# آزمایشگاه معماری کامپیوتر

نیمسال اول ۱۴۰۰

# قوانین آزمایشگاه

۱. تعداد جلسات آزمایشگاه ۱۲ جلسه می‌باشد که در هر جلسه یک سناریو مطرح شده و توصیف می‌گردد.

۲. حداکثر جلسات غیبت ۱ جلسه و تاخیر بیش از ۱۰ دقیقه مجاز نمی‌باشد.

۳. زمان تحویل آزمایش به صورت پیش‌فرض در ساعت آزمایشگاه است.

۴. ابزار مورد استفاده جهت توصیف و شبیه‌سازی سخت‌افزار:

مدلسیم MODELSIM
و
ISE Design Suite 14.x

۵. تعداد اعضای گروه ۲ نفره می‌باشد. پس از گروه‌بندی نام خود را اعلام فرمایید.(آزمایش اول به صورت تک نفره انجام می‌شود)

۶. گزارش‌کار نویسی و تحویل به موقع. تا حد ممکن سعی شود اصول لازم برای گزارش مهندسی رعایت شود (به بهترین گزارش نمره تشویقی تعلق میگیرد.)
مطابق قوانین دانشگاه هرگونه کپی برداری ممنوع می‌باشد و در صورت مشاهده، نمره‌ی هر دو طرف صفر در نظر گرفته می‌شود.

# نحوه نمره‌دهی

| | |
|---|---|
| پیش گزارش و گزارش آزمایش | ۱۵٪ |
| انجام کامل آزمایش و تحویل در اسکایپ | ۴۵٪ |
| آپلود و تحویل به موقع | ۱۵٪ |
| کیفیت انجام آزمایش و پیاده‌سازی | ۱۵٪ |
| حضور فعال در کلاس | ۱۰٪ |

دانشکده مهندسی کامپیوتر

آزمایشگاه معماری کامپیوتر

# نحوه ارسال پروژه‌ها

فایل پروژه به صورت زیپ شده و با فرمت نام‌گذاری زیر در سامانه کورسز بارگزاری شود.

نام‌خانوادگی۱ شماره دانشجویی- نام‌خانوادگی۲ شماره دانشجویی- شماره آزمایش

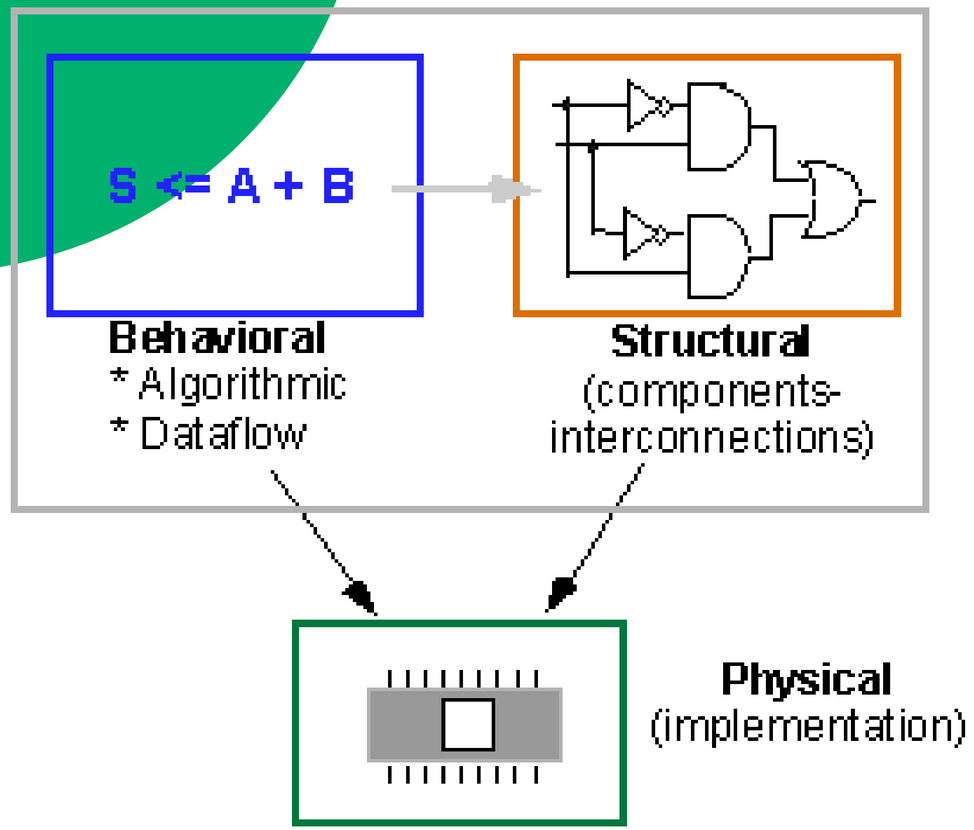
1-XX-XX

# VHDL چیست؟

VHDL مخفف Very High Speed Integrated Circuits (VHSIC) Hardware Description Language، در دهه‌ی ۸۰ میلادی در وزارت دفاع آمریکا و با همکاری IEEE به منظور توسعه مدارهای مجتمع با سرعت بالا، طراحی شد.

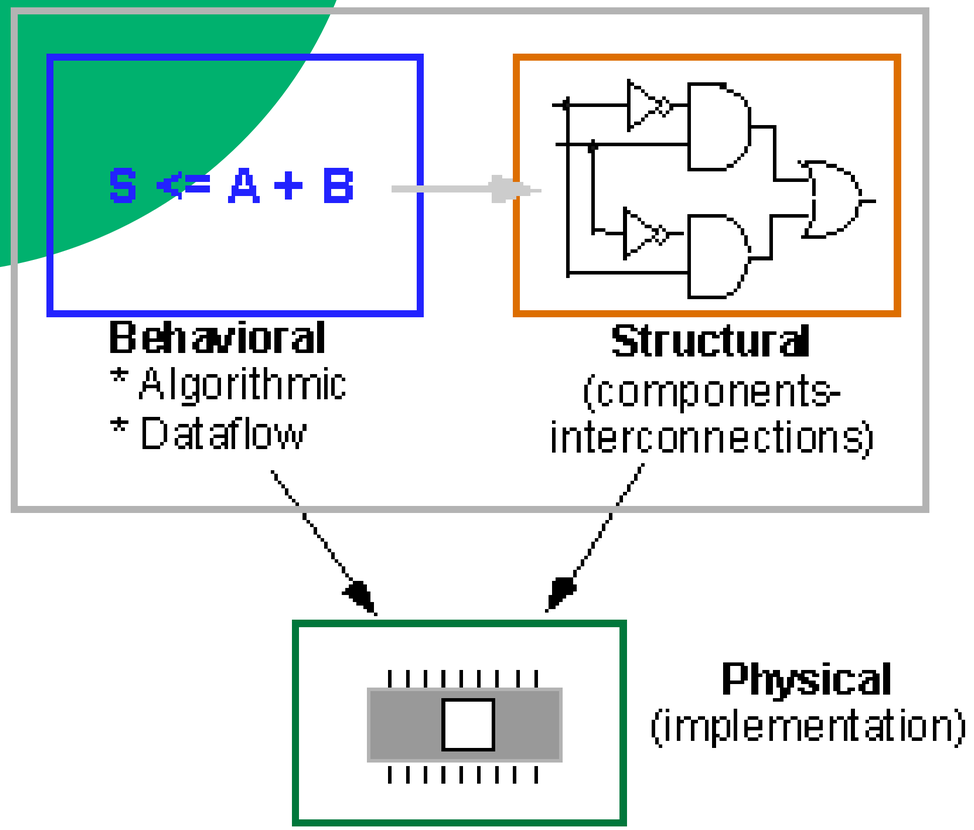
امروزه از این زبان به عنوان یک زبان دارای استاندارد صنعتی برای توصیف سیستم‌های دیجیتال استفاده می‌شود.

# سطوح انتزاع



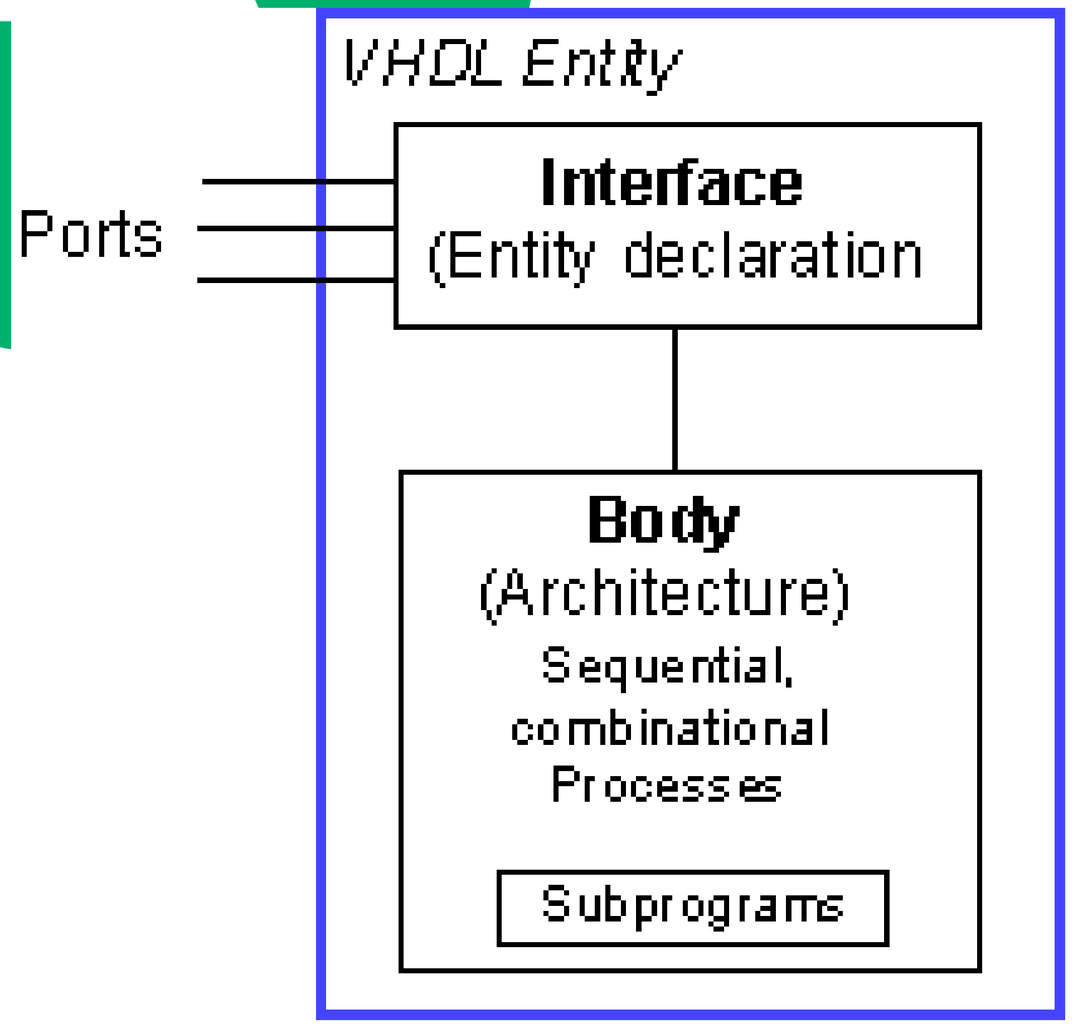

- استفاده از سطوح انتزاع باعث می‌شود تا توصیف سیستم‌های پیچیده قابل انجام شود.
- بالاترین سطح انتزاع، سطح رفتاری (Behavioral) است که سیستم را درقالب این‌که چگونه رفتار می‌کند توصیف می‌کند و به اجزا و ارتباطات میان آن‌ها اهمیتی نمی‌دهد.
- سطح ساختاری (Structural)، سیستم را در قالب مجموعه‌ای از اجزا، گیت‌ها و ارتباطات میانشان تعریف می‌کند، توصیف ساختاری را با توصیف شماتیک و ارتباطات میان گیت‌ها مقایسه می‌کنند.

# سطوح انتزاع




- VHDL این امکان را فراهم می‌سازد تا سیستم دیجیتال در سطح ساختار یا رفتاری تعریف شود.
- سطح رفتاری را می‌توان به دو مدل جریان داده و الگوریتمی تعریف کرد.
- جریان داده، قابلیت تعریف همزمانی و اجرا موازی را فراهم می‌سازد.
- الگوریتمی، قابلیت تعریف عبارات ترتیبی و اجرا ترتیبی را فراهم می‌کند.

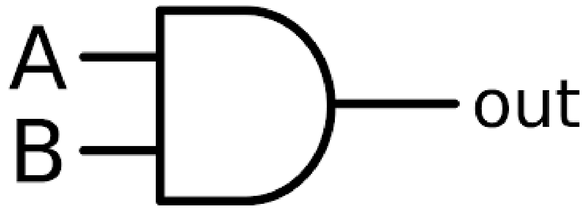
# توصیف VHDL




- یک سیستم دیجیتال در VHDL از Entity تشکیل می‌شود که می‌تواند شامل سایر Entityها باشد. هر Entity توسط یک توصیف Entity و بدنه Architecture تعریف می‌شود.
- می‌توان گفت Entity واسط دنیای خارجی است که در آن سیگنال‌های ورودی/خروجی تعریف می‌شود.
- در بدنه Architecture توصیف Entity صورت می‌گیرد و ترکیبی از entityهای متصل، پروسس‌ها و اجزا است که همگی به شکل همزمان با یکدیگر کار می‌کنند.

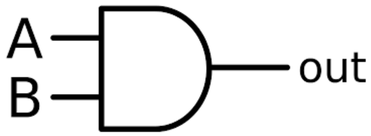
## مفهوم ENTITY

تعریف قطعه
نام قطعه
بیان سیگنالهای ورودی و خروجی و نام هریک
انجام تعریف با کلمه کلیدی Entity



# مفهوم ENTITY

```
entity and_gate is
  port (
    A     : in  std_logic;
    B     : in  std_logic;
    and_result : out std_logic
    );
end and_gate;
```

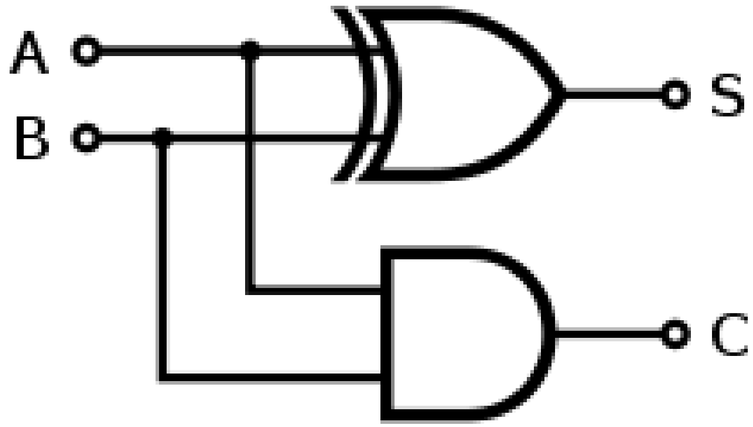


## نحوه کلی تعریف Entity

```
ENTITY entity_name IS
    PORT (
        port_name : signal_mode signal_type;
        port_name : signal_mode signal_type;
        ...);
END entity_name;
```

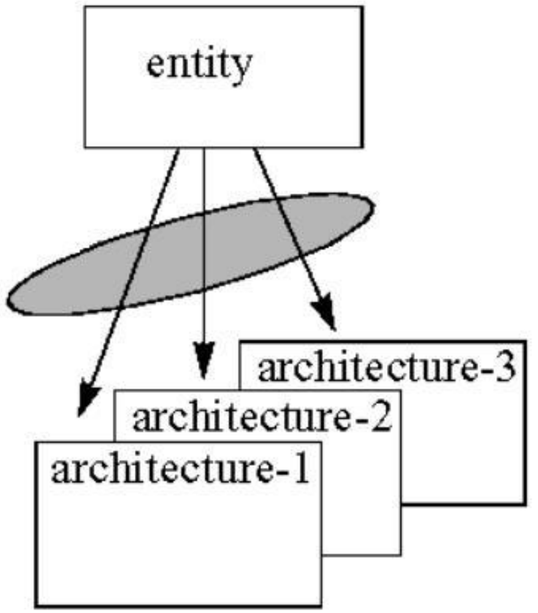
## مثال Entity

```
entity HA is
  port (
    A,B    : in  std_logic;
    S,C: out std_logic
    );
end HA;
```



# توصیف قطعه

- توصیف قطعه در داخل محیطی بنام Architecture انجام می شود.
- هر قطعه یک تعریف دارد اما می تواند چندین توصیف داشته باشد.

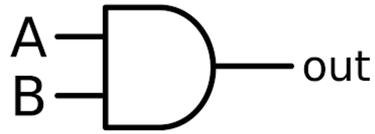


## نحوه انجام توصیف قطعه

```
Architecture  beh of part_name is
Begin
    محل نوشتن کد توصیف قطعه
End beh;
```

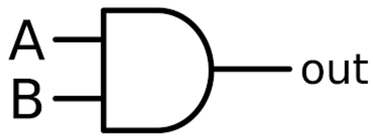
# توصیف گیت AND

```
architecture beh of and_gate is
begin
  and_result <= A and B;
end beh;
```



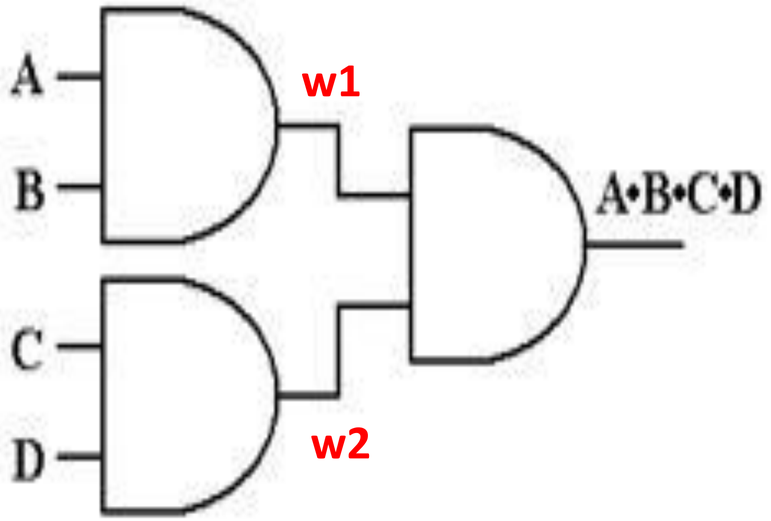
# گیت AND

```
library ieee;
use ieee.std_logic_1164.all;

entity and_gate is
  port (
     A    : in  std_logic;
     B    : in  std_logic;
     and_result : out std_logic
    );
end and_gate;

architecture beh of and_gate is
begin
  and_result <= A and B;
end beh;
```



# ۴ AND ورودی با استفاده از گیت AND دو ورودی

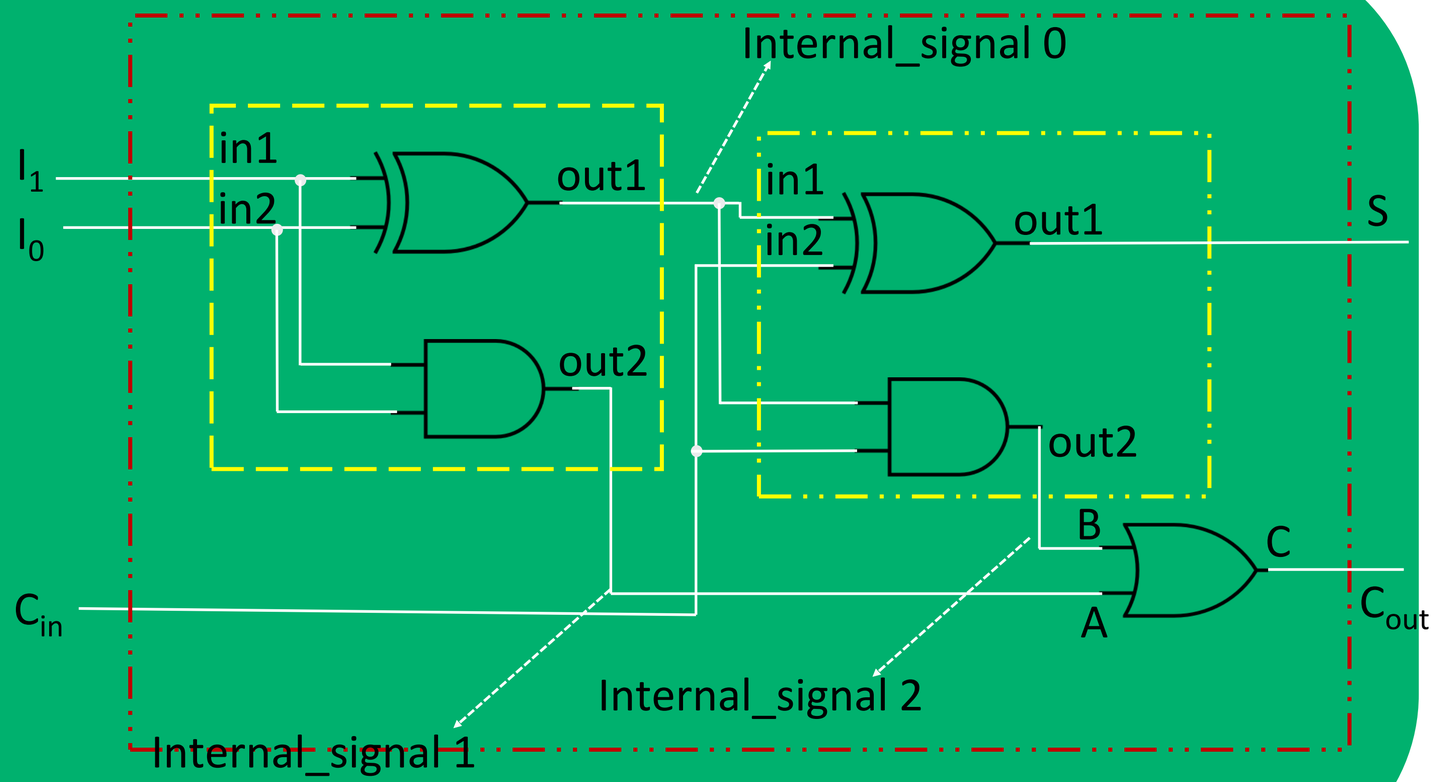
```
library ieee;
use ieee.std_logic_1164.all;

entity and_4_input is
  port (
    A    : in  std_logic;
    B    : in  std_logic;
    C    : in  std_logic;
    D    : in  std_logic;
    and4_result : out std_logic
    );
end ;

architecture structure of and_4_input is
 component and_gate is
  port (
    A    : in  std_logic;
    B    : in  std_logic;
    and2_result : out std_logic
    );
 end component;
  signal w1,w2:  std_logic;

begin
  and_gate_instance0: and_gate port map(A => A, B => B, and2_result => w1);
  and_gate_instance1: and_gate port map(A => C, B => D, and2_result => w2);
  and_gate_instance2: and_gate port map(A => w1, B => w2, and4_result =>
and_result);

end structure;
```



# قواعد کلی VHDL

- VHDL دارای Keywordهایی است که نمی‌توان آن‌ها را به عنوان نام سیگنال و شناسه تعریف کرد.
- کلمات کلیدی و شناسه‌های تعریف شده توسط کاربر Case insensitive هستند.
- استفاده از (--) باعث می‌شود تا نوشته‌های بعد از آن توسط کامپایلر نادیده گرفته شوند.
- VHDL همیشه روی نوع اشیاء حساس است و نیاز دارد که نوع تمامی اشیاء توسط کاربر تعریف شود.

# آزمایش اول

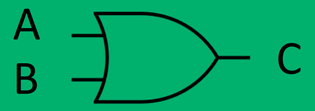

Internal_signal 0
in1
in2
out1
out2
$I_1$
$I_0$
in1
in2
out1
out2
S
B
C
A
$C_{out}$
$C_{in}$
Internal_signal 2
Internal_signal 1
Full adder
half adder

# آزمایش اول

```
Library IEEE;
USE IEEE.std_logic_1164;
Entity and_gate is
Port (
            A, B: in std_logic;
            C    : out std_logic
);
End Entity and_gate;

Architecture gatelevel of and_gate is
Begin
C <= A and B;
End gatelevel;
```

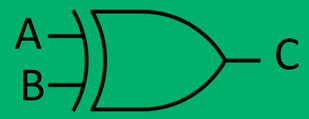
# آزمایش اول

```
Library IEEE;
USE IEEE.std_logic_1164;
Entity or_gate is
Port (
            A, B: in std_logic;
            C    : out std_logic
);
End Entity or_gate;

Architecture gatelevel of or_gate is
Begin
C <= A or B;
End gatelevel;
```

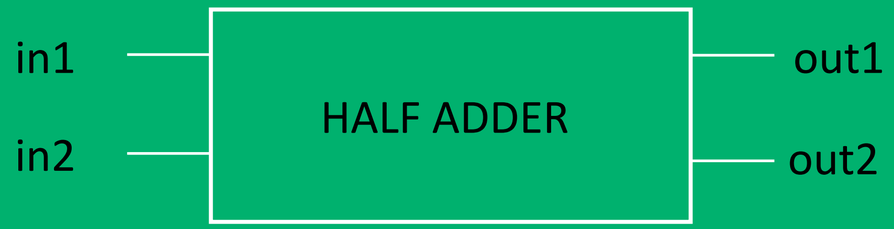


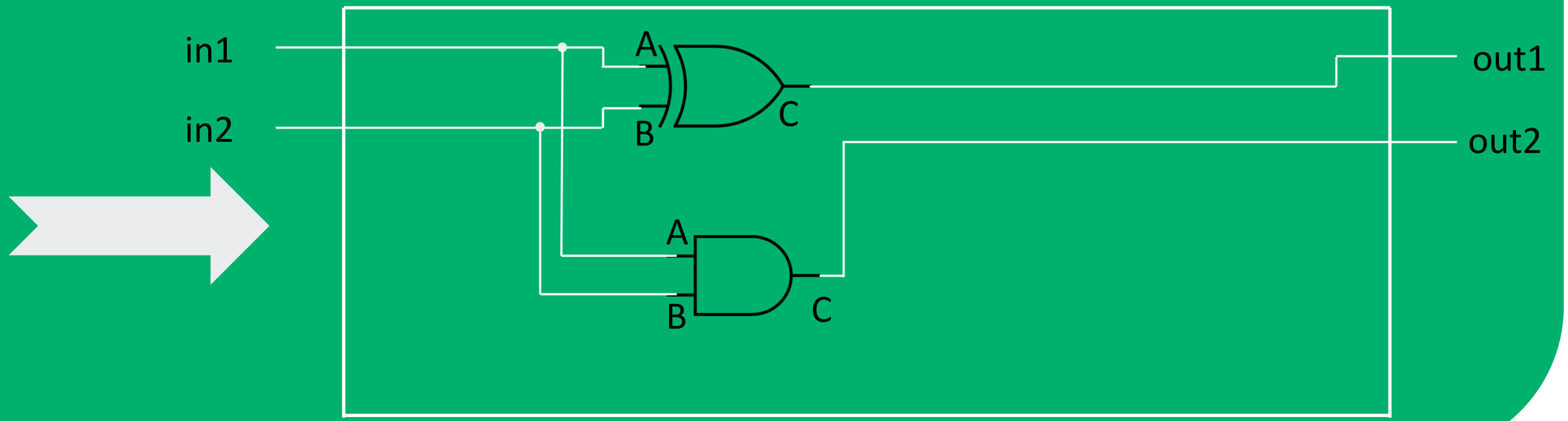
# آزمایش اول

```
Library IEEE;
USE IEEE.std_logic_1164;
Entity xor_gate is
Port (
            A, B: in std_logic;
            C    : out std_logic
);
End Entity xor_gate;

Architecture gatelevel of xor_gate is
Begin
C <= A xor B;
End gatelevel;
```



# آزمایش اول



# آزمایش اول

```
Library IEEE;
USE IEEE.std_logic_1164;
Entity and_gate is
Port (
            A, B: in std_logic;
            C    : out std_logic
);
End Entity and_gate;

Architecture gatelevel of and_gate
is
Begin
C <= A and B;
End gatelevel;
```

```
Library IEEE;
USE IEEE.std_logic_1164;
Entity xor_gate is
Port (
            A, B: in std_logic;
            C    : out std_logic
);
End Entity xor_gate;

Architecture gatelevel of xor_gate is
Begin
C <= A xor B;
End gatelevel;
```

# آزمایش اول

```
Library IEEE;
USE IEEE.std_logic_1164;
Entity half_adder is
Port (
        in1, in2: in std_logic;
        out1, out2    : out std_logic
);
End Entity half_adder;

Architecture structure of half_adder is
        component xor_gate is
        port(
                A, B: in std_logic;
                C     :out srd_logic
        );
        End Component xor_gate
```

# آزمایش اول

```
        component and_gate is
        port(
                A, B: in std_logic;
                C     :out srd_logic
        );
        End Component and_gate
Begin
        xor_gate_instance0: xor_gate port map (A=>in1, B=>in2, C=>out1);
        and_gate_instance0: and_gate port map (A=>in1, B=>in2, C=>out2);
End structure;
```

## تست بنچ

```
library ieee;
use ieee.std_logic_1164.all;

entity and_gate_tb is
end and_gate_tb;

architecture tb of and_gate_tb is
    signal a, b : std_logic;  -- inputs
    signal and_result : std_logic;  -- outputs
begin
    -- connecting testbench signals with half_adder.vhd
    UUT : entity work.and_gate port map (a => a, b => b, and_result =>
and_result);

a <= '0', '1' after 20 ns, '0' after 40 ns, '1' after 60 ns;
    b <= '0', '1' after 40 ns;
end tb ;
```

# تکلیف شماره یک

**قسمت اول:** تعریف قطعات زیر و نوشتن تست بنچ (تک بیتی)

گیت 2 OR ورودی
گیت 2 AND ورودی
2 XOR ورودی
تعریف یک نیم جمع کننده

**قسمت دوم:**
۴ OR ورودی با استفاده از گیت OR دو ورودی
تعریف یک تمام جمع کننده با استفاده از نیم جمع کننده