# برنامه سازی سیستم

دانشکده فنی و حرفه ای مائده

مدرس:
نگار مهدی نژاد

# این مباحث برای درس برنامه سازی سیستم در نظر گرفته شده است

دانشجویان عزیز

جلسه اول کلاس را به آشنایی با مقدمات مدارهای دیجیتال پرداختیم. مطالبی چون:

جبر بول – توابع منطقی – نمودار منطقی – گیتهای منطقی – جدول صحت

در جلسه اول بررسی شد.

در ادامه به بررسی چند مدار ترکیبی پرکاربرد و مقدماتی از مدارهای ترتیبی خواهیم پرداخت.

برای دریافت اطلاعات تکمیلی از این مباحث میتوانید از کتاب مدارهای منطقی موریس مانو ترجمه سپیدنام، فصل 4 و نیمی از فصل 5 استفاده کنید.

# فصل اول
# مرور مدارهای منطقی

مقدماتی برای آشنایی با برنامه سازی سیستمهای دیجیتال
با زبان توصیف سخت افزار VHDL

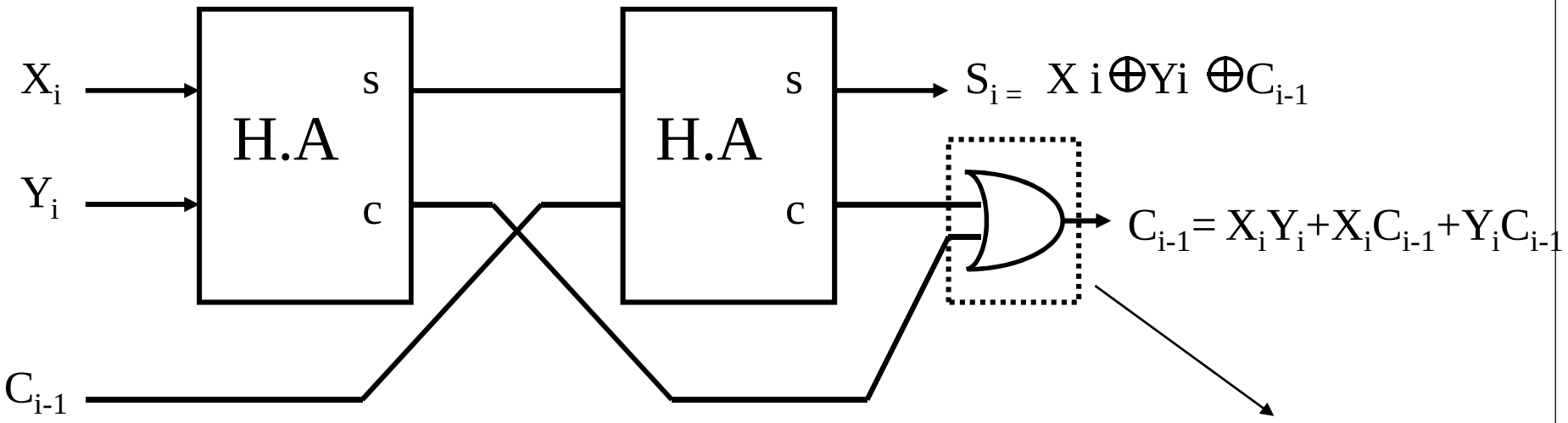
# مدارهای جمع کننده
# Full Adder و Half Adder

- Full Adder: یک مدار ترکیبی با سه ورودی و دو خروجی است که دو بیت داده ویک رقم نقلی را با هم جمع کرده و حاصل جمع ورقم نقلی را محاسبه می کند.

- Half Adder: یک مدار ترکیبی با دو ورودی و دو خروجی است که دو بیت دودویی را با هم جمع کرده و حاصل جمع ورقم نقلی را محاسبه می کند.

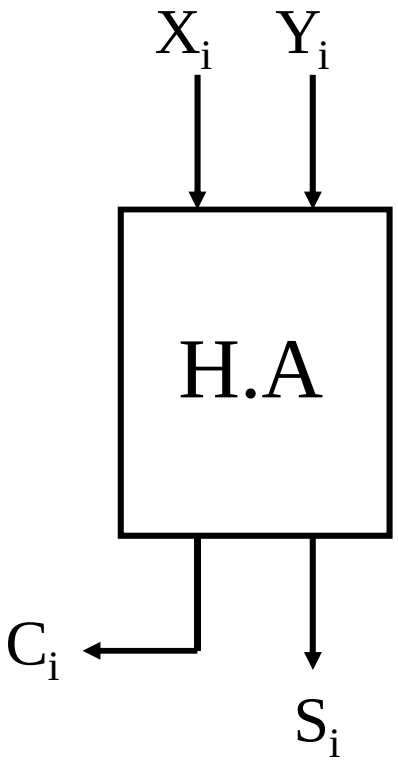
# Half Adder و Full Adder

یک **Full adder** را میتوان توسط 2 عدد **Hull adder** طراحی کرد.




می تواند توسط یک گیت **XOR** جایگزین شود.

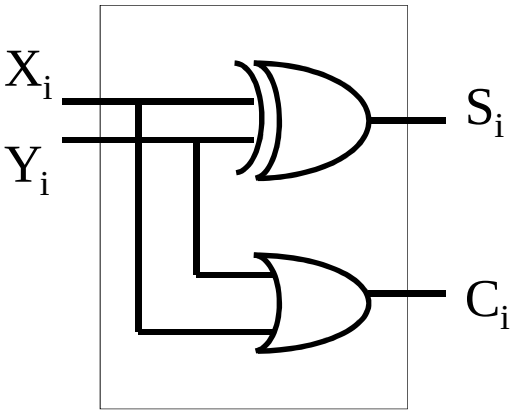
# بلوک دیاگرام (H.A)



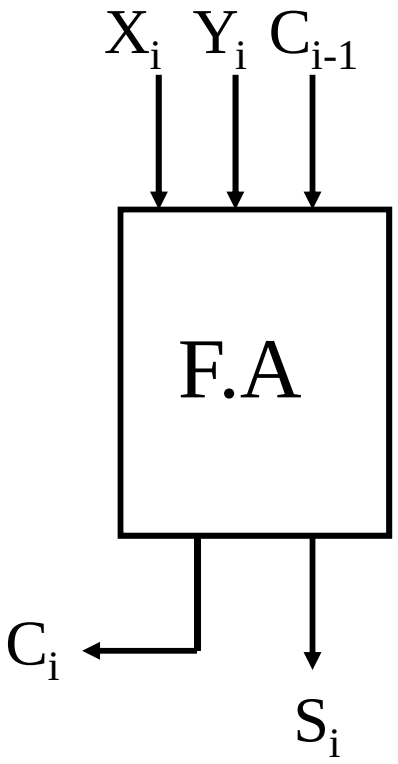

Truth Table

| $X_i$ | $Y_i$ | $C_i$ | $S_i$ |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 |

$$\begin{cases} S_i = X_i \oplus Y_i \\ C_i = X_i Y_i \end{cases}$$



# بلوک دیاگرام (F.A)



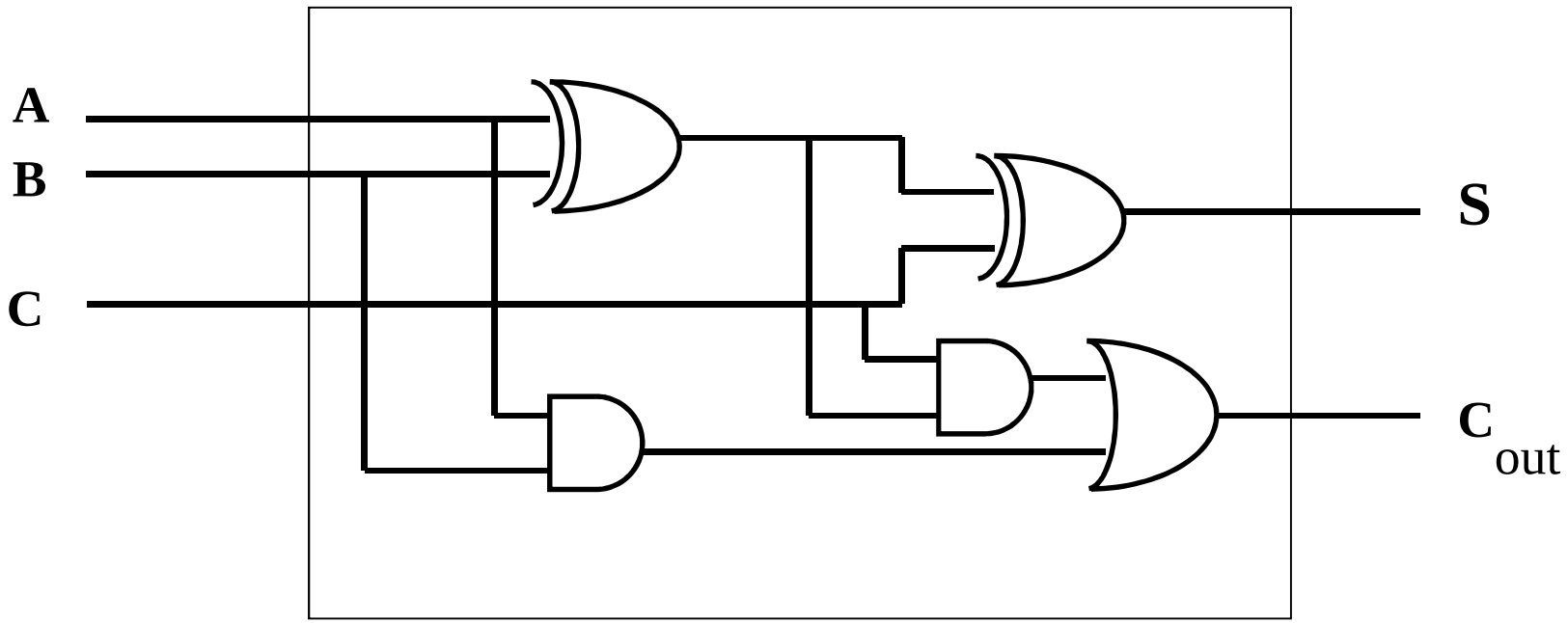

Truth Table

| $X_i$ | $Y_i$ | $C_{i-1}$ | $C_i$ | $S_i$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |

$$\begin{cases} S_i = X_i \oplus Y_i \oplus C_{i-1} \\ \mathbf{C_i} = X_i Y_i + X_i C_{i-1} + Y_i C_{i-1} \end{cases}$$

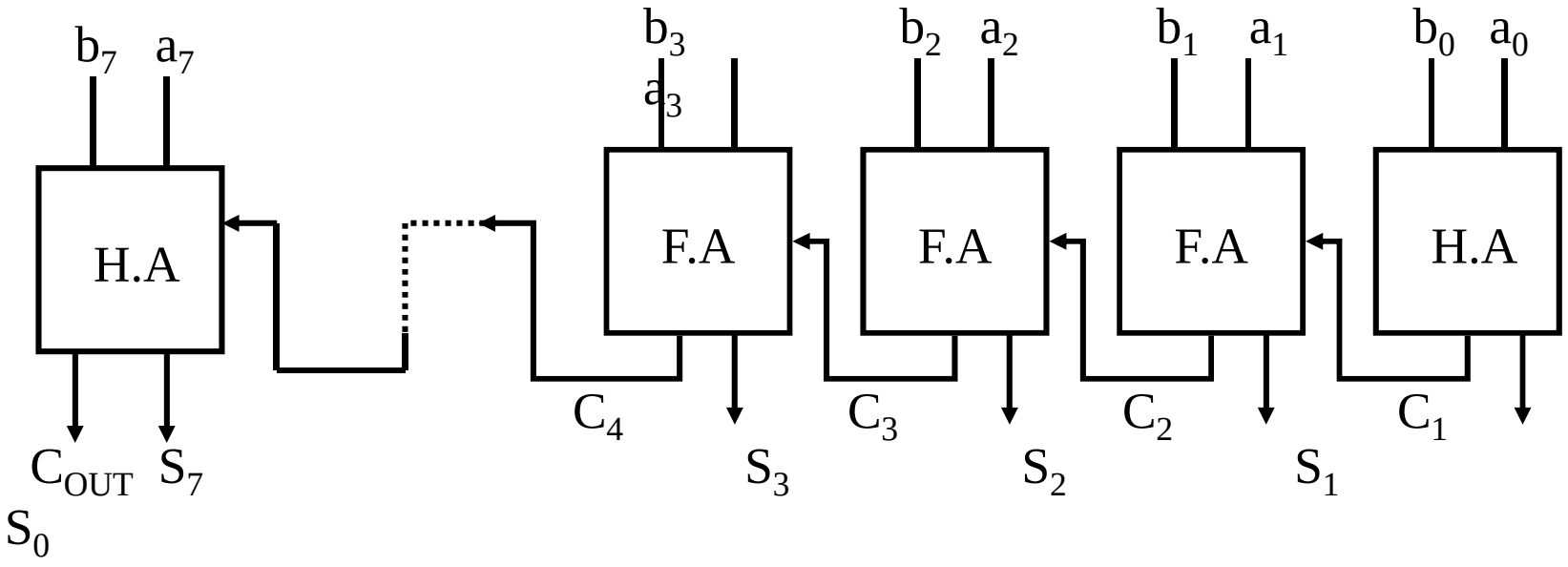
# دیاگرام منطقی (F.A)

$$C_{out} = C\,(A \oplus B) + AB$$



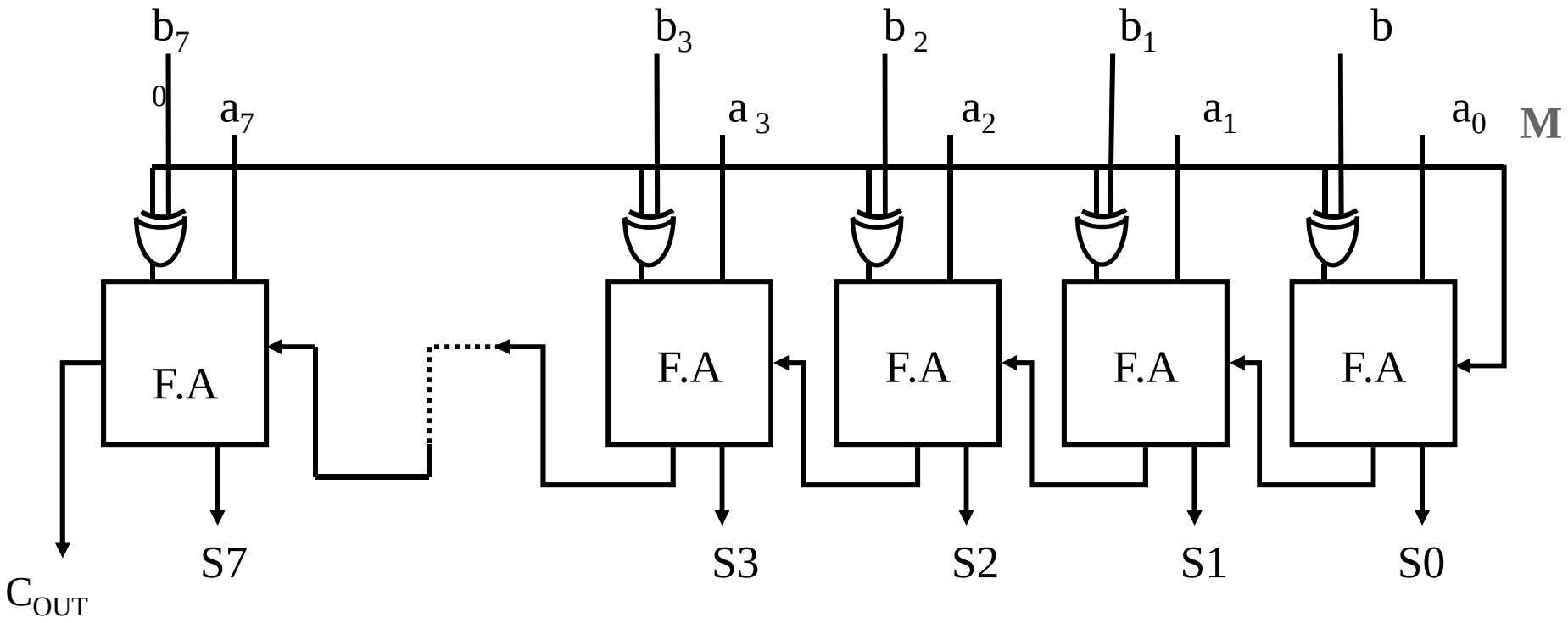
# Ripple Carry Adder (RCA)
# جمع کننده دودویی n بیتی

برای جمع دو مقدار nبیتی از تعداد n تمام جمع کننده که به صورت سریال به یکدیگر متصل شده اند می توان استفاده کرد. به شکلی که رقم نقلی خروجی هر طبقه به رقم نقلی ورودی طبقه بعدی وارد شود.



# مدار جمع کننده تفریق کننده n بیتی

دراین مدار در صورتی که پایه M صفر باشد عمل جمع باینری و در صورتی که یک باشد عمل تفریق باینری انجام میشود. از جمع کننده دودویی nبیتی و گیتهای xor برای ساخت این مدار استفاده شده است.




If M =0 ➡ A+B

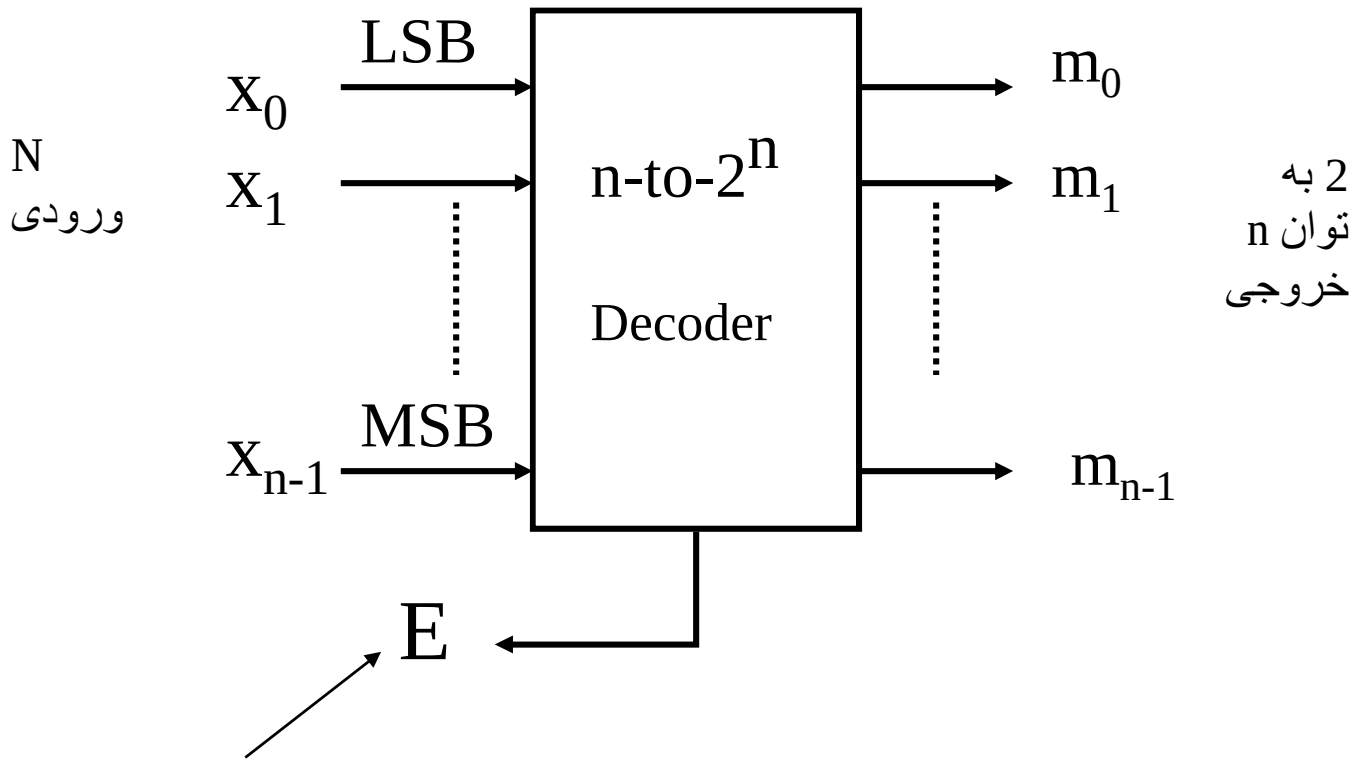
If M =1 ➡ A-B or (A+$\bar{B}$+1)

# دیکدر

- دیکدر n به $2^n$ یک شبکه منطقی ترکیبی است با n خط ورودی و $2^n$ سیگنال خروجی.

- عنصری است که مینترم ها را می سازد.
- در هر لحظه به ازای هر ترکیب باینری ورودیها تنها یکی از خروجی ها فعال است.
- خروجی فعال می تواند مقدار 0 یا 1 داشته باشند.
- دیکدر با خروجی فعال 0 را (A.l) active low و با خروجی فعال 1 را ( active high A.H) می نامند.
- دیکدرها اغلب یک یا چند پایه فعالساز enable دارند.

# بلاک دیگرام دیکدر n به $2^n$

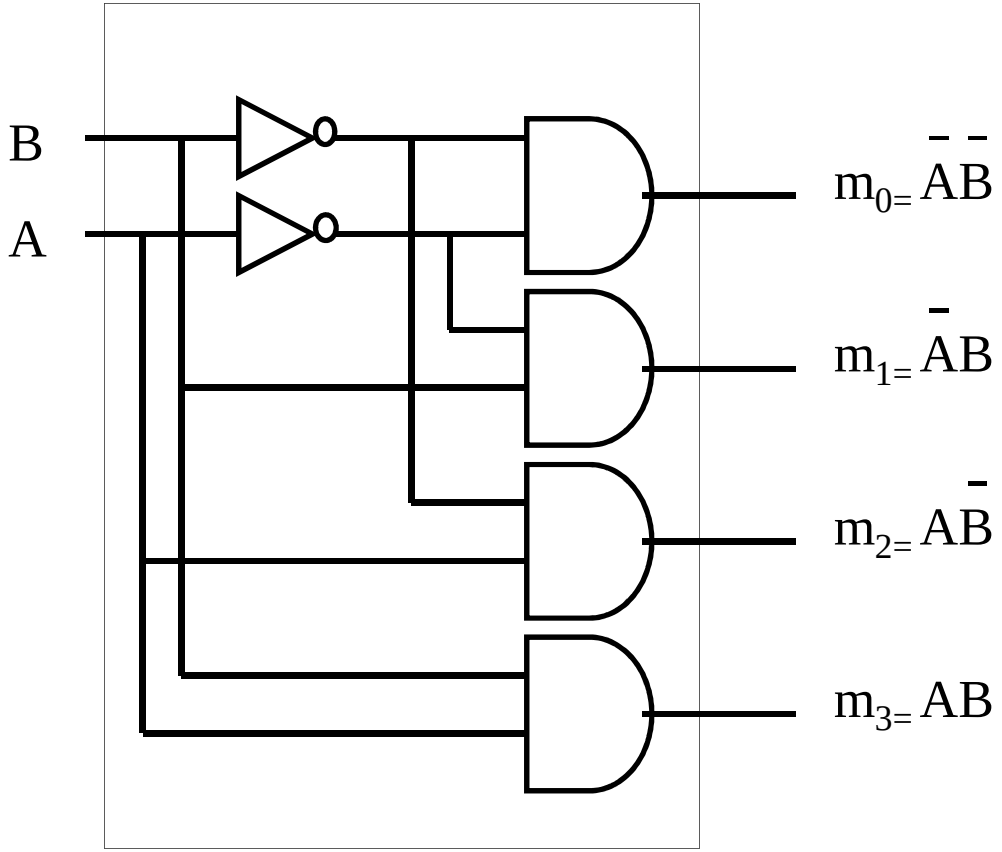



معمولا Active Low هستند.

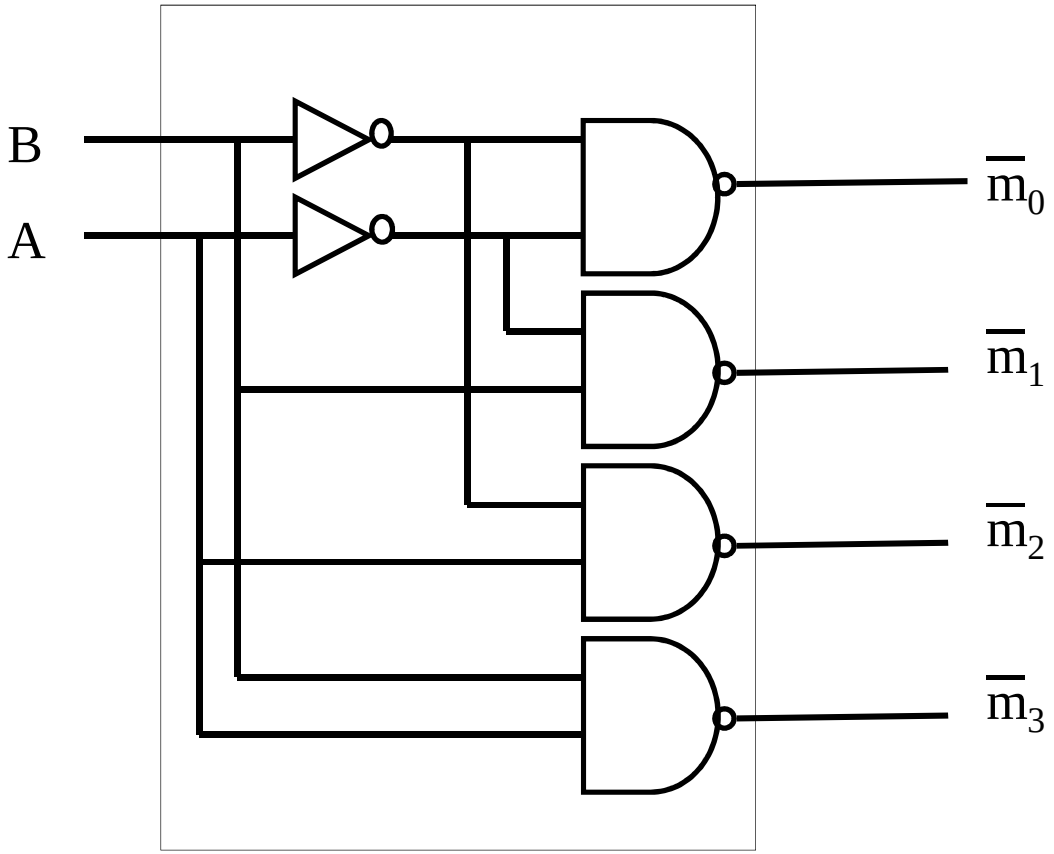
# دیاگرام منطقی (دیکدر 2 به 4 با پایه فعالساز فعال بالا a.h)

Truth Table

| E | A | B | $m_0$ | $m_1$ | $m_2$ | $m_3$ |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 |
| 0 | × | × | 0 | 0 | 0 | 0 |



# دیاگرام منطقی (دیکدر 2 به 4 با پایه فعالساز فعال پایین A.L )

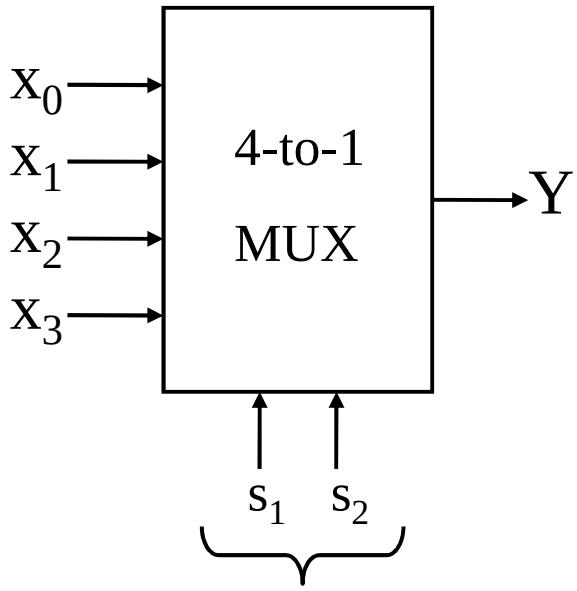


# مالتی پلکسر

بطور کلی مالتی پلکسر( انتخابگر داده ) یک ماجول است که یکی از چند خط ورودی را انتخاب و آن را روی خط خروجی ظاهر می سازد.

این مدار 2 به نمای **n** ورودی و یک خروجی دارد. **N** تعداد پایه های انتخاب است.

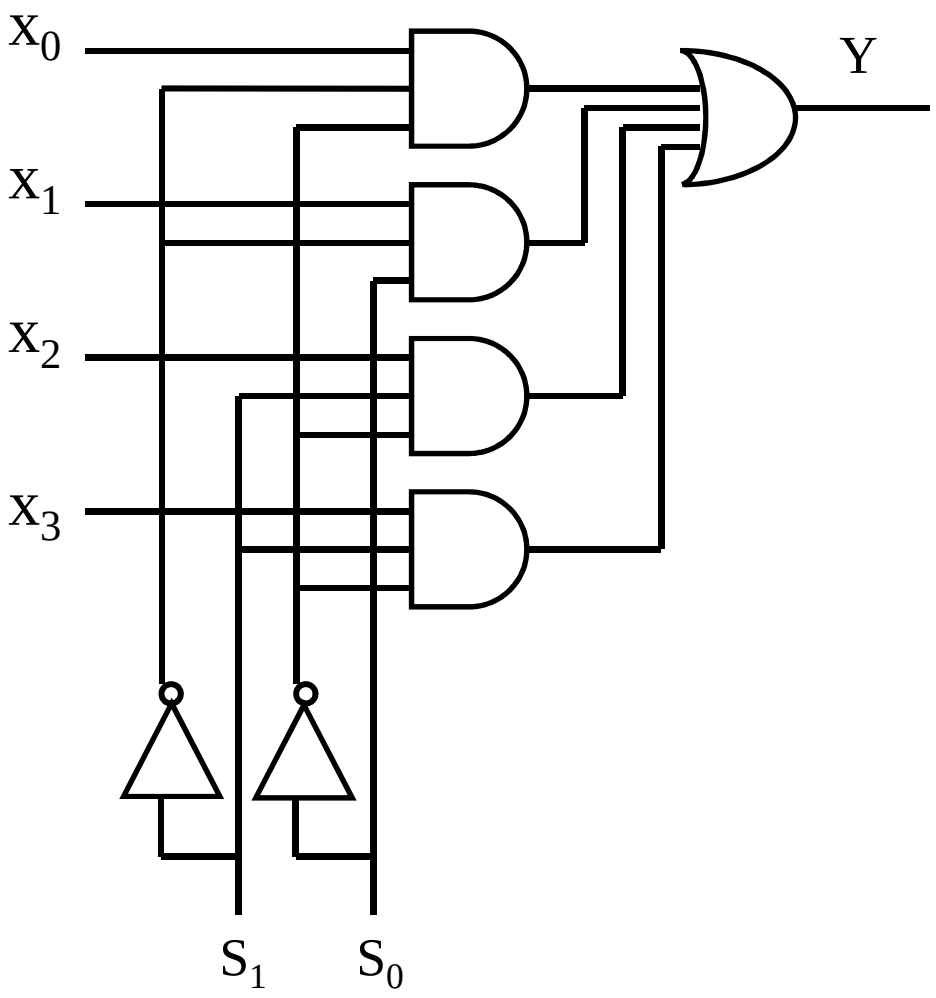
به ازای هر ترکیب باینری از ورودی های انتخاب، تنها یک ورودی داده انتخاب میشود و به خروجی میرود.




کد انتخاب

# مدار مالتی پلکسر 4 به 1

| $S_1$ | $S_0$ | Y |
|---|---|---|
| 0 | 0 | $x_0$ |
| 0 | 1 | $x_1$ |
| 1 | 0 | $x_2$ |
| 1 | 1 | $x_3$ |



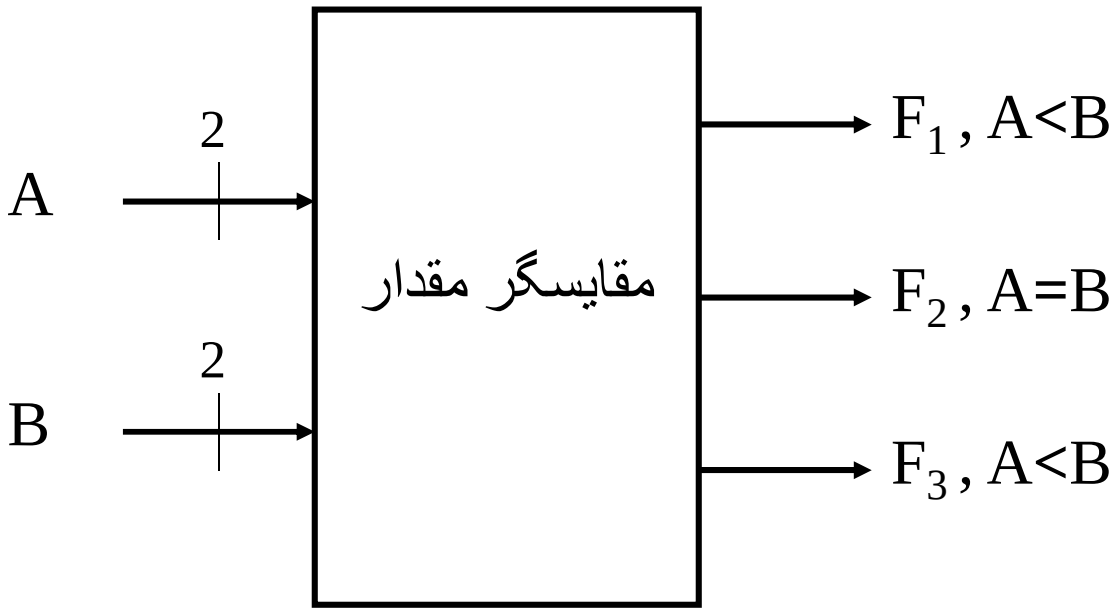
# مقایسه گرها

- مقایسه گر قطعه ای محاسباتی است که اندازه نسبی دو عدد دودویی را معیین می کند.
- در یک مقایسه گر سه تصمیم کاملا دیکد شده در مورد دو کلمه انجام و در خروخی ها قرار می گیرند. یعنی $A=B$ , $A>B$ , $A>B$ اگر

$$A=(A_{n-1}\ A_{n-2}\cdots A_0)$$

$$B=(B_{n-1}B_{n-2}\cdots B_0)$$

# دیاگرام عملیاتی




$$\begin{cases} F_1 = 1, \text{ If } A<B \\ F_2 = 1, \text{ If } A=B \\ F_3 = 1, \text{ If } A>B \end{cases}$$

# مثال :

مقایسه گری طراحی کنید که دو کلمه

$B=(B_1B_0)_2$ و $A=(A_1A_0)_2$ را

در کد دودویی مقایسه کند.

| $A_1$ | $A_2$ | $B_2$ | $B_2$ | $F_1$ | $F_2$ | $F_3$ |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 | 1 | 1 |

# نقشه های کارنو

$F_1$, A<B

| | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | | | | |
| 01 | 1 | | | |
| 11 | 1 | 1 | | 1 |
| 10 | 1 | 1 | | |

$F_2$ ,A=B

| | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 1 | | | |
| 01 | | 1 | | |
| 11 | | | 1 | |
| 10 | | | | 1 |

$F_3$, A>B

| | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | | 1 | 1 | 1 |
| 01 | | | 1 | 1 |
| 11 | | | | |
| 10 | | | 1 | |

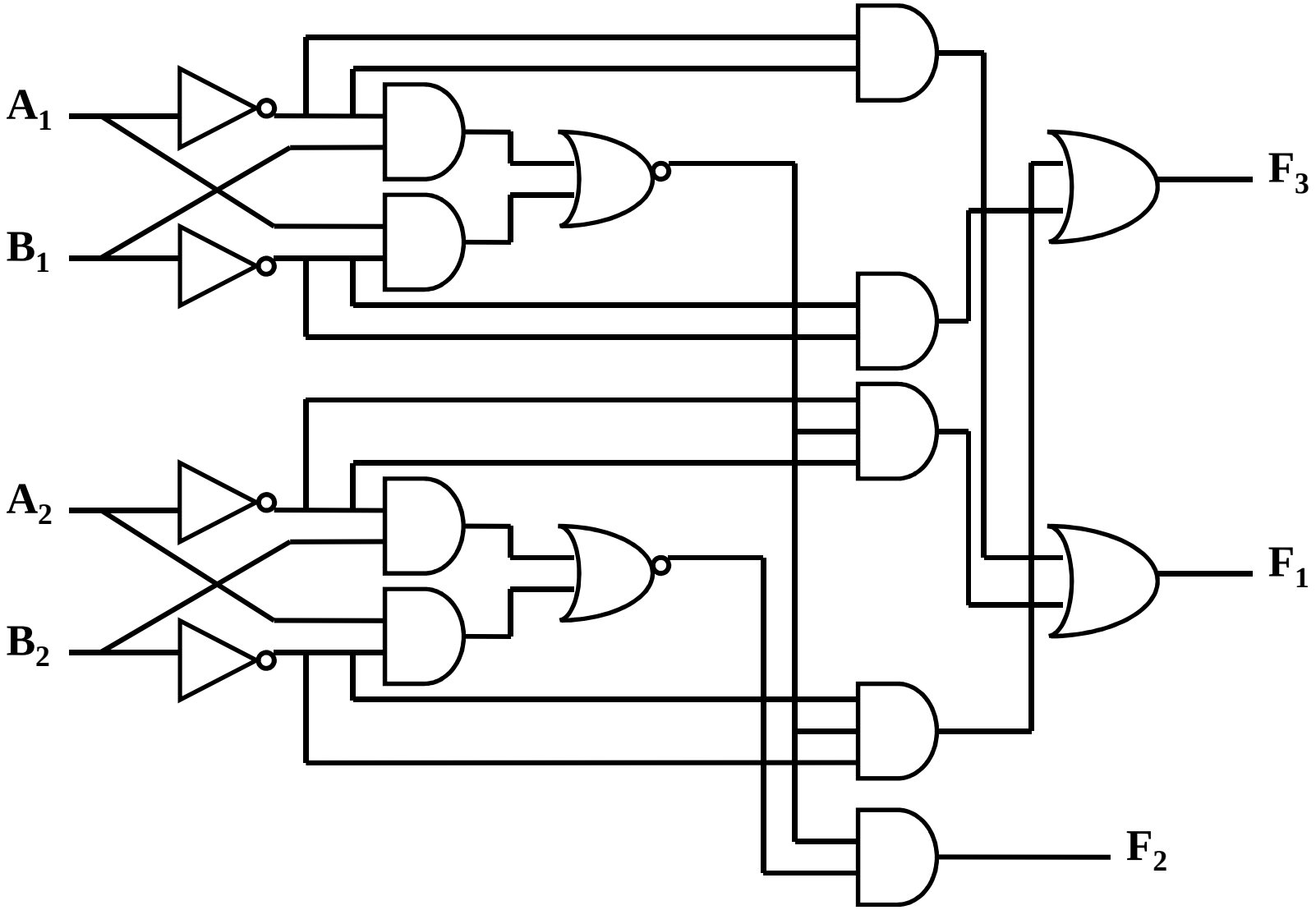
# توابع خروجی

$F_1 = \overline{A_1}\, B_1 + \overline{A_1}\,\overline{A_0}\, B_0 + \overline{A_0}\, B_1 B_0$ For $(A_1A_0)_2 < (B_1B_0)_2$

$F_2 = \overline{A_1}\,\overline{A_0}\,\overline{B_1}\,\overline{B_0} + \overline{A_1}\,A_0\,\overline{B_1}\,B_0 + A_1\overline{A_0}B_1\,\overline{B_0} + A_1A_0B_1B_0$ For $(A_1A_0)_2 = (B_1B_0)_2$

$F_3 = A_1\overline{B_1} + A_1\overline{B_1}\,\overline{B_0} + A_1A_0\overline{B_0}$ For $(A_1A_0)_2 > (B_1B_0)_2$

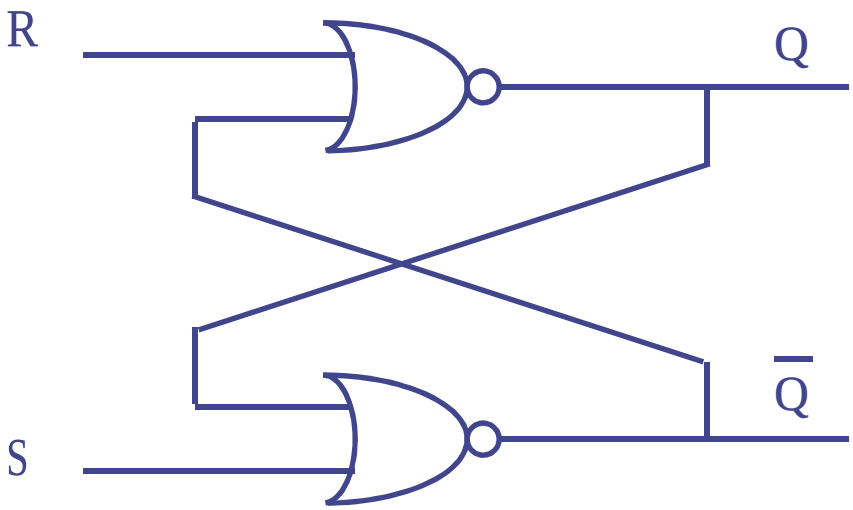
# تحقیق منطقی یک مقایسه گر دو بیت



# مدارات ترتیبی

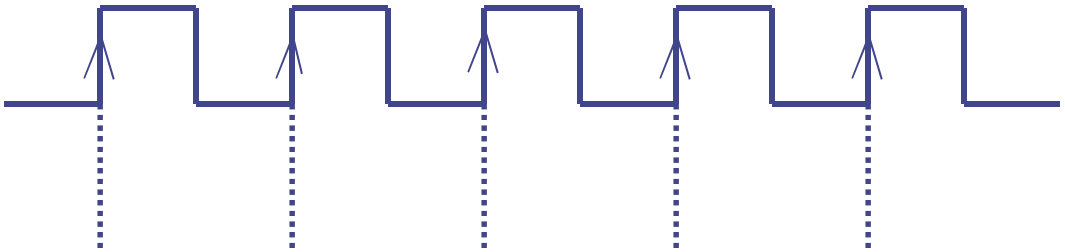
مدار ترتیبی مداری است که در آن خروجی در هر لحظه وابسته به ورودی همان لحظه و خروجی لحظه قبل است. همواره یک فیدبک از خروجی به ورودی وجود دارد.این مدارات حالت حافظه دارند.
رجیسترها، شمارنده ها، حافظه sram مدارات منطقی ترتیبی هستند.
پایه مدارهای ترتیبی لچ ها latch و فلیپ فلاپ ها flip flop هستند.

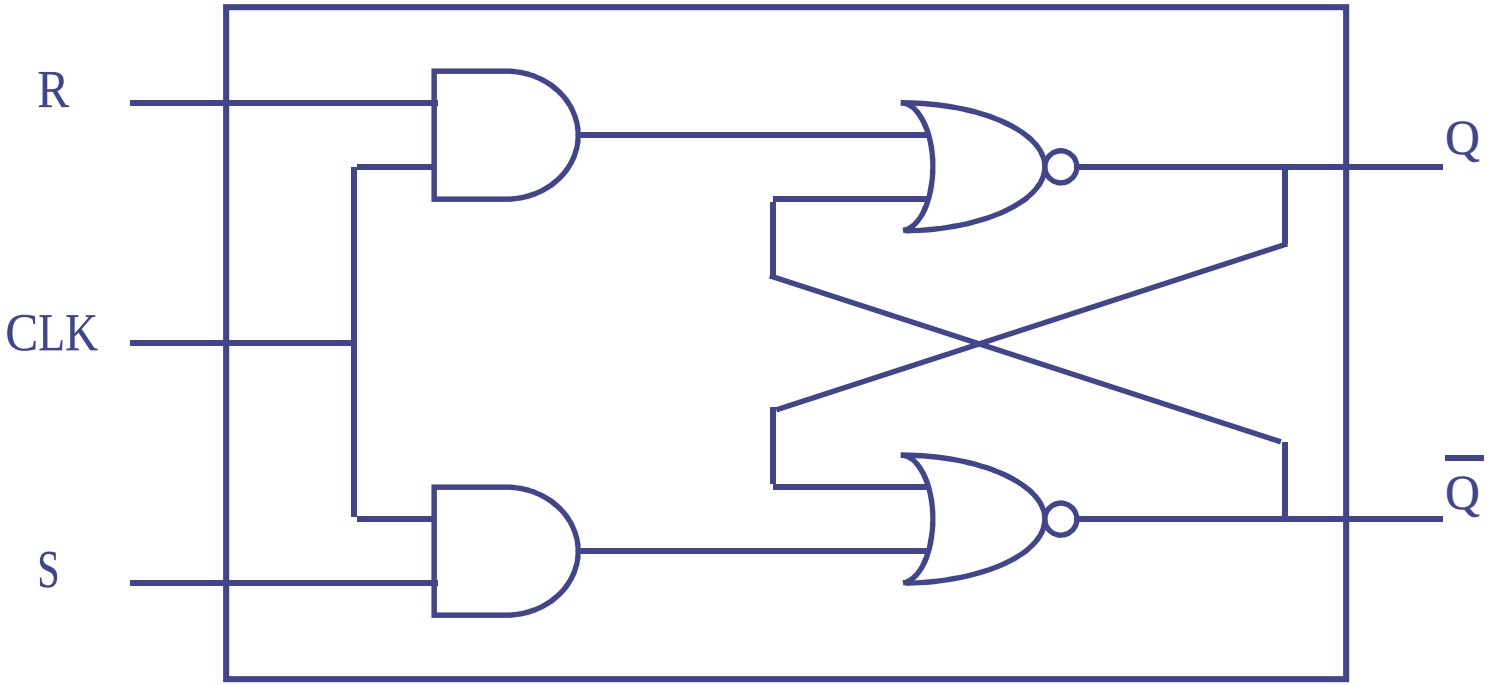
# مفهوم Latch :




| R | S | Q(t+1) | $\overline{Q}$(t+1) |
|---|---|---|---|
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 0 | 0 | Q(t) | $\overline{Q(t)}$ |
| 1 | 1 | نامعين | نامعين |

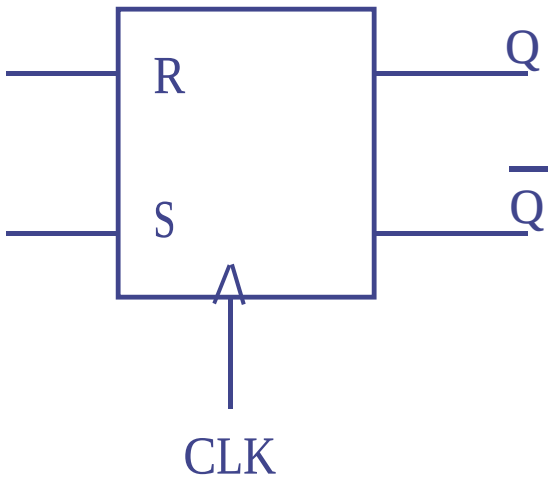
CLK : پالس های ساعت که باعث همگام سازی مدار می شود .





# انواع فلیپ فلاپ ها: RS, JK, T, D

## فلیپ فلاپ RS

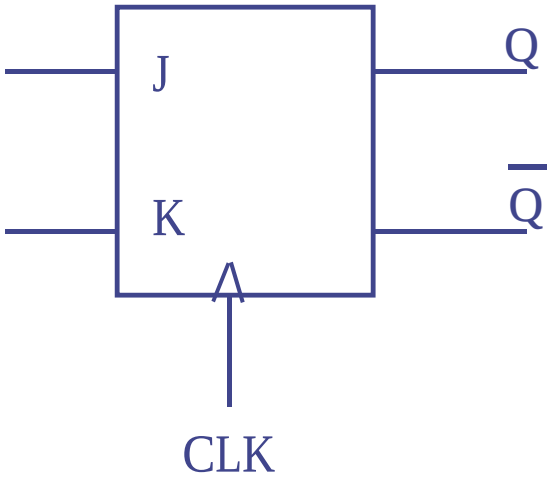



| S | R | Q(t+1) |
|---|---|---|
| 0 | 0 | Q(t) |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | نامعین |

( جدول مشخصه )

# انواع فلیپ فلاپ ها: (ادامه)

## فلیپ فلاپ JK



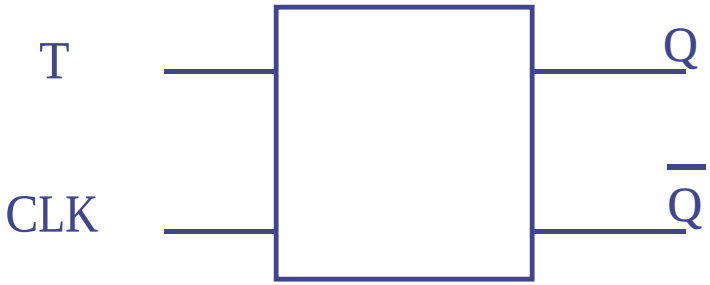

| J | K | Q(t+1) |
|---|---|---|
| 0 | 0 | Q(t) |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | $\overline{Q}$(t) |

T — D

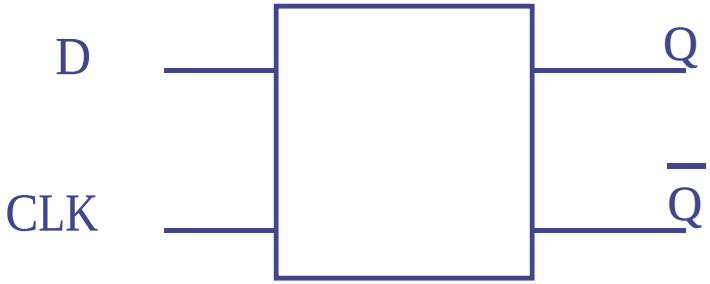
( جدول مشخصه )

# انواع فلیپ فلاپ ها: (ادامه)

## فلیپ فلاپ D ,T




| T | Q(t+1) |
|---|---|
| 0 | Q(t) |
| 1 | $\overline{Q}$(t) |

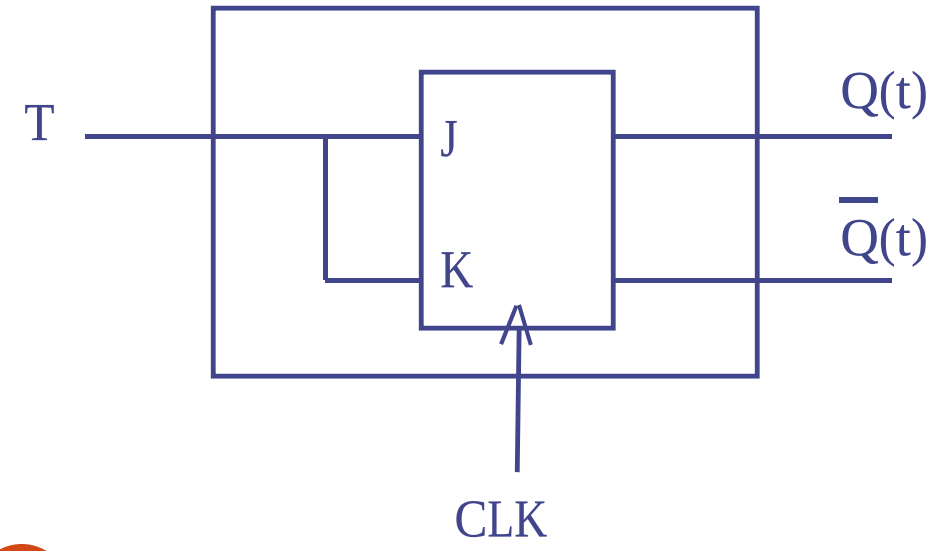



| D | Q(t+1) |
|---|---|
| 0 | 0 |
| 1 | 1 |

( جدول مشخصه )

مثال1: به کمک فلیپ فلاپ JK یک فلیپ فلاپ T بسازید.

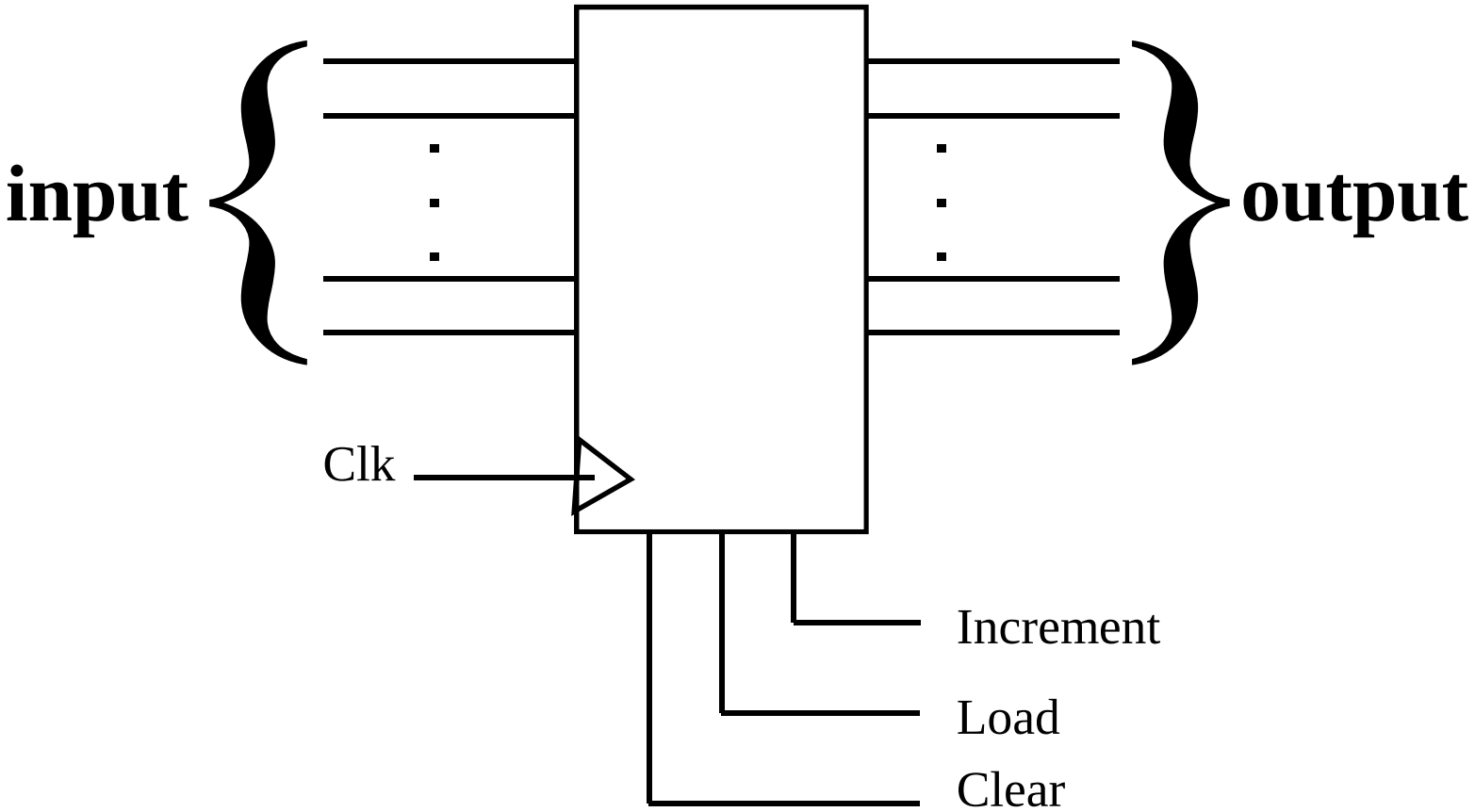
| T | J | K | Q(t+1) |
|---|---|---|---|
| 0 | 0 | 0 | Q(t) |
| 1 | 1 | 1 | $\overline{Q(t)}$ |



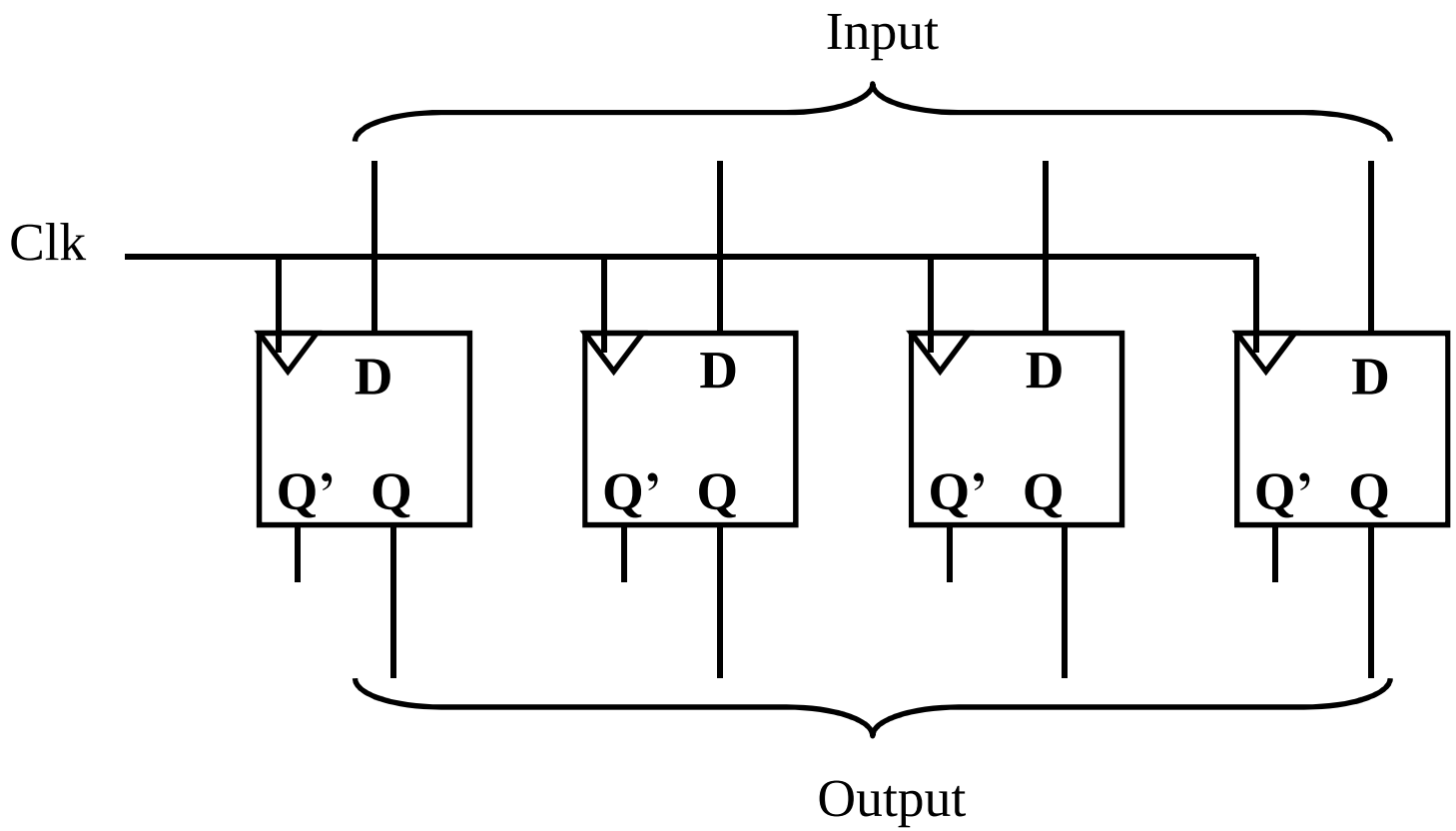
# ثبات ها و شیفت رجیستر

ثبات یا رجیستر:
یک حافظه کوچک و موقت است که از فیلپ فلاپها ساخته میشود. یک رجیستر n بیتی از n فلیپ فلاپ تشکیل شده است.
شیفت رجیستر:
رجیستری برای نگه داری و انتقال سری یا موازی دیتا ست.
برای ساخت رجیسترها معمولا از فلیپ فلاپ های D یا JK استفاده میشود.
بر اساس سری یا موازی بودن ورودی دیتا به رجیسترها 4 نوع شیفت رجیستر خواهیم داشت:
SISO – SIPO – PISO – PIPO
P : موازی  S : سریال  I : ورودی  O : خروجی

# طرح بلوک دیاگرامی ثبات و پایه های آن



# طرح ساده یک ثبات با فلیپ فلاپ D
# شیفت رجیستر 4 PIPOبیتی



# شمارنده ها

رجیسترهایی هستند که به ازای هر پالس ساعت یک مقدار باینری را میشمارند. انواع مختلفی چون توالی شمار صعودی یا نزولی، زوج شمار، فرد شمار و ... دارند.

برای ساخت شمارنده ها اغلب از فلیپ فلاپهای T و JK استفاده میشود.

مثلا یک شمارنده متوالی باینری 4بیتی میتواند با هر پالس ساعت یک مقدار 4بیتی از 0 تا 15 را بشمارد.

# به سوالات زیر پاسخ دهید و در اولین جلسه کلاس همراه داشته باشید:

1) بلاک دیاگرام، جدول صحت، تابع خروجی و نمودار منطقی یک دیکدر 3 به 8 با خروجی A.H را رسم کنید.
2) بلاک دیاگرام، جدول صحت، تابع خروجی و نمودار منطقی یک مالتی پلکسر 8 به 1 را رسم کنید.
3) با استفاده از تمام جمع کننده یک تفریق کننده باینری بسازید. سپس این مدار را به یک تفریق کننده باینری 4 بیتی گسترش دهید.
4) با استفاده از فیلپ فلاپ JK یک فلیپ فلاپ T بسازید.
5) با استفاده از فلیپ فلاپ های D یک شیفت رجیستر 4 بیتی SISO بسازید.
6) در مورد پایه پالس ساعت در مدارات ترتیبی تحقیق کنید.

# فصل دوم

## آشنایی با FPGA و VHDL

### FPGA چیست؟

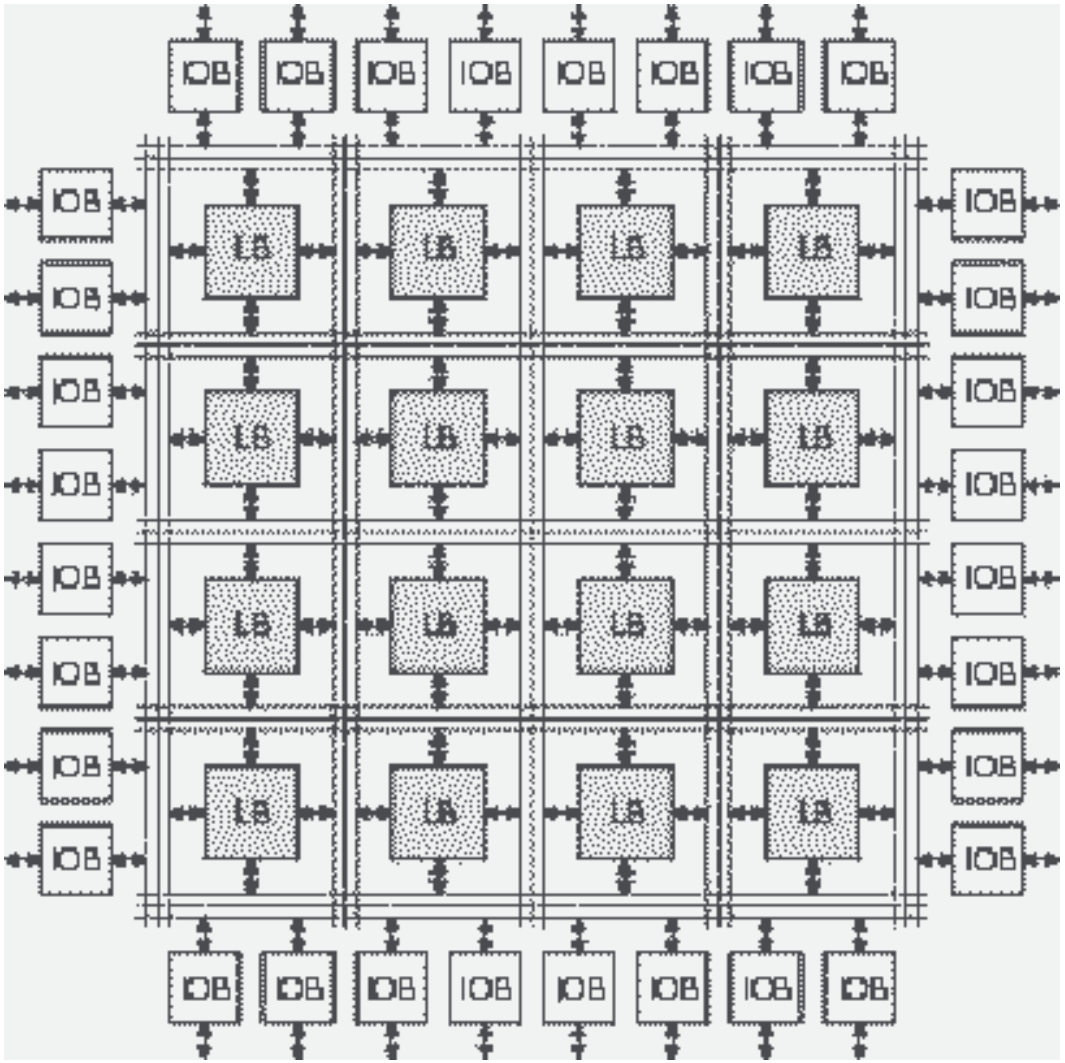
یک تراشه FPGA از منابع مختلف دیجیتالی تشکیل شده است که در ابتدا و به خودی خود، هیچ عملکرد مشخصی را ارائه نمی‌دهد. شما با پیکره‌بندی مناسب آن می‌توانید هر نوع سخت‌افزار دیجیتال را خلق کنید.

# FPGA

FPGAها نسل جدید مدارهای مجتمع دیجیتال قابل برنامه ریزی هستند که عبارت FPGAاز سر کلمه های Field Programmable Logic Gate Arrayگرفته شده است . سرعت اجرای توابع منطقی در FPGAها بسیار بالا و در حد نانو ثانیه است . اگر بخواهیم FPGAها را به طور ساده تشریح کنیم ، عبارت است از یک تراشه که از تعداد بالایی بلوک منطقی -( LB (Logic Block، خطوط ارتباطی و پایه های ورودی / خروجی ( (IOBتشکیل شده است که به صورت آرایه ای در کنار یکدیگر قرار دارند . خطوط ارتباطی که وظیفهء آنها ارتباط بین بلوک های منطقی است از سوئیچ های قابل برنامه ریزی تشکیل شده اند . این سوئیچ ها بسته به نوعی که دارند ، برخی تنها یکبار قابل برنامه ریزی هستند و برخی به تعداد دفعات زیادی برنامه ریزی می شوند . بلوک های منطقی نیز دارای انواع مختلفی هستند که عموما توسط المانی پایه ، تمامی توابع منطقی را ایجاد می کنند . به عنوان مثال بلوک های منطقی در خانواده ACT-1از شرکت Actel، با پایهء مالتی پلکسری عمل می کنند . به این معنا که توسط مالتی پلکسر ، توانایی ایجاد توابع منطقی مختلف را دارند . البته تعداد ورودی های هر بلوک منطقی متفاوت است و به نوع FPGAمربوط می شود . به عنوان مثال بلوک های منطقی در خانوادهء ACT-1، از نوع 8 ورودی است . البته در برخی موارد به بلوک های منطقی ، سلول های منطقی نیز گفته می شود ( . (LCبلوک دیاگرام یک FPGAبه طور ساده در شکل زیر نشان داده شده است . البته بسیاری از سلول های منطقی بر اساس جداول LUTساخته می شوند . LUTاز تعدادی سلولهای حافظه SRAMتشکیل می شود که در هنگام برنامه ریزی FPGA، مقدار دهی می شوند . به طور خلاصه LUTعبارت است از تولید توابع آماده برای استفاده در سلول های منطقی . پیاده سازی توابع مختلف نیز به وسیلهء در کنار هم قرار گرفتن بلوک های منطقی و همچنین تنظیم ارتباط بین هر بلوک و به عهده گرفتن پردازش اطلاعات توسط هر بخش انجام می شود.

FPGAاز یک سری عناصر منطقی که برای کار خاصیت محدود نشده اند و نیز دارای اتصالات قابل رنامه ریزی است. بنابراین هر دو جزء اصلی تشکیل دهنده یک مدار یعنی بلوکها منطقی و همچنین اتصالات بین آنها قابل برنامه ریزی است.

# FPGA



# FPGA

**شرکتهای سازنده :FPGA**

سازندگان FPGAشامل شرکت-های بزرگی همچون Xilinxو Alteraو Actel می-باشند که سری-های مختلفی، از جمله Xilinxشامل: Spartanو Virtexو Altraشامل: Cycloneو Flex10kو Stratixو .... را به بازار عرضه کرده-اند. در کشور ما معمولا با CPLDها و FPGAهای شرکت-های مذکور کار می-شود که از نظر محبوبیت به صورت زیر هستند.

Altera

Xiling

Actel

# FPGA

**انواع نرم افزارهای مربوط به :FPGA**

از سوی شرکتهای مختلفی نرم افزارهای با قابلیتهای مختص به خود طراحی کرده اند که اهم آنها به قرار زیرند:

MAXPLUS II – Quartus – Fandation - Leonardo Spectrum – Modelsim - …

# FPGA

تراشه های Field Programmable Gate Arrays یا همان FPGA برای توسعه سخت افزارهای دیجیتالی پیچیده و اجتناب از ساخت برد هایی با تراشه های گسسته به وجود آمده اند. با استفاده از تراشه های FPGA، بردهایی که قبلا با تعداد زیادی از تراشه های دیجیتال ساخته می شدند، در یک تراشه FPGA با سرعت بالاتر و از همه مهمتر با امکان به روز رسانی Update کردن سخت افزار، پیاده سازی می شوند. استفاده از FPGA این امکان را به ما می دهد تا طرح های سخت افزاری در ابعاد کوچکتر و با قیمت پایین تر ساخته شوند. عیب یابی طرح دیجیتالی که در داخل FPGA پیاده سازی شده به دلیل وجود نرم افزارهای شبیه سازی قدرتمند، بسیار ساده تر است. بسیاری از قطعاتی که در پروژه های دیجیتال مورد استفاده قرار می گیرند (مانند Dual port ram، DSP Module، Digital Synthesizer، FIFO و انواع کدر ها و دیکدر ها ، Ethernet Core، FFT Module و ...) به صورت Soft Core در داخل FPGA به راحتی قابل پیاده سازی بوده و در داخل کتابخانه استاندارد نرم افزار ISE وجود دارند. استفاده از Soft Core پردازنده هایی نظیر ARM در داخل FPGA، این تراشه را به ابزاری قدرتمند برای پیاده سازی انواع طرح های پردازش سیگنال (که نیاز به پردازش سریال دارند) تبدیل کرده است. امروزه بسیاری از شرکت های بزرگ سازنده تراشه ASIC، طرح های خود را بر روی FPGA تست و Verify می کنند زیرا امکان سنتز مدار تا سطح ترانزیستور در تراشه های FPGA وجود دارد.

# FPGA

با استفاده از زبان توصیف سخت افزار ( Verilogیا VHDL ) شما می توانید طرح و ایده سخت افزاری خود را بسته به نیاز و تبحر در سخت افزار، در یکی از سطوح توصیف رفتاری (توصیف سیستمی با ساختاری شبیه زبان C )، سطح Register Transfer Levelیا همان RTLبا استفاده از گیت ها و فیلیپ فلاپها و یا حتی در سطح ترانزیستور پیاده سازی نمایید و یا اینکه در محیط شماتیکی مدار مورد نظر خود را ترسیم و سنتز نمایید.

# VHDL چیست؟

VHDLیکی از زبانهای توصیف سخت افزار ( Hardware Description Language ) است که در طراحی مدارهای الکترونیکی مورد استفاده قرار می گیرد که اغلب برای توصیف مدار های یکپارچه ( IC دیجیتال یا ترکیبی آنالوگ و دیجیتال مورد استفاده قرار میگیرد . VHDL سرنام کلمات VHSIC Hardware Description Language است که VHSIC خود گرفته شده از سرنام کلمات Very High Speed Integrated Circuit می باشد.

VHDLیکی از پروژه هاییست که در ابتدا توسط سازمان دفاع آمریکا ایجاد گردیده و توسعه یافته است و بعدا به صورت تجاری درآمده است. VHDLاغلب برای طراحی بلاک های مداری بواسطه برنامه نویسی مورد استفاده قرار می گیرد و پس از تکمیل هر بلاک می توان آن بلاک را تست و شبیه سازی نمود و بلاک ها خود می توانند بخشی از بلاک های بزرگتر باشند و یا با سایر بلاک ها به صورت موازی کار کنند و بدین ترتیب توسعه و عیب یابی سیستم به سادگی قابل انجام است.

به صورت کلی بوسیله ی VHDLمی توان انواع مدار های منطقی ساده و پیچیده را به صورت کاملا سازمان یافته پیاده ساز نمود و این طراحی را بوسیله ی گیت های پایه به IC تبدیل نمود یا برنامه تولید شده را بر روی قطعاتی مانند FPGAها برنامه ریزی نمود.

# VHDL

از جمله شبیه ساز های VHDLمی توان به نرم افزار های زیر اشاره نمود :

- Active HDL محصول شرکت Aldec
- Cadence Incisive
- Mentor Graphics ModelSim
- Synopsys VCS-MX
- Xilinx Vivado
- Altera Quartus

# پاسخ سوالات زیر را در اولین جلسه کلاس همراه داشته باشید:

1) در مورد FPGA چه میدانید؟ شرح دهید.
2) در مورد VHDL چه میدانید؟ شرح دهید.

# موفق باشید

مهدی نژاد