GIẢI NGÂN HÀNG
HỆ THỐNG NHÚNG PTIT

Allahu akbar!

Câu 1. Khái niệm hệ thống nhúng????

-Định nghĩa tổng quát: hệ thống nhúng là một thuật ngữ để chỉ một hệ thống có khả năng hoạt động tự trị được nhúng vào trong môi trường hay một hệ thống khác quy mô phức tạp hơn. Đó là các hệ thống tích hợp cả các phần cứng(là một hệ thống máy tính xây dựng trên cơ sở sử dụng vi xử lý- microprocessor-based sytem) và phần mềm nhúng trong phần cứng đó để thực hiện các bài toán chuyên biệt
-Định nghĩa theo tổ chức IEEE: hệ thống nhúng là một hệ thống tính toán(máy tính số)nằm trong (hay được nhúng vào) sản phẩm khác lớn hơn và rằng thông thường ẩn đối với người sử dụng. Nói rộng ra và đơn giản hơn, khi một hệ thống tính toán(có thể là PC, IPC, PCL, vi xử lý, vi hệ thống(microcontroller), DSP….) được nhúng vào trong một sản phẩm hay một hệ thống nào đó và thực hiện một số chức năng cụ thể của hệ thống đó thì ta gọi hệ thống tính toán đó là hệ thống nhúng.

1.2. HT nào là hệ thống nhúng (A,B,D,E)

A. Các hệ thống y tế.
B. Các hệ thống điều khiển quy trình công nghiệp.
C. Các thiết bị truyền thông kỹ thuật số.
D. Các hệ thống có độ tin cậy cao và rất cao.

1.3. Nêu tên một số lĩnh vực đời sống và công nghiệp trong đó có sử dụng ưng dụng của hệ thống nhúng. Kể tên thiết bị nhúng được sử dụng:
Ô tô: Đánh lửa điện tử, điều khiển động cơ, hệ thống phanh,
Y tế: Máy thẩm tách, máy pha- lọc, máy trợ tim
Mạng thông tin (WAN, LAN, thoại): Bộ định tuyến, gateway, chuyển mạch mạng

1.4. HTN mà một máy tính đa năng hay máy tính chuyên dụng...?
A. Hệ thống nhúng là máy tính chuyên dụng, dùng cho ứng dụng đặc biệt
B. Phần mềm của hệ thống nhúng là phần mềm hướng ứng dụng
C. Ví dụ về phần mềm hệ thống chạy trên hệ thống nhúng đó là: Monitor , RTOS hướng đích.

2.1 a. Hãy nêu những đặc điểm của các môi trường mà hệ thống nhúng hoạt động.
b. Cho ví dụ về một hệ thống nhúng nào đó và nêu đặc điểm môi trường mà hệ thống nhúng đó đang hoạt động.

Các hệ nhúng thường phải hoạt động trong môi trường khắc nghiệt như công nghiệp, quân đội, nó cần có độ nóng ẩm, nhiệt độ cao, rung, ẩm thấp, độ rung động lớn, nhiễu sóng điện từ..
Một số ví dụ điển hình về hệ thống nhúng các hệ thống đường dẫn trong không lưu, hệ thống định vị toàn cầu, vệ tinh. Các thiết bị gia dụng : tủ lạnh, lò vi sóng, lò nướng…môi trường nhiệt độ cao, độ ẩm lớn, độ rung lớn.

2.2. Tại sao nói hầu hết các hệ thống nhúng hoạt động với sự ràng buộc về thời gian ?
-Phần lớn các hệ thống nhúng hoạt động với sự ràng buộc thời gian: yêu cầu có thời gian cho (đáp ứng) đầu ra nhanh, đúng thời điểm, trong mối tương quan với thời điểm xuất hiện của (sự kiện) đầu vào.
-Hệ thống nhúng có khả năng đáp ứng với sự kiện bên ngoài (từ các tác nhân bị kiểm soát) nhanh nhạy, kịp thời, tức là khả năng theo thời gian thực:
+Các tác vụ có đáp ứng ràng buộc bởi thời hạn chót (deadline)
+Thời gian phát hiện lỗi phải rất ngắn (tối thiểu);
+Khi chạy các chu trình vòng lặp điều khiển bằng phần mềm phải có đáp ứng đầu ra đúng thời hạn;

2.3 a. Nêu các kiểu hoạt động cơ bản của hệ thống nhúng:
-Hệ thông minh
-Hoạt động độc lập: nhận đầu vào từ các tác nhân bị điều khiển, xử lý và cho đầu

ra. Thời gian có đầu ra (đáp ứng) phải trong một khung thời gian nhất định theo ý đồ khi thiết kế.
-Hệ liên kết tự độngHoạt động có liên kết với nhau giữa các HTN và các trung tâm kiểm soát khác.
-Hệ tự phản ứng với sự kiện
b.Hệ thụ động, nhạy cảm với sự kiện, với thời gian, tự phản ứng với sự kiện

2.4: BUS gồm các thành phần nào hợp thành ?
a.Bus của CPU bao gồm : Bus địa chỉ, Bus dữ liệu, Bus điều khiển
-Bus địa chỉ : Trong quá trình hoạt động, CPU điều khiển bus địa chỉ tới những vị trí được chuyển giao từ bên ngoài hoặc từ CPU. Địa chỉ ở đây là vị trí bộ nhớ hoặc vị trí vào/ra.xd
-Bus dữ liệu: được CPU điều khiển trong suốt chu kỳ ghi dữ liệu và được các thiết bị khác điều khiển trong suốt chu kỳ đọc, vận chuyển lệnh và dữ liệu bên trong và ngoài CPU
-Bus điều khiển: được điều khiển bởi CPU. Bus điều khiển quyết định loại của chu kỳ nào diễn ra và khi nào dữ liệu sẽ được đưa lên bus.
b. BUS HT = BUS CPU + mạnh hỗ trợ mở rộng của BUS CPU: tách tín hiệu dồn kênh, khuyếch đại, tạo các tin hiệu điều khiển riêng biệt

2.5: **Hệ thống nhúng tướng tác với môi trường vật lí như thế nào, phương tiện, công cụ gì ? Nêu một số ví dụ ?**
Hệ thống nhúng tương tác với môi trường vật lý qua nhiều phương thức :
- Qua các bộ cảm biến (sensor), ghép nối vào hệ thống nhúng bằng dây dẫn, hay không dây.
- Thông qua các bộ chuyển đổi tín hiệu ADC, DAC
-Thông qua ghép nối, hợp chuẩn dữ liệu
b. Một số ví dụ :
Phương tiện, công cụ :-Phương tiện để tương tác với môi trường vật lý : chuyển đổi tín hiệu : tương tự-số ADC, số-tương tự DAC.
- Công cụ để tương tác với môi trường vật lý : ghép nối và hợp chuẩn tín hiệu.
Các mạch ghép nối vào/ra là các mạch điện tử cho phép CPU trao đổi dữ liệu với các thiết bị ngoại vi như bàn phím, màn hình, máy in….

2.6: Cho một mô hình qui trình điều khiển công nghệ có ứng dụng hệ thống nhúng như hình sau :Khoanh vùng cho biết hệ thống nhúng là phần nào ?
Vùng Nửa trên của hình vẽ \

Các thành phần hợp thành của hệ thống nhúng : phần cứng, phần mềm, RTOS(Real Time Operating System)

2.7 :Để xây dựng một kiến trúc cho một hệ thống nhúng, phải tuân thủ 6 bước cơ bản. Các bước đó là các bước nào ?
Nêu các pha trong quá trình thiết kế một hệ thống nhúng ?
Trả lời :
6 bước cơ bản để xây dựng một kiến trúc cho một hệ thống nhúng gồm :
-Cần kiến thức tốt về phần cứng (thiết kế logic, kiến trúc máy tính, kiến trúc CPU, ngoại vi, hệ điều hành…).
-Thị trường: Cần nhận ra các yêu cầu của thị trường có tác động vào quy trình thiết kế, bao gồm : kỹ thuật, xu hướng thương mại, ảnh hưởng của chính trị, xã hội.
-Mẫu hệ thống là một mẫu mô tả của hệ thống, chứa đựng các đặc tả khác nhau về các thành phần phần cứng và phần mềm, chức năng, các liên kết, giao tiếp
-Các cấu trúc: Đây là bước tạo ra kiến trúc của HTN. Kiến trúc HTN sẽ được hình thành bằng cách phân định toàn bộ HTN thành các thành phần phần cứng, phần mềm, sau đó các thành phần đó lại được phân định đến chi tiết.
-Phân tích và đánh giá kiến trúc: kiến trúc có đạt các yêu cầu, các kiến trúc khác nhau với yêu cầu có cùng chất lượng như nhau, đánh giá xu hướng rủi ro hệ thống, hỏng hóc có thể, hiệu chỉnh.
-Viết tài liệu: Tài liệu về toàn bộ hệ thống theo các chuẩn tài liệu, Tài liệu về từng cấu trúc, Tài liệu tổng thể về kiến trúc hệ thống
-Các pha trong quá trình thiết kế một HTN : Xác định yêu cầu, phân tích, thử nghiệm đánh giá, sản xuất sản phẩm. Pha 2,3 được lặp lại

2.8 :Thế nào là phân hoạch phần cứng và phần mềm khi thiết kế một hệ thống nhúng ?Thế nào là qui trình đồng thiết kế phần cứng và phần mềm và đồng kiểm nghiệm ?
Trả lời :
a.Phân hoạch phần cứng và phần mềm khi thiết kế một HTN :
Là đặc tả, ấn định chức năng cho phần cứng và phần mềm, sau đó đi thiết kế, mô phỏng và hợp nhất.
b.Qui trình đồng thiết kế phần cứng và phần mềm và đồng kiểm nghiệm :
Trong kĩ thuật này, thiết kế phần cứng và phần mềm được tiến hành song song, các phản hồi-hiệu chỉnh thực hiện liên tục, cho tới khi có kết quả tốt nhất qua đồng kiểm nghiệm. Phần cưngs và phần mềm được chạy cùng nhau trên các máy ảo. phần cứng bằng VHDL -> máy ảo, phần mềm trên máy ảo -> hiệu chỉnh liên tục -> máy đích.

2.9:Hãy đặc tả các tác vụ khi thực hiện khởi động hệ thống nguội (cold boot) và khởi động nóng (warm boot)?Hãy cho ví dụ về cách khởi động với một loại CPU tự chọn?

Trả lời:

Các tác vụ khi thực hiện khởi động hệ thống:

Khởi động nguội (cold boot): Bật nguồn (khởi động phần cứng), đưa các vi mạch, phân vùng bộ nhớ, không gian địa chỉ…vào trạng thái ban đầu. Sau hết chương trình nạp hệ thống (boot loader) "nhảy" tới địa chỉ RESTART hay START của phần mềm hệ thống để chuyển điều khiển cho nó.

- Khởi động nóng (waerm boot): CTRL + DEL, bỏ qua một số test phần cứng.

1. Ví dụ về cách khởi động với một loại CPU tự chọn:

Intel CPU 8085: RESET →IP = 0x0000 là địa chỉ của EPROM: khởi động chế độ nối với Console (Keyboard), sau đó nhảy về chương trình khởi động hệ: JMP CLDST (khởi động nguội) ở địa chỉ 0x01F1.

2.10:**Khi thiết kế HTN cần xây dựng một mô hình chính tắc (formal model) với các yêu cầu đặt ra. Vậy các yêu cầu đó là những yêu cầu gì?**

Trả lời:

Xây dựng một mô hình chính tắc của hệ thống với các yêu cầu sau đây:

-Xác định chức năng: xây dựng tập các mối quan hệ tường minh hay không tường minh liên quan tới đầu vào/đầu ra và thông tin trạng thái bên trong của hệ thống.

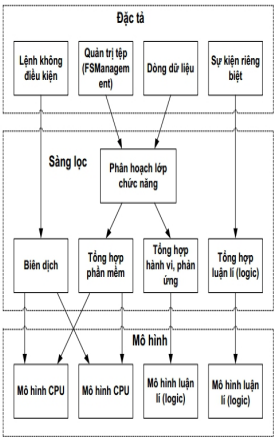
-Xây dựng tập các thuộc tính mà thiết kế sẽ phải thỏa mãn. Kết hợp các thuộc tính và tập các quan hệ vào/ra, trạng thái hệ thống, xác định lại các chức năng hệ thống.

-Xây dựng tập các chỉ số hiệu năng để đánh giá thiết kế theo các tiêu chí: giá thành, năng lượng, độ tin cậy, tốc độ xử lý, kích thước…

-Xây dựng tập các khác biệt coi đó như những thách thức của thiết kế, đề ra giải pháp giải quyết.

-Thực hiện tinh lọc thiết kế để có thiết kế từ ý tưởng đến mô hình.

Xây dựng mô hình hình thức: bước sàng lọc sử dụng cách tổng hợp phần cứng và phần mềm để chuyển hóa xác định chức năng vào mô hình phần cứng của thiết kế được thể hiện trong hình dưới đây. **Ko đc bỏ hình vẽ**



3.1. Nêu các thách thức phải đối mặt khi thiết kế một hệ thống nhúng ?

-Khả năng đáp ứng với các sự kiện bên ngoài phải nhanh nhạy, kịp thời, tức là khả năng theo thời gian thực.

-Các tác vụ có đáp ứng ràng buộc bởi thời hạn chót.

-Thời gian phát hiện lỗi phải rất ngắn.

-Khi chạy các chu trình vòng lặp điều khiển bằng phần mềm phải có đáp ứng đầu ra đúng thời hạn.

-Khi chạy các chu trình vòng lặp điều khiển bằng phần mềm phải có đáp ứng đầu ra đúng thời hạn.

-Có khả năng làm việc ở môi trường khắc nghiệt.

-Có giá thành thấp hay hiệu quả hoạt động, giá thành hợp lý.

-Kích thước nhỏ gọn, nhẹ, dễ dàng dễ vận chuyển, lắp đặt.

-Tiêu thụ năng lượng thấp, khả năng sử dụng nguồn pin, ắc quy.

-Hoạt động tin cậy, chịu lỗi cao

-Tin cậy : đáp ứng các dịch vụ yêu cầu đúng thời hạn sau thời gian từ t_0->t.

-Độ tin cậy cao nếu đạt ~ 10^{-9} sự cố / giờ.

-Tính sẵn sàng cao : nếu thời gian sửa chữa sự cố trung bình là MTTR, thì tinh sẵn sàng A=MTTF/(MTTF + MTTR)

-An toàn và bảo mật.

-Khả năng cấp phần mềm và dự phòng nâng cấp phần cứng.

3.2 :Các tiêu chí phân loại hệ thống nhúng

Tại sao HTN có sự khác nhau

- Hệ thống nhúng hoạt động ở đâu

➢ Hoạt động độc lập : nhận đầu vào từ các tác nhân bi điều khiển, xử lý cho đầu ra.Thời gian có đầu ra (đáp ứng) phải trong một khung thời gian nhất định theo ý đồ khi thiết kế.

➢ Hoạt động có liên kết với nhau giữa các HTN và các trung tâm kiểm soát khác. Loại này gọi là HTN mạng.

- Lĩnh vực ứng dụng

➢ Công cụ tính toán như các máy tính nhưng chỉ để chạy các bài toán nhất định.
➢ Xử lý tín hiệu: các thiết bị video thời gian thực, DVD player, thiết bị y tế…
➢ Truyền thông, mạng: thiết bị mạng như router, switch
➢ Hệ thống điều khiển và thu thập dữ liệu.

- Kiến trúc và quy mô

➢ HTN quy mô nhỏ với các xác định như sau :

❖ Phần cứng ít phức tạp, thiết kế với CPU đơn, loại 4, 8 bits.
❖ Phần mềm đơn giản, dùng một monitor để kiểm soát hoạt động.
❖ Công cụ phát triển phần mềm : soạn thảo chương trình, hợp ngữ và hợp ngữ chéo, môi trường phát triển hợp nhất sử dụng với vi điều khiển đã chọn.
❖ Tiêu thụ năng lượng rất ít.

➢ HTN quy mô phức tạp

❖ Phần cứng phức tạp : thiết kế với CPU 8, 16 ay 32 bits, hay sử dụng vi điều khiển.
❖ Hệ thống có cấu trúc BUS mở rộng để ghép nối các thiết bị ngoại vi.
❖ Phần mềm nhúng tinh vi, có hệ điều hành để thực hiện các nhiệm vụ, thao tác đồng thời.
❖ Công cụ lập trình: C/C++/Visual C++/Java, RTOS, mã nguồn, công cụ ki thuật: Simulator, Debugger. Môi trường phát triển hợp nhất.

➢ HTN tinh vi

❖ Phần cứng và phần mềm rất đặc biệt.
❖ Nhiều CPU và có thể mở rộng, hay các CPU có thể cấu hình được hay mảng logic lập trình được.
❖ Phát triển cho các lớp ứng dụng mới nhất khi các ứng dụng loại này cần phải có quá trình thiết kế đồng thời giữa phần cứng và phần mềm, hợp nhất các linh kiện ở hệ thống cuối cùng, sử dụng công nghệ ASIC để chế tạo CPU, vi mạch đồng xử lý

➢ HTN phần cứng hay HTN phần mềm.
➢ HTN theo an toàn sự cố, hay tự an toàn.
➢ HTN đáp ứng được bảo đảm hay đáp ứng với nỗ lực tối đa.
➢ HTN với nguồn tài nguyên đầy đủ hay nguồn tài nguyên hạn chế.
➢ HTN phản ứng ngay với sự kiện hay phản ứng với sự kiện có thời hạn.

3.2.2 Tại sao HTN có sự khác nhau

- Phần cứng đơn giản hay phức tạp do ứng dụng quyết định :HTN là dành để thực hiện các tác vụ riêng biệt Các tác vụ riêng biệt ở đây phần lớn liên quan tới các xử lý khác nhau chuyên biệt, các sự kiện, các trạng thái của một qui trình công nghệ …

- Về phần cứng, các HTN được thiết kế từ rất nhiều loại CPU nhúng và các CPU nhúng bản thân chúng lại có kiến trúc khác nhau
- Phần mềm đơn giản hay phức tạp do ứng dụng quyết định:Về phần mềm cở sở có thể từ đơn giản cho tới tinh xảo, hệ điều hành thời gian thực (RTOS-Real Time Operating System).

3.3: Hãy nêu sự khác biệt khi thiết kế hệ thống nhúng kiểu trên cùng một bo mạch :

Trả lời

(1)Hệ thống nhúng xây dựng từ bộ vi xử lí
(2)Hệ thống nhúng xây dựng từ các vi điều khiển
(1)Có CPU độc lập
(2)CPU dạng lõi chuyên biệt
(1)Có RAM, ROM, định thời
(2)RAM, ROM, định thời,
(1)I/O độc lập.
(2)I/O trong một vi mạch đơn.
(1)Khả năng mở rộng RAM, ROM, I/O tùy ý.
(2)RAM, ROM có dung lượng cố định, I/O đủ cho mục đích sử dụng.
(1)Đa năng, đắt tiền.
(2)Không đa năng,tiêu hao ít năng lượng, giá cả hợp lí cho ứng dụng nhúng.
(1)*Thiết kế chức năng*
(2)*Thiết kế chức năng*
(1)Cần phải có các vi mạch RAM, ROM hợp thành từ bên ngoài vi mạch.
2)Được thiết kế để có tất cả trong một Chip .
(1)Không thể kết nối với ngoại vi ngoại vi, cần có thêm các vi mạch hỗ trợ cho chức năng này.
(2)Năng lực tính toán được thiết kế tối ưu cho ứng dụng xác định.
(1)Tuy nhiên năng lực tính toán mạnh.
(2)Rất phù hợp để xây dựng các HTN.

3.4 Cho biết tên gọi của các kiểu kiến trúc là gì ? Kiểu kiến trúc nào là thích hợp hơn để xây dựng các hệ thống nhúng (kiểu 1 hay kiểu 2)?

Kiểu 1: kiến trúc Havard
Kiểu 2: kiến trúc Von Neumman
Điểm khác biệt cơ bản ở hai kiến trúc này là chổ nào ?
Bus ở kiến trúc Havard được tách riêng cho lệnh và dữ liệu riêng biệt
Bus ở kiến trúc Von Neumman dùng chung cho cả lệnh và dữ liệu
Tại sao ?

-kiến trúc Havard thích hợp hơn.
Vì : việc dùng tách riêng bus cho lệnh và dữ liệu sẽ giúp tốc độ xử lí nhanh hơn, việc đọc dữ liệu và thực hiện có thể xảy ra đồng thời

3.5 ADC, DAC

A . ADC biến đổi tương tự->số, số hóa cho máy tính xử lí số liệu
B . DAC biến đổi số->tương tự, tín hiệu tác động lại hệ thống tương tự bị điều khiển, ví dụ là tí hiệu phản hồi, tí hiệu hiệu chỉnh
C Để chọn một ADC cần các thông số cơ bản:Kiểu ADC cho từng loại ứng dụng/Thời gian biến đổi /Độ phân giải/Tín hiệu đối thoại với CPU

3.6:Khi nghiên cứu CPU để thiết kế một hệ thống vi xử lí như một hệ thống nhúng, có một số khái niệm sau đây

A/Một trạng thái máy : được định nghĩa là thời gian một chu kỳ xung đồng hồ hệ thống (CLK).

- Một chu kỳ máy: là tập hợp của một số các trạng thái máy để CPU hay một vi mạch khi nắm quyền kiểm soát Bus hệ thống, thực hiện xong một thao tác trên bus hệ thống.

B/ Một chu kỳ lệnh là tập các chu kỳ máy cần thiết để hoàn thành một lệnh của máy

C/Tác động: Tính toán số CLK, qui đổi thời gian, tối ưu các lệnh, tính tổng thời gian cho đáp án.

3.7:**Cho mô hình kiến trúc hệ thống như hình vẽ. Hãy giải thích chức năng của khối 'GIAO DIỆN VỚI CPU', lí do sự có mặt của khối đó ?**

Các chức năng :

- Các vi mạch khuyếch đại BUS,
- Tách địa chỉ/dữ liệu từ BUS dồn kênh của CPU,
- Tạo BUS hệ thống mở rộng.

Lý do:

Tín hiệu phát sinh từ CPU thường có công suất thấp chỉ đủ cho một số tải danh định (fan-out), không đủ để mở rộng BUS, nhất là khi bus khá dài và có nhiều thiết bị nối với nó. Chính vì thế mà hầu hết các BUS được nối một số vi mạch khuyếch bus (bus driver), về cơ bản đó là các vi mạch khuyếch đại tín hiệu số.

A.Thế nào là BUS đồng bộ ?

Có xung đồng hồ hệ thống CLK làm chuẩn cho các hoạt động BUS.
Bus đồng bộ có một tín hiệu trên đường dây BUS clock dạng sóng vuông, với tần số ví dụ, trong khoảng vài MHz ÷ GHz. Mọi hoạt động bus xảy ra đều qui chiếu vào BUS Clock, trong một số nguyên lần chu kỳ này và được gọi là chu kỳ bus.

B.Thế nào là BUS không đồng bộ ?

Bus không đồng bộ không sử dụng xung BUS clock, chu kỳ của nó có thể kéo dài tuỳ ý và có thể khác nhau đối với các cặp thiết bị khác nhau, gọi là đối thoại tuần tự bởi các tín hiệu điều khiển.
Các tín hiệu điều khiển đồng bộ hoạt động là hệ quả của mỗi thao tác giữa các vi mạch chức năng và CPU trên BUS, ví dụ như để "đối thoại".

3.8:Trình điều khiển thiết bị là gì ? Chức năng của trình điều khiển thiết bị là gì ?

- Là một phần mềm để khởi động phần cứng và phần mềm lớp cao hơn sử dụng để quản trị truy nhập vào phần cứng ghép nối vào máy tính.
- Phần mềm này tương tác trực tiếp và điều khiển phần cứng và được tổ chức ở dạng các thư viện phần mềm.
- Khi máy tính có hệ điều hành thì TĐKTB là cầu nối giữa phần cứng và hệ điều hành

A.Chức năng của trình điều khiển thiết

Ghép nối mềm giữa phần mềm hệ thống (hay CPU) và hệ thống các nguồn tài nguyên phần cứng hệ thống.
Các chức năng cơ bản như:Khởi động phần cứng ,Tắt máy ,Cấm phần cứng hoạt ,Cho phép hoạt động (Hardware Enable),Dành lấy phần cứng (Hardware Acquire).Giải phóng phần cứng (Hardware Release) Đọc/ghi dữ liệu (Hardware Read/write)Cài và tháo dỡ phần cứng (Hardware Install/Uninstall):.

B.Nhìn theo kiến trúc phần mềm máy tính, trình thiết bị được đặt ở đâu ?

Dưới phần mềm hệ thống và trên phần cứng

3.9.Thế nào là trình điều khiển xác định theo kiến trúc.

- Đi theo kiến trúc (architecture-specific) của HTN
- Quản trị phần cứng hợp nhất với CPU.
- Các kiến trúc kiểu microcontroller, hay kiến trúc Havard với bộ nhớ trên chip, cổng, vi mạch quản trị bộ nhớ (memory management Unit-MMU), các phần cứng dấu phẩy động, các ADC/DAC hợp nhất trên bo mạch, thuộc lớp này.

Thế nào là trình điều khiển tổng quát (hay trình điều khiển trên bo mạch).

Điều khiển các thiết bị nằm trên bo mạch chính nhưng không hợp nhất vào chip với CPU. Đa dụng, hổ trợ cho nhiều kiến trúc khác nhau có dùng cung một loại thiết bị tương thích.

Hãy nêu một số loại trình điều khiển thiết bị điển hình ?

Điều khiển BUS I2C, PCI, cache L2, MMU,mạng, xử lí kí tự i/o, ISR

3.10Nêu các thao tác của trình điều khiển thiết bị khi được kích hoạt ?.....

A .Khởi động HW, tắt HW, Cấm/cho phép ngắt, tranh dành tài nguyên HW, giải phóng HW, truy nhập thiết bị, đọc/ghi dữ liệu, cài/tháo dỡ phần mềm ĐKTB.
B . Thông tin về thiết bị, thuộc tính thiết bị, định danh kiểu thiết bị, cách thiết bị hoạt động, giao diện với máy tính, ghép nối kỉ thuật,
C . Khởi động các thông số cho PIC, cho phép PIC nhận ngắt, ISR: cấm ngắt (nếu cần), bảo vệ các thanh ghi, thực thi xử lí, khôi phục các thanh ghi, IRET

3.11: Hãy chọn câu trả lời cho là đúng khi nói về hệ thời gian thực:

⇨ **Trả lời :**

- Chọn B : là một hệ hoạt động có tính tiền định

Tính tiền định : Dự đoán trước được thời gian phản ứng tiêu lịch, thời gian phản ứng chậm nhất cũng như trình tự đưa ra các đáp ứng.

Ví dụ : nếu một bộ điều khiển phải xử lý đồng thời nhiều nhiệm vụ, ta phải tham gia quyết định được về trình tự thực hiện các công việc và đánh giá được thời gian xử lý mỗi công việc. ->Người sử dụng có cơ sở đánh giá về khả năng đáp ứng tính thời gian thực của hệ thống.

- Chọn C : là một hệ có khả năng cho đáp ứng kịp thời và chính xác

Các chức năng phải được thực hiện chuẩn xác. Các tính toán, xử lý phải cho ra kết quả trong một chu kì thời gian đã xác định trước. Chính xác về thời gian sẽ cho phép hệ đưa ra đáp ứng một cách kịp thời. Tuy tính chính xác thời gian là một đặc điểm tiêu lịch, nhưng một hệ thống có tính năng thời gian thực không nhất thiết phải có đáp ứng thật nhanh mà quan trọng hơn là phải có phản ứng kịp thời đối với các yêu cầu, tác động bên ngoài.

3.12. Cho biểu đồ liên quan tới tác thời điểm thực hiện một tác vụ ở thời gian thực.

Thông số ai, ri, si....

Thời gian xuất hiện ai (arrival time): Khi sự kiện xảy ra và tác vụ tơơng ứng được kích hoạt.

- Thời điểm bắt đầu thực thi ri(release time): Thời điểm sớm nhất khi việc xử lý đã sẵn sàng và có thể bắt đầu.
- Thời điểm bắt đầu thực hiện si (starting time): Là thời điểm mà tại đó tác vụ bắt đầu việc thực hiện của mình.
- Thời gian tính toán/thực thi ci (Computation time): Là khoảng thời gian cần thiết để bộ xử lý thực hiện xong nhiệm vụ của mình mà không bị ngắt.
- Thời điểm hoàn thành fi (finishing time): Là thời điểm mà tại đó tác vụ hoàn thành việc thực hiện của mình.

- Thời gian rủi ro/ xấu nhất wi(worst case time): khoảng thời gian thực hiện lâu nhất có thể xảy ra, dung sai mềm = wi-di.
- Thời điểm kết thúc di (due time-deadline): Thời điểm mà tác vụ phải hoàn thành.

B/Hệ thời gian thực cứng: là hệ mà dung sai tới hạn chót xấp xỉ bằng không. Nói cách khác phải đúng thời điểm nếu không sẽ là thảm họa. Các tiêu chí hệ thống như sau:

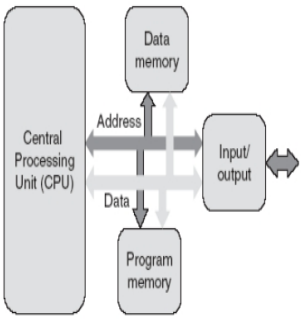
- Phải đảm bảo không để bất kì một sự kiện tới hạn (critical event) nào bị sự cố trong bất kì hoàn cảnh nào của hệ thống;
- Độ trễ đáp ứng cho sự kiện rất nhỏ (xét theo từng lớp ứng dụng)
- Các sự kiện có tính chu kì phải được đảm bảo thực hiện đúng chu kì.

Khi thiết kế hệ này cần tính để kết quả tính toán có được trước hạn chót trước khi hệ phát ra đáp ứng. **C/Hệ thời gian thực mềm**: là hệ phải hợp thời gian nhưng hạn chót có tính mềm dẻo. Như vậy hạn chót có thể có nhiều mức, hạn chót với thời gian T ước tính với trị trung bình, xác xuất đáp ứng đưa ra nằm trong các mức độ khác nhau với độ trễ trung bình và chấp nhận được. Tuy không gây ra thảm họa hệ thống nhưng phải trả giá khi độ trể hệ thống tăng tỷ lệ thuận tùy thuộc vào ứng dụng. Cần có cơ chế bù trừ để loại trừ độ trễ này.

4.1:

A.Hãy đặc tả kiến trúc CPU kiểu Von Neumman và kiến trúc CPU kiểu Harvard.

➢ Kiến trúc CPU kiểu Von Neumann



a) Kiến trúc Von Neumman

-Kiến trúc von-Neumann được nhà toán học John von-Neumann đưa ra vào năm 1945 trong một báo cáo về máy tính EDVAC.

-Hệ thống BUS địa chỉ và BUS dữ liệu, BUS điều khiển chung cho toàn bộ hệ thống, bộ nhớ chia sẻ chung cho toàn hệ thống với vùng mã lệnh (code) và dữ liệu (data) trên cùng không gian địa chỉ bộ nhớ và BUS dữ liệu không thể truyền đồng thời mã lệnh và dữ liệu cùng một thời điểm.

-Các đặc điểm của kiến trúc máy tính von-Neumann:

+Một bộ nhớ duy nhất được dùng để lưu trữ dữ liệu (data) và lệnh (instructions)

+Dữ liệu và lệnh được lưu trữ trong các phần riêng của bộ nhớ

+Bộ nhớ được đánh địa chỉ theo vùng, không phụ thuộc vào loại dữ liệu mà nó lưu trữ.

+Quá trình thực hiện các lệnh diễn ra tuần tự.

-Quá trình thực hiện một lệnh máy :

1)Đọc mã lệnh từ ROM/RAM qua BUS dữ liệu vào CPU, giả mã để xác định làm gì tiếp theo;

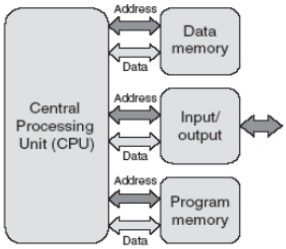
2) Đọc dữ liệu tiếp theo là một phần của lệnh (operands) nếu có qua BUS dữ liệu;

3) Thực hiện lệnh khi đã đọc hết các operands của lệnh;

4) Lưu kết quả ra RAM qua Data BUS.

Như vậy BUS dữ liệu là kênh duy nhất để trao đổi dữ liệu, do vậy ta nói BUS dữ liệu bị "bão hòa", hiệu năng tính toán bị hạn chế. Với các CPU hiện đại BUS dữ liệu được cải tiến rất nhiều, đặc biệt là giao thức BUS và đồng hồ BUS được nâng cao để cải thiện hạn chế nói trên.

➢ Kiến trúc Harvard



b) Kiến trúc Harvard

-Howard Aiken (1900-1973) khi xây dựng máy tính với các relé đã tách các bộ nhớ dữ liệu (RAM) và bộ nhớ chương trình với các bus riêng rẽ để truy cập vào bộ nhớ dữ liệu (RAM) và bộ nhớ chương trình (NVM Non Volatile Memory: ROM, FLASH) chứa phần mềm nhúng (hệ điều hành, Device drivers, ứng dụng nhúng).

-Các bus điều hành độc lập, các chỉ dẫn chương trình và dữ liệu có thể được đưa ra cùng một lúc, cải thiện tốc độ so với thiết kế với chỉ một bus.

-Phân biệt rõ ràng bộ nhớ dữ liệu và bộ nhớ chương trình, CPU có thể vừa đọc một lệnh, vừa truy cập dữ liệu từ bộ nhớ cùng lúc.

-Do các bus độc lập, CPU có khả năng tìm trước, nên với kiến trúc Harvard chương trình chạy nhanh hơn, bởi vì nó có thể thực hiện ngay lệnh tiếp theo khi vừa kết thúc lệnh trước đó.

-Tuy nhiên về kiến trúc có phần phức tạp hơn trong phần cứng, nhưng cho hiệu quả hơn cho các ứng dụng nhúng.

B.Hai kiến trúc này khác nhau ở điểm nào ? Kiến trúc nào thích hợp hơn khi chọn để thiết kế một hệ thống nhúng ?

(1)Hệ thống bus địa chỉ, bus dữ liệu và bus điều khiển chung cho toàn bộ hệ thống.

(2)Các bộ nhớ dữ liệu và bộ nhớ chương trình được tách riêng với các bus riêng rẽ để truy cập vào bộ nhớ dữ liệu và bộ nhớ chương trình chứa phần mềm nhúng.

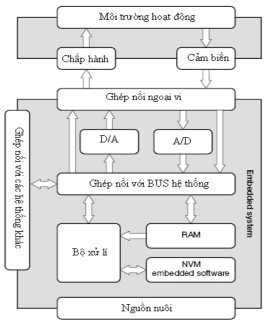
(1)Bus dữ liệu không thể truyền đồng thời mã lệnh và dữ liệu cùng một thời điểm

(2)Các bus điều hành độc lập, các chỉ dẫn chương trình và dữ liệu có thể được đưa ra cùng một lúc
(1)Có một bộ nhớ duy nhất được dùng để lưu trữ dữ liệu và lệnh
(2)Phân biệt rõ ràng bộ nhớ dữ liệu và bộ nhớ chương trình, CPU có thể vừa đọc một lệnh, vừa truy cập dữ liệu từ bộ nhớ cùng lúc
(1)Đơn giản hơn
(2)Phức tạp hơn
(1)Chạy chậm hơn
(2)Chạy nhanh hơn do các bus độc lập (CPU có thể giao tiếp đồng thời với cả bộ nhớ chương trình và dữ liệu).

Với những ưu điểm của kiến trúc Harvard, kiến trúc Harvard thích hợp hơn để thiết kế một hệ thống nhúng.

Với những ưu điểm của kiến trúc Harvard, kiến trúc Harvard thích hợp hơn để thiết kế một hệ thống nhúng.

4.2 **:Nêu (vẽ) mô hình tổng quát phần cứng của một hệ thống nhúng ?**



Hình 1.7- Mô hình tổng quát HTN-

Mô hình với các khối chức năng

B.Nêu chức năng của từng khối của mô hình đó ?

Các khối chức năng:

-Môi trường hoạt động: nơi sử dụng HTN.

-Chấp hành: là các thiết bị công nghệ.

-Cảm biến: thiết bị đặc biệt ghi nhận thông tin công nghệ (vị trí, vòng quay, tốc độ, nhiệt độ, áp suất, kích thước (cao, dài, sâu) ...).

-Ghép nối: là các thiết bị phối hợp, chuyển hóa các thông tin từ cảm biến thành tín hiệu điện để số hóa.

-Các bộ số hóa (A/D) và tương tự hóa (D/A).

-Ghép nối với các hệ thống khác: liên kết các HTN khác, mạng dữ liệu, Trung tâm điều khiển SCADA, ...

-Ghép nối BUS hệ thống

-CPU, RAM, ROM (FLASH).

4.3 **Nói Hệ thống nhúng là một hệ thống đáng tin cậy (dependable), vì các đặc tính sau đây** :

-Tin cậy (Reliability): lý tưởng là không có sự cố hỏng hóc.

HTN là một kiểu máy tính có yêu cầu về chất lượng và độ tin cậy rất cao, hoạt động được trong các môi trường khắc nghiệt về nhiệt độ (cao, hay rất thấp), độ ẩm cao, độ rung động lớn, nhiễu sóng điện từ v.v

-Khả năng duy trì (Maintainability): thời gian bảo trì nhanh chóng,

Phần lớn các hệ thống nhúng hoạt động với sự ràng buộc thời gian: yêu cầu có thời gian cho (đáp ứng) đầu ra nhanh, đúng thời điểm, trong mối tương quan với thời điểm xuất hiện của (sự kiện) đầu vào.
-Tính sẵn sàng (Availability): là kết quả của sự tin cậy và bảo trì.
-Chắc chắn (Safety): nếu có sự cố xảy ra, HTN không gây ra những tác hại khác của toàn hệ thống.
-An ninh (Security): dữ liệu của HTN được bảo mật, truy nhập phải có xác nhận (ví dụ, HTN là các thiết bị).

4.4
A.Từ cách xác định địa chỉ cho dịch vụ xử lí ngắt (Interrupt Service Routine-ISR) sau đây :
-Địa chỉ của ISR đã được định sẵn bên trong CPU ;
-Địa chỉ của ISR được chỉ định một chổ nào đó trong bộ nhớ, hoặc phải thực hiện một lệnh nhảy (JMP addr) tới địa chỉ hiện tại của ISR ;
-Thiết bị ngoại vi phải cung cấp cho CPU địa chỉ của ISR thông qua số hiệu ngắt.
Hãy chọn các khả năng định địa chỉ phù hợp với kiểu tổ chức ngắt sau:
A.Ngắt cố định là : 1 ? hay 2 ? hay 3 ?
B.Ngắt vector là : 1 ? hay 2 ? hay 3 ?

a. Ngắt cố định - 2
b. Ngắt vector - 3

B. Ngắt cứng
Là các yêu cầu ngắt CPU do các tín hiệu đưa đến từ các chân INTR và NMI. Khác với ngắt mềm, ngắt cứng không được khởi động bởi chương trình mà bởi các thành phần có trong phần cứng của hệ vi xử lý (các thiết bị ngoại vi bên trong và bên ngoài). Loại ngắt này là một cơ cấu đơn giản và hiệu quả để bộ vi xử lý phản ứng kịp thời với các sự kiện không đồng bộ xảy ra trong hệ vi xử lý.
Ví dụ minh hoạ đơn giản là hoạt động của ngắt bàn phím. Mỗi khi ấn hay nhả một phím thì ngắt bàn phím sẽ được kích hoạt. Chương trình xử lý ngắt bàn phím sẽ được khởi động bằng cách chuyển ký tự được ấn vào vùng bộ đệm của bàn phím, xếp ngay sau ký tự được ấn lần trước. Cũng như mọi ngắt khác, việc thực hiện chương trình chính sẽ được khôi phục ngay sau khi chương trình con xử lý ngắt bàn phím kết thúc.
C. Ngắt mềm Ngắt mềm là ngắt được gọi bằng một lệnh ở trong chương trình ngôn ngữ máy. Lệnh đó là INT mn (trong đó INT là mã lệnh, mn là số hiệu ngắt). Các ngắt mềm cho phép gọi trực tiếp các chương trình con phục vụ ngắt chứa trong thư viện chương trình. Nói cách khác, lệnh ngắt mềm bản chất là một lệnh gọi chương trình con đặc biệt, nó được gọi một cách chủ động bằng chương trình của người lập trình.

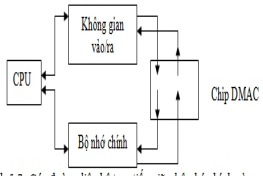
4.5A. Ngắt có che
Ngắt che được là tất cả các yêu cầu ngắt được đưa tới chân INTR của CPU. CPU có thể cấm (che)/phục vụ các ngắt này phụ thuộc vào giá trị của cờ ngắt IF (IF = 0 - cấm, IF = 1 - cho phép). Các lệnh Assembly như CLI (xoá cờ ngắt) và STI (thiết lập cờ ngắt) sẽ tác động đến cờ này. Khi một ngắt bị che, thì mặc dù được gọi, chương trình con phục vụ ngắt tương ứng cũng không được thực hiện.
B. Ngắt không che
Ngắt không che được là tất cả các yêu cầu ngắt được đưa tới chân NMI của CPU. Các ngắt này luôn được phục vụ bất chấp giá trị của cờ IF. Chương trình con phục vụ ngắt loại này thường là các chương trình có chức năng thông báo các sự kiện quan trọng nhất của hệ thống như sự cố nguồn nuôi, sự cố thời gian thực,...
C. Hãy đặc tả các bước mà CPU sẽ thực hiện khi chấp nhận xử lí một ngắt ?
CPU xử lý ngắt (cứng và mềm) thông qua các bước sau:
1. Cất thanh ghi cờ (FR) vào ngăn xếp và giảm SP đi 2 đơn vị (vì thanh ghi cờ là thanh ghi 2 byte)
2. Xoá cả hai cờ cho phép ngắt IF và cờ bẫy TF. Khi đó sẽ che các yêu cầu ngắt khác đưa đến chân INTR và huỷ bỏ chế độ chạy từng lệnh trong khi CPU thực hiện CTCPVN. Tuỳ thuộc vào chế độ ngắt mà người lập trình có thể huỷ bỏ việc che chân INTR bằng lệnh STI.

3. Cất CS hiện hành vào ngăn xếp và giảm SP đi 2 đơn vị.
4. Cất IP hiện hành vào ngăn xếp và giảm SP đi 2 đơn vị.
5. Thực hiện truy cập vào bảng vector ngắt bằng cách xác định địa chỉ vật lý của vector ngắt thông qua số hiệu ngắt tương ứng, qua đó cập nhật giá trị CS và IP mới của CTCPVN.
6. Với cặp CS:IP mới, CPU bắt đầu nhận và thực hiện các lệnh của CTCPVN.
7. Lệnh cuối cùng của CTCPVN là lệnh IRET. Lệnh này thông báo để CPU tải lại giá trị IP, CS và thanh ghi cờ từ ngăn xếp và nhờ vậy, CPU có thể thực hiện tiếp chương trình tại nơi nó bị ngắt.
D. Khi viết mã cho dịch vụ xử lí ngắt (Interrupt Service Routine-ISR), cần đặc biệt quan tâm đến gì ? Tại sao ?

- Đẩy các thanh ghi của CPU vào stack
- Thực hiện việc cấm ngắt để tránh đệ qui ngắt nếu cần hoặc cấm các ngắt khác
- Mã xử lí của ISR
- Khôi phục các thanh ghi của CPU
- Khôi phục lại khả năng chấp nhận ngắt cho các ngắt tạm cấm
- Quay về chương trình chính

4.6Hãy chọn những tiêu chí cho là đúng với cơ chế trao đổi dữ liệu kiểu truy nhập trực tiếp bộ nhớ (DMA) :
A.Cần có một vi mạch (DMAC) điều khiển qui trình DMA.
B.CPU vẫn kiểm soát BUS hệ thống.
C.CPU trao cho vi mạch DMAC quyền kiểm soát BUS hệ thống.
D.CPU vẫn thực hiện chạy chương trình nếu chương trình đó không có đòi hỏi truy nhập BUS hệ thống.
E.Trong khi xẩy ra chế độ DMA, loại CPU nào vẫn có thể tìm lệnh và thực hiện lệnh ở bộ nhớ lệnh (code) nếu không truy nhập tới bộ nhớ dữ liệu ? Tại sao ?
(DMA: Direct Memory Access)
Có 3 phương pháp điều khiển hệ thống vào ra:
-Vào ra bằng chương trình
-Vào ra bằng ngắt
-Truy cập bộ nhớ trực tiếp DMA
Nhược điểm chính của 2 phương pháp đầu là CPU tham gia trực tiếp vào trao đổi dữ liệu và việc trao đổi lượng dữ liệu nhỏ. Để khắc phục hai phương pháp trên một phương pháp mới có tên DMA sẽ sử dụng thêm một Module phần cứng có DMAC (DMA Controller). Vì vậy khi trao đổi dữ liệu không cần CPU.
->Tiêu chí A là đúng

Hình sau cho thấy có các được liên hệ trực tiếp giữa bộ nhớ chính và các ngoại vi qua chip DMAC mà không qua CPU



Hình 5.7: Các đường liên hệ trực tiếp giữa bộ nhớ chính và ngoại vi

->Tiêu chí C là đúng -> Tiêu chí B là sai
Theo như hình trên, khi chương trình yêu cầu truy nhập bus hệ thống thì nó giao tiếp thông qua DMAC. Còn khi thiết bị không có yêu cầu truy nhập bus hệ thống thì CPU vẫn thực hiện chạy chương trình -> Tiêu chí D là đúng
E cũng đúng nhưng mà e đek biết giải thích, thầy cũng chỉ bảo là : E đúng thôi ☺

4.7: **A.Thế nào là lập lịch hướng vào/ra (hướng I/O) ?**
B.Thế nào là lập lịch hướng CPU ?
C.Thế nào là lập lịch tĩnh (static/offline) ? Thế nào là lập lịch động (online) ?
D.Thế nào là lập lịch có thể chen ngang (preemptive algorithms) và lập lịch không thể chen ngang (non-preemptive algorithms) ?
Bài làm:
A.Lập lịch hướng vào/ra(I/O bound) có nghĩa tiến trình sử dụng nhiều thao tác I/O và sử dụng nhiều thời gian để đợi kết quả I/O.
B.Lập lịch hướng CPU(CPU bound) là lớp tiến trình sử dụng phần lớn thời gian của CPU để thực hiện xử lý.
C.Lập lịch tĩnh và động.
-Lập lịch tĩnh(offline): Việc lập lịch được thực hiện dựa trên các hiểu biết hoặc dự báo về các sự kiện tác vụ thực hiện trong hệ thống(thời điểm xuất hiện, thời gian thực hiện, hạn chót ước tính) và được quyết định tại thời điểm thiết kế và được áp dụng cố định trong suốt quá trình hoạt động của hệ thống, phân phối xử lý chuẩn theo thời gian, và được giám sát bởi timer gọi là time trigged(TT system).
-Lập lịch động(online): Bộ xử lý thực hiện việc lập lịch trong quá trình thực thi(run time) dựa trên cơ sở các thông tin hoạt động hiện hành của hệ thống. Sơ đồ lập lịch là không xác định trước và thay đổi động theo quá trình thực hiện, linh hoạt theo sự kiện.
D.Lập lịch chen ngang và không thể chen ngang.
--Lập lịch chen ngang(preemptive): Giải thuật được sử dụng nếu có tác vụ nào đó có thời gian thực thi quá lâu, nó có thể bị tác vụ khác ngắt hay nếu một sự kiện bên ngoài cần thời gian đáp ứng ngắn, sự kiện đó sẽ ngắt sự kiện khác. Hay nói cách khác một

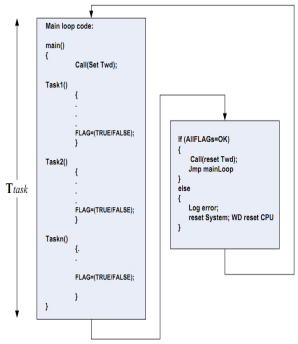
tác vụ sẽ bị ngắt bởi một tác vụ khác hay tác vụ có mức ưu tiên cao hơn ngắt khi đang xử lý, có tính đệ qui.
-Lập lịch không thể chen ngang(non-preemptive): Giải thuật giả định rằng các tác vụ sẽ thực hiện cho tới khi hoàn tất. Như vậy nếu có một tác vụ thực thi quá lâu, thì đáp ứng cho các sự kiện ngoài sẽ lâu.

4.8 Định thời (watch-dog) là gì?

Định thời là khung thời gian thực ấn định cho một xử lý phải hoàn thành.

Bộ giám sát định thời (watchdog timer) là đồng hồ thời gian cứng (dùng các bộ đếm điện tử) với các ứng dụng sau đây:
+Làm đồng hồ thời gian thực cho hệ thống.
+Khởi động/khởi động lại một sự kiện sau một thời gian đặt trước.
+Tạo khung cửa sổ thời gian cho một sự kiện.
+Phân giải khoản thời gian giữa hai sự kiện.
+Chó canh chừng. hay đồng hồ thời gian mềm (lập các giá trị đếm cho một biến chương trình), các thao tác tăng/giảm giá trị đếm thực hiện bằng lệnh máy, do đó phụ thuộc vào CPU clock (mỗi loại CPU có clock khác nhau).
a.Vai trò của định thời trong hệ thời gian thực:

Trong các hệ thời gian thực, trong đó có HTN, watchdog rất quan trọng, được sử dụng để tự động khởi động lại một ứng dụng nhúng hay thậm chí cả hệ thống về trạng thái ban đầu mà không có sự can thiệp của con người. Trong các CPU nhúng ta thấy có vài bộ đếm thời gian cứng, đặt các giá trị khác nhau cho các ứng dụng quan trọng, mà trong khung thời gian đó ứng dụng phải kết thúc hay phải đưa ra được đáp ứng, nếu không (khả năng có sự bất thường) ứng dụng sẽ được watchdog khởi động lại từ đầu.
b.Phát thảo một giải thuật xử lý có tác nhân giám sát của định thời?
Mỗi tác vụ khác biệt thực thi bởi một “main loop”, nếu kết thúc hoàn hảo, đặt cờ trạng thái lên (Flagi set=TRUE). Tất cả các main loop thực hiện tối đa trong 35 micro giây. Sau vòng cuối cùng là đoạn mã kiểm tra:
Nếu tất cả các Flags đều là TRUE, khởi động chu kì watchdog mới (50 micro giây), nếu Flags = FALSE, ghi nhận sự cố và đặt tất cả Flags = FALSE, watchdog không được khởi động lại trong thời gian 50 micro giây, đầu ra của bộ đếm watchdog sẽ kích hoạt RESET hệ thống.



Giải thuật với giám sát định thời

Đoạn mã

```
 if  (all flag are OK)
  {
Call(Reset Twd);  //Hệ hoạt động bình thường,
//đặt mới giá trị 50 micro giây co watchdog
Jmp mainloop;    //Trở về chu kì mới;
}
else  //nếu thực hiện đoạn code này có nghĩa
                        //counter sẽ vượt 50 giây ấn định     xung đầu ra sẽ RESET CPU.
   {
     Log eror;    //Record failure
     Reset System;   //WD reset CPU
}
```

4.9: Dưới đây là mô hình thời gian thực RTOS và hệ điều hành chuẩn chung cho máy tính.Sự khác nhau là...

Trong hệ điều hành thời gian thực thì nhân thời gian thực thao tác các thiết bị chứ không phải hệ điều hành. Do vậy việc cài đặt TĐKTB không phải là thành phần nhúng trong nhân, đặt trên nhân và chỉ có các TĐKTB cần thì đưa vào hệ thống là rõ ràng và hợp lý, vì các thiết bị nối trực tiếp vào hệ thống để nhân điều khiển trực tiếp, đảm bảo chi phí thời gian là ít nhất.

-Trong RTOS có một phần được gọi là bộ lập lịch, lưu vết các trạng thái của mỗi tác vụ và quyết định một tác vụ duy nhất sẽ đi vào trạng thái Running. Bộ lập lịch trong RTOS nhìn vào mức ưu tiện được gán cho mỗi tác vụ và tác vụ nào có mức ưu tiên cao nhất sẽ được thực hiện. Các tác vụ có mức ưu tiên thấp hơn sẽ đợi cho đến khi bộ xử lý được giải phóng. Nếu một tác vụ đang chạy, xuất hiện một tác vụ khác có mức ưu tiên cao hơn được kích hoạt thì RTOS sẽ dừng tác vụ đang chạy và sẽ chạy tác vụ ưu tiên cao hơn.

-TĐKTB là phần hợp nhất của hệ điều hành.

4.10 :

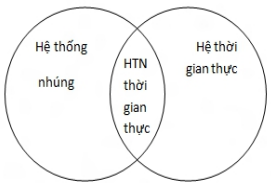
A,Có thể nói trong hệ thời gian thực, hầu hết các hoạt động xử lí đều do bộ lập biểu kiểm soát (như hình dưới). Hãy giải thích tại sao như vậy ?

B.Thế nào là một hệ thống nhúng thời gian thực ?

Bài làm:

A.Một hệ thống thời gian thực cứng phải thực hiện một tập hợp các nhiệm vụ thời gian thực đồng thời và tất cả các nhiệm vụ quan trọng đều phải đáp ứng đúng thời gian quy định của chúng. Trong khi CPU không thể thực hiện cùng một lúc 2 tác vụ và nguồn tài nguyên cũng có giới hạn. Vì vậy cần phải có một kế hoạch phân bổ và gán quy trình thực thi các tác vụ cho bộ xử lý sao cho mỗi tác vụ được thực hiện hoàn toàn. Kế hoạch phân bổ đó chính là "Lập lịch". Lập lịch chính là kế hoạch định thời cho các tác vụ.

B.HTN thời gian thực là HTN có thể được tạo thành khi có phần cứng nhúng và phần mềm hệ thống là RTOS(Hệ điều hành thời gian thực). Mối quan hệ giữa hệ thời gian thực và HTN như hình dưới đây.



Hình cho thấy rằng không phải tất cả HTN đều là HTN thời gian thực và ngược lại không phải tất cả hệ thời gian thực là HTN.

4.11**:A.Thế nào là phần mềm trung gian trong một hệ thống nhúng?**

Trong HTN, PMTG được coi như phần mềm hệ thống. PMTG giống như cầu nối giữa các phần mềm khác của phần mềm hệ thống, cung cấp các dịch vụ cho các phần mềm ứng dụng như: an ninh hệ thống, kết nối mạng, truyền thông cục bộ giữa các ứng dụng trong hệ thống, mang lại sự linh hoạt khi triển khai các ứng dụng. Với vị trí "trung gian", các PMTG làm giảm đáng kể tính phức tạp của các ứng dụng, vì các tiện ích đã có sẵn và chia sẻ ngay trong PMTG.

B.Trong các hìnbh sau a, b, c, d, hình nào sai khi đặt phần mềm trung gian vào các lớp kiến trúc phần mềm của hệ thống nhúng ?

Hình (a) sai vì PMTG được coi như là phần mềm tiện ích, tích hợp thêm vào phần mềm hệ thống.

C.Các thiết bị mạng lớp 3 như chuyển mạch lớp 3 (SW L3), định tuyến (Router), ADSL … là các thiết bị nhúng. Phần mềm mạng lớp 2 và lớp 3 là các phần mềm trung gian. Hãy chọn một mô hình trong các mô hình trên được cho là phù hợp để xây dựng các hệ thống nhúng đề cập ?

Mô hình phù hợp với hệ thống được đề cập ở đây là mô hình (d)

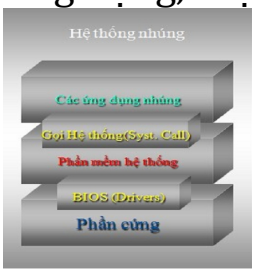
5.1:

A.Nêu mô hình phần mềm của một hệ thống nhúng ?

B.Đặc tả các lớp của mô hình phần mềm hệ thống nhúng đó ?

A.Mô hình phần mềm tổng quát

Khi đề cập tới phần mềm, tổng quát, ta có thể chia ra làm hai lớp: phần mềm hệ thống và phần mềm ứng dụng. Phần mềm ứng dụng là các phần mềm ứng dụng nhúng, các phần mềm này định nghĩa chức năng cũng như mục đích hình thành một HTN cụ thể. Phần mềm hệ thống là phần mềm có chức năng quản lí hoạt động của phần cứng, cung cấp nguồn tài nguyên phần cứng và phần mềm trung gian khác cho phần mềm ứng dụng, thực thi mã phần mềm ứng dụng sao cho hiệu quả và ổn định.



B.Đặc tả chức năng các lớp phần mềm (ứng dụng, trung gian, hệ thống, điều khiển thiết bị+BIOS)

Trình điều khiển thiết bị :

Các chức năng cơ bản của TĐKTB:

1) Khởi động phần cứng (Hardware Startup): Khởi động các phần cứng sau khi bật nguồn hay sau khi RESET máy. Phần lớn các phần cứng là các vi mạch ghép nối hay điều khiển các thiết bị và là khả trình. Cho nên bqớc này là lập trình cho các vi mạch chức năng của bo mạch, để đqa các vi mạch vào trạng thái đầu tiên: trạng thái sẵn sàng thực hiện các lệnh tiếp theo.

2) Tắt máy (Hardware Shutdown): Ngược lại với khởi động, điều khiển tắt máy sẽ cấm tuầntự các vi mạch, kết thúc kết nối với các thiết bị sao cho dữ liệu không bị xáo trộn hay bị hỏng.Cuối cùng là cắt nguồn điện nuôi máy.

3) Cấm phần cứng hoạt động (Hardware Disable): Cho phép phần mềm khác có thể tạm thời vô hiệu hóa một phần cứng khác.

4) Cho phép hoạt động (Hardware Enable): Cho phép phần mềm khác kích hoạt phần cứng.

5) Dành lấy phần cứng (Hardware Acquire): Cho phép phần mềm khác chiếm lấy phần cứng bằng cách khoá phần cứng đối với các phần mềm khác.Ví dụ sau khi HĐH giải quyết tranh chấp phần cứng, gán phần cứng cho một tiến trình tạm thời độc quyền sử dụng, các tiến trình khác phải chờ.
6) Giải phóng phần cứng (Hardware Release): Cho phép phần mềm khác giải khóa phần cứng.
7) Đọc/ghi dữ liệu (Hardware Read/write): Cho phép phần mềm khác thực hiện trao đổi dữ liệu với phần cứng.
8) Cài và tháo dỡ phần cứng (Hardware Install/Uninstall): Cho phép phần mềm khác cài mới hay tháo dỡ phần cứng.

Phần mềm trung gian

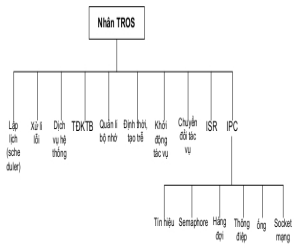
PMTG giống như cầu nối giữa các phần nềm khác của phần mềm hệ thống, cung cấp các dịch vụ cho các phần mềm ứng dụng, như: an ninh hệ thống, kết nối mạng, tryền thông cục bộ giữa các ứng dụng trong hệ thống, mang lại sự linh hoạt khi triển khai các ứng dụng.Với vị trí "trung gian", các PMTG làm giảm đáng kể tính phức tạp của các ứng, vì các tiện ích đã có sẵn và chia sẻ ngay trong PMTG.Tuy nhiên khi đưa PMTG vào hệ thống cũng là tăng thêm một lớp xếp chồng, có tác động đáng kể vào tính mở rộng, hiệu năng của các HTN, vì PMTG tác động vào tất cả các lớp phần mềm khác.

Phần mềm ứng dụng

Phần mềm ứng dụng để chạy trên HTN.HTN được ứng dụng ở nhiều lĩnh vực khác nhau, do đó các ứng dụng là rất cụ thể, và được phát triển bởi nhà chế tạo ra HTN. Ví dụ trong công nghiệp chế tạo các thiết bị cho tự động hóa, robot,… các HTN có những bài toán riêng để giải quyết, tức là phải phát triển phần mềm cho bài toán đó.

Phần mềm hệ thống

Trong phần mềm hỗ trợ đa nhiệm, và nếu chịu sự ràng buộc về thời gian xử lý, thì phải có tính thời gian thực, và phần mềm hệ thống sẽ là một hệ điều hành thời gian thực (RTOS) Chức năng chính của nhân RTOS là quản lí tài nguyên CPU, bộ nhớ, quản lí tác vụ, I/O, thực thi liên lạc giữa các tiến trình, lập lịch, mức ưu tiên, thời gian, dự đoán tình huống sự kiện.



Hình 3.10 Các chức năng nhân RTOS

5.2

Mô hình phần mềm hệ thống nhúng có thể biểu diễn bởi 3 lớp :

-Trên cùng là Lớp phần mềm ứng dụng, là tùy chọn (optional).

- Ở giữa là Lớp phần mềm hệ thống, là tùy chọn (optional).
- Dưới cùng là Lớp phần cứng, là cần thiết phải có (required).
Nói vậy là đúng hay chưa đúng, vì sao ? Hãy giải thích

Trả lời

Nói như vậy là chưa đúng.

- Lớp ứng dụng: bắt buộc . Bởi vì, Phần mềm ứng dụng là các phần mềm ứng dụng nhúng, các phần mềm này định nghĩa chức năng cũng như mục đích hình thành một HTN cụ thể. Mà đã là một hệ thống nhúng thì phải có các ứng dụng nhúng
- Lớp hệ thống: tùy chọn. Bởi vì , Phần mềm hệ thống là phần mềm có chức năng quản lí hoạt động của phần cứng, cung cấp nguồn tài nguyên phần cứng và phần mềm trung gian khác cho phần mềm ứng dụng, thực thi mã phần mềm ứng dụng sao cho hiệu quả và ổn định. Tùy thuộc vào yêu cầu vận hành của một HTN, phần mềm hệ thống có thể đơn giản đến mức không nhất thiết phải có. Mặc khác, HTN là một hệ thống tự trị nên có thể vận hành không cần lớp hệ thống
- Lớp kết hợp phần cứng (Điều khiển thiết bị) : bắt buộc. Bởi vì, lớp kết hợp phần cứng như là các khối chức năng, nó bao gồm Vi xử lý, bộ nhớ, tụ điện, điện trở, mạch tích hợp, bảng mạch in, connector, ... Tất nhiên, đây là thành phần bắt buột phải có cho tất cả các hệ thống nhúng.

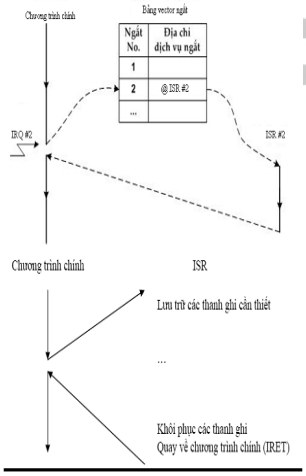
5.3

A.Phát thảo mô hình hoạt động với hệ có sử dụng ngắt kiểu vector ?

B.Đặc tả cách hoạt động của mô hình đó ?

Trả lời :

A.Mô hình



B.Đặc tả cách hoạt động của mô hình đó ?

Tiến trình ngắt:

Khi cần trao đổi thông tin, thiết bị ngoại vi gửi tín hiệu yêu cầu ngắt (Interrupt Request-IRQ) tới đầu vào INTR của CPU.CPU sẽ thực hiện nốt lệnh hiện tại và trả lời bằng tín hiệu nhận biết yêu cầu ngắt (INTA). Chương trình chính lúc này bị tạm dừng

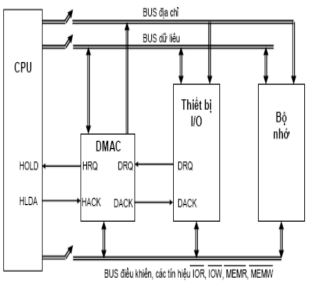
(ngắt) và CPU chuyển sang thực hiện chương trình con phục vụ ngắt (thực thi ISR của ngắt đó), tức là chqương trình con trao đổi thông tin với thiết bị ngoại vi yêu cầu ngắt. Sau khi xong công việc phục vụ ngắt, CPU quay về thực hiện tiếp chương trình chính kể từ lệnh tiếp theo sau khi bị ngắt.

Các tín hiệu yêu cầu phục vụ ngắt từ một thiết bị ngoại vi bất kỳ được gửi tới chấp nhận yêu cầu ngắt của CPU có thể thông qua một khối điều khiển ngắt.Tuỳ theo người lập trình mà yêu cầu ngắt đó có được chuyển tới CPU hay không Trong trường hợp yêu cầu ngắt được gửi tới CPU, xử lý của CPU gồm các bước sau:

Quá trình thực hiện ngắt:

- CPU hoạt động bình thqường
- Khi thiết bị vào/ra sẵn sàng chuyển số liệu sẽ gửi yêu ngắt tới CPU bằng tín hiệu IRQ tới dầu vào INTR (Interrupt Request) của CPU
- CPU thực hiện nốt lệnh đang thực hiện trqước khi trả lời chấp nhận ngắt
- CPU nhận và tìm cách xác định ngắt và trả lời thiết bị vào/ra bằng tín hiệu INTA (Interrupt Acknowledgement)
- Đẩy PSW (Program State Word) và PC (Program Counter) vào ngăn xếp
- Xoá các cờ IF (Interrupt Flag) và cờ TF (Trap Flag)
- TB vào/ra thông qua bộ điều khiển ngắt cho biết địa chỉ của chương trình con phục vụ ngắt ISR của ngắt đó.CPU nạp địa chỉ này vào PC.
- CPU nhảy đến chqương trình con ISR và thực hiện xử lý
- Chqương trình ISR sẽ đẩy các thanh sẽ bị thay đổi trong chương trình con vào ngăn xếp.
- Chqương trình ISR sẽ thực hiện việc chuyển số liệu giữa thiết bị vào/ra và bộ nhớ qua ACC của CPU.
- Sau khi chuyển số liệu xong, CPU khôi phục các thanh ghi
- Khôi phục PC và PSW từ ngăn xếp, trở về chương trình chính thực hiện tiếp nhiệm vụ trước khi có ngắt.

5.4Phát thảo mô hình hoạt động với với DMA ?Đặc tả cách hoạt động của mô hình đó ?



Hình 2.65 Nguyên lí DMA

B.Mô tả nguyên lý hoạt động

1) CPU làm việc bình thqờng
2) Khi thiết bị ngoại vi muốn chuyển số liệu trực tiếp với bộ nhớ thì gửi yêu cầu tới DMAC qua tín hiệu DRQx (DMA Reqest thứ x, mỗi DMAC có khả năng nhận 4 DRQ).
3) Bộ điều khiển DMAC chuyển yêu cầu này tới CPU qua tín hiệu HRQ: Yêu cầu CPU "tách ra " khỏi BUS hệ thống.
4) CPU thực hiện nốt chu kỳ máy đang thực hiện, treo BUS và trả lời DMAC bằng tín hiệu HLDA: Chấp nhận và đã treo BUS.
5) DMAC trả lời thiết bị vào/ra bằng tín hiệu DACKx, DMAC quản lí BUS hệ thống và phát sinh các tín hiệu địa chỉ lên BUS địa chỉ, hqớng tới bộ nhớ,
6) Phát các tín hiệu điều khiển: MEMRD/, MEMWR/, IORD/, IOWR/, để thực hiện DMA vào: đọc thiết bị/ghi bộ nhớ hay DMA ra: dọc bọ nhớ/ghi ra thiết bị.
7) Điều khiển chuyển số liệu giữa bộ nhớ và thiết bị vào/ra, chạy đồng bộ theo BUS-clock.Số liệu chuyển giữa bộ nhớ và thiết bị vào/ra thqờng là cả một khối, có độ dài tùy ý qua lập trình.
8) Khi chuyển xong số liệu DMAC đqa tín hiệu TC hay EOP (Terminal Count, End of Operation-EOP) thành tích cực để báo một quá trình DMA kết thúc (tín hiệu TC đóng vai trò một ngắt khi sử dụng để thông báo cho CPU). DMAC treo BUS, DMAC hũy HRQ.CPU hũy HLDA, CPU trở lại quản lí BUS hệ thống. Chu kì DMA hoàn tất.

5.5: Thế nào ghép nối do CPU chủ động ? Cho vài ví dụ ?
A.Có mấy kỉ thuật để triển khai ghép nối do CPU chủ động ?
B.Thế nào là ghép nối do ngoại vi chủ động ? Cho ví dụ ?
C.Có mấy kỉ thuật để triển khai ghép nối do ngoại vi chủ động ?
D. Dựa vào tiêu chí nào để lựa chọn một mô hình ghép nối cho phù hợp ? Cho ví dụ ?

A.Ghép nối CPU chủ động

Khi các ứng dụng chủ động kích hoạt các trao đổi dữ liệu với thiết bị, CPU thực hiện
Ví dụ ứng dụng: Khi kiểm soát nhiệt độ, thiết bị nhiệt luôn có số liệu để đo và xử lý. Trong
trường hợp này ghép nối với thiết bị nhiệt trở nên đơn giản.

B. Kĩ thuật để triển khai ghép nối do CPU chủ động

1.Vào/ra số liệu bằng chqơng trình không điều kiện
Đầu vào :
Lệnh thực hiện: IN [port_in] hoặc IN , [địa chỉ_ port_in]
Các bước thực hiện:
- CPU đưa ra BUS địa chỉ địa chỉ cổng port_in cho giải mã, tạo CS/ mở port_in.
- CPU đưa ra BUS đ/k tín hiệu IORD
- Số liệu từ vi mạch 3-state chuyển vào BUS dữ liệu và dqới tác động của tín hiệu IORD/ và được đưa vào ACC của CPU qua port_in 3 trạng thái.
- Thực hiện chuyển ACC vào RAM.

Đưa ra:
Lệnh thực hiện: OUT [port_out] hay OUT [port_out], AL
Các bước thực hiện:
- CPU đưa ra BUS địa chỉ địa chỉ cổng port_out cho giải mã, tạo CS/ mở port_out.
- CPU đưa dữ liệu cần ghi ra BUS dữ liệu
- CPU đưa tín hiệu IOWR/ để chốt dữ liệu, dqới tác động của tín hiệu IOWR/ số liệu được
ghi vào thanh ghi chốt, từ đó thiết bị nhận dữ liệu với tín hiệu RD/ của thiết bị.

2.Vào/ra số liệu bằng chương trình có điều kiện (handshaking-đối thoại)
Các bước thực hiện:
- CPU đưa địa chỉ port_status ra BUS địa chỉ, vào giãi mã, tạo CS0, đọc STATUS, giá trị ở bit READY tại D0. (IN port_status)
- CPU kiểm tra giá trị của READY.
- Nếu READY=0, quay lại đọc STATUS
NếuREADY=1, CPU đọc dữ liệu vào ACC (IN port_in)
- Thực hiện lệnh cất dữ liệu vào RAM

C. Thế nào là ghép nối do ngoại vi chủ động ? Cho ví dụ ?

Ghép nối do thiết bị ngoại vi chủ động là ghép nối khi thiết bị ngoại vi kích hoạt yêu cầu trao đổi dữ liệu, CPU đáp ứng.

D. Có mấy kỹ thuật để triển khai ghép nối do ngoại vi chủ động ?

Có hai kỹ thuật để triển khai ghép nối do thiết bị ngoại vi chủ động :

1. Ngắt (Interrupts)

- Thiết bị chủ động vào bất cứ lúc nào, CPU không bị ràng buộc với thiết bị, do đó chi phí thời gian như phương pháp CPU chủ động giảm đi rất nhiều.
- Có hai hình thức ngắt : ngắt cứng với thiết bị và ngắt mềm-sử dụng một nguyên lí mà hệ điều hành hỗ trợ thông qua lệnh máy tính, giúp tạo ra sự chuyển xử lý (chương trình) tạm thời đến một xử lý đột xuất.
- Ngắt mềm thực chất thực hiện một lời gọi hàm đặc biệt được kích hoạt bởi các nguồn ngắt là các sự kiện xuất hiện từ bên trong chɑơng trình và ngoại vi tích hợp trên Chip. Ví dụ như ngắt thời gian, ngắt từ thiết bị như ADC/DAC
- Ngắt cứng có thể được xem như là một lời gọi hàm đặc biệt trong đó nguồn kích hoạt là một sự kiện đến từ bên ngoài chương trình thông qua một cấu trúc phần cứng (được kết nối với thế giới bên ngoài qua các chân ngắt)
- Ngắt cứng thường hoạt động theo cơ chế dị bộ

Quá trình thực hiện ngắt:

- CPU hoạt động bình thường
- Khi thiết bị vào/ra sẵn sàng chuyển số liệu sẽ gửi yêu ngắt tới CPU bằng tín hiệu IRQ tới dầu vào INTR (Interrupt Request) của CPU
- CPU thực hiện nốt lệnh đang thực hiện trước khi trả lời chấp nhận ngắt
- CPU nhận và tìm cách xác định ngắt và trả lời thiết bị vào/ra bằng tín hiệu INTA (Interrupt Acknowledgement)
- Đẩy PSW (Program State Word) và PC (Program Counter) vào ngăn xếp
- Xoá các cờ IF (Interrupt Flag) và cờ TF (Trap Flag)
- TB vào/ra thông qua bộ điều khiển ngắt cho biết địa chỉ của chương trình con phục vụ ngắt ISR của ngắt đó. CPU nạp địa chỉ này vào PC.
- CPU nhảy đến chương trình con ISR và thực hiện xử lý
- Chương trình ISR sẽ đẩy các thanh sẽ bị thay đổi trong chɑơng trình con vào ngăn xếp.
- Chương trình ISR sẽ thực hiện việc chuyển số liệu giữa thiết bị vào/ra và bộ nhớ qua ACC của CPU.
- Sau khi chuyển số liệu xong, CPU khôi phục các thanh ghi
- Khôi phục PC và PSW từ ngăn xếp, trở về chương trình chính thực hiện tiếp nhiệm vụ trước khi có ngắt.

2. Truy nhập trực tiếp vào bộ nhớ (Direct Memory Access-DMA)

- Khi cần trao đổi khối dữ liệu giữa RAM và thiết bị, thì tác vụ này phải xảy ra nhanh nhất có thể và chỉ phụ thuộc vào khả năng của thiết bị và phải thực hiện hoàn toàn bằng cơ chế điện tử, không sử dụng tới các lệnh máy qua chương trình
- Cần có một vi mạch khả trình, có khả năng thay thế CPU
- Quá trình thực hiện DMA:

1) CPU làm việc bình thường

2) Khi thiết bị ngoại vi muốn chuyển số liệu trực tiếp với bộ nhớ thì gửi yêu cầu tới DMAC qua tín hiệu DRQx (DMA Reqest thứ x, mỗi DMAC có khả năng nhận 4 DRQ).

3) Bộ điều khiển DMAC chuyển yêu cầu này tới CPU qua tín hiệu HRQ: Yêu cầu CPU "tách ra " khỏi BUS hệ thống.

4) CPU thực hiện nốt chu kỳ máy đang thực hiện, treo BUS và trả lời DMAC bằng tín hiệu HLDA: Chấp nhận và đã treo BUS.

5) DMAC trả lời thiết bị vào/ra bằng tín hiệu DACKx, DMAC quản lí BUS hệ thống và phát sinh các tín hiệu địa chỉ lên BUS địa chỉ, hướng tới bộ nhớ,

6) Phát các tín hiệu điều khiển: MEMRD/, MEMWR/, IORD/, IOWR/, để thực hiện DMA vào: đọc thiết bị/ghi bộ nhớ hay DMA ra: dọc bọ nhớ/ghi ra thiết bị.

7) Điều khiển chuyển số liệu giữa bộ nhớ và thiết bị vào/ra, chạy đồng bộ theo BUS-clock. Số liệu chuyển giữa bộ nhớ và thiết bị vào/ra thường là cả một khối, có độ dài tùy ý qua lập trình.

8) Khi chuyển xong số liệu DMAC đơa tín hiệu TC hay EOP (Terminal Count, End of Operation-EOP) thành tích cực để báo một quá trình DMA kết thúc (tín hiệu TC đóng vai trò một ngắt khi sử dụng để thông báo cho CPU). DMAC treo BUS, DMAC hủy HRQ. CPU hủy HLDA, CPU trở lại quản lí BUS hệ thống. Chu kì DMA hoàn tất.

E. Dựa vào tiêu chí nào để lựa chọn một mô hình ghép nối cho phù hợp ? Cho ví dụ ?

Dựa vào cách thức thiết bị ngoài sẽ trao đổi dữ liệu để lựa chọn một mô hình ghép nối phù hợp.

Ví dụ: Khi kiểm soát nhiệt độ, thiết bị nhiệt luôn có số liệu để đo và xử lý. Trong trường hợp này sử dụng ghép nối CPU chủ động, ghép nối không điều kiện, điều khiển bằng chương trình …

5.6:

A.Hãy mô tả tổ chức bộ nhớ của dòng CPU kiểu Havard ?

B.Trong kiến trúc Havard, mã chương trình chứa ở RAM hay EPROM ?

C.Trong kiến trúc Havard, có thể truy nhập đồng thời vào RAM và EPROM ?

D. Trong kiến trúc Havard , nói rằng số bit cho dữ liệu và số bit cho lệnh có độ dài khác nhau, ví dụ dữ liệu 8 bit, trong khi lệnh có thể dài tới 32 bit, đúng hay sai ?

E.Sự khác nhau cơ bản của kiến trúc Von Neumann và kiến trúc Havard thể hiện ở điểm nào ?

Trả lời :

A.Hãy mô tả tổ chức bộ nhớ của dòng CPU kiểu Havard ?

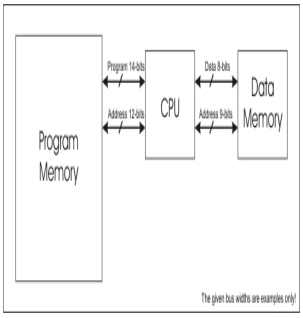
Bộ nhớ gồm hai BUS cho Bộ nhớ chương trình và BUS cho RAM dữ liệu, hoạt động độc lập, đồng thời.

Kiểu kiến trúc này được xây dựng tách riêng các bộ nhớ dữ liệu (RAM) và bộ nhớ chương trình với các BUS riêng rẽ để truy cập vào bộ nhớ dữ liệu (RAM) và bộ nhớ chương trình (NVM – Non Volatile Memory: ROM, Flash) chứa phần mềm nhúng (hệ điều hành, Device drivers, ứng dụng nhúng).

Các bus điều hành độc lập, các chỉ dẫn chương trình và dữ liệu có thể được đưa cùng một lúc, cải thiện tốc độ so với thiết kế chỉ với một BUS.

Phân biệt rõ ràng bộ nhớ dữ liệu và bộ nhớ chương trình, CPU có thể vừa đọc một lệnh, vừa truy cập dữ liệu từ bộ nhớ cùng lúc.

Do các BUS độc lập, CPU có khả năng tìm trước nên với kiến trúc Harvard, chương trình chạy nhanh hơn, vì nó có thể thực hiện ngay lệnh tiếp theo khi vừa kết thúc lệnh trước đó.



B.Trong kiến trúc Havard, mã chương trình chứa ở RAM hay EPROM ?

Trong kiến trúc Havard, mã chương trình chứa ở EPROM.

C. Trong kiến trúc Havard, có thể truy nhập đồng thời vào RAM và EPROM ?

Trong kiến trúc Havard, có thể truy cập đồng thời vào RAM và EPROM. Do kiểu kiến trúc này được xây dựng tách riêng các bộ nhớ dữ liệu (RAM) và bộ nhớ chương trình với các BUS riêng rẽ để truy cập vào bộ nhớ dữ liệu (RAM) và bộ nhớ chương trình (NVM – Non Volatile Memory: ROM, Flash) chứa phần mềm nhúng (hệ điều hành, Device drivers, ứng dụng nhúng).

D. Trong kiến trúc Havard , nói rằng số bit cho dữ liệu và số bit cho lệnh có độ dài khác nhau, ví dụ dữ liệu 8 bit, trong khi lệnh có thể dài tới 32 bit, đúng hay sai ?

Phát biểu trên Đúng.

E.Sự khác nhau cơ bản của kiến trúc Von Neumann và kiến trúc Havard thể hiện ở điểm nào ?

- Havard: Hệ thống BUS cho bộ nhớ, thực hiện lệnh đồng thời trên cả bộ nhớ chương trình và bộ nhớ dữ liệu. (phần a)
- Von Neumman: Thực hiện tuần tự các lệnh, hay có pipeline nhưng vẫn là tuần tự. Không thể cùng lúc thao tác lệnh và dữ liệu trên bộ nhớ.

Theo kiểu kiến trúc này thì, hệ thống BUS địa chỉ và BUS dữ liệu, BUS điều khiển chung cho toàn bộ hệ thống. Bộ nhớ chia sẻ chung cho toàn bộ hệ thống với vùng mã lệnh (code) và dữ liệu trên cùng không gian địa chỉ bộ nhớ và BUS dữ liệu không thể truyền đồng thời mã lệnh và dữ liệu cùng 1 thời điểm.

5.7: Mô hình Module ghép nối được đưa ra như sau :
A.Hãy nêu đặc tả của các khối chức năng bên trong Module ?
B.Phương thức Module hoạt động ?
C. Cho ví dụ về một hay vài vi mạch tích hợp loại này ?
D. Thế nào là một vi mạch ghép nối khả trình ?
E. Thế nào gọi là từ điều khiển (Control Word), chức năng làm gì ?
Trả lời
A.Hãy nêu đặc tả của các khối chức năng bên trong Module ?
- Bộ đệm dữ liệu: nơi lưu trữ dữ liệu trung gian giữa máy tính và thiết bị ngoại vi, đặt bên trong thiết bị ngoại vi.
- Khối logic điều khiển: điều khiển hoạt động của thiết bị ngoại vi theo tín hiệu từ Module I/O gởi tới thiết bị.
- Module ghép nối I/O có chức năng là Nối ghép thiết bị ngoại vi với bus của máy tính.
- Điều khiển và định thời
- Trao đổi thông tin với CPU
- Trao đổi thông tin với thiết bị ngoại vi
- Đệm giữa máy tính với thiết bị ngoại vi
- Phát hiện lỗi của các thiết bị ngoại vi.
- Cấu trúc chung:
- Thanh ghi đệm dữ liệu: đệm dữ liệu trong quá trình trao đổi.
- Cổng nối ghép vào ra: kết nối thiết bị ngoại vi, mỗi cổng có địa chỉ xác định và chuẩn kết nối riêng phụ thuộc sơ đồ chân.
- Thanh ghi trạng thái/điều khiển: lưu trữ thông tin trạng thái cho các cổng vào ra.
- Khối logic điều khiển: điều khiển Module vào ra.

B.Phương thức Module hoạt động ?
Đây là một Module thông minh, phải lập trình chế độ hoạt động.
Phương thức module hoạt động là nhận lệnh từ BUS địa chỉ và điều khiển của CPU, giải mã, logic điều khiển, sau đó đưa tới các cổng ghép điện tử rồi tới thiết bị, hoặc đưa tới các chốt trạng thái thiết bị, đệm dữ liệu đưa tới BUS dữ liệu.
-Nguyên tắc chung:
+Sử dụng lệnh vào ra trong chương trình để trao đổi dữ liệu vớicổng vào ra.
+Khi CPU thực hiện chương trình gặp lệnh vào ra thì CPU điều khiển trao đổi dữ liệu với cổng vào ra. Hoạt động vào ra bằng chương trình
+CPU gặp lệnh trao đổi vào ra, yêu cầu thao tác vào ra
+Module vào ra thao tác vào ra
+Module vào ra thiết lập các bit trạng thái(State)
+CPU kiểm tra các bit trạng thái:
+Nếu chưa sẵn sàng thì quay lại kiểm tra lại

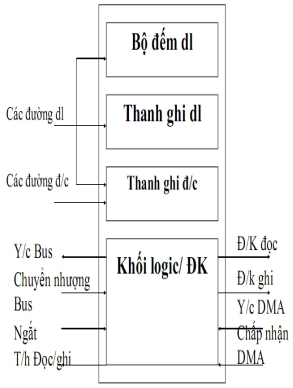
+Nếu sẵn sàng thì chuyển sang trao đổi dữ liệu với Module vào ra.
-Các phương pháp nối ghép
+Sử dụng nhiều đường yêu cầu ngắt.
+Kiểm tra vòng bằng phần mềm (Polling)
+Kiểm tra vòng bằng phần cứng
+Sử dụng bộ điều khiển ngắt.
+Nhiều yêu cầu ngắt đồng thời
+CPU sử dụng nhiều đường yêu cầu ngắt. Nạp vào thanh ghi yêu cầu ngắt.+Hạn chế số lượng Module vào ra
+Các đường ngắt được qui định mức ưu tiên.

C. Cho ví dụ về một hay vài vi mạch tích hợp loại này ?

Ví dụ: PIO 8255, DMA 8237, PIC 8259, UART 8251, 16550, …

Một ví dụ cụ tỉ: DMA : DMA sử dụng thêm một Module phần cứng có DMAC (DMA Controller). Vì vậy khi trao đổi dữ liệu không cần CPU.

- Các thành phần của DMAC



- Thanh ghi dữ liệu: chứa dữ liệu trao đổi.
- Thanh ghi địa chỉ: chứa địa chỉ của ngăn nhớ dữ liệu
- Bộ đếm dữ liệu: chứa số từ dữ liệu cần trao đổi
- Khối logic điều khiển: điều khiển hoạt động của DMAC

-Hoạt động của DMA

- Khi cần vào ra dữ liệu thì CPU nhờ DMAC tiến hành vào ra dữ liệu với thông tin cho biết như sau:
- Địa chỉ thiết bị vào ra.
- Địa chỉ đầu của mảng nhớ chứa dữ liệu và DMAC nạp thanh ghi địa chỉ
- Số từ dữ liệu cần truyền và DMAC nạp vào bộ đếm dữ liệu.
- CPU sẽ đi thực hiện việc khác
- DMAC điều khiển việc trao đổi dữ liệu sau khi truyền một từ dữ liệu thì nội dung thanh ghi địa chỉ tăng lên và nội dung bộ đếm dữ liệu giảm xuống một đơn vị.
- Khi bộ đếm bằng dữ liệu bằng 0, DMAC gởi tín hiệu ngắt CPU để báo kết thúc DMA.

D. Thế nào là một vi mạch ghép nối khả trình ?

Vi mạch ghép nối khả trình là vi mạch phải lập trình chế độ hoạt động trước khi kích hoạt.

E. Thế nào gọi là từ điều khiển (Control Word), chức năng làm gì ?

- Từ điểu khiển là 1 loại lệnh cần nạp vào I/O module.
- Lệnh sẽ được I/O module giải mã để thực hiện đúng chế độ lập trình cho nó.
- Chức năng của từ điều khiển: điều khiển hoạt động của thiết bị ngoại vi theo tín hiệu từ module I/O gửi tới thiết bị.

POPPIN KHIEM®

BAILARINES CALLEJEROS

5.8: **Hãy thiết kế một module ROM với các dữ kiện sau đây:**

- Dung lượng của module là 32 KB
- Sử dụng chip ROM loại 2732, dung lượng 4KB/chip
- Giải địa chỉ từ FFFF-8000.
- Hãy vẽ sơ đồ thiết kế và giải thích nguyên lí hoạt động.

Bài làm : các bước cần thực hiện

Dung lượng module ROM cần thiết kế là 32KB. Chíp 2732 dung lượng 4KB

⇨ Số lượng chíp cần dùng là 32/4=8 chíp

Chip ROM dung lượng 4KB/ chip hay 4K x 8 bit (4K địa chỉ, mỗi địa chỉ có độ dài là 8 bit=1 byte)

Để truy cập 4K địa chỉ cần 12 đường địa chỉ (do 2^{12}= 4069), từ A11 -> A0

Để truy cập vào mỗi chíp trong 8 chíp cần 3 đường địa chỉ : A14, A13, A12

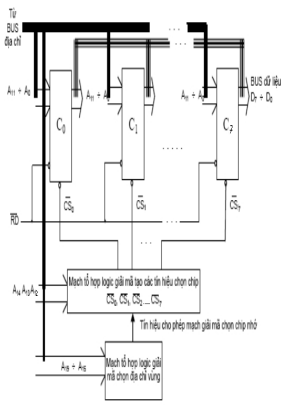
Để xác định vùng địa chỉ cần 5 đường địa chỉ : A19-> A15

Địa chỉ đầu là 8000 hex tương ứng 1000 0000 0000 0000

Tính không gian địa chỉ

	A19 A18 A17 A16	A15 A14 A13 A12	A11 A10 A9 A8	A7 A6 A5 A4	A3 A2 A1 A0	Địa chỉ đầu/địa chỉ cuối (HEX) ?
Chip 0	0 0 0 0 0 0 0 0	1 0 0 0 1 0 0 0	0 0 0 0 1 1 1 1	0 0 0 0 1 1 1 1	0 0 0 0 1 1 1 1	8000 – 8FFF
Chip 1	0 0 0 0 0 0 0 0	1 0 0 1 1 0 0 1	0 0 0 0 1 1 1 1	0 0 0 0 1 1 1 1	0 0 0 0 1 1 1 1	9000- 9FFF
Chip 2	0 0 0 0 0 0 0 0	1 0 1 0 1 0 1 0	0 0 0 0 1 1 1 1	0 0 0 0 1 1 1 1	0 0 0 0 1 1 1 1	A000- AFFF
Chip 3	0 0 0 0 0 0 0 0	1 0 1 1 1 0 1 1	0 0 0 0 1 1 1 1	0 0 0 0 1 1 1 1	0 0 0 0 1 1 1 1	B000- BFFF
Chip 4	0 0 0 0 0 0 0 0	1 1 0 0 1 1 0 0	0 0 0 0 1 1 1 1	0 0 0 0 1 1 1 1	0 0 0 0 1 1 1 1	C000- CFFF
Chip 5	0 0 0 0 0 0 0 0	1 1 0 1 1 1 0 1	0 0 0 0 1 1 1 1	0 0 0 0 1 1 1 1	0 0 0 0 1 1 1 1	D000- DFFF
Chip 6	0 0 0 0 0 0 0 0	1 1 1 0 1 1 1 0	0 0 0 0 1 1 1 1	0 0 0 0 1 1 1 1	0 0 0 0 1 1 1 1	E000- EFFF
Chip 7	0 0 0 0 0 0 0 0	1 1 1 1 1 1 1 1	0 0 0 0 1 1 1 1	0 0 0 0 1 1 1 1	0 0 0 0 1 1 1 1	F000- FFFF

Sơ đồ thiết kế chi tiết :



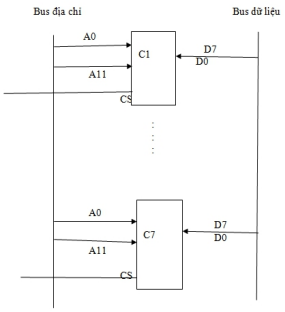
Giải mã thứ nhất có 5 đầu vào từ A19-A15 và 32 đâu ra chọn vùng địa chỉ. Với giá trị: 00001 , chọn vùng địa chỉ yêu cầu (8000-FFFF)hex = 32 K, sẽ có đầu ra cho phép giải mã thứ 2 hoạt động. Giải mã thứ hai dùng A12- A13-A14 tạo ra 8 CS/ chọn 8 chip, trong khi A11-A0 chọn ô nhớ trong mỗi chip

Chọn vùng A19.A18.A17.A16.A15	Chọn chip A14 A13 A12	Chip được chọn th A14 A13 và A12	A11....A0
0 0 0 0 1	0 0 0	C0(4K byte thứ 1)	
0 0 0 0 1	0 0 1	C1(4K byte thứ 1)	
0 0 0 0 1	0 1 0	C2(4K byte thứ 1)	
0 0 0 0 1	0 1 1	C3(4K byte thứ 1)	
0 0 0 0 1	1 0 0	C4(4K byte thứ 1)	
0 0 0 0 1	1 0 1	C5(4K byte thứ 1)	
0 0 0 0 1	1 1 0	C6(4K byte thứ 1)	
0 0 0 0 1	1 1 1	C7(4K byte thứ 1)	

5.9: **Cho trước một bảng các giá trị để thiết kế bộ nhớ RAM với chip loại 4KB/chip sau đây:**

	Địa chỉ đầu/địa chỉ cuối (HEX) ?
Chip 0	F8000 - F8FFF

Chip 1	F9000 - F9FFF
Chip 2	FA000- FAFFF
Chip 3	FB000 - FBFFF
Chip 4	FC000 - FCFFF
Chip 5	FD000 - FDFFF
Chip 6	FE000 - FEFFF
Chip 7	FF000 - FFFFF



POPPIN KHIEM®

BAILARINES CALLEJEROS

5.10:

Cho một thiết bị ngoại vi có khả năng ghép nối với hệ vi xử lí với các thông số sau đây:

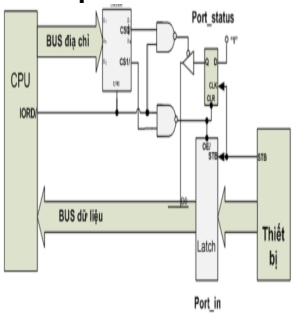
- ***Trao đổi dữ liệu vào hệ vi xử lí mỗi lần 8 bit,***
- ***Có thông báo cho CPU biết qua tín hiệu Strobe (STB) rằng đã có sẵn dữ liệu để CPU đọc vào :***

CPU cung cấp hai cổng với địa chỉ như sau:
300 (hex): cổng để thiết bị thông báo đã có dữ liệu sẵn sàng để CPU đọc vào;
301(hex): là địa chỉ cổng ghép nối thiết bị đặt dữ liệu vào đó và CPU sẽ đọc dữ liệu vào.

a.Hãy chọn giải pháp thiết kế, lên sơ đồ thiết kế ghép nối.
b.Phát thảo lưu đồ trình điều khiển ghép nối đó.

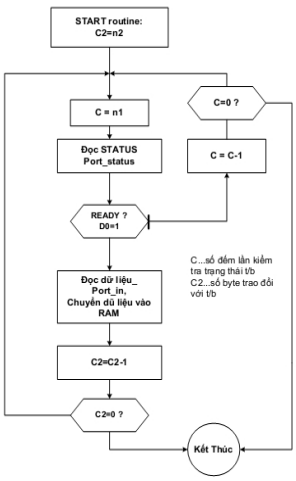
Trả lời:

Có thông báo cho CPU biết qua tín hiệu Strobe (STB) rằng đã có sẵn dữ liệu để CPU đọc vào nên Vào/ra số liệu có điều kiện, điều khiển bằng chương trình.và là đọc vào có đối thoại.
Về mặt thực hiện, ở đây cần ít nhất 2 cổng: 1 cổng để đọc trạng thái thiết bị, cổng kia để
trao đổi dữ liệu vào hay ra. Trạng thái thông thqờng ở mức dộ đơn giản chỉ cần 1 bit để thể hiện,
ví dụ STATUS=1, thiết bị sẵn sàng , và ngqợc lại.



- Cần CS0/ để đọc trạng thái READY của thiết bị qua port_status, giả định nối vào D0 của BUS dữ liệu.
- Cần CS1/ để đọc dữ liệu qua port_in, hợp thành từ Flip/flop, cổng 3-state, nối vado bit D0 của BUS dữ liệu.

Các bước thực hiện:
- CPU đqa địa chỉ port_status ra BUS địa chỉ, vào giãi mã, tạo CS0, đọc STATUS, giá trị ở bit READY tại D0. (IN port_status)
- CPU kiểm tra giá trị của READY.
- Nếu READY=0, quay lại đọc STATUS
  NếuREADY=1, CPU đọc dữ liệu vào ACC (IN port_in)
- Thực hiện lệnh cất dữ liệu vào RAM
- Lưu đồ điều khiển:



5.11:

Cho một thiết bị ngoại vi có khả năng ghép nối với hệ vi xử lí với các thông số sau đây:

-Nhận dữ liệu từ hệ vi xử lí mỗi lần 8 bit,

-Có thông báo cho CPU biết qua tín hiệu Strobe (STB) rằng khi nào thiết bị sẵn sàng nhận dữ liệu, để CPU gởi dữ liệu ra cho thiết bị:

CPU cung cấp hai cổng với địa chỉ như sau:

300 (hex): cổng để thiết bị thông báo sẵn sàng nhận dữ liệu từ CPU;

301(hex): là địa chỉ cổng để CPU gởi dữ liệu ra cho thiết bị.

a. Hãy chọn giải pháp thiết kế, lên sơ đồ thiết kế ghép nối.

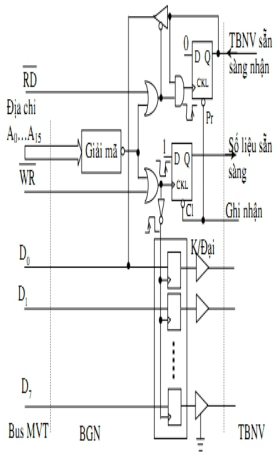
b. Phát thảo lưu đồ trình điều khiển ghép nối đó.

Trả lời

Dựa theo đề bài : thiết bị ngoại vi nhận dữ liệu từ vi xử lý và có thông báo cho CPU biết khi nào thiết bị sẵn sàng để nhận dữ liệu (tức là có điều kiện) để CPU gửi dữ liệu cho thiết bị

⇨ Đưa ra có đối thoại

Thiết kế mạch ghép nối đưa tín hiệu ra có đối thoại :



Bộ VXL kiểm tra trạng thái của thiết bị ngoại vi. Nếu thấy thiết bị ngoại vi sẵn sàng nhận thì gửi tín hiệu ghi cùng với địa chỉ tương ứng để tạo xung chọn mạch.
Sườn lên của xung này nạp số liệu từ BUS vào thanh ghi, sườn xuống đưa tín hiệu số liệu sẵn sàng lên "1" để báo cho thiết bị ngoại vi. Sau khi nhận số liệu, thiết bị ngoại vi sinh ra tín hiệu ghi nhận để xoá tín hiệu số liệu sẵn sàng và thiết lập tín hiệu thiết bị ngoại vi sẵn sàng. Lúc đó bộ VXL có thể gửi tín hiệu tiếp sau.

POPPIN KHIEM®

BAILARINES CALLEJEROS

5.12 Đây là mô hình nguyên lý hoạt động của bộ định thời.

a.Giải thích cách hoạt động của mô hình nguyên lý hoạt động bộ định thời

Watchdog sẽ thực hiện một tái khởi động (reset) hệ thống hay một hành động (xử lý), hay kích hoạt một xử lý hiệu chỉnh nếu chương trình chính bị lỗi, không tiến triển được do một điều kiện nào đó không thể đạt được trong khung thời gian dự tính (chương trình bị treo). Trong các hệ thời gian thực, trong đó có HTN, watchdog rất quan trọng, được sử dụng để tự động khởi động lại một ứng dụng nhúng hay thậm chí cả hệ thống về trạng thái ban đầu mà không có sự can thiệp của con người. Trong các CPU nhúng ta thấy có vài bộ đếm thời gian (specialized timers) cứng, đặt các giá trị khác nhau cho các ứng dụng quan trọng, mà trong khung thời gian đó ứng dụng phải kết thúc hay phải đưa ra được đáp ứng, nếu không (khả năng có sự bất thường) ứng dụng sẽ được chó canh chừng khởi động lại từ đầu. Các hệ có cài trình gởi rối sẽ ghi lại vào bộ lưu trữ đặc biệt để hỗ trợ khác phục sự cố.

B.Theo đề bài:

Ttask= 40µs

Tdeadline = 50 µs.

Twd= 55µs.

a.Khi vòng lặp kết thúc bình thường, watchdog timer sẽ được nạp lại giá trị 55µs cho một chu kì mới.

b.Khi vòng lặp kết thúc không bình thường Tdeadline>55µs ,đầu ra của watchdog timer sẽ kích hoạt RESET,khởi động lại hệ thống.

c.Phác thảo giải thuật logic cho phần mềm:

Mỗi tác vụ khac biệt được thực thi bởi một "main loop",nếu kết thúc hoàn hảo,đặt cờ trạng thái lên (Flagi set= TRUE).Tất cả các main loop thực hiện tối đa trong 50µs.Sau vòng cuối cùng là đoạn mã kiểm tra.

Nếu tất cả các Flags đều là True,khởi động chu kì watchdog mới(55µs),nếu Flags= FALSE,ghi nhận sự cố và đặt tất cả Flags=FALSE,watchdog không được khởi động lại(kick the dog) trong thời gian 55µs, đầu ra của bộ đếm watchdog sẽ kích hoạt RESET hệ thống.

Đoạn mã

```
if (all flag are OK)
{
Call(Reset Twd); //Hệ hoạt động bình thqờng,
//đặt mới giá trị 55 micro giây co watchdog
Jmp mainloop; //Trở về chu kì mới; }
```

else //nếu thực hiện đoạn code này có nghĩa
//counter sẽ vượt 55 giây ấn định xung đầu ra sẽ RESET CPU.
{
Log eror; //Record failure
Reset System; //WD reset CPU
}

5.13 Dưới đây là giải thuật với định thời watchdog

A. Twd>Ttask => đúng

Ttask là thời gian thực hiện các tác vụ trong vòng lặp.
Twd là thời gian được nạp cho 1 chu kì,nếu vòng lặp kết thúc với thời gian lớn hơn Twd thì đầu ra của watchdog timer sẽ kích hoạt RESET,khởi động lại hệ thống.

D.Một hard deadline nên được đáp ứng.

Nếu bất kì hard deadline nào bị bỏ lỡ thì hệ thống sẽ không đúng.
Yêu cầu một phương pháp để đảm bảo thời hạn được đáp ứng.
Hệ thống thời gian thực cứng là một hệ thống trong đó tất cả các tới hạn là khó.
ví dụ :kiểm soát nhà máy điện hạt nhân,kiểm soát chuyến bay.
Cách sử dụng:
Khi một máy tính là một phần của một hệ thống thời gian thực cứng, tất cả các phần mềm đang chạy trên nó có thể được điều chỉnh để đáp ứng thời hạn tất cả các hệ thống kiểm soát.

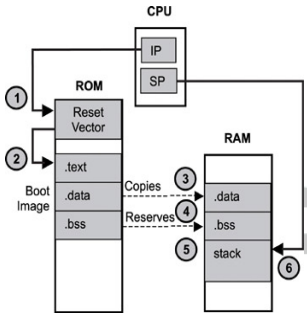
E.Một soft deadline có thể được bỏ qua.

hệ thống thời gian thực mềm là hệ thống thời gian thực trong đó một số thời hạn được coi nhẹ hơn.
ví dụ: thiết bị chuyển mạch điện thoại,ứng dụng đa phương tiện.

5.14 :Nêu kịch bản khởi động chạy phần mềm HTN từ ROM, RAM cho dữ liệu

Một số HTN có giới hạn bộ nhớ, do đó hệ sẽ khởi động trực tiếp từ ROM. Trong trường hợp này sẽ không có quá trình copy mã lệnh vào RAM để chạy. Tuy nhiên không gian cho dữ liệu vẫn phải xác định ở RAM (nhớ khi lập trình).

Có 2 thanh ghi cơ bản là IP (Instruction register)-thanh ghi lệnh, trỏ vào lệnh tiếp theo sẽ thực hiện (.text) và SP (Stack Pointer), trỏ vào địa chỉ tiếp theo trong ngăn xếp. Ngôn ngữ C sử dụng ngăn xếp để truyền các thông số khi kích hoạt một hàm. Vùng ngăn xếp phải ở RAM và SP phải trỏ vào đó khi khởi động CPU.
Qui trình khởi động như sau:
1) Thanh ghi IP được thiết kế cứng để thực hiện lệnh đầu tiên trong bộ nhớ, đó là reset vector.
2) Reset vector nhảy tới lệnh đầu tiên của phần .text của mã boot (tức boot image), .text thường trú trong ROM; CPU dùng IP để thực hiện .text, khởi động bộ nhớ, kể cả RAM.
3) Phần .data của boot image được copy vào RAM để có thể đọc/ghi.
4) Xác lập .bss trong RAM.
5) Xác lập ngăn xếp .stack trong RAM, khởi động SP trỏ vào địa chỉ đầu của stack.
6) Hoàn tất khâu khởi động, CPU tiếp tục thực hiện các lệnh trong .text cho tới khi hệ thống hoặc shutdown hay RESET.
Lưu ý ở đây là các lệnh đầu ở bước 1) không ở định dạng chuẩn ELF mà đơn giản là mã máy nhị phân sẵn sàng chạy. Nhưng boot image ở định dạng ELF (viết ra từ công cụ phát triển) nhưng không có program header và header table, do đó lưu ý viết mã thực thi sao cho có thể khởi tạo các phần .datta, .bss, .stack trong RAM. (ví dụ nếu dùng hợp ngữ thì gán nhãn, …).



POPPIN KHIEM®

BAILARINES CALLEJEROS

5.15 :Nêu kịch bản khởi động chạy phần mềm ở RAM sau khi mã đã copy từ ROM vào RAM

Ở kịch bản này, boot loader sẽ chuyển một chương trình nhỏ từ ROM vào RAM và kích hoạt nó chạy. Thường mã chương trình hệ thống trong ROM rất lớn mà được ghi kiểu nén để nạp vừa ROM, nên boot loader phải giải nén trước khi khởi động phần mã này và nó cần không gian nhớ ở RAM để thực hiện.

1) --- đến 6) giống như ở trên. Môi trường làm việc cho loader được khởi động ở RAM (3, 4, 5).

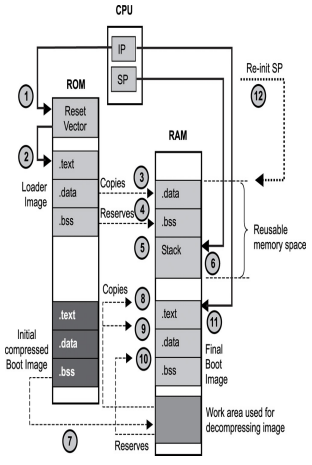
7) Loader copy phần mã nén của image vào RAM.

8) –> 10) Copy các phần mã đã giải nén vào các vùng làm việc tạm trong RAM (8, 9, 10) . Hoàn tất giải nén image. Image trong RAM ở hình là đoạn Final Boot image.

10) …

11) Loader chuyển điều khiển cho image bằng một lệnh JMP vào .text (nạp cho IP địa chỉ này trước khi JMP tới đó).

12) Vùng RAM mà loader chiếm khi được copy từ ROM là tái sử dụng, SP được tái khởi động để trỏ vào đó và được dùng như ngăn xếp cho một chương trình mới nào đó. Vung RAM để giải nén giải phóng để sử dụng khác. Hệ thống đi vào hoạt động



5.16:Nêu kịch bản khởi động chạy phần mềm ở RAM sau khi tải xuống từ hệ phát triển (đang phát triển hệ thống).

Chạy từ RAM sau khi tải xuống từ hệ phát triển (đang phát triển hệ thống)

Là kịch bản trong quá trình phát triển. Môi trường phát triển có một PC hổ trợ. Sử dụng để phát triển các phần mềm ứng dụng cho hệ thống nhúng. Phần mềm phát triển nằm trên PC và sẽ tải xuống hệ đích để chạy thử hay nạp vào hệ đích ở pha cuối cùng. Trong ROM có một chương trình gọi là Debug Agent đóng vai trò kép như một loader như ở các kịch bản trên.

1) …6) giống như trước

7) Tải ứng dụng từ PC vào hệ đích.

8) Kiểm tra sự hợp nhất của phần mềm tải xuống.

9) Giải nén ứng dụng nếu cần.

10) … 12) chương trình debug tái định vị ứng dụng vào vị trí trong RAM (10, 11, 12).

13) Debug chuyển điều khiển cho image đã tải xuống. Hệ chạy