به نام خدا

درس معماری کامپیوتر
نیم‌سال دوم ۰۴-۰۳
استاد: دکتر اسدی

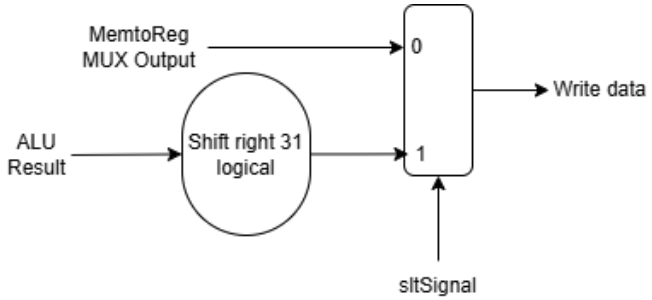
دانشکده مهندسی کامپیوتر

پاسخنامه تمرین سری چهارم

---

۱. (آ) ۱. در ابتدا نیاز به یک سیگنال کنترلی جدید داریم تا این دستور را تشخیص دهد (نام آن را bgeSignal می‌گذاریم) و این سیگنال را با NOT بیت علامت جواب ALU، AND می‌کنیم و آن را با PCSrc، OR می‌کنیم و به عنوان سیگنال select برای MUXای که سیگنال کنترلی آن PCSrc بود، در نظر می‌گیریم.

۲. سیگنال جدیدی با نام sltSignal اضافه می‌کنیم. سپس MUXهای زیر را به Write data در Register File متصل می‌کنیم:



۳. ابتدا یک سیگنال جدید با نام luiSignal اضافه می‌کنیم و آن را به یک MUX جدید که قبل از Write data Register File قرار می‌دهیم که اگر صفر بود مقدار پیش‌فرض و اگر یک بود، Immediate که ۱۶ شیفت چپ خورده است را انتخاب می‌کند. برای این کار به یک بلوک جدید برای ۱۶ شیفت چپ ورودی Immediate خود نیز نیاز داریم.

(ب) جواب این قسمت وابسته به تغییراتی است که اعمال کرده‌اید و برای برخی از سیگنال‌ها ممکن است جواب صحیح دیگری نیز وجود داشته باشد.

۱. bgeSignal = 1, PCSrc = X, ALUSrc = 0, RegWrite = 0, ALU operation = sub, MemWrite = 0, MemRead = 0, MemtoReg = X

۲. sltSignal = 1, PCSrc = 0, RegWrite = 1, ALUSrc = 0, ALU operation = sub, MemWrite = 0, MemRead = 0, MemtoReg = X

۳. luiSignal = 1, PCSrc = 0, RegWrite = 1, ALUSrc = X, ALU operation = X, MemWrite = 0, MemRead = 0, MemtoReg = X

۲. (آ) حداقل زمان چرخه ساعت ماشین باید به اندازه زمان اجرای طولانی‌ترین دستور یعنی LW باشد. بنابراین حداقل زمان چرخه ساعت در این ماشین برابر با ۸ns است.
زیرا اگر زمان چرخه ساعت کمتر از این مقدار باشد، دستور LW زمان کافی برای اجرا نخواهد داشت.

(ب) زمان اجرای برنامه به صورت زیر خواهد بود:

$$ExecTime = CPI \times IC \times ClockCycleTime = 1 \times 4539 \times 8 = 36.312\mu s$$

(ج) مدت زمانی که پردازنده بیکار است به صورت زیر است:

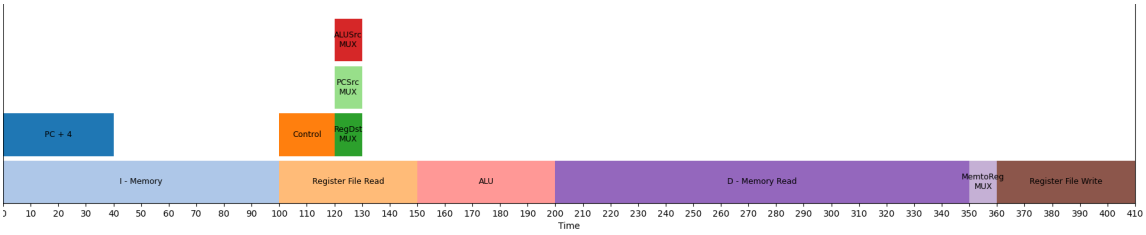
$$[(0.44 \times 2) + (0.24 \times 0) + (0.12 \times 1) + (0.18 \times 3) + (0.02 \times 6)] \times 4539 = 7.53474\mu s$$

$$\frac{7.53474}{36.312} \times 100 = 20.75$$

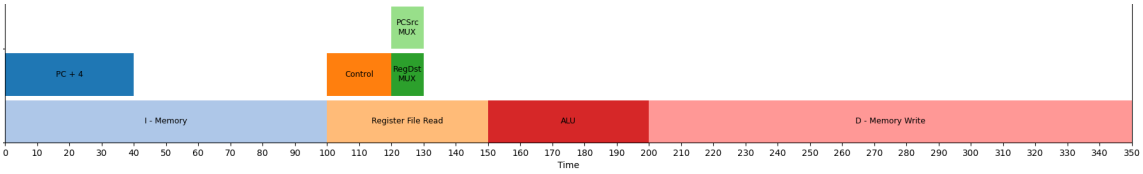
۳. (آ) مسیر بحرانی هر دستور مشخص شده است:

۱. دستورات R-Type:

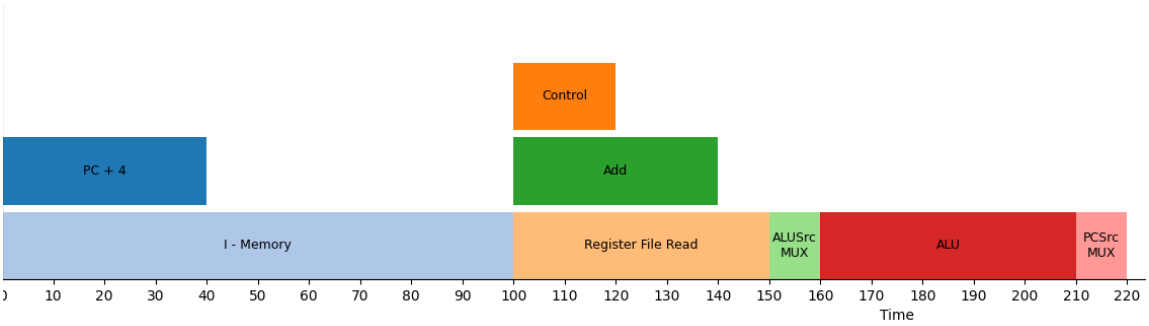
۲. دستور lw:



۳. دستور sw:



۴. دستور beq:



(ب) با توجه به جدول تاخیرات بالا، تاخیر هر دستور به صورت زیر است:

۱. ۲۷۰ps
۲. ۴۱۰ps
۳. ۳۵۰ps
۴. ۲۲۰ps

(ج) با توجه به تاخیرها می‌توان فهمید که مسیر بحرانی توسط دستور lw مشخص می‌شود که تاخیری برابر با ۴۱۰ps دارد.

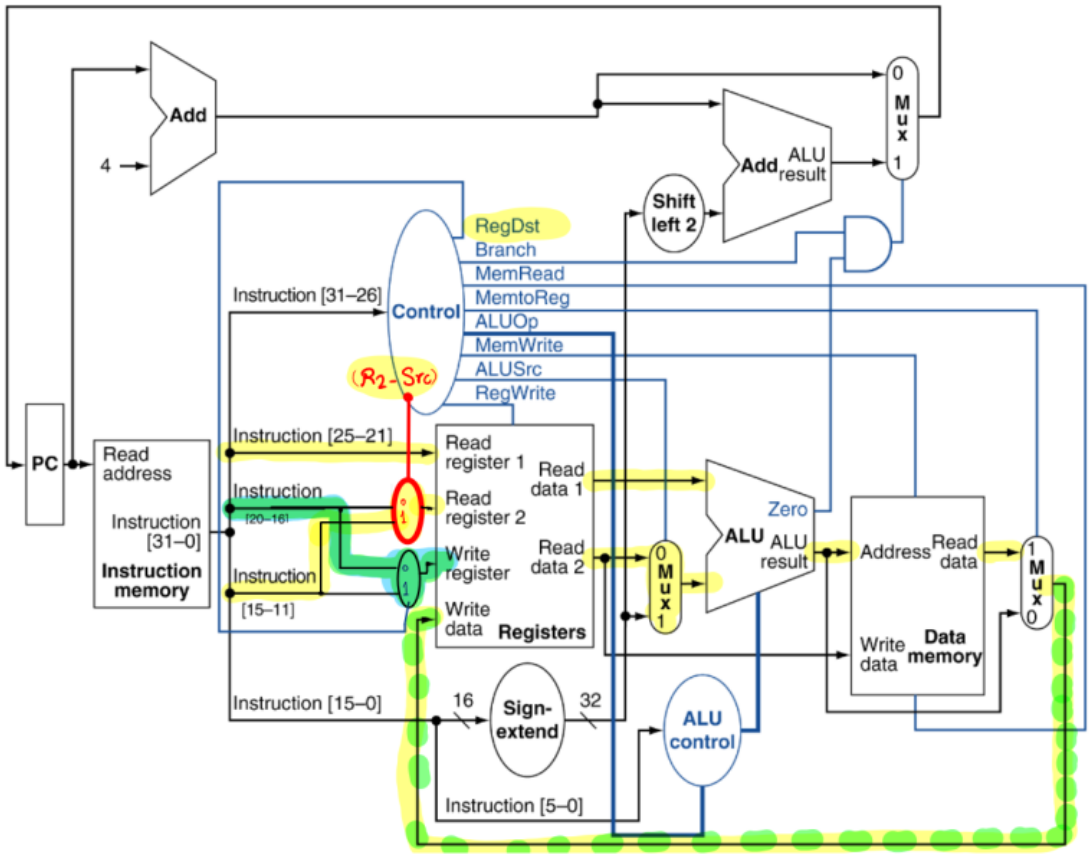
(د) با توجه به اینکه مسیر بحرانی ۴۱۰ps بود پس باید حداقل طول چرخه نیز برابر با ۴۱۰ps باشد و از این می‌توان نتیجه گرفت که طول کلاک باید به صورت زیر محاسبه شود تا بتوان یک چرخه را در یک کلاک پوشش داد.

$$freq = \frac{1}{410ps} = 2.43GHz$$

۴. ۱. دستور lwr:

از آنجا که باید رجیستر rd خوانده شود، کافی است یک مالتی‌پلکسر به Read Register2 اضافه کنیم. تغییر دیگری لازم نیست صرفا باید سیگنال کنترلی‌های مناسب انتخاب کرد.

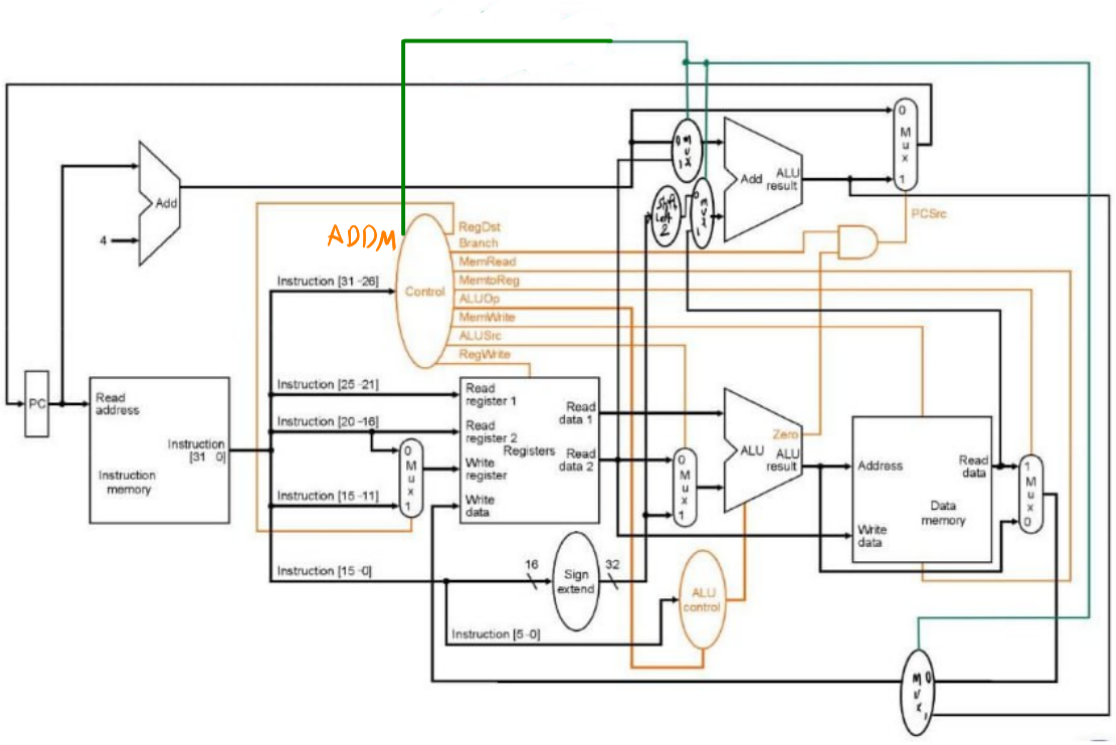
RegWrite = 1, RegDst = 0, Branch = 0, MemRead = 1, MemWrite = 0, ALUSrc = 0, ALUOp = 00 (Add), RegWrite = 1, MemToReg = 1



۲. دستور addm:

یک سیگنال ADDM اضافه می‌کنیم که مشخص می‌کند آیا دستور addm است یا نه. همچنین همچنین عملیات جمع کردن محتویات Reg[rs] و offset و همچنین جمع مقدار موجود در DataMemory[Reg[rs]+offset] با Reg[rt] را توسط همان جمع کننده‌ای انجام می‌دهیم که مقصد بعدی PC را محاسبه می‌کند. این موضوع باعث می‌شود که عملیات‌های جمع در همان یک کلاک انجام شده و single cycle بودن سیستم نقض نشود. همچنین تعداد MUX اضافه شده که در شکل زیر مشخص هستند.

ADDM = 1, RegWrite = 1, RegDst = 1, Branch = 0, MemRead = 1, MemWrite = 0, ALUSrc = 1, ALUOp = 00 (Add), RegWrite = 1, MemToReg = X


ADDM
Add
4
PC
Read address
Instruction [31 0]
Instruction memory
Instruction [31 -26]
Control
RegDst
Branch
MemRead
MemtoReg
ALUOp
MemWrite
ALUSrc
RegWrite
Instruction [25 -21]
Instruction [20 -16]
Instruction [15 -11]
Instruction [15 -0]
Instruction [5 -0]
Read register 1
Read register 2
Registers
Write register
Write data
Read data 1
Read data 2
16
Sign extend
32
ALU control
Add
ALU result
Shift left 2
PCSrc
ALU
Zero
ALU result
Address
Read data
Data memory
Write data
M u x

۵. تاخیر دستور Jump:

$$OLDD_J = 250 + 30 + 20 = 300ps$$

$$NEWD_J = 250 + 45 + 20 = 315ps$$

تاخیر دستور Branch:

$$OLDD_B = 250 + 80 + 20 + 100 + 20 + 20 = 490ps$$

$$NEWD_B = 250 + 80 + 20 + 90 + 20 + 20 = 480ps$$

تاخیر دستور LOAD:

$$OLDD_L = 250 + 80 + 100 + 300 + 20 + 80 = 830ps$$

$$NEWD_L = 250 + 80 + 90 + 255 + 20 + 80 = 775ps$$

تاخیر دستور STORE:

$$OLDD_S = 250 + 80 + 100 + 300 = 730ps$$

$$NEWD_S = 250 + 80 + 90 + 255 = 675ps$$

تاخیر دستور R-Type:

$$OLDD_R = 250 + 80 + 20 + 100 + 20 + 80 = 550ps$$

$$NEWD_R = 250 + 80 + 20 + 90 + 20 + 80 = 540ps$$

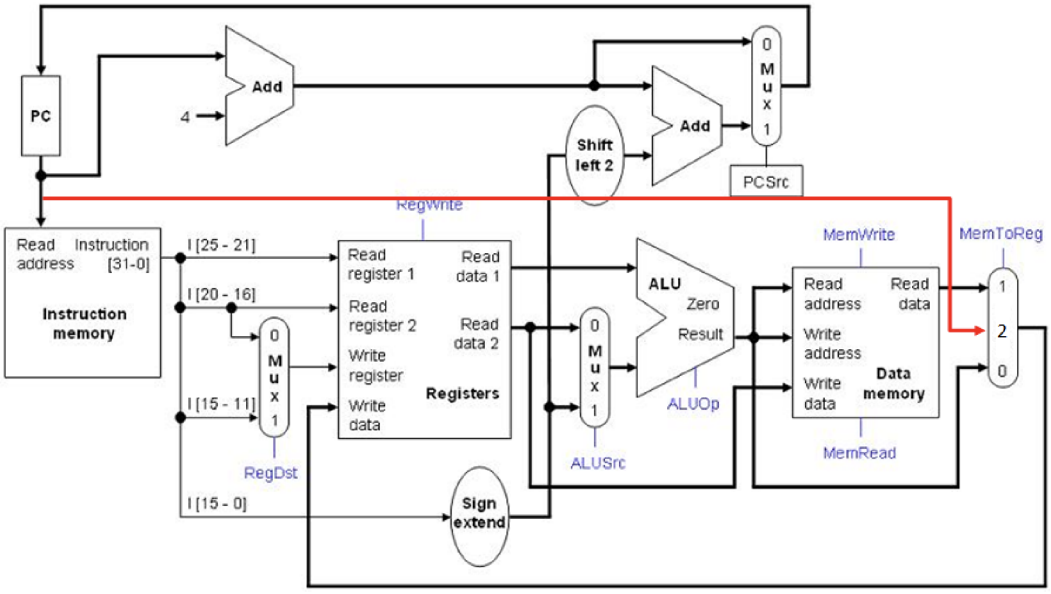
با توجه به اینکه در هر دو حالت LOAD مسیر بحرانی را مشخص می‌کند پس می‌توان میزان Speed Up را به صورت زیر محاسبه کرد:

$$SpeedUp = \frac{Time_{Old}}{Time_{New}} = \frac{830}{775} \approx 1.071$$

۶. (آ) سیگنال های کنترلی به صورت

RegDst = PCSrc = MemRead = MemWrite = 0, RegWrite = 1, ALUSrc = ALUOp = X, MemToReg = 2

می باشد و نیاز به یک مالتی‌پلکسر با ورودی بیشتر است که در شکل نمایش داده شده است:



(ب) از آنجا که چرخه ساعت توسط کندترین دستور مشخص می شود و در این سوال نیز مربوط به دستور lw است پس داریم:

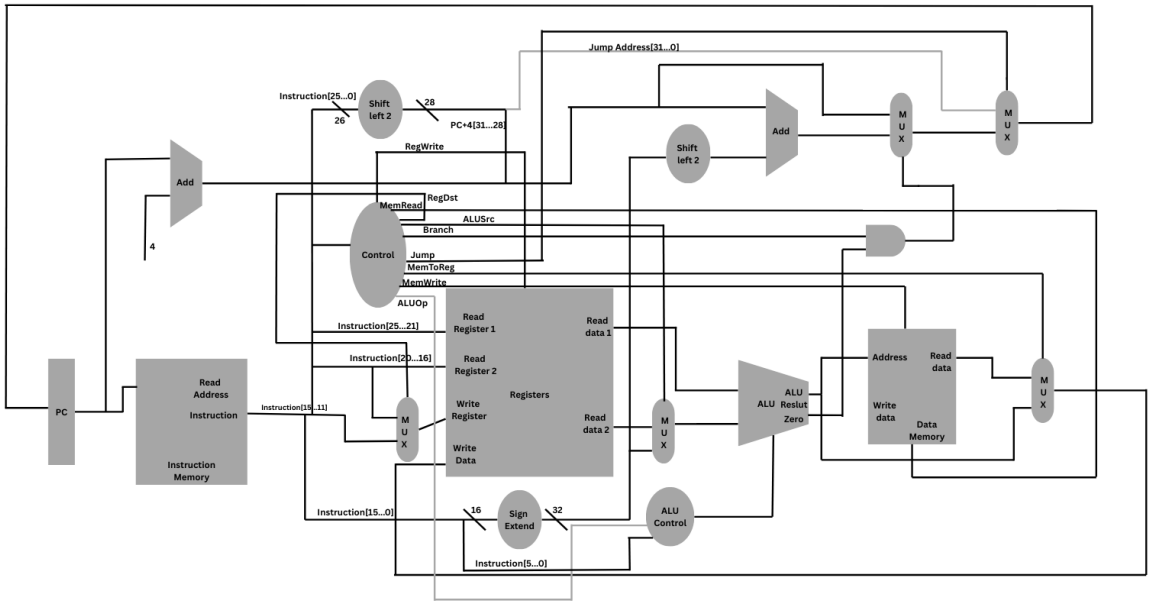
$$t_{sw} = 6(IF) + 3(RF) + 4(EX) + 10(MEM) + 1(MUX) + 3(RF) = 27ns$$

در نتیجه برای فرکانس ساعت داریم:

$$f = \frac{1}{27} \times 10^9 Hz$$

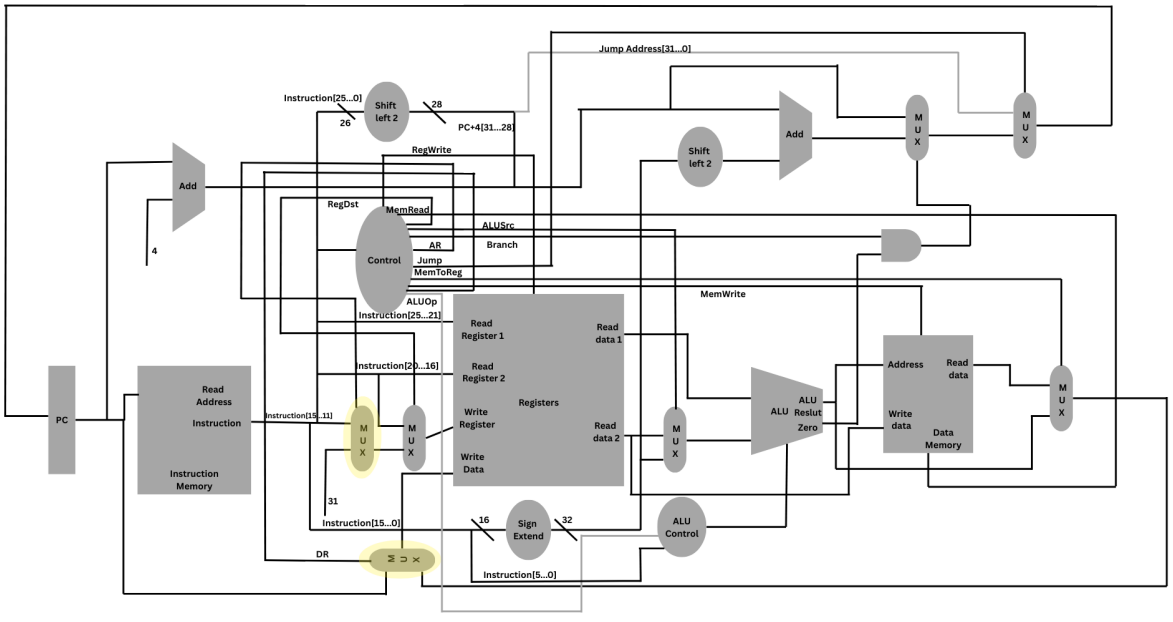
(ج) واحد ALU بهبودیافته باعث می‌شود تاخیر مسیر بحرانی یک نانوثانیه کاهش یابد اما استفاده از بانک ثبات بهبویافته باعث می‌شود تاخیر مسیر بحرانی دو نانوثانیه کاهش یابد (یک نانوثانیه در عملیات خواندن و یک نانوثانیه در علمیات نوشتن) و بدین ترتیب می‌توان نتیجه گرفت استفاده از واحد ALU بهبودیافته گزینه بهتری است.

۷. توجه: در پاسخ این سوال صرفا برای فهم بیشتر، شکل پردازنده تغییریافته رسم شده و کشیدن آن لزومی نداشته و تنها توضیح تغییرات نیز کفایت می‌کند.
شکل تقریبی پردازنده اصلی قبل از اعمال تغییرات به شکل زیر است:

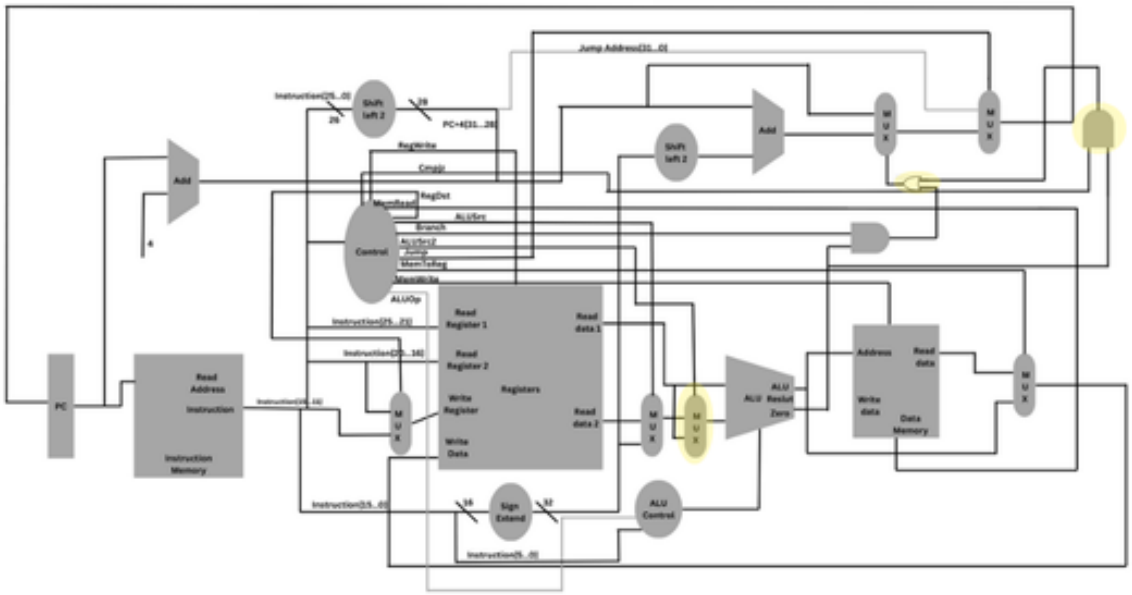


ابتدا تغییرات اعمال شده در پردازنده را بررسی میکنیم و سپس به تاخیر هر دستور میپردازیم:

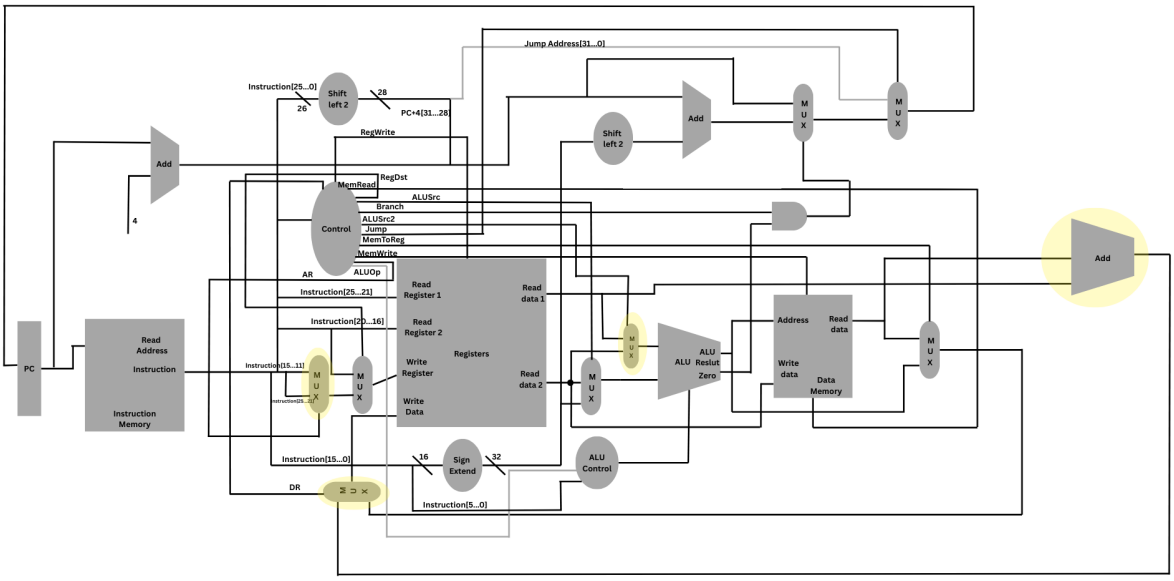
jal:



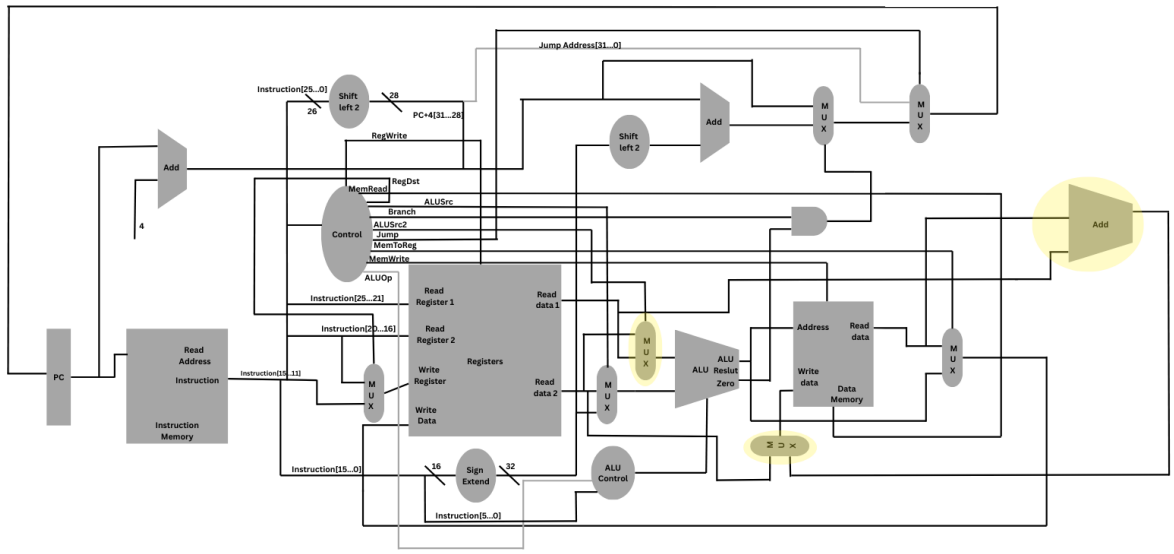
cmpjz:



addm:


Jump Address[31...0]
Instruction[25...0]
Shift left 2
26
28
PC+4[31...28]
RegWrite
Add
4
RegDst
MemRead
ALUSrc
Branch
ALUSrc2
Jump
MemToReg
MemWrite
Control
AR
ALUOp
Instruction[25...21]
Instruction[20...16]
Instruction[15...11]
Instruction[15...0]
DR
Read Register 1
Read Register 2
Registers
Write Register
Write Data
Read data 1
Read data 2
PC
Read Address
Instruction
Instruction Memory
Sign Extend
16
32
ALU Control
ALU
ALU Reslut
Zero
Address
Read data
Write data
Data Memory
MUX
Instruction[5...0]

las:



محاسبه تاخیر هر دستور:

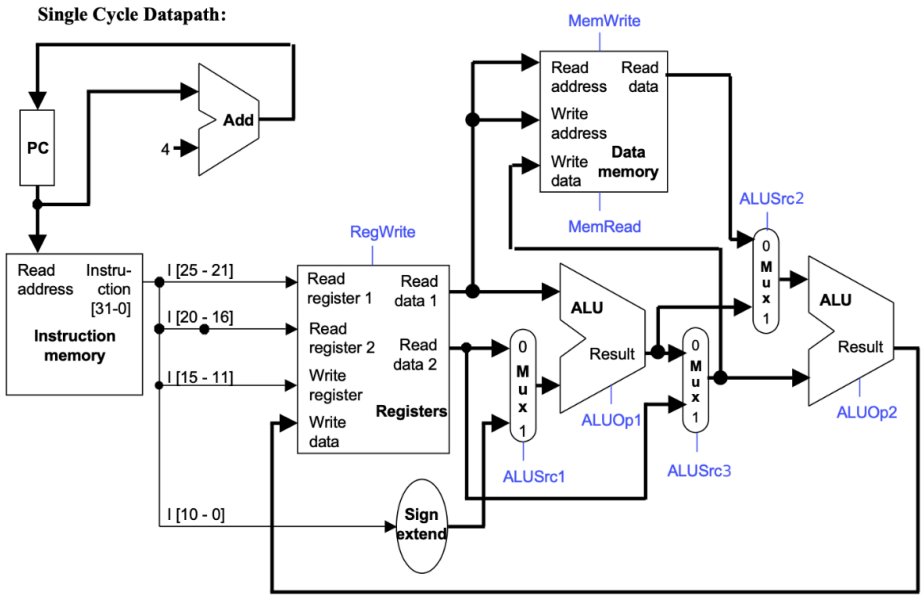
jal:
Instruction-Memory → Control → Registers ⟶ Delay= 340ps

cmpjz:
Instruction-Memory → Registers → MUX → MUX → ALU → AND → OR → MUX → MUX ⟶ Delay= 475ps

addm:
Instruction-Memory → Registers → MUX → ALU → Data-Memory → Add → MUX → Registers ⟶ Delay= 795ps

las:
Instruction-Memory → Registers → MUX → ALU → Data-Memory → Add → MUX → Data-Memory ⟶ Delay= 935ps

۸. الف) باید یک مالتی‌پلکسر اضافه کرد تا قابلیت نوشتن نتیجه‌ی اولین ALU در حافظه اضافه شود. به این صورت:



inst	ALUsrc۱	ALUsrc۳	ALUsrc۲	ALUop۱	ALUop۲	MemRead	MemWrite	RegWrite
addi_st	۱	۰	X	ADD	X	۰	۱	۰

ب)

inst	Minimum time	Explain
lw_add	۱۴ns	IMEM (۳ns) + RF_read (۲ns) + DMEM (۳ns) + ALU (۴ns) + RF_write (۲ns)
addi_st	۱۲ns	IMEM (۳ns) + RF_read (۲ns) + ALU (۴ns) + DMEM (۳ns)
sll_add	۱۵ns	IMEM (۳ns) + RF_read (۲ns) + ALU (۴ns) + ALU (۴ns) + RF_write (۲ns)