ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC BÁCH KHOA

MÔN HỌC: KIẾN TRÚC MÁY TÍNH (THỰC HÀNH) (CO2008)

Bài tập/Thực hành 6

CHƯƠNG 4 KIẾN TRÚC MIPS: SINGLE CLOCK CYCLE

LỚP THỰC HÀNH L03 – HỌC KỲ 212

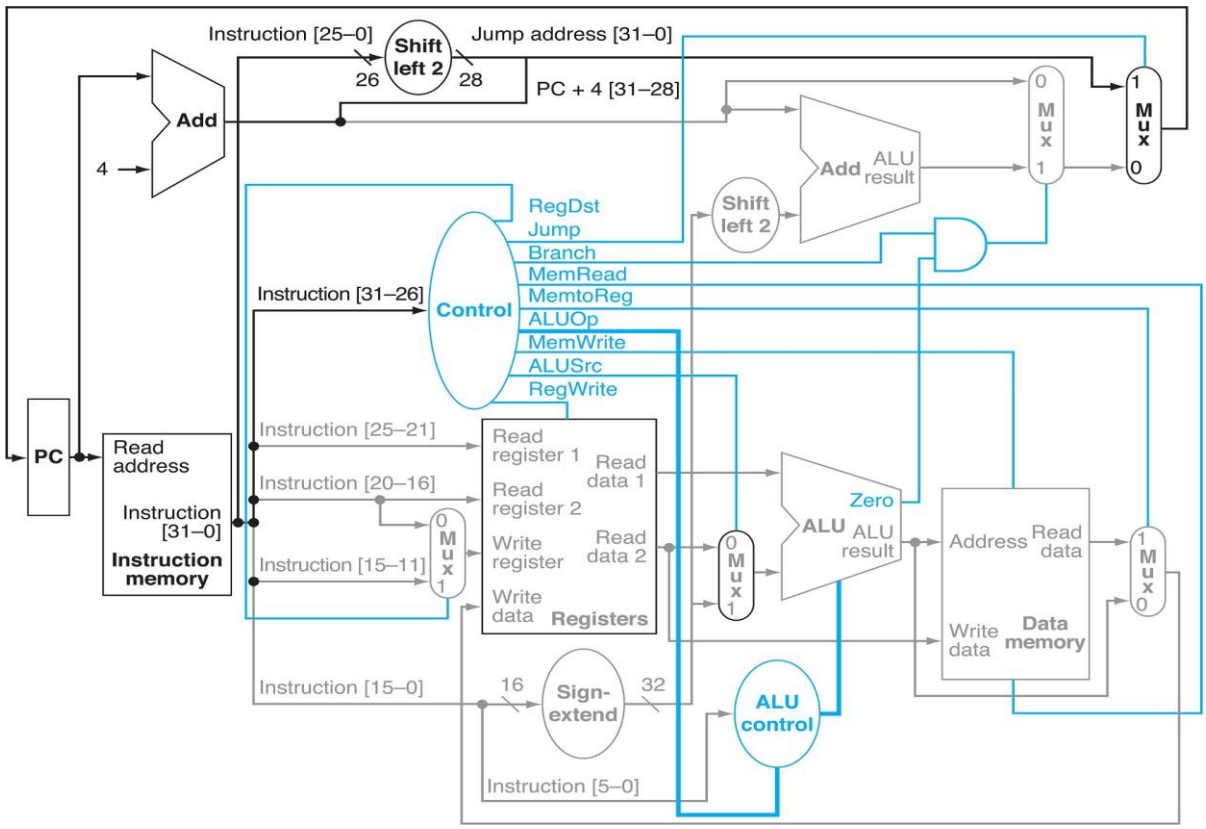
Giảng viên hướng dẫn: Vũ Trọng Thiên

Sinh viên thực hiện	Mã số sinh viên
Phạm Duy Quang	2011899

Thành phố Hồ Chí Minh, tháng 05 năm 2022

Bài tập và Thực hành

Bài 1: Trả lời ngắn gọn các câu hỏi trong hình 1:



Hình 1: Kiến trúc máy tính single clock cycle

- Thanh ghi PC dùng để làm gì.

  Thanh ghi PC dùng để chứa địa chỉ của lệnh chuẩn bị được thực thi.

- Instruction memory chứa gì? input, output là gì?

- Instruction memory là nơi chứa toàn bộ các lệnh của các chương trình đang được thực thi.

- Với input là địa chỉ của câu lệnh cần thực thi, đọc lệnh từ bộ nhớ lệnh, và output là các lệnh MIPS.

- Registers là tập hợp bao nhiêu thanh ghi, input, output là gì?

- Registers chưa tất cả 32 thanh ghi đa dụng của MIPS.

- Các thanh ghi được đánh chỉ số từ 0 đến 31 và được truy xuất thông qua chỉ số của thanh ghi nên cac ngõ vào (Input) là Read register 1 và Read register 2 đều có kích thước 5 bit. Tương ứng với ngõ vào thì ngõ ra cũng có 2 output là Read data 1 và Read data 2. Ngõ vào ra dùng để truy xuất thanh ghi nguồn. Cần thêm input Write Register để thực hiện việc cập nhật giá trị của thanh ghi được điều khiển bởi khối ALU.

- Input và output của ALU là gì?

- Input của ALU chứa giá trị các toán hạng tham gia vào các phép toán được thực hiện bởi khối ALU lần lượt là toán hạng 1 và toán hạng 2.
- Output gồm Result chứa giá trị kết quả của phép toán trên hai toán hạng và Zero = 1 nếu output Result = 0 và ngược lại output Zero = 0.

- Bộ Control nhận input là trường nào? Output dùng để làm gì?

- Bộ control nhận input từ trường opcode 6 bit – là Instruction cụ thể là từ bit [31-26] là output của Instruction memory.
- Output của bộ control là giá trị điều khiển hành vi ALU để yêu cầu khối điều khiển ALU sinh ra tác vụ thích hợp.

- Data memory chứa gì? Input, output là gì?

- Data memory là nơi lưu trữ dữ liệu của các chương trình đang thực thi. Input là ngõ vào đỉa chỉ (Address) chứa địa chỉ của ô nhớ sẽ chứa dữ liệu và dữ liệu đầu vào (Write data) để chưa giá trị của dữ liệu muốn lưu trữ.
- Sau khi giá trị của ô nhớ được truy xuất thì dữ liệu sẽ được truyền vào đầu ra Output là Read data.

- Bộ chọn (MUX) có chức năng gì? Ví dụ.

- Lựa chọn dữ liệu đầu vào hoặc đầu ra thích hợp để thực hiện việc truy xuất dữ liệu ở các bước tiếp theo.
- Ví dụ: Bộ chọn (MUX) sau thanh ghi sẽ được điều khiển bởi khối điều khiển để tùy thuộc vào lệnh đang thực thi sẽ lựa chọn dữ liệu từ khối ALU hay dữ liệu từ bộ nhớ dữ liệu để đưa vào tập thanh ghi.

- Sign-extend dùng để làm gì? Ví dụ.

- Thực hiện việc mở rộng dấu nếu nhận input là 16 bit, thì sẽ thực hiện mở rộng dấu đến 32 bit.
- Ví dụ: Lệnh chứa dữ liệu 16 bit có input = 0xCABA sẽ được mở rộng thành 32 bit là 0x0000CABA.

Bài 2: Các tín hiệu điều khiển sau dùng để làm gì:

- RegDst

- Thanh ghi đích cho thao tác ghi sẽ từ thanh ghi rt (bit từ [20:16]) – khi ở mức thấp (not set).
- Thanh ghi đích cho thao tác ghi sẽ từ thanh ghi rd (bit từ [15:11]) – khi ở mức cao (set).

- RegWrite

- Khối “Registers” chỉ thực hiện mỗi chức năng đọc thanh ghi (not set).

- Ngoài chức năng đọc, khối “Register” sẽ thực hiện thêm chức năng ghi. Thanh ghi được ghi là thanh ghi có chỉ số được đưa vào từ ngõ “Write register” và dữ liệu dùng ghi vào thanh ghi này được lấy từ ngõ “Write data” (set).

- MemRead

- Không làm gì (not set).

- Khối “Data register” thực hiện chức năng đọc dữ liệu. Địa chỉ dữ liệu cần đọc được đưa vào từ ngõ “Address” và nội dung đọc được xuất ra ngõ “Read data” (set).

- MemWrite

- Không làm gì (not set).

- Khối “Data register” thực hiện chức năng ghi dữ liệu. Địa chỉ dữ liệu cần ghi được đưa vào từ ngõ “Address” và nội dung ghi vào lấy từ ngõ “Write data” (set).

- MemtoReg

- Giá trị đưa vào ngõ “Write data” đến từ ALU (not set).

- Giá trị đưa vào ngõ “Write data” đến từ khối “Data memory” (set).

- Branch

- Cho biết lệnh nạp vào không phải “beq”. Thanh ghi PC nhận giá trị là PC + 4 (not set).

- Lệnh nạp vào là lệnh “beq”, kết hợp với điều kiện bằng thông qua cổng AND nhằm xác định xem lệnh tiếp theo có nhảy đến địa chỉ mới hay không. Nếu điệu kiện bằng đúng, PC nhận giá trị mới từ kết quả của bộ cộng “Add” (set).

- Jump

  Nhảy không điều kiện đến label (chứa địa chỉ mới).

- ALUSrc

- Input thứ hai cho ALU đến từ “Read data 2” của khối “Registers” (not set).

- Input thứ hai cho ALU đến từ output của khối “Sign-extend” (set).

Bài 3: Xác định giá trị của các tín hiệu điều khiển.

```
1 lw  $s0, 8($a0)        # load $s0 from memory at address $t2 + 8
2 sw  $s0, 8($a0)        # store $s0 to memory at address $a0 + 8
3 add $s0, $s1, $s2      # add s0 = s1 + s2
4 beq $t2, $t1, label    # branch on equal, if $t2 == $t1 branch to label
5 j   label              # jump to label
```

Trả lời

lw $s0, 8($a0)

Tín hiệu	Giá trị
RegDsT	0
RegWrite	1
ALUSrc	1
ALUOp	00
Nguồn PC	0
MemWrite	0
MemRead	1
MemtoReg	1

sw $s0, 8($a0)

Tín hiệu	Giá trị
RegDsT	x (don't care)
RegWrite	0
ALUSrc	1
ALUOp	00
Nguồn PC	0
MemWrite	1
MemRead	0
MemtoReg	x (don't care)

add $s0, $s1, $s2

Tín hiệu	Giá trị
RegDsT	1
RegWrite	1
ALUSrc	0
ALUOp	10
Nguồn PC	0
MemWrite	0

MemRead	**0**
MemtoReg	**0**

beq $t2, $t1, $label

Tín hiệu	Giá trị
RegDsT	**x (don't care)**
RegWrite	**0**
ALUSrc	**0**
ALUOp	**01**
Nguồn PC	**1**
MemWrite	**0**
MemRead	**0**
MemtoReg	**x (don't care)**

j label

Tín hiệu	Giá trị
RegDsT	**0**
RegWrite	**0**
ALUSrc	**0**
ALUOp	**00**
Nguồn PC	**1**
MemWrite	**0**
MemRead	**0**
MemtoReg	**0**

Bài 4: Xác định critical path, thời gian chu kỳ của hệ thống.

Cho thời gian delay của các khối như bảng bên dưới:

(a) Xác định critical path (longest-latency – Đường đi có độ trễ lâu nhất) và thời gian hoàn thành của các kiểu lệnh sau:

Bảng 1: Delay các khối phần cứng

Resources	Delay
Mux	10ns
Add	10ns
Shift left	10ns
Instruction memory	200ns
Registers	150ns
Sign extend	10ns
ALU	100ns
Data memory	200ns

- Load

Instruction memory → Registers → ALU → Data memory → Mux

= 200 + 150 + 100 + 200 + 10 = 660 ns.

- Store

Instruction memory → Registers → ALU → Data memory → Mux

= 200 + 150 + 100 + 200 + 10 = 660 ns.

- ALU

Instruction memory → Registers → Mux → ALU → Mux

= 200 + 150 + 10 + 200 + 10 = 570 ns.

- Branch

Instruction memory → Registers → Mux → ALU → Mux

= 200 + 150 + 10 + 200 + 10 = 570 ns.

- Jump

Instruction memory → Sign extended → Shift left → Add → Mux

= 200 + 10 + 10 + 10 + 10 = 240 ns.

(b) Xác định thời gian cycle của hệ thống trên.

Gợi ý: Máy tính single clock cycle thực thi 1 lệnh bất kỳ trong một chu kỳ đơn. Xác định thời gian chu kỳ sao cho trong 1 chu kỳ thì đảm bảo lệnh bất kỳ sẽ thực thi xong.

Thời gian để thực hiện 1 lệnh bất kỳ trong một chu kỳ đơn là: 660 ns.