معماری کامپیوتر

عملیات نقل و انتقال ثبات‌ها

فصل چهارم کتاب موریس مانو

محمدعلی شفیعیان

http://shafieian-education.ir

زمستان ۹۷

عملیات نقل و انتقال ثبات‌ها

- زبان نقل و انتقال ثبات‌ها (Register Transfer Logic ≡ RTL)
- نقل و انتقال ثبات‌ها
- گذرگاه تبادلات با حافظه
- ریزعملیات ریاضی (حسابی)
- ریزعملیات منطقی
- ریزعملیات شیفت

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:
  - ✓ ثبات‌هایی که در سیستم موجود است.
  - ✓ عملیاتی که سیستم انجام می دهد.

- برای شناسایی یک سیستم باید بدانیم:
  - ✓ چه عملیاتی روی داده‌ها انجام می‌شود.
  - ✓ چه اطلاعاتی بین ثبات‌ها منتقل می‌شود.

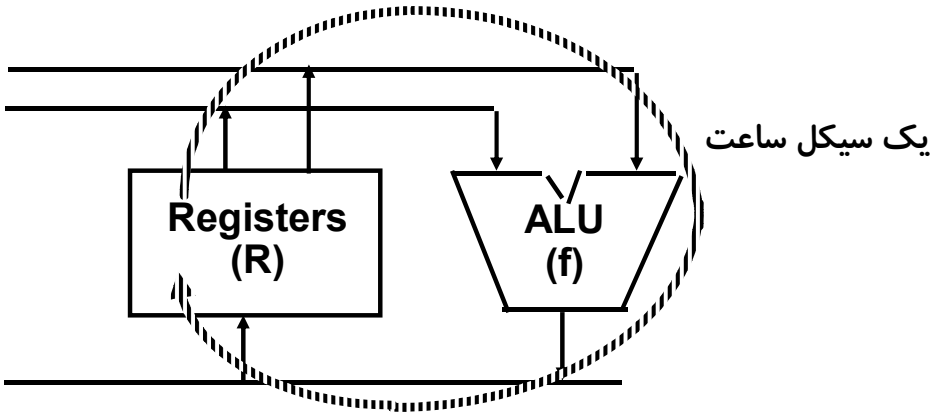
ریزعمل ها

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

- عملیات داخلی ثبات‌ها نمونه هایی از ریزعمل ها هستند.
  - ✓ شیفت (Shift)
  - ✓ بارکردن (Load)
  - ✓ پاک کردن (Clear)
  - ✓ اضافه کردن (Increment)
  - ✓ ...

ریزعمل ها

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



$R3 \leftarrow f(R1, R2)$

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

سازمان کامپیوتر

تعریف سازمان (Organization) داخلی کامپیوتر

سازمان سخت‌افزاری داخل کامپیوتر با سه خصوصیت زیر تعریف می‌شود:

- ✓ **مجموعه ثبات‌ها و وظیفه هر یک.**
- ✓ **مجموعه ریزعمل‌ها (رشته ریزعمل‌هایی که روی ثبات‌ها انجام می‌شود).**
- ✓ **سیگنال‌های کنترلی که ترتیب ریزعمل ها را مشخص می کنند.**

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:
  - ✓ ثبات‌های سیستم
  - ✓ تبدیل داده‌ها درون ثبات‌ها
  - ✓ انتقال داده‌ها بین ثبات‌ها

زبان انتقال ثبات

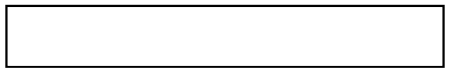
- به جای مشخص کردن یک سیستم با کلمات، آن را با یک نوشتن خاص که زبان انتقال ثبات (RTL) نامیده می شود نشان می‌دهند.

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

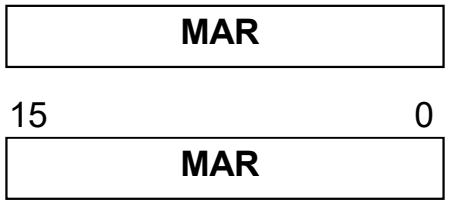
- زبان انتقال ثبات:
  - ✓ یک زبان سمبولیک است.
  - ✓ یک ابزار آسان برای شرح سازمان داخلی کامپیوترهای دیجیتال است.
  - ✓ فرآیند طراحی سیستم های دیجیتال را تسهیل می کند.

نامگذاری ثبات‌ها

- **ثباتها معمولا با حروف بزرگ نامگذاری می شوند. گاهی اوقات پس از اسم آنها اعداد قرار می‌گیرد.**(A, R13, IR)
- **اغلب نام‌ها نشان‌دهنده کاری است که ثبات انجام می شود مثلاً:**
  - MAR - memory address register
  - PC - program counter
  - IR- instruction register



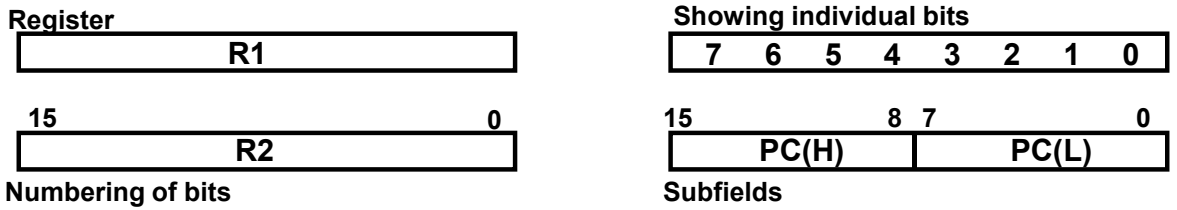
- **با نشان دادن بیتهای ثبات**



بلوک دیاگرام

ثبات‌ها و محتوای آنها به طور نمادین می تواند به صورت های زیر نمایش داده شود

روش های معمول رسم بلوک دیاگرام ثبات‌ها



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 ها فقط زمانی که یک شرط خاص برقرار باشد باید اجرا شوند.**
- **این مساله 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 می‌شود**
- **تابع کنترلی به شکل زیر نشان داده می شود:**

P: R2 ← R1

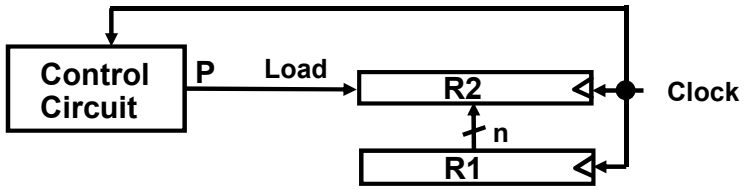
بدین معنی که اگر P برابر ۱ بود انتقال از R1 به R2 انجام شود. یا:

if (P = 1) then (R2 ← R1)

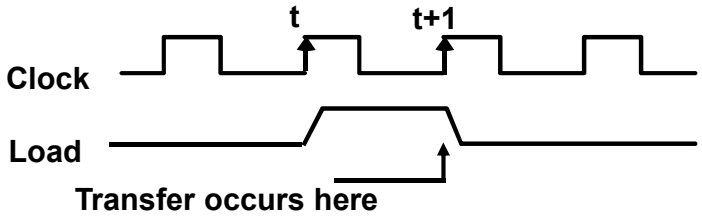
پیاده سازی سخت افزاری انتقالات کنترلی

P: R2 ← R1

بلوک دیاگرام



دیاگرام زمان بندی



- فرض می شود است که ثبات ها حساس به لبه مثبت هستند

عملیات همزمان

- اگر بخواهیم 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.

علائم اولیه

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

ارتباط بین ثبات ها

- در یک سیستم دیجیتال با ثبات‌های فراوان، اتصال مستقیم هر ثبات با ثبات‌های دیگر امکان‌پذیر نیست.

- برای اتصال n ثبات به یکدیگر به n(n-1)خط ارتباطی نیاز است.
- اگر هر ثبات نیز m بیت باشد در کل به n(n-1)m خط ارتباطی نیاز است.
  - ✓ برای سیستم های با تعداد ثبات زیاد عملی نیست.

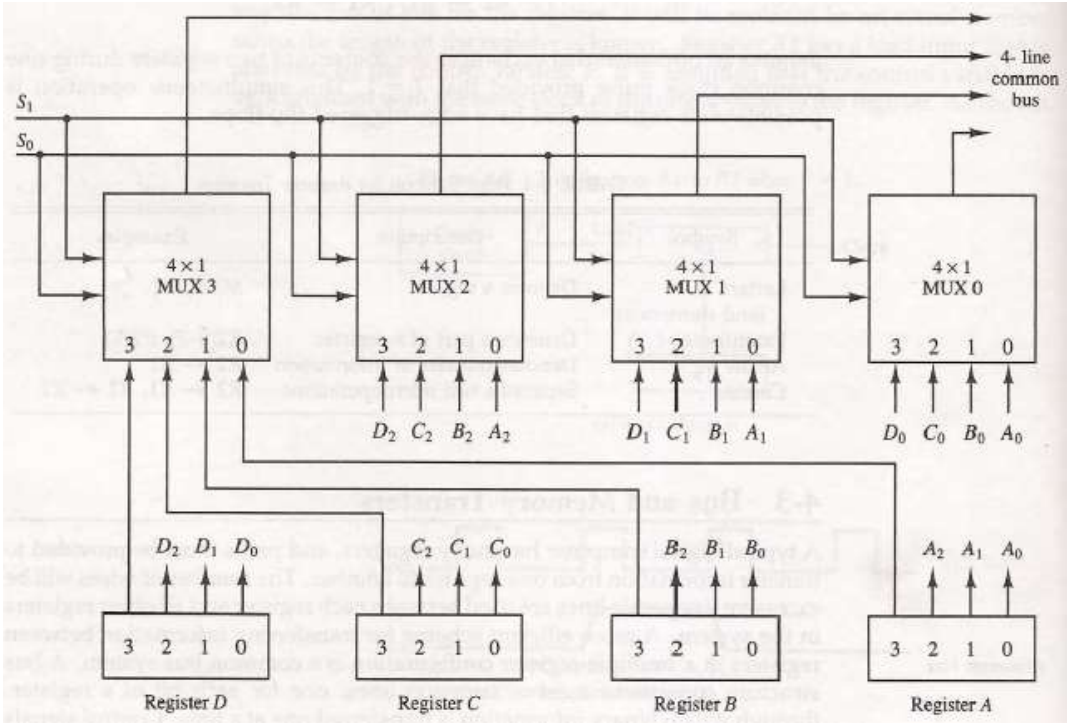
- به جای این کار از یک مجموعه مدار متمرکز به نام گذرگاه (bus) برای انتقال اطلاعات استفاده می‌شود.

- همچنین مدارهای کنترلی برای اینکه تعیین کنیم کدام ثبات، ثبات منبع و کدام ثبات مقصد است.

گذرگاه

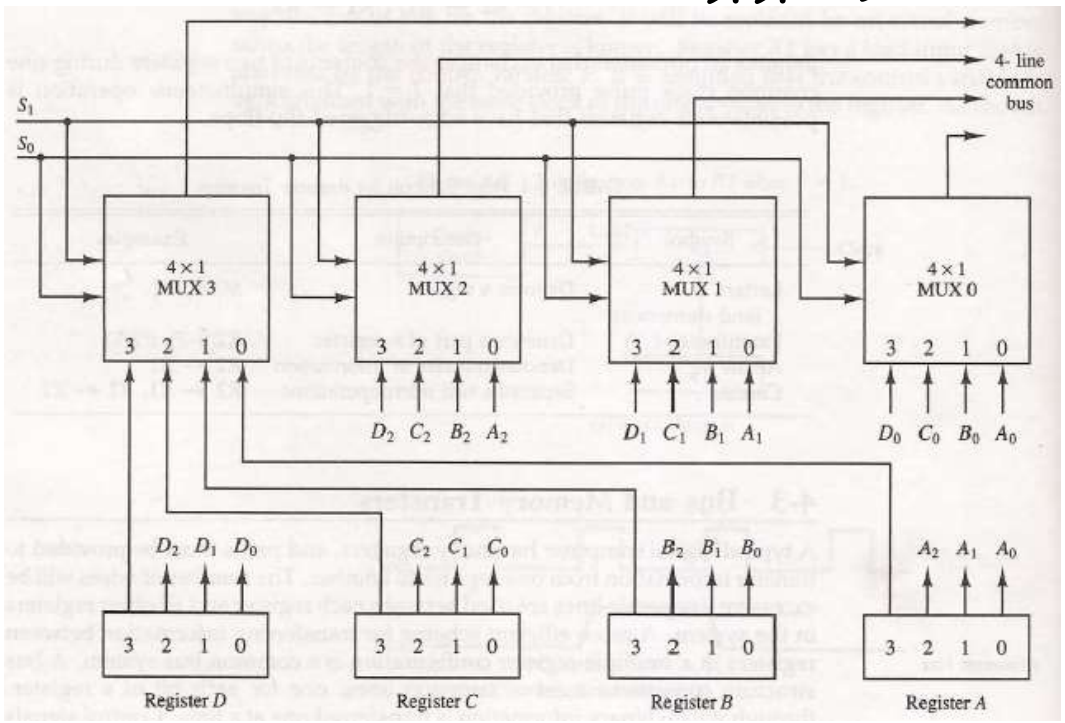
- ✓ گذرگاه یک مسیر (متشکل از گروهی از سیم ها) که اطلاعات روی آن منتقل می شود.
- ✓ انتقال می تواند از منابع مختلف به مقصدهای مختلف باشد.

- ✓ ساختار گذرگاه از مجموعه ای از خطوط مشترک تشکیل می شود که تعداد آنها یک خط به ازای هر یک از بیت های ثبات ها است.

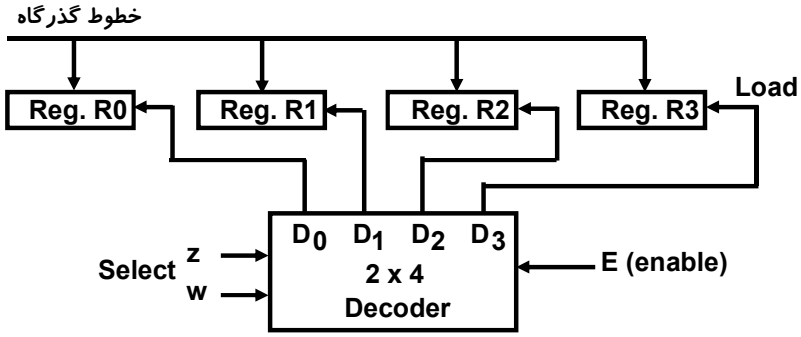


گذرگاه

- ✓ تعداد خطوط گذرگاه = تعداد بیت های ثبات ها = تعداد مالتی پلکسرها
- ✓ اگر n = تعداد ثبات ها باشد، آنگاه MUX nx1 (مالتی پلکسر n ورودی خواهد داشت)
- ✓ یک سیستم گذرگاه، k ثبات n بیتی را برای تشکیل یک گذرگاه مشترک n خطی راه‌دهی می‌نماید.
- ✓ تعداد مولتی پلکسرهای مورد نیاز برای ساخت گذرگاه برابر n است.
- ✓ اندازه هر مولتی پلکسر kx1 است.



انتقال از گذرگاه به یک ثبات



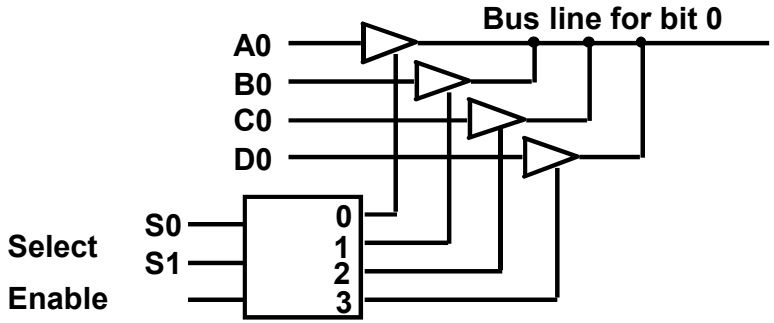
S_1	S_0	ثباتی که انتخاب می‌شود
0	0	A
0	1	B
1	0	C
1	1	D

انتقال اطلاعات از گذرگاه به داخل یکی از چند ثبات مقصد با اتصال خطوط گذرگاه به ورودی های تمام ثبات های مقصد و فعال کردن کنترل بار کردن ثبات مقصد خاصی که انتخاب شده تحقق می یابد.

گذرگاه با بافر سه حالته

بافر سه حالته

گذرگاه با بافر سه حالته



تعداد بافرهای سه حالته در هر قسمت از مدار = تعداد ثبات ها

تعداد مدارها = تعداد بیت های ثبات

نشان دادن انتقال گذرگاه در RTL

$$R2 \leftarrow R1 : \begin{cases} 1)\ BUS \leftarrow R1 \\ 2)\ R2 \leftarrow BUS \end{cases}$$

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

$$R2 \leftarrow R1$$

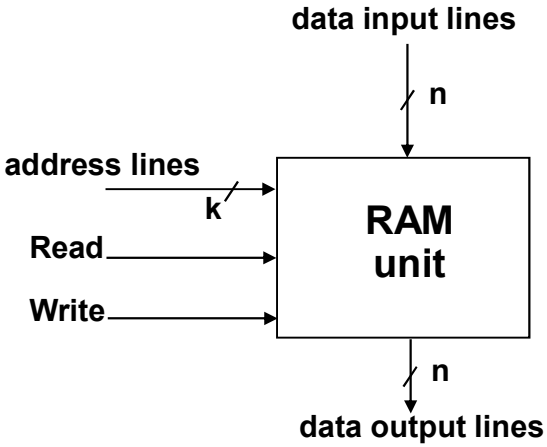
یا

$$BUS \leftarrow R1, R2 \leftarrow BUS$$

- در اولی گذرگاه به ضمنی وجود دارد درحالی‌که در دومی به طور صریح بیان شده است.

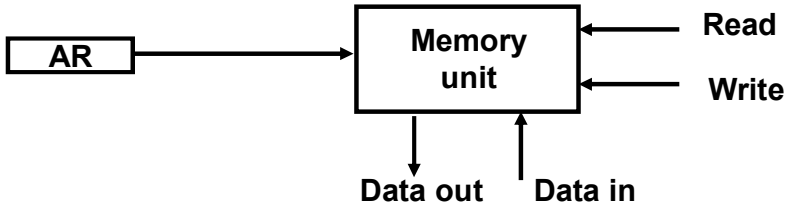
حافظه (RAM)

- حافظه (RAM) مداری است ترکیبی که شامل تعدادی ثبات است.
- هر ثبات یک کلمه (word) را تشکیل می دهد.
- هر ثبات با یک آدرس مشخص می شود.
- برای 2^r کلمه حافظه خطوط آدرس دهی باید r تا باشد (از 0 تا r-1).
- هر ثبات (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



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 AR به عنوان آدرس روی خطوط آدرس حافظه مورد استفاده قرار می گیرد.



خواندن از حافظه

- برای خواندن یک کلمه از حافظه، زبان انتقال ثبات به شکلی شبیه زیر نوشته می شود:

$$R1 \leftarrow M[AR]$$

- برای انجام مثال فوق اعمال زیر انجام می پذیرد:
  - ✓ محتوای AR 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 به شکلی شبیه زیر نوشته می شود:

$$M[AR] \leftarrow R1$$

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

مروری بر ریز عمل های انتقال ثبات

A ← B	انتقال از ثبات A به ثبات B
AR ← DR(AD)	انتقال قسمت AD از ثبات DR به ثبات AR
A ← constant	انتقال مقدار ثابت باینری به A
BUS ← R1, R2 ← BUS	انتقال همزمان از R1 به گذرگاه و از گذرگاه به R2
AR	ثبات آدرس
DR	ثبات داده
M[R]	کلمه حافظه مشخص شده با ثبات R
M	گاهی اوقات به جای M[AR] به کار می رود.
DR ← M	انتقال از کلمه مشخص شده با AR به ثبات DR
M ← DR	انتقال از ثبات DR به کلمه مشخص شده با AR

انواع ریزعمل ها

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

  - ✓ ریزعمل های انتقال ثبات
  - ✓ ریزعمل های حسابی
  - ✓ ریزعمل های منطقی
  - ✓ ریزعمل های شیفت

ریزعمل های حسابی

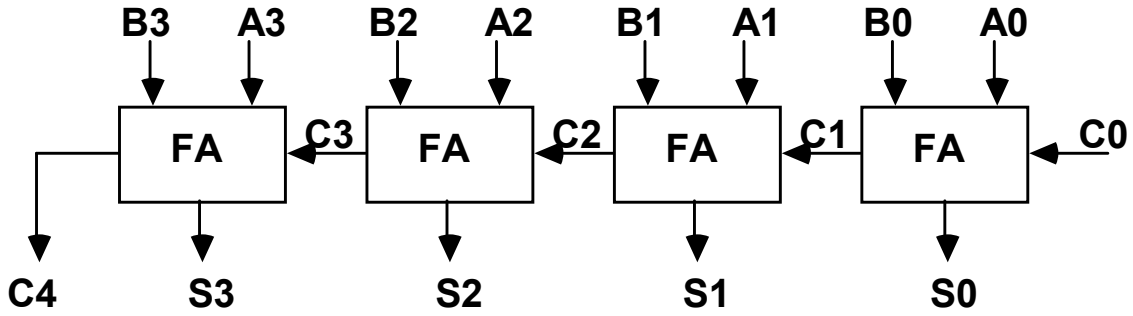
- ریزعمل های حسابی پایه عبارت اند:
  - ✓ جمع
  - ✓ تفریق
  - ✓ افزایش یک واحد
  - ✓ کاهش یک واحد

ریزعمل های معمول حسابی

جمع	R3 ← R1 + R2
تفریق	R3 ← R1 - R2
مکمل یک R2	R2 ← R2’
مکمل دوR2	R2 ← R2’+ 1
تفریق	R3 ← R1 + R2’+1
افزایش یک واحد	R1 ← R1 + 1
کاهش یک واحد	R1 ← R1 – 1

جمع کننده دودویی

Binary Adder
جمع کننده دودویی

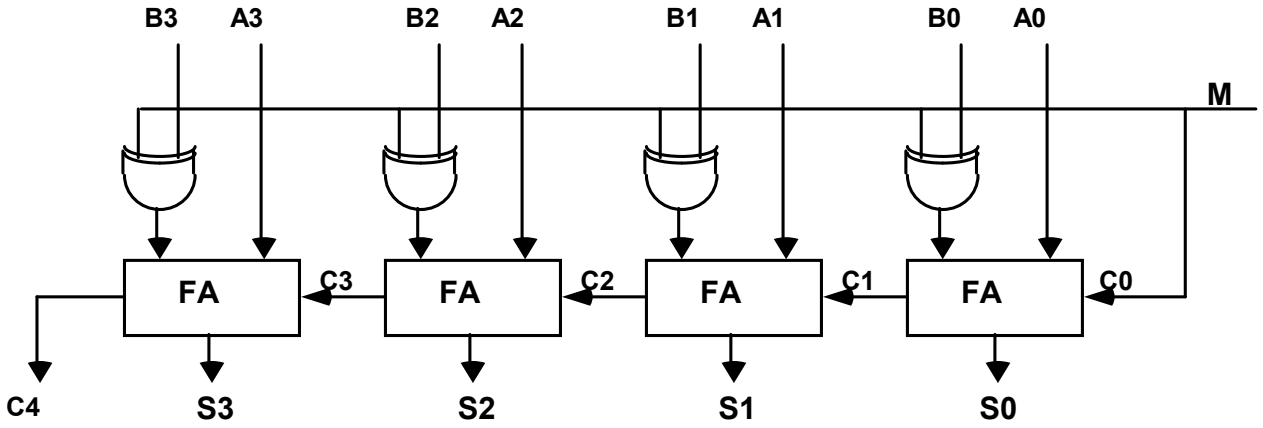


جمع کننده دودویی ۴ بیتی

تفریق کننده دودویی

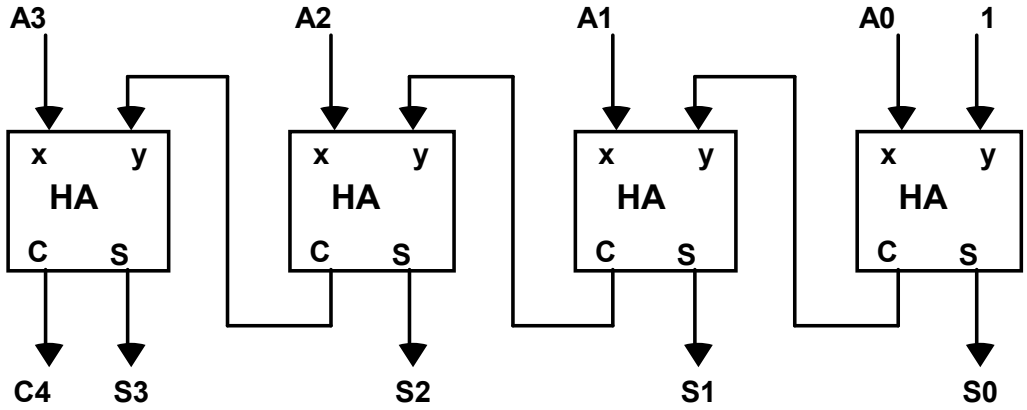
Binary Adder-Subtractor

جمع کننده–تفریق کننده دودویی

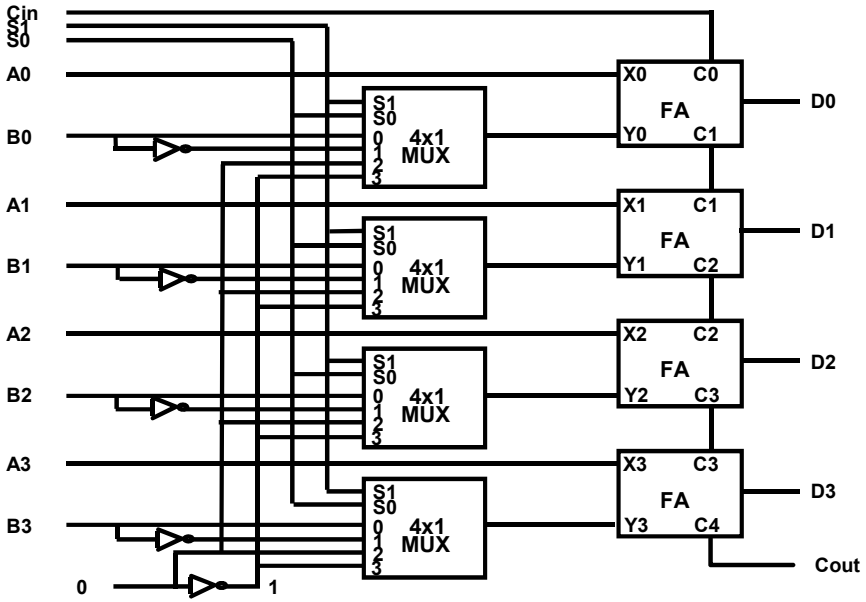


افزایشگر دودویی

Binary Incrementor
افزایشگر 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	انتقال A	Transfer A
1	0	1	0	D = A + 1	افزایش A	Increment A
1	1	0	1	D = A - 1	کاهش A	Decrement A
1	1	1	1	D = A	انتقال A	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 توان برای دستکاری بیتی (bit manipulations) داده به کار رود.
- به طور کلی ۱۶ عملیات متفاوت منطقی می تواند روی دو متغیر دودویی انجام شود.

جدول ۵-۴ جدولهای ارزش برای 16 تابع دو متغیره

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

ریز عمل های منطقی

جدول ۵-۴ جدولهای ارزش برای 16 تابع دو متغیره

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

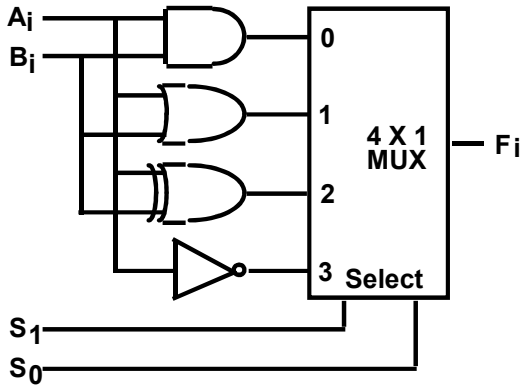
۱۶ عمل منطقی مختلف روی ۲ متغیر

با n متغیر می توان 2^{2^n} عمل منطقی تعریف کرد.

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

جدول ارزش برای
دو متغیر دودویی

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

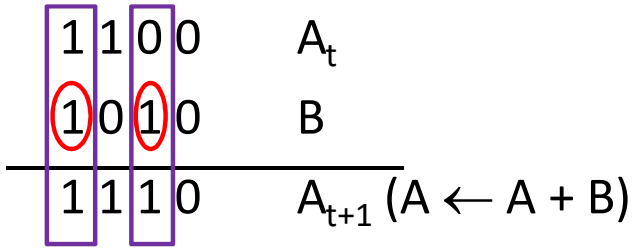
کاربردهای ریزعمل‌های منطقی

- 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	$A \leftarrow A + B$	یک کردن انتخابی
• Selective-complement	$A \leftarrow A \oplus B$	مکمل کردن انتخابی
• Selective-clear	$A \leftarrow A \bullet B'$	پاک کردن انتخابی
• Mask (Delete)	$A \leftarrow A \bullet B$	ماسک (حذف)
• Clear	$A \leftarrow A \oplus B$	پاک کردن
• Insert	$A \leftarrow (A \bullet B) + C$	درج کردن
• Compare	$A \leftarrow A \oplus B$	مقایسه
• . . .		

یک کردن انتخابی

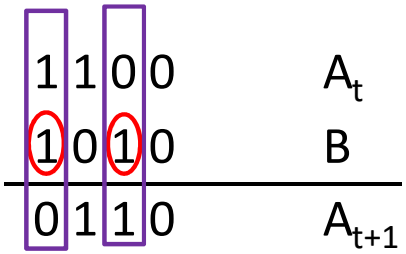
- در یک کردن انتخابی B برای تعیین بیت هایی از A که قرار است یک شوند مورد استفاده قرار می گیرد.



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



A_t

B

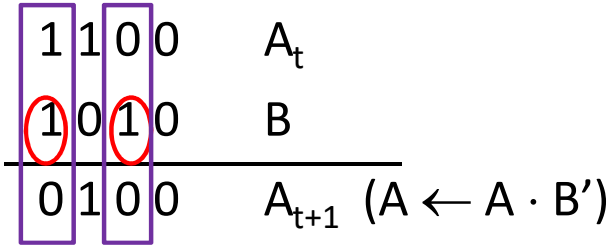
A_{t+1}

$(A \leftarrow A \oplus B)$

- 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

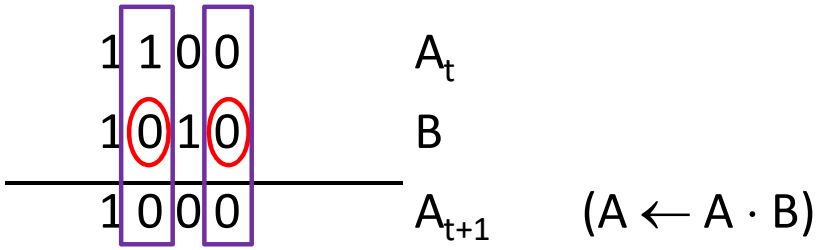
- در مکمل کردن انتخابی B برای تعیین بیت هایی از A که قرار است پاک(صفر) شوند مورد استفاده قرار می گیرد.



- 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.



- 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 پاک کردن (مقایسه)

- 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.

1 1 0 0	A_t
1 0 1 0	B
0 1 1 0	A_{t+1}

$(A \leftarrow A \oplus B)$

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.

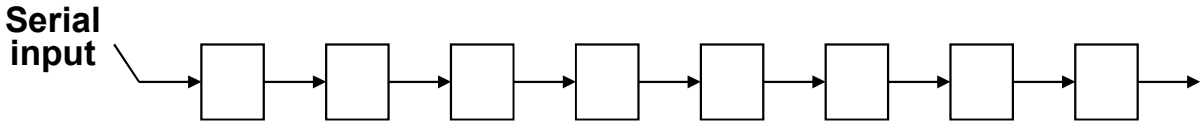
1101 1000 1011 0001	A (Original)
1101 1000 1011 1010	A (Desired)

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)

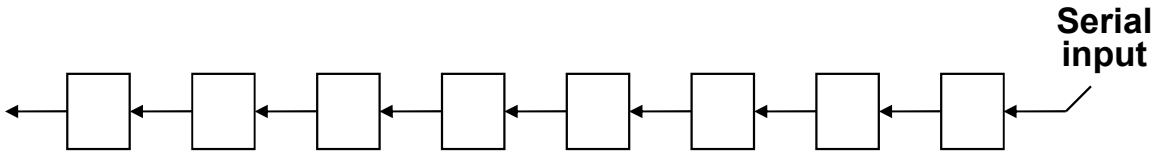
ریزعمل شیفت

- در سیستم های دیجیتال سه نوع شیفت مختلف وجود دارد:
  - ✓شیفت منطقی (logical shift)
  - ✓شیفت چرخشی (circular shift)
  - ✓شیفت حسابی (arithmetic shift)
- تفاوت این شیفت ها در بیت ورودی سریال است.

•عملیات شیفت به راست:

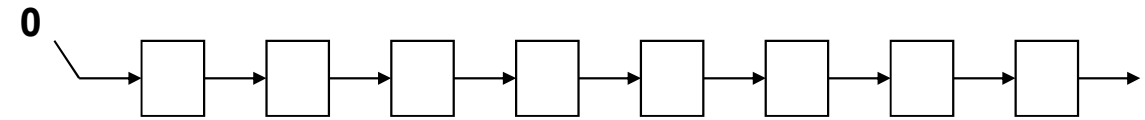


•عملیات شیفت به چپ:

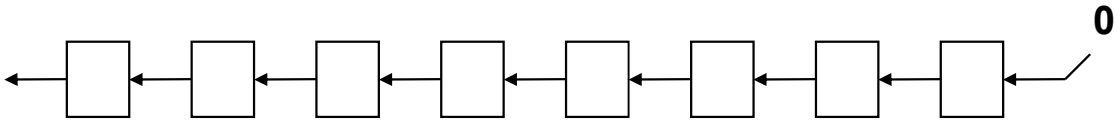


شیفت منطقی

- در شیفت منطقی بیت ورودی صفر است.
- شیفت منطقی به راست:



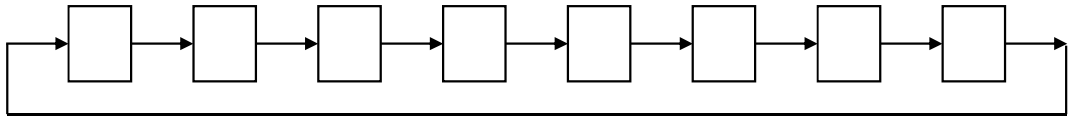
- شیفت منطقی به چپ:



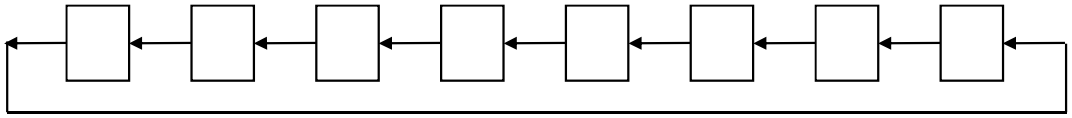
- در زبان انتقال ثبات از علائم زیر استفاده می شود:
  - *shl* شیفت منطقی به چپ
  - *shr* شیفت منطقی به راست
  - مثال:
    - R2 ← *shr* R2
    - R3 ← *shl* R3

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:

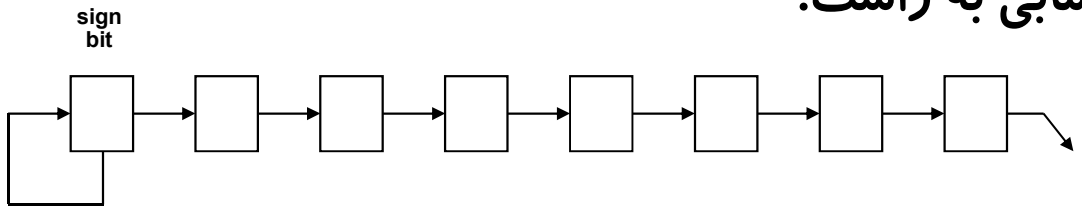
- *cil* شیفت چرخشی به چپ
- *cir* شیفت چرخشی به راست

- مثال:

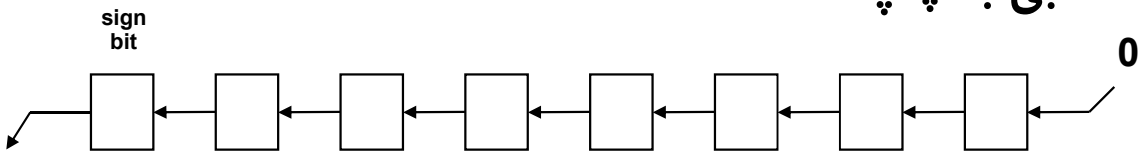
  - R2 ← *cir* R2
  - R3 ← *cil* R3

شیفت حسابی

- شیفت حسابی برای اعداد علامت دار معنی دار است.
- شیفت حسابی به چپ عدد درون ثبات را در ۲ ضرب می کند.
- شیفت حسابی به راست عدد درون ثبات را بر ۲ تقسیم می کند.
- مهمترین ویژگی شیفت حسابی آن است که به هنگام شیفت (ضرب و تقسیم) علامت ثبات را حفظ می کند.
- شیفت حسابی به راست:

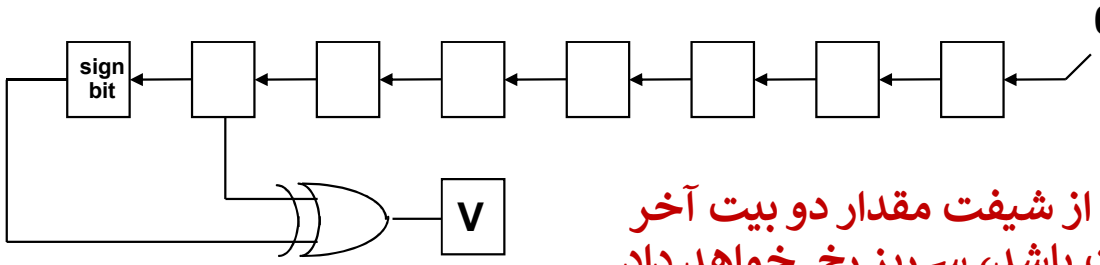


- شیفت حسابی به چپ:



شیفت حسابی

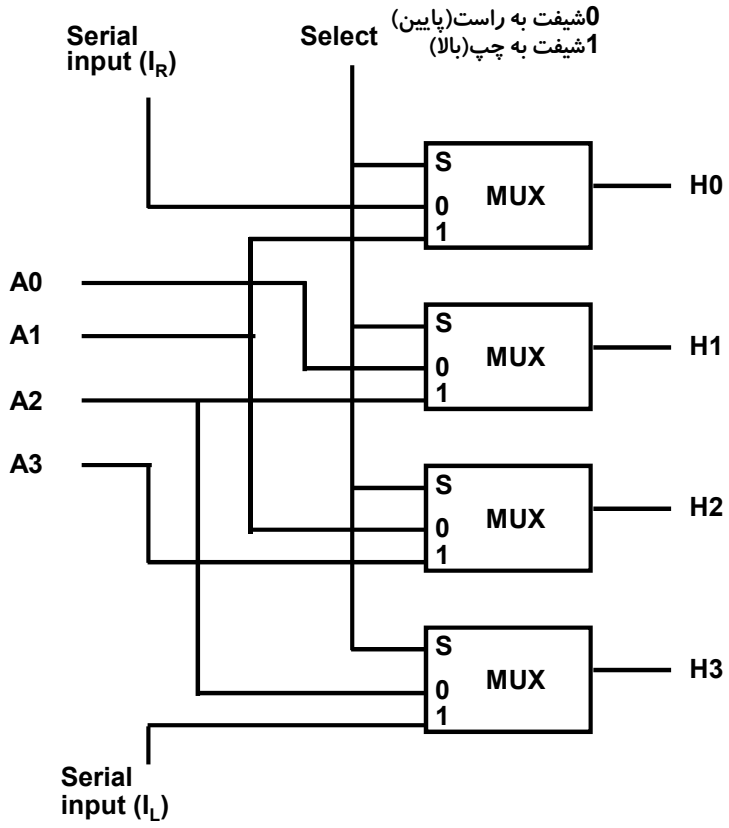
- در شیفت به چپ باید مساله سرریز(overflow) چک شود.



اگر قبل از شیفت مقدار دو بیت آخر
متفاوت باشد، سرریز رخ خواهد داد.

- 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

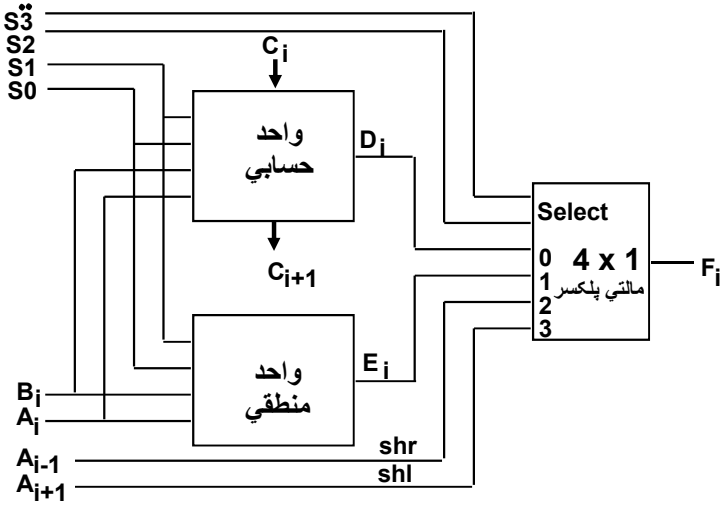
پیاده سازی سخت افزاری شیفت ها



Function table

Select	Output			
S	H_0	H_1	H_2	H_3
0	I_R	A_0	A_1	A_2
1	A_1	A_2	A_3	I_L

واحد عمليات شيفت، منطقي، حسابي



S3	S2	S1	S0	Cin	عمليات	توضيح
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 $\wedge$ B	AND
0	1	0	1	X	F = A $\vee$ B	OR
0	1	1	0	X	F = A $\oplus$ 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

8 ريز عمل حسابي

4 ريز عمل منطقي

2 ريز عمل شيفت

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

http://shafieian-education.ir