**Lý thuyết Kiến Trúc Máy Tính**

**Màu xanh: Đã học qua 1 lượt khá kỹ**

**Màu vàng : Những cụm từ cần chú ý**

**Màu đỏ : Chú ý cấp độ cao hơn**

1. **Trình bày các thế hệ máy tính theo sự phát triển của công nghệ**

**Bài làm**

* 1. **Thế hệ thứ 1 (1944-1959):**

**◼ Sử dụng bóng đèn điện tử làm linh kiện chính**

**◼ Mật độ tích hợp linh kiện: 1000 linh kiện/ foot^3 (1 foot= 30.48 cm)**

**◼ Ví dụ: siêu máy tính ENIAC ( Electronic Numerical Integrator and Computer)**

* 1. **Thế hệ thứ 2(1960-1964):**

**◼ Sử dụng transistors làm linh kiện chính**

**◼ ~ 100,000 linh kiện/ foot^3**

**◼ Ví dụ: UNIVAC 1107, UNIVAC III, IBM 7070, 7080, 7090,…**

* 1. **Thế hệ thứ 3 (1964-1975):**

**◼ Sử dụng mạch tích hợp (IC) làm linh kiện chính**

**◼ ~ 10 triệu linh kiện/ foot^3**

**◼ Ví dụ: UNIVAC 9000 series, IBM System 360, System 3, System 7**

* 1. **Thế hệ 4(1975-1989):**

**◼ Sử dụng LSI – Large Scale Integrated Circuit làm linh kiện chính**

**◼ ~ 1 tỷ linh kiện / foot^3**

**◼ Ví dụ: IBM System 3090, IBM RISC 6000, IBM RT, Cray 2 XMP**

* 1. **Thế hệ thứ 5 (1990- nay):**

**◼ Sử dụng VLSI – Very Large Scale Integrated Circuit**

**◼ Mật độ tích hợp linh kiện rất cao**

**◼ Hiệu năng xử lý rất cao**

**◼ Hỗ trợ xử lý song song**

**◼ Tích hợp khả năng xử lý âm thanh và hình ảnh.**

**VD : máy tính sử dụng CPU Intel Pentium II, III, IV**

1. Trình bày khái niệm Kiến trúc máy tính và các thành phần của Kiến trúc máy tính

**Bài làm**

**- Kiến trúc máy tính (Computer architecture) là một khoa học về lựa chọn và kết nối các thành phần phần cứng của máy tính nhằm đạt được các yêu cầu:**

**+ Hiệu năng / tốc độ (performance): nhanh** 🡪 **tốt**

**+ Chức năng (functionality): nhiều tính năng** 🡪 **tốt**

**+ Giá thành (cost): hợp lý** 🡪 **tốt**

**- Các thành phần của Kiến trúc máy tính :**

**(3 ý)**

1. Trình bày các thành phần cơ bản của **tổ chức máy tính ( hệ thống máy tính).**

**Bài làm**

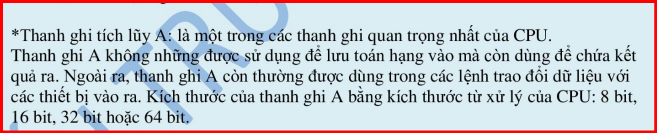
* **Bộ xử lý trung tâm (Central Processing Unit - CPU):**
  + **CPU bao gồm:**
    - **Bộ điều khiển (Control Unit - CU)**
    - **Bộ tính toán số học và logic (Arithmetic and Logic Unit - ALU)**
    - **Các thanh ghi (Registers)**
* **Bộ nhớ trong (Internal Memory):**
  + **Bộ nhớ trong bao gồm:**
    - **ROM (Read Only Memory):** 
      * **Lưu trữ lệnh và dữ liệu của hệ thống**
      * **Thông tin trong ROM vẫn tồn tại khi mất nguồn nuôi**
    - **RAM (Random Access Memory)**
      * **Lưu trữ lệnh và dữ liệu của hệ thống và người dùng**
      * **Thông tin trong RAM sẽ mất khi mất nguồn nuôi**
* **Tổ chức vào ra: Các giao diện ghép nối với các thiết bị ngoại vi (Peripheral devices)**
  + **Thiết bị vào (Input devices): nhập dự liệu và điều khiển**
    - **Bàn phím (Keyboard)**
    - **Chuột (Mice)**
    - **Ổ đĩa (Disk drives)**
    - **Máy quét (Scanner)**
  + **Các thiết bị ra (Output devices): xuất dữ liệu**
    - **Màn hình (Monitor/screen)**
    - **Máy in (Printer)**
    - **Máy vẽ (Plotter)**
    - **Ổ đĩa (Disk drives)**

1. Trình bày cấu trúc của **bộ xử lý trung tâm**(CPU) (**chú ý trình bày chức năng các thanh ghi điển hình**)

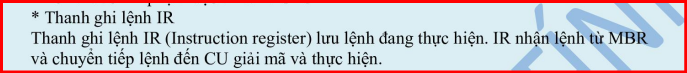
**Bộ xử lý trung tâm (Central Processing Unit - CPU):**

* 1. **CPU bao gồm:**
     1. **Bộ điều khiển (Control Unit - CU)**
     2. **Bộ tính toán số học và logic (Arithmetic and Logic Unit - ALU)**
     3. **Các thanh ghi (Registers) :**

**- Thanh ghi tích luỹ A (Accummulator)**

****

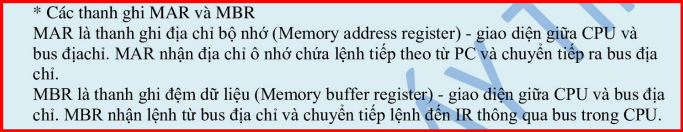
**- Thanh ghi lệnh IR (Instruction Register)**

****

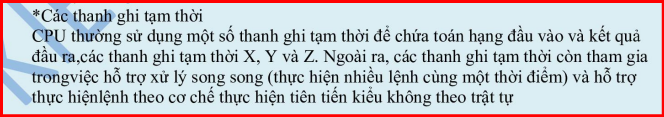
**- Thanh ghi PC : là thanh ghi chứa địa chỉ của lệnh tiếp theo.**

**- Thanh ghi địa chỉ bộ nhớ MAR (Memory Address Register)**

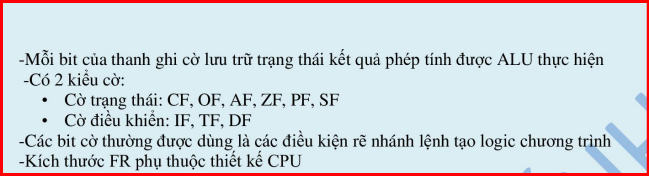
**- Thanh ghi đệm dữ liệu MBR (Memory Buffer Register)**

****

**- Các thanh ghi tạm thời Y và Z**

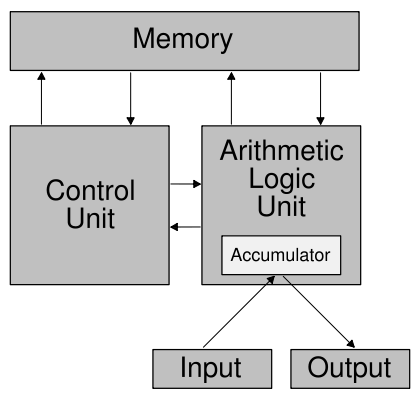
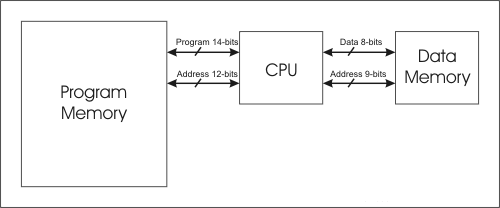
****

**- Thanh ghi cờ FR (Flag Register)**

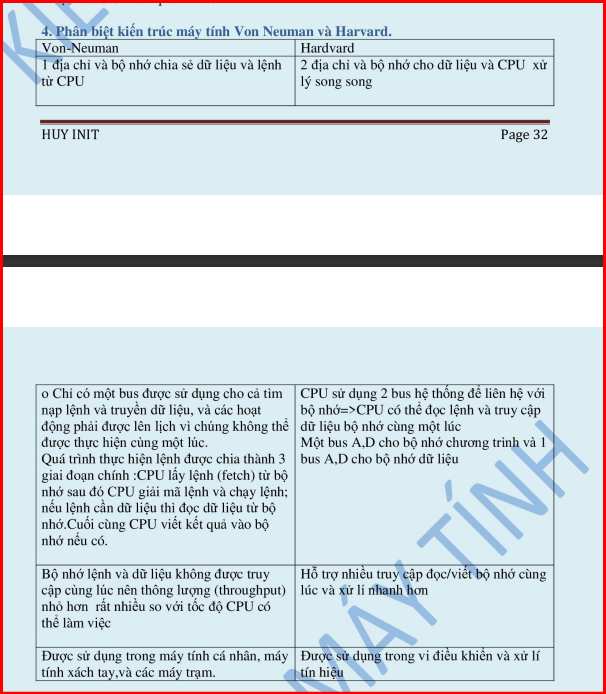
****

5. Phân biệt kiến trúc máy tính Von Neuman và Harvard.

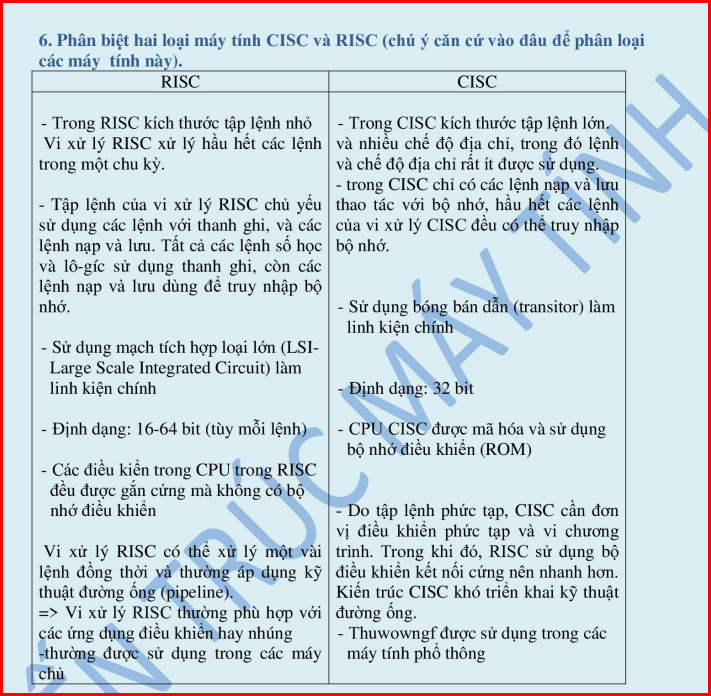
**Kiến trúc Von-neuman: Kiến trúc Havard:**

**** ****

**- Phân biệt :**

* **Máy tính dựa trên kiến trúc Harvard có khả năng đạt được tốc độ xử lý cao hơn máy tính dựa trên kiến trúc von-Neumann do kiến trúc Harvard hỗ trợ hai hệ thống bus độc lập với băng thông lớn hơn.**
* **Hệ thống nhớ trong kiến trúc Harvard hỗ trợ nhiều thao tác đọc/ghi bộ nhớ tại một thời điểm 🡺 giảm xung đột truy nhập bộ nhớ, đặc biệt khi CPU sử dụng kỹ thuật đường ống (pipeline).**
* ****

1. Phân biệt hai loại máy tính CISC và RISC (chú ý căn cứ chính để phân loại các máy tính này).

****

1. Trình bày khái niệm lệnh và quá trình thực hiện lệnh.

**Bài làm**

* **Lệnh máy tính (computer instruction):**
  + **Là một từ nhị phân (binary word);**
  + **Mỗi lệnh được gán một nhiệm vụ cụ thể;**
  + **Lệnh được lưu trữ trong bộ nhớ**
  + **Lệnh được đọc (fetch) từ bộ nhớ vào CPU để giải mã và thực hiện.**
* **Mỗi lệnh có thể được thực hiện theo 5 giai đoạn:**
  + **Đọc lệnh (Instruction fetch - IF): lệnh được đọc từ bộ nhớ về CPU;**
  + **Giải mã (Instruction decode - ID): CPU giải mã lệnh;**
  + **Thực hiện (Instruction execution – EX): CPU thực hiện lệnh;**
  + **Lấy dữ liệu trong bộ nhớ (MEM) – MEMory access**
  + **Lưu kết quả (Write back - WB): kết quả thực hiện lệnh (nếu có) được lưu vào bộ nhớ.**

1. Trình bày cấu trúc lệnh của máy tính và các dạng toán hạng, cho ví dụ

**Bài làm**

* **Dạng tổng quát của lệnh gồm 2 thành phần chính:**
  + **Mã lệnh (Opcode - operation code): mỗi lệnh có mã lệnh riêng**
  + **Địa chỉ của các toán hạng (Addresses of Operands): mỗi lệnh có thể gồm một hoặc nhiều toán hạng. Có thể có các dạng địa chỉ toán hạng sau:**
    - **3 địa chỉ (3 toán hạng)**
    - **2 địa chỉ (2 toán hạng)**
    - **1 địa chỉ (1 toán hạng)**
    - **1,5 địa chỉ**
    - **0 địa chỉ (không toán hạng)**
* **Toán hạng 3 địa chỉ (3 toán hạng):**
  + **Dạng:**
    - **opcode addr1, addr2, addr3**
    - **Mỗi địa chỉ addr1, addr2, addr3 tham chiếu đến một ô nhớ hoặc một thanh ghi.**
  + **Ví dụ:**

**ADD #10, #20, R1; R1= 10 + 20**

**ADD R1, R2, R3; R1 + R2 🡪 R3**

**R1 cộng với R2, kết quả gán vào R3.**

**Ri là thanh ghi của CPU.**

**ADD (A), (B), (C); M[A]+M[B] 🡪 M[C]**

**A, B, C là các ô nhớ.**

* **Toán hạng 2 địa chỉ:**
  + **Dạng:**
    - **opcode addr1, addr2**
    - **Mỗi địa chỉ addr1, addr2 tham chiếu đến một ô nhớ hoặc một thanh ghi.**
  + **Ví dụ:**

**ADD R1, R2; R1 + R2 🡪 R2**

**R1 cộng với R2, kết quả gán vào R2.**

**Ri là thanh ghi của CPU.**

**ADD (A), (B); M[A]+M[B] 🡪 M[B]**

**A, B là các ô nhớ.**

* **Toán hạng 1 địa chỉ:**
  + **Dạng:**
    - **opcode addr1**
    - **Địa chỉ addr1 tham chiếu đến một ô nhớ hoặc một thanh ghi.**
    - **Ở dạng 1 địa chỉ, thanh ghi Racc (Accumulator) được sử dụng như địa chỉ addr2 trong dạng 2 địa chỉ.**
  + **Ví dụ:**

**ADD R1; R1 + Racc 🡪 Racc**

**R1 cộng với Racc, kết quả gán vào Racc.**

**R1 là thanh ghi của CPU.**

**ADD (A); M[A]+ Racc 🡪 Racc**

**A là một ô nhớ.**

* **Toán hạng 1,5 địa chỉ:**
  + **Dạng:**
    - **opcode addr1, addr2**
    - **Một địa chỉ tham chiếu đến một ô nhớ và địa chỉ còn lại tham chiếu đến một thanh ghi.**
    - **Dạng 1,5 địa chỉ là dạng hỗn hợp giữa ô nhớ và thanh ghi.**
  + **Ví dụ:**

**ADD A, R1; M[A] + R1 🡪 R1**

**Nội dung ô nhớ A cộng với R1, kết quả gán vào R1.**

**R1 là thanh ghi của CPU và A là một ô nhớ.**

* **Toán hạng 0 địa chỉ: được sử dụng trong các lệnh thao tác với ngăn xếp: PUSH và POP**

1. Trình bày kỹ thuật đường ống 5 giai đoạn trong hệ thống máy tính. Tranh chấp dữ liệu là gì và cách khắc phục.

**Bài làm**

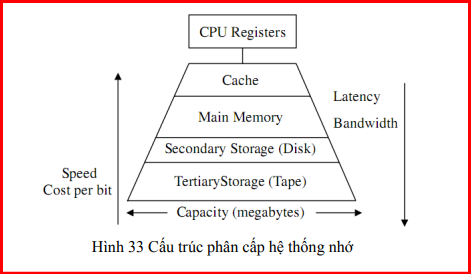
* **Kỹ thuật đường ống 5 giai đoạn**
  + **Instruction Fetch - IF: Đọc lệnh từ bộ nhớ (hoặc cache)**
  + **Instruction Decode - ID: Giải mã lệnh và đọc các toán hạng**
  + **Execute - EX: Thực hiện lệnh; nếu là lệnh truy nhập bộ nhớ: tính toán địa chỉ bộ nhớ**
  + **Memory Access - MEM: Đọc/ghi bộ nhớ; no-op nếu không   
    truy nhập bộ nhớ**
  + **Write Back - WB: Ghi kết quả vào các thanh ghi.**

**- Tranh chấp dữ liệu là một trong các vấn đề lớn của cơ chế ống lệnh và tranh chấp dữ liệu kiểu đọc sau khi ghi (RAW – Read After Write) là dạng xung đột dữ liệu hay gặp nhất.**

**- Cách khắc phục:**

* **Nhận dạng RAW hazard khi nó diễn ra**
* **Khi RAW hazard xảy ra, tạm dừng (stall) pipeline cho đến khi lệnh phía trước hoàn tất giai đoạn WB.**
* **Có thể sử dụng compiler để nhận dạng RAW và:**
  + **Chèn thêm các lệnh NO-OP vào giữa các lệnh có thể gây ra RAW;**
  + **Thay đổi trật tự các lệnh trong chương trình và chèn các lệnh độc lập vào giữa các lệnh có thể gây ra RAW;**
* **Sử dụng phần cứng để nhận dạng RAW và dự đoán trước giá trị dữ liệu phụ thuộc.**

1. Trình bày **hệ thống bộ nhớ phân cấp** trong các hệ thống máy tính

****

* **CPU registers (các thanh ghi của CPU):**
  + **Dung lượng rất nhỏ, khoảng từ vài chục bytes đến vài KB**
  + **Tốc độ truy nhập rất cao (các thanh ghi hoạt động với tốc độ của CPU); thời gian truy nhập khoảng 0,25ns**
  + **Giá thành đắt**
  + **Sử dụng để lưu toán hạng đầu vào và kết quả của các lệnh.**
* **Cache (bộ nhớ cache):**
  + **Còn được gọi là “bộ nhớ thông minh” (smart memory)**
  + **Dung lượng tương đối nhỏ (khoảng 64KB đến 16MB)**
  + **Tốc độ truy nhập cao; thời gian truy nhập khoảng 1-5ns**
  + **Giá thành đắt**
  + **Sử dụng để lưu lệnh và dữ liệu cho CPU xử lý.**
* **Main memory (bộ nhớ chính):**
  + **Gồm ROM và RAM có dung lượng khá lớn (khoảng 256MB-4GB)**
  + **Tốc độ truy nhập chậm; thời gian truy nhập khoảng 50-70ns**
  + **Giá thành tương đối rẻ**
  + **Sử dụng để lưu lệnh và dữ liệu của hệ thống và của người dùng**
* **Secondary/Tertiary memory (bộ nhớ thứ cấp – bộ nhớ ngoài):**
  + **Có dung lượng rất lớn, khoảng từ 20GB-1000GB**
  + **Tốc độ truy nhập rất chậm**
  + **Giá thành rẻ**
  + **Sử dụng để lưu dữ liệu lâu dài dưới dạng các tệp (files)**

1. **Trình bày tổ chức và hoạt động của IC nhớ.**
2. Trình bày đặc điểm bộ nhớ ROM, ứng dụng bộ nhớ ROM trong hệ thống máy tính và phân loại bộ nhớ ROM theo công nghệ chế tạo

**Bài làm**

* **ROM là bộ nhớ chỉ đọc (Read Only Memory)**
  + **Việc ghi thông tin vào ROM chỉ có thể được thực hiện bằng các thiết bị hoặc phương pháp đặc biệt;**
* **ROM là bộ nhớ ổn định**
  + **Thông tin trong ROM vẫn được duy trì khi mất nguồn nuôi**
* **ROM là bộ nhớ bán dẫn: mỗi ô nhớ của ROM là một cổng bán dẫn**
* **ROM thường được sử dụng để lưu chương trình khởi động của máy tính**

**\* Các loại ROM:**

* **ROM nguyên thuỷ (Orginal ROM):** 
  + **ROM các thế hệ đầu tiên;**
* **PROM (Programmable ROM):**
  + **ROM có thể lập trình được;**
  + **Thông tin có thể được ghi vào PROM nhờ một thiết bị đặc biệt gọi là bộ lập trình PROM.**
* **EPROM (Erasable programmable read-only memory):**
  + **Là ROM có thể lập trình và xoá được;**
  + **Thông tin trong EPROM có thể xoá được sử dụng tia cực tím có cường độ cao.**
* **EEPROM: (Electrically Erasable PROM):**
  + **Là PROM có thể xoá được thông tin bằng điện**
  + **Có thể ghi được thông tin sử dụng phần mềm chuyên dụng**
* **Flash memory:** 
  + **Là một dạng EEPROM nhưng có tốc độ ghi và đọc thông tin nhanh hơn.**
  + **Bộ nhớ flash chỉ có thể đọc/ghi theo khối.**

1. Trình bày đặc điểm bộ nhớ RAM, phân loại bộ nhớ RAM theo công nghệ chế tạo và ứng dụng bộ nhớ RAM.

**Bài làm**

* **RAM (Random Access Memory) là bộ nhớ truy nhập ngẫu nhiên**
* **RAM là bộ nhớ không ổn định:**
  + **Tất cả thông tin trong RAM sẽ bị mất khi mất nguồn nuôi**
* **RAM là bộ nhớ bán dẫn: mỗi ô nhớ của RAM là một cổng bán dẫn**
* **RAM được sử dụng để lưu các thông tin của hệ thống và của người dùng:**
* **Hai loại RAM cơ bản:**
  + **RAM tĩnh (Static RAM – SRAM):**
    - **Mỗi bít SRAM là một mạch lật – flip-flop**
    - **Thông tin lưu trong các bit SRAM luôn ổn định và không phải “làm tươi” định kỳ**
    - **SRAM nhanh hơn nhưng đắt hơn DRAM.**
  + **RAM động (Dynamic RAM – DRAM):**
    - **Mỗi bít DRAM dựa trên một tụ điện**
    - **Thông tin lưu trong các bit DRAM không ổn định và phải được “làm tươi” định kỳ**
    - **DRAM chậm hơn nhưng rẻ hơn SRAM.**

1. Trình bày khái niệm và đặc điểm hoạt động của bộ nhớ cache.

**Bài làm**

**- Khái niệm:**

* **Cache là một thành phần trong hệ thống nhớ phân cấp của máy tính:**
  + **Cache đóng vai trò trung gian, trung chuyển dữ liệu từ bộ nhớ chính về CPU và ngược lại;**

**- Đặc điểm:**

* **Dung lượng của cache thường nhỏ:**
  + **Với các hệ thống cũ: 16K, 32K,..., 128K**
  + **Với các hệ thống mới: 256K, 512K, 1MB, 2MB, hoặc lớn hơn**
* **Cache có tốc độ truy nhập nhanh hơn nhiều so với bộ nhớ chính;**
* **Giá thành cache (tính theo bit) thường đắt hơn nhiều so với bộ nhớ chính.**
* **Với các hệ thống CPU mới, cache thường được chia thành nhiều mức (levels):**
  + **Mức 1: 16-32KB có tốc độ rất cao**
  + **Mức 2: 1-16MB có tốc độ khá cao**

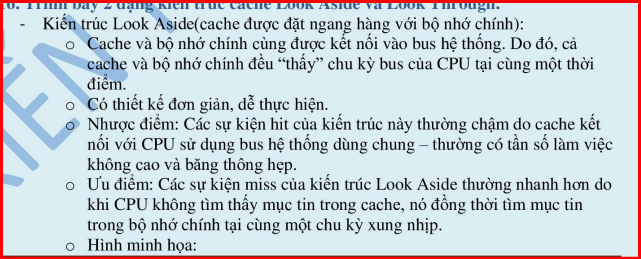
**- Nguyên lý hoạt động của cache :**

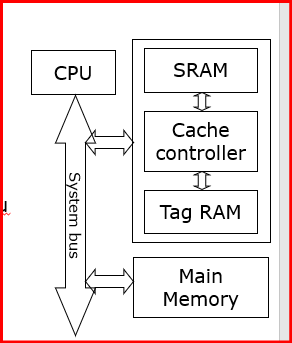
* **Cache được coi là bộ nhớ thông minh:**
  + **Cache có khả năng đoán trước yêu cầu về dữ liệu và lệnh của CPU;**
  + **Dữ liệu và lệnh được chuyển từ bộ nhớ chính về cache 🡪 CPU 🡪 giảm thời gian truy nhập hệ thống nhớ.**
* **Cache hoạt động dựa trên 2 nguyên lý cơ bản:**
  + **Nguyên lý lân cận về không gian (Spatial locality)**
  + **Nguyên lý lân cận về thời gian (Temporal locality)**
* **Nguyên lý lân cận về không gian:**
  + **Nếu một ô nhớ đang được truy nhập thì xác xuất các ô nhớ liền kề với nó được truy nhập trong tương lai gần là rất cao;**
* **Áp dụng:**
  + **Lân cận về không gian được áp dụng cho nhóm lệnh/dữ liệu có tính tuần tự cao;**
* **Giải thích:**
  + **Do các lệnh trong một chương trình thường tuần tự 🡪 cache đọc cả khối lệnh từ bộ nhớ chính 🡪 phủ được lân cận của ô nhớ đang được truy nhập.**
* **Nguyên lý lân cận về thời gian:**
  + **Nếu một ô nhớ đang được truy nhập thì xác xuất nó được truy nhập lại trong tương lai gần là rất cao;**
* **Áp dụng:**
  + **Lân cận về thời gian được áp dụng cho dữ liệu và nhóm lệnh trong vòng lặp;**
* **Giải thích:**
  + **Các phần tử dữ liệu thường được cập nhật, sửa đổi thường xuyên;**
  + **Cache đọc cả khối lệnh từ bộ nhớ chính 🡪 phủ được cả khối lệnh của vòng lặp.**

1. Trình bày 2 dạng kiến trúc cache Look Aside và Look Through.

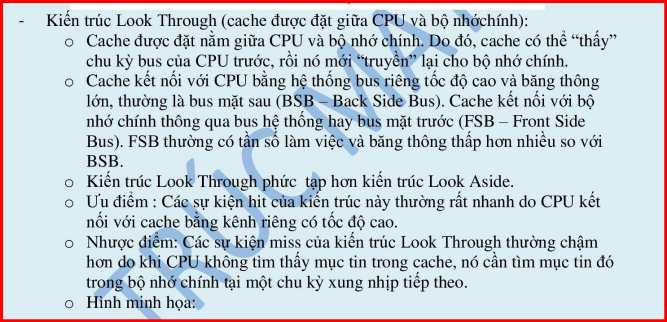
**Bài làm**

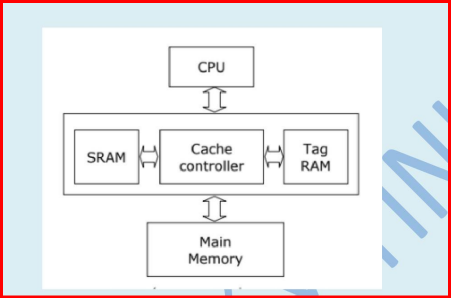
**- Look Aside:**

****

****

**- Look through:**

****

****

1. Trình bày các chính sách thay thế khối (lệnh, dữ liệu) của bộ nhớ cache

**Bài làm**

* **Vì sao phải thay thế dòng cache?**
  + **Ánh xạ dòng (bộ nhớ) 🡪 dòng (cache) thường là ánh xạ nhiều 🡪 một;**
  + **Nhiều dòng bộ nhớ chia sẻ một dòng cache 🡪 các dòng bộ nhớ được nạp vào cache sử dụng một thời gian và được thay thế bởi dòng khác theo yêu cầu thông tin phục vụ CPU.**
* **Chính sách thay thế (replacement policies) xác định các dòng cache nào được chọn bị thay thế bởi các dòng khác từ bộ nhớ.**

**- Có ba chính sách thay thế được sử dụng hiện nay:**

**+ Thay thế ngẫu nhiên (Random Replacement)**

**\* Các dòng cache được lựa chọn để thay thế một cách ngẫu nhiên, không theo một quy luật nào.**

**\* Ưu điểm : Cài đặt đơn giản**

**\* Nhược điểm :**

**+ Hệ số miss cao :**

**Phương pháp này không xem xét đến các dòng cache đang thực sự đƣợc sử dụng.**

**Nếu một dòng cache đang được sử dụng và bị thay thế sẽ xảy ra miss và nó lại cần được đọc từ bộ nhớ chính vào cache.**

**+ Thay thế kiểu vào trước ra trước (FIFO – First In First Out)**

* + **Các dòng cache được đọc vào cache trước sẽ bị thay thế trước**
  + **Ưu:**
    - **Có hệ số miss thấp hơn o với thay thế ngẫu nhiên do phƣơng pháp này có xem xét đến yếu tố lân cận theo thời gian – các dòng nhớ có thời gian tồn tại trong cache lâu nhất có thể có xác suất được sử dụng thấp hơn.**
  + **Nhược:**
    - **Hệ số miss vẫn còn cao**
      * **Thay thế vẫn chưa thực sự xem xét đến các dòng cache đang được sử dụng. Một dòng cache “già” vẫn có thể đang được sử dụng.**
    - **Cài đặt phức tạp do cần có mạch điện tử để theo dõi trật tự nạp các dòng bộ nhớ vào cache.**

**+Thay thế các dòng ít được sử dụng gần đây nhất (LRU – Least Recently Used).**

* + **Các dòng cache ít được sử dụng gần đây nhất được lựa chọn để thay thế.**
  + **Ưu:**
    - **Có hệ số miss thấp nhất so với thay thế ngẫu nhiên và thay thế FIFO**
    - **Do thay thế LRU có xem xét đến các dòng đang được sử dụng - tuân theo yếu tố lân cận theo thời gian một cách chặt chẽ**
  + **Nhược:**
    - **Cài đặt phức tạp do cần phải có mạch điện tử chuyên dụng để theo dõi tần suất sử dụng các dòng cache.**

1. Trình bày các phương thức ghi dữ liệu trong bộ nhớ cache (khi hit và miss)

**Bài làm**

**- Với trường hợp ghi thông tin và nếu đó là trường hợp hit, có thể áp dụng một trong 2 chính sách ghi: ghi thẳng (write through) và ghi trễ (write back). + Với phương pháp ghi thẳng, mẩu tin cần ghi được lưu đồng thời ra cache và bộ nhớ chính.**

**+ Với phương pháp ghi trễ, mẩu tin trước hết được ghi ra cache và dòng cache chứa mẩu tin sẽ đƣợc ghi ra bộ nhớ chính khi nó bị thay thế.**

**- Với trường hợp ghi thông tin và nếu đó là trƣờng hợp miss, cũng có thể áp dụng một trong hai chính sách ghi: ghi có đọc lại (write allocate / fetch on write) và ghi không đọc lại (write non-allocate).**

**+ Với phương pháp ghi có đọc lại, mẩu tin trƣớc hết đƣợc ghi ra bộ nhớ chính, và sau đó dòng nhớ chứa mẩu tin vừa ghi đƣợc đọc vào cache.**

**+ Với phương pháp ghi không đọc lại, mẩu tin chỉ đƣợc ghi ra bộ nhớ chính.**

1. Trình bày các phương thức đọc dữ liệu trong bộ nhớ cache (khi hit và miss)

**Bài làm**

**- Xét trường hợp đọc thông tin và nếu đó là trường hợp hit (mẩu tin cần đọc có trong cache): mẩu tin được đọc từ cache vào CPU và bộ nhớ chính không tham gia. Nhƣ vậy thời gian CPU truy nhập mẩu tin bằng thời gian CPU truy nhập cache.**

**- Ngược lại, nếu đọc thông tin và đó là trƣờng hợp miss (mẩu tin cần đọc không có trong cache): mẩu tin trƣớc hết đƣợc chuyển từ bộ nhớ chính vào cache, sau đó nó đƣợc đọc từ cache vào CPU**

**=> Đây là trƣờng hợp xấu nhất: thời gian CPU truy nhập mẩu tin bằng thời gian truy nhập cache cộng với thời gian cache truy nhập bộ nhớ chính – còn gọi là miss penalty (gấp đôi thời gian truy cập khi đoán trượt)**

1. Trình bày các tham số hiệu năng của bộ nhớ cache.

**Bài làm**

* **Kích thước cache:**
  + **Số liệu thống kê cho thấy:** 
    - **Kích thước cache không ảnh hưởng nhiều đến hệ số miss**
    - **Hệ số miss của cache lệnh thấp hơn nhiều so với cache dữ liệu**

**8KB cache lệnh có hệ số miss < 1%**

**256KB cache lệnh có hệ số miss < 0.002%**

**----> tăng kích thước cache lệnh không giảm miss hiệu quả.**

**8KB cache dữ liệu có hệ số miss < 4%**

**256KB cache dữ liệu có hệ số miss < 3%**

**----> tăng kích thước cache dữ liệu lên 32 lần, hệ số miss giảm 25%   
(từ 4% xuống 3%).**

* **Kích thước cache:**
  + **Cache có kích thước lớn:**
    - **Có thể tăng được số dòng bộ nhớ lưu trong cache**
    - **Giảm tần suất tráo đổi các dòng cache của các chương trình khác nhau với bộ nhớ chính**
    - **Cache lớn thường chậm hơn cache nhỏ (tại sao?)**
      * **Không gian tìm kiếm địa chỉ ô nhớ lớn hơn**
  + **Xu hướng tương lai: cache càng lớn càng tốt (tại sao?)**
    - **Hỗ trợ đa nhiệm tốt hơn**
    - **Hỗ trợ xử lý song song tốt hơn**
    - **Hỗ trợ tốt hơn các hệ thống CPU nhiều nhân**
* **Tách cache:**
  + **Cache có thể được tách thành cache lệnh (I-Cache) và cache dữ liệu (D-Cache) để cải thiện hiệu năng, do:**
    - **Dữ liệu và lệnh có tính lân cận khác nhau;**
    - **Dữ liệu thường có tính lân cận về thời gian cao hơn lân cận về không gian; lệnh có tính lân cận về không gian cao hơn lân cận về thời gian;**
    - **Cache lệnh chỉ cần hỗ trợ thao tác đọc; cache dữ liệu cần hỗ trợ cả 2 thao tác đọc và ghi 🡪 tách cache giúp tối ưu hoá dễ dàng hơn;**
    - **Tách cache hỗ trợ nhiều lệnh truy nhập đồng thời hệ thống nhớ 🡪 giảm xung đột tài nguyên cho CPU pipleline.**
* **Tạo cache thành nhiều mức:**
  + **Cải thiện được hiệu năng hệ thống do hệ thống cache nhiều mức có khả năng dung hoà tốt hơn tốc độ của CPU với bộ nhớ chính.**

**CPU L1 L2 L3 Bộ nhớ chính**

**1ns 5ns 15ns 30ns 60ns**

**1ns 5ns 60ns**

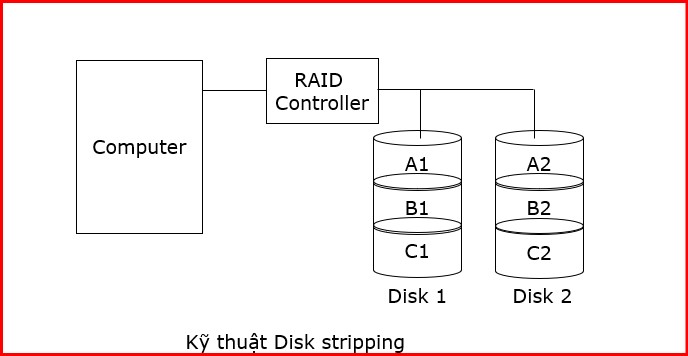
* + **Trên thực tế, đa số cache được tổ chức thành 2 mức: L1 và L2. Một số cache có 3 mức: L1, L2 và L3.**
  + **Giảm giá thành hệ thống nhớ.**

1. RAID là gì? Trình bày các kỹ thuật cơ bản tạo RAID.

**Bài làm**

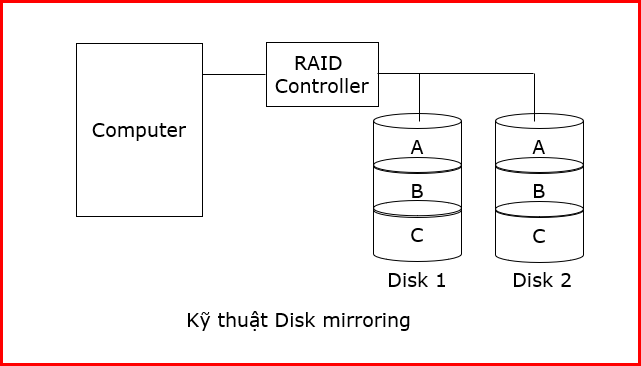
* **RAID (Redundant Array of Independent Disks) là một công nghệ tạo các thiết bị lưu trữ tiên tiến trên cơ sở các ổ đĩa độc lập, nhằm các mục đích:**
  + **Tốc độ cao (High performance / speed)**
  + **Tính tin cậy cao (High reliability)**
  + **Dung lượng lớn (Large volume)**
* **RAID:**
  + **Một mảng của các ổ đĩa HDD, SSD;**
  + **Các đĩa cứng theo chuẩn SATA, SCSI (SAS), NVME mới hỗ trợ tạo RAID.**
* **Hai kỹ thuật chính được sử dụng trong RAID:**
  + **Tạo lát đĩa (Disk Stripping):**
    - **Ghi: Dữ liệu được chia thành các khối, mỗi khối được ghi đồng thời vào một đĩa độc lập;**
    - **Đọc: Các khối dữ liệu được đọc đồng thời ở các đĩa độc lập, và được ghép lại tạo dữ liệu hoàn chỉnh.**

**🡺 tốc độ truy nhập được cải thiện.**

****

* + **Soi gương đĩa (Disk Mirroring):**
    - **Ghi: Dữ liệu được chia thành các khối, mỗi khối được ghi đồng thời vào nhiều đĩa độc lập;**
    - **Tại mọi thời điểm ta luôn có nhiều hơn 1 bản sao của dữ liệu.**

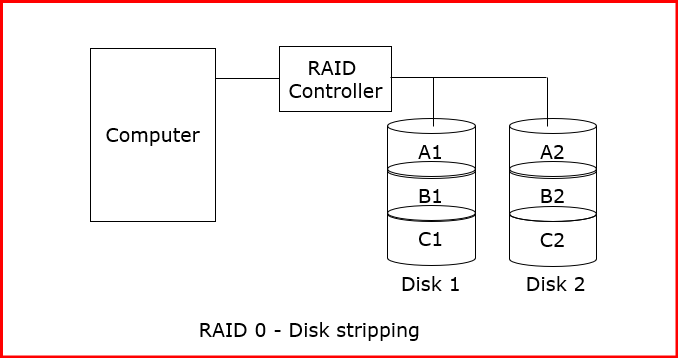
**🡺 Tính tin cậy được cải thiện.**

****

1. Trình bày các loại RAID cơ bản: RAID0, RAID1, RAID10, RAID5 và RAID6

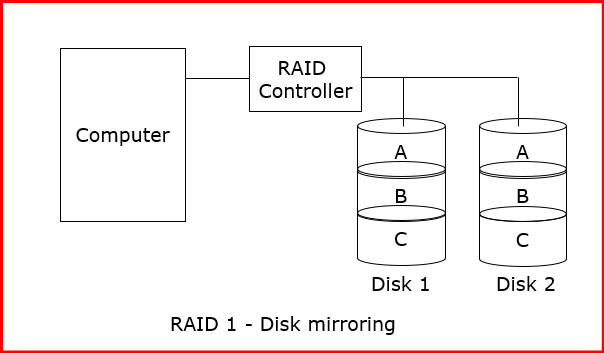
**Bài làm**

**- RAID0 :**

****

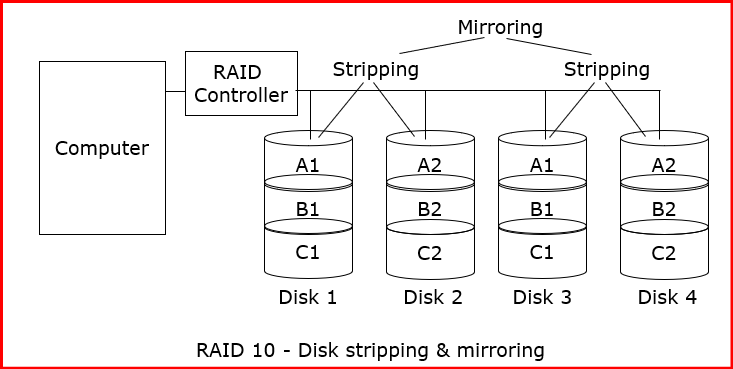
* **Đặc điểm:**
  + **Yêu cầu tối thiểu 2 ổ đĩa cứng vật lý**
  + **Sử dụng kỹ thuật tạo lát đĩa (disk stripping hoặc parallel read/write)**
* **Ưu điểm:**
  + **Nhanh: tốc độ truy nhập tỷ lệ thuận với số đĩa của RAID**
  + **Tăng dung lượng: dung lượng RAID bẳng tổng dung lượng các đĩa đơn.**
* **Nhược điểm:**
  + **Tính an toàn chỉ tương đương một đĩa đơn.**

**- RAID 1:**

****

* **Đặc điểm:**
  + **Yêu cầu tối thiểu 2 ổ đĩa cứng vật lý**
  + **Sử dụng kỹ thuật soi gương đĩa (disk mirroring)**
* **Ưu điểm:**
  + **An toàn cao: do tại mỗi thời điểm RAID luôn chứa nhiều bản copy của dữ liệu ở các đĩa vật lý khác nhau.**
* **Nhược điểm:**
  + **Tốc độ tương đương một đĩa đơn.**
  + **Dung lượng tương đương một đĩa đơn.**

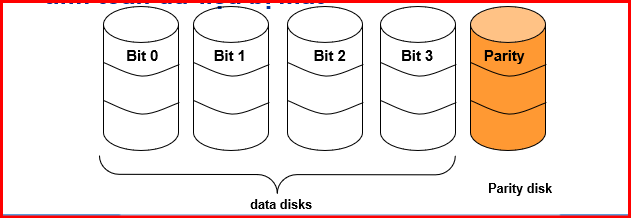
**- RAID 10 :**

****

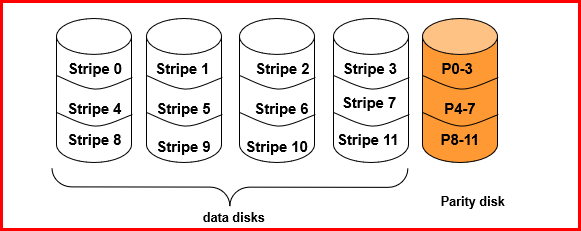
* **Đặc điểm:**
  + **Yêu cầu tối thiểu 4 ổ đĩa cứng vật lý**
  + **Sử dụng kỹ thuật tạo lát đĩa (disk stripping) và soi gương đĩa (disk mirroring)**
* **Ưu điểm:**
  + **An toàn cao: do tại mỗi thời điểm RAID luôn chứa nhiều bản copy của dữ liệu ở các đĩa vật lý khác nhau.**
  + **Nhanh: tốc độ truy nhập tỷ lệ với số đĩa của RAID**
* **Nhược điểm:**
  + **Dung lượng tăng nhưng chỉ bằng ½ tổng dung lượng các đĩa đơn.**
  + **Đắt tiền.**

**- RAID 5:**

* **Sử dụng 1 đĩa parity**
  + **Mỗi bit của đĩa parity là một hàm parity của các bit tương ứng trên tất cả các đĩa khác**
* **Đọc truy cập tất cả các đĩa dữ liệu**
* **Ghi truy cập tất cả các đĩa cộng với đĩa parity**
* **Khi lỗi đĩa, đọc các đĩa còn lại cộng với đĩa parity để tính toán dữ liệu bị mất**

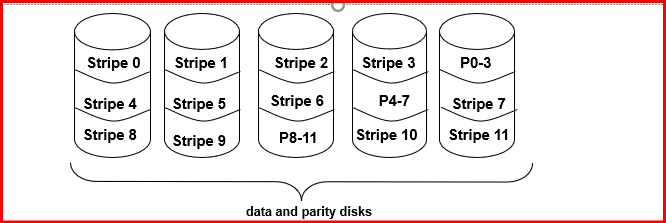
****

* **Đọc truy cập tất cả các đĩa dữ liệu**
* **Ghi truy cập tới tất cả các đĩa dữ liệu cộng với đĩa parity**
* **Tải nặng trên đĩa parity**

****

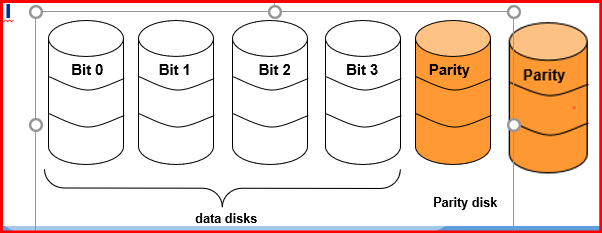
**- RAID 50 :**

* **Parity phân bố đan xen giữa các khối**
* **Giống như phương thức parity, nhưng phân bố thông tin parity trên tất cả các đĩa (cũng như dữ lệu trên tất cả các đĩa)**
* **Hoạt động đọc tốt hơn, hoạt động ghi rộng**
* **Đọc có thể tốt hơn SLEDs và RAID-0**

****

**- RAID 6:**

* **Level 5 với một bit parity mở rộng**
* **Có thể chịu được 2 lỗi**
  + **Các số lẻ của 2 lỗi đồng thời là gì?**
* **Có thể thực hiện tốt hơn mức 5 khi đọc, chậm hơn khi ghi**

****

1. Trình bày NAS là gì và đặc điểm của NAS?

**Bài làm**

* **Khái niệm:**

**- NAS là một máy chủ chuyên dùng làm thiết bị lưu trữ;**

**- Đặc điểm :**

* **NAS được kết nối vào mạng (thường là LAN tốc độ cao) và cung cấp các dịch vụ lưu trữ thông qua mạng;**
* **NAS thường dựa trên nền tảng là một RAID có tốc độ cao, dung lượng lớn và độ tin cậy rất cao.**
* **NAS có thể cung cấp dịch vụ lưu trữ cho hầu hết các loại máy chủ có cấu hình phần cứng khác nhau và chạy các hệ điều hành khác nhau.**

1. Trình bày SAN là gì và đặc điểm của SAN?

**Bài làm**

**Khái niệm:**

* **SAN là một mạng của các máy chủ chuyên dụng cung cấp dịch vụ lưu trữ;**
* **Đặc điểm:**
  + **Tốc độ truy nhập rất cao;**
  + **Dung lượng cực lớn;**
  + **Độ an toàn rất cao**
    - **An toàn dữ liệu cục bộ**
    - **An toàn dữ liệu với các bản copy được đồng bộ ở khoảng cách xa về địa lý**
* **SAN thường được tổ chức dưới dạng các hệ thống file phân tán (Distributed File System).**

1. Trình bày các thanh ghi của vi xử lý Intel 8086
2. Trình bày khuôn dạng lệnh của vi xử lý Intel 8086

**Bài làm**

**- Dạng tổng quát của lệnh: 2 thành phần:**

**+ Mã lệnh và địa chỉ của các toán hạng**

**• Độ dài của từ lệnh: 8, 16, 24, 32 và 64 bit.**

**• Lệnh của 8086/8088 có thể có độ dài 1-6 byte**

****

1. Trình bày các nhóm lệnh hợp ngữ của vi xử lý Intel 8086
2. Trình bày các chế độ định địa chỉ của vi xử lý Intel 8086 và cho ví dụ bằng lệnh hợp ngữ
3. **Chế độ địa chỉ thanh ghi (register addressing mode). 2. Chế độ địa chỉ tức thì (immediate addressing mode). 3. Chế độ địa chỉ trực tiếp (direct addressing mode). 4. Chế độ địa chỉ gián tiếp qua thanh ghi (register indirect addressing mode). 5. Chế độ địa chỉ tương đối cơ sở (based indexed relative addressing mode). 6. Chế độ địa chỉ tương đối chỉ số (indexed relative** **addressing mode). 7. Chế độ địa chỉ tương đối chỉ số cơ sở (based indexed relative addressing mode).**
4. Hoạt động ngắt là gì và phân loại ngắt của VXL Intel 8086.

**- Ngắt là việc tạm dừng việc chương trình đang chạy để CPU có thể chạy một chương trình khác nhằm xử lý một yêu cầu do bên ngoài đưa tới CPU như yêu cầu vào/ra hoặc do chính yêu cầu của bên trong CPU như lỗi trong khi tính toán.**

**- Trong hệ vi xử lý 8086 có thể xếp các nguyên nhân gây ra ngắt CPU vào 3 nhóm như sau:**

**+ Nhóm các ngắt cứng: đó là các yêu cầu ngắt CPU do các tín hiệu đến từ các chân INTR và NMI.**

**+ Nhóm các ngắt mềm: khi CPU thực hiện các lệnh ngắt dạng INT N, trong đó N là số hiệu (kiểu) ngắt nằm trong khoảng 00-FFH (0-255).**

**+ Nhóm các hiện tượng ngoại lệ: đó là các ngắt do các lỗi nảy sinh trong quá trình hoạt động của CPU như phép chia cho 0, xảy ra tràn khi tính toán.**

1. Trình bày quá trình xử lý ngắt của VXL Intel 8086.

**- Khi có yêu cầu ngắt kiểu N đến CPU và nếu yêu cầu đó được phép, CPU thực hiện các công việc sau:**

**1. SP ← SP-2, [SP] ← FR, trong đó [SP] là ô nhớ do SP chỉ ra. (chỉ ra đỉnh mới của ngắn xếp, cất thanh ghi cờ vào đỉnh ngăn xếp)**

**2. IF ← 0, TF ← 0. (cấm các ngắt khác tác động vào CPU, cho CPU chạy ở chế độ bình thường)**

**3. SP ← SP-2, [SP] ← CS. (chỉ ra đỉnh mới của ngăn xếp, cất phần địa chỉ đoạn của địa chỉ trở về vào đỉnh ngăn xếp)**

**4. SP ← SP-2, [SP] ← IP (chỉ ra đỉnh mới của ngăn xếp, cất phần địa chỉ lệch của địa chỉ trờ về vào đỉnh ngăn xếp)**

**5. [N\*4] → IP, [N\*4+2] → CS (lấy lệnh tại địa chỉ mới của chương trình con phục vụ ngắt kiểu N tương ứng trong bảng vectơ ngắt)**

**6. Tại cuối chương trình phục vụ ngắt, khi gặp lệnh IRET [SP] → IP, SP ← SP+2 [SP] → CS, SP ← SP+2 [SP] → FR, SP ← SP+2 (bộ vi xử lý quay lại chương trình chính tại địa chỉ trở về và với giá trị cũ của thanh ghi cờ được lấy ra từ ngăn xếp).**