển tới tệp vị trí van
15	Giá trị vị trí van gửi tới kích đầu ra của RTOS
	RTOS nhận toàn bộ thông tin AO.





Hình 16: Sơ đồ hoạt động khi cập nhập số liệu vòng DDC ở ví dụ 2.

4. Một vài lưu ý về các hệ điều hành thời gian thực

Các vấn đề nêu trong các phần trên mô tả hoạt động của một hệ điều hành phù hợp cho các ứng dụng thời gian thực cỡ nhỏ đến trung bình. Đối với những hệ lớn cần lưu ý một số vấn đề sau:

Phân phối số liệu chung

Xung đột có thể xảy ra khi vài nhiệm vụ cùng xam nhập một vùng số liệu chung. Vấn đề này không xảy ra nếu hệ điều hành chỉ thực hiện 1 nhiệm vụ tại một thời điểm. Giả sử nhiệm vụ A lấy tin tức từ đầu của danh sách (1) và xóa các vị trí đó về 0 (2). Nhiệm vụ B ưu tiên cao hơn đưa tin tức vào đầu danh sách. Nếu nhiệm vụ B ngắt nhiệm vụ A vào giữa bước 1 và 2 thì tin tức mới từ nhiệm vụ B sẽ bị xóa về 0. Đây là loại lỗi hệ điều hành không thể tự sửa, vì thực tế các nhiệm vụ có thể chạy ở thời điểm bất kì, và trong một tổ hợp bất kỳ gây nên hoạt động sai trái của hệ thống. Giải quyết vấn đề này bằng cách bổ sung cờ (flag) hoặc bảng hiệu (Semaphone). Khi một vùng số liệu bị xâm nhập, cờ hoặc bảng hiệu nhận giá trị 1. Trong ví dụ trên nhiệm vụ B sẽ để nhiệm vụ A kết thúc trước khi nó xâm nhập số liệu.

Phân phối thủ tục con chung

Thủ tục con có thể được nhiều nhiệm vụ khác nhau sử dụng. Thủ tục con gồm phần cố định (chương trình) và phần động (các biến). Khi thủ tục con đang phục vụ một nhiệm vụ ưu tiên thấp, nếu nhiệm vụ ưu tiên cao hơn yêu cầu, nó sẽ cất (và nạp lại) phần động. Điều này được thực thi bằng cách tạo một thủ tục con vào lại (REETRANT). Nghĩa là các biến được cất bằng thủ tục ngắt để phục vụ cho việc nạp lại cho nhiệm vụ trước.

Phân phối thiết bị chung

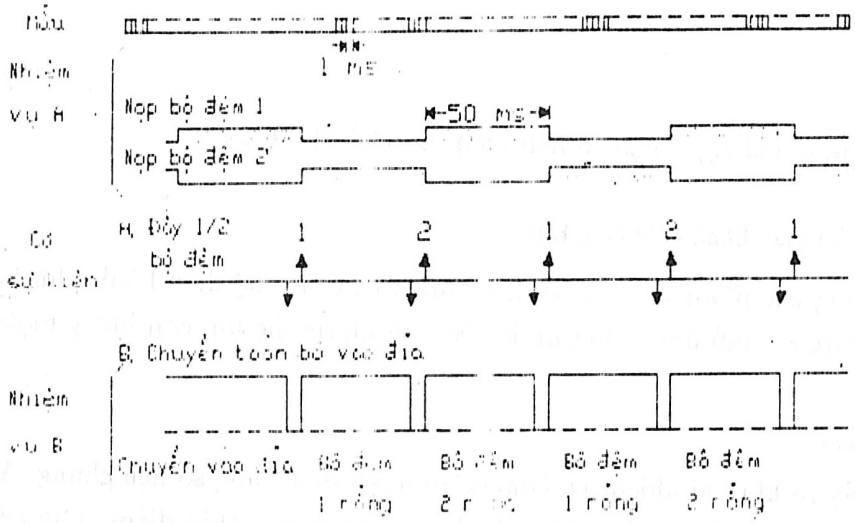
Ở đây cũng giống như phân phối số liệu chung, các thiết bị vào ra thông thường hoạt động trong thời gian riêng của chúng (thời gian thực) và không thể ngắt giữa chừng. Ví dụ việc in không hề bị ngắt bằng một chuỗi kí tự mới. Các nhiệm vụ sẽ phân phối tất cả các thiết bị chung như máy in, thiết bị cuối, đĩa, biến đổi AD vv.. Hệ điều hành chỉ có thể đồng ý hoặc từ chối việc thâm nhập tới nhiệm vụ tùy theo công việc.

Đồng bộ nhiệm vụ

Các sự kiện (events) tạo ra sự phụ thuộc lẫn nhau giữa các nhiệm vụ, giữa các ngắt và các thay đổi trạng thái tài nguyên. Để đồng bộ các nhiệm vụ cùng hoạt động ở tốc độ khác nhau người ta sử dụng cờ sự kiện (event flag).

Ví dụ số liệu đo được cắt mẫu với khoảng thời gian 1 mS, và cứ 50 đến 100 mẫu phải cất chúng lên đĩa. Vấn đề là phải chuyển số liệu với cờ hiệu như thế nào để không có lỗi.

Hình 17 giới thiệu trình tự các nhiệm vụ. Nhiệm vụ A nạp đầy 2 bộ đệm 50 mẫu (1) và 2 luân phiên nhau. Mỗi lần bộ đệm đầy thì cờ sự kiện đưa lên và nhiệm vụ B khởi hành. Nhiệm vụ B



Hình 17: Đồng bộ nhiệm vụ bằng cờ sự kiện

nhanh chóng (dưới 1mS) chuyển số liệu trong bộ đệm vào đĩa (việc này có thể thực hiện tự động bằng phần cứng) và đưa cờ hiệu của nó lên, cho phép khởi động lại nhiệm vụ A để lấp đầy bộ đệm tiếp, như vậy không có số liệu nào bị bỏ qua và cả 2 nhiệm vụ hoạt động đồng bộ.

Sự chết tắc (Deadblock)

Khi 2 nhiệm vụ cùng đòi hỏi tài nguyên mà nhiệm vụ khác đang chiếm, thì cả 2 đều đứng chờ gây nên sự chết tắc. Điều này có thể diễn giải như sau: Nhiệm vụ A yêu cầu tài nguyên 1 và

– Nhiệm vụ B yêu cầu tài nguyên 2 và 1

Khi nhiệm vụ A kết thúc việc sử dụng tài nguyên 1, nó không thể sử dụng tài nguyên 2 được vì nhiệm vụ B đang dùng và ngược lại cũng vậy. Tình trạng chết tắc đã xuất hiện. Ta giải quyết vấn đề này bằng cách đảm bảo tài nguyên được giải phóng khi kết yêu cầu, không nhường cho nhiệm vụ 2 tài nguyên đã bị chiếm bởi nhiệm vụ 1, hoặc từ chối yêu cầu của nhiệm vụ 2 bất cứ một tài nguyên nào đã được dành cho nhiệm vụ 1. Các giải pháp này có khả năng làm giảm mâu thuẫn trong hoạt động của các nhiệm vụ.

5. Kết luận

Xây dựng hệ điều hành thời gian thực với các đặc trưng được mô tả ở trên (và các đặc trưng khác chưa được đề cập đến) có thể có các xung đột giữa các mục tiêu mà khó có thể tránh khỏi. Nguyên tắc tốt nhất là xét kỹ yêu cầu ứng dụng rồi tạo ra hệ thống phù hợp. Khi đó hệ thống chỉ chứa các đặc trưng yêu cầu tối thiểu và giảm được các khả năng xung đột.

Một cách tránh xung đột là việc phân bố các nhiệm vụ của hệ điều hành thời gian thực ra 2 loại các nhiệm vụ hiện (foreground) và các nhiệm vụ ẩn (background). Các nhiệm vụ foreground các nhiệm vụ hoạt động bộ chặt chẽ (kiểm tra báo động, điều khiển số trực tiếp, nhập số liệuvv. Các nhiệm vụ background là các nhiệm vụ hoạt động ít nghiêm khắc hơn. (Nén và phân tích liệu, truyền số đo về trung tam, phát triển chương trìnhvv..). Bằng cách này ta có thể thực hiện các nhiệm vụ trong môi trường đơn chương trình đảm bảo các yêu cầu khắt khe của hệ thống điều khiển.

Tài liệu tham khảo

1. George C. barney, Intelligent instrumentation, Prentice/Hall International.
2. C. Verkerk, Introduction to real time operating systems, September 19,1990, Real time College. Trieste 1-27 October 1990.
3. Real time operating system RT/68.
4. David M. Auslander & Paul Sagues, Microprocessors for measurement and control, Osborn/Mc Graw-Hill Berkeley. California.

Abstract

On the Real Time Operating Systems for Measurements and Automatic Control

The paper deals with the design and functions of Real-time Operating Systems. It gives an overlook for multitasking, priority and enterrupt handling as well as basic requirements and structure of an Real-time Operating System. Some examples are also given for illustration of the principles.