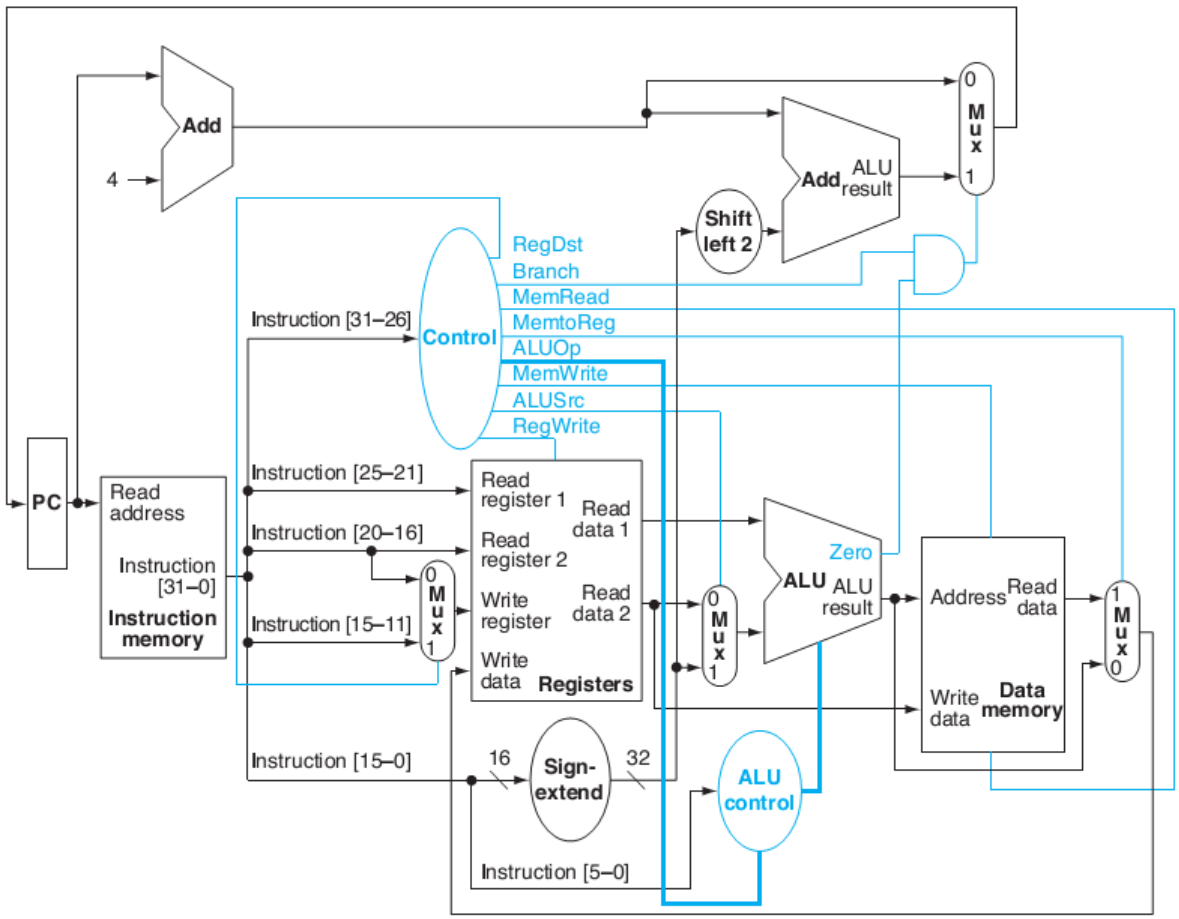


Câu 1.

Cho một kiến trúc máy tính MIPS với datapath và tín hiệu điều khiển như hình.
Đối với lệnh: *addi Rt, Rs, Imm*

a) Lệnh addi chạy được với datapath như trên không? Những khối nào sẽ cần sử dụng cho lệnh trên, khối nào không cần sử dụng?

b) Cho biết giá trị của các tín hiệu điều khiển?

c) Những khối nào có cho dữ liệu output nhưng dữ liệu này không sử dụng? Những khối nào không cho output?

d) Giả sử có lệnh mới như sau "*addi Rt, Rs, Rx, Imm*" *(ý nghĩa Rt = Rs + Rx + Imm)* thì phải thay đổi hay thêm vào hình trên những block nào? (1đ)

Trả lời:

a) Được.
Tất cả các block đều được sử dụng ngoài trừ ***Data memory***

b)

RegDst	Branch	MemRead	MemtoReg	ALUOp	MemWrite	ALUSrc	RegWrite
0	0	0	0		0	1	1

ALU cần thực hiện thao tác cộng.

Giá trị mà ALUOp nên nhận ở đây phụ thuộc vào thiết kế của khối ALU Control (sao cho đảm bảo kết quả của ALU Control ra tương ứng với thao tác cộng)

Nếu theo thiết kế trong sách tham khảo chính thì ALUOp có thể nhận 2 giá trị **00** giống như lw hoặc sw cho lệnh addi này.

c)

- Những block có cho dữ liệu output nhưng dữ liệu này không sử dụng: **Bộ cộng thứ 2** (bộ cộng mà có một input qua khối shift_left_2 trước khi vào bộ cộng)
- Những block không cho output: **Data memory**

d)

- Việc cộng thực hiện trên 3 toán hạng nên: hoặc sử dụng thêm 1 ALU hoặc chỉnh sửa lại ALU đang có bằng cách đưa thêm một input thứ 3 vào
- Trường opcode (6 bits), 3 thanh ghi (mỗi thanh ghi 5 bits) → số bits trống còn lại trong format lệnh trên là 11 bits.
  Vậy trường Imm có thể sử dụng bao nhiêu bits tùy vào thiết kế, nhưng không quá 11 bits này. Gọi n là số bits cho trường Imm
  → Khối **Sign-extend** hiện tại là mở rộng có dấu từ số tức thời 16 bits thành 32 bits; vì vậy hoặc sử dụng thêm một khối Sign-extend với input là n bits hoặc chỉnh sửa khối Sign-extend sao cho có thể nhận cả input 16 bits và n bits

Câu 2.

Một bộ xử lý MIPS 32 bits có datapath như hình và thực thi đoạn chương trình assembly như sau: (Biết khi bắt đầu thanh ghi *$t0 = 0x00000064* và *$t1 = 0x100010FC)*

```
or $t9, $zero, $t0
add $s0, $zero, $t1
sw $t9, 12($s0)
```

a. Giá trị output của khối "Instruction memory" là bao nhiêu khi bộ xử lý trên thực thi ở câu lệnh thứ 3?

b. Khi bộ xử lý trên thực thi ở câu lệnh thứ 3, điền các giá trị cho các thanh ghi, tín hiệu điều khiển và các ngõ input/output của datapath theo yêu cầu của bảng sau:

Ngõ vào/ra		Điều khiển		Kết quả	
Thanh ghi	Giá trị	Tín hiệu	Giá trị	Ngõ	Giá trị
Instruction[25-21]		*RegDst*		*ALUResult (Của ALU)*	
Instruction [20-16]		*RegWrite*		*WriteData (của khối Registers)*	
Instruction [15-11]		*ALUSrc*		*WriteData (Của khối Data Memory)*	
ReadData1		*Branch*			
ReadData2		*MemtoReg*			
		MemWrite			
		MemRead			

Trả lời:

a.

Giá trị output của khối "Instruction memory" khi bộ xử lý trên thực thi ở câu lệnh thứ 3 là mã máy của lệnh "sw $t9, 12($s0)": 0xAE19000C

$$= 10101110000110010000000000001100_{(2)}$$

b.

Ngõ vào		Điều khiển		Kết quả	
Thanh ghi	Giá trị	Tín hiệu	Giá trị	Ngõ	Giá trị
Instruction[25-21]	$16/10000_2$	*RegDst*	X	*ALUResult (của ALU)*	*0x10001108*
Instruction [20-16]	$25/11001_2$	*RegWrite*	0	*WriteData (của khối Registers)*	X
Instruction [15-11]	$0/00000_2$	*ALUSrc*	1	*WriteData (Của khối Data Memory)*	0x64
ReadData1	*0x100010FC*	*Branch*	0		
ReadData2	*0x00000064*	*MemtoReg*	X		
		MemWrite	1		
		MemRead	0		

X tức là bằng 0 hay 1 đều được, không quan tâm vì giá trị này sẽ không đươc sử dụng, không ảnh hưởng đến lệnh đang chạy.

Lưu ý:
Nếu đề hỏi thêm kết quả ngõ ra của bộ cộng nằm sau khối "Shift left 2" là bao nhiêu khi cho biết PC tại lệnh thứ 3, trả lời ALU Result của bộ cộng này = 0xC*4 + PC (tại lệnh thứ 3) + 4

Câu 3.

Một bộ xử lý MIPS 32 bits (có datapath và control như hình) thực thi đoạn chương trình assembly như sau:

addi $t0, $t1, 8
lw $s0, 4($t0)
sw $t0, 4($t0)

Biết khi bắt đầu thanh ghi PC = 0x400000; *$t1 = 0x10010000; $s0 = 0x00000001; word nhớ tại địa chỉ 0x1001000c đang có nội dung (hay giá trị) bằng 0x0000ffff.*
Khi bộ xử lý trên thực thi ở câu lệnh thứ hai, điền các giá trị (tín hiệu, input và output) cho từng khối vào bảng sau:

Tên khối	Ngõ	Giá trị
Instruction Memory	Read address	
	Instruction[31-0]	
Registers	Read register 1	
	Read register 2	
	Write register	
	Write data	
	Read data 1	
	Read data 2	
ALU	Input thứ nhất của ALU	
	Input thứ hai của ALU	
	ALU result	
	Zero	
Data Memory	Address	
	Write data	
	Read data	
Control	Instruction [31-26]	
	RegDst	
	Branch	
	MemRead	

	MemtoReg	
	ALUOp **(Chỉ cần cho biết ALU thực hiện phép toán gì)**	
	MemWrite	
	ALUSrc	
	RegWrite	

Trả lời:

Tên khối	Ngõ	Giá trị **(Sinh viên điền vào cột này)**
Instruction Memory	Read address	0x400004
	Instruction[31-0]	0x8d100004
Registers	Read register 1	$01000_{(2)}$
	Read register 2	$10000_{(2)}$
	Write register	$10000_{(2)}$
	Write data	*0x0000ffff*
	Read data 1	*0x10010008*
	Read data 2	*0x00000001*
ALU	Input thứ nhất của ALU	*0x10010008*
	Input thứ hai của ALU	$4_{(10)}$
	ALU result	0x1001000c
	Zero	0
Data Memory	Address	**0x1001000c**
	Write data	*0x00000001*
	Read data	*0x0000ffff*
Control	Instruction [31-26]	**$100011_{(2)}$**
	RegDst	**0**
	Branch	**0**
	MemRead	**1**
	MemtoReg	**1**
	ALUOp **(Chỉ cần cho biết ALU thực hiện phép toán gì)**	+
	MemWrite	**0**

Có thể thay đổi thứ tự

	ALUSrc	**1**
	RegWrite	**1**

Chú ý:
Nếu đề bài cho thêm hai bảng sau:

ALU control lines	Function
0000	AND
0001	OR
0010	add
0110	subtract
0111	set on less than
1100	NOR

Instruction opcode	ALUOp	Instruction operation	Funct field	Desired ALU action	ALU control input
LW	00	load word	XXXXXX	add	0010
SW	00	store word	XXXXXX	add	0010
Branch equal	01	branch equal	XXXXXX	subtract	0110
R-type	10	add	100000	add	0010
R-type	10	subtract	100010	subtract	0110
R-type	10	AND	100100	AND	0000
R-type	10	OR	100101	OR	0001
R-type	10	set on less than	101010	set on less than	0111

Hỏi cụ thể ALUOp bằng bao nhiêu, output của khối ALU control bằng bao nhiêu, instruction [5-0] bằng bao nhiêu?

Câu 4.

Cho một bộ xử lý MIPS 32 bits (có datapath và control như hình).
Biết *PC = 0x400000; $t1 = 0x00008000; $t3 = 0x00000015; Word nhớ tại địa chỉ 0x00008008 có nội dung/giá trị bằng 0x00000015*
Nếu đoạn chương trình sau được thực thi:

```
        addi $s0, $t1, 4
        lw $t2, 4($s0)
        beq $t3, $t2, ABC
        add $t2, $t3, $t4
ABC: sub $t3, $t4, $t5
```

Khi bộ xử lý trên thực thi **ở câu lệnh thứ ba**, hỏi:

a. Với khối "Instruction Memory" các ngõ "Read address" và "Instruction[31-0]" bằng bao nhiêu
b. Với khối "Registers", các ngõ "Read register 1", "Read register 2", "Write register", "Write data", "Read data 1" và "Read data 2", "RegWrite" bằng bao nhiêu?
c. Với khối "ALU", input thứ 1, input thứ hai, "ALU result" và "zero" bằng bao nhiêu?

d. Với khối "Data Memory", "Address", "Write data", "Read data", "MemWrite", "MemRead" bằng bao nhiêu?
e. Các tín hiệu điều khiển của 3 MUX: RegDst, ALUSrc và MemToReg bằng bao nhiêu?
f. Đầu vào và đầu ra của khối "Sign-extend" bằng bao nhiêu?
g. Đầu vào và đầu ra của khối "Shift left 2" bằng bao nhiêu?
h. Cổng "AND" trong trường hợp này có kết quả bằng bao nhiêu?
i. Ngõ "ALU Result" của bộ "Add" (mà có một đầu vào là kết quả của "Shift left 2") có giá trị bao nhiêu?
j. Thanh ghi PC cuối cùng có giá trị bao nhiêu?

Trả lời:

Khi đoạn chương trình trên chạy tới lệnh thứ 3, tức là lệnh "beq $t3, $t2, ABC", lệnh này khi chạy sẽ được chuyển thành "beq $t3, $t2, 1"

a.

"Read address" có giá trị là giá trị của PC khi tới câu lệnh thứ 3, mà PC ở lệnh thứ 1 khi bắt đầu chạy bằng 0x400000, vậy "Read address" lúc này bằng 0x400000 + 0x8 = **0x400008**
Giá trị ở ngõ ra instruction[31-0] là mã máy của lệnh thứ 3, tức mã máy của "beq $t3, $t2, 1": **0x116a00001 = 0001 0001 0110 1010 0000 0000 0000 0001$_{(2)}$**

b. Với khối "Registers"
Read register 1 nhận giá trị từ Instruction[25-21] = **01 011$_{(2)}$ = 11$_{(10)}$ (chỉ số thanh ghi $t3)**
Read register 2 nhận giá trị từ Instruction[20-16] = **0 1010$_{(2)}$ = 10$_{(10)}$ (chỉ số thanh ghi $t2)**

Vì khối "Registers" lúc này không thực hiện chức năng ghi nên giá trị đầu vào của Write register lúc này có thể là từ Instruction[20-16] hoặc từ Instruction[15-11] tùy vào tín hiệu điều khiển RegDst điều khiển MUX bằng 0 hay 1. Dù giá trị đưa vào là gì thì ngõ Write register cũng không được sử dụng với lệnh beq nên:
Write register = X
Tương tự, **Write data = X**

Read data 1 = nội dung thanh ghi có chỉ số 11, tức nội dung thanh ghi $t3 = **0x15**
Read data 2 = nội dung thanh ghi có chỉ số 10, tức nội dung thanh ghi $t2 = **0x15**
(Sau khi lệnh thứ nhất thực thi: $s0 = 0x8004. Khi lệnh thứ hai thực hiện, lệnh này lấy nội dung của word nhớ tại địa chỉ 4 + $s0 = 0x8008 đưa vào thanh ghi $t2; mà đề cho nội dung word nhớ tại 0x8008 đang bằng 0x15, vậy sau khi lệnh thứ 2 thực hiện, thanh ghi $t2 = 0x15)

RegWrite = 0

c. Input thứ nhất của ALU = Read data 1 = 0x15
Input thứ nhất của ALU = Read data 2 = 0x15
ALU result của ALU = 0 (ALU thực hiện phép trừ hai input)
Zero = 1

d. Address của khối "Data memory" = ALU result = 0 (nhưng không sử dụng)
Write data của khối "Data memory" = Read data 2 của khối "Registers" = 0x15 (nhưng không sử dụng)
Read data = X
MemWrite = 0
MemRead = 0

e. RegDst = X

ALUSrc = 0
MemToReg = X

f. Đầu vào của "Sign-extend" = Instruction [15-0] = **0000 0000 0000 0001$_{(2)}$**
Đầu ra của "Sign-extend" = **0000 0000 0000 0000 0000 0000 0000 0001$_{(2)}$**

g. Đầu vào của "Shift left 2" = **0000 0000 0000 0000 0000 0000 0000 0001$_{(2)}$**
Đầu ra của "Shift left 2" = **0000 0000 0000 0000 0000 0000 0000 0100$_{(2)}$**

h. Cổng "AND" nhận input thứ nhất là tín hiệu "Branch" từ khối "Control"; do lệnh đang thực hiện là lệnh beq nên Branch có giá trị 1. Input thứ hai của cổng "AND" là Zero từ khối ALU; do ALU thực hiện phép trừ có kết quả đang là 0 nên zero cũng đang bằng 1
AND có hai input đều là 1 nên đầu ra bằng 1.

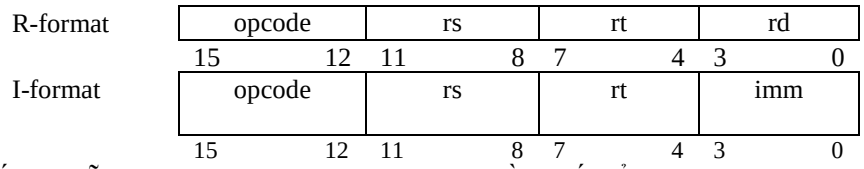
i. Ngõ "ALU Result" của bộ "Add" (mà có một đầu vào là kết quả của "Shift left 2") =
$400008_{(16)} + 4_{(10)} +$ **0000 0000 0000 0000 0000 0000 0000 0100$_{(2)}$**
$= 400008_{(16)} + 4_{(16)} + 4_{(16)}$
$= 400010_{(16)}$

j. PC cuối cùng trong trường hợp này sẽ bằng 0x400010 (Vì cổng AND có kết quả bằng 1 → điều khiển MUX (sau bộ cộng Add) → PC nhận giá trị là ALU result của bộ cộng Add; nên PC = 0x400010)

Câu 5.

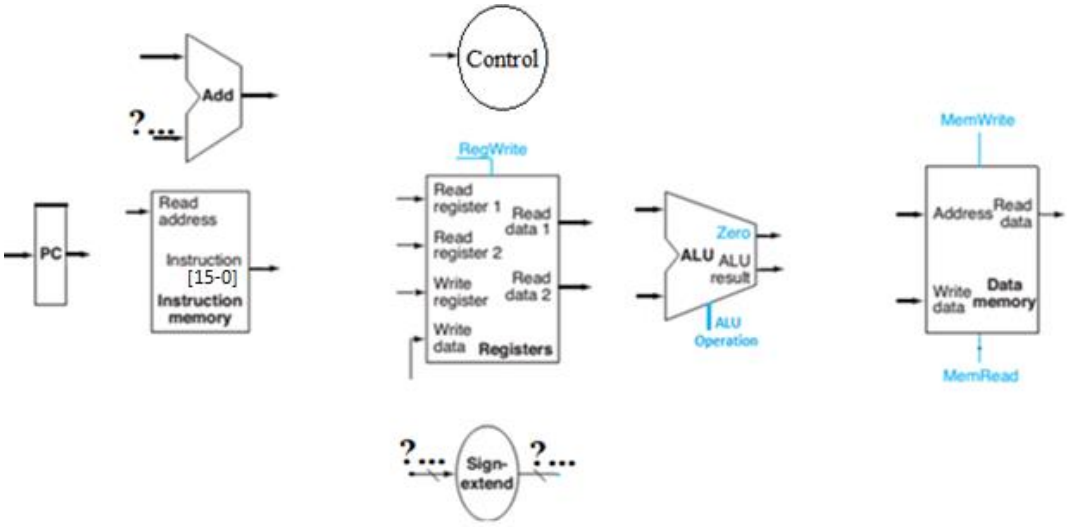
Cho một processor 16 bit có 4 lệnh như sau

Lệnh	Chức năng	Định dạng
add rd, rs, rt	R[rd] = R[rs] + R[rt]	R-format
addi rt, rs, imm	R[rt] = R[rs] + SignExt(imm)	I-format
lw rt, imm(rs)	R[rt] = M[R[rs] + SignExt(imm)]	I-format
bne rs, rt, imm	if(R[rd] != R[rs]) PC = PC + 2 + imm*2	I-format
Lưu ý: rd, rs, rt: thanh ghi imm: số tức thời R[x]: giá tri thanh ghi x M[y]: nội dung của từ nhớ tại địa chỉ y SignExt(imm): mở rộng có dấu số tức thời imm từ 4 bit thành 16 bit		



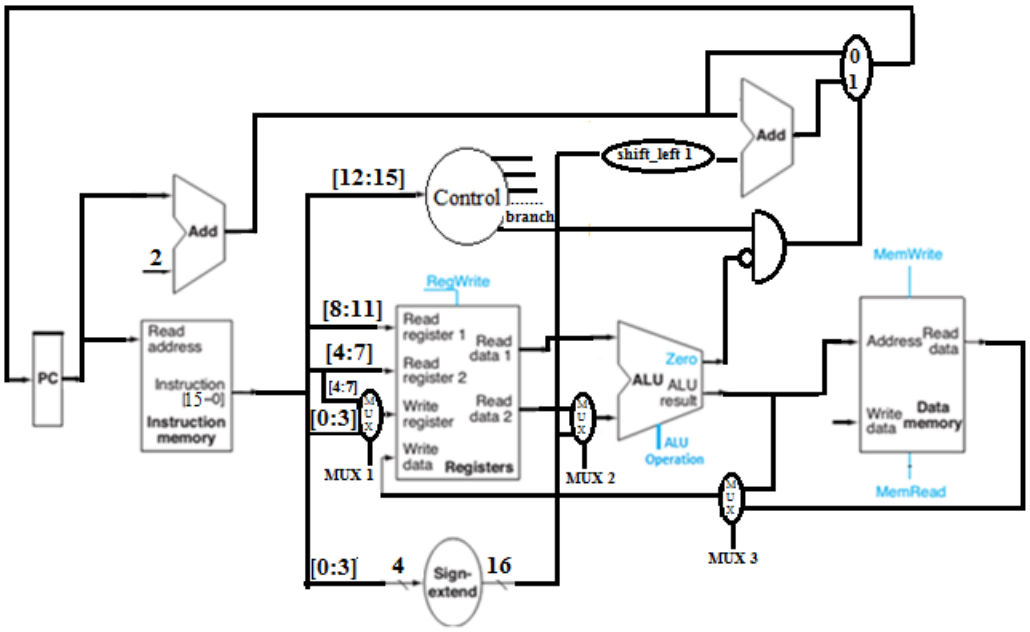
Với các khối cho sẵn như hình 3, vẽ thêm các đường cần thiết để hoàn chỉnh datapath cho processor có tập lệnh trên (Tại mỗi dấu "?" phải điền vào giá trị tương ứng)

- Có thể dùng thêm bộ MUX, bộ cộng, bộ dịch trái/phải và các loại cổng logic nếu cần
- ALU chỉ nhận input đầu vào là số 16 bit



Trả lời:

Đáp án gợi ý:



Đầu ra Control sẽ nối tới ngõ điều khiển của MUX 1, MUX 2, MUX 3, và các ngõ RegWrite, ALUOperation, MemWrite, MenRead.