TỔNG QUAN

1_Tổng quan máy tính
2_Biểu diễn số nguyên
3_ Biểu diễn số thực
4,5,6_Lập trình Hợp Ngữ
7_Mạch Logic
8_Thiết kế CPU
9_Pipeline
10_Bộ nhớ
11_Hệ thống nhập xuất

DỊCH LUẬN LÝ

-Dịch trái (sll – shift left logical): Thêm vào các bit 0 bên phải
-Dịch phải (srl – shift right logical): Thêm vào các bit 0 bên trái

DỊCH SỐ HỌC

-Dịch phải (sra – shift right arithmetic): Thêm các bit = giá trị bit dấu bên trái

QUÁ TRÌNH PHÁT TRIỂN

Thế hệ Khoảng thời gian Công nghệ
1.1940 – 1956 Vacuum tubes (đèn chân không)
2 1956 – 1963 Transistors (linh liện bán dẫn)
3 1964 – 1971 Integrated Circuits (vi mạch tích hợp)
4 1971 – nay Microprocessors (vi xử lý)
5 Tương lai Parallel Processing

ĐỊNH LUẬT MOORE

Số lượng transistor tích hợp trong 1 IC tăng gấp đôi mỗi 1.5 năm(18 tháng).

5 THÀNH PHẦN CƠ BẢN:

Input, output, memory, processor, datapath, control.
<Bộ xử lý, bộ nhớ chính, hệ thống kết nối , thiết bị nhập/ xuất>

WAFER (ĐẾ CHIP): Tấm silicon mỏng đã được cấy vật liệu khác nhau để tạo ra những vi mạch
-Có kích thước trung bình từ 25,4mm (1inch) – 200mm (7.9inch).

CHIP: Có thể hiểu là mạch tích hợp (Integrated Circuit) gắn trên đế chip (wafer) nhằm xử lý các công việc trên máy tính.
-Chip có kích thước rất nhỏ nhưng có thể chứa hàng chục triệu transistor, số lượng trasistor càng lớn thì tốc độ truyền và xử lý tín hiệu càng nhanh
-Hiện nay có các loại chip xử lý: 4, 8, 16, 32, 64 bit

-Nhiệm vụ cơ bản nhất của CPU là phải thực hiện các lệnh được yêu cầu, gọi là instruction
-Các CPU sẽ sử dụng các tập lệnh (instruction set) khác nhau để có thể giao tiếp với nó

-KÍCH THƯỚC LỆNH BỊ ẢNH HƯỞNG BỞI:

+Cấu trúc đường truyền bus +Kích thước và tổ chức bộ nhớ +Tốc độ CPU
-Giải pháp tối ưu lệnh:
+Dùng lệnh có kích thước ngắn, mỗi lệnh chỉ nên được thực thi trong đúng 1 chu kỳ CPU
+Dùng bộ nhớ cache

BỘ LỆNH MIPS

Được xây dựng theo kiến trúc (RISC) với 4 nguyên tắc: +Càng đơn giản, càng ổn định. +Càng nhỏ gọn, xử lý càng nhanh. +Tăng tốc xử lý cho những trường hợp thường xuyên xảy ra
+Thiết kế đòi hỏi sự thỏa hiệp tốt

-A==B
beq $s0, $s1, label #if ($s0==$s1) goto label
-A!=B
bne $s0, $s1, label #if ($s0!=$s1) goto label
-A < B
slt $t0, $s0, $s1 # if (a < b) then $t0 = 1
bne $t0, $0, Label # if (a < b) then goto Label
-A > B
slt $t0, $s1, $s0 # if (b < a) then $t0 = 1
bne $t0, $0, Label # if (b < a) then goto Label
-A ≥ B
slt $t0, $s0, $s1 # if (a < b) then $t0 = 1
beq $t0, $0, Label # if (a ≥ b) then goto Label
- A ≤ B
slt $t0, $s1, $s0 # if (b < a) then $t0 = 1
beq $t0, $0, Label # if (b ≥ a) then goto Label

NGUYÊN TẮC LƯU DỮ LIỆU TRONG BỘ NHỚ

-MIPS thao tác và lưu trữ dữ liệu trong bộ nhớtheo 2 nguyên tắc:
+ALIGNMENT RESTRICTION:Các đối tượng lưu trong bộ nhớ (từ nhớ) phải bắt đầu tại địa chỉ là bội số của kích thước đối tượng.
-Mà mỗi từ nhớ có kích thước là 32 bit = 4 byte = kích thước lưu trữ của 1 thanh ghi trong CPU
-Như vậy, từ nhớ phải bắt đầu tại địa chỉ là bội số của 4
+BIG ENDIAN

CÓ 3 FORMAT LỆNH TRONG MIPS:

-R-format: Dùng trong các lệnh tính toán số học (add, sub, and, or, nor, sll, srl, sra…)

6 bits	5	5	5	5	6
opcode	rs	rt	rd	shmat	funct

-I-format: Dùng trong các lệnh thao tác với hằng số, chuyển dữ liệu với bộ nhớ, rẽ nhánh

6 bits	5	5	16
opcode	rs	rt	immediate

-J-format: Dùng trong các lệnh nhảy (jump – C: goto)

6 bits	26
opcode	target address

QUY TRÌNH THỰC THI LỆNH (EXECUTE CYCLE)

-Tính địa chỉ lệnh, Nạp lệnh, Giải mã lệnh, Tính địa chỉ của toán hạng, Nạp toán hạng, Thực hiện lệnh, Tính địa chỉ của toán hạng chứa kết quả, Ghi kết quả.
-Các bước này được lặp đi lặp lại cho tất cả các lệnh tiếp theo
-Quy trình này gọi là Instruction cycle – vòng lặp xử lý lệnh

Tên cổng	Hình vẽ đại diện	Hàm đại số Bun
AND		x.y hay xy
OR		x + y
XOR		x ⊕ y
NOT		x' hay $\overline{x}$
NAND		(x .y)' hay $\overline{x.y}$
NOR		(x + y)' hay $\overline{x + y}$
NXOR		(x ⊕ y)' hay $\overline{x \oplus y}$

logic

- x SHL y = x . 2^y; x SHR y = x / 2^y
- AND dùng để tắt bit (AND với 0 luôn = 0); OR dùng để bật bit (OR với 1 luôn = 1)
- XOR, NOT dùng để đảo bit (XOR với 1 = đảo bit đó)
- x AND 0 = 0; x XOR x = 0
- Lấy giá trị tại bit thứ i của x: (x SHR i) AND 1;
- Gán giá trị 1 tại bit thứ i của x: (1 SHL i) OR x;
- Gán giá trị 0 tại bit thứ i của x: NOT(1 SHL i) AND x; Đảo bit thứ i của x: (1 SHL i) XOR x

BIỂU DIỄN SỐ THỰC SAU THEO DẠNG SỐ CHẤM ĐỘNG CHÍNH XÁC ĐƠN (32 bit): X = -5.25

Bước 1: Đổi X sang hệ nhị phân
X = $-5.25_{10} = -101.01_2$
Bước 2: Chuẩn hóa theo dạng $\pm 1.F * 2^E$
X = -5.25 = -101.01 = $-1.0101 * 2^2$
Bước 3: Biểu diễn Floating Point
- Số âm: bit dấu Sign = 1
- Số mũ E = 2 -> Phần mũ exponent với số thừa K=127 được biểu diễn:->Exponent = E + 127 = 2 + 127 = 129_{10} = $1000\ 0001_2$
- Phần định trị = 0101 0000 0000 0000 0000 000 (Thêm 19 số 0 cho đủ 23 bit)
- Kết quả nhận được: 1 1000 0001 0101 0000 0000 0000 0000 000

NGÔN NGỮ MÁY (MACHINE LAGUAGE)

-Ngôn ngữ máy cho phép người lập trình đưa ra các hướng dẫn đơn giản mà bộ vi xử lý (CPU) có thể thực hiện được ngay:+Các hướng dẫn này được gọi là chỉ thị / lệnh (instruction) hoặc mã máy (machine code)
+Mỗi bộ vi xử lý (CPU) có 1 ngôn ngữ riêng, gọi là bộ lệnh (instruction set)
-Trong cùng 1 dòng vi xử lý (processor family) bộ lệnh gần giống nhau

INSTRUCTION

-Là dãy bit chứa yêu cầu mà bộ xử lý trongCPU (ALU) phải thực hiện
-Instruction gồm 2 thành phần:
+Mã lệnh (opcode): thao tác cần thực hiện
+Thông tin về toán hạng (operand): các đối tượng bị tác động bởi thao tác chứa trong mã lệnh

ISA (Instruction Set Architecture)

-Tập lệnh dành cho những bộ vi xử lý có kiến trúc tương tựnhau

-MỘT SỐ ISA THÔNG DỤNG:

-Dòng vi xử lý 80x86 (gọi tắt x86) của Intel
+IA-16: Dòng xử lý 16 bit +IA-32: Dòng xử lý 32 bit
+IA-64: Dòng xử lý 64 bit
-MIPS: Dùng rất nhiều trong hệ thống nhúng (embedded system)
-PowerPC của IBM

-THIẾT KẾ ISA: CISC & RISC

+Có 2 trường phái thiết kế bộ lệnh:
-Complete Instruction Set Computer (CISC): bộ lệnh gồm rất nhiều lệnh, từ đơn giản đến phức tạp
-Reduced Instruction Set Computer (RISC): bộ lệnh chỉ gồm các lệnh đơn giản

BỘ PHẬN PHIÊN DỊCH (INTERPRETER)

HỢP NGỮ (ASSEMBLY LANGUAGE)

ADDRESSING MODE: Là phương thức định vị trí (địa chỉ hóa) các toán hạng trong kiến trúc MIPS
-Có 5 phương pháp chính:
+Immediate addressing (Vd: addi $t0, $t0, 5)
Toán hạng = hằng số 16 bit trong câu lệnh
+Register addressing (Vd: add $t0, $t0, $t1) Toán hạng = nội dung thanh ghi
+Base addressing (Vd: lw $t1, 8($t0)) Toán hạng = nội dung ô nhớ (địa chỉ ô nhớ = nội dung thanh ghi + hằng số 16 bit trong câu lệnh)
+PC-relative addressing (Vd: beq $t0, $t1, Label)
Toán hạng = địa chỉ đích lệnh nhảy = nội dung thanh ghi PC + hằng số 16 bit trong câu lệnh
+Pseudodirect addressing (Vd: j 2500)
Toán hạng = địa chỉ đích lệnh nhảy = các bit cao thanh ghi PC + hằng số 26 bit trong câu lệnh

THIẾT KẾ MODUL NHỚ32kb x 32bit từ chip nhớ 32kb x 8bit.

-32kb x 32bit ($32=2^5$bit ->32KB = 2^{15}->15 đường địa chỉ a0-a14.
-32bit = 32 đường dữ liệu d0-d31.
32kb x 8bit (15 đường địa chỉ a0-a14. 8 đường dữ liệu d0-d7)

CHIPSET: là tập hợp nhiều chip gắn kết lại với nhau trên cùng 1 đế chip (wafer) để xử lý nhiều công việc trên máy tính
-Một số chipset thông dụng:
CPU: Đơn vị xử lý trung tâm; **GPU**: Đơn vị xử lý đồ họa trên máy;
RAM: Bộ nhớ truy cập tức thời chuyên phục vụ cho CPU; **BẮC CẦU BẮC** (tích hợp trên mainboard): Hỗ trợ truyền thông tin cho CPU, RAM, nằm sát CPU (Hệ thống Mainboard AMD không có chipset này vì được tích hợp ngay trên CPU); **BẮC CẦU NAM** (tích hợp trên mainboard): Quản lý thiết bị ngoại vị như HDD, Mouse, Keyboard…Nằm cuối mainboard

- Bus (hệ thống dẫn đường) : liên kết các thành phần lại với nhau . Gồm có : bus cục bộ (đường dẫn cục bộ nối các thành phần bên trong 1 thiết bị) và bus hệ thống (hệ thống dẫn đường liên quan các thiết bị quan trọng như CPU, bộ nhớ và các mạch ra vào).
- PHÂN LOẠI : bus địa chỉ, bus dữ liệu, bus điều khiển.
- NGẮT : là dừng chương trình đang thực hiện để thực hiện 1 chương trình khác. Phân loại : 3 loại **+ Ngắt cứng** : sinh ra do các tín hiệu INTR (ngắt che được) hay NMI(ko che được) , gồm có 2 loại : ngắt che được và ngắt không che được.**+ Ngắt mềm** : sinh ra do câu lệnh INT. **+ Ngắt tự động** (ngoại lệ) : sinh ra do thực hiện các lệnh của CPU như chia 0, đặt cờ ngắt....

S Exp Significand (Fraction)

-Largest positive normalized number: +1.[23 số 1] * 2^{127}
0 1111 1110 1111 1111 1111 1111 1111 111
-Smallest positive normalized number: +1.[23 số 0] * 2^{-126}
0 0000 0001 0000 0000 0000 0000 0000 000
-Tương tự cho số negative (số âm)
-Largest positive denormalized number: +0.[23 số 1] * 2^{-127}
0 0000 0000 1111 1111 1111 1111 1111 111
Tuy nhiên IEEE 754 quy định là +0.[23 số 1] * 2^{-126}vì muốn tiến gần hơn với "Smallest positive normalized number = +1.[23 số 0] * 2^{-126}"
-Smallest positive denormalized number: +1.[22 số 0]1 * 2^{-127}
0 0000 0000 0000 0000 0000 0000 0000 001. Tuy nhiên IEEE 754 quy định là +0.[22 số 0]1 * 2^{-126}
-Tương tự cho số negative (số âm)

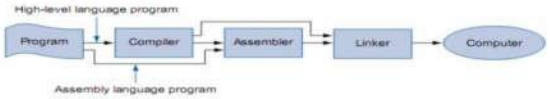
COMPILER -Trình biên dịch ngôn ngữ cấp cao -> hợp ngữ
-Compiler phụ thuộc vào: +Ngôn ngữ cấp cao được biên dịch
+Kiến trúc hệ thống phần cứng bên dưới mà nó đang chạy

ASSEMBLER-Trình biên dịch hợp ngữ -> ngôn ngữ máy
-Một bộ vi xử lý (đi kèm 1 bộ lệnh xác định) có thể có nhiều Assembler của nhiều nhà cung cấp khác nhau chạy trên các OS khác nhau
-Assembly program phụ thuộc vào Assembler mà nó sử dụng (do các mở rộng, đặc điểm khác nhau giữa các Assembler)
Làm sao để chạy những tập tin này trên máy tính? **->Linker & Loader**

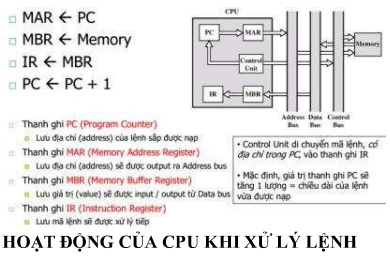
LINKER

-Thực tế khi lập trình, ta sẽ dùng nhiều file (header / source) liên kết và kèm theo các thư viện có sẵn. -Cần chương trình Linker để liên kết các file sau khi đã biên dịch thành mã máy này (Object file)
-Tập tin thực thi (ví dụ: .exe, .bat, .sh)
Khi double click vào những tập tin thực thi, cần chương trình tính toán và tải vào memory để CPU xử lý **-> Loader**

QUÁ TRÌNH THỰC THI FILE TRÊN MÁY



QUÁ TRÌNH NẠP LỆNH (FETCH CYCLE)

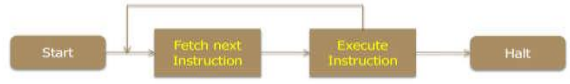


HOẠT ĐỘNG CỦA CPU KHI XỬ LÝ LỆNH

-CPU xử lý lệnh qua 2 bước, gọi là chu kỳ lệnh:
+Nạp lệnh (Fetch): Di chuyển lệnh từ memory vào thanh ghi (register) trong CPU+Thực thi lệnh (Excute): Giải mã lệnh và thực thi thao tác yếu

HOẠT ĐỘNG CỦA CPU KHI XỬ LÝ LỆNH

-CPU xử lý lệnh qua 2 bước, gọi là chu kỳ lệnh:
+Nạp lệnh (Fetch): Di chuyển lệnh từ memory vào thanh ghi (register) trong CPU+Thực thi lệnh (Excute): Giải mã lệnh và thực thi thao tác yếu



->Sẽ cần 4 chip nhớ .

MẠCH TỔ HỢP (TÍCH HỢP)
-Gồm n ngõ vào (input); m ngõ ra (output)
+Mỗi ngõ ra là 1 hàm luận lý của các ngõ vào
-Mạch tổ hợp không mang tính ghi nhớ: Ngõ ra chỉ phụ thuộc vào Ngõ vào hiện tại, không xét những giá trị trong quá khứ
ĐỘ TRỄ MẠCH (PROPAGATION DELAY / GATE DELAY) = Thời điểm tín hiệu ra ổn định - thời điểm tín hiệu vào ổn định
THIẾT KẾ- Lập bảng chân trị. - Viết hàm luận lý - Vẽ sơ đồ mạch và thử nghiệm
MỘT SỐ MẠCH TỔ HỢP CƠ BẢN
- Mạch toàn cộng (Full adder) - Mạch giải mã (Decoder)- Mạch mã hoá (Encoder)
SOP – SUM OF PRODUCTS-POS – PRODUCT OF SUM

MẠCH TOÀN CỘNG (FULL ADDER - FA)
-Mạch tổ hợp thực hiện phép cộng số học 3 bit
-Gồm 3 ngõ vào (A, B: bit cần cộng – C_i: bit nhớ) và 2 ngõ ra (kết quả có thể từ 0 đến 3 với giá trị 2 và 3 cần 2 bit biểu diễn – S: ngõ tổng, C_0 : ngõ nhớ)
MẠCH MÃ HOÁ NHỊ PHÂN (BINARY ENCODER)
Có 2^n (hoặc ít hơn) ngõ vào, n ngõ ra
-Quy định chỉ có duy nhất một ngõ vào mang giá trị = 1 tại một thời điểm
-Nếu ngõ vào = 1 đó là ngõ thứ k thì các ngõ ra tạo thành số nhị phân có giá trị = k
MẠCH MÃ HOÁ THEO THỨ TỰ (PRIORITY ENCODER)
-Các ngõ vào được xem như có độ ưu tiên
-Giá trị ngõ ra phụ thuộc vào các ngõ vào có độ ưu tiên cao nhất
-Ví dụ: Độ ưu tiên ngõ vào x3 > x2 > x1 > x0
MẠCH GIẢI MÃ (DECODER)
Có n ngõ vào, 2^n (hoặc ít hơn) ngõ ra
-Quy định chỉ có duy nhất một ngõ ra mang giá trị = 1 tại một thời điểm
-Nếu các ngõ vào tạo thành số nhị phân có giá trị = k thì ngõ ra = 1 đó là ngõ thứ k
MẠCH DỒN (MULTIPLEXER - MUX) Còn gọi là mạch chọn dữ liệu
-Chọn n ngõ trong 2^n ngõ vào để quyết định giá trị của duy nhất 1 ngõ ra
-Mạch dồn 2^n– 1 có 2^n ngõ nhập, 1 ngõ xuất và n ngõ nhập chọn
MẠCH TÁCH DEMULTIPLEXER (DEMUX)
-Chọn n ngõ trong 2^n ngõ vào để quyết định giá trị của duy nhất 1 ngõ ra
-Mạch DEMUX 1-2^n có 1 ngõ nhập, 2^n ngõ xuất và n ngõ nhập chọn

-5 THÀNH PẦN CƠ BẢN CỦA MÁY TÍNH: processor (control, datapath), memory, devices(input,output).
-Bộ vi xử lí (CPU): datapath (registers,ALU), control unit, stalling: CPU={registers, ALU, control unit, internal bus}
-Lệnh muốn thực thi: gửi địa chỉ lệnh chứa trong PC _bộ nhớ lệnh_xd toán hạng_đọc thanh ghi chứa toán hạng có địa chỉ tương ứng .
-Xây dựng đường đi dữ liệu (Datapath): xd kiến trúc phần tử cần thiết cho câu lệnh, xây dựng phân khúc, xd hoàn chỉnh datapath cho câu lệnh.
-Tập thanh ghi: 3 ngõ nhận địa chỉ, 1 ngõ ghi, 2 ngõ đọc, 1 tín hiệu điều khiển ghi
-Đơn vị số học: 2 ngõ vào toán hạng, 1 ngõ ra, 1 bit zero, 1 tín hiệu điều khiển
-Thêm 2 thành phần cơ bản: bộ nhớ dữ liệu(1 ngõ nhận địa chỉ ô nhớ, 1 ngõ nhận dữ liệu cần ghi, 1 ngõ dữ liệu đọc,2 tín hiệu điều khiển đọc/ghi), bộ mở rộng dấu(1 nhập, 1 ra)-> 1,J format
-Datapath cho R_format
Tín hiệu đk ALU : 0000(and), 0001(or), 0010(add), 0110(sub),0111(slt), 1100(nor)
-CPU đơn kì thực tế k sd vì thời gian thiện lệnh khác nhau, trùng lắp phần tử
-CPU đa chu kì: thực thi 1 câu lệnh thành nhiều chu kì clock, thời gian thực thi theo giản đồ trạng thái, dùng 1 bộ nhớ chung cho các câu lệnh, thêm vào 1 số thanh ghi chứa dữ liệu/ kq trung gian

MỘT SỐ ĐẲNG THỨC CƠ BẢN

x + 0 = x	x . 0 = 0
x + 1 = 1	x . 1 = x
x + x = x	x . x = x
x + x' = 1	x . x' = 0
x + y = y + x	xy = yx
x + (y + z) = (x + y) + z	x(yz) = (xy)z
x(y + z) = xy + xz	x + yz = (x + y)(x + z)
(x + y)' = x'.y' (De Morgan)	(xy)' = x' + y' (De Morgan)
(x')' = x	

MẠCH TUẦN TỰ
-Khác với mạch tổ hợp, ở mạch tuần tự thì ngõ ra không chỉ phụ thuộc vào giá trị hiện thời của ngõ vào, mà còn phụ thuộc giá trị quá khứ
-Mạch tuần tự có khả năng "ghi nhớ các trạng thái trong quá khứ"
MẠCH LẬT
-Là 1 thành phần cấu thành mạch tuần tự. -Có chức năng lưu trữ 1 bit nhị phân. -Có nhiều loại mạch lật, sự khác nhau ở chỗ số ngõ vào và cách thức các ngõ vào tác động đến trạng thái bit nhị phân
MẠCH TUẦN TỰ ĐỒNG BỘ
LATENCY: thời gian hoàn thành 1 công việc nhất định
THROUGHPUT: số lượng công việc có thể hoàn thành trong 1 khoảng thời gian nhất định
PIPELINE: là throughput trên toàn bộ cv đc giao
Trên tài nguyên k đổi, cv thực hiện song song chạy trong 1 ống pipeline
PIPELINING : lệnh thực hiện gối đầu nhau, tiết kiệm thời gian rỗi giữa các công đoạn, tăng tốc độ xử lí lệnh
BƯỚC THỰC THI LỆNh TRONG MIPS:
+Ifteh: Instruction Fetch, Increment PC(nạp lệnh)
+Dcd: Instruction Decode, Read Registers(giải mã lệnh)
+Exec: (thực thi) +Men-ref: Calculate Address(tính toán địa chỉ toán hạng) +Arith-log: Perform Operation(tính toán số học) +Mem: (lưu chuyển với bộ nhớ)
Load: Read Data from Memory +Store: Write Data from Memory +WB: Write Data Back to Register (lưu dữ liệu vào thanh ghi)
TRỞ NGẠI CỦA PIPELINE:
STRUCTURAL HAZARDS: nhiều lệnh dùng chung 1 tài nguyên tại 1 thời điểm
DATA HAZARD: lệnh sau dùng dữ liệu kq của lệnh trc
CONTROL HAZARD: rẽ nhánh gây ra, lện sau đợi kq rẽ nhánh lệnh trước ->Hiện tượng stalls or bubbles
***Structural hazards #1*:** single memory-> Giải pháp: tạo 2 bộ nhớ đệm Cache Level 1/CPU (L1 instruction Cache and L1 Data Cache),phần cứng phức tạp hơn…
***Structural hazards #2*:** register -> 2 Giải pháp: +RegFile có tốc độ truy cập nhanh (ít hơn ½ t / ALU/ 1 clock)
+Tạo RegFile vs 2 ngõ Read and write độc lập
Control hazard: rẽ nhánh ->Giải pháp: trì hoãn stall đến lúc điều kiện rẽ nhánh được thực hiện-> chèn lệnh rác(or hoãn việc nạp lệnh sạng lệnh kế(trong 2 chu kì clock))
+**optimization 1 (sự tối ưu hóa):** chèn thêm pép so sánh rẽ nhánh đặc biệt tại stage 2(decode)-> quyết định giá trị mới cho thanh ghi (lợi ích: chỉ có 1 lệnh không cần thiết được nạp -> cần 1 no-op là đủ)
+**optimization 2**: tái định nghĩa rẽ nhánh: khi thực thi rẽ nhánh, một lệnh ngay sau lệnh rẽ nhánh sẽ đc thực thi (brand-delay slot: xấu nhất: đặt 1 lệnh no-op vào trong br. Tốt nhất: tìm đc 1 lệnh trc lệnh rẽ nhánh, k ảnh hưởng đến chương trình.)
Giải páp Data hazard:
+Forwarding: Trì hoãn lệnh sub lại sau đó dùng forwarding.
+Load: vị trí lệnh sau 1 load gọi là load delay slot.+ Lệnh dùng kq của load, hardware interlock có thể sẽ hoãn nó đúng 1 chu kì clock (lệnh k liên quan-> hoãn lệnh đó).

-Các intruction trong MIPS có khích thước ntn? -> có kích thước khác nhau, nhỏ nhất là 32 bit, lớn nhất là 64 bit.
-IA-64 là tên gọi của ? -> là tên gọi của 1 kiến trúc bộ lệnh
-Phát biểu nào không đúng về kiến trúc bộ lệnh (KTBL)
+KTBL có thể có nhiều cài đặt khác nhau
+Ktbl là một thành phần quan trọng của KT vi xử lí
+Ktbl của AMD hoàn toàn khác kiến trúc bộ lệnh của Intel
+X86, MIPS là các ktbl
->D (do ktbl là phần mềm, còn kt vi xư li là phần cứng nên không thể là câu b)
Thông số FSB ghi kèm các bộ xử lí là loại bus? -> nối bộ vi xử lí và bộ nhớ chính
-Hệ số nguyên dạng bù 2 có miền giá trị nào? –(2n-1) –> (2n-1 - 1)
-X = 0101 0101 phép toán cho kết quả vẫn là X là?
->((X AND 0Fh) SHL 4) OR ((X AND F0h) SHR 4)
-Sau khi nhân 2 số nguyên có dấu bằng thuật toán Booth, điều gì sẽ xảy ra? ->Thừa số thứ nhất (M) giữ nguyên giá trị, thừa số thứ 2 (Q) mất giá trị
-Trong hệ số chấm động CXD 32 bit, số dương có thể chuẩn hóa lớn nhất là ? ->1.[23 số 1] * 2127
-Rút gọn biểu thức (x + y + z')*(x + y' + z')* (x' + y + z')->Xy + z'
-Multiplexer có 12 ngõ vào cần bao nhiêu cổng? -> 4 cổng
-Ngõ vào bằng 1 tại ngõ k thì ngõ ra tạo thành số nhị phân gia trị k, đó là mạch gì? -> binary encoder
-Trạng thái cấm xảy ra trong mạch nào -> mạch lật RS.
-Trong RS, Q(t+1) = Q(t) = 0 thì giá trị 2 ngõ vào ntn? -> S = 0, R = x.
-Mục đích của pipeline là? -> tăng throughtput
-Control haxard xảy ra khi nào? ->1 lệnh phải chờ kết quả của lệnh khác
-Branch – delay-slot là gì? -> lệnh luôn được thực hiên ngay phía sao lệnh rẽ nhánh, bất chấp kết quả của lệnh ra nhánh
-Kiểm tra delayed branch thực thi trong pipeline, nếu complie không tìm được lệnh tiếp thì làm thế nào? -> chèn lệnh no-op vào
-Trong data hazard, phương pháp Forwarding không giải quyết được khi gặp tình huống nào?? -> lw->add
-Data hazard xảy ra khi -> một lệnh sử dụng kết quả của lệnh trước
-Sắp xếp theo thứ tự tăng dần tốc độ truy cập? CD->HDD->RAM->cache
-Nhóm phần cứng trong bộ nhớ trong? Register, cache, RAM
-Thiết bị tạo bởi công nghệ bán dẫn? -> RAM
-DRAM gồm 3 loại, sắp xếp chúng theo tốc độ xử lí tăng dần? -> SIMM, DIMM, RIMM
-Thiết bị có tốc độ truy xuất nhanh nhất là thanh ghi
-Set-associative, set dùng để chỉ cái gì? Chỉ 1 nhóm các line
-Associative – mapping ưu điểm hơn so với direct là ? ->xác suất cache hit cao

HỆ THỐNG NHẬP XUẤT
-I/O devices : 2 cách tổ chức:
+**port-mapped I/O:** có thể cần instruction riêng cho I/O, sd không gian địa chỉ riêng cho các thiết bị, mỗi thiết bị được gán 1 port
+**memory-mapped I/O:** k cần thêm instruction riêng, dùng chung kgian bộ nhớ, mỗi thiết bị được cấp một vùng địa chỉ, làm việc với thiết bị giống như làm việc với bộ nhớ.
-Giao tiếp với thiết bị: control register, data register
->Cơ chế:
Polling: CPU chủ động kt trạng thái của thiết bị
Interrupt-driven: thiết bị chủ động thông báo trạng thái vs CPU
DMA: giao tiếp không qua CPU
Interrupt: 1 trong 4 loại exception (interrupt, trap, fault, abort)
Interrupt service handler: cách để đến được đoạn code xử lí interrupt : centralized dispatch, vectored dispatch

BỘ NHỚ:
Từ trái sang phải: dung lượng tăng dần_tốc độ giảm dần_giá thành 1 bit giảm dần->Bộ nhớ trong: bộ nhớ Cache L1_Cache L2_bộ nhớ chính
-Phân loại: +PP truy cập: tuần tự (băng từ), trực tiếp(các loại đĩa), ngẫu nhiên (bộ nhớ bán dẫn RAM,ROM), liên kêt(cache)+Kiểu vật lí: bộ nhớ bán dẫn, bộ nhớ từ (HDD, FDD), bộ nhớ quang (CD-ROM,DVD) +Bộ nhớ ngoài: băng từ(magnetic tape), đĩa từ (magnetic disk),đĩa quang (optical disk), flash disk
-Hệ thống nhớ lưu trữ lớn RAID (redundant array of inexpensive disk)
+Xem như 1 ổ logic duy nhất có dung lượng lớn+Dữ liệu được lưu trữ pân tán trên các ổ đĩa vật lí -> truy cập song song +Sd dung lượng dư thừa để lưu trữ thông tin, khôi pục thông tin khi đĩa bị hỏng-> an toàn thông tin-Có 7 loại RAID (0-6).
-Bộ nhớ trong: +Bộ nhớ chính: dạng các module nhớ DRAM(bit lưu trữ trên tụ điện-> cần mạch refresh,cấu trúc đơn giản, dung lượng lớn, speed chậm, rẻ tiền, dùng làm bộ nhớ chính)
*Chương trình đang thực hiện, dữ liệu đang thao tác, tồn tại trên mọi hệ thống, ngăn nhớ đc đánh địa chỉ trực tiếp bởi CPU, dung lượng < kgian địa chỉ bộ nhớ mà CPU quản lí, công nghệ lưu trữ DRAM
*Phân loại Dram: SIMM(cũ, chậm); DIMM(phổ biến); RIMM(mới, n nhất)
+Bộ nhớ đệm: tích hợp trên chip của CPU, sd công nghệ lưu trữ SRAM (bit lưu trữ bằng các flip flop-> thông tin ổn định, cấu trúc phức tạp, dung lượng chip nhỏ, speed nhanh, đắt tiền, dùng làm bộ nhớ Cache)
-Khi đọc 1 ô nhớ từ bộ nhớ, nếu chưa có cache miss: chép ô nhớ đó và 1 số ô nhớ lân cận từ bộ nhớ chính vào cache; nếu có cache hit: đọc từ cache, k cần truy xuất bộ nhớ chính. -Cache là bản copy một pần bộ nhớ chính, dùng công nghệ SRAM, truy xuất cao hơn bộ nhớ chính. -Nguyên lí cơ sở khi truy xuất: temporal locality (cục bộ về thời gian), spatial locality (cục bộ về không gian). CPU (truy xuất từng byte/word)-> cache(truy xuất từng block) -> RAM
-PP ánh xạ: +Direct mapping (ánh xạ trực tiếp)
Vd: bộ nhớ chính =4GB=2^{32}byte-> N=32 bit
dung lượng cache=256KB=2^{18}byte -> dùng 18 bit đánh địa chỉ ô nhớ trong Cache. -1line=1 block=32 byte=2^5 byte->W=5 bit. -Số Line trong Cache= $\frac{2^{18}}{2^5} = 2^{13}$Line -> L=13 bit. -T=N-(L+W)=32-(13+5)=14 bit
Xđịnh (W,L,T)

+Asscociative mapping:
VD: bộ nhớ chính = 4GB -> N=32 bit -1 Line= 1Block= 32 byte= 2^5 -> W= 5 bit -T= N-W=32-5 =27 bit
+Set associcative mapping:
Vd: Bộ nhớ chính= 4GB -> N=32 bit. -Dung lượng Cache=256 KB = 2^{18}byte
1Line=1block =32 byte = 2^5 byte -> W= 5 bit. -Số line trong Cache= $\frac{2^{18}}{2^5} = 2^{13} Line$->L=13 bit
Một set trong cache có 4 Line =2^2Line. -Số set trong Cache = $\frac{2^{13}}{2^2} = 2^{11} set$-> S= 11 bit
T= N-(S+W)=32- (11+5)=16 bit. -XĐ (W,S,T) theo kiểu 4-way associative mapping
-Tham số ảnh hưởng bộ nhớ cache: block size,cache size
-Thuật toán thay thế: Random, FIFO(thay thế line nằm lâu nhất trong Cache), LFU (Line có số lần truy cập ít cùng 1 thời điểm), LRU (Line có thời gian lâu nhất k đc tham chiếu tới, tối ưu nhất)
-Write Policy: +1 line bị thay đổi trong cache, sẽ thực hiện thao tác ghi lên lại RAM : write through, write back(line bị thay thế) +nhiều processor chia sẻ RAM, mỗi processor có cache riêng: bus watching with WT(loại bỏ line khi thay đổi trong 1 cache khác), hardware transparency(tự động cập nhật các cache khác khi line bị 1 cache thay đổi), noncacheable share memory (pần bộ nhớ dùng chung sẽ k đc đưa vào cache)
-Số lượng và loại cache: +Mức cache : L1, L2…..
+Mức thấp : onchip, cao: offchip truy cập qua external bus or bus dành riêng
-Cache trên các bộ xử lý Intel: +80486: 8 KB cach L1 trên chip (on-chip)+Pentium: có 2 level cache L1 trên chip +cache lệnh: 8KB +cache dữ liệu: 8KB+ Pentium 4(2000): có 2 level cache L1 và L2 trên chip
+cache L1: 2 cache, mỗi cache 8KB; kích thước Line = 64byte; 4-way associative mapping
+cache L2: 256KB; kích thước Line = 128byte; 8-way associative mapping