معماری کامپیوتر (۱۳۹۱-۱۱-۱۳)

جلسه‌ی سیزدهم

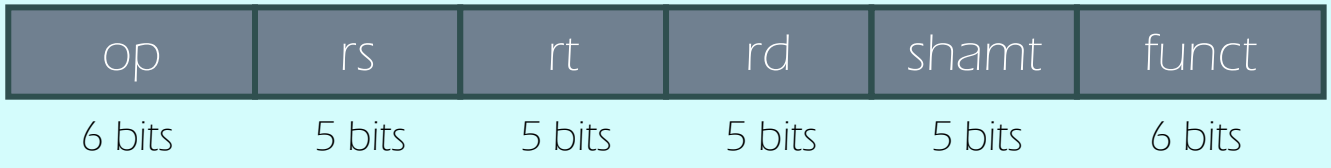
دانشگاه شهید بهشتی
دانشکده‌ی مهندسی برق و کامپیوتر
بهار ۱۳۹۱
احمد محمودی ازناوه

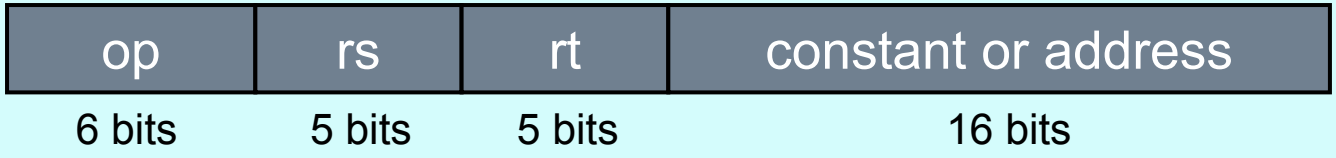
فهرست مطالب

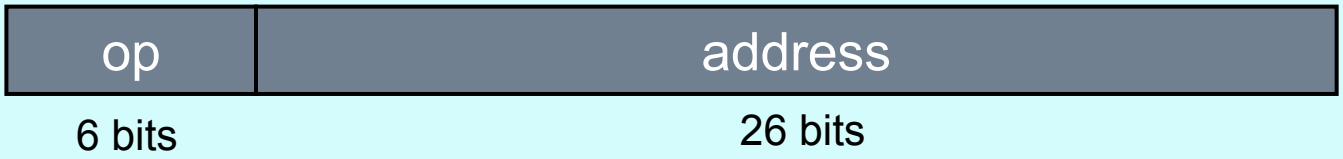
- مروری بر جلسه‌ی پیش
  - نحوه‌ی اجرای یک دستورالعمل
  - مسیر گذار داده
- واحد کنترل

پیش‌گفتار (ادامه...)

- در این بخش یک پیاده‌سازی ساده‌سازی شده از پردازنده‌های MIPS ارائه خواهد شد. که شامل دستورات زیر می‌باشد:
  - Memory reference: `lw`, `sw`
  - Arithmetic/logical: `add`, `sub`, `and`, `or`, `slt`
  - Control transfer: `beq`, `j`

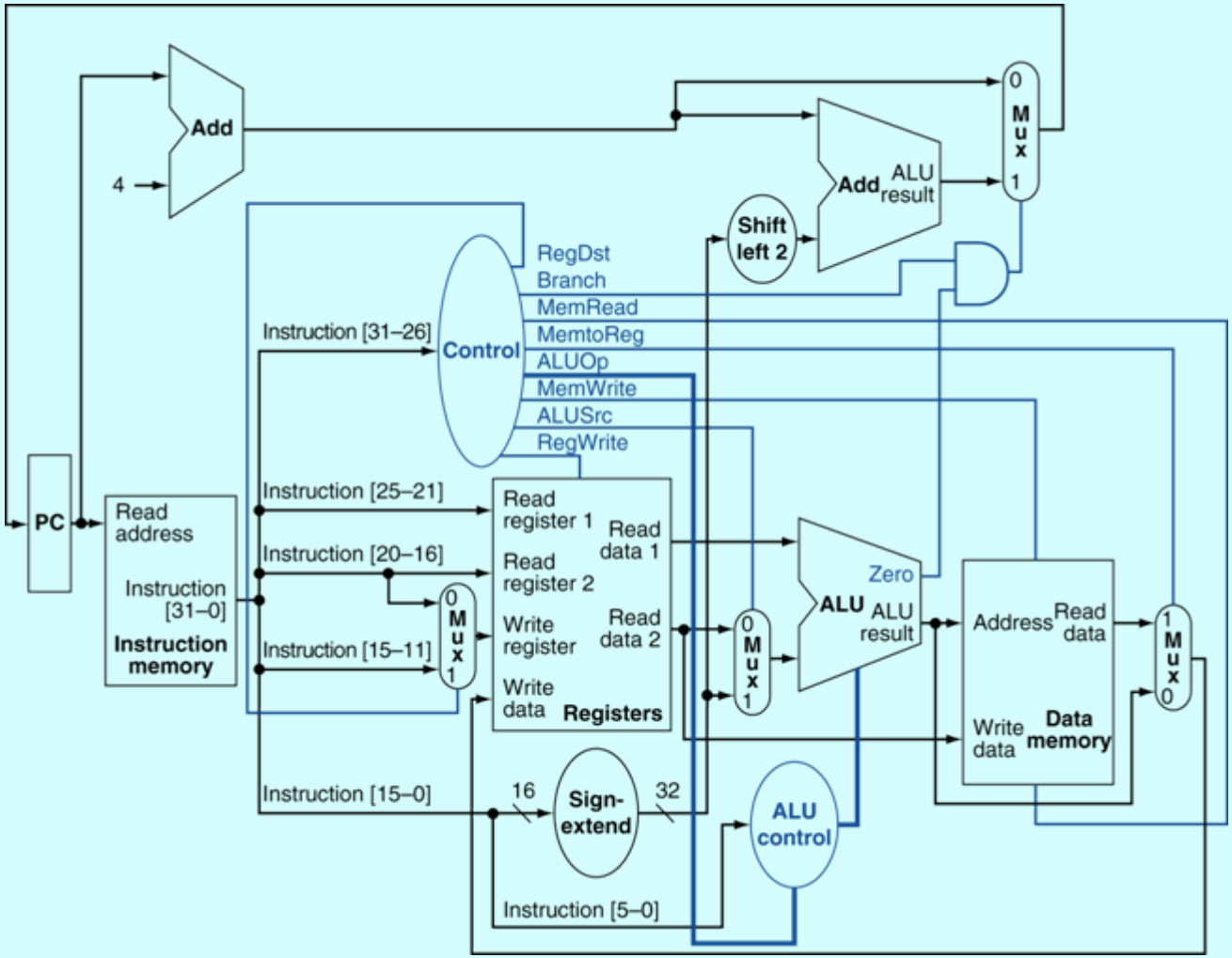






داده‌گذر همراه با واحد کنترل

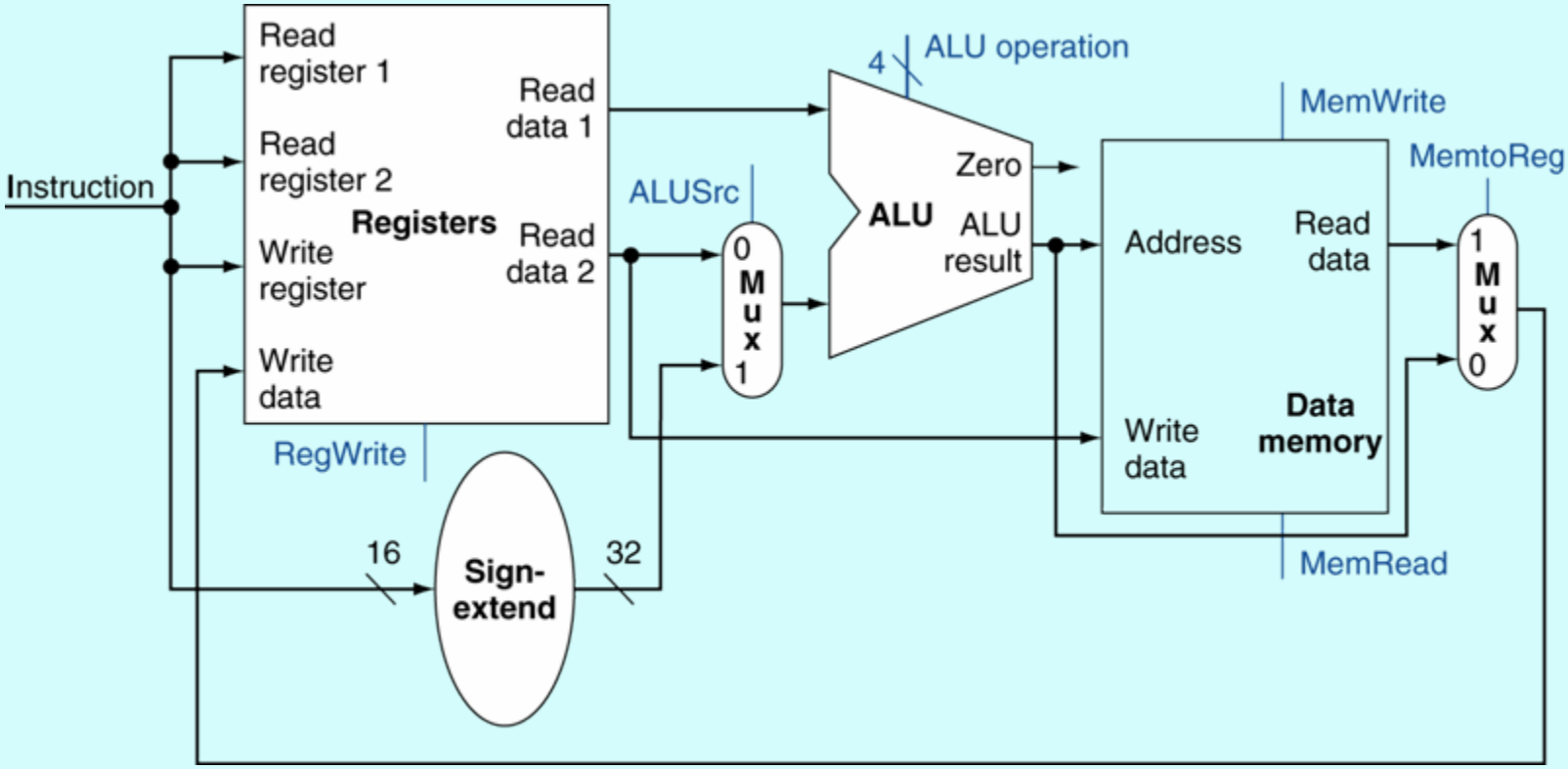
- با توجه به قالب دستور، می‌توان برچسب برخی سیگنال‌های داده و کنترلی را مشخص نمود:



ترکیب اجزا

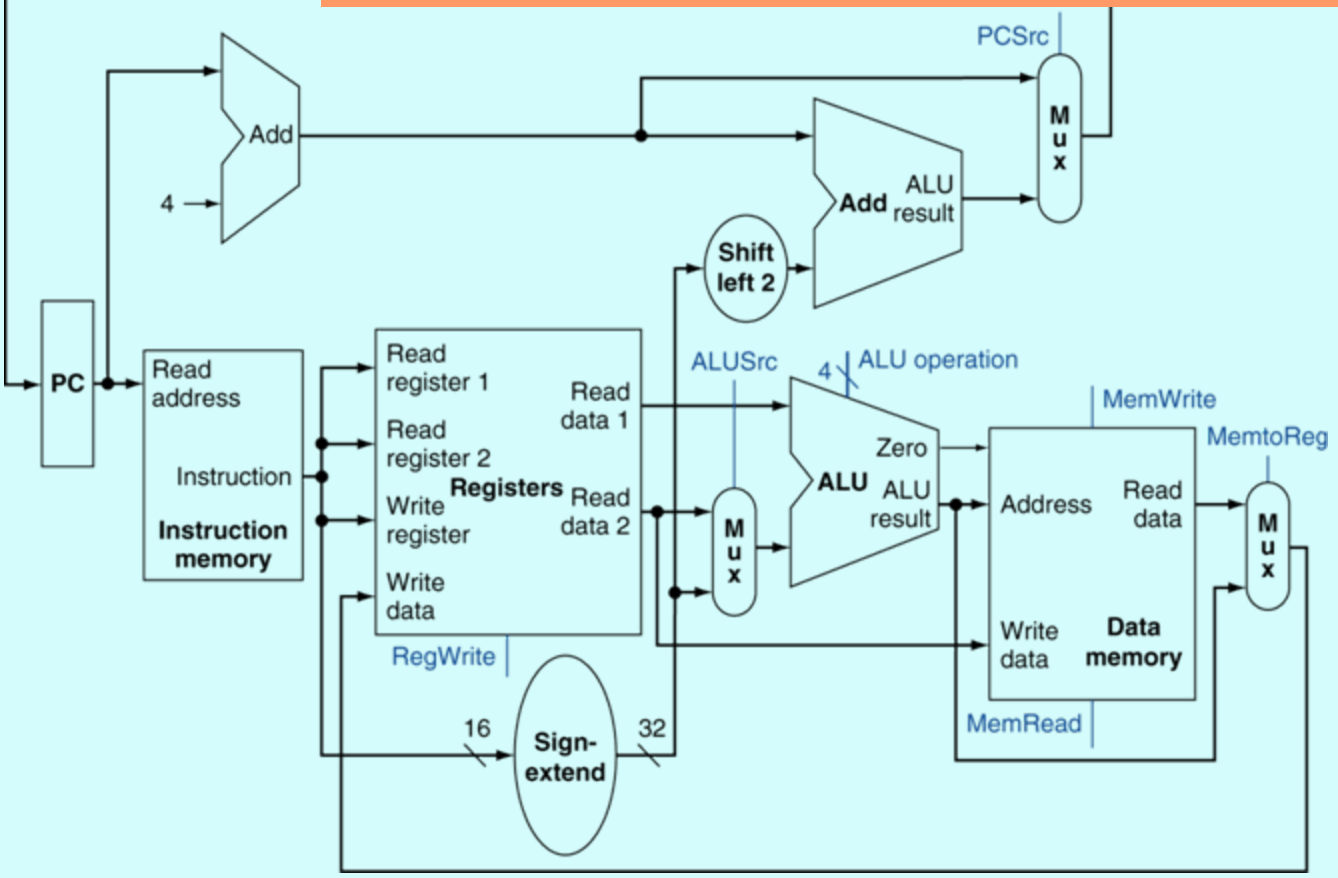
- با ترکیب اجزای مختلف، که برای دستورهای متنوع لازم هستند، یک «مسیر گذار داده» ساخته خواهد شد، که برای دستورهای مختلف توسط واحد کنترل هدایت می‌شود.
- ساده‌ترین داده گذر تمام دستورات را در یک سیکل اجرا خواهد کرد، در این صورت هیچ منبعی را نمی‌توان دوبار استفاده کرد.
- در این حالت باید از برخی منابع چند نسخه قرار داد.
- جاهایی که بیش از یک ورودی داریم، برای انتخاب از مالتی‌پلکسر استفاده می‌کنیم.

ترکیب داده‌گذر دستورات حسابی و منطقی و دستورات حافظه



داده‌گذر کامل شده

ساخت‌افزار مورد نیاز برای اجرای دستورالعمل‌ها آماده است، تنها بخشی که باقی می‌ماند، آماده کردن سیگنال‌های کنترلی است



سؤال ۱: هر کدام از این سیگنال‌ها، برای چه دستوراتی می‌باید فعال باشند؟

سؤال ۲: چرا باید از دو حافظه‌ی مستقل استفاده کنیم؟؟

واحد کنترل ALU

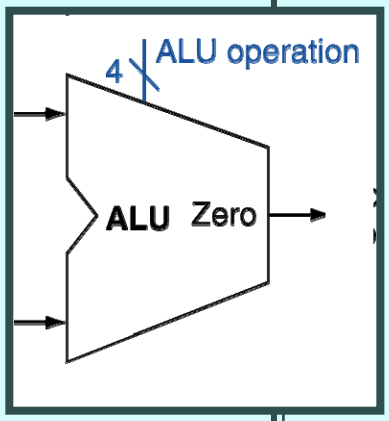
- در ادامه زیربخش کوچکی از دستورهای پردازنده‌ی MIPS را پیاده‌سازی خواهیم کرد.
- به داده‌گذر معرفی شده در بخش قبل، یک واحد کنترل برای اجرای دستورات زیر اضافه می‌کنیم:

- lw, sw
- beq
- arithmatic-logical instruction
  - add, sub, AND, OR, set on less than

- در ادامه، «دستور ل» را هم به این طرح ساده خواهیم افزود.

خطوط کنترلی ALU

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



برای دستورات مختلف از خطوط کنترل ALU استفاده می‌کنیم:

- Load/Store: F = add
- Branch: F = subtract
- R-type: F depends on funct field

واحد کنترل ALU

- ورودی‌های واحد کنترل ALU:
  - دو بیت‌ورودی به نام ALUOp دارد
  - بخش funct دستورالعمل‌های نوع R

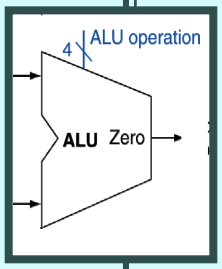
opcode	ALUOp	Operation	funct	ALU function	ALU control
lw	00	load word	XXXXXX	add	0010
sw	00	store word	XXXXXX	add	0010
beq	01	branch equal	XXXXXX	subtract	0110
R-type	10	add	100000	add	0010
		subtract	100010	subtract	0110
		AND	100100	AND	0000
		OR	100101	OR	0001
		set-on-less-than	101010	set-on-less-than	0111

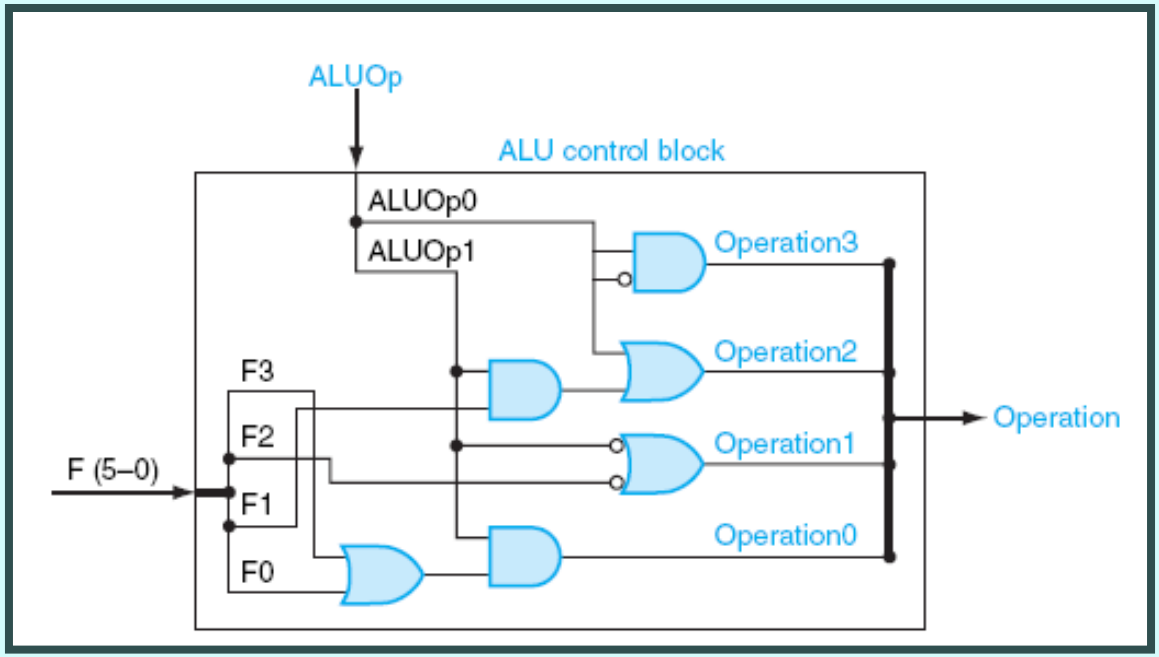
واحد کنترل ALU (ادامه...)

- استفاده از «واحد کنترل چندسطحی»، باعث کوچک شدن واحد کنترل اصلی می‌شود.
- چنین کاری می‌تواند باعث افزایش سرعت واحد کنترل شود.
  - سرعت واحد کنترل در تعیین سرعت پالس ساعت سیستم اهمیت دارد.
- در گام بعد، واحد کنترل ALU ساخته خواهد شد. به نظر شما بهترین شیوه‌ی پیاده‌سازی چیست؟

جدول درستی واحد کنترل ALU

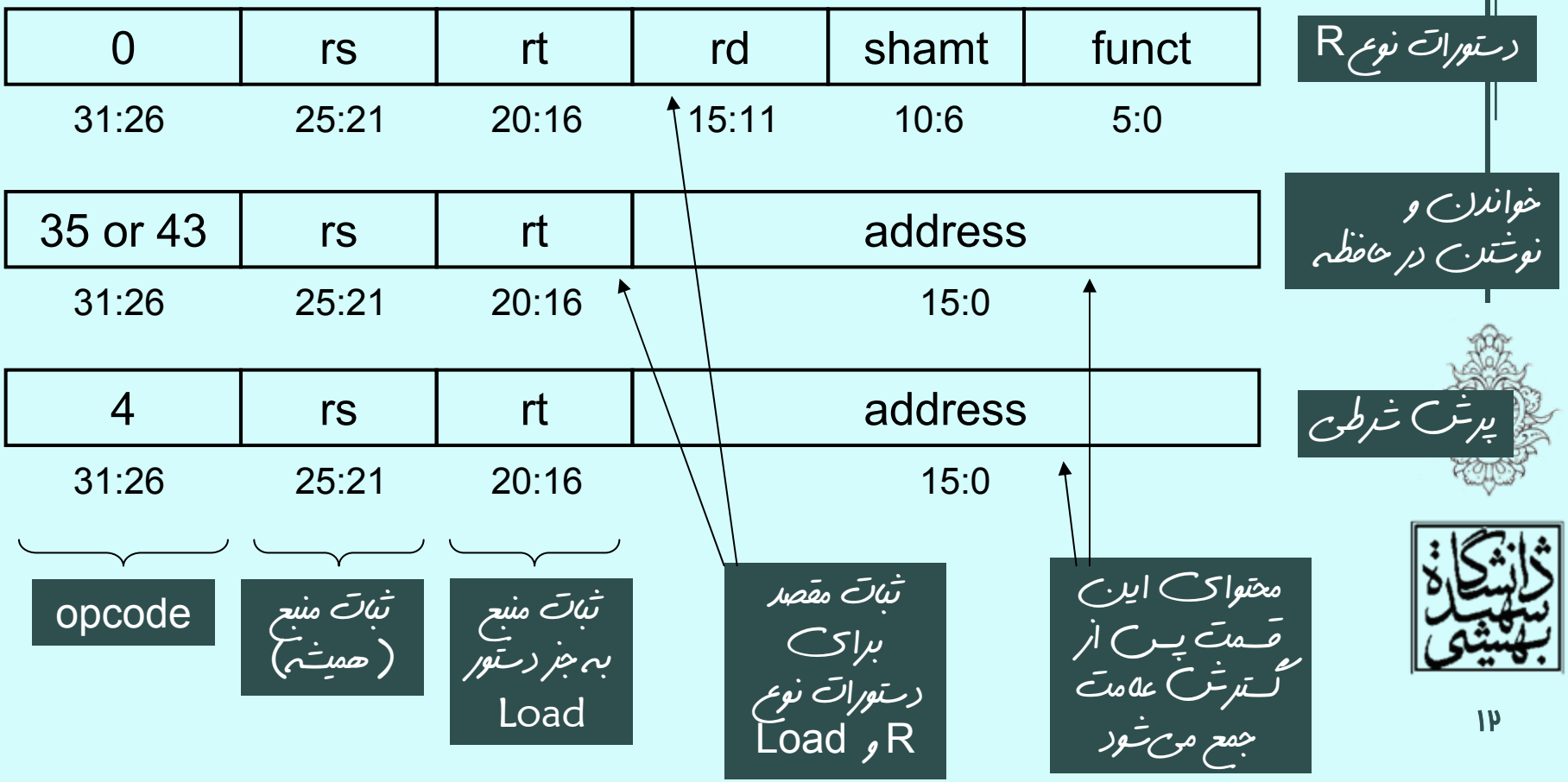
Instruction opcode	ALUOp	Instruction operation	Funct field	Desired ALU action	ALU control input
LW	00	load word	XXXXXX	add	0010
SW	00	store word	XXXXXX	add	0010
Branch equal	01	branch equal	XXXXXX	subtract	0110
R-type	10	add	100000	add	0010
R-type	10	subtract	100010	subtract	0110
R-type	10	AND	100100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





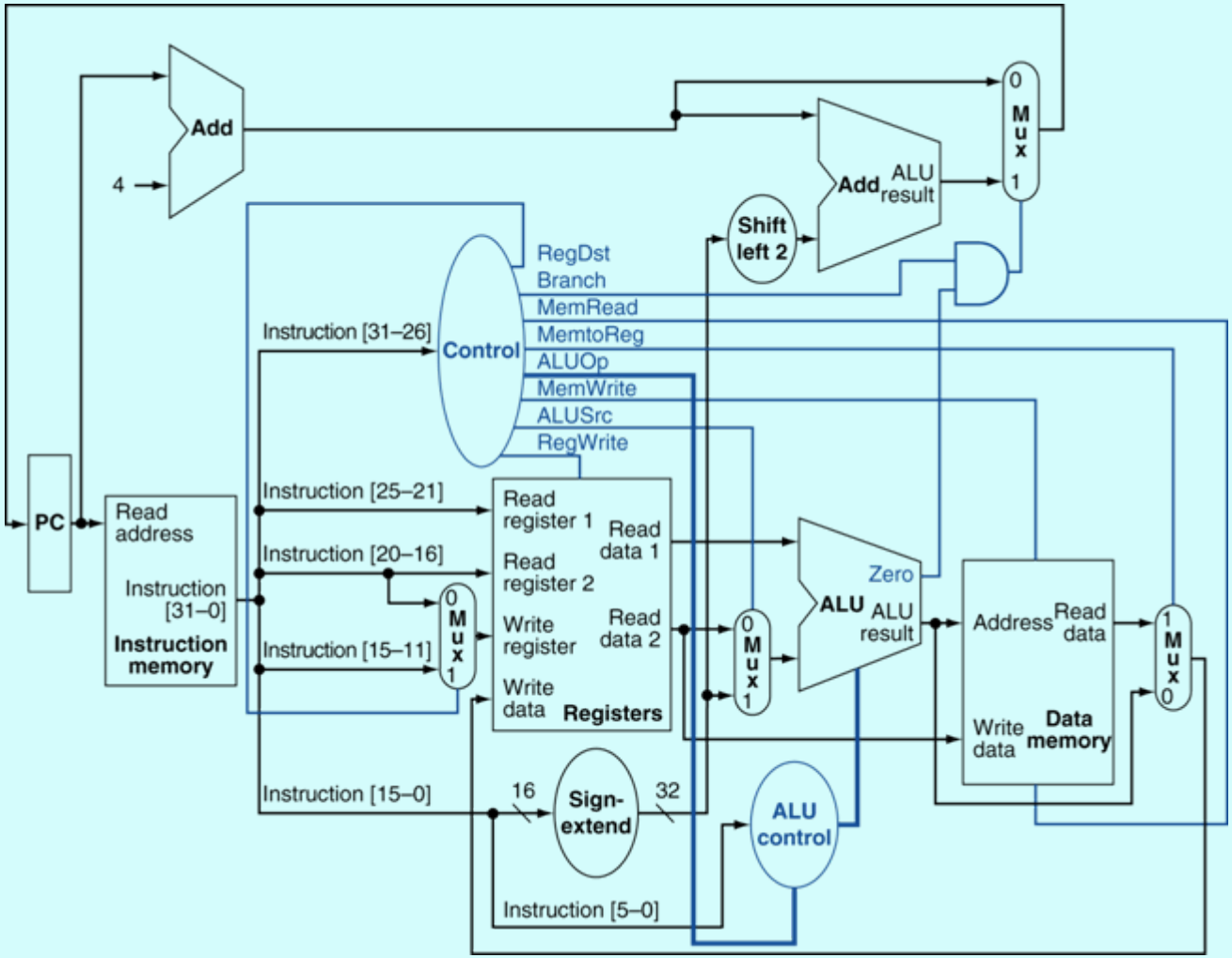
واحد کنترل اصلی

- در ادامه، واحد کنترل اصلی شرح داده خواهد شد.
- سیگنال‌های کنترلی که از دستورالعمل گرفته می‌شوند:



مسیر گذار داده همراه با واحد کنترل

- با توجه به قالب دستور، می‌توان برچسب برخی سیگنال‌های داده و کنترلی را مشخص نمود:

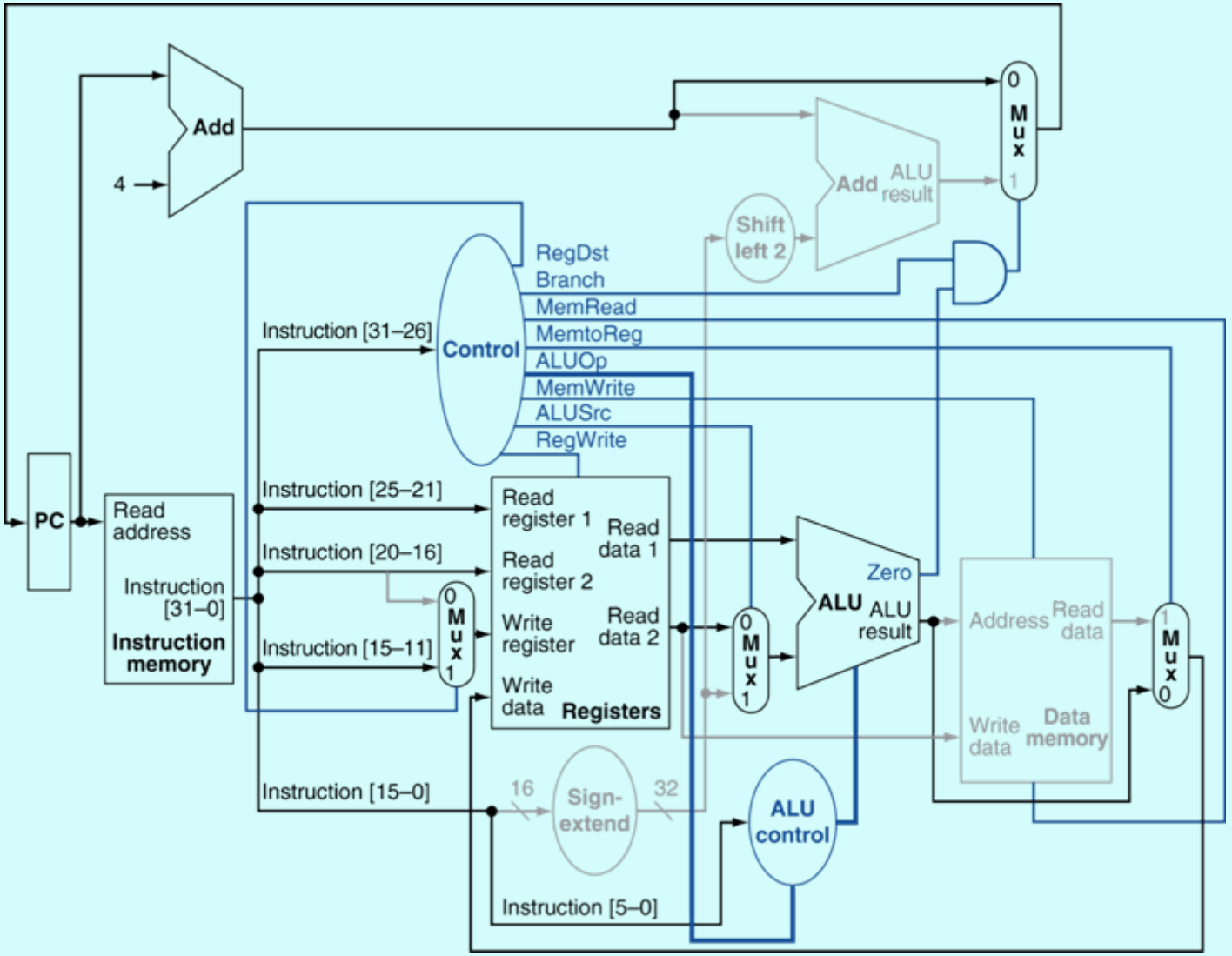


سیگنال‌های کنترلی

- سایر سیگنال‌های کنترلی، با توجه به opcode تعیین می‌شود.
- تنها PCSrc استثناست که به نتیجه‌ی ALU وابسته است.

Signal name	Effect when deasserted	Effect when asserted
RegDst	The register destination number for the Write register comes from the rt field (bits 20:16).	The register destination number for the Write register comes from the rd field (bits 15:11).
RegWrite	None.	The register on the Write register input is written with the value on the Write data input.
ALUSrc	The second ALU operand comes from the second register file output (Read data 2).	The second ALU operand is the sign-extended, lower 16 bits of the instruction.
PCSrc	The PC is replaced by the output of the adder that computes the value of PC + 4.	The PC is replaced by the output of the adder that computes the branch target.
MemRead	None.	Data memory contents designated by the address input are put on the Read data output.
MemWrite	None.	Data memory contents designated by the address input are replaced by the value on the Write data input.
MemtoReg	The value fed to the register Write data input comes from the ALU.	The value fed to the register Write data input comes from the data memory.

نحوه‌ی اجرای دستورات نوع R



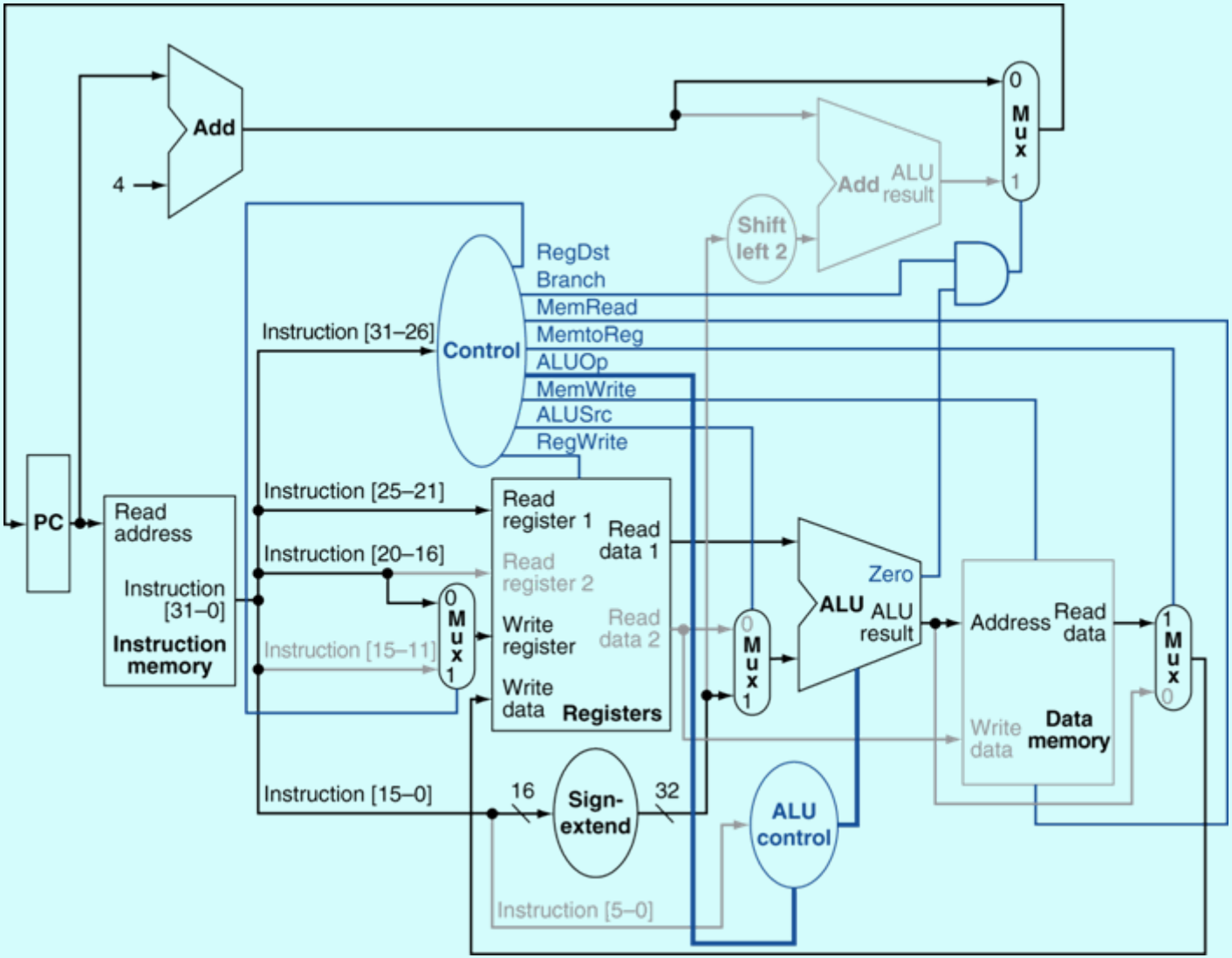
نحوه‌ی اجرای دستورات نوع R (ادامه...)

- هرچند همه‌ی دستورها در یک گام اجرا می‌شوند، می‌توان مراحل اجرای یک دستور را در چهار گام تصور نمود:

```
add $t1,$t2,$t3
```

- واکشی دستور و افزایش مقدار PC
- خواندن ثبات‌های $t2 و $t3 از بانک ثبات
- واحد کنترل ALU از روی بیت‌های funct خطوط کنترلی را برای انتخاب عمل مناسب انتخاب می‌کند.
- نتیجه‌ی محاسبات در ثبات مقصد ($t1) نوشته می‌شود.

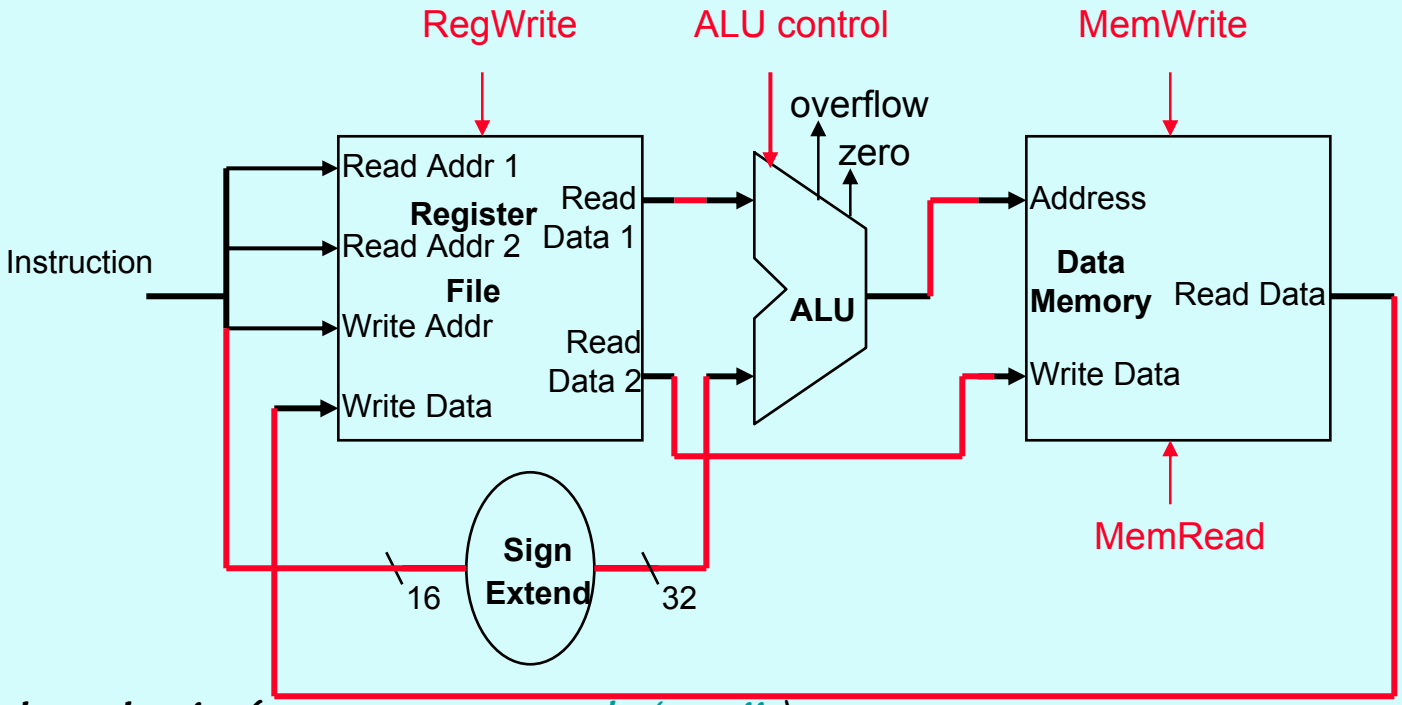
نحوه‌ی اجرای دستورات خواندن(نوشتن) از(در) حافظه



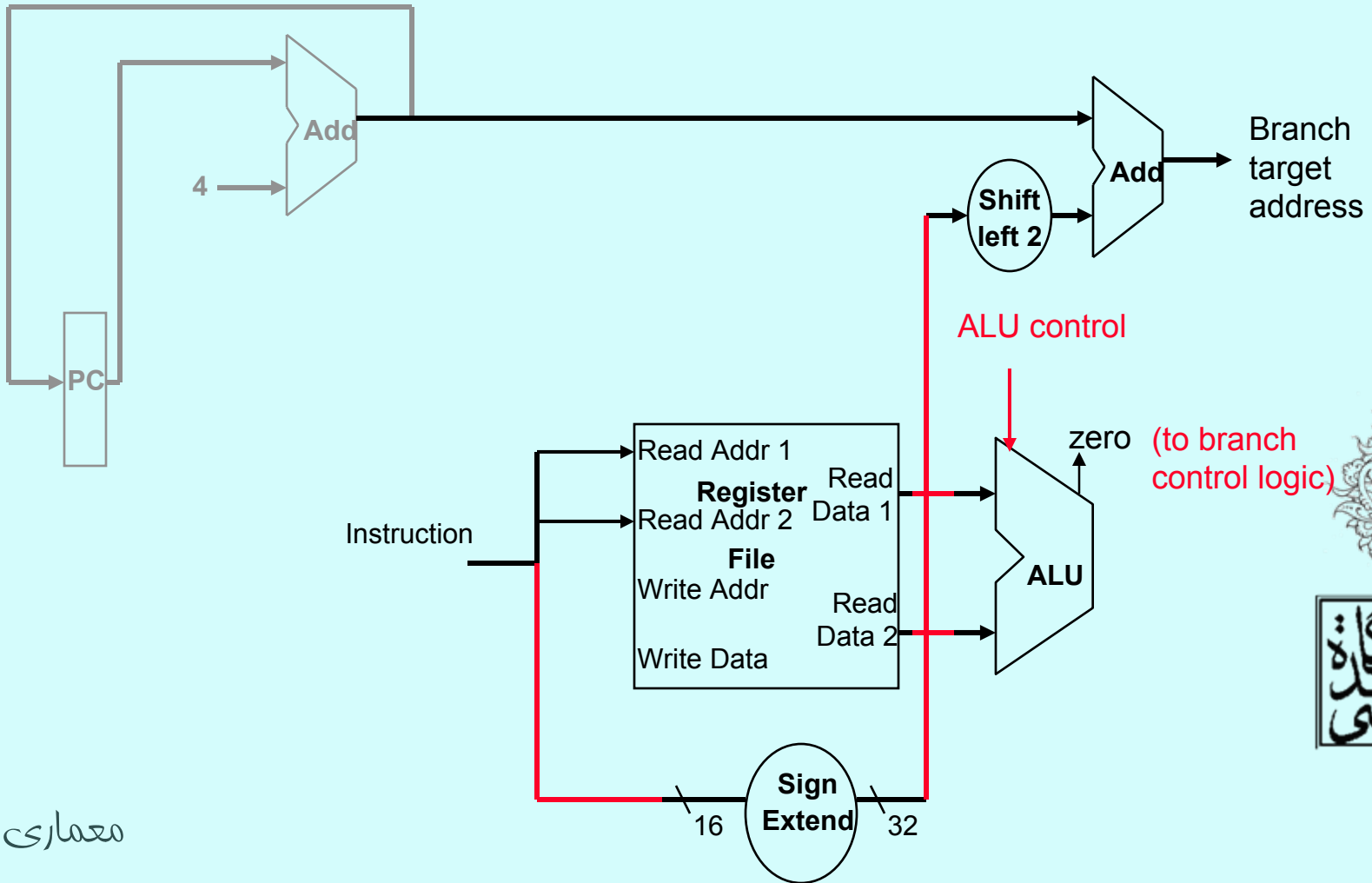
اجرای دستورات خواندن از حافظه (ادامه...)

```
lw $t1,offset($t2)
```

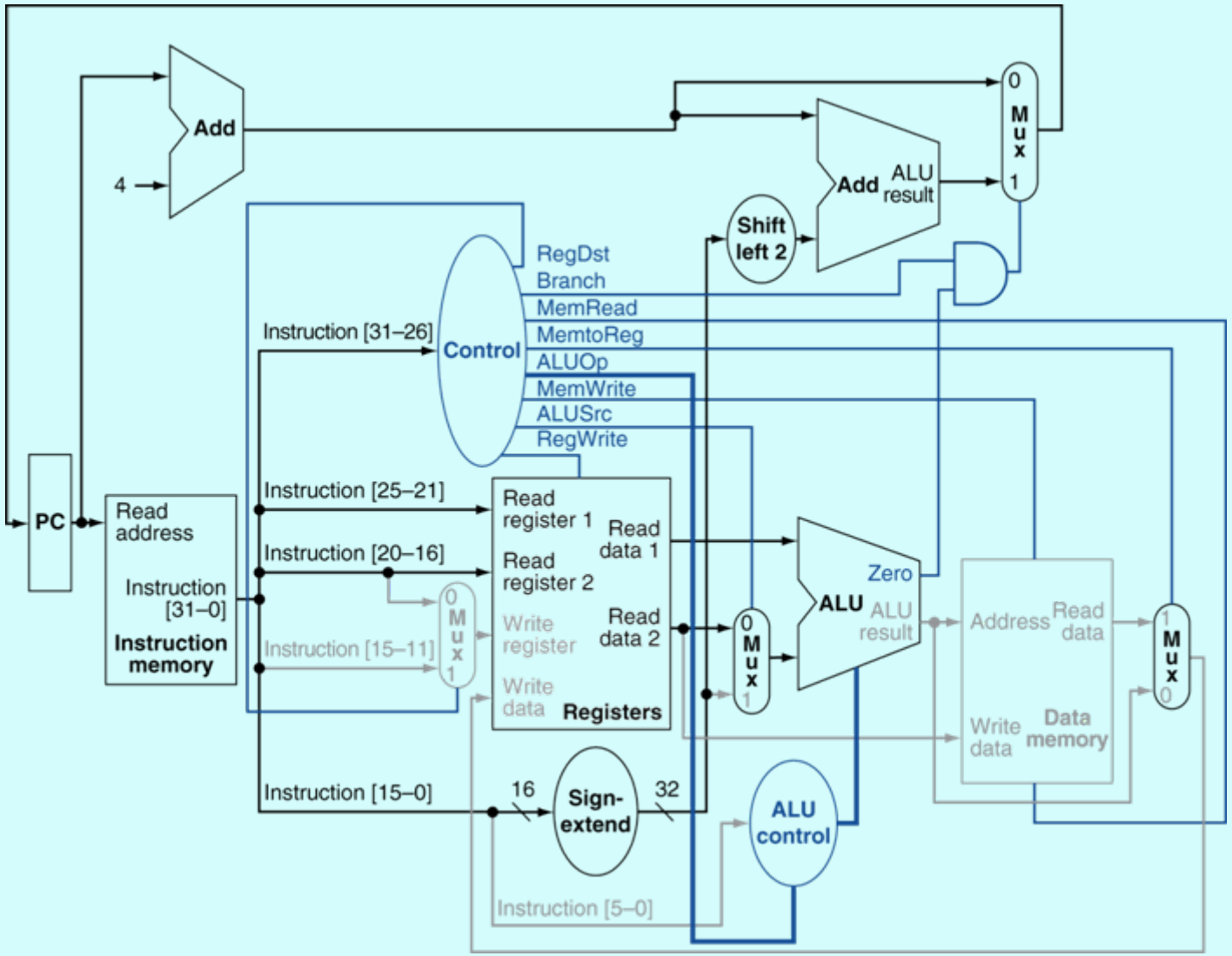
- واکشی دستور و افزایش مقدار PC
- خواندن ثبات $t2 از بانک ثبات
- ALU محتوای ثبات را با offset جمع می‌کند.
- حاصل به عنوان آدرس حافظه مورد استفاده قرار می‌گیرد.
- داده‌ی خوانده شد در حافظه در ثبات مقصد ($t1) نوشته می‌شود.



نحوه‌ی اجرای دستورات پرش شرطی



نحوه‌ی اجرای دستورات پرش شرطی (ادامه...)



نحوه‌ی اجرای دستورات پرش شرطی (ادامه...)

```
beq $t1,$t2,offset
```

- واکشی دستور و افزایش مقدار PC
- خواندن ثبات‌های $t1 و $t2 از بانک ثبات
- ALU عمل تفریق را انجام می‌دهد، بخش ثابت پس از گسترش علامت و شیفت به چپ به PC+4 اضافه می‌شود
- بر اساس خروجی zero در ALU در مورد این آدرس PC چه باشد، تصمیم‌گیری می‌شود.

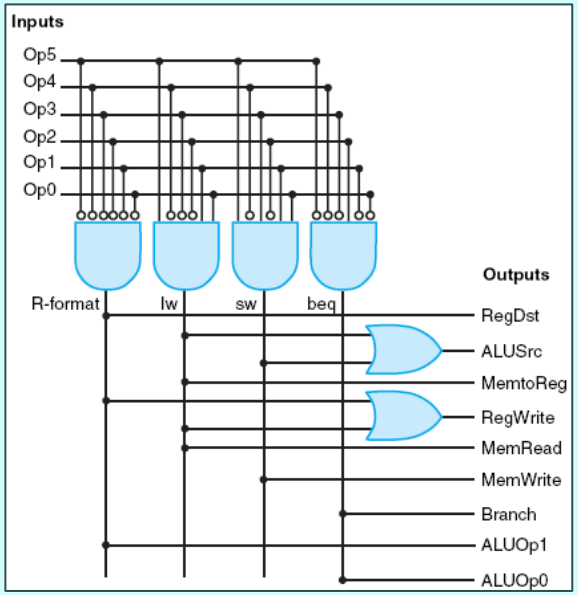
واحد کنترل اصلی

- خروجی واحد کنترل، سیگنال‌های کنترلی هستند.
- شش بیت opcode، ورودی واحد کنترل محسوب می‌شوند.

Instruction	RegDst	ALUSrc	Memto-Reg	Reg-Write	Mem-Read	Mem-Write	Branch	ALUOp1	ALUOp0
R-format	1	0	0	1	0	0	0	1	0
lw	0	1	1	1	1	0	0	0	0
sw	X	1	X	0	0	1	0	0	0
beq	X	0	X	0	0	0	1	0	1

واحد کنترل اصلی (ادامه...)

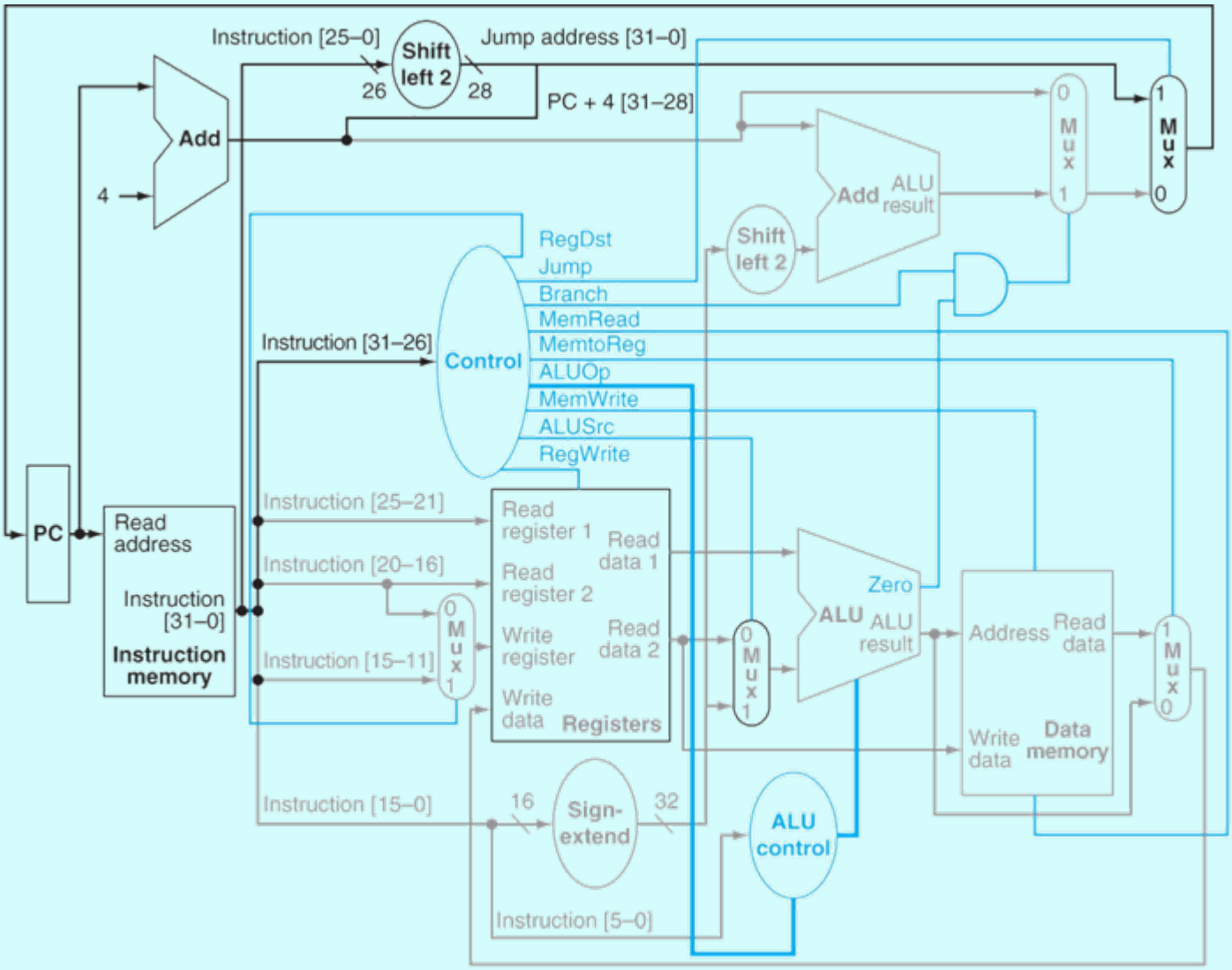
Input or output	Signal name	R-format	lw	sw	beq
Inputs	Op5	0	1	1	0
	Op4	0	0	0	0
	Op3	0	0	1	0
	Op2	0	0	0	1
	Op1	0	1	1	0
	Op0	0	1	1	0
Outputs	RegDst	1	0	X	X
	ALUSrc	0	1	1	0
	MemtoReg	0	1	X	X
	RegWrite	1	1	0	0
	MemRead	0	1	0	0
	MemWrite	0	0	1	0
	Branch	0	0	0	1
	ALUOp1	1	0	0	0
	ALUOp0	0	0	0	1



افزودن دستور پرش (Jump)

- چهار بیت پرارزش آدرس از PC+4 گرفته می‌شود.
- در دو بیت کم‌ارزش 00 قرار می‌گیرد.
- بیست و شش بیت دیگر از بخش آدرس دستورالعمل گرفته می‌شود.

افزودن دستور پرش (ادامه...)



معایب اجرای همه‌ی دستورها در یک سیکل

- کامپیوترهای اولیه از چنین ساختار تبعیت می‌کردند.
- طول سیکل ساعت باید با توجه به کندترین دستور انتخاب شود.

Making the common case fast

- این مسأله با اصول طراحی در تناقض است.
- در بخش بعدی شیوه‌ای دیگر، به نام خط لوله را بررسی خواهیم کرد.