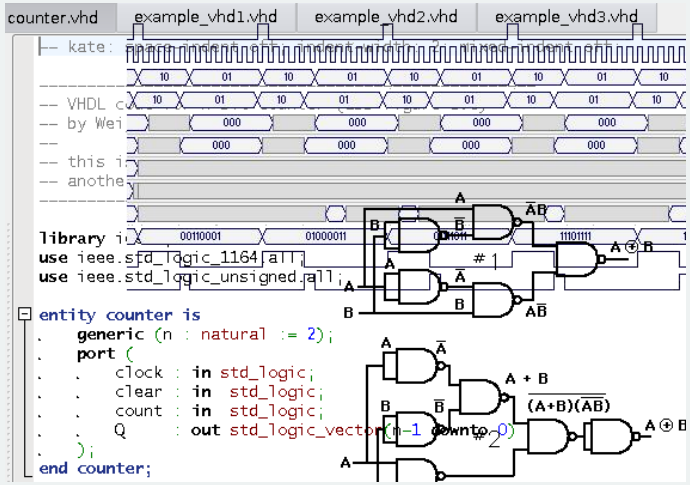
МИЭТ
Национальный исследовательский университет «МИЭТ»
Кафедра ПКИМС

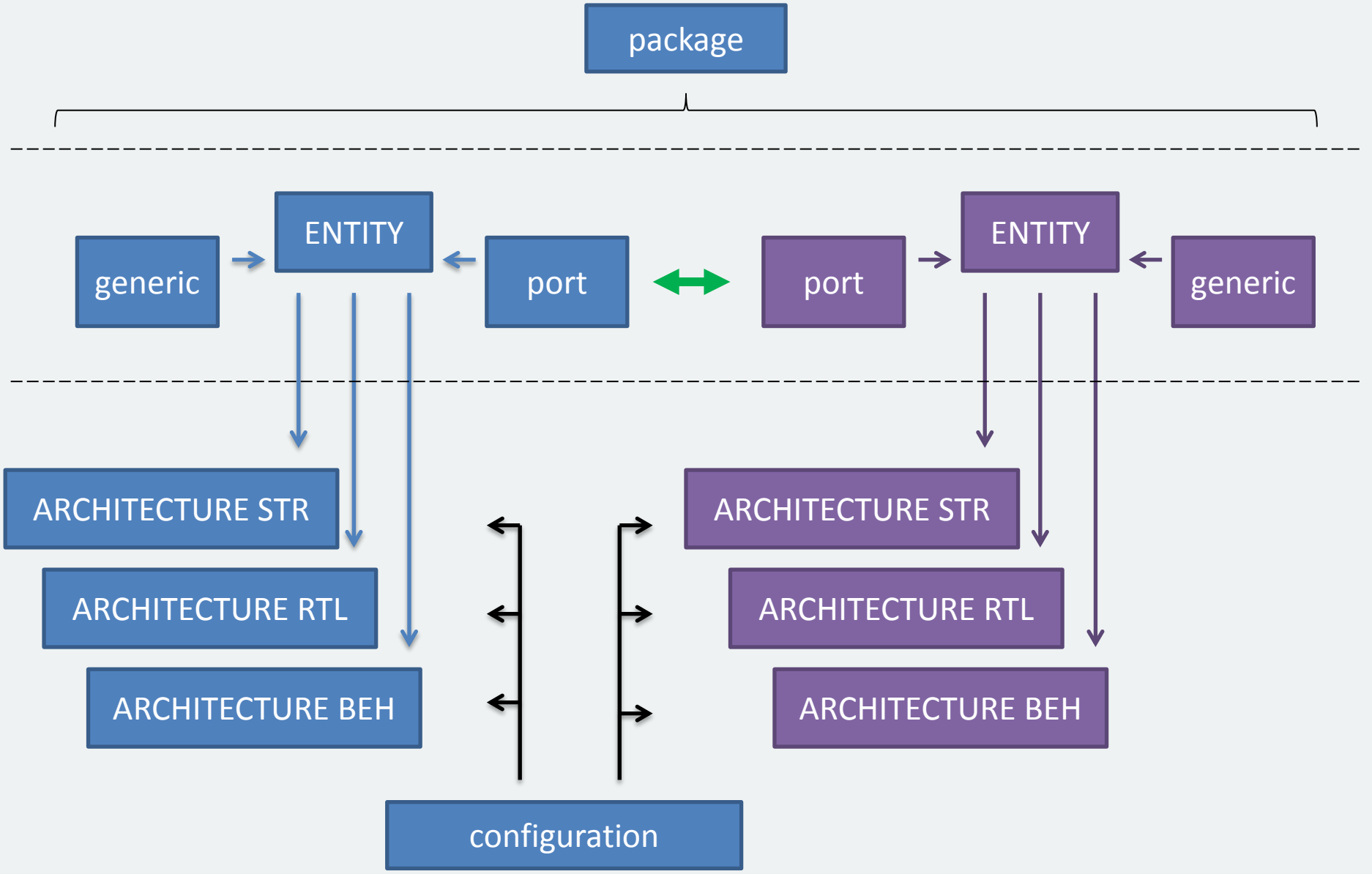
Лингвистические средства проектирования



Лекция 2

Основы описания структурного, регистрового и поведенческого архитектурных тел.

Архитектура проекта на VHDL



Интерфейс устройства

```
entity <имя_интерфейса> is

   [ generic(<список_параметров>); ]
   [ port(<список_портов>); ]

[ begin

   <типы данных, значения констант>;

]

end [entity] [<имя_интерфейса>];
```

Архитектура устройства

```
architecture <имя_архитектруры> of <имя_интерфейса> is
   [<список_объявлений>;]
begin
   <список_параллельных_операторов>;
end [architecture] [<имя_архитектуры>];
```

Операторы языка VHDL для описания RTL

NOT

x	y
0	1
1	0

NAND

x1	x2	y
0	0	1
0	1	1
1	0	1
1	1	0

NOR

x1	x2	y
0	0	1
0	1	0
1	0	0
1	1	0

XOR

x1	x2	y
0	0	0
0	1	1
1	0	1
1	1	0

AND

x1	x2	y
0	0	0
0	1	0
1	0	0
1	1	1

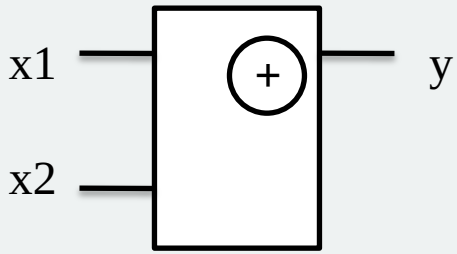
OR

x1	x2	y
0	0	0
0	1	1
1	0	1
1	1	1

XNOR

x1	x2	y
0	0	1
0	1	0
1	0	0
1	1	1

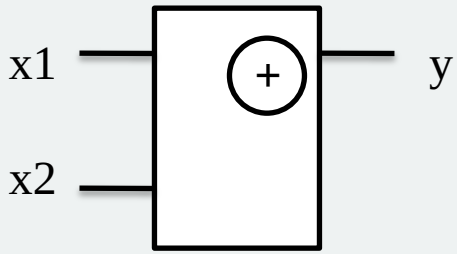
Элемент «исключающее или» (1)




```
entity xor2 is
  port(x1, x2: in std_logic;
               y: out std_logic);
end xor2;
```

```
architecture RTL of xor2 is
begin
   y <= x1 xor x2;
end RTL;
```

Элемент «исключающее или» (2)



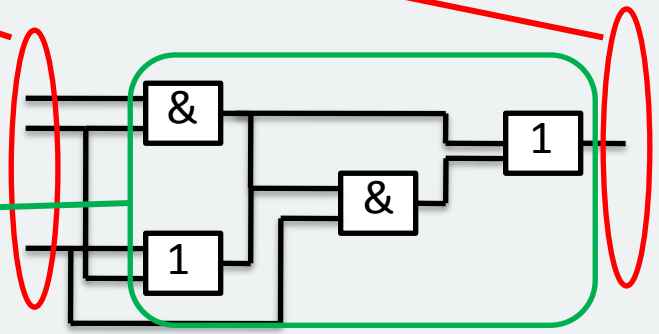

```
entity xor2 is
  port(x1, x2: in std_logic;
               y: out std_logic);
end xor2;
```

```
architecture RTL of xor2 is
begin
   y <= (x1 and (not x2)) or (not x1 and x2);
end RTL;
```

Структурное архитектурное тело

Структурное архитектурное тело описывает поведение схемы косвенно, через поведение компонентов.

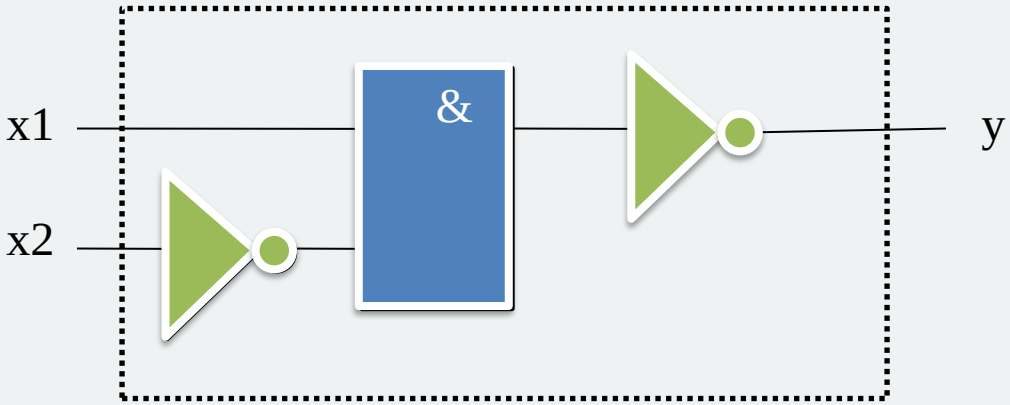
```
entity  DEVICE is
  port(...);
end DEVICE;


architecture STR of DEVICE is
  ...
begin

  ...

end STR;
```



Пример. Этап 1: проектирование библиотеки

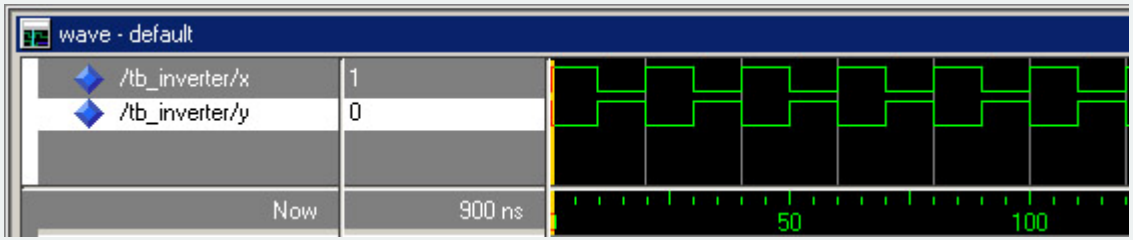
```
entity inv is
  port(x: in bit;
       y: out bit);
end inv;

architecture RTL of inv is
begin
  y <= not x;
end RTL;
```

+ тестбенч

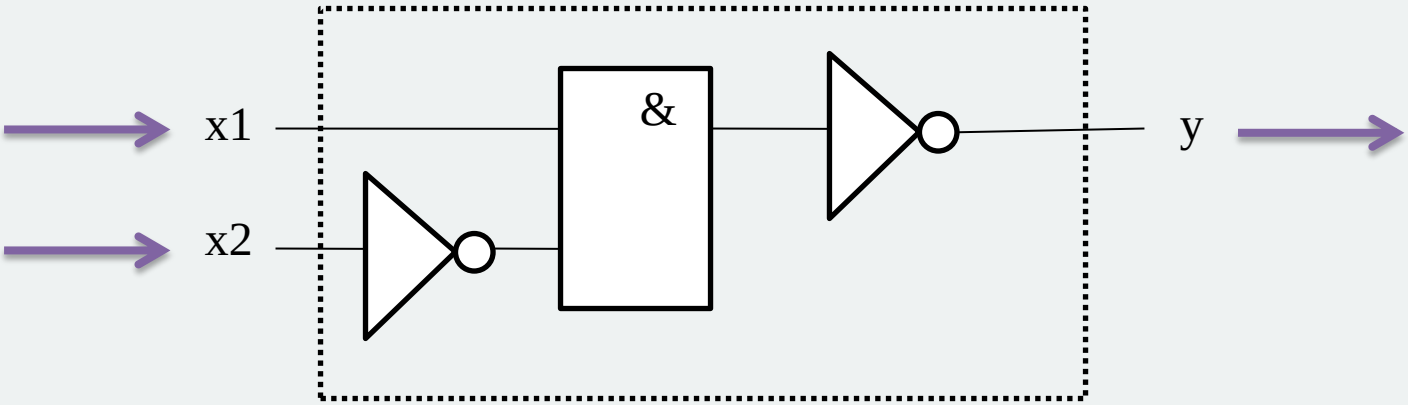


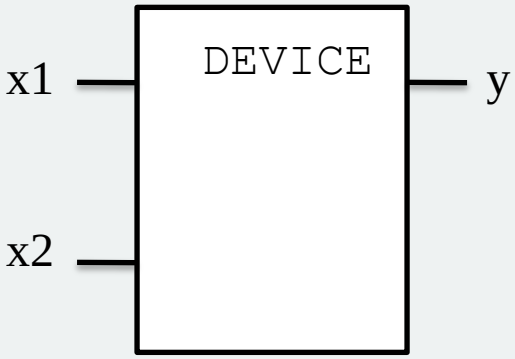

```
entity and2 is
  port(x1, x2: in bit;
               y: out bit);
end and2;

architecture RTL of and2 is
begin
  y <= x1 and x2;
end RTL;
```

+ тестбенч



Пример. Этап 2: описание интерфейса схемы



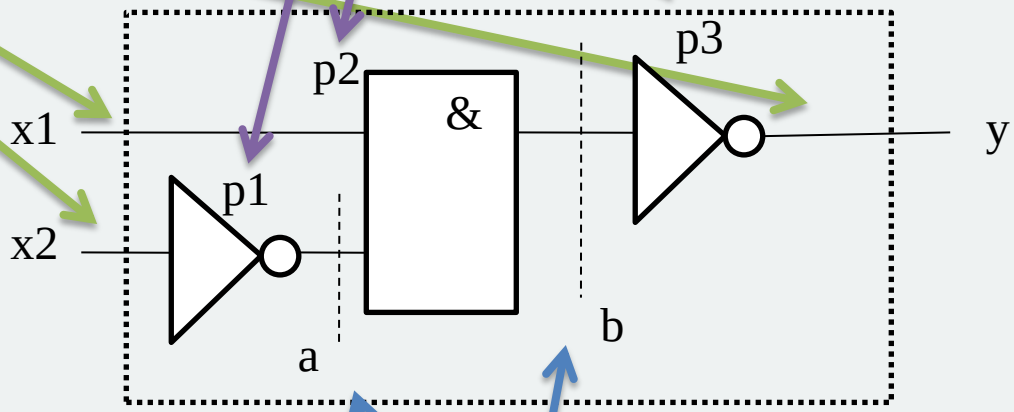



```
entity DEVICE is
  port(x1: in bit;
       x2: in bit;
        y: out bit);
end DEVICE;
```

Пример. Этап 3: расстановка недостающих объектов

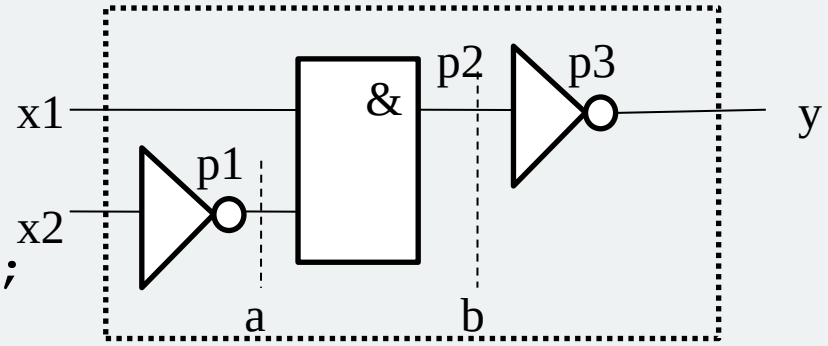
Задаём метки (имена) компонентов

Имена портов известны из интерфейса!



Добавляем в схему промежуточные сигналы

Пример. Этап 4: описание взаимосвязи компонентов




```
architecture STR of DEVICE is
  component inv
    port(x: in bit; y: out bit);
  end component;
  component and2
    port(x1, x2: in bit; y: out bit);
  end component;

  signal a, b: bit;

begin

  p1 : inv port map(x2, a);
  p2 : and2 port map(x1, a, b);
  p3 : inv port map(b, y);

end STR;
```

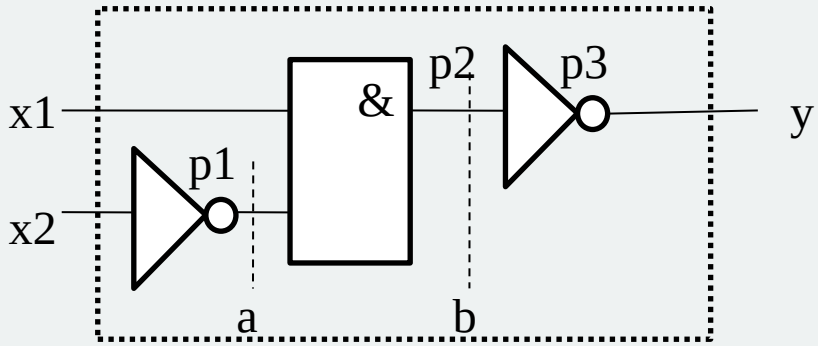
Какие типы элементов будут использованы?

Перечень недостающих сигналов

Связь элементов посредством сигналов и портов

Ассоциативное описание портов

```
architecture STR of DEVICE is
  ...
  component inv
    port(y: out bit; x: in bit);
  end component;
  ...
begin
  p1 : inv port map(x2, a);
  ...
```




```
p1 : inv port map(x2, a);

p2 : and2 port map(x1, a, b);

p3 : inv port map(b, y);
```

```
p1 : inv port map(x=>x2, y=>a);

p2 : and2 port map(x1=>x1, x2=>a, y=>b);

p3 : inv port map(y=>y, x=>b);
```

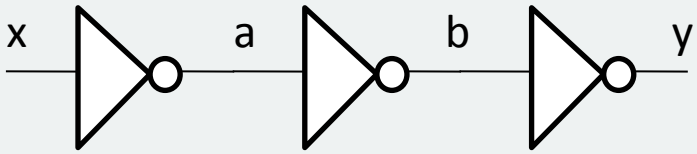
Использование настраиваемых параметров (2)

```
entity <имя_интерфейса> is
  [ generic(<список_параметров>); ]
  [ port(<список_портов>); ]
end [entity] [<имя_интерфейса>];
```

Задержка указывается с помощью настраиваемого параметра

```
entity inv is
  generic(t: time);
  port(x: in bit;
       y: out bit);
end inv;

architecture RTL of inv is
begin
  y <= not x after t;
end RTL;
```

Использование настраиваемых параметров

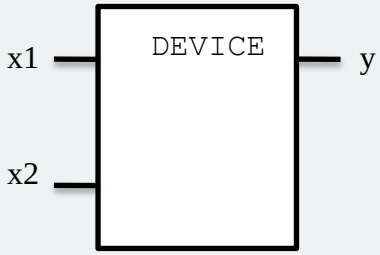



```
entity <имя_интерфейса> is
  [ generic(<список_параметров>); ]
  [ port(<список_портов>); ]
end [entity] [<имя_интерфейса>];
```

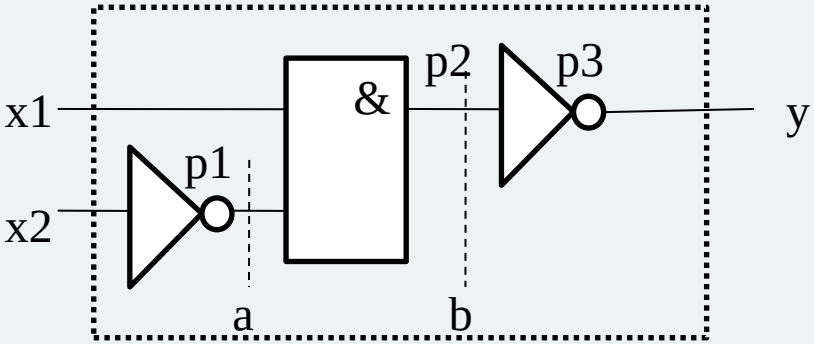
```
p1 : inv port map(x, a);
p2 : inv port map(a, b);
p3 : inv port map(b, y);
```

```
p1 : inv port map(x, a)
     generic map(4 ns);
p2 : inv port map(a, b)
     generic map(3.8 ns);
p3 : inv port map(b, y)
     generic map(3.6 ns);
```

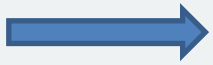
RTL-описание схем




```
entity DEVICE is
  port(x1: in bit;
       x2: in bit;
        y: out bit);
end DEVICE;
```



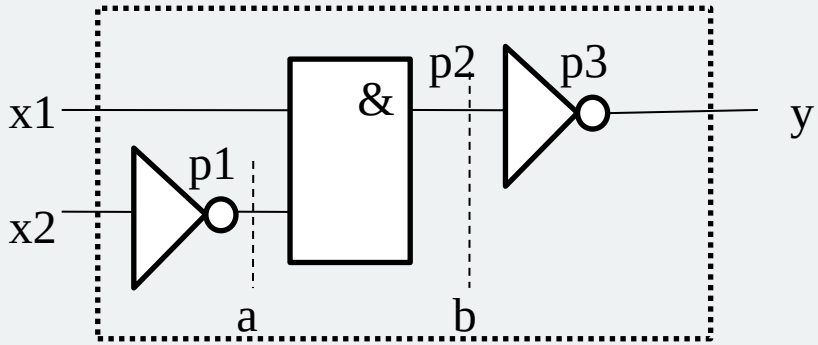

```
architecture STR of DEVICE is
  component inv
    port(x: in bit; y: out bit);
  end component;
  component and2
    port(x1, x2: in bit; y: out bit);
  end component;

  signal a, b: bit;

begin

  p1 : inv port map(x2, a);
  p2 : and2 port map(x1, a, b);
  p3 : inv port map(b, y);

end STR;
```



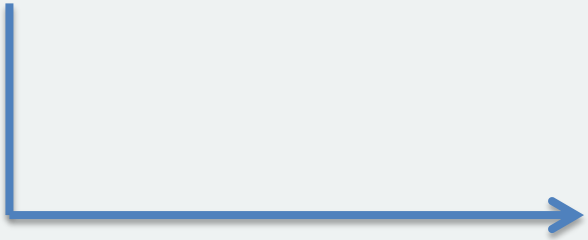
```
architecture RTL of DEVICE is

  signal a, b: bit;

begin

  p1 : a <= not x2;
  p2 : b <= a and x1;
  p3 : y <= not b;

end RTL;
```

Формы RTL-описания схем

```
architecture RTL of DEVICE is
  signal a, b: bit;
begin
  p1 : a <= not x2;
  p2 : b <= a and x1;
  p3 : y <= not b;
end RTL;
```

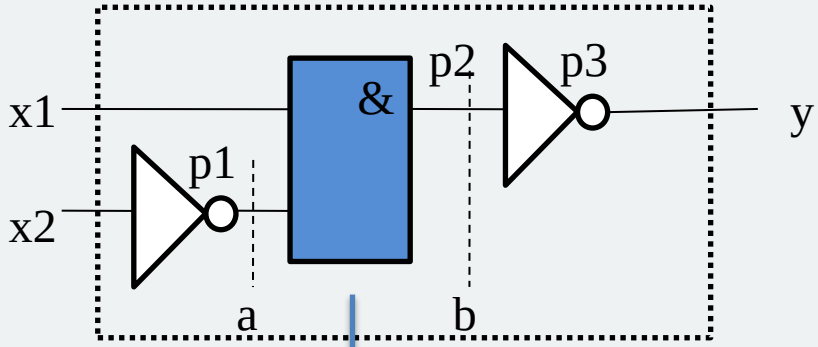


```
architecture RTL of DEVICE is
begin
  p1 : y <= not (x1 and not x2);
end RTL;
```



```
architecture RTL of DEVICE is
begin
  p1 : y <= x1 nand (not x2);
end RTL;
```

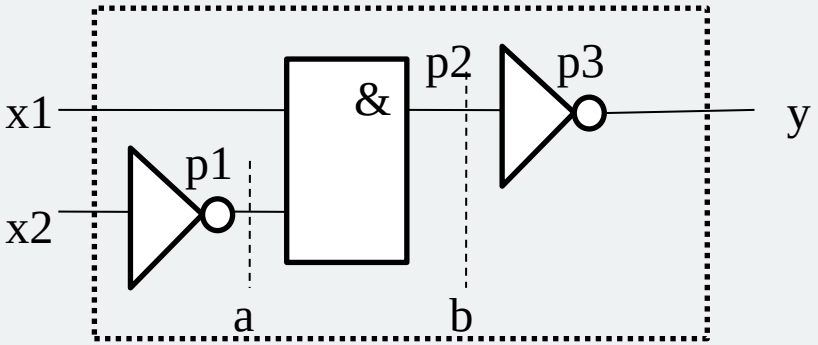
Комбинирование описаний сложных схем




```
architecture STR of DEVICE is
  component and2
    port(x1, x2: in bit; y: out bit);
  end component;

  signal a, b: bit;
begin

  p1 : a <= not x2;
  p2 : y <= not b;

  p3 : and2 port map(x1, a, b);

end STR;
```

Поведенческое архитектурное тело

x1	x2	y
0	0	1
0	1	1
1	0	0
1	1	1

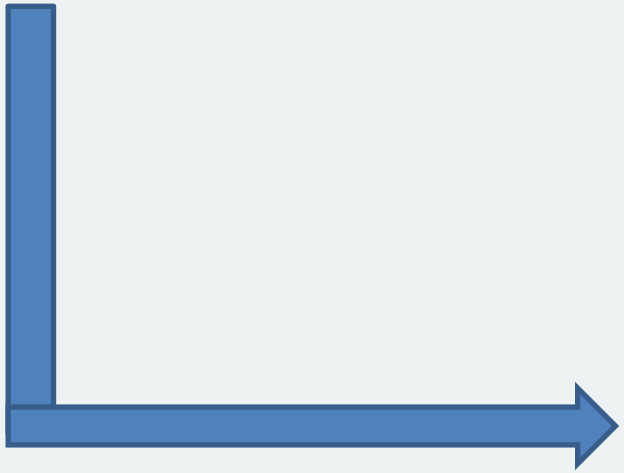



```
entity DEVICE is
  port(x1: in bit;
       x2: in bit;
        y: out bit);
end DEVICE;
```

```
architecture BEH of DEVICE is
begin

  process(x1, x2)
  begin
    if(x1='1' and x2='0') then
      y <= '0';
    else
      y <= '1';
    end if;
  end process;

end BEH;
```

Описание тестовых последовательностей с помощью процессов

```
signal x : std_logic;

...

p1 : x <= '0' after 0 ns, '1' after 10ns, '0' after 20 ns,
...
```



```
signal x : std_logic;

...

p1 : process
     begin
        x <= '0';
        wait for 10 ns;
        x <= '1';
        wait for 10 ns;
end process;
```