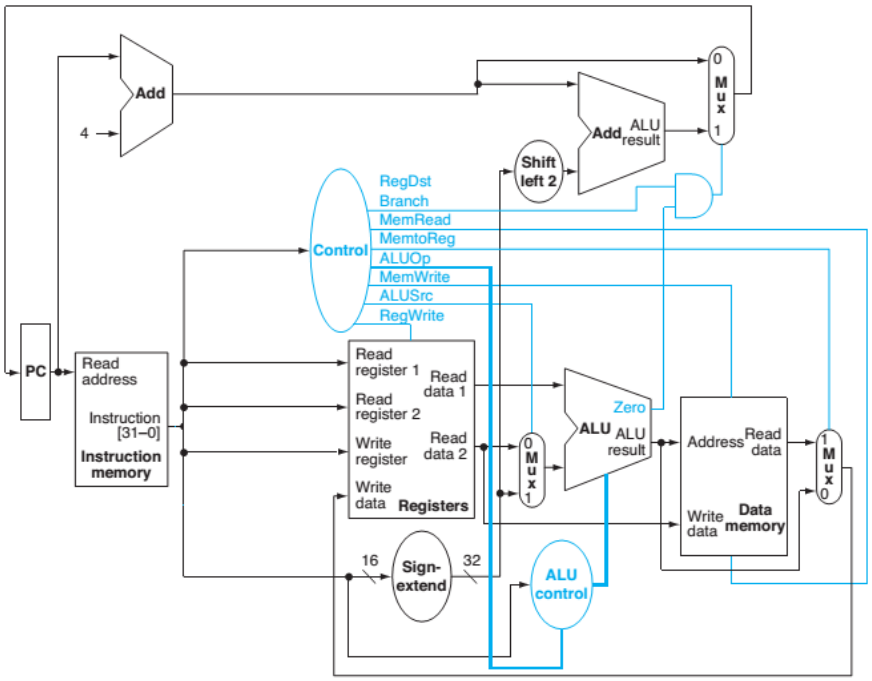
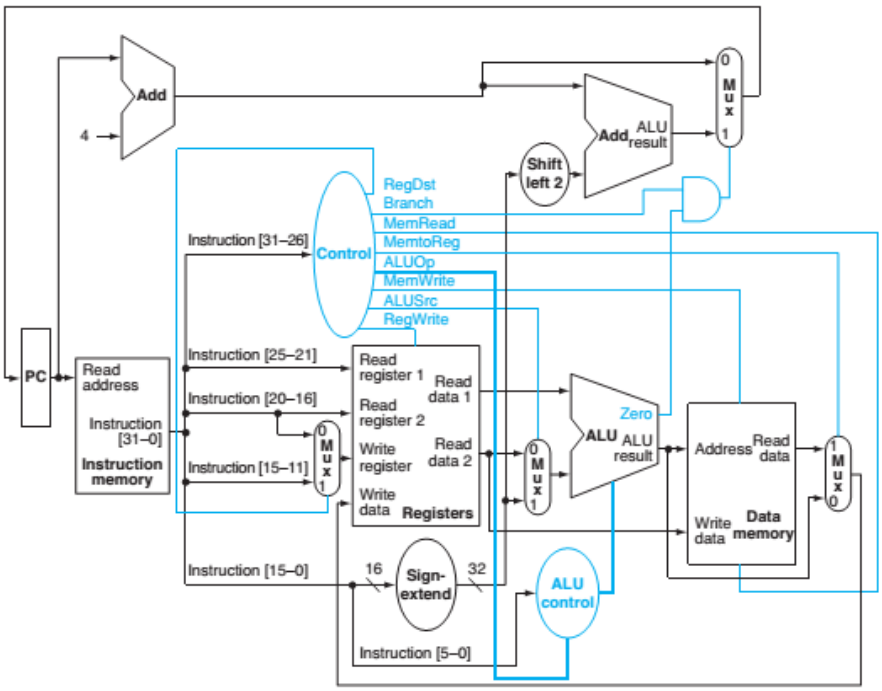
Bài tập chương 4 - Datapath



Hình 1.



Hình 2.

Bài 1. (4.1 – sách tham khảo)

Cho 2 lệnh như sau:

	Lệnh	Ý nghĩa
a.	add rd, rs, rt	Reg[rd] = Reg[rs] + Reg[rt]
b.	lw rt, offs(rs)	Reg[rt] = Mem[Reg[rs]+offs]

Với từng lệnh trong bảng này:

1. Giá trị các tính hiệu điều khiển từ khối "Control" sẽ như thế nào?
2. Các khối nào trong datapath hình 1 cần thiết, khối nào không cần thiết?
3. Khối nào trong datapath hình 1 có output đầu ra, nhưng output này không được sử dụng cho lệnh? Khối nào không có output?

Cho thời gian trễ (thời gian cần để hoàn thành) của từng khối trong hình 1 như sau (khối nào không có trong bảng xem như thời gian trễ bằng 0:

	I-Mem	Add	Mux	ALU	Regs	D-Mem	Control
a.	400ps	100ps	30ps	120ps	200ps	350ps	100ps
b.	500ps	150ps	100ps	180ps	220	1000ps	65ps

4. Tính thời gian trễ lớn nhất của lệnh "and" trong kiến trúc MIPS và cho biết "critical path" của lệnh?

Chú ý: "Critical path" của một lệnh là đường đi có thời gian trễ lớn nhất trong số các đường có thể khi lệnh thực thi.

5. Tính thời gian trễ lớn nhất của lệnh "lw" trong kiến trúc MIPS và cho biết "critical path" của lệnh?
6. Tính thời gian trễ lớn nhất của lệnh "beq" trong kiến trúc MIPS và cho biết "critical path" của lệnh?

---oOo---

Đáp án:

1.

	RegWrite	MemRead	MemWrite	ALUOp	ALUSrc	MemToReg	Branch
a.	1	0	0	Add	0 (Reg)	0 (ALU)	0
b.	1	1	0	Add	1 (Imm)	1 (Mem)	0

2.

a. Tất cả các khối đều được sử dụng, ngoài trừ khối "Data Memory" và bộ cộng dùng cho lệnh nhảy

b. Tất cả các khối đều được sử dụng, ngoài trừ bộ cộng dùng cho lệnh nhảy

3.

	Các khối có output, nhưng không sử dụng	Các khối không có output
a.	bộ cộng dùng cho lệnh nhảy	Data Memory
b.	bộ cộng dùng cho lệnh nhảy	Không (Tất cả các khối đều có output)

4.

a. Độ trễ lớn nhất: 400 + 200 + 30 + 120 + 30 + 200 = 980ps
Critical path: I-Mem, Regs, Mux, ALU, Mux, Regs

b. Độ trễ lớn nhất: 500 + 220 + 100 + 180 + 100 + 220 = 1320ps
Critical path: I-Mem, Regs, Mux, ALU, Mux, Regs

5.

a. Độ trễ lớn nhất: 400 + 200 + 30 + 120 + 350 + 30 + 200 = 1330ps
Critical path: I-Mem, Regs, Mux, ALU, D-Mem, Mux, Regs

b. Độ trễ lớn nhất: 500 + 220 + 100 + 180 + 1000 + 100 + 220 = 2320ps
Critical path: I-Mem, Regs, Mux, ALU, D-Mem, Mux, Regs

6.

a. Độ trễ lớn nhất: 400 + 200 + 30 + 120 + 30 = 780ps
Critical path: I-Mem, Regs, Mux, ALU, Mux

b. Độ trễ lớn nhất: 500 + 220 + 100 + 180 + 100 = 1100ps
Critical path: I-Mem, Regs, Mux, ALU, Mux

---oOo---

Bài 2. (4.2 – sách tham khảo)

Giả sử tập lệnh có thêm hai lệnh mới như sau:

	Lệnh	Ý nghĩa
a.	add3 rd, rs, rt, rx	Reg[rd] = Reg[rs] + Reg[rt] + Reg[rx]
b.	sll rt, rd, shift	Reg[rd] = Reg[rt] << shift (dịch trái shift bits)

Với từng lệnh trên:

1. Khối nào trong hình 1 có thể sử dụng?
2. Khối mới nào cần được thêm vào?
3. Tín hiệu mới nào cần được thêm vào từ khối "Control" để hỗ trợ?

---oOo---

Đáp án:

1.

a. Lệnh này cần sử dụng các khối: instruction memory, Registers (cả 2 cổng đọc và cổng ghi)

b. Lệnh này cần sử dụng các khối: instruction memory, Registers (nhưng chỉ 1 cổng đọc và cổng ghi), khối sign-extend (đường truyền số tức thời tới ALU)

2.

a. Các khối mới cần được thêm vào: Thêm một cổng đọc vào khối Registers và thêm một ALU để tính tổng Rx với Rs + Rt (hoặc sửa ALU đang có thành ALU với 3 input)

b. Các khối mới cần được thêm vào: Đưa thêm tính năng dịch vào ALU hiện tại

3.

a. Các tín hiệu điều khiển mới cần thêm vào:
   - Thêm một tín hiệu điều khiển để điều khiển ALU mới (trong trường hợp ALU câu 2.2.a chọn thêm 1 ALU mới)
   - Thay đổi lại khối "ALU Control" để điều khiển ALU 3 đầu vào (trong trường hợp ALU câu 2.2.a chọn sửa lại ALU 2 đầu vào thành 3 đầu vào)

b. Các tín hiệu điều khiển mới cần thêm vào:
   - Thay đổi lại khối "ALU Control" để điều khiển ALU có thêm tính năng sll

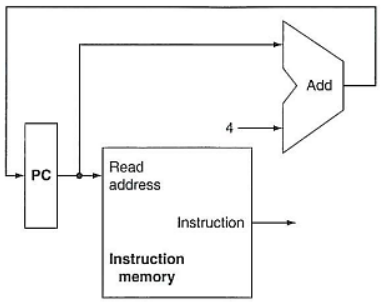
---oOo---

Bài 3. (4.6 – sách tham khảo)

Giả sử các khối trong datapath có độ trễ như sau:

	I-Mem	Add	Mux	ALU	Regs	D-Mem	Sign-Extend	Shift-left-2
a.	400ps	100ps	30ps	120ps	200ps	350ps	20ps	2ps
b.	500ps	150ps	100ps	180ps	220ps	1000ps	90ps	20ps

1. Giả sử việc duy nhất được thực hiện trong processor chỉ là nạp lệnh liên tục (như hình 2), chu kỳ xung clock cần cho thiết kế là bao nhiêu?



2. Giả sử processor chỉ thực hiện duy nhất mỗi lệnh nhảy (như beq nhưng không cần điều kiện bằng), chu kỳ xung clock cần cho thiết kế là bao nhiêu?
3. Như câu 2, nhưng lệnh nhảy trong trường hợp này có xét đến điều kiện bằng (như beq), chu kỳ xung clock cần cho thiết kế là bao nhiêu?

Cho khối chức năng sau:

a.	Add 4 (bộ cộng dùng để cộng PC với 4)
b.	Data Memory

4. Dạng lệnh nào cần các khối chức năng trên
5. Dạng lệnh nào mà các khối chức năng trên nằm trong critical path?

---oOo---

Đáp án:

1.

a. 400ps

b. 500ps

2.

Critical path cho lệnh này: instruction memory, sign-extend, shift-left-2, bộ cộng (để tính địa chỉ mới) và Mux.

a. 400 + 20 + 2 + 100 + 30 = 552ps

b. 500 + 90 + 20 + 150 + 100 = 860ps

3.

Ngoài đường dẫn tính địa chỉ mới cho lệnh nhảy (instruction memory, sign-extend, shift-left-2, bộ cộng, và Mux), còn một đường dẫn khác qua: instruction memory, Registers, Mux, ALU, Mux để tính điều kiện bằng.

Độ trễ của đường dẫn tính điều kiện bằng:

a. 400 + 200 + 30 + 120 + 30 = 780ps
b. 500 + 220 + 100 + 180 + 100 = 1100ps

Vì đường này có độ trễ dài hơn đường tính địa chỉ mới, nên chu kỳ xung clock cần cho thiết kế:

a. 780ps
b. 1100ps

4.

a. Tất cả các lệnh, ngoài trừ các lệnh nhảy thuộc nhóm "not PC-relative" (jal, jalr, j, jr)
b. Các lệnh liên quan đến 'load' và 'store'

5.

a. Không lệnh nào (Vì khối "Instruction memory" luôn có độ trễ cao hơn "Add 4" và tất cả các lệnh (bao gồm cả NOP) đều cần phải qua Instruction memory cho việc đọc lệnh).
b. 'load' và 'store'

---oOo---

Bài 4. (4.7 – Sách tham khảo)

Cho độ trễ của các khối trong datapath như sau:

	I-Mem	**Add**	**Mux**	**ALU**	**Regs**	**D-Mem**	**Sign-extend**	**Shift-left-2**
a.	400ps	100ps	30ps	120ps	200ps	350ps	20ps	0ps
b.	500ps	150ps	100ps	180ps	220ps	1000ps	90ps	20ps

1. Chu kỳ xung clock là bao nhiêu nếu datapath chỉ hỗ trợ các lệnh thuộc nhóm logic và số học (như add, and, …)?
2. Chu kỳ xung clock là bao nhiêu nếu datapath chỉ hỗ trợ lệnh lw?
3. Chu kỳ xung clock là bao nhiêu nếu datapath hỗ trợ các lệnh: add, beq, lw, sw?

Giả sử tỉ lệ các lệnh được thực hiện trong một đoạn lệnh như sau (Processor không pipeline):

	add	**addi**	**not**	**beq**	**lw**	**sw**
a.	30%	15%	5%	20%	20%	10%
b.	25%	5%	5%	15%	35%	15%

4. Bao nhiêu phần trăm chu kỳ xung clock có sử dụng khối "Data memory"?
5. Bao nhiêu phần chu kỳ xung clock có sử dụng khối "Sign-extend"?

---oOo---

Đáp án:

1. Critical path: I-Mem, Regs, Mux, ALU, Mux, Regs
   a. 400ps + 200ps + 30ps + 120ps + 30ps + 200ps = 980ps
   b. 500ps + 220ps + 100ps + 180ps + 100ps + 220ps = 1320ps
2. Critical path: I-Mem, Regs, Mux, ALU, D-Mem, Mux, Regs
   a. 400 + 200 + 30 + 120 + 350 + 30 + 200 = 1330ps
   b. 500 + 220 + 100 + 180 + 1000 + 30 + 220 = 2320ps
3. Đáp án như câu 2, vì lw có critical path dài nhất trong số các lệnh trên.
4. "Data memory" bị truy xuất chỉ với lw và sw
   a. 20% + 10% = 30%
   b. 35% + 15% = 50%
5. Thật sự khối "Sign-extend" đều có tính toán ra một kết quả nào đó trong mỗi chu kỳ, nhưng output của nó chỉ được cần cho các lệnh addi, beq, lw và sw; và bỏ qua với các lệnh còn lại. Vì vậy:
   a. 15% + 20% + 20% + 10% = 65%
   b. 5% + 15% + 35% + 15% = 70%

---oOo---

Bài 5. (4.9 – Sách tham khảo)

	Lệnh
a.	lw $1, 40($6)

b.	label: bne $1, $2, label

1. Mã máy của hai lệnh trên là gì
2. Chỉ số cung cấp cho input "Read register 1", "Read register 2" của khối "Registers" là gì? Các thanh ghi này có thật sự được đọc và được sử dụng không? (Xem datapath hình 2)
3. Chỉ số cung cấp cho output "Write register" của khối "Registers" là gì? Thanh ghi này có thật sự được ghi vào không? (Xem datapath hình 2)

---oOo---

Đáp án:

1.

	Binary	Hexadecimal
a.	100011 00110 00001 0000000000101000	8CC10028
b.	000101 00001 00010 1111111111111111	1422FFFF

2.

	Read register 1	Thật sự được đọc và được sử dụng?	Read register 2	Thật sự được đọc và được sử dụng?
a.	6 ($00110_{(2)}$)	Được đọc, được sử dụng	1($00001_{(2)}$)	Được đọc, nhưng **không** được sử dụng
b.	1($00001_{(2)}$)	Được đọc, được sử dụng	2($00010_{(2)}$)	Được đọc, được sử dụng

3.

	Write register 1	Thanh ghi thật sự được ghi không?
a.	1 ($00001_{(2)}$)	Được
b.	Hoặc là 2 ($00010_{(2)}$) hoặc là 31 ($11111_{(2)}$) (không biết vì tín hiệu RegDst là 'x' trong trường hợp này	Không

---oOo---