**CÁC KHÁI NIỆM TỔNG QUAN VỀ HỆ THỜI GIAN THỰC**

1. Khái niệm chung về hệ thống thời gian thực:

Khái niệm về hệ thống máy tính kỹ thuật số thời gian thực là một khái niệm mới nổi so với hầu hết các lý thuyết và thực hành kỹ thuật. Khi được yêu cầu hoàn thành một nhiệm vụ hoặc cung cấp một dịch vụ trong thời gian thực, cách hiểu thông thường là nhiệm vụ này phải được thực hiện theo yêu cầu và được hoàn thành trong khi người yêu cầu đợi hoàn thành như một phản hồi đầu ra. Nếu phản hồi cho yêu cầu quá chậm, người yêu cầu có thể coi việc thiếu phản hồi là một lỗi.

Khái niệm về tính toán thời gian thực thực sự không có gì khác biệt. Các yêu cầu về dịch vụ thời gian thực trên nền tảng điện toán kỹ thuật số thường được biểu thị bằng ngắt không đồng bộ. Cụ thể hơn, các đầu vào cấu thành yêu cầu dịch vụ thời gian thực chỉ ra một sự kiện trong thế giới thực được hệ thống cảm nhận — ví dụ: một khung video mới đã được số hóa và đặt trong bộ nhớ để xử lý. Nền tảng máy tính bây giờ phải xử lý đầu vào liên quan đến yêu cầu dịch vụ và tạo ra phản hồi đầu ra trước thời hạn quy định so với một sự kiện được cảm nhận trước đó. Hệ thống điện toán kỹ thuật số thời gian thực phải tạo ra phản hồi theo yêu cầu trong khi người dùng và / hoặc hệ thống chờ đợi. Sau thời hạn được thiết lập cho phản hồi, liên quan đến thời gian yêu cầu, người dùng bỏ cuộc hoặc hệ thống không đáp ứng yêu cầu nếu không có phản hồi nào được đưa ra.

Một cách phổ biến để định nghĩa thời gian thực như một danh từ là thời gian mà một quá trình diễn ra hoặc xảy ra. Được sử dụng như một tính từ, thời gian thực liên quan đến các ứng dụng hoặc quy trình máy tính có thể đáp ứng với giới hạn thấp đối với các yêu cầu của người dùng. Một trong những cách tốt nhất và chính xác nhất để xác định thời gian thực cho các hệ thống máy tính là làm rõ ý nghĩa của hành vi thời gian thực chính xác. Một hệ thống thời gian thực chính xác phải tạo ra phản hồi đầu ra chính xác về mặt chức năng (về mặt kỹ thuật và toán học) trước thời hạn xác định rõ ràng liên quan đến yêu cầu dịch vụ.

Khái niệm hệ thống nhúng có lịch sử tương tự và liên quan đến hệ thống thời gian thực và chủ yếu là phạm vi thu hẹp để loại trừ các nền tảng máy tính để bàn đa năng có thể được đưa vào hệ thống thời gian thực. Ví dụ, trung tâm điều khiển sứ mệnh của NASA Johnson Space Center bao gồm một số lượng lớn các máy trạm để bàn thương mại để xử lý dữ liệu đo từ xa gần thời gian thực. Thông thường, các hệ thống thời gian thực trên máy tính để bàn chỉ cung cấp các dịch vụ thời gian thực mềm hoặc các dịch vụ gần thời gian thực hơn là các dịch vụ thời gian thực cứng. Các hệ thống nhúng thường cung cấp các dịch vụ thời gian thực cứng hoặc hỗn hợp các dịch vụ thời gian thực cứng và mềm.

Ngành công nghiệp hệ thống nhúng thời gian thực có đầy đủ các thuật ngữ chuyên ngành đã phát triển như một tập hợp con của thuật ngữ hệ thống máy tính nói chung. Để giúp bạn với thuật ngữ đó, cuốn sách này bao gồm bảng chú giải thuật ngữ các định nghĩa phổ biến trong ngành. Mặc dù cuốn sách này cố gắng xác định thuật ngữ được nói theo ngữ cảnh, đôi khi bảng thuật ngữ có thể giúp ích nếu bạn muốn đọc văn bản không theo thứ tự hoặc khi định nghĩa ngữ cảnh không rõ ràng ngay lập tức.

1. Lịch sử ra đời của hệ thống thời gian thực

Nguồn gốc của thời gian thực xuất phát từ lịch sử gần đây của quá trình vận hành sử dụng nền tảng máy tính kỹ thuật số. Trên thực tế, một văn bản xác thực ban đầu về khái niệm này đã được xuất bản vào năm 1965 [Martin65]. Khái niệm thời gian thực cũng bắt nguồn từ mô phỏng máy tính, trong đó một mô phỏng chạy ít nhất cũng nhanh như quy trình vật lý trong thế giới thực mà nó mô hình được cho là chạy trong thời gian thực. Nhiều mô phỏng phải đánh đổi giữa việc chạy bằng hoặc nhanh hơn thời gian thực với độ trung thực của mô hình thấp hơn hoặc nhiều hơn. Điều này cũng đúng đối với giao diện người dùng đồ họa thời gian thực (GUI), chẳng hạn như giao diện được cung cấp bởi các công cụ trò chơi máy tính. Không muộn hơn văn bản năm 1965 của Martin về các hệ thống thời gian thực, một bài báo cuối cùng đã được xuất bản đặt nền tảng cho định nghĩa toán học về thời gian thực cứng— “Các thuật toán lập lịch trình cho đa chương trình trong môi trường thời gian thực cứng” [Liu73 ]. Liu và Layland cũng định nghĩa khái niệm thời gian thực mềm vào năm 1973; tuy nhiên vẫn chưa có định nghĩa chính thức được chấp nhận rộng rãi về thời gian thực mềm. Các tài liệu nghiên cứu và công việc quan trọng đã được hoàn thành để xác định các hệ thống QoS (Chất lượng dịch vụ) trong đó các hệ thống đôi khi được phép bỏ lỡ thời hạn hoặc sử dụng các chiến lược trong đó độ trễ khởi động và bộ đệm được sử dụng để cho phép co giãn trong các hệ thống thời gian thực. Phạm vi mở rộng của các khái niệm thời gian thực mềm và việc sử dụng hệ điều hành nỗ lực hết sức cho các yêu cầu thời gian thực mềm đã được thêm vào ấn bản thứ hai của cuốn sách này trong Chương 11 dành cho độc giả làm việc với Linux nhúng và các yêu cầu không nghiêm ngặt như thực cứng- thời gian, nơi thời hạn không bao giờ được bỏ lỡ vì theo định nghĩa, điều này có nghĩa là toàn bộ hệ thống bị lỗi.

Khái niệm về hệ thống thời gian thực cứng được hiểu rõ hơn dựa trên kinh nghiệm và các vấn đề nhận thấy với các hệ thống thực địa — một trong những ví dụ nổi tiếng nhất sớm là quá tải hướng dẫn mô-đun mặt trăng Apollo 11. Hệ thống Apollo 11 bị quá tải tài nguyên CPU có nguy cơ khiến các dịch vụ hướng dẫn xuống đất bị bỏ lỡ thời hạn và dẫn đến việc hủy bỏ chuyến hạ cánh đầu tiên lên Mặt trăng. Trong khi xuống mô-đun Mặt Trăng và sử dụng hệ thống radar, phi hành gia Buzz Aldrin ghi nhận cảnh báo của hệ thống dẫn đường bằng máy tính. Như đã kể lại trong cuốn sách Thất bại không phải là một lựa chọn [Kranz00], đài phát thanh Buzz, “Cảnh báo chương trình. Đó là 1202. " Eugene Kranz, giám đốc điều hành sứ mệnh của Apollo 11, tiếp tục giải thích, “Báo động cho chúng ta biết rằng máy tính đang chậm trễ trong công việc của nó. Nếu tiếp tục báo động, các cập nhật về hướng dẫn, điều hướng và hiển thị phi hành đoàn sẽ trở nên không đáng tin cậy. Nếu các báo động được duy trì, máy tính có thể ngừng hoạt động, có thể hủy bỏ nhiệm vụ. " Cuối cùng, dựa trên kinh nghiệm về tình trạng quá tải này có được trong mô phỏng, quyết định là nhấn và bỏ qua báo động — như chúng ta đều biết, Đại bàng đã hạ cánh và Neil Armstrong sau đó đã đặt chân lên Mặt trăng một cách an toàn. Nói chung, làm thế nào bạn biết rằng một hệ thống đang bị quá tải đối với tài nguyên CPU, bộ nhớ hoặc IO?

Rõ ràng, sẽ có lợi nếu duy trì một số biên độ tài nguyên khi chi phí thất bại quá cao đến mức có thể chấp nhận được (như trường hợp của tàu đổ bộ mặt trăng), nhưng biên lợi nhuận bao nhiêu là đủ? Khi nào là an toàn để tiếp tục hoạt động bất chấp việc thiếu hụt tài nguyên? Trong một số trường hợp, sự thiếu hụt tài nguyên có thể chỉ là tình trạng quá tải tạm thời mà từ đó hệ thống có thể phục hồi và tiếp tục cung cấp dịch vụ đáp ứng các yêu cầu thiết kế. Cảnh báo 1202 có thể không phải là nguyên nhân gốc rễ của quá tải và trên thực tế, một số tài khoản chỉ ra báo động 1201 là nguyên nhân, nhưng mấu chốt là tình trạng quá tải bộ xử lý được chỉ ra bởi 1202 là kết quả của nhiều máy tính hơn mức có thể được xử lý bởi thời hạn yêu cầu khi việc bổ sung xử lý cảnh báo đã được thêm vào khối lượng công việc bình thường. Peter Adler đưa ra một tài khoản chi tiết hơn cho những người đọc quan tâm đến các tài khoản lịch sử chính xác do các kỹ sư có liên quan trực tiếp đưa ra [Adler 98].

Kể từ Apollo 11, các vấn đề thời gian thực thú vị hơn được quan sát tại hiện trường đã cho thấy thiết kế hệ thống thời gian thực thậm chí còn phức tạp hơn so với việc đảm bảo lợi nhuận đơn thuần. Ví dụ, tàu vũ trụ Mars Pathfinder gần như bị mất do sự cố xử lý thời gian thực. Vấn đề không phải do quá tải, mà là sự đảo ngược ưu tiên khiến thời hạn bị bỏ lỡ mặc dù có biên CPU hợp lý. Kịch bản đảo ngược trước nó của Pathfinder được mô tả chi tiết trong Chương 6, “Dịch vụ đa nguồn”. Như bạn sẽ thấy, việc đảm bảo quyền truy cập an toàn, loại trừ lẫn nhau vào bộ nhớ dùng chung có thể gây ra đảo ngược ưu tiên. Trong khi truy cập an toàn là cần thiết để đảm bảo tính đúng đắn của chức năng, thì thời hạn phản hồi đáp ứng cũng là một yêu cầu đối với các hệ thống thời gian thực. Một hệ thống thời gian thực phải tạo ra các câu trả lời đúng về mặt chức năng đúng thời hạn, trước thời hạn cho tính đúng đắn của toàn bộ hệ thống. Với một số kinh nghiệm phát triển hệ thống, hầu hết các kỹ sư đều quen thuộc với cách thiết kế và kiểm tra một hệ thống cho đúng chức năng. Hơn nữa, hầu hết các kỹ sư phần cứng đều quen thuộc với các phương pháp thiết kế thời gian logic kỹ thuật số và xác minh tính đúng đắn. Khi phần cứng, phần sụn và phần mềm được kết hợp trong một hệ thống nhúng thời gian thực, thời gian phản hồi phải được thiết kế và kiểm tra để đảm bảo rằng hệ thống tích hợp đáp ứng các yêu cầu về thời hạn. Điều này yêu cầu thiết kế và kiểm tra cấp hệ thống vượt ra ngoài các phương pháp phần cứng hoặc phần mềm thường được sử dụng.

1. Các tiêu chuẩn của hệ thời gian thực

Tiêu chuẩn thời gian thực POSIX (Portable Operating Systems Interface) đã được hợp nhất thành một tiêu chuẩn 1003.1 được cập nhật duy nhất kể từ năm 2013 [POSIX 1003.1]. Ban đầu, các tiêu chuẩn thời gian thực được IEEE (Viện Kỹ sư Điện và Điện tử) cũng như Nhóm Mở công nhận được viết dưới dạng phần mở rộng cụ thể cho các tiêu chuẩn cơ sở của POSIX, bao gồm:

* IEEE Std 2003.1b-2000: Đặc điểm kỹ thuật thử nghiệm cho phần 1 của POSIX bao gồm các phần mở rộng thời gian thực
* IEEE Std 1003.13-1998: Tiêu chuẩn hồ sơ thời gian thực để giải quyết các ứng dụng thời gian thực nhúng và các thiết bị có dấu chân nhỏ hơn
* IEEE Std 1003.1b-1993: Mở rộng thời gian thực
* IEEE Std 1003.1c-1995: Luồng
* IEEE Std 1003.1d-1999: Phần mở rộng thời gian thực bổ sung
* IEEE Std 1003.1j-2000: Tiện ích mở rộng thời gian thực nâng cao
* IEEE Std 1003.1q-2000: Truy tìm

1. Hệ điều hành thời gian thực

Hệ điều hành thời gian thực (RTOS) là hệ điều hành (OS) dành cho các ứng dụng thời gian thực xử lý dữ liệu và sự kiện có giới hạn thời gian được xác định nghiêm ngặt. RTOS khác với hệ điều hành chia sẻ thời gian, chẳng hạn như Unix, quản lý việc chia sẻ tài nguyên hệ thống với bộ lập lịch, bộ đệm dữ liệu hoặc ưu tiên tác vụ cố định trong môi trường đa nhiệm hoặc đa chương trình. Các yêu cầu về thời gian xử lý cần được hiểu đầy đủ và ràng buộc hơn là chỉ giữ ở mức tối thiểu. Tất cả quá trình xử lý phải xảy ra trong các ràng buộc đã xác định. Hệ điều hành thời gian thực hướng sự kiện và ưu tiên, có nghĩa là hệ điều hành có khả năng giám sát mức độ ưu tiên liên quan của các tác vụ cạnh tranh và thực hiện các thay đổi đối với mức độ ưu tiên của tác vụ. Các hệ thống hướng sự kiện chuyển đổi giữa các nhiệm vụ dựa trên mức độ ưu tiên của chúng, trong khi các hệ thống chia sẻ thời gian chuyển đổi nhiệm vụ dựa trên các ngắt đồng hồ.

Nhiều hệ thống nhúng thời gian thực bao gồm một RTOS, cung cấp các giao diện lập lịch CPU, quản lý bộ nhớ và trình điều khiển cho IO khi khởi động hoặc chương trình cơ sở BSP (Board Support Package). Trong văn bản này, rất nhiều mã được bao gồm dựa trên VxWorks RTOS của Wind River Systems hoặc Linux. VxWorks RTOS có sẵn để kiểm duyệt học thuật từ Wind River đến chương trình Đại học miễn phí. Tương tự như vậy, Linux có sẵn miễn phí trong một số bản phân phối có thể được điều chỉnh cho phù hợp với các nền tảng nhúng. Các tính năng chính mà RTOS hoặc bản phân phối Linux thời gian thực nhúng phải có bao gồm:

* Một hạt nhân hoàn toàn có thể ưu tiên trước để một tác vụ ngắt hoặc thời gian thực có thể ưu tiên cho bộ lập lịch hạt nhân và các dịch vụ hạt nhân.
* Độ trễ gián đoạn giới hạn tốt thấp.
* Độ trễ chuyển đổi ngữ cảnh quy trình, tác vụ hoặc luồng có giới hạn tốt thấp.
* Khả năng kiểm soát hoàn toàn tất cả tài nguyên phần cứng và ghi đè bất kỳ quản lý tài nguyên hệ điều hành tích hợp sẵn nào.
* Các công cụ theo dõi thực thi.
* Các công cụ biên dịch chéo, gỡ lỗi chéo và giao diện từ máy chủ đến đích để hỗ trợ phát triển mã trên bộ vi xử lý nhúng.
* Hỗ trợ đầy đủ cho giao tiếp, điều khiển và lập lịch tác vụ đồng bộ và không đồng bộ POSIX 1003.1b (Hiện được tích hợp trong IEEE POSIX 1003.1 2013).
* Các tùy chọn an toàn đảo ngược ưu tiên cho các bán phần loại trừ lẫn nhau được sử dụng với lập lịch trình chạy đến hoàn thành ưu tiên - ưu tiên trước (semaphore kết hợp chung được đề cập trong văn bản này bao gồm các tính năng mở rộng các khái niệm ban đầu cho các semaphores do Dijkstra giới thiệu). Lưu ý rằng đối với Linux CFS (Bộ lập lịch biểu hoàn toàn công bằng), các tính năng loại trừ lẫn nhau an toàn với nghịch đảo là không cần thiết, nhưng nếu lập lịch mặc định cho các luồng được thay đổi thành FIFO (Nhập trước, Xuất trước) trong Linux và các ưu tiên theo thời gian thực được chỉ định, thì cần có một số phương pháp để đối phó với khả năng đảo ngược mức độ ưu tiên không bị ràng buộc.
* Khả năng khóa dải địa chỉ bộ nhớ vào bộ nhớ cache.
* Khả năng khóa dải địa chỉ bộ nhớ vào bộ nhớ hoạt động nếu là bộ nhớ ảo.
* Bộ nhớ với phân trang được thực hiện.
* Đóng dấu thời gian chính xác cao, bộ hẹn giờ khoảng thời gian, đồng hồ thời gian thực và bộ đếm thời gian ảo.

1. So sánh hệ điều hành thời gian thực và hệ điều hành tiêu chuẩn

Hầu hết chúng ta đều quen thuộc với hệ điều hành hoặc hệ điều hành mà chúng ta sử dụng trên máy tính của mình. Hệ điều hành phổ biến nhất cho máy tính cá nhân bao gồm Windows của Microsoft, OS X của Apple và rất nhiều biến thể Linux có thể lấy từ các nhà phát triển tương ứng của họ. Những gì hầu hết mọi người không biết là Hệ điều hành thời gian thực hoặc thường được gọi bằng từ viết tắt RTOS. Đây là những hệ điều hành được sử dụng cho các ứng dụng chuyên biệt hơn yêu cầu phản hồi gần với thời gian thực nhất có thể. Sự khác biệt đáng kể nhất giữa cả hai là cách họ tiếp cận từng nhiệm vụ. Các hệ điều hành tiêu chuẩn tập trung vào việc thực hiện nhiều tính toán trong khoảng thời gian ngắn nhất trong khi RTOS lại nhấn mạnh vào việc có thời gian phản hồi có thể dự đoán được.

Hệ điều hành tiêu chuẩn được sử dụng rộng rãi ngày nay, một phần do sự phổ biến nhanh chóng của máy tính cá nhân. Các thiết bị sử dụng hệ điều hành tiêu chuẩn, ngoài máy tính và máy tính xách tay, cũng bắt đầu xuất hiện. RTOS được sử dụng trong các lĩnh vực chuyên biệt hơn, nơi thời gian phản hồi quan trọng hơn nhiều so với khả năng xử lý một số lượng lớn các lệnh trong một khoảng thời gian nhất định. Ví dụ trong số đó là máy tính quét các cấp độ và trạng thái trong một cơ sở. Điều quan trọng là các màn hình nhìn thấy các thay đổi xảy ra ngay lập tức.

Hầu hết các hệ điều hành sử dụng kiến ​​trúc chia sẻ thời gian trong đó mỗi tác vụ được gán một khoảng thời gian nhỏ để thực hiện các lệnh của nó trước khi chuyển sang tác vụ khác. Quá trình chuyển đổi quá nhanh nên nó thường xuất hiện dưới dạng thời gian thực đối với người dùng. Một số RTOS cũng sử dụng thiết kế này nhưng với mật độ tác vụ thấp hơn nhiều để đảm bảo rằng bộ xử lý không bao giờ được tải, điều này có thể làm tăng thời gian phản hồi. Một thiết kế khác được sử dụng cho RTOS là kiến ​​trúc hướng sự kiện. Trong thiết kế này, hệ thống chỉ chuyển đổi nhiệm vụ khi xảy ra sự kiện hoặc ngắt.

Lập trình cho RTOS nghiêm ngặt hơn nhiều so với một hệ điều hành tiêu chuẩn vì mã cần phải hoạt động nhất quán mọi lúc. Hệ điều hành tiêu chuẩn không được quan tâm vì thời gian phản hồi không có tầm quan trọng lớn trong ứng dụng của nó.