معماری کامپیوتر

عملیات نقل و انتقال ثبات‌ها

فصل چهارم کتاب موریس مانو

محمدعلی شفیعیان

http://shafieian-education.ir

سرفصل مطالب

- زبان انتقال ثبات (Register Transfer Language)
- انتقال ثبات (Register Transfer)
- انتقال حافظه و گذرگاه (Bus and Memory Transfers)
- ریزعمل های حسابی (Arithmetic Microoperations)
- ریزعملهای منطقی (Logic Microoperations)
- ریزعمل های شیفت (Shift Microoperations)
- واحد شیفت حسابی (Arithmetic Logic Shift Unit)

سیستم دیجیتالی ساده

- مدارات ترتیبی و ترکیبی می تواند برای ساختن سیستم های دیجیتالی ساده استفاده شود
- سیستمهای دیجیتالی ساده معمولاً با یکی از موارد زیر شناخته می شوند:
  - ثبات هایی که در سیستم موجود است.
  - عملیاتی که سیستم انجام می دهد.
- برای شناسایی یک سیستم باید بدانیم:
  - چه عملیاتی روی داده ها انجام می شود.
  - چه اطلاعاتی بین ثباتها منتقل می شود.

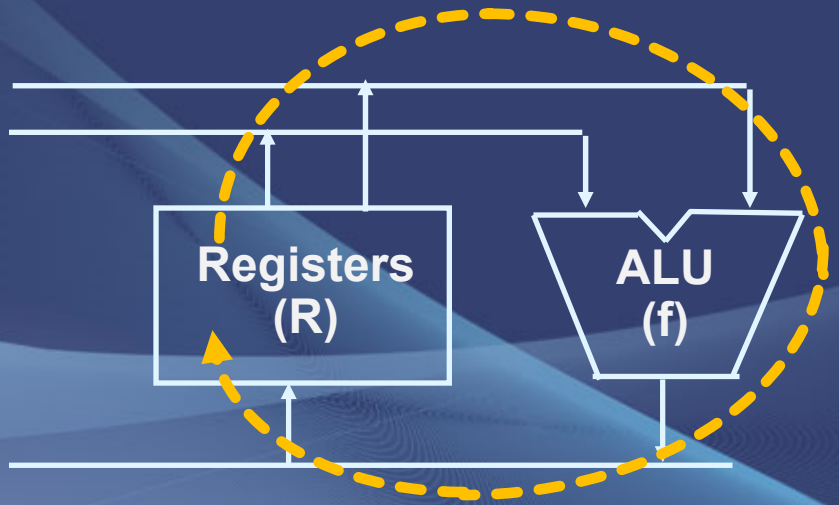
ریزعمل ها (۱)

- عملیاتی که روی داده های ذخیره شده در ثباتها انجام می شود ریز عمل نامیده می شود که نتیجه عمل یا در ثبات قبلی ذخیره می شود یا در یک ثبات دیگر.

- عملیات داخلی ثباتها نمونه هایی از ریزعمل ها هستند.
  - شیفت Shift
  - بار کردن Load
  - پاک کردن Clear
  - اضافه کردن Increment
  - ...

ریزعمل ها (۲)

عملیات پایه که روی داده های ذخیره شده در یک یا چند ثبات در طی یک پالس ساعت انجام می شود ریزعمل نام دارد.



یک سیکل ساعت

$R \leftarrow f(R, R)$

f می تواند shift، load، clear، increment، add، subtract، complement، and، xor یا ... باشد.

سازمان کامپیوتر

- سازمان سخت افزاری (organization) داخلی کامپیوتر با سه خصوصیت زیر تعریف می شود:

- مجموعه ثبات ها و وظیفه هر یک

- مجموعه ریزعمل هایی که روی ثبات ها انجام می شود

- سیگنال های کنترلی که ترتیب ریزعمل ها را مشخص می کنند (واحد کنترلی که موجب اجرای رشته ریزعمل ها می شود).

سطح انتقال ثبات

بررسی کامپیوتر از این دیدگاه سطح انتقال ثبات

« Register Transfer Level »

نامیده می شود.

- **در این سطح تمرکز بر موارد زیر است:**
  - **ثبات های سیستم**
  - **تبدیل داده ها درون ثبات ها**
  - **انتقال داده ها بین ثبات ها**

زبان انتقال ثبات
REGISTER TRANSFER LANGUAGE

- به جای مشخص کردن یک سیستم با کلمات، آن را با یک نمادی خاص که زبان انتقال ثبات نامیده می شود نشان می دهند.

- زبان انتقال ثبات می تواند برای نشان دادن هر ترتیب از ریزعمل ها (رشته ریزعمل ها) مورد استفاده قرار گیرد.

- زبان انتقال ثبات:
  - یک زبان سمبولیک است.
  - یک ابزار آسان برای شرح سازمان داخلی کامپیوترهای دیجیتال است.
  - فرآیند طراحی سیستم های دیجیتال را تسهیل می کند.

نامگذاری ثباتها

- ثبات ها معمولاً با حروف بزرگ انگلیسی نامگذاری می شوند. گاهی اوقات پس از اسم آنها اعدادی قرار می گیرد (A, R13, IR).
- اغلب نام ها نشان دهنده کاری است که ثبات انجام می شود مثلاً:
  - MAR - Memory Address Register
  - PC - Program Counter
  - IR - Instruction Register

انتقال ثبات

- کپی شدن اطلاعات یک ثبات به ثبات دیگر انتقال ثبات نام دارد.
- یک انتقال ثبات به شکل زیر نشان داده می شود:

$R2 \leftarrow R1$

  - در این حالت محتوای ثبات R1 به R2 منتقل می شود.
  - انتقال در یک پالس انجام می شود.
  - محتوای R1 تغییر نمی کند.

انتقال ثبات

- یک انتقال ثبات مثل زیر:

$R3 \leftarrow R5$

موارد زیر را در سیستم ایجاب می کند:

– خطوط انتقال از R5 به R3
– بار شدن موازی در R3
– خطوط کنترل لازم برای انجام عملیات

توابع کنترلی

- اغلب اوقات، عملیات فقط زمانی که یک شرط خاص برقرار باشد باید اجرا شوند.
- این مسأله شبیه if در زبان های برنامه نویسی است.
- در سیستم های دیجیتال شرط با یک سیگنال کنترلی (*control signal*) یا تابع کنترلی (*control function*) انجام می شود

- تابع کنترلی به شکل زیر نشان داده می شود:

P: R2 ← R1

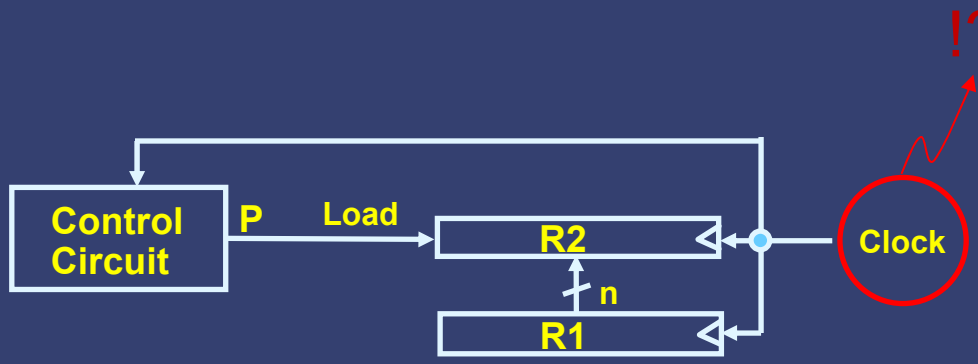
بدین معنی که اگر P برابر ۱ بود انتقال از R1 به R2 انجام شود، یا:

if (P = 1) then (R2 ← R1)

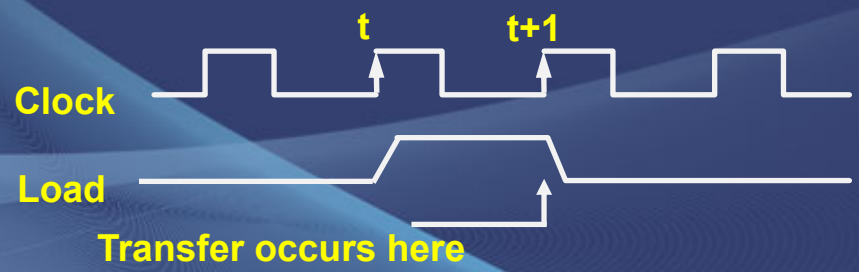
پیاده سازی سخت افزاری انتقالات کنترلی

P: R2 ← R1

بلوک دیاگرام



دیاگرام زمان بندی



فرض می شود که ثبات ها حساس به لبه مثبت هستند

عملیات همزمان

- اگر تعداد دو یا بیشتر عملیات همزمان انجام شود آنها را با کاما (،) از هم جدا می کنیم.

$$P:\ R3 \leftarrow R5,\ MAR \leftarrow IR$$

- در اینجا اگر P=1 باشد، به طور همزمان R5 به R3 و IR به MAR منتقل می شود.

چندین شرط

- اگر چندین شرط را داشته باشیم، می توانیم آنها را and و or کنیم.

$X_1\ X_2 :\ R3 \leftarrow R2$

- در اینجا اگر X1 و X2 هر دو ۱ باشند R2 به R3 منتقل می شود، به عبارت دیگر:

$if\ X_1\ , X_2 = 1\ \ then\ \ \ R3 \leftarrow\ \ R2$

چندین شرط

$X_1 + X_2 : R3 \leftarrow R2$

- در اینجا اگر X1 یا X2 برابر ۱ باشند R2 به R3 منتقل می شود، به عبارت دیگر:

$if\ X_1\ or\ X_2 = 1\ \ then\ \ \ R3 \leftarrow\ \ R2$

چند مثال

$$X_1 + X_2 : R3 \leftarrow R1 + R2$$

این عبارت یعنی:

$$if\ X_1\ or\ X_2 = 1\ \ then\ \ \ R3 \leftarrow\ \ R1 + R2$$

$$X_1 + X_2 : R3 \leftarrow R1 \vee R2$$

این عبارت یعنی:

$$if\ X_1\ or\ X_2 = 1\ \ then\ \ \ R3 \leftarrow\ \ R1\ or\ R2$$

$$X_1 + X_2 : R3 \leftarrow R1 \wedge R2$$

این عبارت یعنی:

$$if\ X_1\ or\ X_2 = 1\ \ then\ \ \ R3 \leftarrow\ \ R1\ and\ R2$$

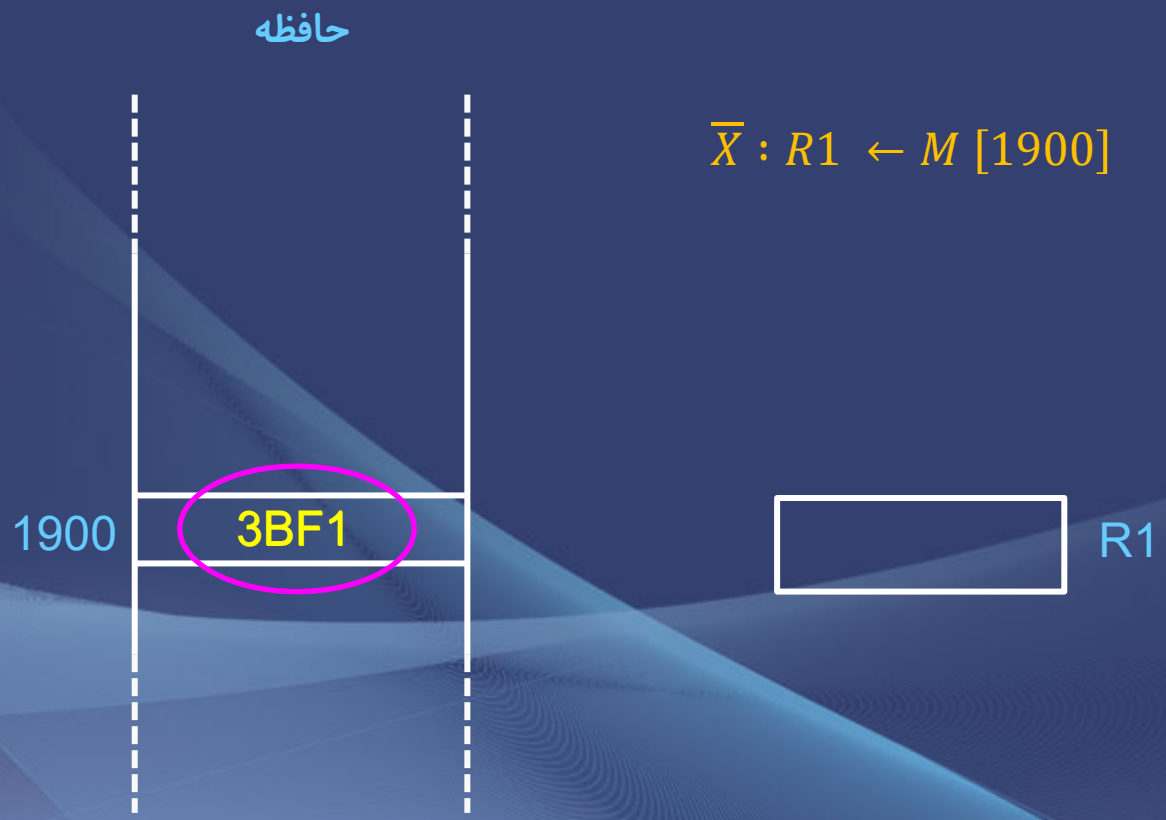
چند مثال

$\overline{X} : R1 \leftarrow M\,[AR]$

در اینجا، به شرطی که سیگنال کنترلی X برابر با صفر باشد، محتوای سطری از حافظه که ثبات آدرس (AR) به آن اشاره می کند در ثبات R1 قرار می گیرد.

$\overline{X} : R1 \leftarrow M\,[1900]$

در اینجا، به شرطی که سیگنال کنترلی X برابر با صفر باشد، محتوای سطر ۱۹۰۰ از حافظه در ثبات R1 قرار می گیرد.



$$\overline{X} : R1 \leftarrow M\,[1900]$$

علائم اولیه

سمبل	شرح	مثال
حروف بزرگ	نشان دهنده یک ثبات	MAR , R2
پرانتز ()	نشان دهنده قسمتی از یک ثبات	R2(0-7) , R2(L)
پیکان ←	نشان دهنده انتقال اطلاعات	R2 ← R1
دو نقطه :	نشان دهنده پایان تابع کنترلی	P :
کاما ،	جداکننده دو ریزعمل	A←B , B←A

ارتباط بین ثبات ها

- در یک سیستم دیجیتال با ثبات های فراوان که انتقال اطلاعات بین آنها الزامی است، اتصال مستقیم هر ثبات با ثبات دیگر به دلیل ایجاد پیچیدگی زیاد در سیم بندی امکان پذیر نیست.

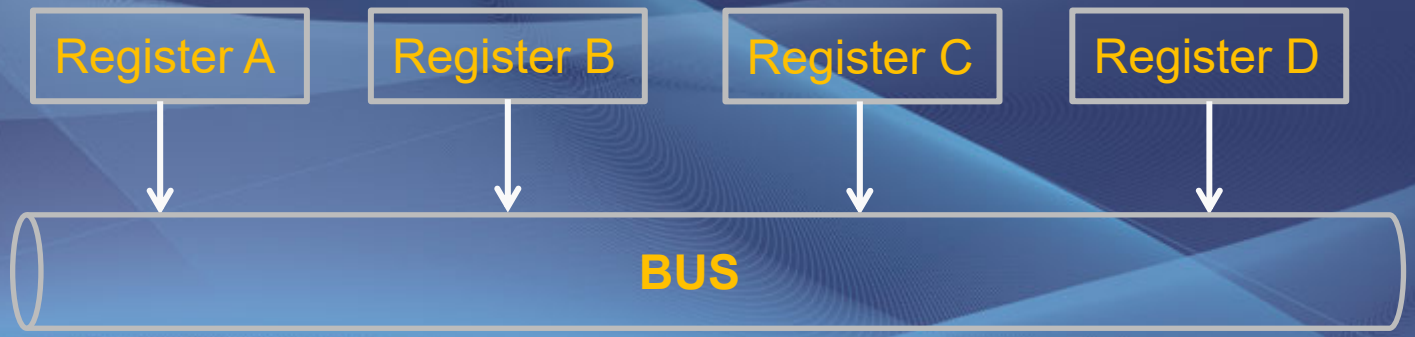
- برای اتصال n ثبات به یکدیگر به $n(n-1)$خط ارتباطی نیاز است.
- هزینه: $O(n^2)$
  - برای سیستم های با تعداد ثبات زیاد، عملی نیست

- به جای این کار از یک مجموعه مدار متمرکز (مسیر ارتباطی مشترک) به نام گذرگاه مشترک (bus) برای انتقال اطلاعات استفاده می شود.
- همچنین از توابع کنترلی تعیین کنیم کدام ثبات، ثبات منبع و کدام ثبات مقصد است.

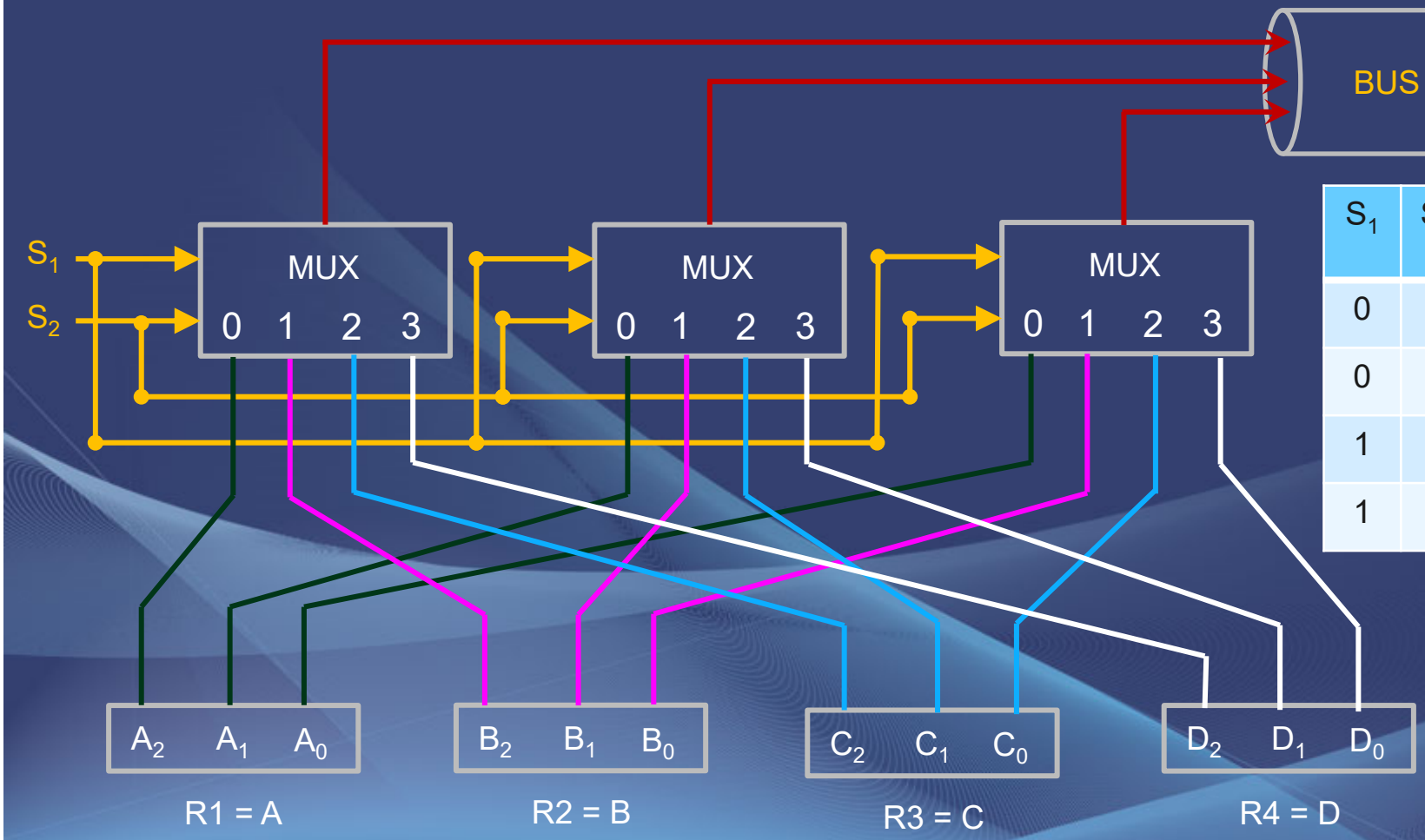
گذرگاه مشترک BUS

- در کامپیوتر، تعداد زیادی ثبات وجود دارد که معمولاً انتقال اطلاعات بین آنها الزامی است و چون ارتباط دو به دوی آنها پیچیدگی زیادی در سیم بندی را به همراه دارد بهتر است از یک مسیر ارتباطی مشترک به نام **BUS** یا گذرگاه مشترک استفاده شود.
- گذرگاه یک مسیر(متشکل یک از گروه از سیم ها) است که اطلاعات روی آن منتقل می شود. انتقال می تواند از منابع مختلف به مقاصد مختلف باشد.

از یک ثبات به گذرگاه: $BUS \leftarrow R$

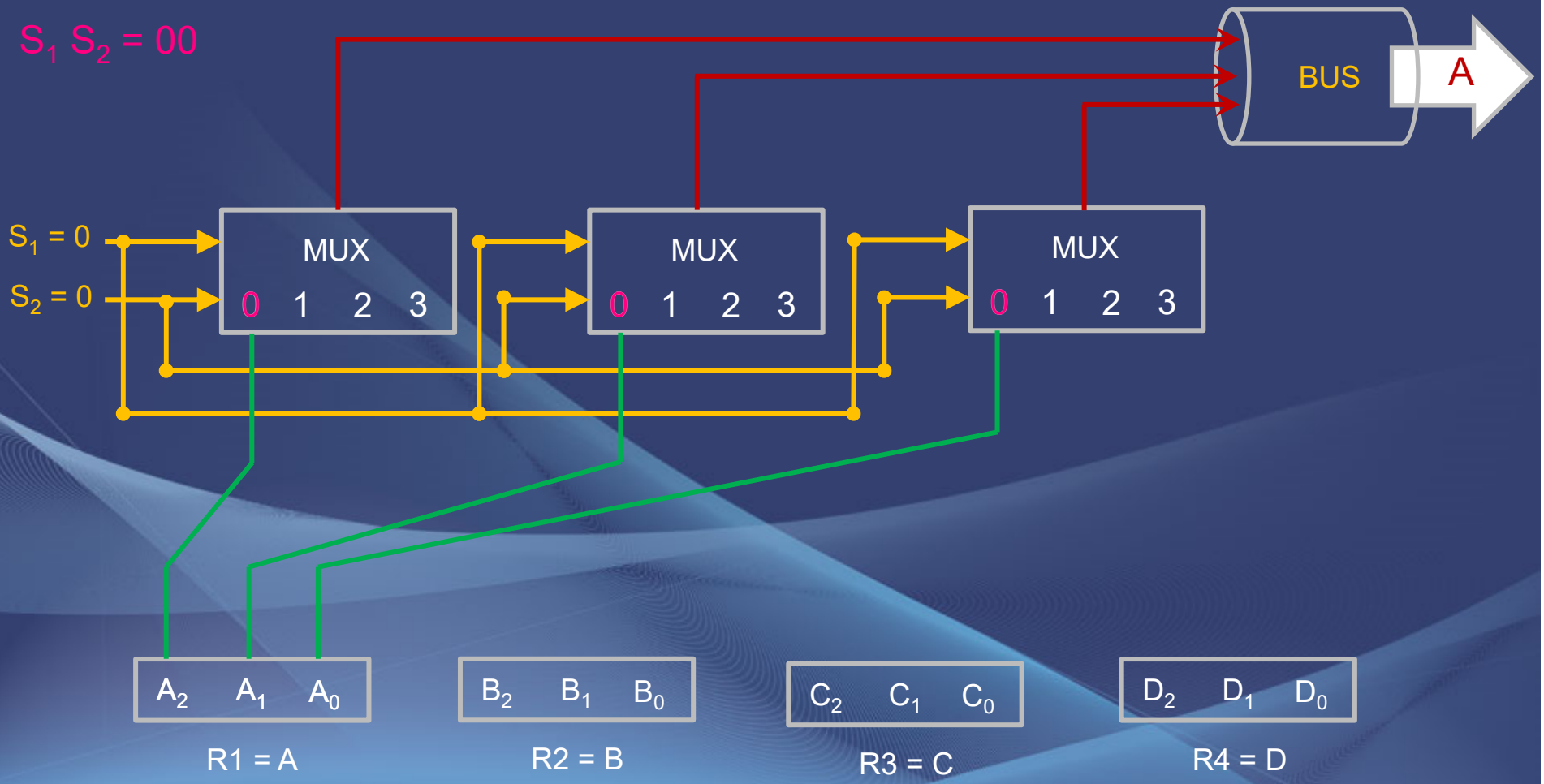


گذرگاه مشترک به کمک مالتی پلکسر (*MUX*)

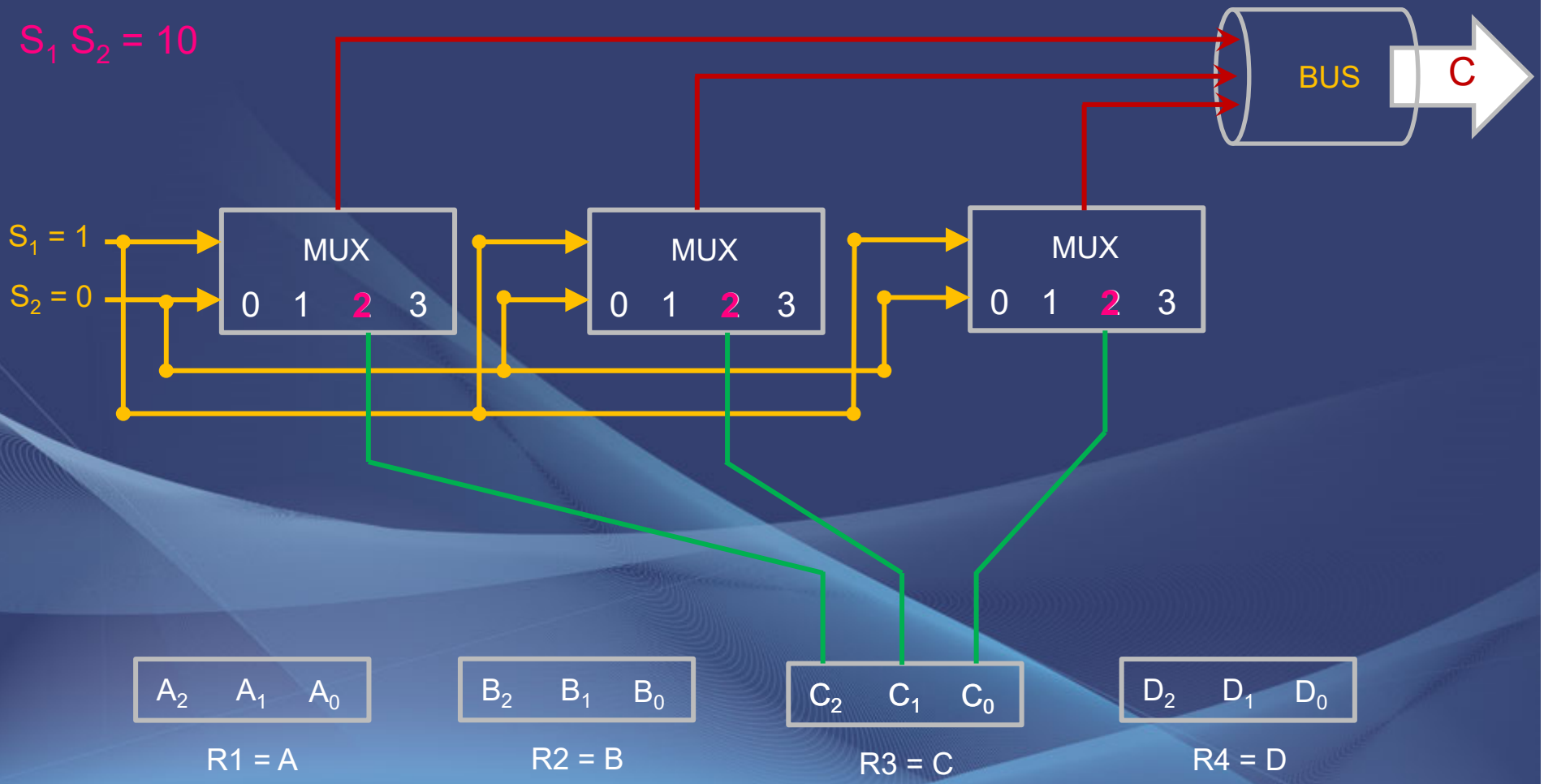


S_1	S_2	داده روی BUS
0	0	A
0	1	B
1	0	C
1	1	D

گذرگاه مشترک به کمک مالتی پلکسر (*MUX*)



گذرگاه مشترک به کمک مالتی پلکسر (*MUX*)



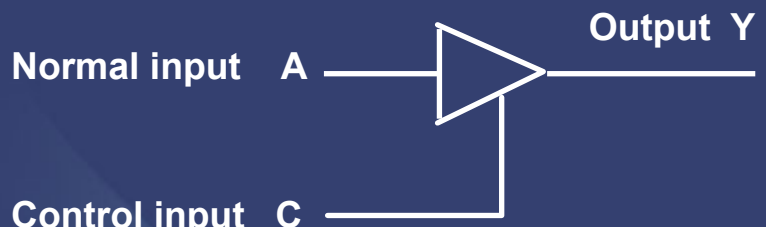
گذرگاه مشترک به کمک مالتی پلکسر (*MUX*)

- اگر k ثبات n بیتی داشته باشیم، برای ایجاد گذرگاه مشترک به n مالتی پلکسر k x 1 نیاز داریم.

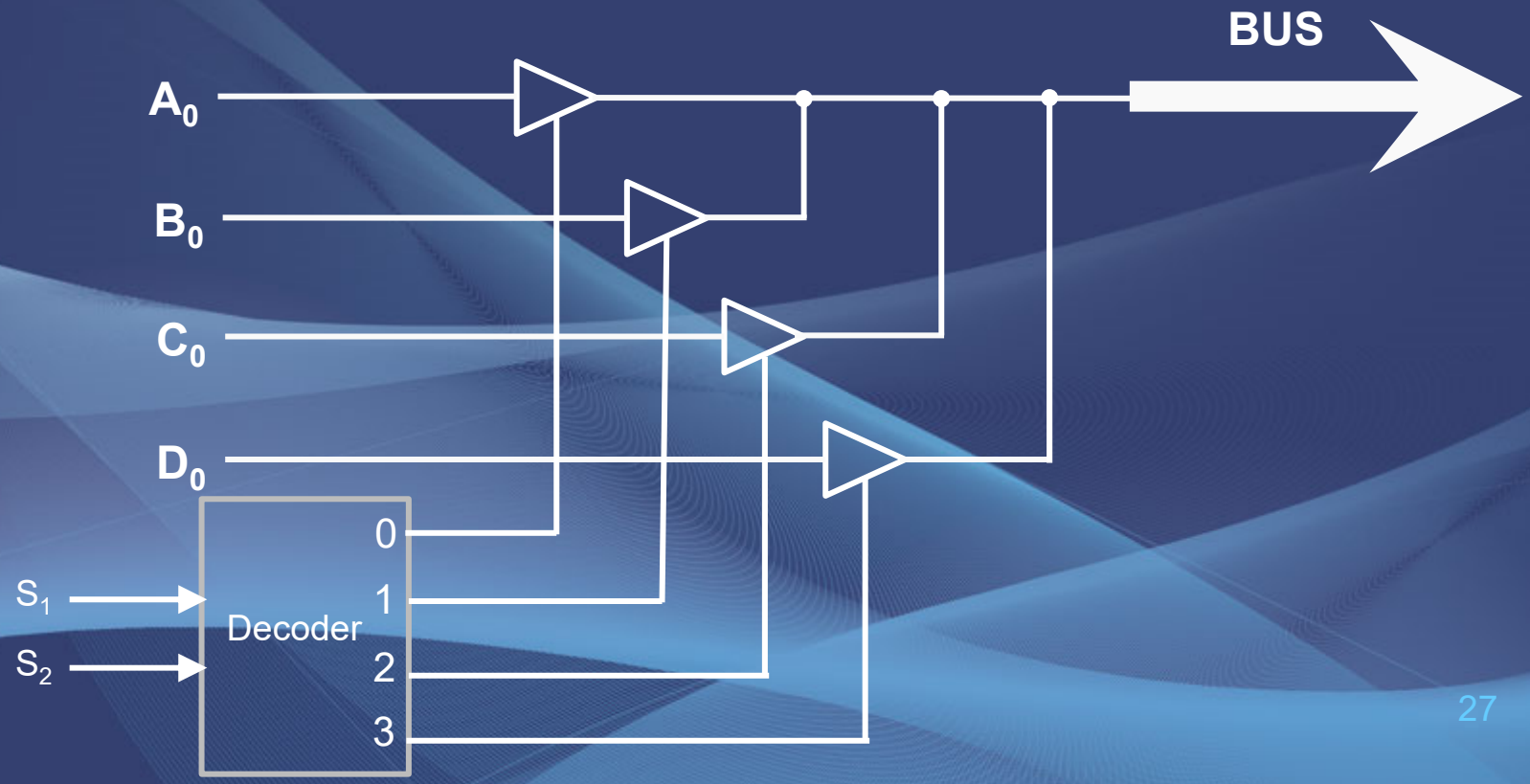
- تعداد بیت های ثبات ها با تعداد MUX ها برابر است.

گذرگاه مشترک به کمک بافر سه حالته

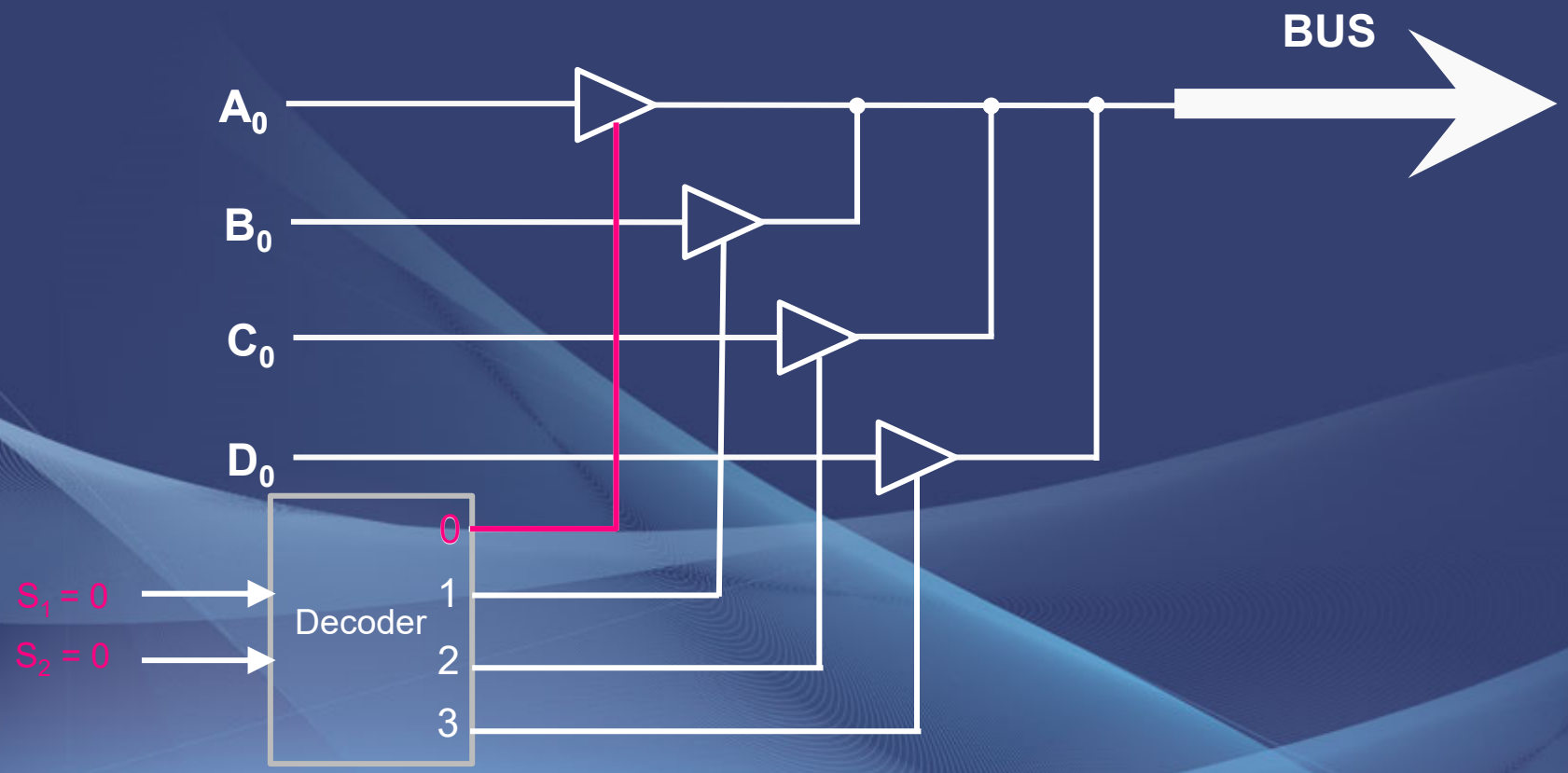
بافر سه حالته



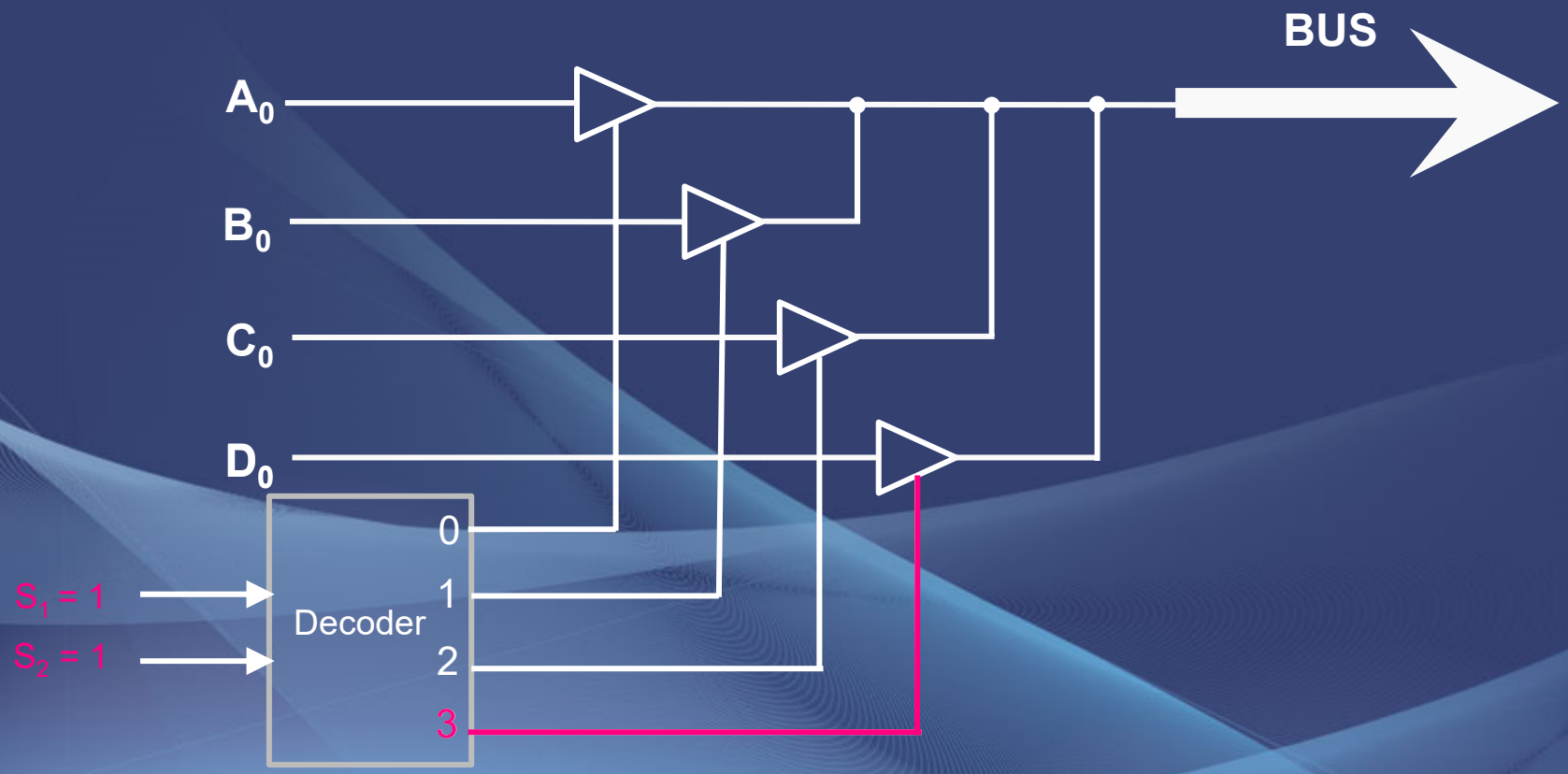
if C=1 → Y = A
if C=0 → High-Impedance



گذرگاه مشترک به کمک بافر سه حالته



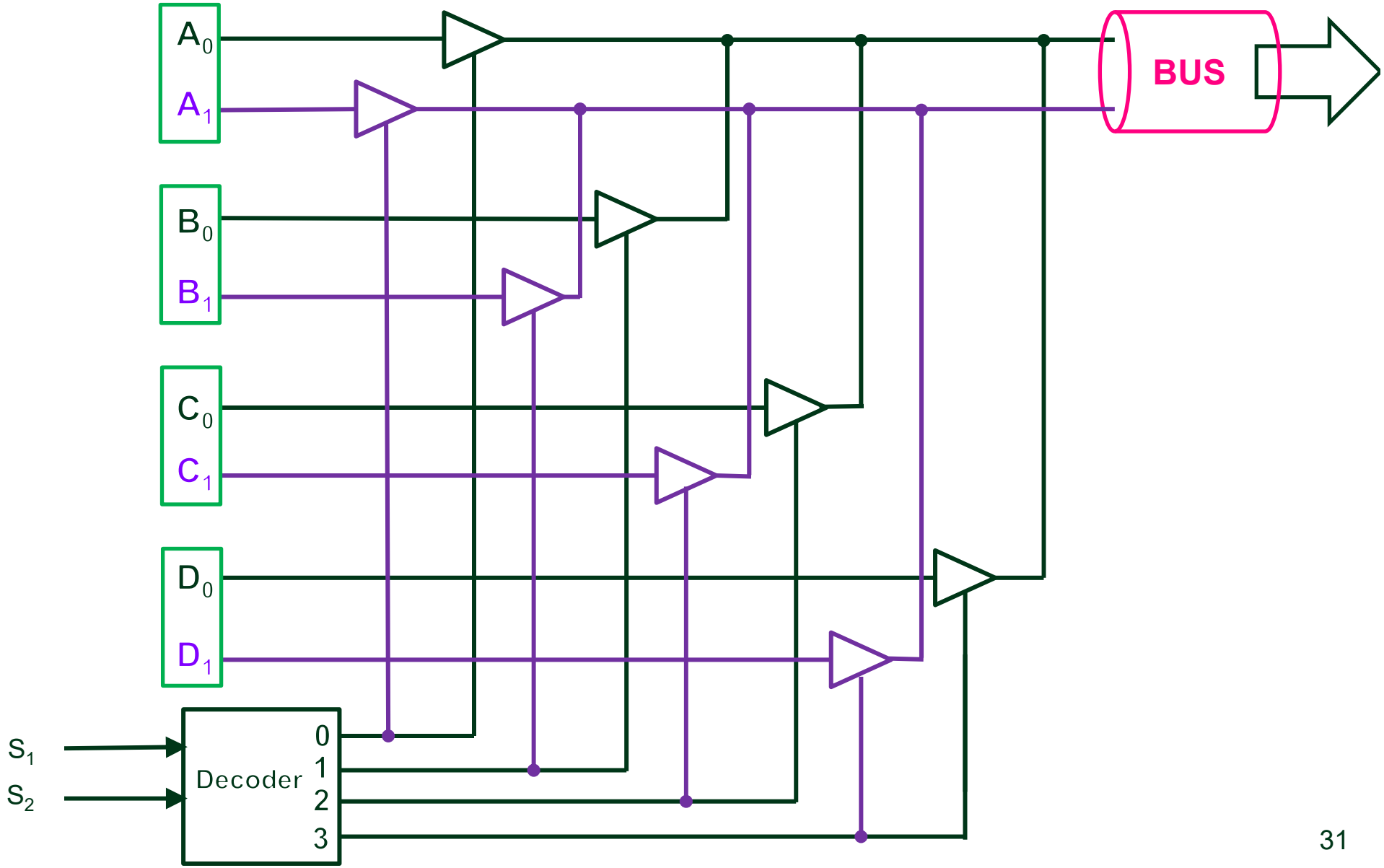
گذرگاه مشترک به کمک بافر سه حالته



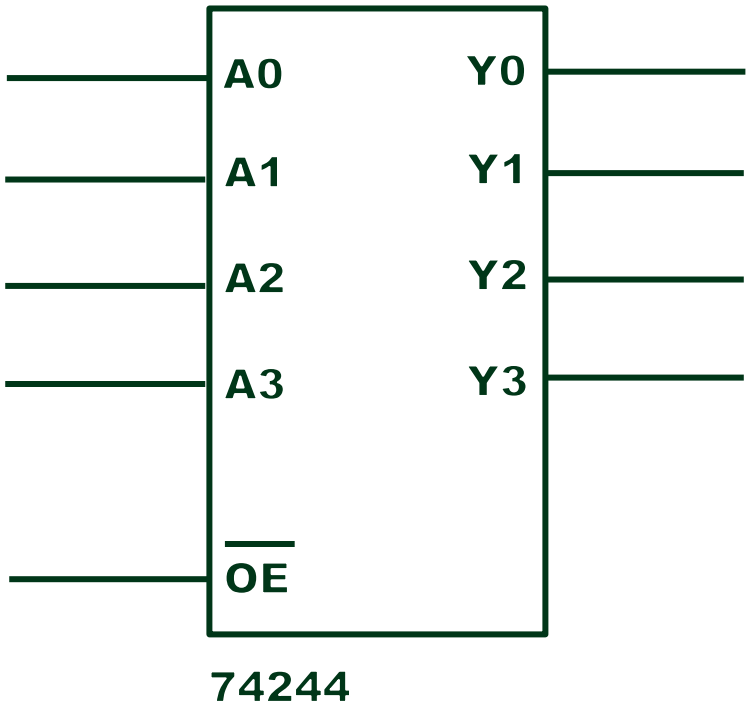
گذرگاه مشترک به کمک بافر سه حالته

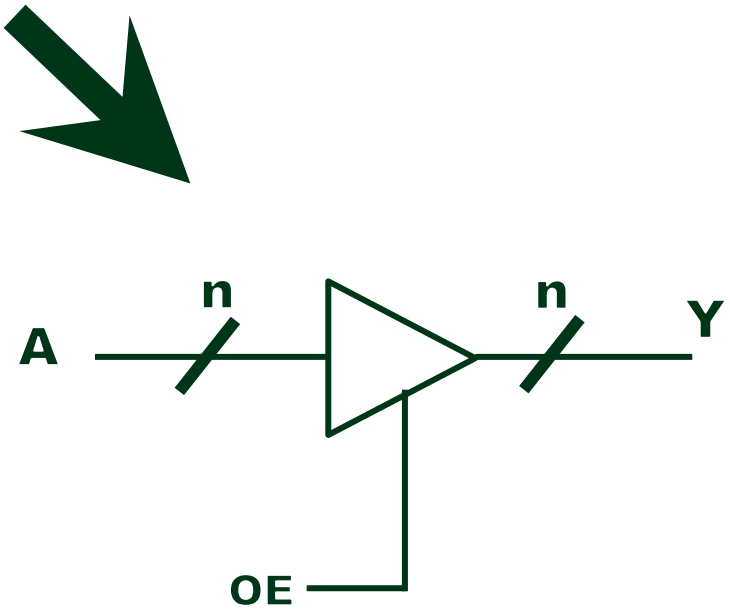
اگر k ثبات n بیتی داشته باشیم، برای ایجاد گذرگاه مشترک به کمک بافر سه حالته به یک دیکدر k خروجی و ($\log_2 k$ x k) و n x k بافر سه حالته نیاز داریم.

گذرگاه مشترک به کمک بافر سه حالته

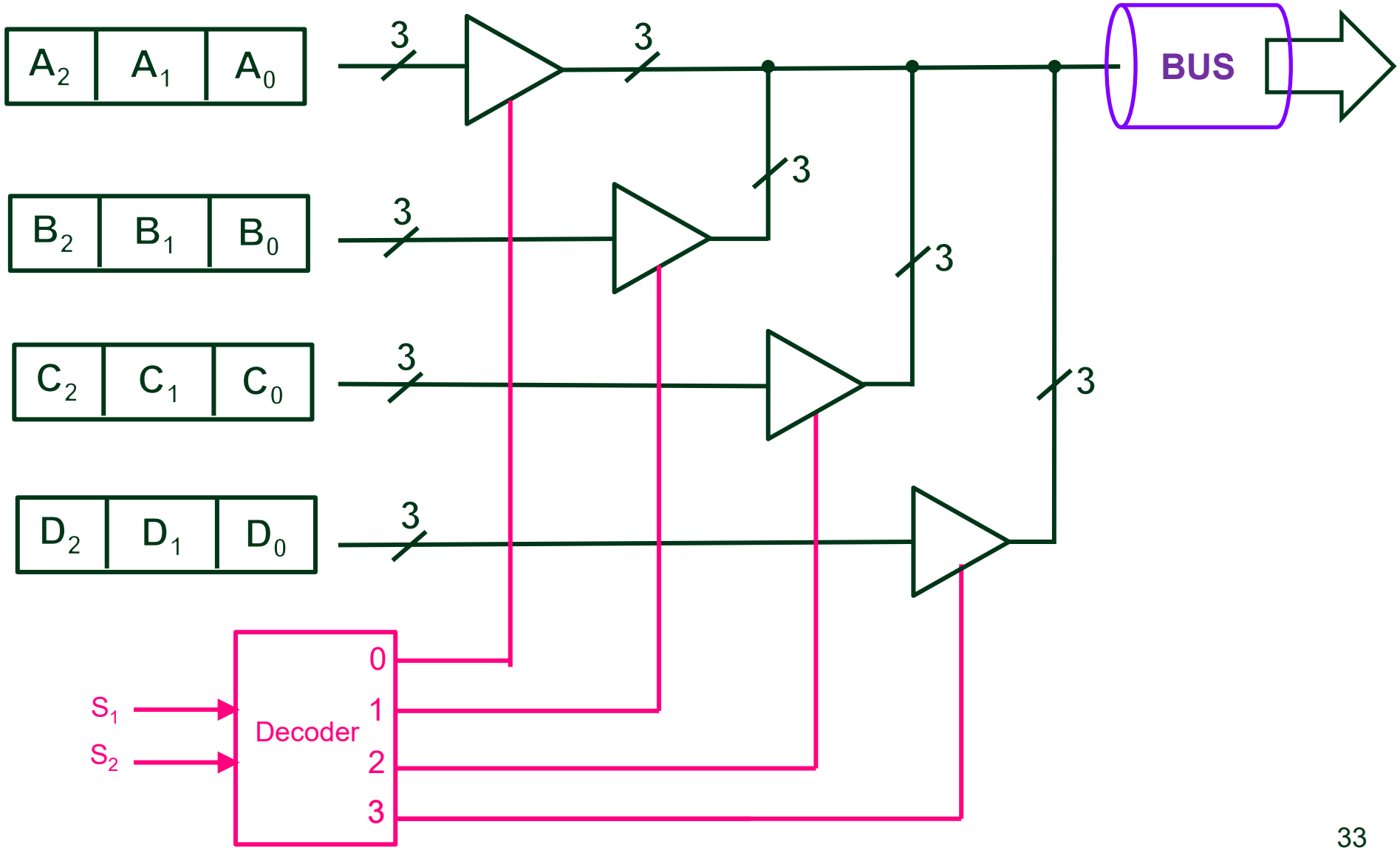


گذرگاه مشترک به کمک بافر سه حالته





گذرگاه مشترک به کمک بافر سه حالته



نشان دادن انتقال گذرگاه در RTL

- انتقال ثبات از طریق گذرگاه می تواند به یکی از دو شکل زیر نشان داده شود.

$R2 \leftarrow R1$

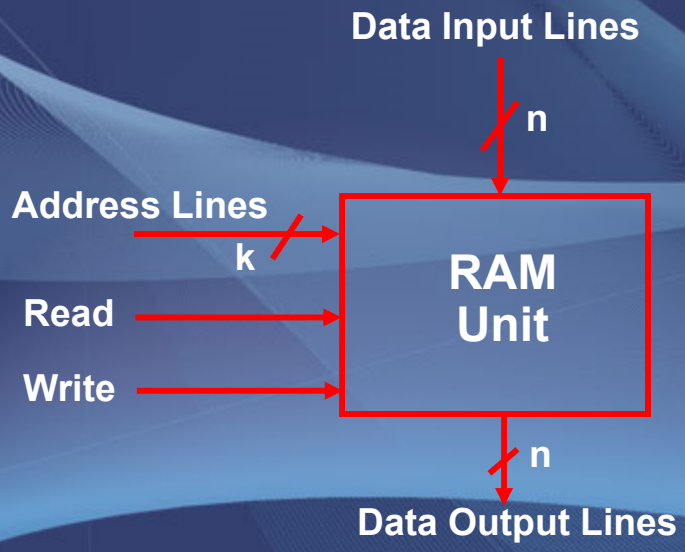
یا

$BUS \leftarrow R1, R2 \leftarrow BUS$

- در اولی گذرگاه به صورت ضمنی وجود دارد درحالی که در دومی به طور صریح بیان شده است.

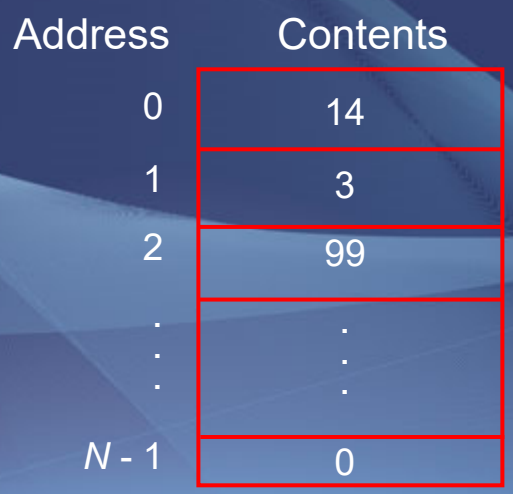
حافظه (RAM)

- هر کلمه با یک آدرس مشخص می شود.
- برای r کلمه آدرس دهی از 0 تا r-1 می باشد.
- هر کلمه می تواند n بیت را ذخیره کند.
- یک RAM با $r = 2^k$ کلمه را در نظر بگیرید. این RAM به موارد زیر نیاز دارد:
  - n خط داده ورودی
  - n خط داده خروجی
  - k خط آدرس
  - یک خط کنترل خواندن
  - یک خط کنترل نوشتن



مدل حافظه

- حافظه را به عنوان یک آرایه بزرگ از n عدد صحیح در نظر بگیرید، که بوسیله اندیس قابل دستیابی است.(حافظه با دستیابی تصادفی موسوم به ram)



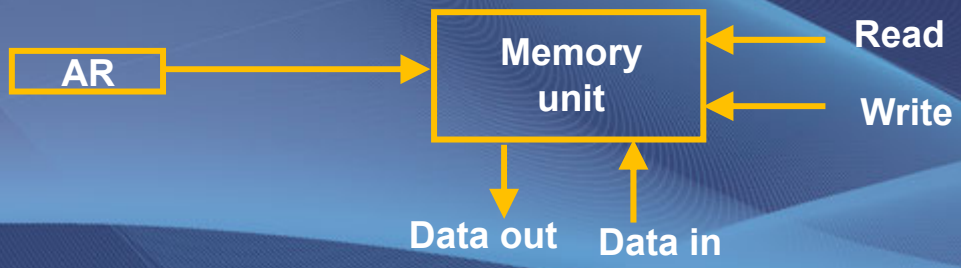
به عنوان نمونه ، M[1] شامل مقدار ۳ است. ما می توانیم در این مکانها نوشتن و خواندن را انجام دهیم. این مکانها صرفا در دسترس ماست. تمام مکانهای "مجرد" (از قبیل متغیرها در C) باید مکانهایی را در M تعیین کنند.

انتقال حافظه

- در سطح انتقال ثبات یک حافظه به صورت یک نماد **M** نشان داده می شود.
- چون یک حافظه شامل چندین آدرس مختلف است، باید جای مورد نظر در حافظه مشخص شود.

- در سیستم های کامپیوتری برای دستیابی به حافظه، آدرس دلخواه در یک ثبات مشخص قرار داده می شود. این ثبات ***Memory Address Register*** ***(MAR یا AR)*** نامیده می شود.

- وقتی حافظه دستیابی می شود، محتوای **MAR** به عنوان آدرس روی خطوط آدرس حافظه مورد استفاده قرار می گیرد.



خواندن از حافظه

- برای خواندن یک کلمه ازحافظه زبان انتقال ثبات بصورت زیر است:

$R1 \leftarrow M[MAR]$

- برای انجام مثال فوق اعمال زیر انجام می پذیرد:
  - محتوای MAR روی خطوط آدرس فرستاده می شود.
  - سیگنال Read (= 1) به واحد حافظه فرستاده می شود.
  - محتوای آدرس مشخص شده روی خطوط داده قرار می گیرد.
  - این مقدار از گذرگاه به ثبات R1 منتقل می شود.

نوشتن در حافظه

- برای نوشتن در یک کلمه حافظه زبان انتقال ثبات بصورت زیر نوشته می شود:

$M[MAR] \leftarrow R1$

- برای انجام مثال فوق اعمال زیر انجام می پذیرد:
  - محتوای MAR روی خطوط آدرس فرستاده می شود.
  - سیگنال write (= 1) به واحد حافظه فرستاده می شود.
  - مقدار ثبات R1 به گذرگاه منتقل می شود.
  - مقدار روی باس به محل مشخص شده در حافظه منتقل می شود.

خلاصه ای از ریزعمل های انتقال ثبات

A ← B	Transfer content of reg. B into reg. A
AR ← DR(AD)	Transfer content of AD portion of reg. DR into reg. AR
A ← constant	Transfer a binary constant into reg. A
ABUS ← R1,	Transfer content of R1 into bus A and, at the same time,
R2 ← ABUS	Transfer content of bus A into R2
AR	Address register
DR	Data register
M[R]	Memory word specified by reg. R
M	Equivalent to M[AR]
DR ← M	Memory *read* operation: transfers content of Memory word specified by AR into DR
M ← DR	Memory *write* operation: transfers content of DR into memory word specified by AR

انواع ریزعمل ها

ریزعمل ها در سیستم کامپیوتری به چهار دسته مختلف تقسیم می شوند:

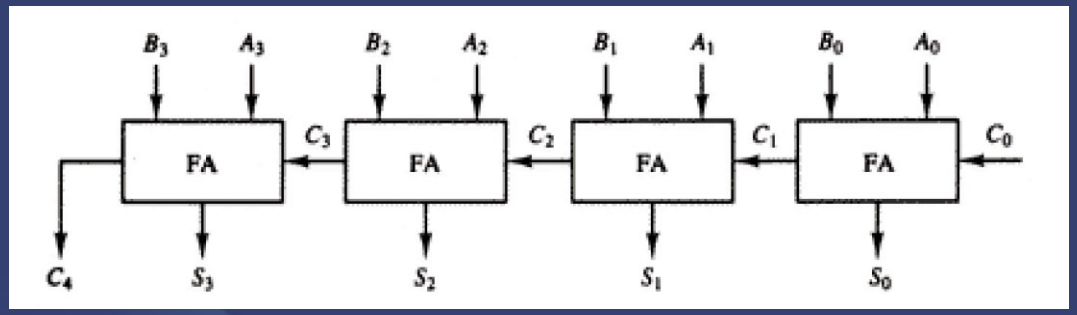
- ریزعمل های انتقال ثبات (Register Transfer Microoperations)
- ریزعمل های حسابی (Arithmetic Microoperations)
- ریزعمل های منطقی (Logic Microoperations)
- ریزعمل های شیفت (Shift Microoperations)

ریزعمل های حسابی

- ریزعمل های حسابی پایه عبارت اند:
  - جمع
  - تفریق
  - افزایش یک واحد
  - کاهش یک واحد

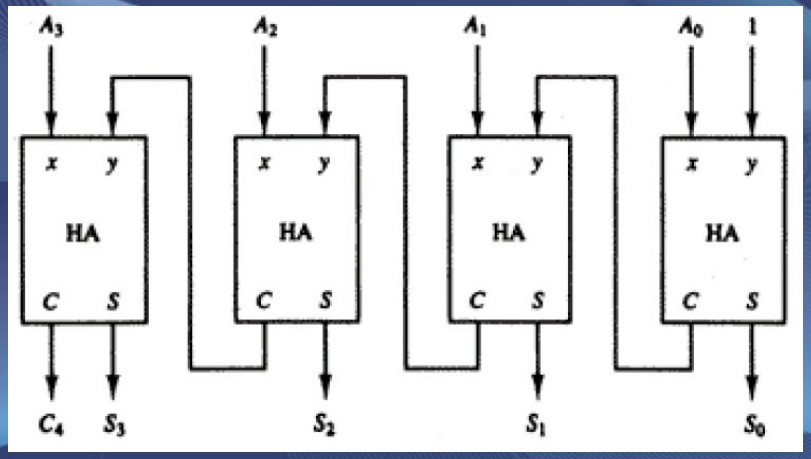
R3 ← R1 + R2	Contents of R1 plus R2 transferred to R3
R3 ← R1 - R2	Contents of R1 minus R2 transferred to R3
R2 ← R2’	Complement the contents of R2
R2 ← R2’+ 1	2's complement the contents of R2 (negate)
R3 ← R1 + R2’+ 1	subtraction
R1 ← R1 + 1	Increment
R1 ← R1 - 1	Decrement

افزایشگر، جمع کننده و تفریق کننده دودویی

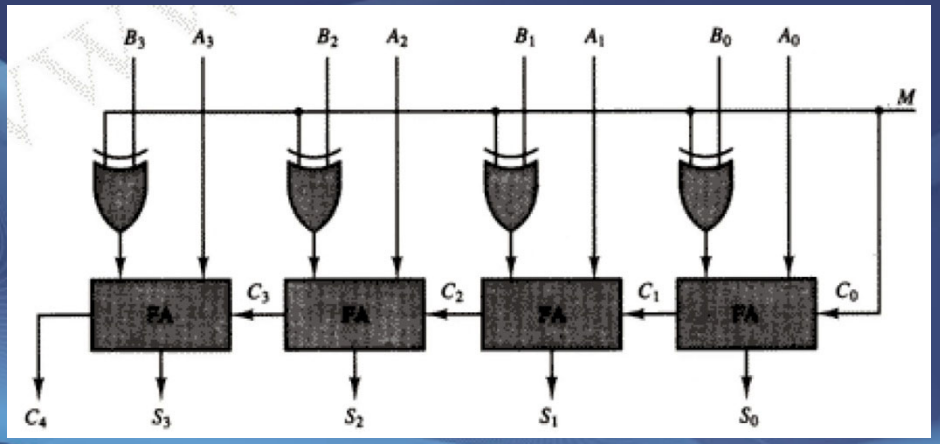


Binary Adder
جمع کننده دودویی

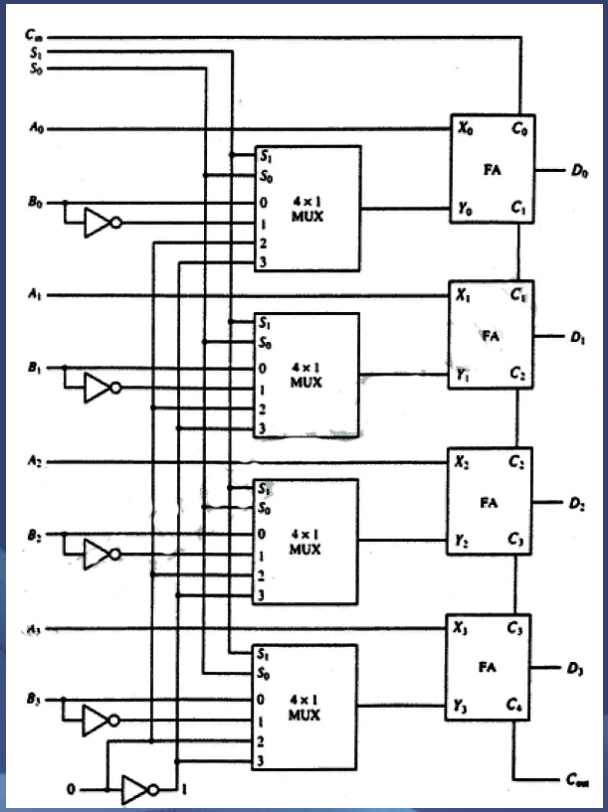
Binary Incrementer
افزایشگر دودویی



Binary Adder-Subtractor
جمع کننده-تفریق کننده دودویی



مدار عملیات حسابی



S1	S0	Cin	Y	Output	ریزعمل ها	
0	0	0	B	D = A + B	جمع	Add
0	0	1	B	D = A + B + 1	جمع با بیت انتقال	Add with carry
0	1	0	B’	D = A + B’	تفریق با بیت قرضی	Subtract with borrow
0	1	1	B’	D = A + B’+ 1	تفریق	Subtract
1	0	0	0	D = A	A انتقال	Transfer A
1	0	1	0	D = A + 1	A افزایش	Increment A
1	1	0	1	D = A - 1	A کاهش	Decrement A
1	1	1	1	D = A	A انتقال	Transfer A

ریز عمل های منطقی

- ریز عمل های منطقی ریزعمل هایی هستند که عملیات دودویی را روی رشته ای از بیت های ثبات انجام می دهند.
  - عملیات منطقی روی یک بیت داده کار می کنند به همین دلیل به آنها bit-wise می گویند. مثلا در یک ثبات هشت بیتی وقتی عمل not انجام می شود روی هر بیت به طور مستقل انجام می شود.
  - از عملیات منطقی می تواند برای دستکاری بیتی (bit manipulations) داده ها استفاده شود
- به طور کلی ۱۶ عملیات متفاوت منطقی می تواند روی دو متغیر دودویی انجام شود.

x	y	F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

- بیشتر سیستمها فقط چهار عمل زیر را پیاده سازی می کنند.
  - AND ($\wedge$), OR ($\vee$), XOR ($\oplus$), Complement/NOT
- عملیات دیگر می توانند با استفاده از این چهار ریزعمل ساخته شوند.

لیست ریزعمل های منطقی

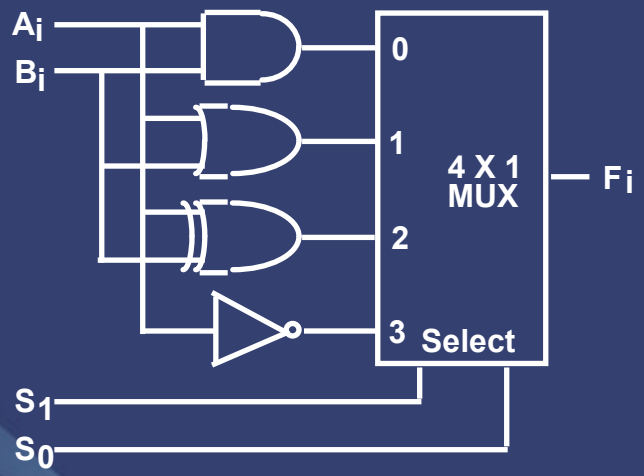
- لیست ریزعمل های منطقی

با n متغیر می توان 2^{2^n} عمل منطقی تعریف کرد ⟵ ۱۶ عمل منطقی مختلف روی ۲ متغیر

- جدول ارزش برای دو متغیر دودویی

x y	0 0 1 1 0 1 0 1	*Boolean Function*	*Micro-Operations*	*Name*
	0 0 0 0	F0 = 0	F ← 0	Clear
	0 0 0 1	F1 = xy	F ← A ∧ B	AND
	0 0 1 0	F2 = xy'	F ← A ∧ B’	
	0 0 1 1	F3 = x	F ← A	Transfer A
	0 1 0 0	F4 = x'y	F ← A’∧ B	
	0 1 0 1	F5 = y	F ← B	Transfer B
	0 1 1 0	F6 = x ⊕ y	F ← A ⊕ B	Exclusive-OR
	0 1 1 1	F7 = x + y	F ← A ∨ B	OR
	1 0 0 0	F8 = (x + y)'	F ← (A ∨ B)’	NOR
	1 0 0 1	F9 = (x ⊕ y)'	F ← (A ⊕ B)’	Exclusive-NOR
	1 0 1 0	F10 = y'	F ← B’	Complement B
	1 0 1 1	F11 = x + y'	F ← A ∨ B	
	1 1 0 0	F12 = x'	F ← A’	Complement A
	1 1 0 1	F13 = x' + y	F ← A’∨ B	
	1 1 1 0	F14 = (xy)'	F ← (A ∧ B)’	NAND
	1 1 1 1	F15 = 1	F ← all 1's	Set to all 1's

پیاده سازی سخت افزاری عملیات منطقی



جدول توابع

S_1	S_0	Output	μ-operation
0	0	$F = A \wedge B$	AND
0	1	$F = A \vee B$	OR
1	0	$F = A \oplus B$	XOR
1	1	$F = A'$	Complement

کاربردهای ریزعملهای منطقی

- ریزعمل های منطقی می توانند برای دستکاری بیتی مورد استفاده قرار گیرند. یعنی برای تغییر بیت های یک قسمت دلخواه از یک ثبات.

- فرض کنید داده ها در ثبات A هستند. ثبات B می تواند برای تغییر محتویات A به کار رود.

– Selective-set	$A \leftarrow A + B$
– Selective-complement	$A \leftarrow A \oplus B$
– Selective-clear	$A \leftarrow A \cdot B'$
– Mask (Delete)	$A \leftarrow A \cdot B$
– Clear	$A \leftarrow A \oplus B$
– Insert	$A \leftarrow (A \cdot B) + C$
– Compare	$A \leftarrow A \oplus B$
– . . .	

ست کردن انتخابی

- در یک کردن انتخابی B برای تعیین بیت هایی از A که قرار است یک شوند مورد استفاده قرار می گیرد.

1 1 0 0	A_t	
1 0 1 0	B	
1 1 1 0	A_{t+1}	$(A \leftarrow A + B)$

- به ازای بیت هایی که در B مقدار یک دارند، بیت های معادل آنها در A یک می شود. بقیه بیت های A بدون تغییر می مانند.

مکمل کردن انتخابی

- در مکمل کردن انتخابی ثبات B برای تعیین بیت هایی از A که قرار است مکمل شوند مورد استفاده قرار می گیرد.

1 1 0 0	A_t	
1 0 1 0	B	
0 1 1 0	A_{t+1}	$(A \leftarrow A \oplus B)$

- به ازای بیت هایی که در B مقدار یک دارند، بیت های معادل آنها در A مکمل(NOT) می شود. بقیه بیت های A بدون تغییر می مانند.

پاک کردن انتخابی

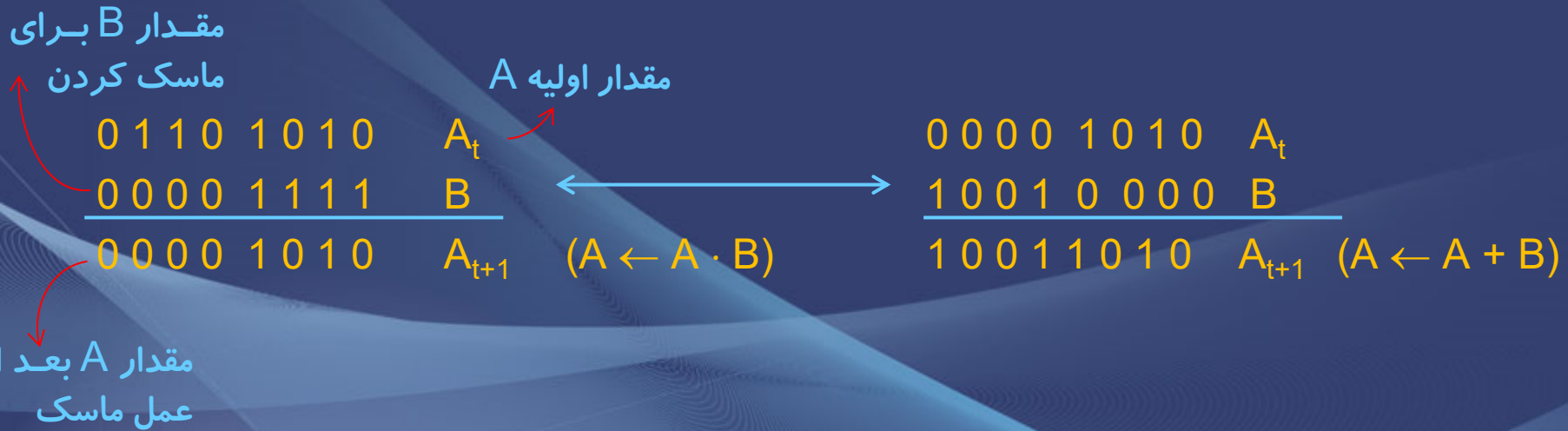
- در پاک کردن انتخابی ثبات B برای تعیین بیت هایی از A که قرار است پاک(صفر) شوند مورد استفاده قرار می گیرد.

$$
\begin{array}{ll}
1\ 1\ 0\ 0 & A_t \\
1\ 0\ 1\ 0 & B \\
\hline
0\ 1\ 0\ 0 & A_{t+1} \qquad (A \leftarrow A \cdot B')
\end{array}
$$

- به ازای بیت هایی که در B مقدار یک دارند، بیت های معادل آنها در A صفر می شود. بقیه بیت های A بدون تغییر می مانند.

عملیات ماسک کردن

- در عمل ماسک کردن B برای تعیین بیت هایی از A که قرار است پاک(صفر) شوند مورد استفاده قرار می گیرد.
-
- اگر ثبات A برابر با 1010 0110 باشد و بخواهیم چهار بیت بزرگتر A را برابر 1001 کنیم :



- به ازای بیت هایی که در B مقدار صفر دارند، بیت های معادل آنها در A صفر می شود. بقیه بیت های A بدون تغییر می مانند.

عملیات درج

- عملیات درج برای وارد کردن رشته بیت مورد نظر به درون ثبات مورد استفاده قرار می گیرد.
- روش انجام عمل درج:
  - ابتدا یک عمل ماسک برای پاک کردن بیت های مورد نظر انجام می شود.
  - سپس یک عمل OR برای قرار دادن بیت های جدید مورد استفاده قرار می گیرد.
  - مثال:
    - فرض کنید می خواهیم 1010 را به قسمت کم ارزش ثبات A وارد کنیم.

1101 1000 1011 0001 A (Original)
1101 1000 1011 1010 A (Desired)

```
1101 1000 1011 0001          A (Original)
1111 1111 1111 0000          Mask
1101 1000 1011 0000          A (Intermediate)
0000 0000 0000 1010          Added bits
1101 1000 1011 1010          A (Desired)
```

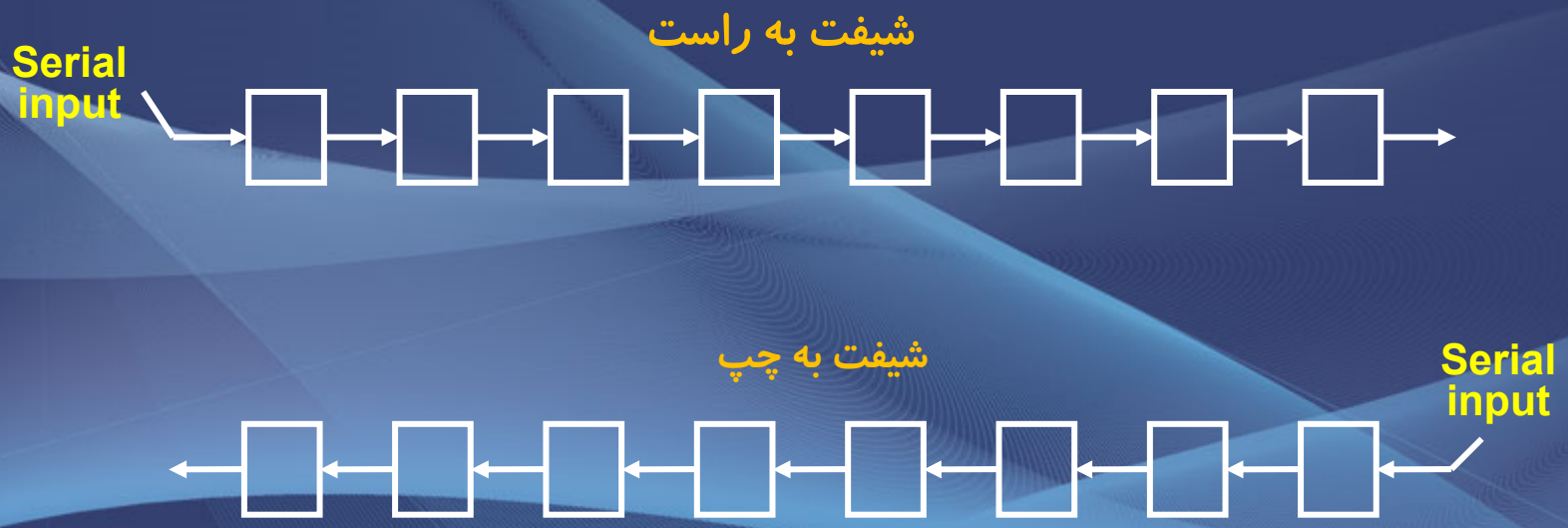
عملیات پاک کردن

- در عمل پاک کردن اگر بیت های A و B مشابه بود، بیت معادل در A صفر می شود، در غیر این صورت بدون تغییر می ماند. اگر دو عدد برابر باشند نتیجه تمام صفر خواهد بود(برای مقایسه مناسب است)

1 1 0 0	A_t	
1 1 0 0	B	
0 0 0 0	A_{t+1}	$(A \leftarrow A \oplus B)$

ریز عمل شیفت

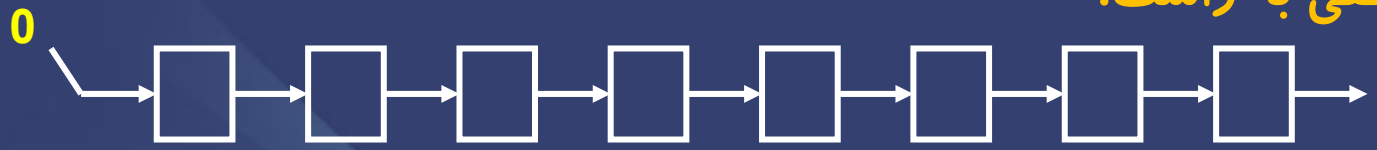
- در سیستم های دیجیتال سه نوع شیفت مختلف وجود دارد:
  - *شیفت منطقی*
  - *شیفت چرخشی*
  - *شیفت حسابی*
- تفاوت این شیفت ها در بیت ورودی سریال است.



شیفت منطقی

- در شیفت منطقی که برای اعداد بی علامت استفاده می شود، بیت ورودی، صفر است.

- شیفت منطقی به راست:



- شیفت منطقی به چپ:

- در زبان انتقال ثبات (*RTL*) از علائم زیر استفاده می شود:
  - *shl* شیفت منطقی به چپ
  - *shr* شیفت منطقی به راست

مثال:

- R2 ← *shr* R2
- R3 ← *shl* R3

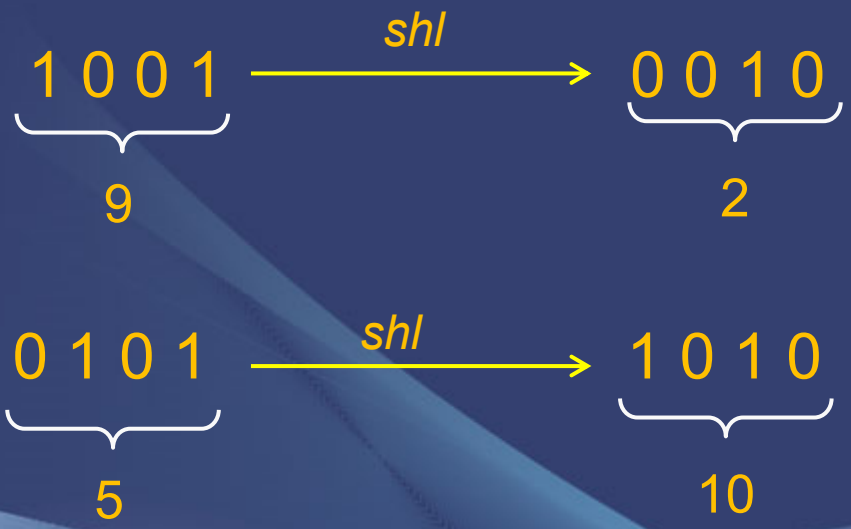
شیفت منطقی (مثال)

- شیفت منطقی به راست

- شیفت منطقی به راست، معادل جزء صحیح تقسیم بر ۲ است .

شیفت منطقی (مثال)

- شیفت منطقی به چپ

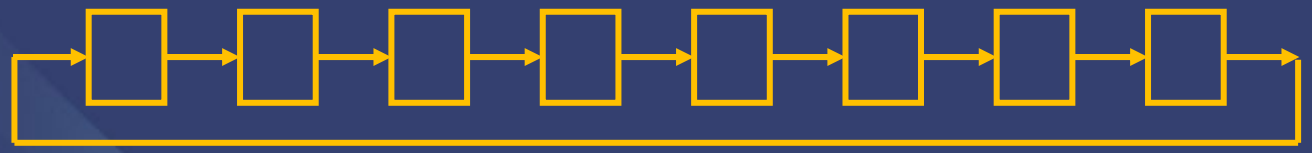


- شیفت منطقی به چپ، معادل ضرب در ۲ است (به شرطی که سریز رخ ندهد).

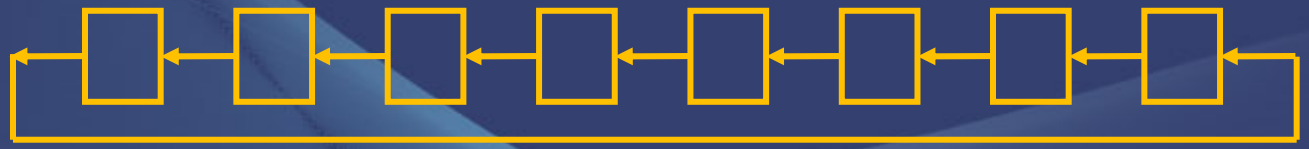
- سرریز وقتی رخ می دهد که بیت سمت چپ یک باشد و هنگام شیفت منطقی به چپ، آن یک را از دست می دهیم.

شیفت چرخشی

- در شیفت چرخشی بیت ورودی سریال، بیت خروجی از سمت دیگر ثبات است.
- شیفت چرخشی به راست:



- شیفت چرخشی به چپ:



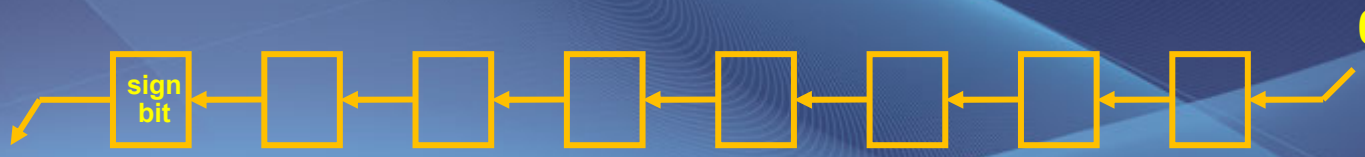
- در زبان انتقال ثبات از علائم زیر استفاده می شود:
  - *cil* شیفت چرخشی به چپ
  - *cir* شیفت چرخشی به راست
  - مثال:
    - R2 ← *cir* R2
    - R3 ← *cil* R3

شیفت حسابی

- شیفت حسابی برای اعداد علامت دار مکمل ۲ مناسب است.
- مهمترین ویژگی شیفت حسابی آن است که به هنگام شیفت (ضرب و تقسیم) علامت ثبات را حفظ می کند.

- شیفت حسابی به راست:

- شیفت حسابی به چپ:



شیفت حسابی

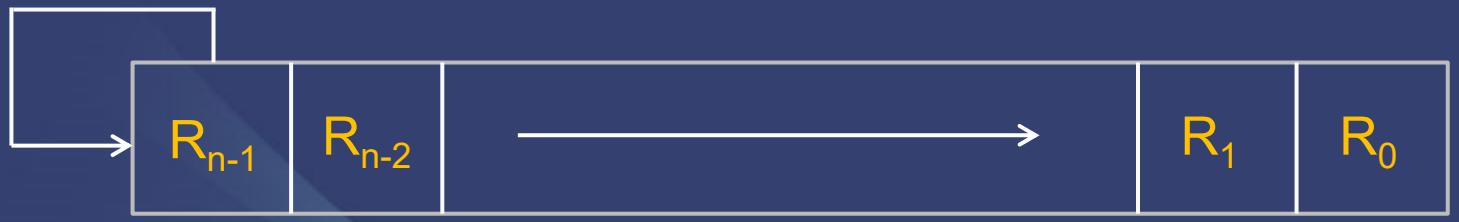
- در زبان انتقال ثبات از علائم زیر استفاده می شود:

  - *ashl* شیفت حسابی به چپ
  - *ashr* شیفت حسابی به راست

  - مثال:
    - R2 ← *ashr* R2
    - R3 ← *ashl* R3

شیفت حسابی به راست

- در شیفت حسابی به راست، علامت عدد از سمت چپ وارد می شود

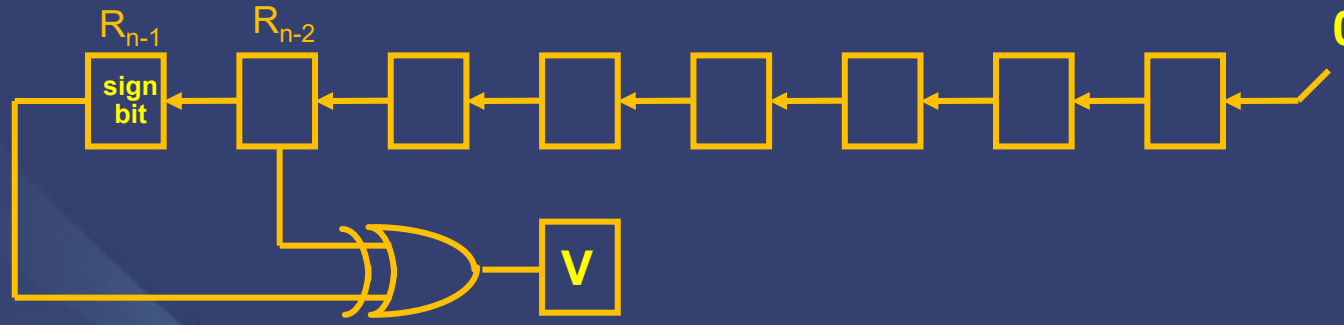


- شیفت حسابی به راست معادل جزء صحیح تقسیم عدد درون ثبات بر ۲ است.

$$\underbrace{1\ 0\ 0\ 1}_{-7} \xrightarrow{ashr} \underbrace{1\ 1\ 0\ 0}_{-4} = \left[\frac{-7}{2}\right]$$

شیفت حسابی به چپ

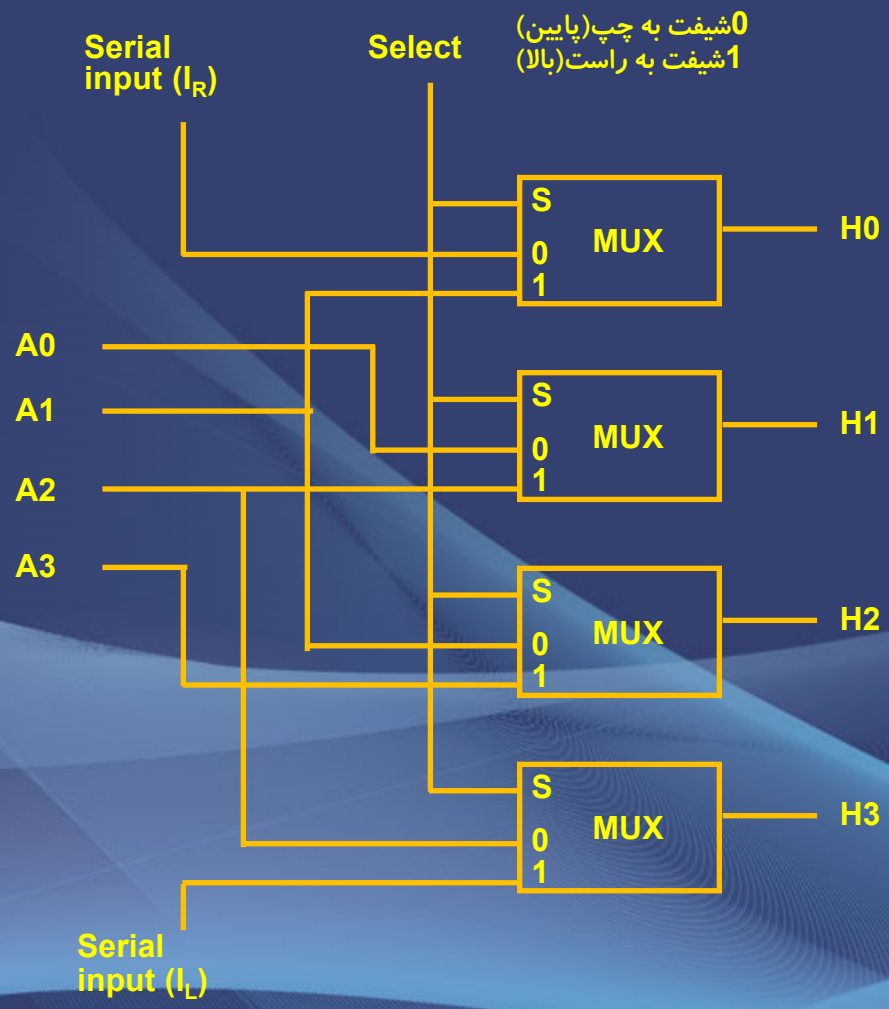
- در شیفت به چپ باید مساله سرریز(*overflow*) چک شود.



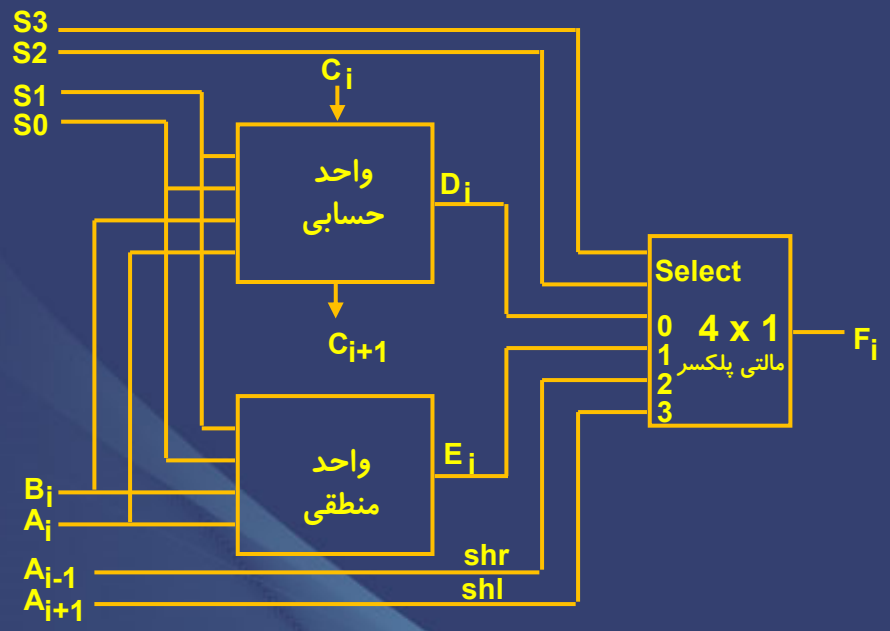
- اگر **قبل از شیفت** مقدار دو بیت آخر متفاوت باشد، سرریز رخ داده است؛ به عبارتی : $V = R_{n-1} \oplus R_{n-2}$
- در شیفت حسابی به چپ نیز مثل شیفت منطقی به چپ از سمت راست صفر وارد می شود :

- شیفت حسابی به چپ عدد درون ثبات را در ۲ ضرب می کند.

پیاده سازی سخت افزاری شیفت ها



واحد عملیات شیفت، منطقی، حسابی



S3	S2	S1	S0	Cin	عملیات	توضیح
0	0	0	0	0	F = A	Transfer A
0	0	0	0	1	F = A + 1	Increment A
0	0	0	1	0	F = A + B	Addition
0	0	0	1	1	F = A + B + 1	Add with carry
0	0	1	0	0	F = A + B’	Subtract with borrow
0	0	1	0	1	F = A + B’+ 1	Subtraction
0	0	1	1	0	F = A - 1	Decrement A
0	0	1	1	1	F = A	TransferA
0	1	0	0	X	F = A ∧ B	AND
0	1	0	1	X	F = A ∨ B	OR
0	1	1	0	X	F = A ⊕ B	XOR
0	1	1	1	X	F = A’	Complement A
1	0	X	X	X	F = shr A	Shift right A into F
1	1	X	X	X	F = shl A	Shift left A into F

پایان فصل
عملیات نقل و انتقال ثبات‌ها

برای کسب اطلاعات بیشتر در مورد این درس می‌توانید به وب سایت آموزشی در لینک زیر مراجعه نمایید

kws=2vkdilhldqdhgxfdwlrq1u