دانشکده‌ی مهندسی برق و کامپیوتر– معماری کامپیوتر

کوییز دوم– بهار ۱۳۹۱

نام و نام خانوادگی: شماره دانشجویی:

۱. شکل پیوست شده، یک پردازنده MIPS با ۵ طبقه خط لوله مجهز به پیش‌فرستادن، تشخیص و رفع مخاطره و مدیریت استثنا را نشان می‌دهد. برنامه زیر بر روی این پردازنده اجرا می‌شود.

آدرس دستور	دستور
0x00003000	lw $1,0($0)
0x00003004	lw $2,4($0)
0x00003008	lw $3,8($0)
0x0000300C	add $1,$1,$2
0x00003010	add $1,$1,$3
0x00003014	addi $1,$1,15
0x00003018	sw $1,12($0)
0x0000301C	lui $10,0x7FFF
0x00003020	ori $10,$10,0xFFFF
0x00003024	add $1,$10,$10
0x00003028	sw $1,12($0)
0x0000302C	add $1,$2,$3
0x00003030	xor $7,$8,$9
0x00003034	slt $10,$11,$12

اگر فرض کنیم در هر خانه از حافظه داده، عدد معادل با اندیس آن خانه قرار گرفته باشد (یعنی مثلاً در خانه با اندیس ۷ عدد ۷ قرار گرفته باشد) مقدار سینگالهای خواسته شده را در سیکل‌های مورد نظر بنویسید. (اولین سیکل را سیکل ۱ می‌نامیم)

	سیکل ۱۳	سیکل ۱۲	سیکل ۹	سیکل ۸
ForwardA				
ForwardB				
EX/MEM.RegisterRd				
MEM/WB.RegisterRd				
IF/ID.Regwrite				
ID/EX.Regwrite				
EX/MEM.Regwrite				
MEM/WB.Regwrite				
PC				
PcWrite				
ID.Flush				
IF.Flush				
EX.Flush				
EPC				

دانشکده‌ی مهندسی برق و کامپیوتر – معماری کامپیوتر

به نام خدا

نام و نام خانوادگی شماره دانشجویی:

پرسش‌های چندگزینه‌ای: لطفا برای هر سوال علاوه بر انتخاب گزینه، پاسخ مشروح هم بنویسید.

۲. یک واحد محاسباتی مجهز به خط لوله دارای پنج قسمت با زمان‌های اجرای ۳۶، ۳۹، ۲۳، ۲۸ و ٦٤ نانوثانیه است. اگر از ثبات‌هایی با تاخیر یک نانوثانیه در بین قسمت‌های مختلف خط لوله استفاده شده باشد. حداکثر تسریع این واحد محاسباتی در صورت استفاده از خط لوله چقدر است؟

الف) ۲/۹۲ ب) ۳/۵ ج) ۵ د) ۷/۹۲

۳. در پردازنده‌ای با ساختار خط لوله دستورات در هشت مرحله اجرا می‌شوند. چنانچه دستوری از نوع پرش شرطی (branch) باشد، به دستورهای بعدی اجازه‌ی ورود به خط لوله داده نمی‌شود تا این که دستور پرش به پایان برسد. برنامه‌ای در حال اجراست که ۱۰۰ دستور دارد و در آن بعد از هر ۱۹ دستور معمولی یک دستور پرش شرطی ظاهر می‌شود.اگر تاخیر هر مرحله و ثبات‌های وابسته به آن ۱۰ نانو ثانیه باشد، اجرای این برنامه چقدر طول می‌کشد؟

الف) ۱۷۰۰ ب) ۱٤۲۰ ج) ۱۳۵۰ د) ۱۰۷۰

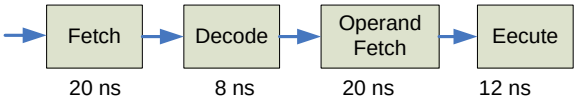
٤. در یک پردازنده دارای خط لوله از یک خط لوله‌ی یازده سطحی برای واکشی و اجرای دستورات استفاده می‌شود. اکر ده درصد دستورات برنامه پرش باشد، حداکثر و حداقل تسریع قابل احتصال توسط این پردازنده نسبت به پردازنده‌ی مشابه بدون خط لوله چقدر خواهد بود؟ (فرض کنید مشکلات وابستگی داده و دسترسی حافظه برای اجرای دستورات وجود ندارد)

الف) حداکثر ۱۰ و حداقل ۸,۵ ب) حداکثر ۱۱ و حداقل ۵,۵

ج) حداکثر ۱۰ و حداقل ۸,۱ د) حداکثر ۱۱ و حداقل ۹,۵

۵. به فرض داشتن خط لوله‌ی چهارسطحی برای اجرای دستورات در یک پردازنده، اگر در یک برنامه به طور متوسط در هر ۱۰ دستور یک پرش شرطی وجود داشته باشد و به احتمال پنجاه درصد پرش انجام شود، حداکثر تسریع به دست آمده برای اجرای این برنامه نسبت به زمانی که پردازنده به خط لوله مجهز نیست چقدر خواهید بود؟

الف) ۲,۳ ب) ۲,۵ ج) ۲,٦ د) ۲,۷



٦. یک پردازنده دارای چهار گروه دستورالعمل‌های نوع الف تا د می‌باشدو نسبت وقوع این دستورالعمل‌ها در یک برنامه‌ی bench mark در جدول زیر نشان داده شده است.علاوه بر آن در جدول مشخص شده است که هر گروه از دستورالعمل‌ها نیاز به چه مراحلی در اجرا دارند و زمان اجرای هر مرحله چه مقدار است. نسبت افزایش سرعت اجرای این برنامه در صورت پیاده‌سازی خط لوله نسبت به پیاده‌سازی معمولی آن چقدر است؟

		WB	MEM	EXE	D	F
نوع دستور	درصد وقوع	7ns	12ns	10ns	7ns	10ns
الف	20%	√	√	√	√	√
ب	40%	√	-	√	√	√
ج	20%	-	√	√	√	√
د	20%	-	-	√	√	√

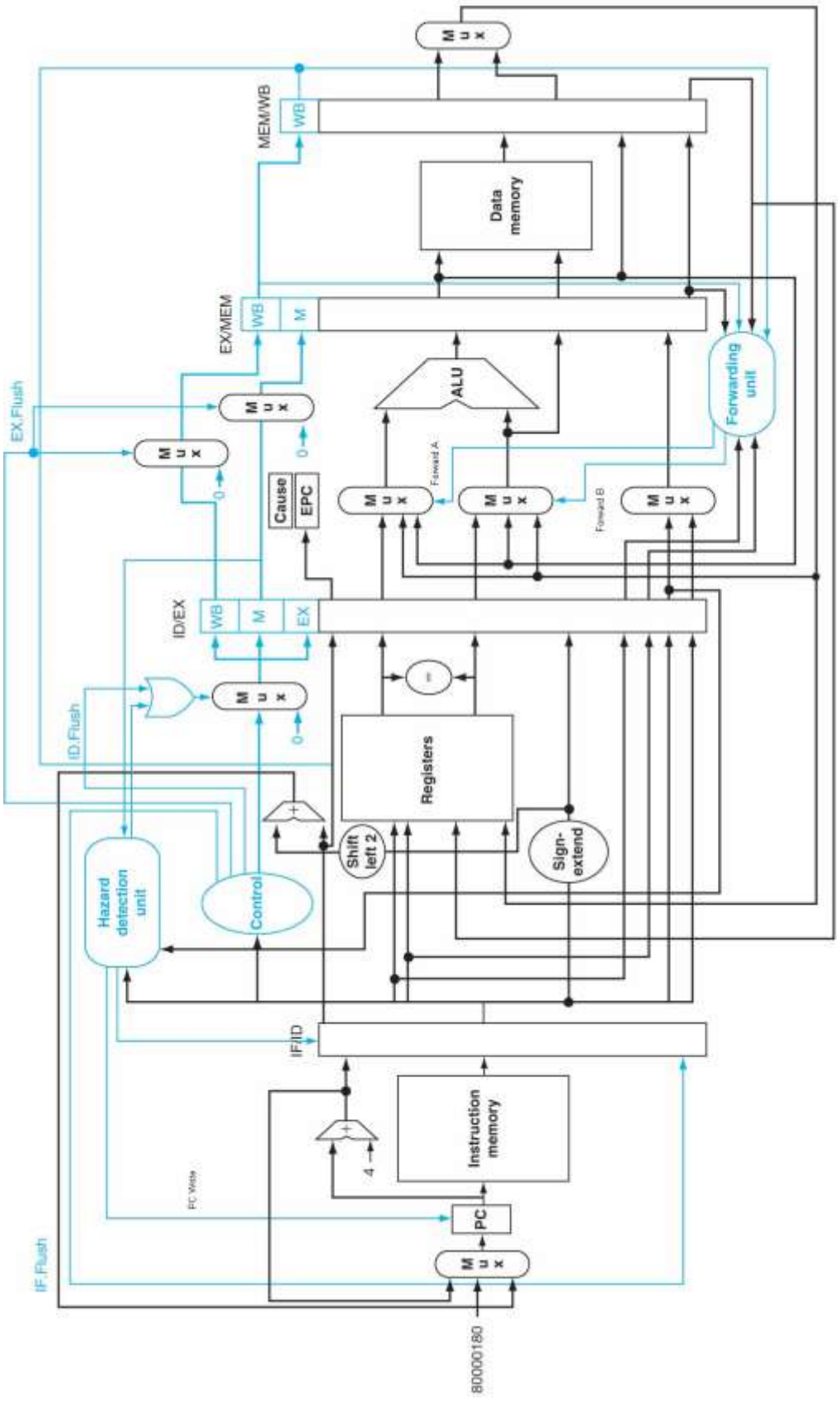
الف) ٥ ب) ٤ ج) ٣,٦ د) ٣

دانشکده‌ی مهندسی برق و کامپیوتر- معماری کامپیوتر

کوییز دوم- بهار ۱۳۹۱

نام و نام خانوادگی:

شماره دانشجویی:



پاسخ سوال ۱)

برای بدست آوردن مقدار سیگنالهای کنترلی کافی است بدانیم در سیکل‌های مورد نظر، چه دستوری در چه طبقه‌ای از خط لوله قرار گرفته است. سپس با داشتن دستور می‌توان مقدار سیگنالهای کنترلی را بدست آورد.

هنگامیکه دستورها زیاد باشد (مثل این سوال)، ترسیم نمودار چند سیکل ساعتی خط لوله برای محاسبه دستورات درون خط لوله برای یک سیکل خاص، بسیار زمان‌بر است. راه دیگر محاسبه است. واضح است که اگر پرش، انشعاب، استثنا یا تعلیق رخ ندهد، دستورها پشت سرهم اجرا خواهند شد و در هر سیکل یک دستور وارد خط لوله می‌شود و در عوض یک دستور از خط لوله خارج می‌گردد. در سیکل اول دستور lw $1,0($0) در طبقه Fetch از خط لوله قرار دارد، در سیکل دوم همین دستور در طبقه Decode و دستور lw $2,4($0) در طبقه Fetch قرار می‌گیرد. اگر هیچ مشکلی پیش نیاید، در سیکل هشتم، دستور هشتم در طبقه Fetch قرار دارد، پس دستور هفتم در طبقه Decode است و ... . یعنی در سیکل ۸، نمودار تک سیکل ساعتی خط لوله به صورت زیر است:

lui $10,0x7FFF	sw $1,12($0)	addi $1,$1,15	add $1,$1,$3	add $1,$1,$2
Fetch	Decode	Execute	Memory	WriteBack

قبل از آنکه مقدار سیگنالهای کنترلی را برای این سیکل بدست آوریم، ابتدا ثابت می‌کنیم فرض ما مبنی بر عدم مشکل درست است.

- **پرش و انشعاب:** در کل این کد هیچ دستور پرش یا انشعاب وجود ندارد
- **استثنا:** اولاً همه دستورها، دستورهای شناخته شده پردازنده MIPS هستند، پس هیچگاه استثنا برای «دستورالعمل تعریف نشده» نداریم. مقدار رجیسترهای $1 و $2 و $3 نیز به اندازه‌ای بزرگ نیستند که حاصل جمع آنها باعث سرریز شود. پس استثنا برای سرریز نیز اتفاق نمی‌افتد.
- **تعلیق:** چون دستور پرش و انشعاب نداریم، مخاطره کنترلی نیز نداریم و در نتیجه تعلیق به دلیل دستورهای پرش و انشعاب نخواهیم داشت. همچنین پردازنده مجهز به پیش‌فرستادن می‌باشد. پس تمامی مخاطرات داده‌ای به جز خواندن رجیستری که می‌خواهیم از حافظه مقداری را درون آن بریزیم، تعلیقی را ایجاد نمی‌کنند. اگر دقت کنید بین بارگذاری $1 و استفاده از آن دو دستور، و بین بارگذاری هر یک از رجیسترهای $2 و $3 و استفاده از آنها یک دستور فاصله وجود دارد. هنگامیکه پیش‌فرستادن داشته باشیم، وجود فاصله یک دستوری برای رفع تعلیق این دستور کافی است. پس هیچ تعلیقی نخواهیم داشت.

مقدار سینگالها در سیکل ۸ به صورت زیر است:

سیگنال	مقدار	دلیل
ForwardA	10	دستورهای طبقه MEM و WB هر دو باید در رجیستر $1 مقدار بنویسند، دستور addi که در طبقه EXE است، می‌خواهد مقدار $1 را بخواند. مقدار موجود در طبقه MEM مقدار به‌روزتری است. پس باید این مقدار به طبقه EXE فرستاده شود. پس باید Forward A برابر با 10 باشد. (برای توضیح بیشتر به «شرایط پیش‌فرستادن» در کتاب یا اسلایدهای استاد مراجعه کنید.)
ForwardB	00	عملوند دوم دستور addi یک بلافصل می‌باشد. لذا نیاز به پیش‌فرستادن نیست.

		توجه کنید که در شکل تفاوتی بین بلافصل و مقدار رجیستر rt قائل نشده است.
EX/MEM.RegisterRd	0x1	مقدار این سیگنال (که خود یک رجیستر خط لوله است)، آدرس رجیستر مقصد دستور موجود در طبقه MEM را مشخص می‌کند. این دستور add $1,$1,$3 می‌باشد پس مقدار این سیگنال 0x1 می‌باشد.
MEM/WB.RegisterRd	0x1	همانند سیگنال بالایی است با این تفاوت که برای طبقه WB است. در این سیکل دستور add $1,$1,$3 در این طبقه وجود دارد. پس مقدار این سیگنال 0x1 می‌باشد.
IF/ID.Regwrite	0	مشخص می‌کند که دستور فعلی طبقه ID نیاز به نوشتن در رجیستر فایل دارد یا نه. چون دستور sw است، نیاز به نوشتن در رجیستر فایل ندارد، پس مقدار سیگنال یک است.
ID/EX.Regwrite	1	همانند سیگنال بالاست با این تفاوت که برای طبقه EXE می‌باشد. دستور addi است پس نوشتن در رجیستر فایل دارد.
EX/MEM.Regwrite	1	مشابه بالا
MEM/WB.Regwrite	1	مشابه بالا
PC	0x0000301C	مقدار PC آدرس دستوری را مشخص می‌کند که در مرحله Fetch قرار دارد. یعنی آدرس دستور lui .
PcWrite	1	هنگامی این سیگنال صفر است که نیاز به تعلیق داشته باشیم. چون تعلیق نداریم پس مقدار آن باید یک باشد.
ID.Flush	0	هر گاه لازم به خالی کردن خط لوله باشد، مقدار این سیگنال یک می‌شود. خط لوله در دو صورت خالی می‌شود: انشعاب و استثنا. در اینجا هیچکدام رخ نداده است، پس مقدار سیگنال صفر است.
IF.Flush	0	مشابه بالا
EX.Flush	0	مشابه بالا
EPC	بدون تغییر	چون استثنا رخ نداده است، پس مقدار این رجیستر بدون تغییر می‌ماند.

در سیکل ۹ نیز شرایط همانند سیکل ۸ است. نه تعلیق داریم، نه استثنا، نه پرش و نه انشعاب. نمودار تک سیکل ساعتی خط لوله برای سیکل ۹ به صورت زیر است:

ori $10,$10,0xFFFF	lui $10,0x7FFF	sw $1,12($0)	addi $1,$1,15	add $1,$1,$3
Fetch	Decode	Execute	Memory	WriteBack

با توجه به نمودار بالا می‌توان مقدار سیگنالها را برای سیکل ۹ به صورت زیر حساب کرد:

سیگنال	مقدار	دلیل
ForwardA	00	دستور sw در طبقه EXE قرار دارد. برای محاسبه آدرس حافظه لازم است مقدار رجیستر $0 در ورودی اول ALU قرار گیرد یعنی نیاز به پیش‌فرستادن نداریم.
ForwardB	00	برای محاسبه آدرس حافظه به بلافصل نیز نیاز داریم.
EX/MEM.RegisterRd	0x1	دستور addi در طبقه MEM قرار گرفته است. این دستور I-Type می‌باشد و فیلد Rd برای آن بی معنی است اما توجه داشته باشید که رجیستر

		EX/MEM.RegisterRD همواره مقدار رجیستر مقصد دستوری که در مرحله MEM قرار دارد را نگه‌داری می‌کند. یعنی بسته به دستور مقدار آن با فیلد Rd یا Rt برابر است.
MEM/WB.RegisterRd	0x1	مشابه سیکل ۸
IF/ID.Regwrite	1	lui نیاز به نوشتن در رجیستر فایل دارد
ID/EX.Regwrite	0	sw نبار به نوشتن در رجیستر فایل ندارد
EX/MEM.Regwrite	1	addi نیاز به نوشتن در رجیستر فایل دارد
MEM/WB.Regwrite	1	add نیاز به نوشتن در رجیستر فایل دارد
PC	0x00003020	مشابه سیکل ۸
PcWrite	1	مشابه سیکل ۸
ID.Flush	0	مشابه سیکل ۸
IF.Flush	0	مشابه سیکل ۸
EX.Flush	0	مشابه سیکل ۸
EPC	بدون تغییر	مشابه سیکل ۸

باز هم تا سیکل ۱۲ هیچ رخداد خاصی وجود ندارد. در سیکل ۱۲ نمودار تک سیکل ساعتی خط لوله به صورت زیر است:

`add $1,$2,$3`	`sw $1,12($0)`	`add $1,$10,$10`	`ori $10,$10,0xFFFF`	`lui $10,0x7FFF`
Fetch	Decode	Execute	Memory	WriteBack

در این سیکل نیز از پرش، انشعاب و تعلیق خبری نیست. اما دستور در مرحله EXE دستور add می‌باشد که ممکن است باعث بروز استثنا شود اینکه دو دستور lui و ori باعث بارگذاری مقدار 0x7FFFFFFF در $10 می‌شوند و add رجیستر $10 را با خودش جمع می‌کند. مقدار 0x7FFFFFFF بزرگترین عدد مثبت ۳۲ بیتی است، پس حاصل جمع با خودش حتماً سرریز خواهد کرد. پس در این سیکل استثنا رخ می‌دهد.

هنگامیکه در سیکل ۱۲ هستیم، واحد کنترل وجود سرریز را تشخیص می‌دهد اما در سیکل ۱۲ خط لوله خالی نمی‌شود بلکه در سیکل ۱۳ خالی می‌شود. لذا مقدار سیگنالها در سیگنال ۱۲ به صورت زیر است:

سیگنال	مقدار	دلیل
ForwardA	10	$10 باید به عنوان ورودی اول ALU قرار گیرد. دستورهای مرحله MEM و WB هر دو در این رجیستر می‌نویسند اما مقدار صحیح در مرحله MEM قرار دارد.
ForwardB	01	بازهم $10 باید به عنوان ورودی دوم ALU قرار گیرد. پس باید از مرحله MEM مقدار صحیح را بگیریم. لذا مقدار 01 مناسب است. (طبق شکل صورت سوال این مقدار انتخاب شد. متاسفانه کتاب رویکرد ثابتی در ارائه Data Path ندارد و بسته به شکل، مقدار سیگنالهای کنترلی ممکن است تغییر کند)
EX/MEM.RegisterRd	0xA	دستور ori باید در رجیستر $10 بنویسد.
MEM/WB.RegisterRd	0xA	دستور lui باید در رجیستر $10 بنویسد (با وجود اینکه عدد ۱۰ در فیلد Rt این دستور قرار دارد)

IF/ID.Regwrite	0	sw نیاز به نوشتن در رجیستر فایل ندارد
ID/EX.Regwrite	1	add نبار به نوشتن در رجیستر فایل دارد
EX/MEM.Regwrite	1	ori نیاز به نوشتن در رجیستر فایل دارد
MEM/WB.Regwrite	1	lui نیاز به نوشتن در رجیستر فایل دارد
PC	0x0000302C	مشابه سیکل ۸
PcWrite	1	توجه داشته باشید که استثنا رخ داده اما استثنا نیاز به تعلیق ندارد و باید بلافاصله به روتین مدیریت استثنا پرش کند.
ID.Flush	1	استثنا رخ داده پس باید خط لوله خالی شود.
IF.Flush	1	استثنا رخ داده پس باید خط لوله خالی شود.
EX.Flush	1	استثنا رخ داده پس باید خط لوله خالی شود.
EPC	بدون تغییر	آدرس دستور بعد از دستوری که موجب استثنا شده است در پشت رجیستر EPC قرار دارد اما تا لبه بالارونده کلاک اتفاق نیفتد، این مقدار درون EPC بارگذاری نمی‌شود.

اکنون لبه بالارونده کلاک اتفاق می‌افتد و وارد سیکل ۱۳ می‌شویم. پردازنده به آدرس روتین مدیریت استثنا برای سرریز یعنی 0x80000180 پرش می‌کند و خط لوله خالی می‌شود. توجه کنید که طبق مثال کتاب، مرحله WB خالی نمی‌شود!!!

??	nop	nop	nop	ori $10,$10,0xFFFF
Fetch	Decode	Execute	Memory	WriteBack

به جای ?? اولین دستور روتین مدیریت استثنا یعنی دستور موجود در آدرس 0x80000180 قرار می‌گیرد:

سیگنال	مقدار	دلیل
ForwardA	00	برای nop نیاز به Forwarding نداریم
ForwardB	00	برای nop نیاز به Forwarding نداریم
EX/MEM.RegisterRd	0x0	دستور nop در رجیستر صفر می‌نویسد.
MEM/WB.RegisterRd	0xA	دستور ori در رجیستر ۱۰ می‌نویسد
IF/ID.Regwrite	0	nop نیاز به نوشتن ندارد.
ID/EX.Regwrite	0	nop نیاز به نوشتن ندارد.
EX/MEM.Regwrite	0	nop نیاز به نوشتن ندارد.
MEM/WB.Regwrite	1	ori نیاز به نوشتن دارد.
PC	0x80000180	آدرس دستور فعلی
PcWrite	1	nop موجب هیچ عملی نمی‌شود پس تعلیق ندارد.
ID.Flush	0	استثنا رخ داده و خط لوله خالی شده. دیگر نیاز به خالی کردن خط لوله نیست.
IF.Flush	0	مشابه بالا
EX.Flush	0	مشابه بالا
EPC	0x00003028	دقت کنید که دستور add که در آدرس 0x00003024 قرار دارد موجب استثنا شده است اما همواره آدرس دستور بعدی در EPC قرار می‌گیرد و وظیفه کم کردن عدد ۴ بر عهده روتین مدیریت استثنا است.

پاسخ سوال ۲)

در حالت غیرخط لوله پردازنده می‌تواند تک سیکلی (Single-Cycle) یا چند سیکلی (Multi-Cycle) باشد. برای هنگامیکه پردازنده چند سیکلی باشد به اطلاعات بیشتری از جمله واحدهای مورد نیاز در اجرای هر دستور نیاز داریم. پس با توجه به اطلاعات صورت سوال، مقصود حالت تک سیکلی است. در این حالت تمام قسمت‌های پشت سر هم قرار گرفته و طول سیکل برابر تا مجموع تاخیر آنها خواهد بود:

$$Single - Cycle\ CPU\ Cycle = ۳۶ + ۳۹ + ۲۳ + ۲۸ + ۶۴ = ۱۹۰$$

اگر تعداد دستورها را n فرض کنیم. زمان اجرای تک سیکلی برابر است با:

$$Single - Cycle\ CPU\ Time = IC.CPI.Period = n \times ۱ \times ۱۹۰$$

دقت کنید که در پردازنده‌های تک سیکلی، *CPI* برای تمامی دستورها برابر یک است. چون طبق تعریف، معماری تک سیکل معماری است که تمامی دستورها در یک سیکل اجرا شوند. پس *CPI* برای همه دستورها یک است.

در حالت خط لوله، طول سیکل برابر است با بزرگترین تاخیر بین همه قسمت‌ها بعلاوه تاخیر ثبات‌های خط لوله. پس:

$$Pipeline\ CPU\ Cycle = ۶۴ + ۱ = ۶۵$$

تعداد دستورها n و تعداد طبقات خط لوله ۵ است. لذا:

$$Pipeline\ CPU\ Time = \left(n + ۵ - ۱\right) \times ۶۵$$

در نتیجه تسریع برابر است با:

$$\text{تسریع} = \frac{۱۹۰n}{۶۵n + ۴ \times ۶۵}$$

حداکثر تسریع هنگامی اتفاق می‌افتد که تعداد دستورها بسیار بزرگ باشد $(n \to \infty)$ و تعلیق ایجاد نشود (یعنی هیچ‌گاه در خط لوله، حباب نداشته باشیم. پس حدکثر تسریع برابر است با:

$$\text{حداکثر تسریع} = \lim_{n\to\infty} \text{تسریع} = \frac{۱۹۰}{۶۵} = ۲/۹۲$$

پس گزینه «الف» صحیح است.

پاسخ سوال ۳)

راه اول:

طبق صورت سوال رویکرد اتخاذ شده برای انشعاب رویکرد تعلیق است. هرجا رویکرد تعلیق اختیار شود به این معنی است که می‌توان به تعداد کافی nop درج کرد تا دیگر لازم نباشد تعلیق به صورت سخت‌افزاری انجام شود (در این صورت تعداد سیکل‌های لازم برای اجرای هر یک از دستورها، تعداد طبقات خط لوله می‌باشد). پس از هر ۱۹ دستور معمولی، یک انشعاب وجود دارد. کافی است پس از هر انشعاب به جز دستور انشعاب آخر، ۷ دستور nop قرار دهیم. پس تعداد nopهای اضافه شده برابر است با:

$$\text{تعداد } nop \text{ اضافه شده} = ۷ \times \left(\frac{۱۰۰}{۲۰} - ۱\right) = ۲۸$$

عدد $\frac{۱۰۰}{۲۰}$ تعداد دستورهای انشعاب است. پس تعداد کل دستورها پس از افزودن nop برابر است با:

$$\text{تعداد دستور} = ۱۰۰ + ۲۸ = ۱۲۸ \xrightarrow{\text{زمان سیکل یک} = ۱۰ns} \text{اجرا زمان} = \left(۱۲۸ + ۸ - ۱\right) \times ۱۰ = ۱۳۵۰\boldsymbol{ns}$$

راه دوم:

اگر رویکرد اتخاذ شده برای یک دستور خاص، تعلیق باشد، آن دستور مرزی است که دستورهای قبل و بعد آنرا از نظر زمان اجرایی از هم مستقل می‌کند. یعنی در این سوال هر دستور انشعاب باعث می‌شود که دستورهای قبل و بعد آن از نظر زمان اجرایی همانند دو برنامه مجزا عمل کنند. پس زمان اجرای برنامه برابر است با تعداد دستورهای انشعاب ضرب‌در زمان اجرای بخش‌های مستقل. تعداد دستورهای انشعاب $\frac{۱۰۰}{۲۰}$ است و زمان اجرای هر ۲۰ دستور $\left(۲۰ + ۸ - ۱\right) \times ۱۰ = ۲۷۰ns$ است. پس زمان اجرا برابر است با:

$$\text{زمان اجرا} = ۲۷۰ \times ۵ = ۱۳۵۰\boldsymbol{ns}$$

پس گزینه «ج» صحیح است.

پاسخ سوال ۴)

حداکثر تسریع در صورت عدم وجود تعلیق رخ می‌دهد برابر است با تعداد طبقات خط لوله. در صورت سوال ذکر شده که مشکلات وابستگی داده و دسترسی حافظه نداریم. معنی آن عدم تعلیق است. لذا حداکثر تعلیق ۱۱ است.

حداقل هنگامی حاصل می‌شود که از تعلیق برای پرش‌ها استفاده می‌کنیم. یعنی در صورت برخورد با پرش تا خروج آن از خط لوله، حباب وارد خط لوله کنیم. این بدان معنی است که دستور پرش تکه‌هایی در برنامه ایجاد می‌کند که از نظر زمان اجرا مستقل هستند.

فرض می‌کنیم k طبقه خط لوله داشته باشیم و n تعداد کل دستورها و m تعداد دستورهای پرش باشد. همچنین فرض می‌کنیم پرش اول بعد از $(n_۱ - ۱)$ دستور قرار گرفته باشد. پرش دوم بعد از $(n_۲ - ۱)$ دستور و ... . تعداد سیکل کل برابر است با مجموع سیکل‌های لازم برای هر مجموعه مستقل:

$$\text{تعداد سیکل کل خط لوله} = (n_۱ + k - ۱) + (n_۲ + k - ۱) + \cdots + (n_m + k - ۱)$$

$$= \sum_{i=۱}^{m} (n_i + k - ۱) = n + m(k - ۱)$$

پس زمان اجرا برای پریود T برابر است با:

$$\text{زمان اجرا خط لوله} = (n + m(k - ۱))T$$

در این سوال ده درصد دستورها پرش هستند . پس $m = ۰/۱n$ با جایگذاریم داریم:

$$\text{زمان اجرا خط لوله} = (n + ۰/۱n(۱۱ - ۱))T = ۲nT$$

زمان اجرای تک سیکل نیز برابر است با

$$\text{زمان اجرا تک سیکل} = ۱۱nT$$

$$\text{حداقل تسریع} = \frac{۱۱nT}{۲nT} = ۵/۵$$

پس گزینه «ب» صحیح است.

پاسخ ۵)

در صورت سوال مشخص نشده که رویکرد اتخاذ شده برای دستورهای پرش چیست. این سوال را برای رویکرد تعلیق و فرض عدم پرش حل می‌کنیم. حل برای رویکردهای دیگر مخاطرات کنترلی نیاز به اطلاعات بیشتری دارد که در سوال موجود نیست.

رویکرد (همیشه) تعلیق:

در این رویکرد چه پرش انجام شود چه نشود، حباب وارد خط لوله می‌شود. لذا فرقی ندارد که پرش انجام شود یا نه. زمان اجرای بدون خط لوله برابر است با

$$\text{زمان اجرا بدون خط لوله} = \left(۲۰ + ۸ + ۲۰ + ۱۲\right) \times n = ۶۰n$$

طبق سوال قبل زمان اجرای خط لوله از رابطه $\left(n + m\left(k - ۱\right)\right)T$ بدست می‌آید که در آن $k = ۴$ و $m = ۰/۱\text{n}$ است. پس

$$\text{زمان اجرا خط لوله} = \left(n + ۰/۱n\left(۴ - ۱\right)\right) \times ۲۰ = ۲۶n$$

پس تسریع برابر است با:

$$\text{تسریع} = \frac{۶۰\text{n}}{۲۶n} = ۲/۳$$

پس گزینه «الف» صحیح است.

رویکرد فرض عدم پرش (تعلیق در صورت پرش)

زمان اجرای بدون خط لوله همانند رویکرد قبل برابر با $۶۰n$ است. فرمول زمان اجرای خط لوله همانند رویکرد قبل است با این تفاوت که چون در نیمی از موارد پرش اتفاق می‌افتد، تعداد دستورهای پرش که انجام می‌شوند برابر با $m = \frac{۰/۱}{۲}n = ۰/۰۵n$ است. لذا:

$$\text{زمان اجرا خط لوله} = \left(n + ۰/۰۵n\left(۴ - ۱\right)\right) \times ۲۰ = ۲۳n$$

پس تسریع برابر است با:

$$\text{تسریع} = \frac{۶۰n}{۲۳n} = ۲/۶$$

پس گزینه «ج» صحیح است.

پاسخ سوال ۶)

مشخص نیست که مقصود سوال از پیاده‌سازی معمولی تک سیکلی است یا چند سیکلی. این سوال را برای هر دو حالت حل می‌کنیم.

تک سیکلی:

$$\text{زمان اجرا بدون خط لوله} = \left(۱۰ + ۷ + ۱۰ + ۱۲ + ۷\right) \times n = ۴۶n$$

$$\text{زمان اجرا خط لوله} = \left(n + ۵ - ۱\right) \times ۱۲ = ۱۲n + ۴۸$$

$$\text{تسریع} = \frac{۴۶n}{۱۲n + ۴۸} \quad \overset{n \to \infty}{=} \quad ۳/۸$$

که در هیچ‌یک از این گزینه‌ها نیست.

چند سیکلی:

در این حالت زمان اجرای دستورات متفاوت است. زمان اجرای بدون خط لوله برابر است با:

$$\text{زمان اجرا بدون خط لوله} = ۴۶ \times ۰/۲n + ۳۴ \times ۰/۴n + ۳۹ \times ۰/۲n + ۲۷ \times ۰/۲n = ۳۶n$$

چون خط لوله با تعداد طبقات متفاوت داریم، زمان اجرای نوع‌های مختلف دستور با هم متفاوت است (در MIPS خط لوله تک با تعداد طبقات ثابت داشتیم)

گروه	درصد وقوع (درصد)	زمان اجرا
الف	۲۰	$۱۲\left(۵ + ۰/۲n - ۱\right)$
ب	۴۰	$۱۲\left(۴ + ۰/۴n - ۱\right)$
ج	۲۰	$۱۲\left(۴ + ۰/۲n - ۱\right)$
د	۲۰	$۱۲\left(۳ + ۰/۳n - ۱\right)$
زمان اجرای کل:		$۱۹۲ + \left(n - ۴\right) \times ۱۲$

در نتیجه تسریع برابر است با:

$$\text{تسریع} = \frac{۳۶n}{۱۹۲ + \left(n - ۴\right) \times ۱۲} \quad \overset{n \to \infty}{=} \quad ۳$$

پس گزینه «د» صحیح است.

موفق باشید

گروه حل تمرین

به نام خدا

دانشکده‌ی مهندسی برق و کامپیوتر– معماری کامپیوتر

تکلیف نخست – زمستان ۱۳۹۰ تاریخ تحویل: چهارشنبه هفدهم اسفند

۱) هر یک از توصیفات زیر مربوط به کدام یک از دسته‌های کامپیوتر (رومیزی، سرور، درون‌کار، ابررایانه) می‌باشد؟

آ: متشکل از صد تا هزار پردازنده، چندین ترابایت حافظه اصلی و چندین پتابایت حافظه ذخیره‌سازی است که از چندین میلیون دلار تا صدها میلیون قیمت دارد.

ب: تکامل بسیاری از تکنولوژی‌های محاسباتی از این دسته حاصل شده است.

پ: بزرگ‌ترین دسته‌ی کامپیوترها که یک برنامه‌ی خاص یا مجموعه‌ای خاص از برنامه‌های مرتبط با هم را اجرا می‌کنند.

ت: کامپیوترهای رومیزی که بدون صفحه‌نمایش و صفحه کلید هستند و معمولاً از طریق شبکه قابل دستیابی هستند.

ث: کامپیوترهایی که کارایی مناسبی را برای تک کاربر به ازای هزینه‌ی کمی رقم می‌زنند.

ج: اعتمادپذیری به آن‌ها بسیار مهم است و Crash در آنها هزینه بالایی را به دنبال دارد

چ: برای محاسبات سنگین و پیچیده مانند هواشناسی، کشف منابع نفتی و ... استفاده می‌شوند و بیشترین قابلیت در زمان خود را دارند.

ح: اگرچه بزرگ‌ترین دسته کامپیوترها هستند اما خیلی از افراد هنگام استفاده از آنها متوجه نمی‌شوند که در حال کار با یک کامپیوتر هستند.

۲) کارایی کامپیوتر به چه عواملی بستگی دارد؟ هر یک با تاثیر بر چه کمیتی از فرمول کارایی (تعداد دستور،CPI و فرکانس)، باعث تغییر در کارایی می‌شوند؟

۳) چرا MIPS (=Million Instruction Per Second) معیار مناسبی برای ارزیابی عملکرد نیست؟

٤) دو پیاده‌سازی متفاوت از مجموعه دستورالعمل (ISA) یکسان را در نظر بگیرید. چهار نوع رده دستورالعمل A، B، C و D وجود دارد که فرکانس و IPC (Instruction per Cycle) هر کدام در جدول زیر داده شده است:

پیاده‌سازی	فرکانس	IPC			
	GHz	A	B	C	D
P_1	۱/۵	۱	۰/۵	۰/۲۵	۰/۲۵
P_2	۲	۰/۵	۰/۵	۰/۵	۰/۵

آ) دستورالعمل‌های یک برنامه به رده‌های به صورت زیر تقسیم شده است:

۱۰٪از رده A، ۵۰٪از رده B، ۲۰٪از رده C، ۲۰٪ از رده D .

کدام پیاده‌سازی سریع‌تر است؟

ب) CPIمیانگین برای هر پیاده‌سازی چقدر است؟

پ) تعداد سیکل‌های ساعت (Clock Cycles) در صورتی که تعداد دستورالعمل‌ها باشد، برای هر دو پیاده‌سازی چند است؟

۵) دو پیاده‌سازی متفاوت P_1 و P_2 با مجموعه دستورالعمل یکسان را در نظر بگیرید. پنج رده دستورالعمل (A تا E) در مجموعه دستورالعمل وجود دارند و برای هر رده اطلاعات زیر داده شده است:

پیاده‌سازی	فرکانس GHz	CPI				
		A	B	C	D	E
P_1	۱	۱	۲	۳	٤	۳
P_2	۱/۵	۲	۲	۲	٤	٤

آ) اگر اوج (peak) کارایی به صورت «بیشترین نرخی که یک کامپیوتر می‌تواند دنباله‌ای از دستورالعمل‌ها را اجرا کند» تعریف شود، اوج کارایی P_1 و P_2 را بر حسب دستورالعمل بر ثانیه حساب کنید.

ب) اگر تعداد دستورالعمل‌هایی که در یک برنامه مشخص اجرا می‌شوند بین همه رده‌های دستورالعمل بجز رده A به صورت مساوی تقسیم شده باشند و در رده A تعداد دستورالعمل‌ها دو برابر بقیه باشد، کدام پیاده‌سازی سریع‌تر است؟

٦) اگر تعداد ترانزیستورهای استفاده شده در یک تراشه در سال ۱۹۸۰ برابر ٦٤۰۰۰ عدد باشد و تعداد آنها در سال ۱۹۸۳ در همان تراشه به ۲۵٦۰۰۰ عدد افزایش یابد، پیش‌بینی می‌کنید در سال ۲۰۲۲ تعداد ترانزیستورهای موجود در آن به چند عدد برسد؟

۷) فرض کنید نسخه‌های جدیدی از پردازنده‌ای با مشخصات زیر طراحی کرده‌ایم:

نسخه	ولتاژ (ولت)	فرکانس (گیگاهرتز)
۱	۵	۰/۵
۲	۳/۳	۱

آ) خازن بار (Load Capacitive) بین دو نسخه چند درصد کاهش می‌یابد اگر توان دینامیکی ۱۰درصد کاهش یابد؟

ب) با فرض اینکه خازن بار نسخه۲، ۸۰٪ خازن باز نسخه ۱ باشد، در صورتی که توان دینامیکی نسخه۲، ٤۰٪ نسبت به نسخه ۱ کاهش یابد، ولتاژ نسخه۲ را بیابید.

۸) جدول زیر انواع دستورالعمل تفکیک شده برای یک پردازنده را برای یک برنامه اجرا شده بر روی پردازنده های با تعداد هسته متفاوت نشان می دهد:

تعداد	تعداد دستورات ($\times 10^6$)				CPI			
پردازنده‌ها	FP	Int	L/S	Branch	FP	Int	L/S	Branch
۱	٥٦۰	۲۰۰۰	۱۲۸۰	۲٥٦	۱	۱	٤	۲
۸	۸۰	۲٤۰	۱٦۰	۳۲	۱	۱	٤	۲

فرض کنید هر پردازنده فرکانس ۲ گیگاهرتز دارد. در هر یک از پردازنده‌ها چقدر عملیات L/S را بهبود دهیم تا برنامه دو برابر سریع‌تر اجرا شود؟

۹) جدول زیر را برای دو پردازنده در نظر بگیرید.

پردازنده	فرکانس GHz	CPI
P_1	٤	۱/۲۵
P_2	۳	۰/۷۵

آ)یک تصور غلط این است که «یک پردازنده با فرکانس بیشتر را به عنوان پردازنده‌ای با کارایی بیشتر در نظر بگیریم.» این ادعا را با توجه به داده‌های داده شده بررسی کنید.

ب)یک تصور غلط دیگر این است که «پردازنده‌ای که دستورالعمل‌های بیشتری را انجام می‌دهد زمان اجرای بیشتری نیاز دارد.» با در نظر گرفتن اینکه پردازنده P_1 یک دنباله 10^6 دستورالعملی را اجرا میکند و CPI پردازنده‌ها تغییر نمی‌کند، تعداد دستورالعمل‌هایی که پردازنده P_2 اجرا می کند (در همان مدت زمان که P_1، 10^6 دستورالعمل را اجرا می‌کند) را حساب کنید.

پ) برای پردازنده‌های داده شده MIPS و کارایی را به دست آورید. چرا در این سوال MIPS بیشتر کارایی بالاتر را نتیجه می‌دهد؟

۱۰) در یک کامپیوتر شخصی پردازنده ۵۰٪ از زمان اجرا را به خود اختصاص می دهد. اگر بخواهیم قدرت محاسباتی پردازنده را ۵ برابر کنیم باید ۵ برابر بیشتر برای پردازنده هزینه کنیم. با فرض اینکه پردازنده یک سوم از هزینه کامپیوتر را به خود اختصاص دهد، آیا این ارتقا به صرفه خواهد بود؟

سربلند و پیروز باشید

سینا آقاسی، محسن فاریابی

تکلیف دوم – بهار ۱۳۹۱ تاریخ تحویل: چهارشنبه شانزدهم فروردین

۱- با فرض این که متغیرهای f و g و h به ترتیب در $s0 و $s1 و $s2 قرار دارند، و آدرس پایه‌ی آرایه‌های A و B در $t0 و $t1، برای کدهای زیر

آ) $f = -g + h + B[1]$

ب) $f = A[B[3]]$

معادل اسمبلی MIPS با کمترین تعداد دستور را به دست آورید. (همه متغیرها و پردازنده را ۳۲ بیتی فرض کنید)

۲- در زیر محتویات حافظه یک کامپیوتر که دارای پردازنده ۳۲ بیتی MIPS می‌باشد را مشاهده می‌کنید. پس از اجرای برنامه زیر، محتویات حافظه و رجیسترهای $t0 تا $t7 را در مبنای ۱۶ نمایش دهید.

```
.text
main:
lw $t0,0($0)
sb $t0,3($0)
sh $t0,3($0)
lh $t1,0($0)
lhu $t2,0($0)
lb $t3,0($0)
lbu $t4,0($0)
```

۷	0x8C
۶	0xCC
۵	0x4B
۴	0x00
۳	0x12
۲	0x56
۱	0x99
۰	0xFF

آ) فرض کنید پردازنده در حالت Big-Endian است.

ب) فرض کنید پردازنده در حالت Little-Endian است.

۳- یک برنامه‌نویس تازه کار در حال نوشتن یک اسمبلر برای زبان اسمبلی MIPS است. همانطور که می‌دانید، پردازنده‌های MIPS دارای دستور برای منفی کردن عدد نیستند! اما این برنامه‌نویس می‌خواهد شبه دستور زیر را به اسمبلر خود اضافه کند:

```
neg $r1,$r2
```

این شبه دستور $r2 را منفی کرده و در $r1 قرار می‌دهد (r2 و r1 نشان‌دهنده دو رجیستر هستند)

این برنامه‌نویس چه راه حل‌هایی پیش رو دارد؟ ۳ مورد

۴- با توجه به قالب دستورات MIPS در قطعه کد زیر، بین دو دستوری که با برچسب‌های back و here مشخص شده‌اند، حداکثر چند دستور می‌تواند قرار بگیرد؟

```
back:     add  $s0, $s3, $s4
            K instructions
here:     beq  $s0, $s2, back
```

۵- تکه‌ای از برنامه به صورت زیر است:

آدرس	دستورالعمل
0xFA003000	add $t1,$t2,$t3
0xFA003004	li $a0,1
0xFA003008	jal 0x1000
0xFA00300C	slt $t1,$v0,$zero

آ) آدرس دستوری که بعد از دستور jal اجرا می‌شود را بدست آورید.
ب) محتویات رجیستر $ra را پس از اجرای دستور jal بدست آورید.

۶- کد زیر به زبان C را در نظر بگیرید:

```
void main()
{
      int List[]={-7,8,9,96,3,0,7,-2};
      int n=8;
      int max=List[0];

      while ( --n != 0)
            max=Maximum(max,List[n]);
}
int Maximum ( int num1 , int num2)
{
      if ( num1 > num2)
            return num1;
      else
            return num2;
}
```

n همان تعداد اعضای درون List است. فرض کنید پردازنده ۳۲ بیتی است و تمامی متغیرها در برنامه بالا ۳۲ بیتی هستند. همچنین آدرس شروع سگمنت کد را 0xFF000000 و آدرس شروع سگمنت داده را 0x00000000 در نظر بگیرید.
آ) این کد را به زبان اسمبلی MIPS تبدیل کنید. از برچسب‌ها برای آدرس‌دهی استفاده کنید. (استفاده از شبه دستور مجاز است)
ب) کدی که اسمبلر از کد قسمت (آ) تولید می‌کند را بنویسید. (شبه‌دستور و برچسب مجاز نیست)
ج) پس از تشخیص نوع دستورهای قسمت قبل (R/I/J Type) کد معادل هر دستور را به تفکیک هر فیلد بنویسید. اعداد هر فیلد را در مبنای ۱۶ بنویسید.
د) با استفاده از قسمت (ج) کد زبان ماشین دستورهای قسمت (ب) را در مبنای ۱۶ بنویسید.

۷- در زیر کد ماشین یک برنامه آمده است. با فرض اینکه سگمنت کد از آدرس 0x00003000 شروع شود، کد ماشین زیر را به کد اسمبلی MIPS تبدیل کنید:

```
0x0C000C02
0x08000C0A
0x20020000
0x20090005
0x2129FFFF
0x00095080
0x8D4B0000
0x004B1020
0x1522FFFB
0x3E000008
```

۸- به زبان اسمبلی MIPS برنامه‌ای بنویسید که مجموع n جمله اول سری فیبونانچی را حساب کند. فرض کنید n در $s0 قرار گرفته باشد. این برنامه باید با تابع بازگشتی نوشته شود. (پیشنهاد می‌شود ابتدا کد C را بنویسید و پس از اطمینان از درستی عملکرد کد، آنرا به زبان اسمبلی تبدیل کنید)

$$fibo(n) = \begin{cases} ۱ & n = ۱و۲ \\ fibo\left(n - ۱\right) + fibo\left(n - ۲\right) & n \neq ۱و۲ \end{cases}$$

سربلند و پیروز باشید

گروه حل تمرین

تکلیف سوم – بهار ۱۳۹۱ تاریخ تحویل: چهارشنبه بیست و سوم فروردین

۱- با فرض داشتن یک Stack Machine، برنامه‌ای بنویسید که عبارت زیر را در این ماشین محاسبه کند.

$$X = (A * B - C)/D + (C/E + A - B * F)$$

۲- قسمتی از حافظه کامپیوتری که با پردازنده ۳۲ بیتی MIPS کار می‌کند به صورت زیر است:

آدرس	دستور
0xFFFF3000	addi $t0,$0,4
0xFFFF3004	lw $t1,$0(8)
0xFFFF3008	j 0x500
0xFFFF300C	beq $0,$0,3

برای هر یک از دستورهای فوق، نوع‌های نشانی‌دهی که دارند را مشخص کرده و آدرس موثر برای هر یک از آنها را مشخص کنید.

آ) دستور addi
ب) دستور lw
پ) دستور j
ت) دستور beq

۳- با استفاده از بلاک‌های ۴ بیتی Carry Select Adder یک جمع‌کننده ۱۶ بیتی Carry Select Adder بسازید و مدار آنرا ترسیم کنید. (از ترسیم داخل بلاک‌های ۴ بیتی صرفنظر کنید اما نام تمامی سینگالها باید واضح باشد)

۴- دو عدد ۱۴ و ۱۰ را با استفاده از الگوریتم‌های زیر در هم ضرب کنید. برای نمایش باینری اعداد از ۵ بیت استفاده کنید و تمامی مراحل را نمایش دهید.

آ) ضرب سریال (بهبود نیافته) (مراحل را همانند شکل ۷-۳ کتاب مرجع اصلی نمایش دهید)
ب) ضرب سریال بهبود یافته (برای نمایش از جدولی مشابه شکل ۷-۳ کتاب مرجع اصلی استفاده کنید)
پ) ضرب Booth (مراحل را همانند اسلاید ۱۰ جلسه دهم نمایش دهید)

۵- ضرب کننده شکل ۸-۳ کتاب را در نظر بگیرید. اگر بخواهیم از این روش استفاده کنیم و یک ضرب کننده سریع ۶۴ بیت در ۶۴ بیت بسازیم و برای ساخت جمع کننده‌ها تنها Full Adder در اختیار داشته باشیم و استفاده از هیچ گیتی مجاز نباشد،

آ) چند Full Adder لازم داریم؟
ب) اگر تاخیر Full Adder (از هر ورودی به هر خروجی) ۵ نانوثانیه باشد، تاخیر تولید محاسبه حاصلضرب را حساب کنید.

۶- با استفاده از الگوریتم‌های زیر ۱۲۰ را بر ۱۵ تقسیم کنید. برای نمایش ۱۲۰ از ۸ بیت و برای نمایش ۱۵ از ۴ بیت استفاده کنید.

آ) تقسیم بهبود نیافته که در شکل ۹-۳ کتاب مرجع اصلی مدار آن ترسیم شده است. (مراحل را همانند شکل ۱۱-۳ کتاب مرجع اصلی نمایش دهید)

ب) تقسیم بهبود یافته که در شکل ۱۲-۳ کتاب مرجع اصلی مدار آن ترسیم شده است. (برای نمایش از جدولی مشابه شکل ۱۱-۳ کتاب مرجع اصلی استفاده کنید)

پیروز و سربلند باشید

گروه حل تمرین

تکلیف چهارم – بهار ۱۳۹۱ تاریخ تحویل: چهارشنبه سی فروردین

۱- فرض کنید مقدار اولیه رجیسترها در یک پردازنده MIPS، صفر باشد. پس از اجرای دستورهای زیر، مقدار رجیسترهای $t2 تا $t5 چقدر خواهد بود؟

```
li $t0,0x7B012345
li $t1,256
multu $t0,$t1
mfhi $t2
mflo $t3
div $t0,$t1
mfhi $t4
mflo $t5
```

۲- فرض کنید برای نمایش اعداد ممیز شناور در سیستم کامپیوتری ساخته خود می‌خواهیم نسخه جدیدی از استاندارد IEEE 754 ایجاد کنیم. در این نسخه، یک بیت برای علامت، ۷ بیت برای نما و ۲۰ بیت برای مانتیس در نظر می‌گیریم. برای این نسخه موارد زیر را بدست آورید:

آ) بزرگترین عدد مثبت قابل نمایش

ب) کوچکترین عدد مثبت قابل نمایش

پ) بزرگترین عدد منفی قابل نمایش

ت) کوچکترین عدد منفی قابل نمایش

۳- اعداد زیر را به فرم استاندارد IEEE 754 و در مبنای ۱۶ نمایش دهید:

آ) $-\infty$ با دقت ساده

ب) ۱۴/۳۷۵ با دقت ساده

پ) $2^{-123} + 2^{-126}$

ت) ۷– با دقت مضاعف

۴- اعداد زیر در قالب اعداد اعشاری با دقت ساده IEEE 754 می‌باشند. آنها را از مبنای ۱۶ به دهدهی تبدیل کنید:

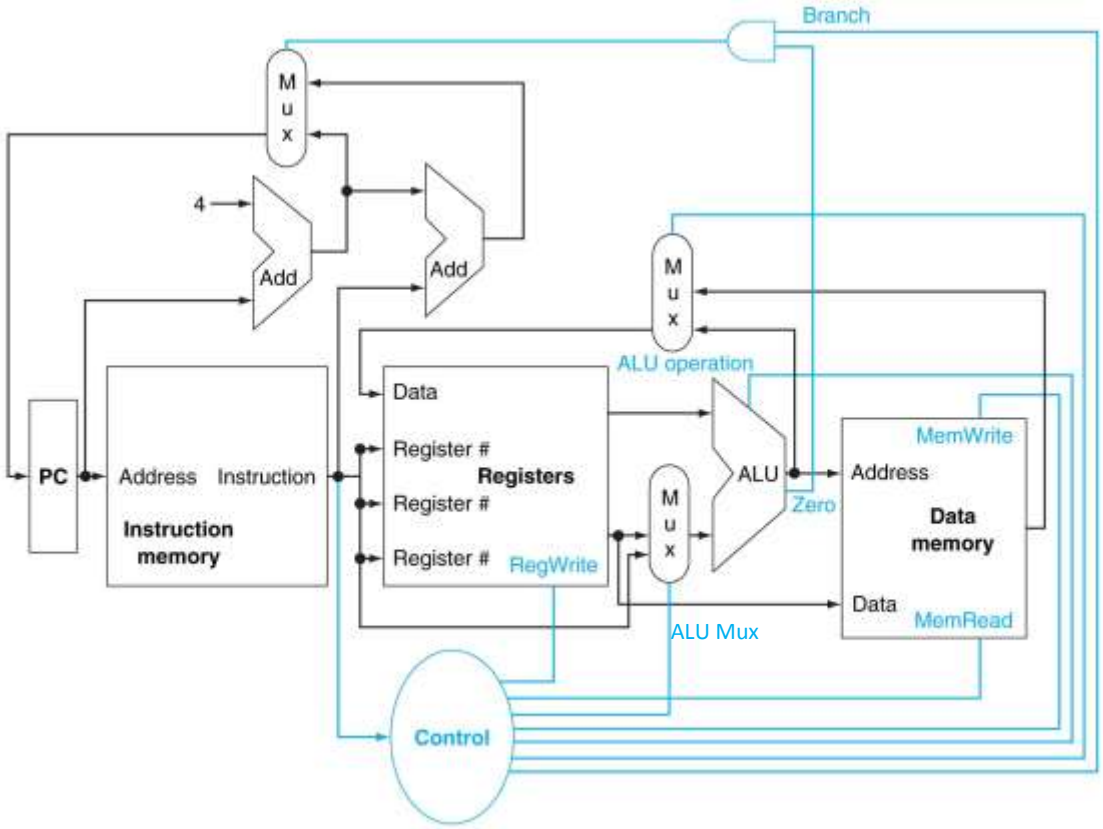
آ) 42E48000

ب) 3F880000

پ) 00800000

ت) C7F00000

۵- شکل زیر (شکل ۲-۴ از کتاب مرجع اصلی) نسخه ساده شده‌ای از پردازنده MIPS را نشان می‌دهد.



آ) برای دستورات زیر، مقدار سیگنال‌های کنترلی خواسته داده شده را مشخص کنید.

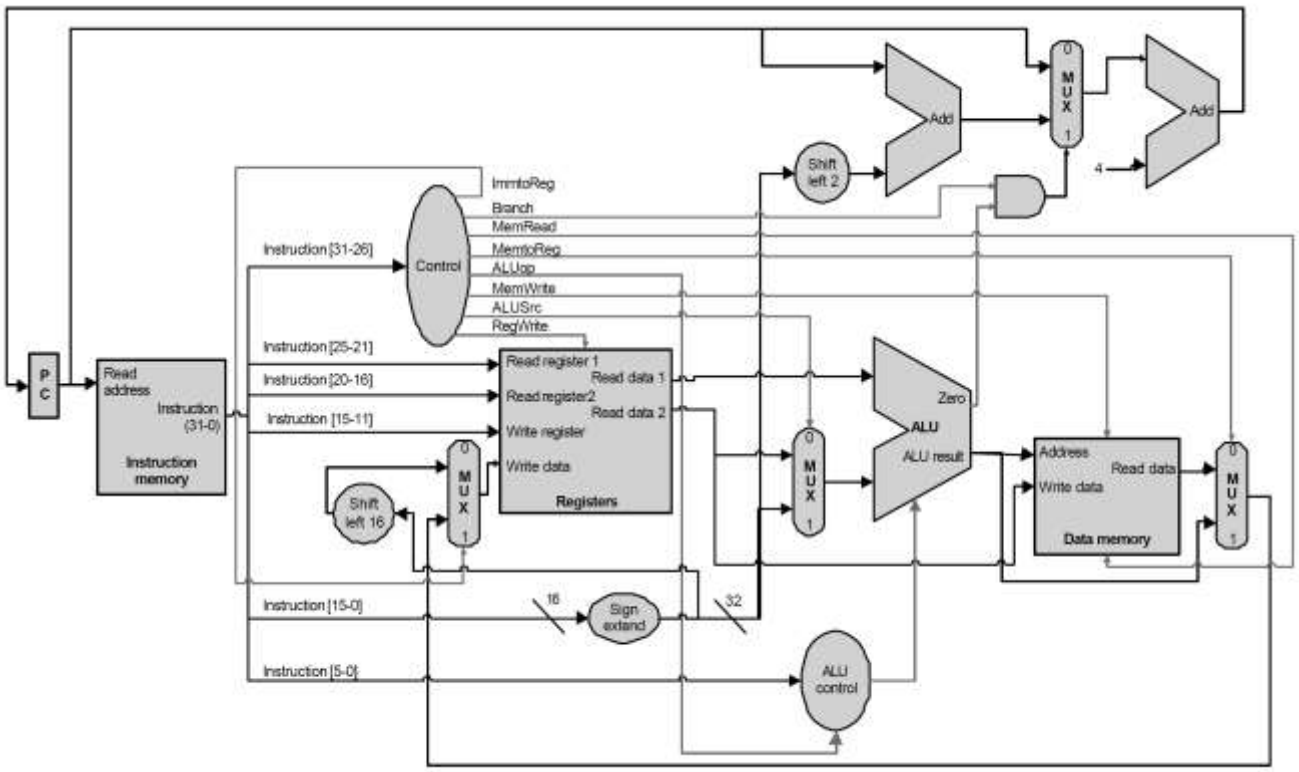
```
A. add Rd, Rs, Rt
B. lw Rt, offse(Rs)
C. beq Rs,Rt,immediate
```

	RegWrite	MemRead	ALUMux	MemWrite	Branch
A					
B					
C					

ب) پردازنده فوق دستور beq را اجرا می‌کند اما توانایی اجرای دستور bne را ندارد. می‌خواهیم دستور bne را به آن اضافه کنیم. شرط لازم برای افزودن یک دستور این است که Data Path قابلیت انجام آن دستور را داشته باشد. در شکل بالا، Data Path قابلیت اجرای دستور bne را ندارد. با کمترین تغییر، Data Path را به گونه‌ای تغییر دهید که بتواند دستور bne را اجرا کند. (توجه داشته باشید که دستورات قبلی باید همچنان قابل اجرا باشند)

پ) پردازنده فوق می‌تواند دستور j target را اجرا کند؟ چرا؟

۶- شکل زیر Data Path نسخه‌ای از MIPS را نشان می‌دهد:



از بین دستورات، با ذکر دلیل مشخص کنید کدام دستورات قابلیت اجرا ندارند؟

A. **add rd, rs, rt**
B. **lw rt, offset(rs)**
C. **j target**
D. **lui rt, imm**
E. **bne rs, rt, label**

۷- در سوال قبل، چگونه می‌توان Data Path را تغییر داد به گونه‌ای که از jr $rs پشتیبانی کند؟

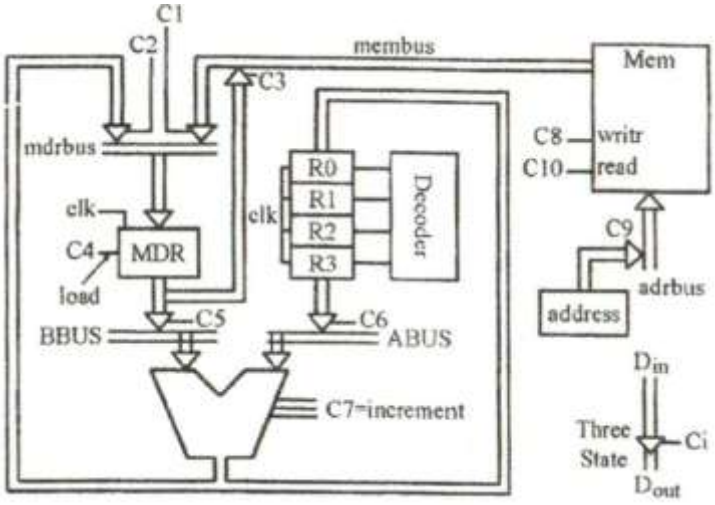
۸- شکل ۲۴-۴ از کتاب مرجع اصلی را در نظر بگیرید. بلاکهای Sign-Extend و Shift Left 2 را به بهینه‌ترین نحو ممکن طراحی کنید و مدار آنرا ترسیم کنید.

پیروز و سربلند باشید

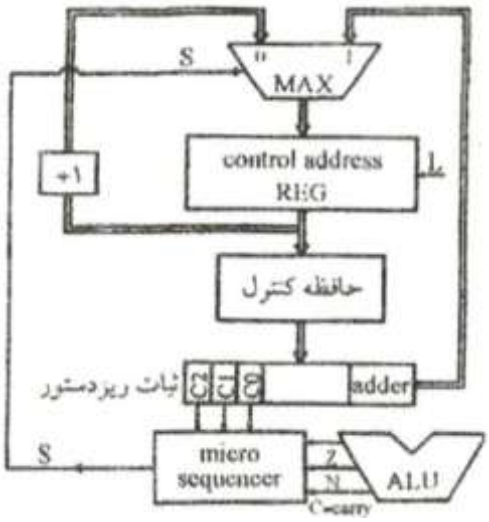
گروه حل تمرین

تکلیف پنجم – بهار ۱۳۹۱ تاریخ تحویل: دوشنبه هجدهم فروردین

۱- با در نظر گرفتن ساختار زیر مقدار سینگالهای کنترلی را در سه کلاک متوالی برای Increment کلمه‌ای از حافظه تعیین کنید. (فرض کنید که واکشی دستور Increment قبلاً انجام شده است و هر کلاک می‌تواند شامل چند فاز جزئی باشد.)

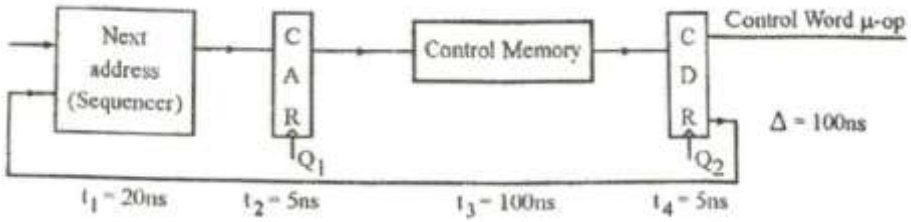


۲- [اختیاری] شکل زیر قسمتی از مدار کنترل ریزبرنامه‌ریزی شده یک کامپیوتر را نشان می‌دهد. معادله S، خروجی microsequencer را بر حسب ورودی‌هایش (Z، N، Carry ، C_0، C_1 و C_2) بنویسید:



$C_2C_1C_0$	معنی
000	ریز دستور بعدی
001	jump if N=1
010	jump if Z=1
011	jump
100	jump if C=1
101	jump if N=1
110	jump if Z=1
111	jump

۳- [اختیاری] نمودار یک واحد کنترل به روش ریزبرنامه‌ریزی (Microprogram) نشان داده شده است. tها تاخیر در اجزا و Δ تاخیر در اجرای ریزعمل ($\mu - Op$) می‌باشد. حداقل پریود کلاک را محاسبه کنید. CAR (ثبات آدرس کنترل)، آدرس ریز دستور را مشخص می‌کند. CDR (ثبات داده کنترل) ریزدستوری که از حافظه کنترل خوانده می‌شود را در خود ذخیره می‌کند. (Q_1 و Q_2 ورودی کلاک می‌باشد که دارای اختلاف فاز می‌باشند)



۴- فرض کنید Datapath یک پردازنده تک سیکلی از S بخش متوالی با تاخیر یکسان تشکیل شده باشد. اگر همین پردازنده را بخواهیم با تکنیک خط لوله پیاده‌سازی کنیم، پردازنده جدید S طبقه خط لوله خواهد داشت. گفته می‌شود در این حالت بیشینه تسریع برابر است با تعداد طبقات خط لوله یعنی S . این ادعا را ثابت کنید.

۵- چرا با وجود اینکه خط لوله زمان اجرای یک دستور را افزایش می‌دهد، استفاده از خط لوله یک مزیت محسوب می‌شود؟

۶- فرض کنید کامپیوتری دارای پردازنده‌ای با s طبقه خط لوله باشد. کامپیوتر را روشن می‌کنیم. پس از روشن شدن کامپیوتر برنامه‌ای با n دستور در حافظه بارگذاری شده و اجرا می‌شود. اگر از زمان بارگذاری برنامه در حافظه صرفنظر کنیم، چند سیکل طول می‌کشد تا برنامه به کار خود پایان دهد؟

پیروز و سربلند باشید

گروه حل تمرین

تکلیف ششم – بهار ۱۳۹۱ تاریخ تحویل: چهارشنبه سوم خرداد

۱- کد زیر را در نظر بگیرید:

```
sub $2, $1,$3
and $12,$2,$5
or  $13,$6,$2
nor $14,$2,$2
sw  $15,100($2)
lw  $15,80($13)
xor $8,$8,$15
add $7,$15,$3
```

آ) هر یک از مخاطراتی که در اجرای کد اتفاق می‌افتد را مشخص کرده و نوع آنها را بنویسید (ساختاری، داده‌ای، کنترلی)

ب) در صورتی که تنها راهکار، تعلیق خط لوله باشد، اجرای این کد به چند سیکل نیاز دارد؟ نمودار چند سیکل ساعتی خط لوله (Multiple-Clock-Cycle Pipeline Diagram) را ترسیم کنید.

پ) با استفاده از پیش‌فرستادن داده (Forwarding) به چند سیکل نیاز دارد؟ نمودار چند سیکل ساعتی خط لوله (Multiple-Clock-Cycle Pipeline Diagram) را ترسیم کنید.

۲- در یک پردازنده دستورها به صورت خط لوله اجرا می‌شوند که دارای سه ایستگاه (Stage) به قرار زیر است:

- ایستگاه اول برای خواندن دستورها از حافظه
- ایستگاه دوم برای اجرای دستور
- ایستگاه سوم برای ذخیره حاصل در حافظه یا بارگیری از حافظه

دو روش را با هم مقایسه می‌کنیم. روش اول حافظه دستور و داده مشترک است و روش دوم حافظه دستور و داده مجزاست. اگر قطعه برنامه‌ای دارای ۱۰۰ دستور باشد که همگی رجوع به حافظه دارند و هیچ نوع وابستگی داده بین آنها نیست، آنگاه نسبت زمان اجرای روش اول را به روش دوم حساب کنید.

۳- یک برنامه شامل دو حلقه‌ی تو در توست که در انتهای هر حلقه یک پرش شرطی وجود دارد. با هر بار اجرای حلقه خارجی، حلقه داخلی ۲۰ بار اجرا می‌شود. حلقه خارجی نیز خود ۱۰ بار اجرا می‌گردد. میزان خطای پیش‌بینی را به ازای سه راهکار مطرح شده به دست آورید:

آ) همیشه پرش خواهیم داشت

ب) استفاده از یک بیت برای پیش‌بینی

پ) استفاده از دو بیت برای پیش‌بینی

۴- اگر یک خط لوله‌ی سه مرحله‌ای را به چهار مرحله‌ای تبدیل کنیم، پریود ساعت از T به $0.9T$ کاهش می‌یابد. (از تأخیر ثبات‌های بین دو مرحله چشم‌پوشی می‌کنیم)

آ) زمان اجرای n دستور چه تفاوتی می‌کند؟

ب) در صورتی که ۳۰ درصد دستورها پرشی باشند، پاسخ چه تفاوتی خواهد داشت؟ (فرض بر این است تا پایان اجرای یک دستور شرطی حباب وارد خط لوله نمی‌شود). آیا ترتیب دستورات پرشی مؤثر است؟

پ) میزان دستورهای پرشی چقدر باشد، تا در حالت چهار مرحله‌ای افت سرعت نداشته باشیم؟

۵- کد زیر قرار است بر روی پردازنده خط لوله MIPS که دارای ۵ مرحله بدون پیش‌فرستادن می‌باشد، کار کند:

```
    and $7,$8,$0
L1: add $1,$7,$3
    xor $5,$7,$6
    or $12,$1,$4
    add $18,$19,$24
    beq $3,$12,L1
    add $18,$19,$24
```

فرض کنید از روش انشعاب تاخیر یافته (Delayed Branch) استفاده کرده باشیم و شرط دستور انشعاب برقرار نگردد:

آ) مراحل اجرای این کد را بر روی خط لوله نشان داده و بگویید اجرای این کد چند سیکل طول می‌کشد؟

ب) با تعویض جای دستورهای فوق، سعی کنید زمان اجرا را کاهش دهید و زمان اجرای جدید را محاسبه کنید.

۶- قطعه کد زیر به زبان C++ را در نظر بگیرید:

```
do
{
      i --;
      sum += i;
} while( i  != 0);
goto Exit;
```

فرض کنید می خواهیم کد بالا را به زبان اسمبلی پردازنده خط لوله MIPS که از انشعاب تاخیر یافته و پیش‌فرستادن استفاده می‌کند، کامپایل کنیم. یک برنامه‌نویس اسمبلی که با انشعاب تاخیر یافته و پیش‌فرستادن آشنایی ندارد کد زیر را نوشته است:

```
L: addi $t0,$t0,-1
   nop
   nop
   add $t1,$t1,$t0
   bne $t0,$0,L
   nop
   j Exit
```

آ) کد بالا را تا حد امکان بهینه کنید. (با جابه‌جایی دستورها و حذف nop)

ب) اگر مقدار اولیه $t0 برابر n باشد، تسریعی که از تغییرات قسمت (آ) حاصل شده است را بر حسب n حساب کنید.

۷- در صورت بروز استثناء چه روندی در پردازنده خط لوله MIPS رخ می‌دهد، گام به گام توضیح دهید.

پیروز و سربلند باشید

گروه حل تمرین

۱- یک DRAM دارای $۲۵۶K$ سطر می‌باشد. اگر مدت زمانی که طول می‌کشد تا یک سطر تازه‌سازی شود $۵۰ns$ بوده و هر سطر حداقل هر $۴۰ms$ نیاز به تازه‌سازی داشته باشد، چند درصد از پهنای باند هدر می‌رود؟

۲- کد زیر را در نظر بگیرید اولین دستور آن در خانه صفر حافظه دستور قرار گرفته است:

```
addi $t1,$t1,100
L: addi $t1,$t1,-1
I Instructions here
bne $t1,$0,L
```

این کد را بر روی پردازنده ۳۲ بیتی MIPS که دارای یک سطح حافظه دستورالعمل ۱ کیلوبایتی با بلاک‌هایی به طول ۳۲ کلمه و نگاشت مستقیم می‌باشد، اجرا می‌کنیم. اگر I_{max} را بیشینه مقدار I بنامیم بطوریکه در اجرای برنامه بالا نرخ فقدان حافظه نهان دستورالعمل کمینه باشد، I_{max} را بیابید. فرض کنید هیچیک از I دستور، دستور انشعاب (شرطی یا غیرشرطی) نبوده و مقدار $\$t1$ را تغییر نمی‌دهند.

۳- سوال بالا را در نظر بگیرید. اگر تعداد دستورها از I_{max} بیشتر شود، زمان اجرای برنامه به دلیل افزایش نرخ فقدان، به شدت افزایش می‌یابد. یکی از راه حل‌هایی که کامپایلرهای پیشرفته امروزی برای حل این مشکل استفاده می‌کنند، تکنیک Loop Fusion است.

در تکنیک Loop Fusion یک حلقه به دو یا چند حلقه با همان محدوده اندیس شکسته شده و دستورهای حلقه اولیه بین حلقه‌های حاصل توزیع می‌شوند (به شرطی که دستورها به هم وابسته نباشند). برای مثال حلقه زیر به دو حلقه شکسته می‌شود:

```
int i, a[100], b[100];
 for (i = 0; i < 100; i++) {
   a[i] = 1;
   b[i] = 2;
 }
```

→

```
int i, a[100], b[100];
 for (i = 0; i < 100; i++) {
   a[i] = 1;
 }
 for (i = 0; i < 100; i++) {
   b[i] = 2;
 }
```

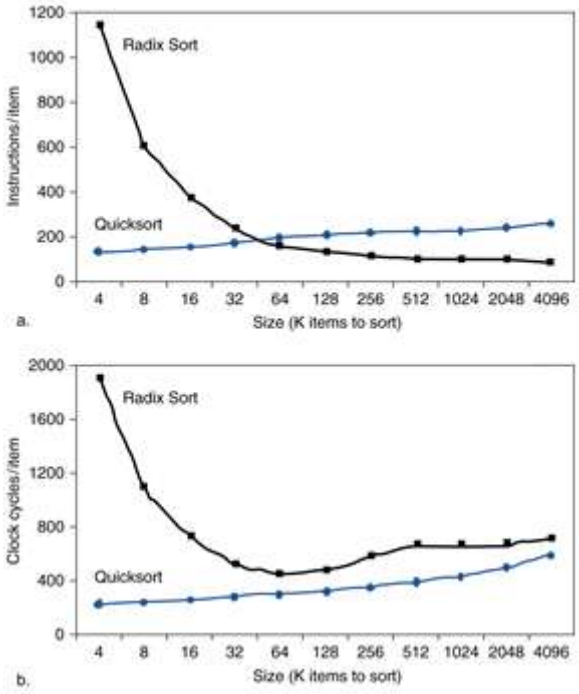
فرض کنید در سوال بالا $I = I_{max} + ۱$. زمان اجرای برنامه را برای حالات زیر محاسبه کنید:

آ) بدون استفاده از تکنیک Loop Fusion

ب) با استفاده از تکنیک Loop Fusion و شکستن به دو حلقه با $I - ۱$ دستور و یک دستور

از فقدان حافظه داده و تعلیق‌های دیگر چشم‌پوشی کنید. نرخ ساعت $۱GHz$، جریمه فقدان ۱۰۰۰ سیکل و CPI برای بقیه دستورها یک می‌باشد.

۴- شکل زیر شکل ۵-۱۸ از کتاب مرجع اصلی انتخاب شده است:



چرا با وجود اینکه به ازای Kهای بزرگ الگوریتم مرتب‌سازی سریع به تعداد دستورهای بیشتری برای اجرا نیاز دارد (نمودار اول) ، زمان اجرای کمتری نسبت به مرتب‌سازی مبنایی نیاز دارد؟(نمودار دوم) در روشهای کلاسیک و استاندارد تحلیل الگوریتم‌ها از تاثیر چه عاملی چشم‌پوشی می‌شود که نمی‌تواند این نتیجه را توجیه کند؟

۵- یک حافظه نهان یک سطحی شامل شانزده بلوک ۱۰۲۴ کلمه‌ای است. حافظه اصلی دارای 2^{20} کلمه است که در آن آدرس‌دهی بر مبنای کلمه صورت می‌پذیرد. تعداد بیت‌های حافظه‌ی مورد نیاز برای طراحی این حافظه نهان را برای حالت‌های زیر حساب کنید. از بیت‌های لازم برای نگهداری اطلاعات مربوط به LRU چشم‌پوشی کنید.

آ) نگاشت مشتقیم

ب) نگاشت انجمی کامل

پ) نگاشت شبه‌انجمی با درجه آزادی ۲

۶- یک حافظه‌ی نهان شامل چهار بلوک یک کلمه‌ای در اختیار داریم.با این فرض که حافظه‌ی نهان در ابتدا خالیست جدول زیر را پر کنید. برای جایگزینی یک بلوک از شیوه‌ی LRU استفاده کنید. (ستون اول درخواست‌های انجام شده از حافظه می‌باشد)

آدرس بلوک مورد نظر	Direct Map		Fully associative		2-way set associative	
	Hit or Miss	شماره بلوک	Hit or Miss	شماره بلوک	Hit or Miss	شماره بلوک
۰						
۴						
۸						
۰						
۴						
۸						

۷- در پردازنده‌ای مقدار مبنای CPI برابر با یک است. این پردازنده که با فرکانس $۲GHz$ کار می‌کند؛ از حافظه نهان دو سطحی برای داده استفاده می‌کند. هر دسترسی به حافظه اصلی ۴۰۰ سیکل و هر دسترسی به حافظه نهان سطح دوم ۱۰ سیکل مصرف می‌کند. اگر در برنامه‌ای، نرخ فقدان در حافظه نهان سطح اول ۵ درصد و نرخ فقدان حافظه نهان سطح دوم ۲ درصد باشد، CPI موثر را محاسبه نمایید.

پیروز و سربلند باشید

گروه حل تمرین

پاسخ سوال ۱)

$$\frac{۲۵۶ \times ۲^{۱۰} \times ۵۰ns}{۴۰ms} \times ۱۰۰ = ۳۲/۷۶۸ \text{ درصد}$$

پاسخ سوال ۲)

نرخ فقدان زمانی کمینه است که دستورهایی که وارد حافظه نهان شده‌اند در طول اجرای برنامه از حافظه نهان خارج نشوند. برای این امر باید هیچیک از دو بلوک دستور، به یک بلوک از حافظه نهان نگاشت نشوند، یعنی باید تعداد کل دستورهای برنامه از اندازه حافظه نهان کوچکتر باشد.

اندازه حافظه نهان یک کیلوبایت یعنی ۲۵۶ کلمه می‌باشد. هر دستور نیز یک کلمه است پس ظرفیت حافظه نهان ۲۵۶ دستور است. پس:

$$I_{max} = ۲۵۶ - ۳ = \mathbf{۲۵۳}$$

توجه: اگر اولین دستور از خانه صفر حافظه شروع نشود باید فاصله اولین دستور برنامه تا ابتدای بلاکی که در آن قرار می‌گرفت از عدد بالا کم شود. مثلاً اگر آدرس اولین دستور در خانه ۲۰ حافظه بود، از ابتدای بلوک متناظرش که بلوک صفر می‌باشد به اندازه ۱۲ بایت یعنی $\frac{۲۰}{۴} = ۵$ کلمه فاصله دارد لذا $I_{max} = ۲۵۶ - ۳ - ۵ = ۲۴۸$. و اگر اولین دستور در خانه ۱۵۲ از حافظه اصلی قرار داشت، از ابتدای بلوک متناظرش که ۱ می‌باشد به اندازه ۶ کلمه فاصله داشت، در نتیجه $I_{max} = ۲۵۶ - ۳ - ۶ = ۲۴۷$

پاسخ سوال ۳)

برای محاسبه زمان اجرا کافی است تعداد سیکلهای لازم برای اجرای برنامه را بدون تاثیر سلسله مراتب حافظه با تعداد سیکلهای تعلیق به خاطر فقدان جمع کرده و در پریود کلاک ضرب کنیم.

برای فهم بهتر سوال، در سه حالت زمان اجرا را حساب می‌کنیم:

حالت اول $I = I_{max}$

تمامی دستورها در حافظه نهان جای می‌گیرند. ابتدا از تاثیر سلسله مراتب حافظه صرفنظر کنیم. دستور اول که مقدار ۱۰۰ را در $t1 قرار می‌دهد، یک بار اجرا می‌شود اما بقیه دستورها به تعداد دفعات اجرای حلقه یعنی ۱۰۰ بار اجرا می‌شوند. لذا تعداد سیکلها برابر است با $۲۵۵ \times ۱۰۰ + ۱$

برای محاسبه تعداد سیکلهایی که صرف انتقال به حافظه نهان شده است کافی است تعداد فقدان‌ها را حساب کرده و در جریمه فقدان ضرب کنیم. جریمه فقدان طبق سوال ۱۰۰۰ سیکل است. چون اندازه برنامه با اندازه حافظه نهان برابر است برای هر بلوک از دستورها یک بار فقدان داریم. کلاً ۲۵۶ دستور داریم و بلوک‌ها ۳۲ دستوری هستند پس ۸ بلوک داریم. پس تعداد فقدان‌ها ۸ است. لذا تعداد سیکل‌های جریمه برابر با ۸ × ۱۰۰۰ است. پس زمان اجرا برابر است با:

$$CPU\ Time = \left(۱ + ۲۵۵ \times ۱۰۰ + ۱۰۰۰ \times ۸\right) \times ۱ns = ۳۳/۵۰۱\mu s$$

آ) حالت دوم، $I = I_{max} + ۱$ بدون استفاده از *Loop Fusion*

تعداد سیکل اجرا بدون در نظر گرفتن تاثیر سلسله مراتب حافظه همان تعدادی است که در قسمت قبل محاسبه شد. اما برای محاسبه تعداد فقدان‌ها. دستورها ۲۵۷ دستور هستند. در نتیجه ۹ بلاک را اشغال خواهند کرد. اما حافظه نهان دارای ۸ بلاک است. چون دستورها پشت سر هم هستند و نگاشت مستقیم است، پس بلاک صفر و ۸ دستور به یک نقطه از حافظه نهان نگاشت می‌شوند. پس بلاک‌های ۱ تا ۷ از دستور تنها یکبار فقدان خواهند داشت، اما بلاک‌های صفر و ۸ به تعداد دفعات اجرای حلقه فقدان خواهند داشت. پس تعداد فقدان‌ها ۲ × ۱۰۰ + ۷ و جریمه آن ۱۰۰۰ × $\left(۲ \times ۱۰۰ + ۷\right)$ می‌باشد. پس زمان اجرا برای این حالت برابر است با:

$$CPU\ Time = \left(۱ + ۲۵۵ \times ۱۰۰ + ۱۰۰۰ \times \left(۷ + ۱۰۰ \times ۲\right)\right) \times ۱ns = ۲۳۲/۵۰۱\mu s$$

ب) حالت سوم، $I = I_{max} + ۱$ با استفاده از *Loop Fusion* و شکستن به دو حلقه با $I - ۱$ دستور و یک دستور

اگر از تکنیک Loop Fusion استفاده کنیم باید یک حلقه را به دو حلقه با همان محدوده اندیسی بشکنیم. لذا ساختار دو حلقه مشابه و به صورت زیر است:

```
addi $t1,$t1,100
L: addi $t1,$t1,-1
253 Instructions here
bne $t1,$0,L
```

```
addi $t1,$t1,100
L: addi $t1,$t1,-1
1 Instruction here
bne $t1,$0,L
```

در واقع کد بالا همانند دو برنامه مجزا عمل می‌کند. قسمت اول همان حالت اول است که در بالا بررسی شد! قسمت دوم نیز مشابه است و به صورت زیر حساب می‌شود:

تعداد سیکل‌های ساعت بدون در نظر گرفتن سلسله مراتب حافظه برابر است با ۱۰۰ × ۳ + ۱. تنها یک بار فقدان داریم. پس سیکل جریمه ۱ × ۱۰۰۰ است. پس تعداد سیکل لازم برای قسمت دوم ۱۳۰۱ = ۱ × ۱۰۰۰ + ۱۰۰ × ۳ + ۱می‌باشد. زمان اجرای این حالت برابر است با:

$$CPU\ Time = ۳۳/۵۰۱\mu s + ۱۳۰۱ \times ۱ns = ۳۴/۸۰۲\mu s$$

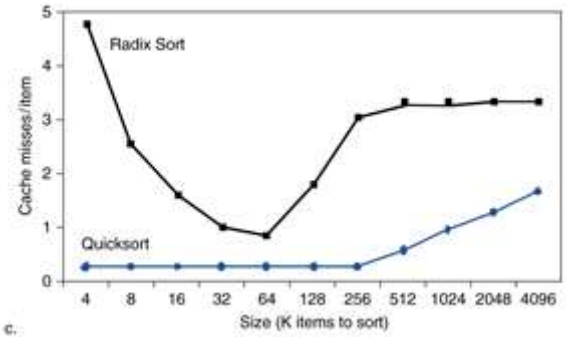
اگر مقدار دو حالت دوم و سوم را مقایسه کنید، مشاهده می‌کنید که در حالت دوم زمان اجرا تقریباً ۲۳۳ میکروثانیه است در حالیکه در حالت سوم حدود ۳۵ میکروثانیه است یعنی با وجود اینکه تعداد دستورهای حالت سوم بیشتر است و برخی دستورها به جای ۱۰۰ بار، ۲۰۰ بار اجرا می‌شوند، اما بازهم زمان اجرای حالت سوم کمتر است. نسبت زمان اجرای حالت دوم و سوم حدود ۷ برابر است.

این مثال تاثیر آشنایی با سخت‌افزار در کارایی نرم‌افزارها را نشان می‌دهد. (و این یکی از دلایل نیاز به یادگیری معماری کامپیوتر برای دانشجویان گرایش نرم‌افزار است)

پاسخ سوال ۴)

زمان اجرا تحت تاثیر سه عامل فرکانس، CPI و تعداد دستورها می‌باشد. در یک پردازنده ثابت، فرکانس برای هر دو الگوریتم یکسان است. اگر چه مرتب‌سازی مبنایی دارای تعداد دستورهای کمتری است، اما دارای CPI میانگین بیشتری است و بیشتر بودن CPI به حدی است که زمان اجرای آنرا بیشتر می‌کند.

دلیل بیشتر بودن CPI، وجود فقدان‌های بیشتر در اجرای مرتب‌سازی مبنایی است. قسمت آخر از همان شکل ۵-۱۸ این مطلب را نشان می‌دهد:



اگر فقدان اتفاق افتد، باید تعدادی سیکل تلف شود تا داده مورد نظر از حافظه اصلی به حافظه نهان انتقال یابد و این به معنی افزایش CPI برای اجرای یک دستور است. در واقع عاملی که روشهای کلاسیک و استاندارد تحلیل الگوریتم‌ها از آن چشم‌پوشی می‌کنند، **تاثیر سلسله مراتب حافظه در زمان اجرای یک الگوریتم** است! (این نیز دلیل دیگری برای نیاز به یادگیری معماری کامپیوتر برای دانشجویان گرایش نرم‌افزار است)

پاسخ سوال ۵)

حافظه مجازی هر نگاشتی که داشته باشد، هر سطر آن دارای ساختار زیر است (طبق صورت سوال از بیت‌های LRU چشم‌پوشی شد) :

Valid Bit	Tag	Block Data

در این سوال، حافظه نهان یک حافظه ۱۶ سطری ۱۰۲۴ کلمه‌ای است. یعنی حافظه نهان این سوال ۱۶ سطر همانند شکل بالا دارد که بخش Block Data آن ۱۰۲۴ کلمه‌ای است. پس اگر تعداد بیت‌های لازم برای Tag را T و اندازه هر کلمه را w فرض کنیم، اندازه حافظه نهان برابر است با:

$$\text{اندازه حافظه نهان} = ۱۶ \times \left(۱ + T + ۱۰۲۴w\right)$$

در حالت کلی آدرس یک خانه از حافظه را می‌توان به فیلدهای زیر تقسیم کرد:

Tag	Set	Block Offset	Byte Offset

در این سوال حافظه اصلی ما دارای واحد آدرس‌پذیر کلمه است. پس Byte Offset دارای صفر بایت خواهد بود. تعداد کلمات داخل یک بلاک ۱۰۲۴ کلمه است پس تعداد بیت‌های Block Offset برابر با $\log_۲ ۱۰۲۴ = ۱۰$ است. تعداد بیت‌های Set به درجه آزادی بستگی دارد و اگر درجه آزادی a باشد تعداد بیت‌های لازم برای فیلد Set برابر با $\log_۲ \frac{۱۶}{a}$ خواهد بود. همچنین حافظه اصلی دارای $۲^{۲۰}$ کلمه است و آدرس‌دهی بر مبنای کلمه است (واحد آدرس‌پذیر کلمه است نه بایت) پس اندازه آدرس برابر با ۲۰ بیت است. پس تعداد بیت‌های لازم برای Tag برابر است با:

$$T = ۲۰ - ۰ - ۱۰ - \log_۲ \frac{۱۶}{a}$$

آ) نگاشت مستقیم:

نگاشت مستقیم همان نگاشت شبه‌انجمنی با درجه آزادی (درجه انجمنی) یک است. لذا $a = ۱$

با جایگذاری در فرمول $T = ۶$ پس

$$\text{اندازه حافظه نهان} = ۱۶ \times \left(۱ + ۶ + ۱۰۲۴w\right)$$

آ) نگاشت انجمنی کامل:

نگاشت انجمنی کامل همان نگاشت شبه‌انجمنی است با درجه آزادی ۱۶ (برای این سوال). لذا $a = ۱۶$. پس $T = ۱۰$ در نتیجه:

$$\text{اندازه حافظه نهان} = ۱۶ \times \left(۱ + ۱۰ + ۱۰۲۴w\right)$$

آ) نگاشت شبه‌انجمنی با درجه آزادی ۲:

$a = ۲$. پس $T = ۷$. لذا

$$\text{اندازه حافظه نهان} = ۱۶ \times \left(۱ + ۷ + ۱۰۲۴w\right)$$

پاسخ ۶)

نگاشت مستقیم:

در این نگاشت شماره بلوک در حافظه نهان برابر است با باقیمانده آدرس بلوک در حافظه اصلی بر تعداد بلاک‌ها در حافظه نهان. تعداد بلاک‌ها در حافظه نهان طبق صورت سوال ۴ است. لذا داریم:

آدرس بلوک	شماره بلوک
۰	$۰ \bmod ۴ = ۰$
۴	$۴ \bmod ۴ = ۰$
۸	$۸ \bmod ۴ = ۰$
۰	$۰ \bmod ۴ = ۰$
۴	$۴ \bmod ۴ = ۰$
۸	$۸ \bmod ۴ = ۰$

چون حافظه نهان در ابتدا خالی است، پس درخواست اول همواره با فقدان روبه‌رو می‌شود. درخواست دوم بلوک ۴ از حافظه اصلی را درخواست می‌کند که باید در بلوک ۰ از حافظه نهان قرار گرفته باشد که وجود ندارد. لذا با فقدان رو به رو می‌شود. درخواست سوم نیز همانند درخواست دوم با فقدان رو به رو می‌شود. درخواست سوم بلوک صفر از حافظه اصلی را می‌خواهد که باید در خانه ۰ از حافظه نهان قرار داشته باشد، اما در این خانه، بلوک ۸ از حافظه اصلی قرار دارد، لذا بازهم فقدان داریم. دو درخواست دیگر نیز با فقدان رو به رو می‌شوند.

نگاشت انجمنی:

در این نگاشت هر بلوک از حافظه اصلی می‌تواند در هر یک از بلوک‌های حافظه نهان باشد یا قرار گیرد. طبق صورت سوال، سیاست جانشینی، LRU است یعنی بلاکی از حافظه نهان جانشین می‌شود اخیراً کمتر استفاده شده باشد. در این سوال برای درخواست‌ها داریم:

۱. بلاک صفر از حافظه اصلی درخواست می‌شود حافظه نهان خالی است پس فقدان اتفاق می‌افتد. اکنون باید بلاک درخواستی از حافظه اصلی به حافظه نهان منتقل شود. چون حافظه نهان خالی است، هر جایی از آن می توان این بلاک را قرار داد. ما خانه صفر را انتخاب می‌کنیم (معمولاً در کامپیوتر شروع از صفر است)

۲. بلاک ۴ از حافظه اصلی درخواست می‌شود اما در حافظه نهان وجود ندارد پس فقدان داریم. پس از انتقال بلاک درخواستی از حافظه اصلی باید آنرا در بلاک مناسب از حافظه نهان قرار دهیم. بلاک صفر از حافظه نهان پر است اما بلاک ۱ و ۲ خالی است. طبق سیاست LRU بلاک ۴ حافظه اصلی در بلاک ۱ حافظه نهان قرار می‌گیرد.

۳. بلاک ۸ از حافظه نهان درخواست می‌شود. بازهم فقدان . بلاک‌های ۱ و ۲ از حافظه نهان پر است. طبق سیاست LRU این بلاک حتماً باید در بلاک ۲ حافظه نهان قرار گیرد.

۴. اکنون بلاک صفر حافظه اصلی درخواست می‌شود که در بلاک صفر از حافظه نهان قرار دارد. لذا برخورد اتفاق می‌افتد.

۵. بلاک ۴ از حافظه اصلی درخواست می‌شود که در بلاک ۱ از حافظه نهان است. پس برخورد داریم.

۶. بلاک ۸ از حافظه اصلی درخواست می‌شود که در بلاک ۲ از حافظه نهان جای گرفته است. بنابراین بازهم برخورد داریم.

آدرس بلوک	شماره بلوک	Hit/Miss
۰	۰	Miss
۴	۱	Miss
۸	۲	Miss
۰	۰	Hit
۴	۱	Hit
۸	۲	Hit

نگاشت شبه‌انجمنی با درجه آزادی ۲:

در این نگاشت ۴ بلاک از حافظه نهان به دو مجموعه دو بلاکی تقسیم می‌شود. باقیمانده تقسیم آدرس بلاک بر تعداد مجموعه‌ها شماره مجموعه را مشخص می‌کند. پس از تعیین شماره مجموعه، هر یک از بلاک‌های آن مجموعه می‌تواند حاوی بلاک مورد نظر ما باشد یا بلاک مورد نظر ما در صورت فقدان می‌تواند با صلاحدید LRU در یکی از آنها جانشین شود. تعداد مجموعه‌ها برابر است با تعداد بلاک‌های حافظه نهان تقسیم بر درجه آزادی که برابر با $\frac{۴}{۲} = ۲$ است. تمامی آدرسهای درخواستی دارای باقیمانده ۰ در تقسیم به ۲ هستند. لذا همه درخواست‌ها به مجموعه صفر از حافظه نهان نگاشت می‌شوند. پس در یکی از بلاک‌های صفر یا یک حافظه نهان قرار دارند یا درصورت فقدان در آنها جانشین می‌شوند. در این سوال برای درخواست‌ها داریم:

۱. بلاک صفر از حافظه اصلی درخواست می‌شود حافظه نهان خالی است پس فقدان اتفاق می‌افتد. اکنون باید بلاک درخواستی از حافظه اصلی به حافظه نهان منتقل شود. چون مجموعه صفر خالی است، هر جایی از آن می توان این بلاک را قرار داد. ما خانه صفر را انتخاب می‌کنیم (معمولاً در کامپیوتر شروع از صفر است)

۲. بلاک ۴ از حافظه اصلی درخواست می‌شود اما در حافظه نهان وجود ندارد پس فقدان داریم. پس از انتقال بلاک درخواستی از حافظه اصلی باید آنرا در بلاک مناسب از مجموعه صفر از حافظه نهان قرار دهیم. بلاک صفر از حافظه نهان پر است اما بلاک ۱ خالی است. پس حتماً در بلاک ۱ جانشین می‌شود.

۳. بلاک ۸ از حافظه نهان درخواست می‌شود. باید در مجموعه صفر این بلاک را بیابیم. اما بلاک ۸ در این مجموعه قرار ندارد. پس فقدان. هر دو بلاک‌های ۰ و یک مربوط به مجموعه صفر بوده و هر دو پر هستند. طبق LRU باید بلاک صفر جانشین شود چون دیرتر مورد استفاده قرار گرفته است.

۴. اکنون بلاک صفر حافظه اصلی درخواست می‌شود بازهم در مجموعه صفر وجود ندارد. پس فقدان داریم. اما این دفعه بلاک صفر حافظه اصلی در بلاک ۱ حافظه نهان جانشین می‌شود.

۵. بلاک ۴ از حافظه اصلی درخواست می‌شود که در مجموعه صفر وجود ندارد. لذا فقدان اتفاق می‌افتد و طبق LRU باید در بلاک صفر جانشین شود.

۶. بلاک ۸ از حافظه اصلی درخواست می‌شود که در مجموعه صفر وجود ندارد. پس فقدان داریم و طبق LRU بلاک مقصد ۱ است.

آدرس بلوک	شماره مجموعه	شماره بلوک	Hit/Miss
۰	۰	۰	Miss
۴	۰	۱	Miss
۸	۰	۰	Miss
۰	۰	۱	Miss
۴	۰	۰	Miss
۸	۰	۱	Miss

بنابر آنچه گفته شد پاسخ نهایی سوال به صورت زیر است:

آدرس بلوک مورد نظر	Direct Map		Fully associative		2-way set associative	
	Hit or Miss	شماره بلوک	Hit or Miss	شماره بلوک	Hit or Miss	شماره بلوک
۰	Miss	۰	Miss	۰	Miss	۰
۴	Miss	۰	Miss	۱	Miss	۱
۸	Miss	۰	Miss	۲	Miss	۰
۰	Miss	۰	Hit	۰	Miss	۱
۴	Miss	۰	Hit	۱	Miss	۰
۸	Miss	۰	Hit	۲	Miss	۱

پاسخ سوال ۷)

مقصود از CPI موثر همان CPI کل است. کل CPI لازم برابر است با CPI پایه بعلاوه کلاک‌هایی که به خاطر فقدان در خط لوله تعلیق اتفاق می‌افتد. چون دو سطح حافظه نهان داریم پس باید برای هر یک جداگانه حساب کنیم. پس:

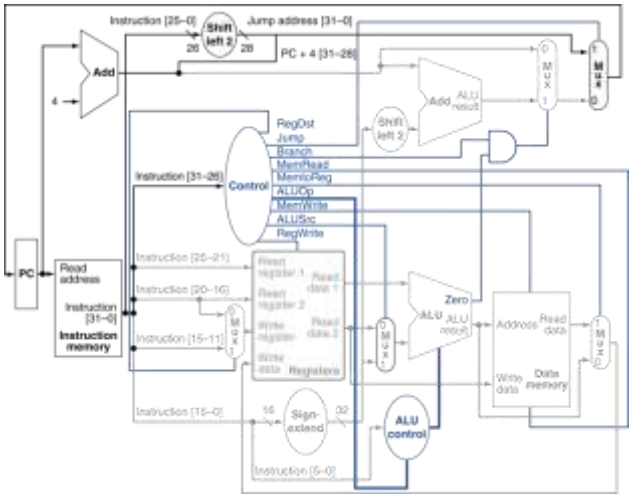
$$\begin{aligned} Effective\ CPI &= Total\ CPI \\ &= Base\ CPI + Primary\ Stalls\ per\ instruction \\ &\quad + Secondary\ Stalls\ per\ instruction \end{aligned}$$

$$Effective\ CPI = ۱ + ۰/۰۵ \times ۱۰ + ۰/۰۲ \times ۴۰۰ = ۹/۵$$

موفق باشید

گروه حل تمرین

پردازنده تک سیکلی MIPS



کلیات

هدف از فاز اول پروژه پایانی درس معماری کامپیوتر طراحی، پیاده‌سازی و تست قسمتی از پردازنده ۳۲ بیتی MIPS به صورت تک سیکل بدون خط لوله با استفاده از زبان Verilog می‌باشد.

توضیحات بیشتر:

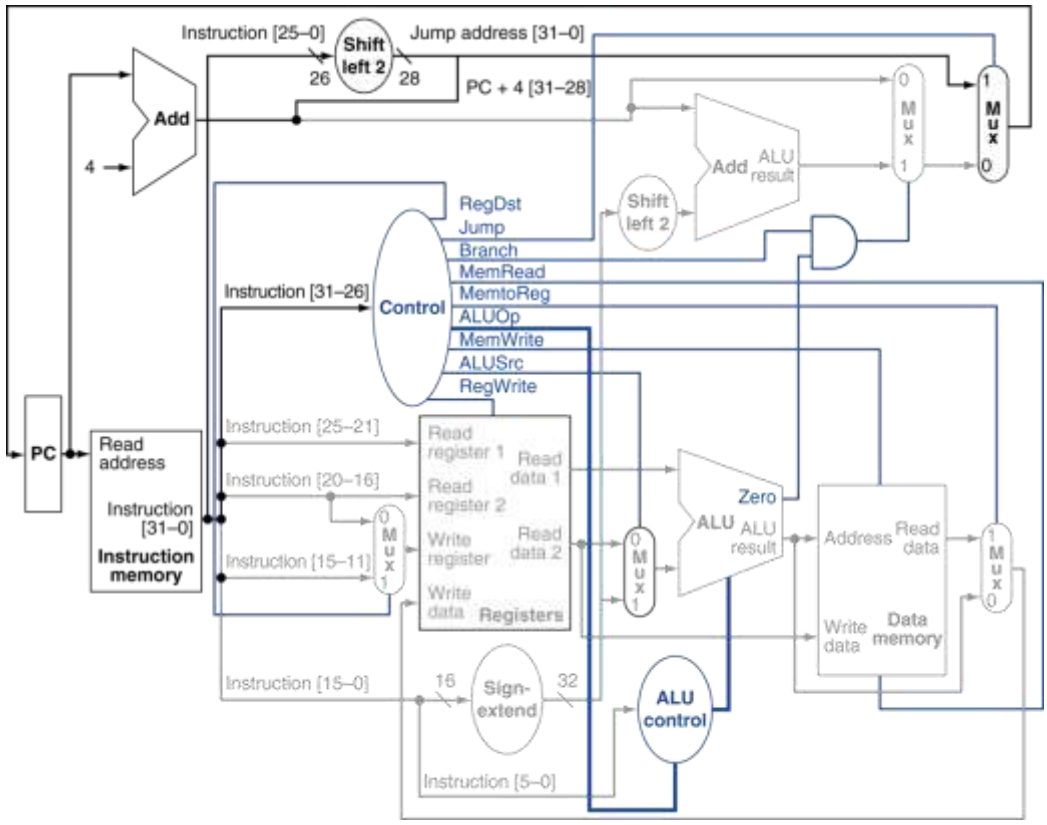
در درس معماری با پردازنده ۳۲ بیتی MIPS و زبان اسمبلی آن آشنا شده‌اید. در فاز اول پروژه قصد داریم برخی دستورهای این پردازنده را طراحی، پیاده‌سازی و تست کنیم.

۱- طراحی:

قبل از هرگونه کدنویسی ابتدا باید Datapath و واحد کنترل این پردازنده را بر روی کاغذ طراحی کنیم.

آ) Datapath

برای طراحی DataPath از طرح زیر استفاده می‌کنیم (شکل ۲۴-۴ صفحه ۳۲۹ کتاب مرجع اصلی):



شکل ۱

Datapath بالا برای دستورهای lw، sw، add، sub، AND، OR، slt، beq و j طراحی شده است. شما برای این پروژه باید با ایجاد تغییراتی در طرح فوق Datapath را به گونه‌ای طراحی کنید که قابلیت اجرای دستورهای گروههای ۱ و ۲ را داشته باشد.

ب) واحد کنترل

پس از طراحی Datapath کافی است با استفاده از واحد کنترل، مسیر حرکت داده را در Datapath مشخص کنیم. در این فاز چون پردازنده تک سیکل و بدون خط لوله می‌باشد، واحد کنترل آن بسیار ساده و یک مدار Combinational می‌باشد. یعنی مداری که تنها بر اساس ورودی خود تصمیم گرفته و خروجی را تولید می‌کند. در واقع واحد کنترل یک مدار است که بر اساس دستور Fetch شده، مقدار هر یک از سیگنالهای خروجی را تولید می‌کند.

۲- پیاده‌سازی

برای پیاده‌سازی طراحی خود از زبان Verilog بهره بگیرید. استفاده از هر سه شیوه Structural، Behaivral و Data Flow در این پروژه مجاز می‌باشد. در واقع ترکیبی از این سه روش پیاده‌سازی را آسان می‌کند.

برای پیاده‌سازی از برنامه‌نویسی Modular بهره بگیرید. ماژولهای لازم را تشخیص داده و سپس هر یک از آنها را به تنهایی طراحی، پیاده‌سازی و تست نمایید. سپس همه آنها را در پردازنده استفاده کنید. روش استفاده شده در ساخت سخت‌افزارهای جدید معمولاً Top-Down است. یعنی ابتدا سخت‌افزار را در نظر گرفته و آنرا به ماژولهایی تقسیم می‌کنیم (تنها اسم و عملکرد آنها را مشخص می‌کنیم و هیچ‌گونه طراحی، پیاده‌سازی و تستی انجام نمی‌شود) سپس هر یک از ماژولها را در نظر گرفته و آنرا به ماژولهایی تقسیم می‌کنیم (باز هم اسم و عملکرد آنها را مشخص می‌کنیم) این روند را تا زمانی ادامه می‌کنیم که به ماژولی برسیم که نتوان آنرا به ماژولهای دیگری تقسیم کرد. سپس این ماژولها را به صورت جداگانه طراحی، پیاده‌سازی و تست می‌کنیم. سپس آنها را با هم ترکیب کرده و ماژول‌های بزرگتر را می‌سازیم. در نهایت نیز سخت‌افزار اصلی که در این پروژه پردازنده می‌باشد را تولید می‌کنیم.

توجه داشته باشید که در شکل ۱، Instruction Memory و Data Memory جزئی از پردازنده نیستند و در بیرون آن قرار دارند. پس نیاز به تولید آنها نیست. کد این دو ماژول در اختیار شما قرار داده می‌شود.

همچنین دقت نمایید که برخی اجزا در شکل ۱، بهتر است به ماژول تبدیل نشوند و در کد پردازنده آنها را بنویسیم.

وجود ماژول ALU در کد شما اجباری است.

۳- تست

به طور کلی تست یکی از سخت‌ترین، پیچیده‌ترین و پرهزینه‌ترین بخش در ساخت یک محصول است. چه نرم‌افزار باشد چه سخت‌افزار.

برای تست روشهای مختلفی ارائه شده است. روشهایی هستند که درستی محصول را ۱۰۰ درصد تضمین می‌کنند اما در خیلی از موارد امکان اجرای آن روش به دلیل هزینه، زمان و ... نیست. این روشهای زمان زیادی را صرف تست می‌کنند. در واقع به ازای تمام ورودی‌ها، خروجی مدار را با آنچه انتظار می‌رود مقایسه می‌کنند.

راه دیگر استفاده از Test Vectorها می‌باشد. یعنی لیستی از ورودی‌هایی که کارکرد نادرست محصول برای آنها محتمل‌تر است. اگرچه این روش درستی را به طور ۱۰۰ درصد تضمین نمی‌کند اما سریعتر و به صرفه‌تر است.

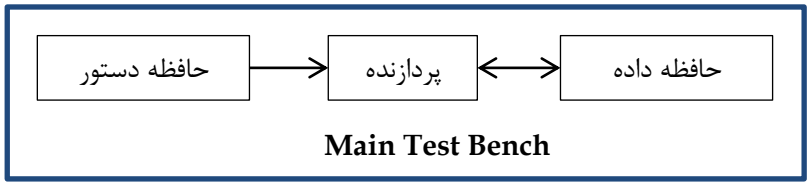
در طراحی سخت‌افزار نیز از روشهای تست استفاده می‌شود. در مراحل مختلف تولید یک سخت‌افزار بارها عمل تست انجام می‌شود. یکی از این تست‌ها، تست قبل از Fabrication می‌باشد. یعنی قبل از اینکه طرح به کارخانه ارسال شده و سخت‌افزار توسط کارخانه ساخته شود. ولی در این مرحله سخت‌افزار هنوز ساخته نشده است، پس چگونه می‌توان آنرا تست

کرد؟ تنها راه استفاده از شبیه‌سازی است. کد سخت‌افزار را توسط نرم‌افزارهای پیشرفته شبیه‌سازی می‌کنند و سپس تست‌های مورد نظر را بر روی آن انجام می‌دهند.

در این پروژه نیز شما باید هر ماژول را شبیه‌سازی و تست کنید. در نهایت نیز پردازنده را تست کنید. (مقصود تست قبل از Fabrication است). انتخاب روش تست بر عهده شما است و برای هر ماژول ممکن است متفاوت باشد.

برای تست هر یک از ماژولها می‌توانید از Test Bench استفاده کنید. Test Bench یک ماژول است که درون آن یک Instance از ماژولی که می‌خواهیم تست کنیم به همراه ماژولهای مرتبط با آن، گرفته و به سیگنالهای مورد نظر (مثلاً کلاک) مقدار مناسب می‌دهیم. در صورت استفاده از Test Bench می‌توانید از تمام قابلیت‌های Verilog برای مقداردهی به سیگنالها استفاده کنید.

استفاده از Test Bench برای ماژول پردازنده الزامی است. این Test Bench باید دارای طرح زیر باشد (جزئیات صرفنظر شده است):



دستورهای گروه ۱

Instruction	Description	Instruction	Description
add	Add	slt	Set On Less Than
addi	Add Immediate	beq	Branch Equal
lw	Load Word	j	Jump
sw	Save Word	jr	Jump Register
nor	NOR	jal	Jump and Link

دستورهای گروه ۲

Instruction	Description	Instruction	Description
lh	Load Half	srl	Shift Right Logical
sll	Shift Left Logical	sra	Shift Right Arithmetic
lui	Load Upper Immediate		

نکات:

۱- برنامه‌نویسی باید به زبان Verilog انجام گیرد.

۲- برنامه‌نویسی Modular الزامی است.

۳- هر Module باید در یک فایل نوشته شود.

۴- استفاده از Test Bench برای ماژول پردازنده اجباری است اما برای بقیه ماژولها اختیاری است. هرچند توصیه می‌شود برای همه ماژولها Test Bench نوشته شود.

۵- وجود ماژول ALU الزامی است.

۶- برای حافظه داده و حافظه دستور باید از ماژولهایی که در اختیار شما قرار می‌گیرد استفاده کنید.

۷- در صورتی که همه دستورها (جدول صفحه ۳۳ و ۳۴ اسلایدهای اسمبلی MIPS) به جز syscall را پیاده‌سازی کنید، نمره ویژه‌ای دریافت می‌کنید.

۸- دستورهای گروه ۲ دارای پیچیدگی بیشتری نسبت به گروه ۱ در پیاده‌سازی می‌باشند. لذا دارای نمره بیشتری خواهند بود.

۹- همکاری بین گروها ممنوع می‌باشد. (یکسان بودن کد گروهها بررسی می‌شود)

۱۰- استفاده از کد آماده مجاز نمی‌باشد. (مگر آنکه توسط کمک تدریس در اختیار شما قرار گیرد)

۱۱- گروه‌ها باید ۲ نفره باشند. نه کمتر و نه بیشتر!

۱۲- نمره فاز اول پروژه شما ۳ نمره از نمره پایانی است.

۱۳- تحویل به دو صورت غیرحضوری و حضوری انجام می‌گیرد.

۱۴- در تحویل غیرحضوری باید تمامی طراحی‌های خود (برای هر ماژول به صورت جدا)، به همراه کد همه ماژولها و TestBench آنها و دیگر فایلهای ضروری (مثل فایلهای Do) را فشرده کرده و تحت قالب یک فایل به ایمیل

computer_arch_3902@yahoo.com

ارسال نمایید. عنوان ایمیل باید به صورت

[Project_Phase_1][Student ID 1][Student ID 2]

باشد که به جای Student ID 1 و Student ID 2 شماره دانشجویی اعضای گروه قرار می‌گیرد. برای ارسال طراحی‌های خود می‌توانید آنها را با نرم‌افزار مناسب رسم کرده و یا پس از رسم بر روی کاغذ آنها را اسکن و ارسال نمایید. طرح ارسال شده باید واضح بوده و نام تمامی سیگنالها مشخص شود.

۱۵- تحویل حضوری شامل پرسیدن سوال و مشاهده نتایج شبیه‌سازی و بررسی درستی پردازنده می‌باشد. هر دو نفر باید به تمام بخش‌های پروژه مسلط باشند وگرنه نمره آنها کسر می‌شود.

۱۶- به همراه داشتن مستندات طراحی در روز تحویل حضوری الزامی است.

۱۷- **حضور تمامی افراد گروه در روز تحویل پروژه الزامی است.**

۱۸- آخرین مهلت تحویل غیرحضوری روز سه‌شنبه ۹ خرداد ساعت ۲۰:۵۹:۵۹ می‌باشد. تحویل حضوری نیز روز بعد، یعنی چهارشنبه ۱۰ خرداد انجام می‌گیرد. زمان‌بندی تحویل هر گروه متعاقباً اعلام خواهد شد.

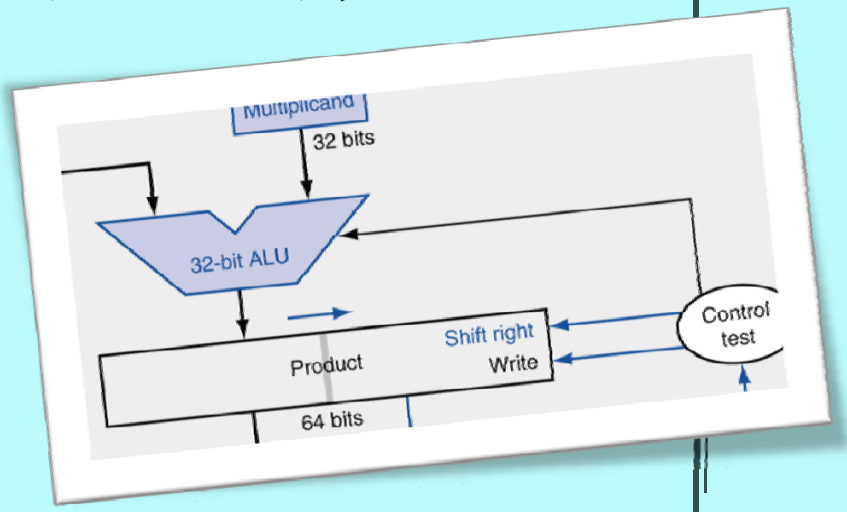
پیروز و سربلند باشید

گروه حل تمرین

معماری کامپیوتر
مثال تقسیم

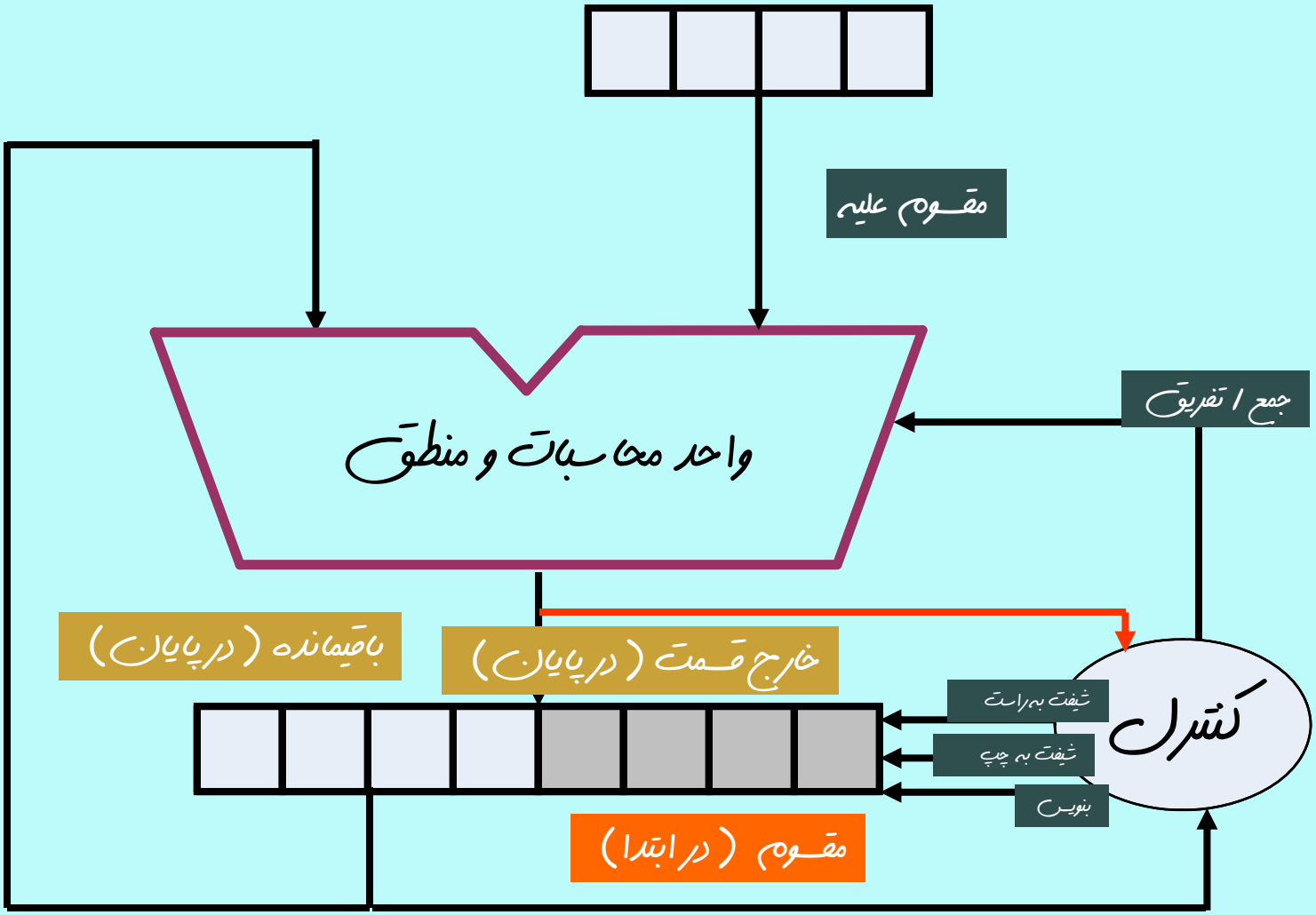
دانشگاه شهید بهشتی
دانشکده‌ی مهندسی برق و کامپیوتر
زمستان ۱۳۹۰
احمد محمودی ازناوه

سخت‌افزار بهینه‌سازی شده

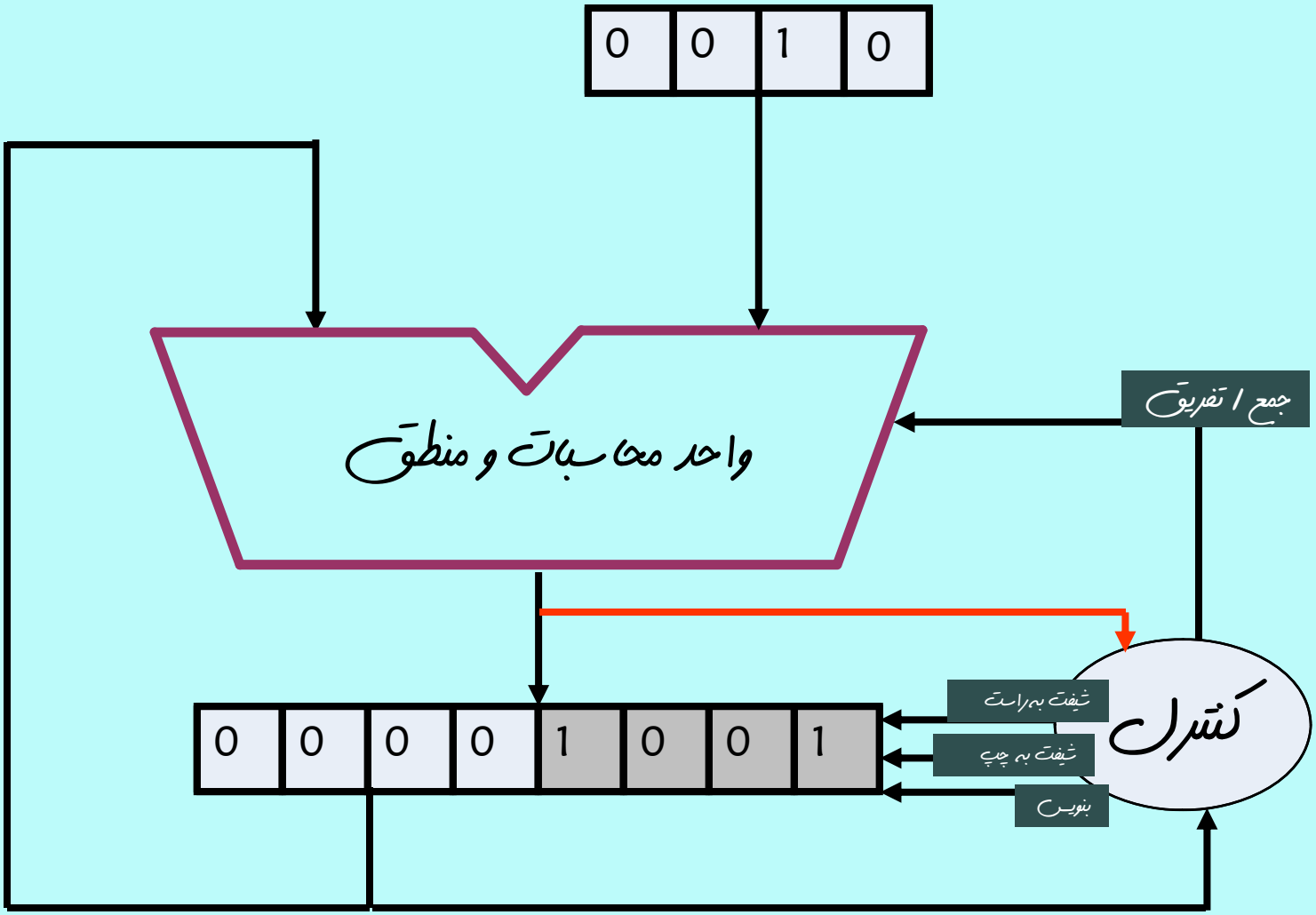


شبیه به مدار ضرب نیست؟!

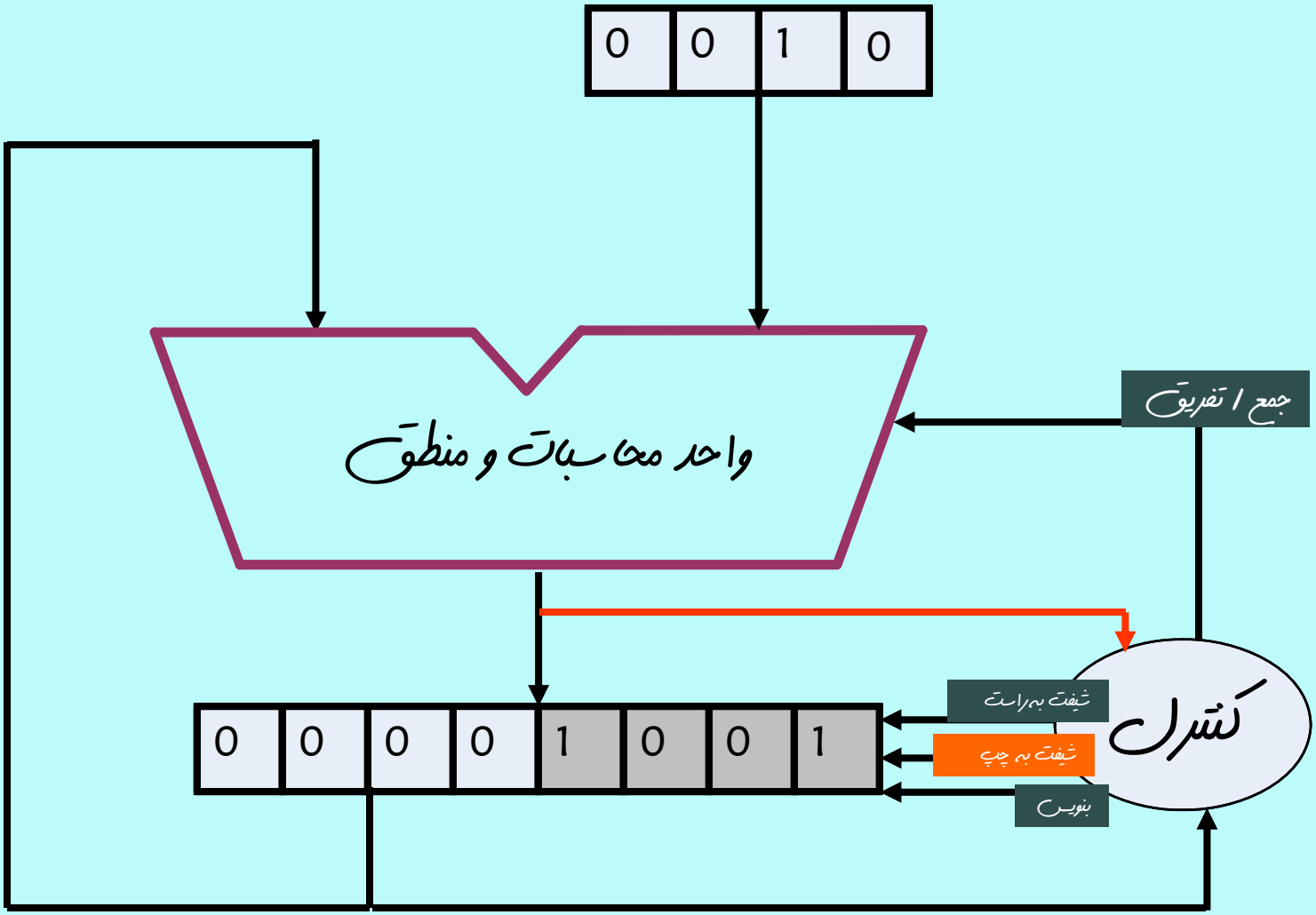
سخت‌افزار تقسیم



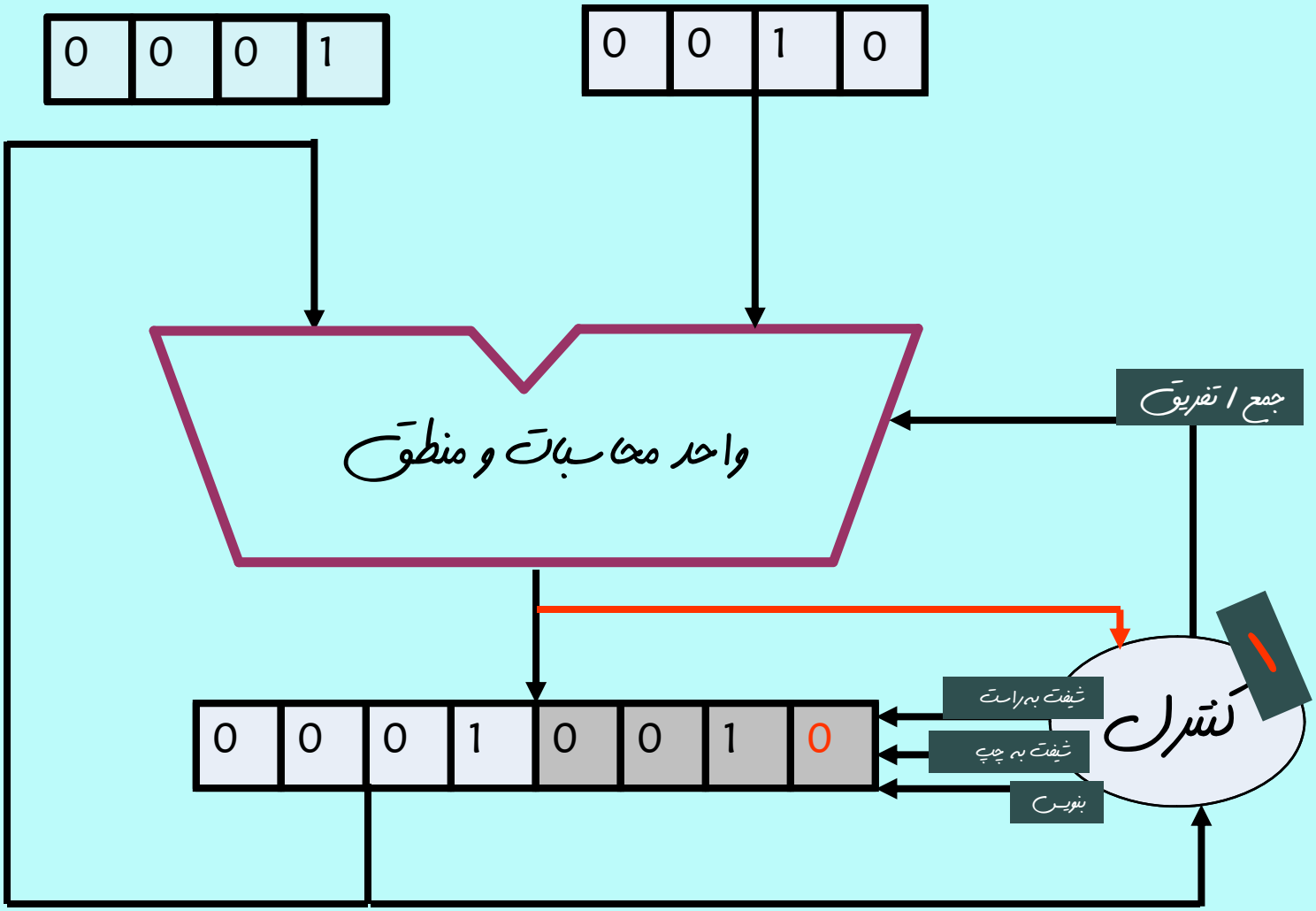
مثال تقسیم


0 0 1 0
واحد محاسبات و منطق
جمع / تفریق
0 0 0 0 1 0 0 1
کنترل
شیفت به راست
شیفت به چپ
بنویس

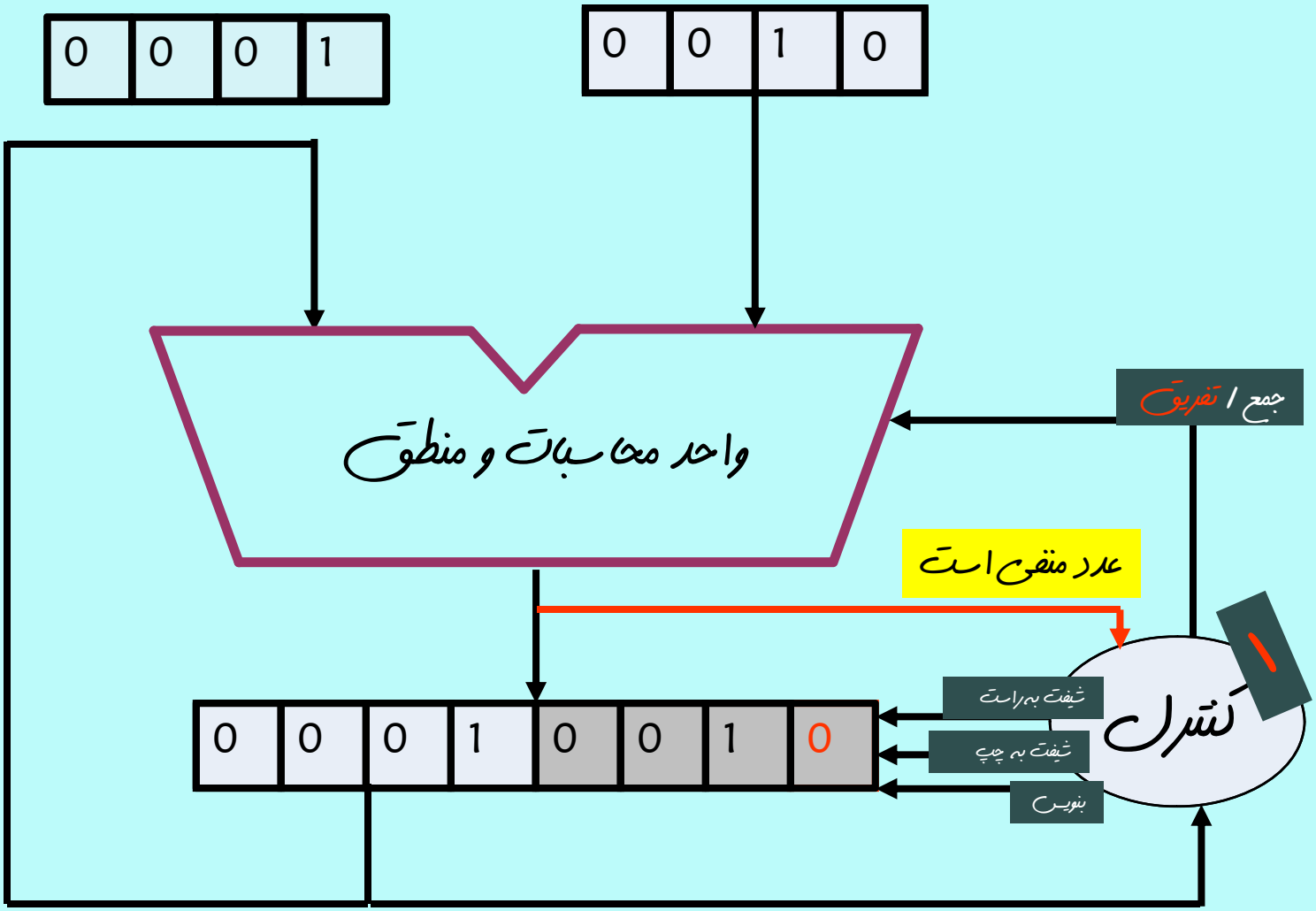
مثال تقسیم (ادامه...)



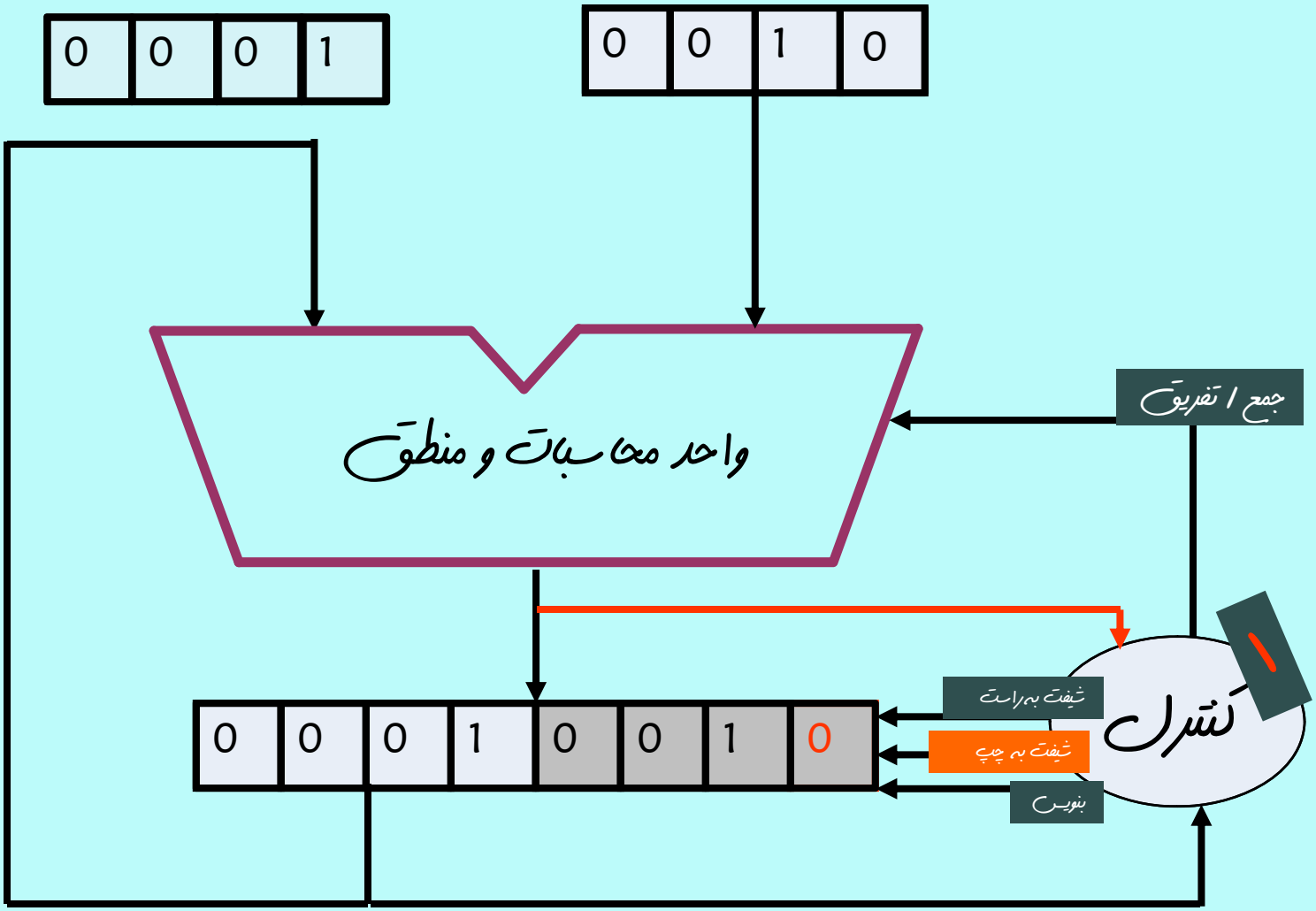
مثال تقسیم (ادامه...)



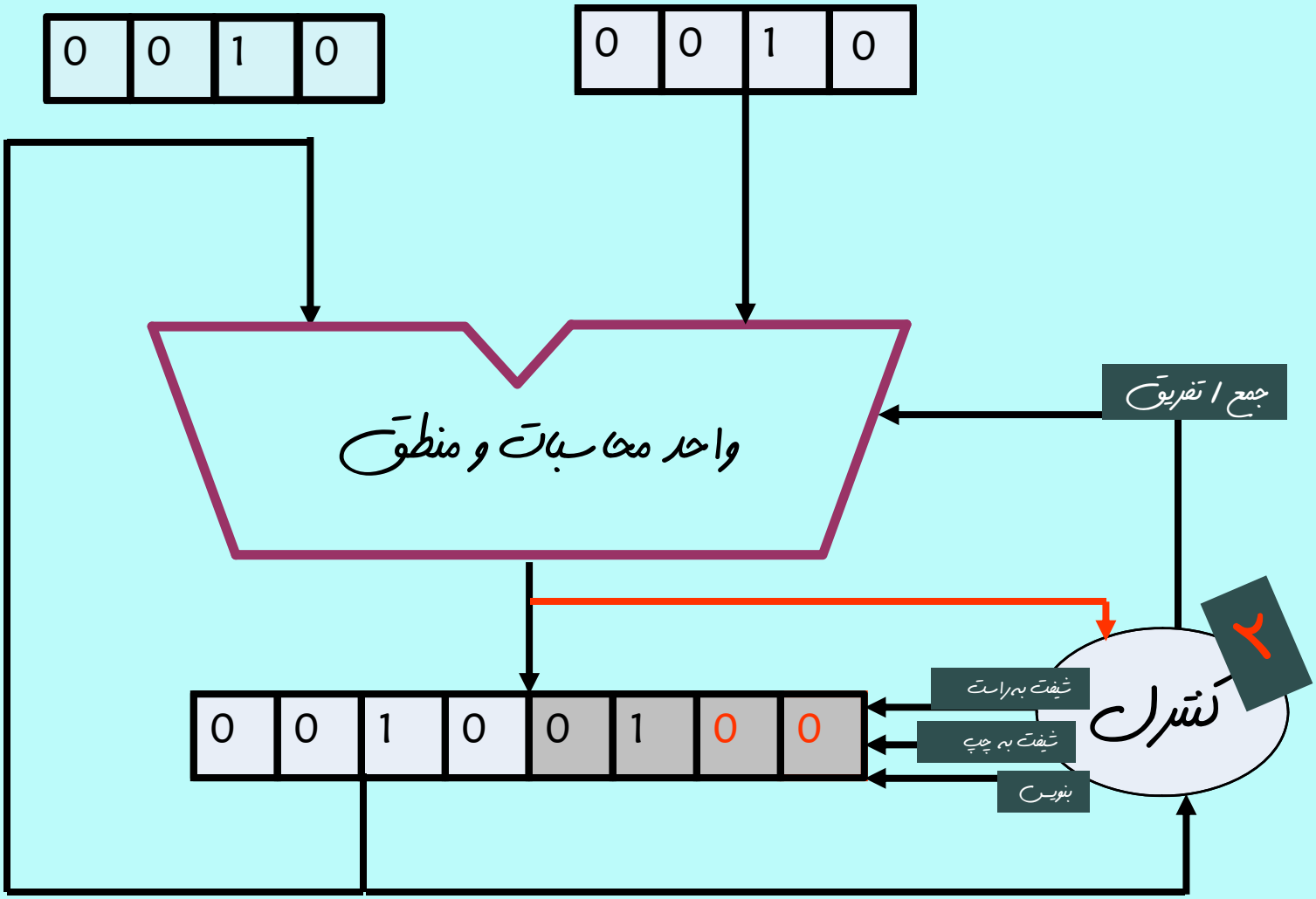
مثال تقسیم (ادامه...)



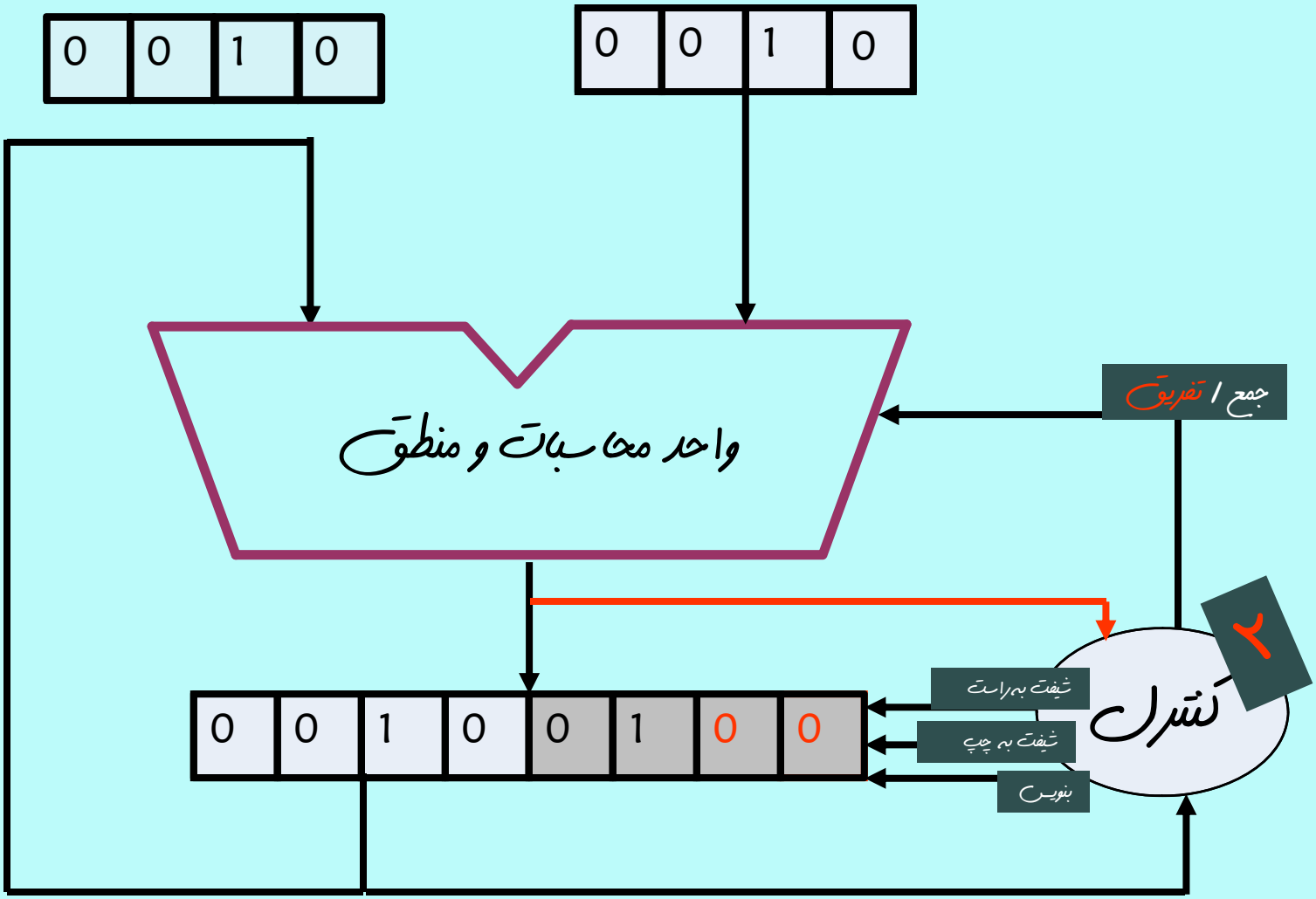
مثال تقسیم (ادامه...)



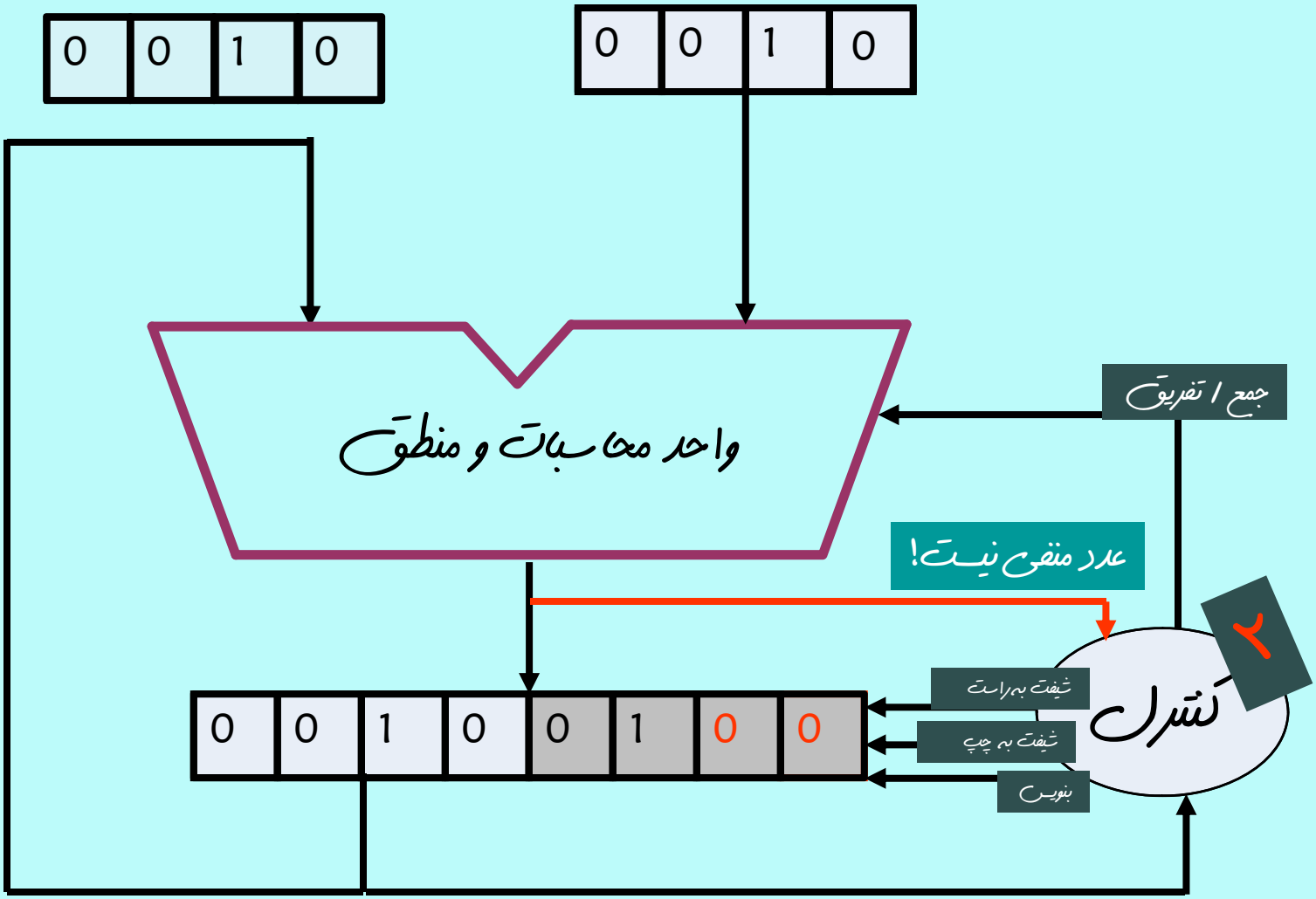
مثال تقسیم (ادامه...)



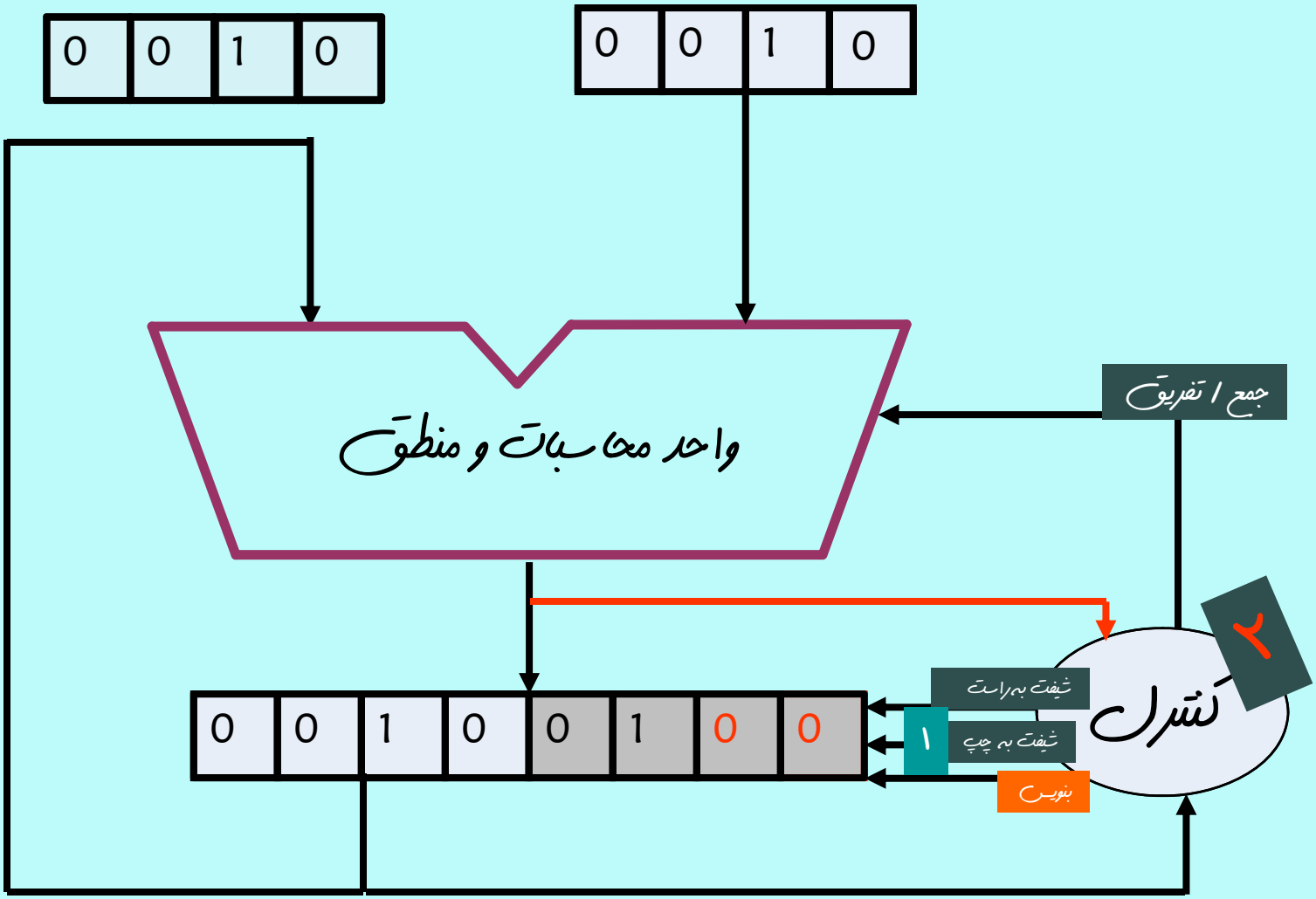
مثال تقسیم (ادامه...)



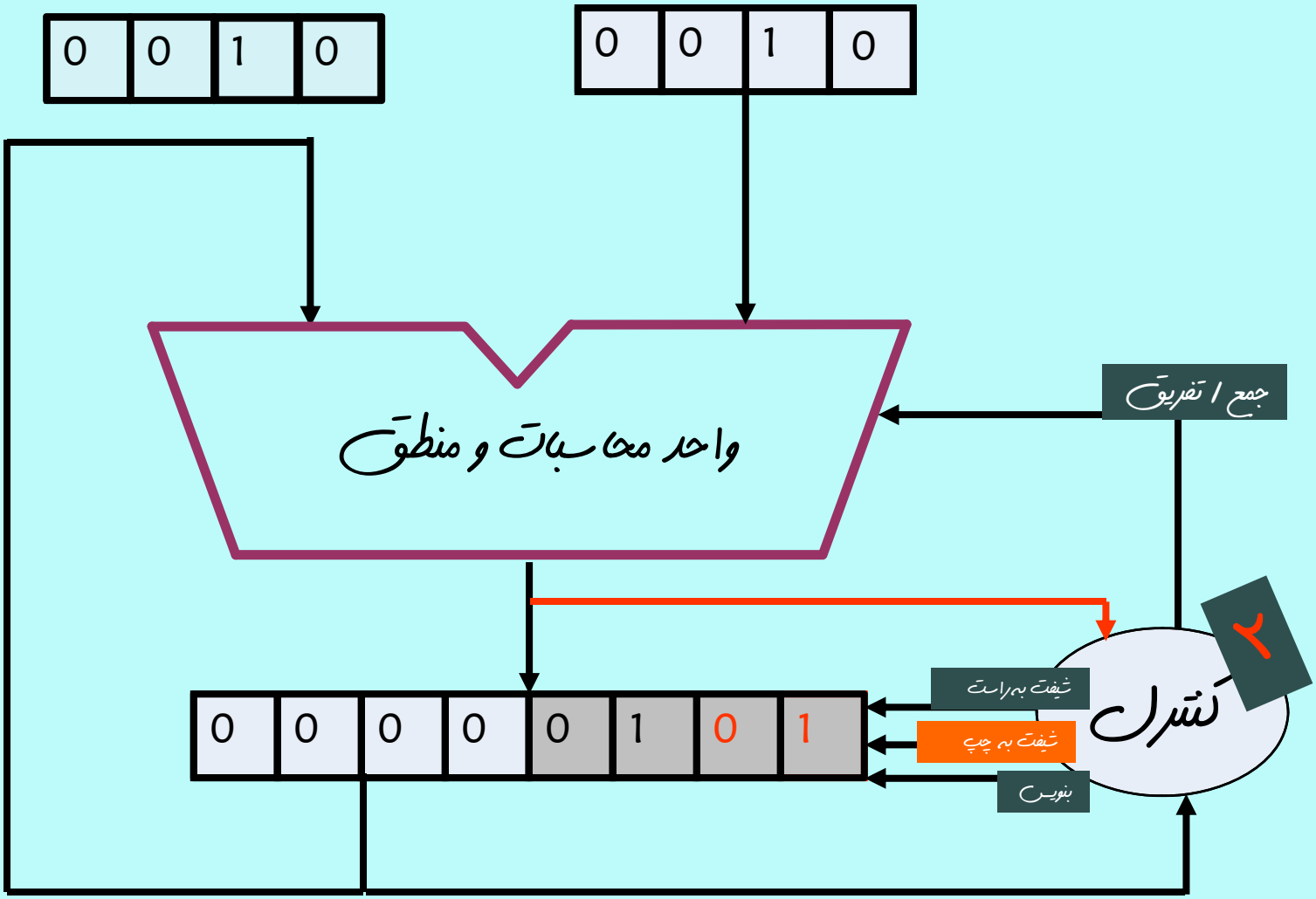
مثال تقسیم (ادامه...)



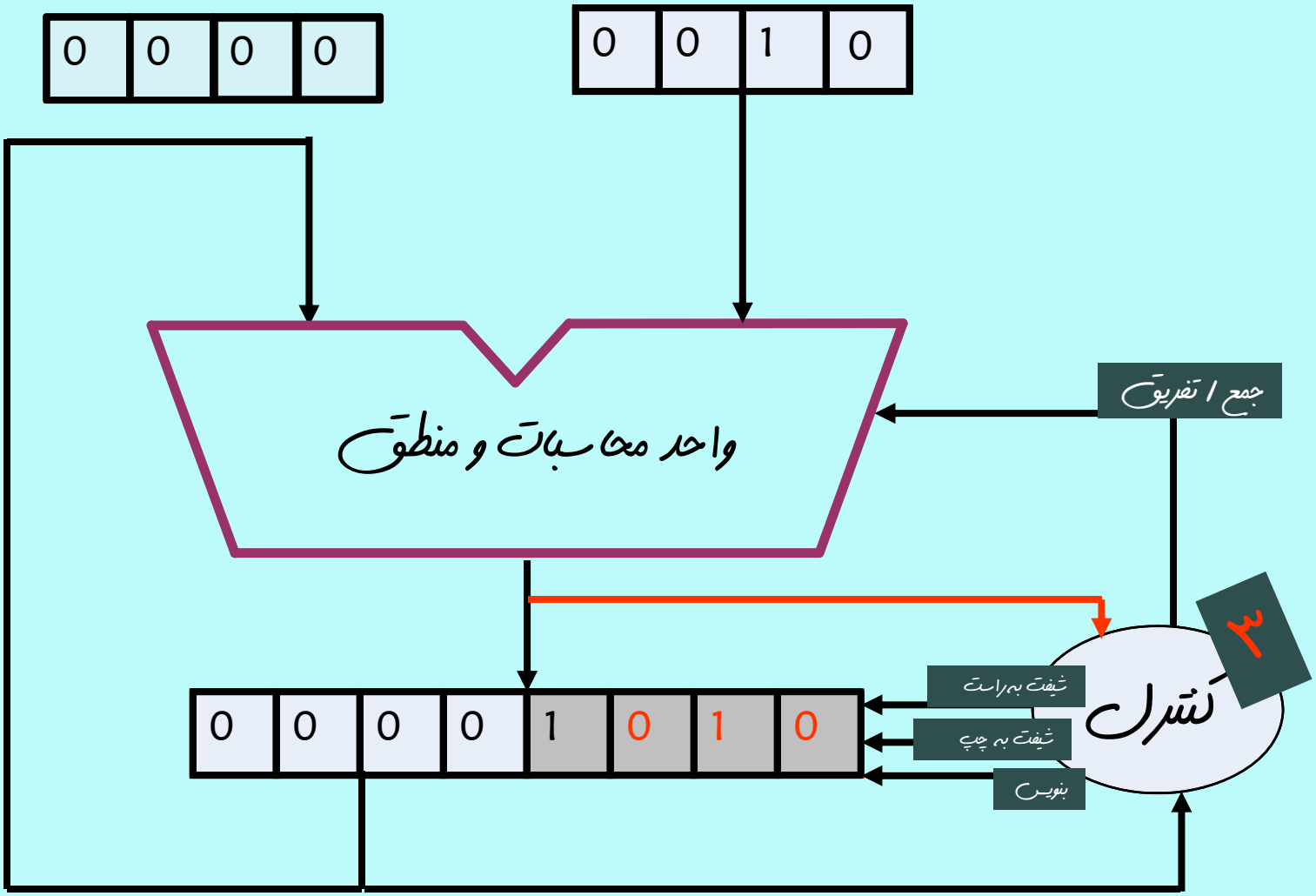
مثال تقسیم (ادامه...)



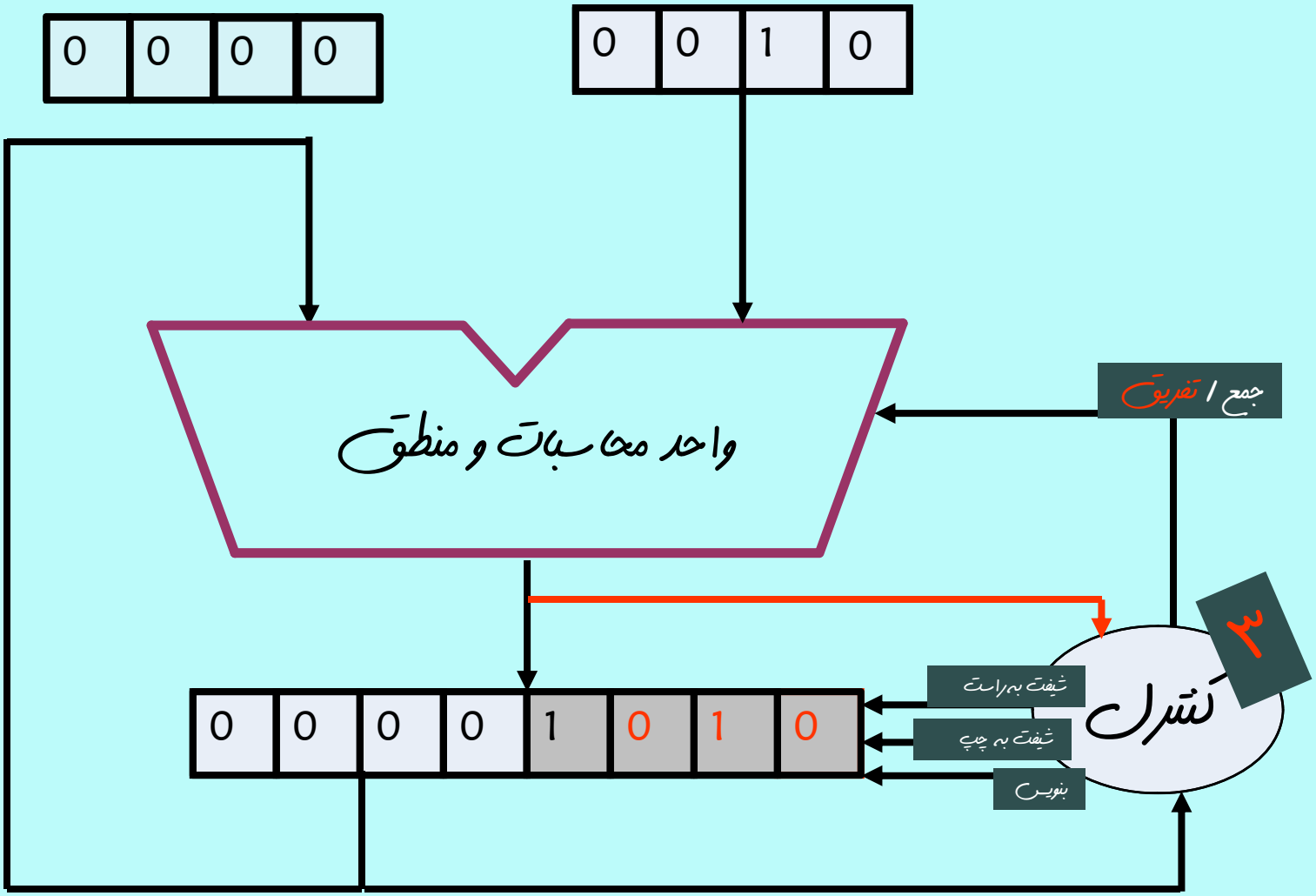
مثال تقسیم (ادامه...)



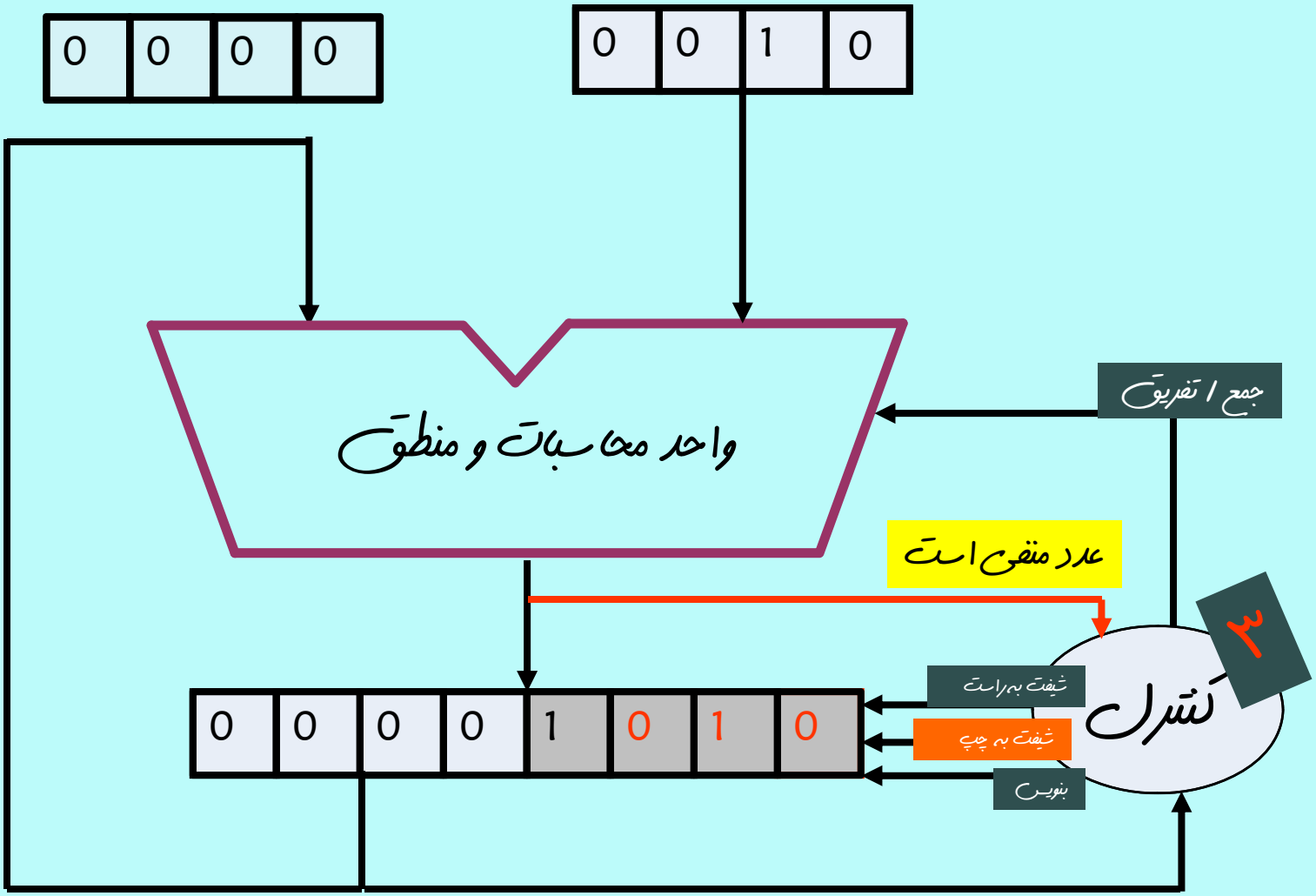
مثال تقسیم (ادامه...)



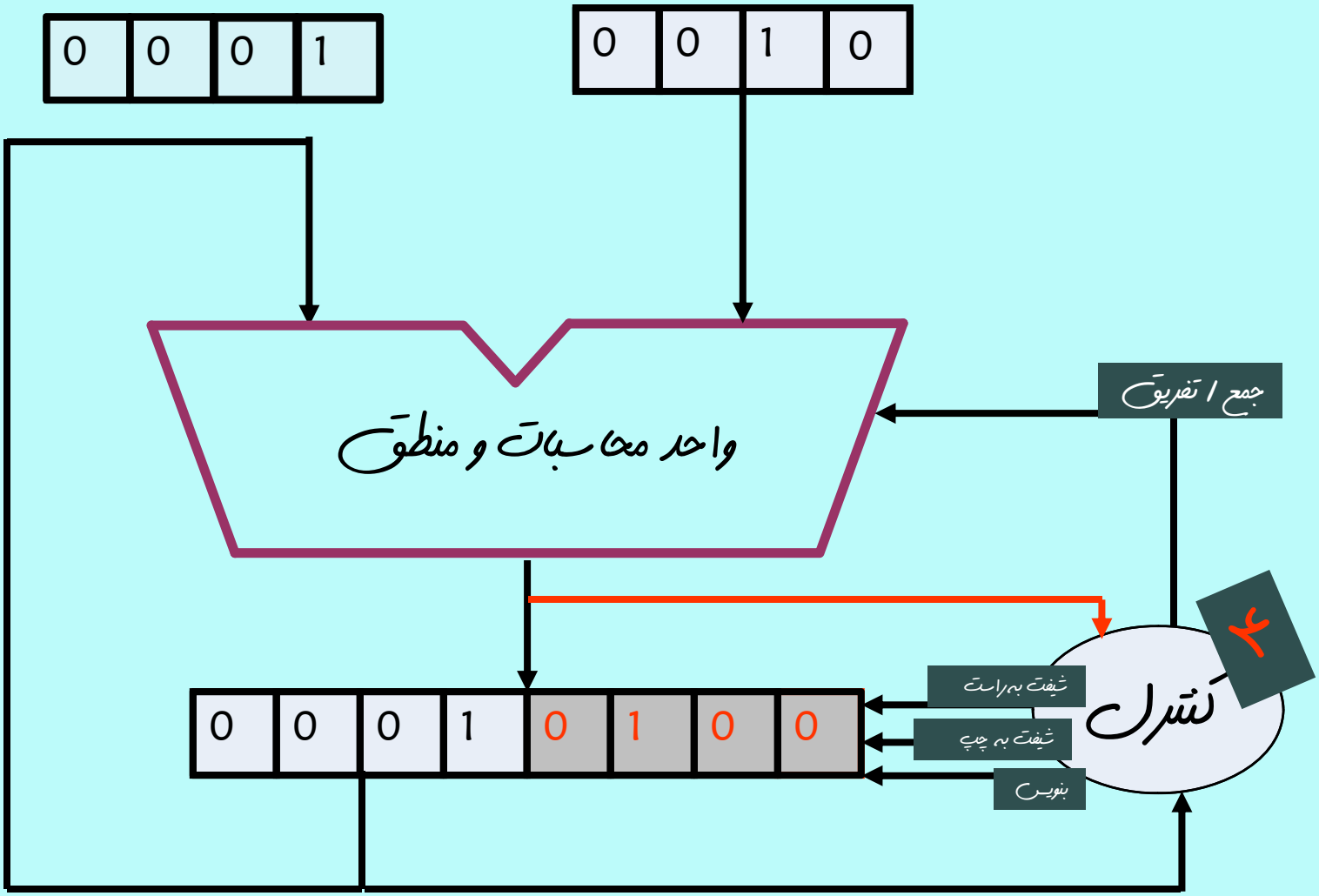
مثال تقسیم (ادامه...)



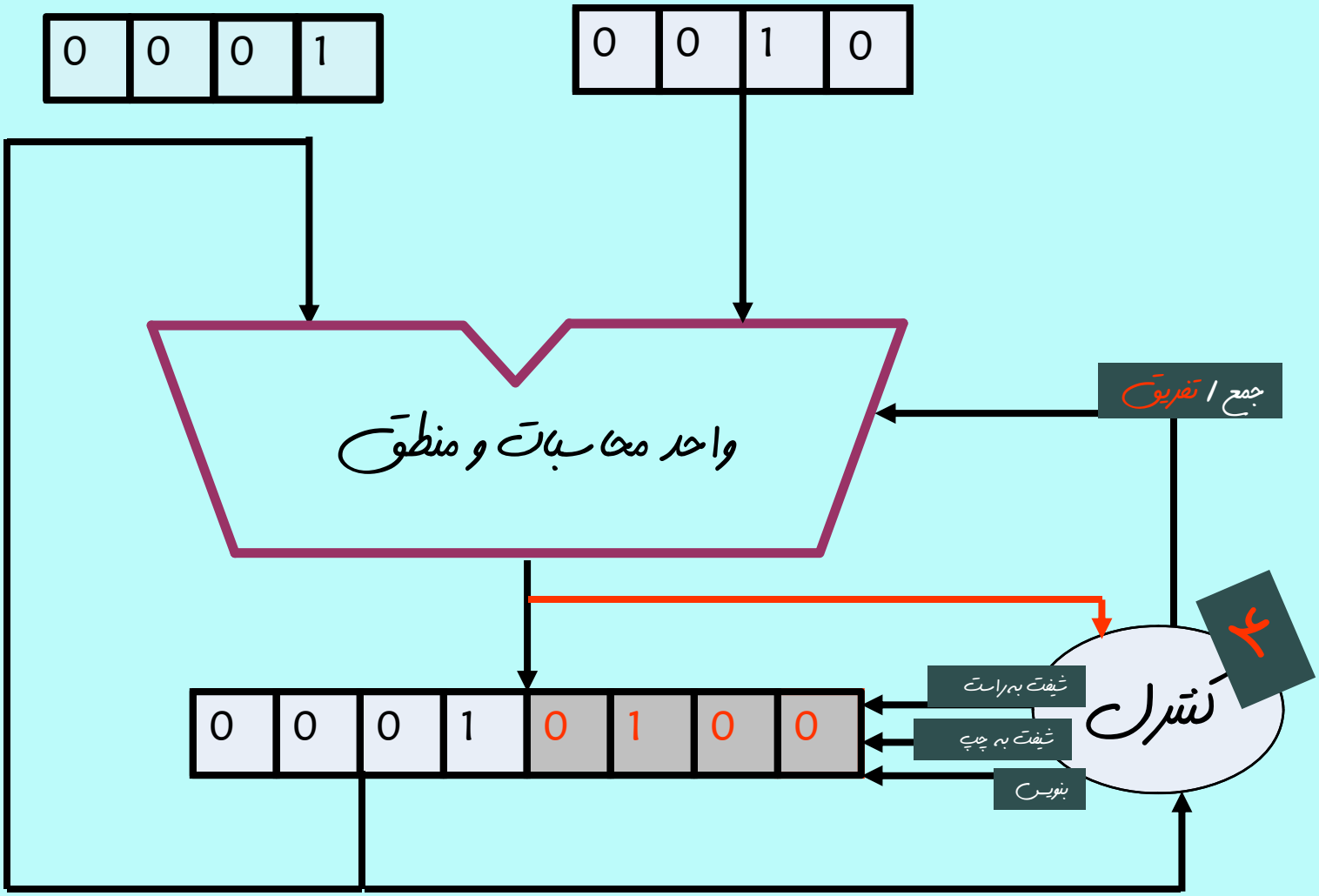
مثال تقسیم (ادامه...)



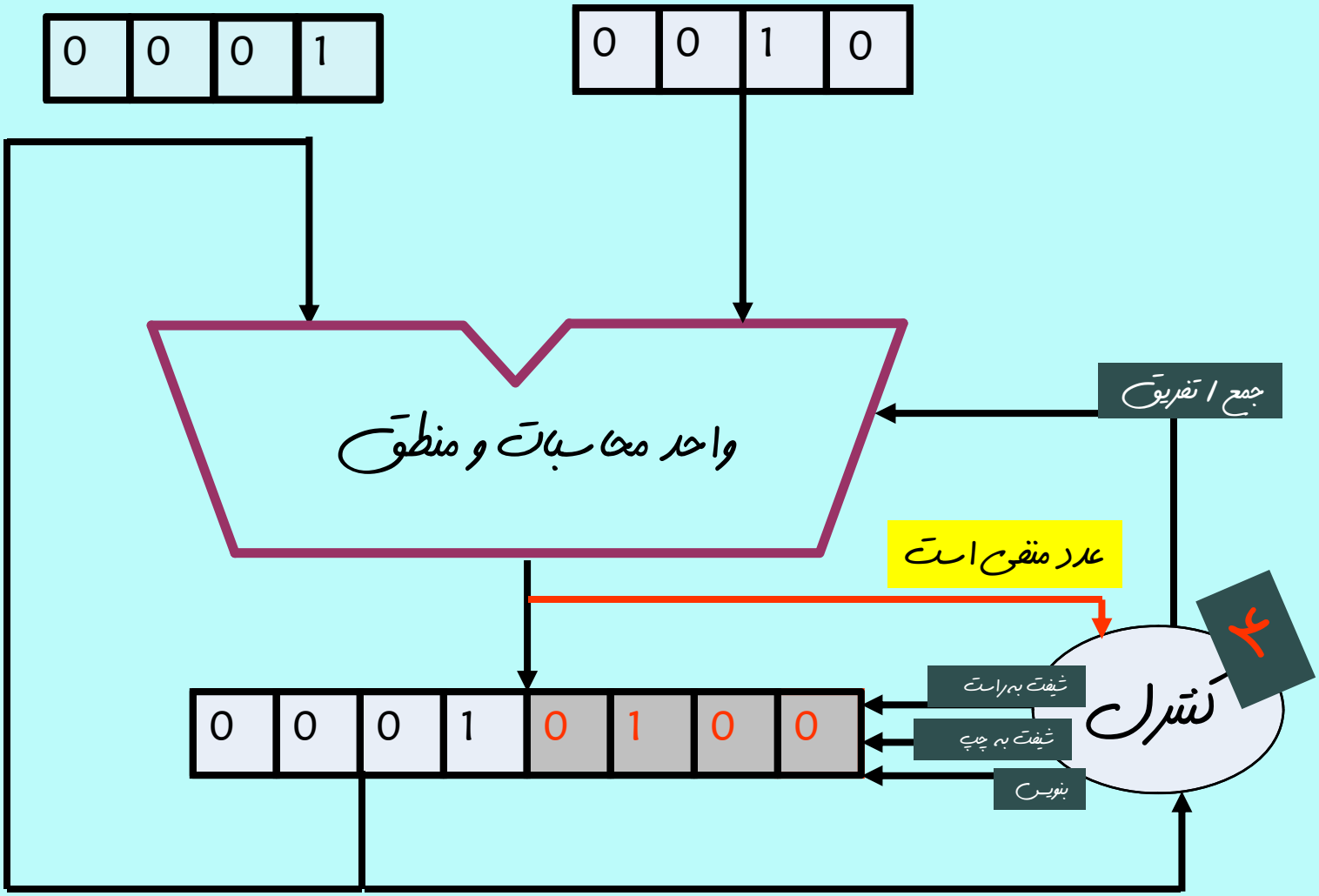
مثال تقسیم (ادامه...)



مثال تقسیم (ادامه...)



مثال تقسیم (ادامه...)



مثال تقسیم (ادامه...)