



ПРАКТИЧЕСКАЯ РАБОТА №1. АППАРАТНЫЕ РЕСУРСЫ ПЛИС. АРИФМЕТИЧЕСКИЕ ОПЕРАЦИИ

Преподаватель кафедры ВТ
Инженер лаборатории специализированных
вычислительных систем

Дуксин Н.А.

ОЦЕНИВАНИЕ В РАМКАХ СЕМЕСТРА

	Отсутствие допуска	Допуск	Автомат
Лекции	< 12 посещений	$12 \leq \text{посещений} < 14$	14-16
Практики	< 4 практик или ср.ар. по коллоквиуму < 3.5	4 практики и $3.5 \leq \text{ср.ар. По коллоквиуму} \leq 4.5$	6 практик и $\geq 4.5 \text{ ср.ар. По коллоквиуму}$
Лабы	< 4 работ	1-3, (4 или 5 или 6)	1-6
СР	+0.5 балла к ср. ар. по коллоквиуму		

Регламент выполнения работы озвучивается на занятии при выдаче работы

Билет 4 вопроса: 2 теоретических, 1 практический по курсу практик, 1 практический по курсу лабораторных

ОСНОВНЫЕ ВОПРОСЫ

- Маршрут проектирования
- Аппаратные ресурсы ПЛИС семейства Artix седьмой серии
- Арифметика. Примеры.
- Задание
- Выводы

01

МАРШРУТ ПРОЕКТИРОВАНИЯ

АППАРАТНАЯ БАЗА РЕАЛИЗАЦИИ
В МАРШРУТЕ ПРОЕКТИРОВАНИЯ

01

МАРШРУТ ПРОЕКТИРОВАНИЯ



02

АППАРАТНЫЕ РЕСУРСЫ ПЛИС

ПЛИС СЕМЕЙСТВА Artix СЕРИЯ 7
AMD (XILINX)

02 АППАРАТНЫЕ РЕСУРСЫ ПЛИС

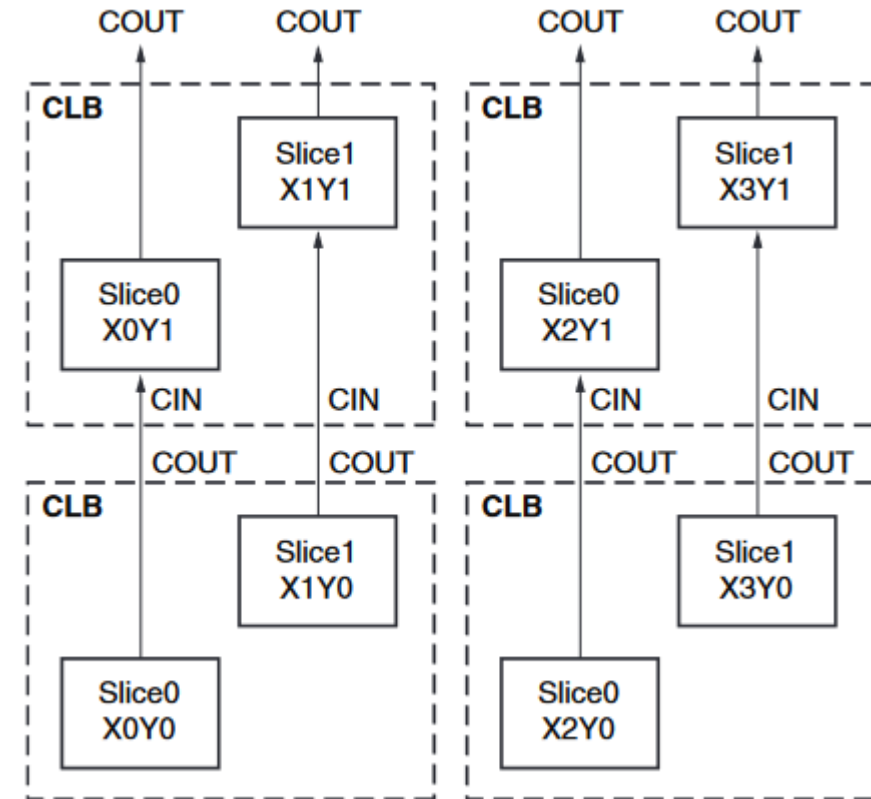
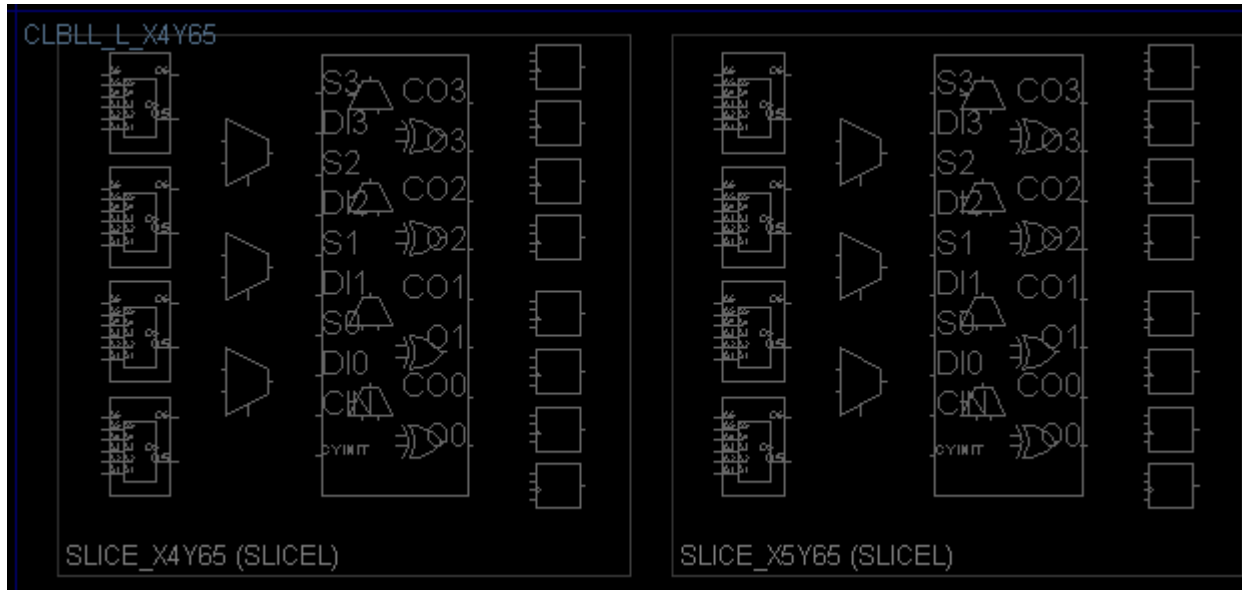
Аппаратные ресурсы ПЛИС семейства Artix седьмой серии

- CLB
 - LUT
 - Carry chain
 - Мультиплексоры
 - Триггеры
- BRAM
- DSP

02

АППАРАТНЫЕ РЕСУРСЫ ПЛИС. CLB

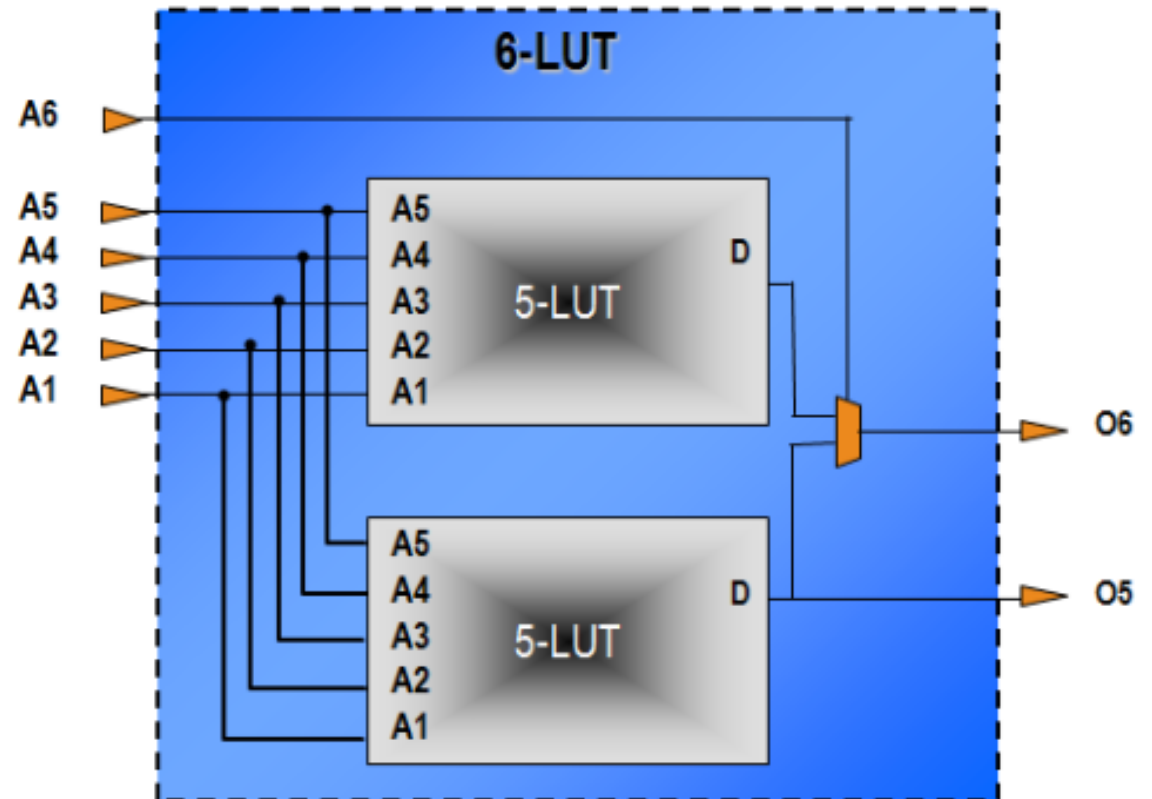
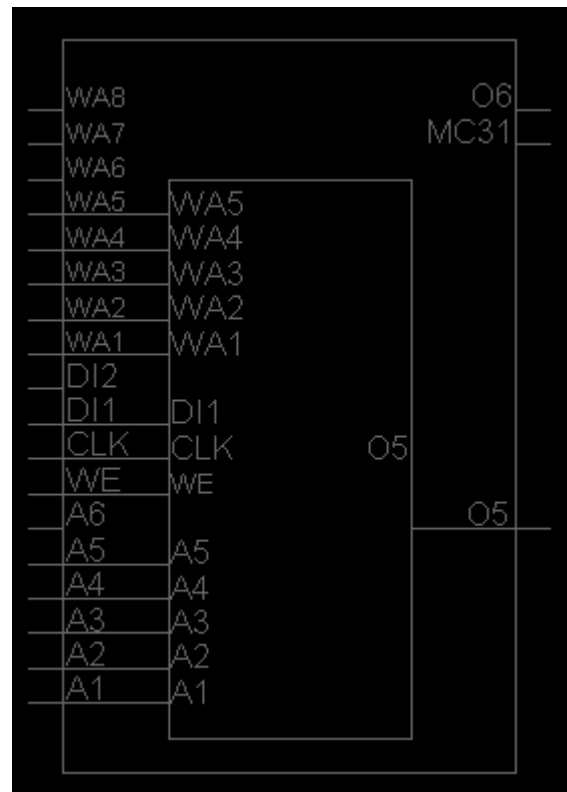
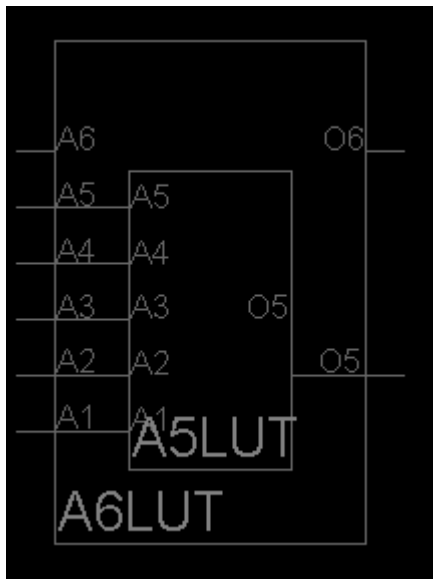
Configurable Logic Blocks



02

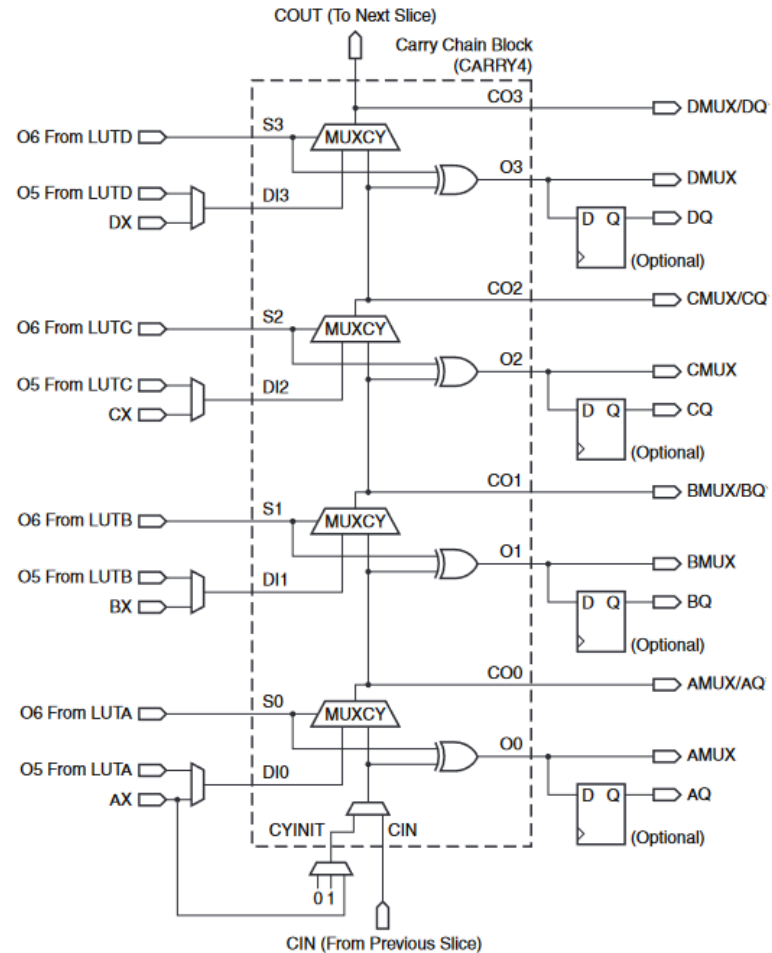
АППАРАТНЫЕ РЕСУРСЫ ПЛИС. LUT

Look Up Table



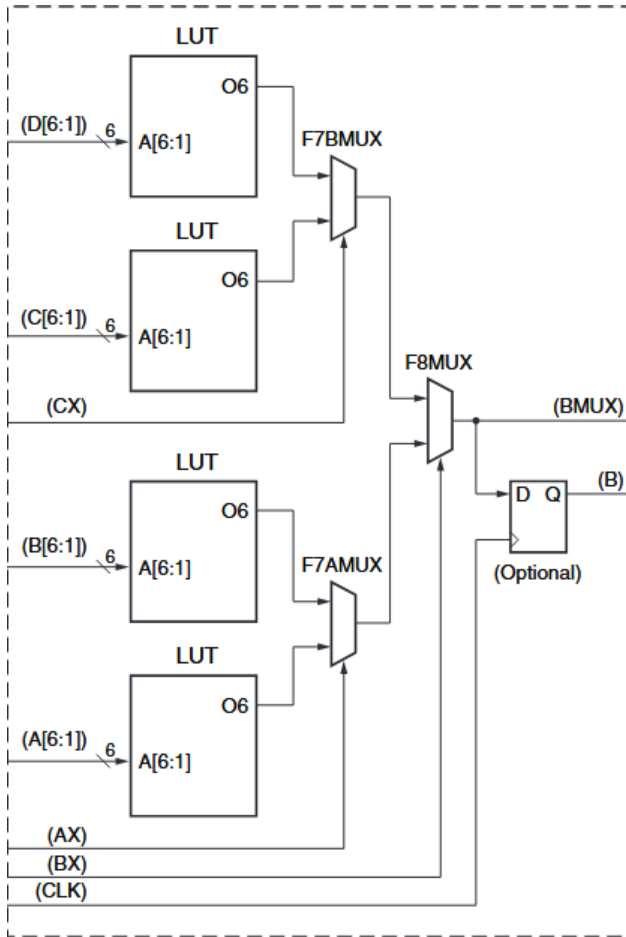
02

АППАРАТНЫЕ РЕСУРСЫ ПЛИС. CARRY CHAIN



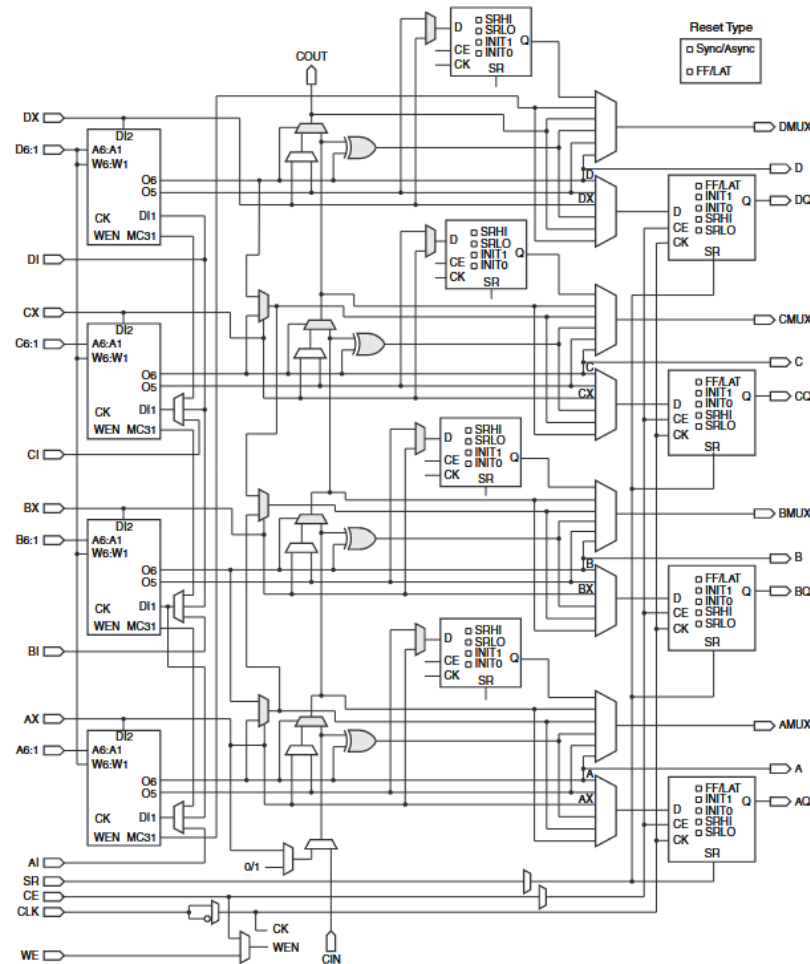
02

АППАРАТНЫЕ РЕСУРСЫ ПЛИС. МУЛЬТИПЛЕКСОРЫ



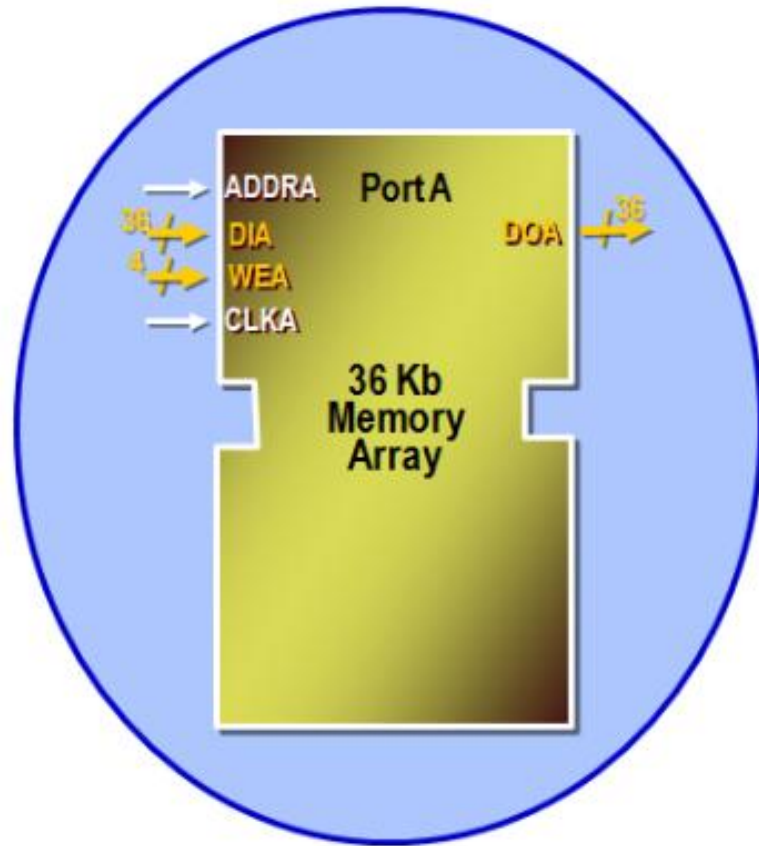
02

АППАРАТНЫЕ РЕСУРСЫ ПЛИС. ТРИГГЕРЫ



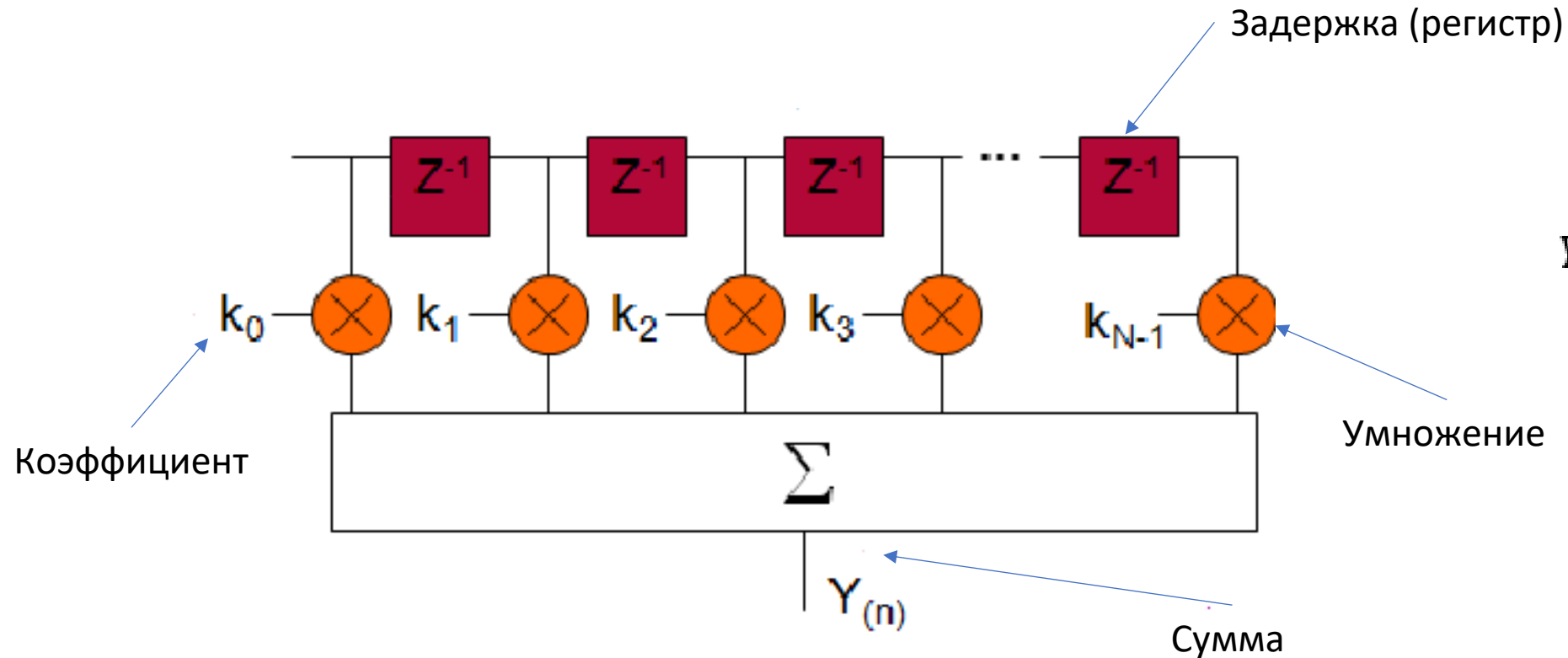
02

АППАРАТНЫЕ РЕСУРСЫ ПЛИС. BRAM



02

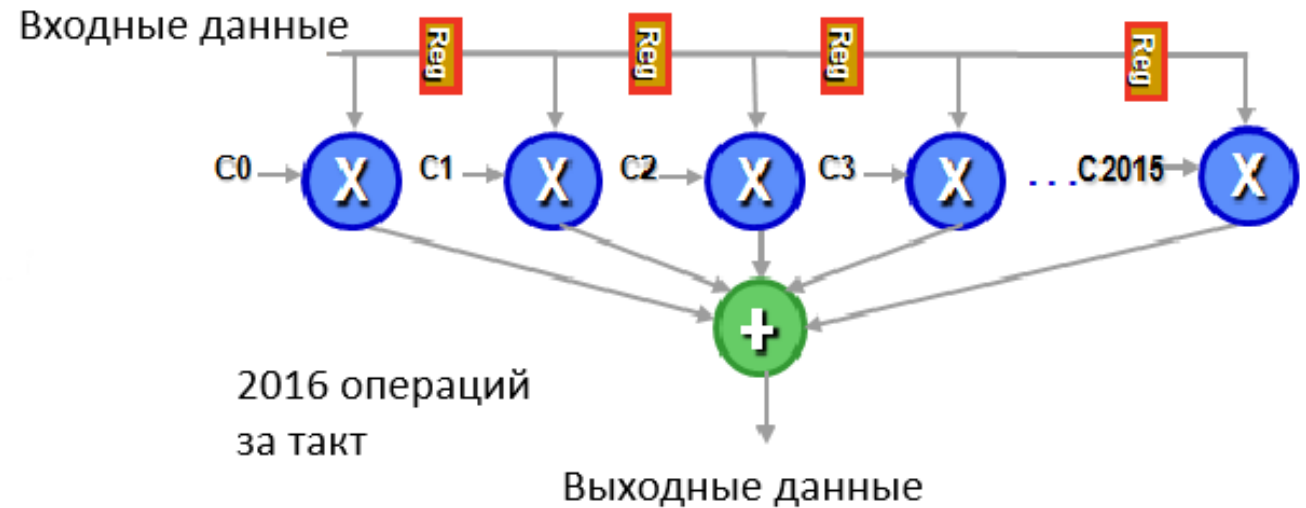
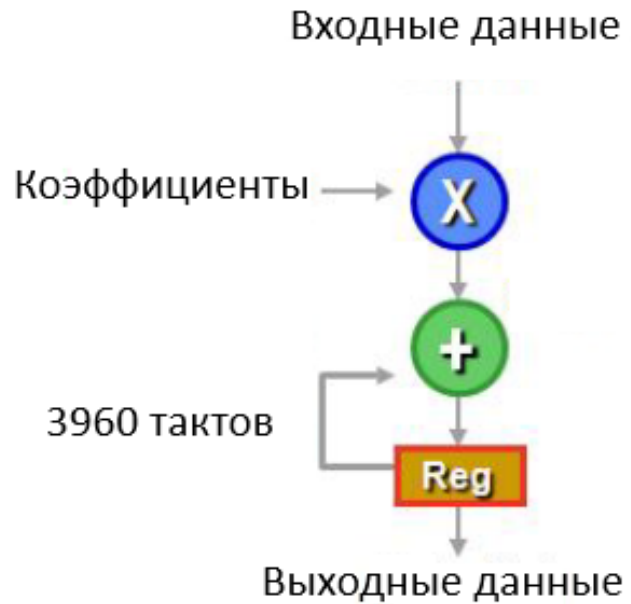
АППАРАТНЫЕ РЕСУРСЫ ПЛИС. DSP



$$Y(n) = \sum_{i=0}^{N-1} k_i * x_{(n-i)}$$

02

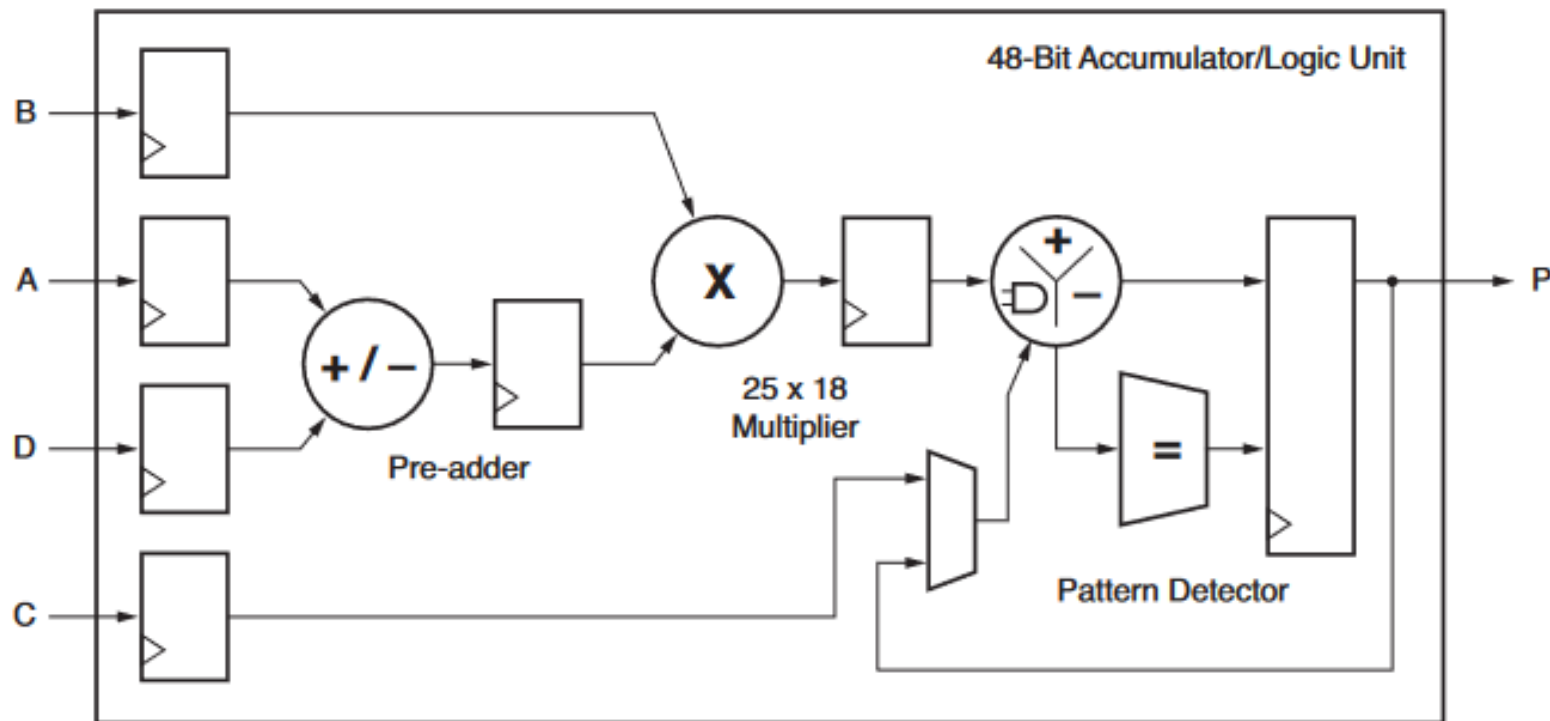
АППАРАТНЫЕ РЕСУРСЫ ПЛИС. DSP



02

АППАРАТНЫЕ РЕСУРСЫ ПЛИС. DSP

UG479



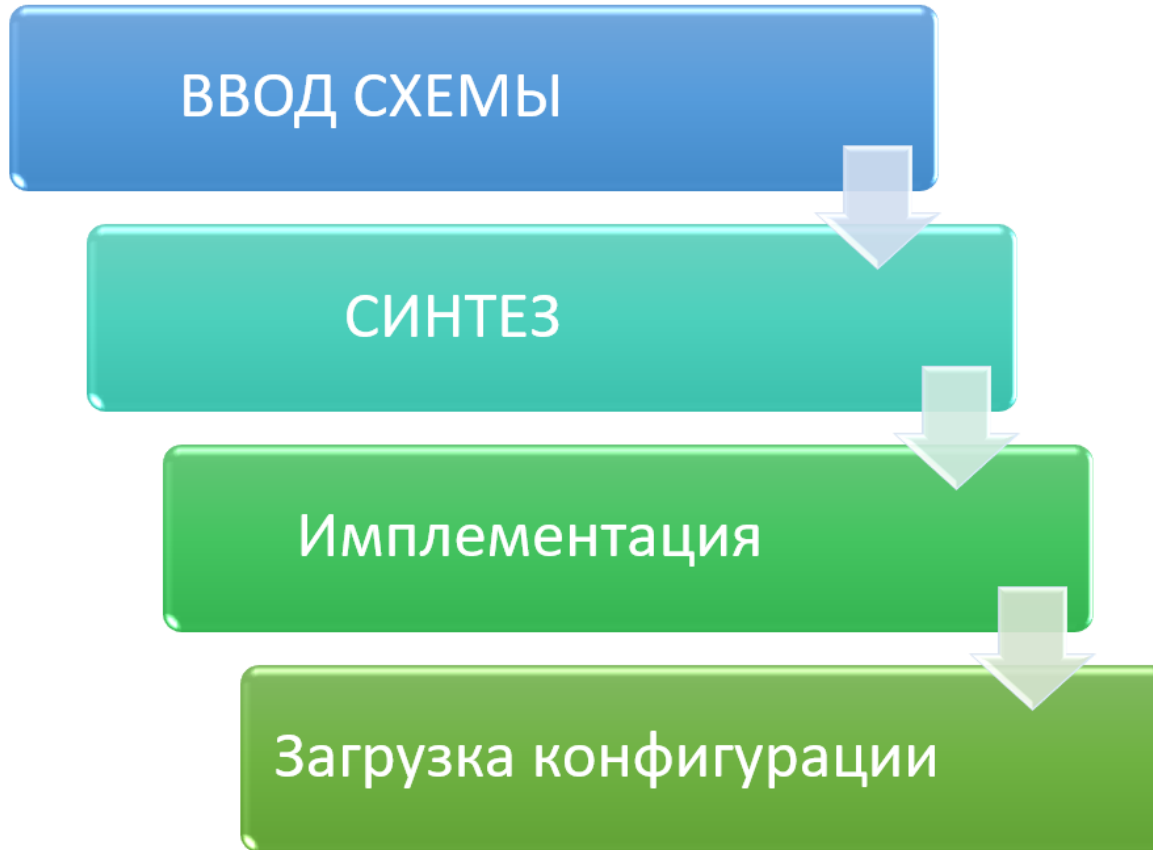
03

АРИФМЕТИКА

ОСОБЕННОСТИ
АРИФМЕТИЧЕСКИХ ОПЕРАЦИЙ

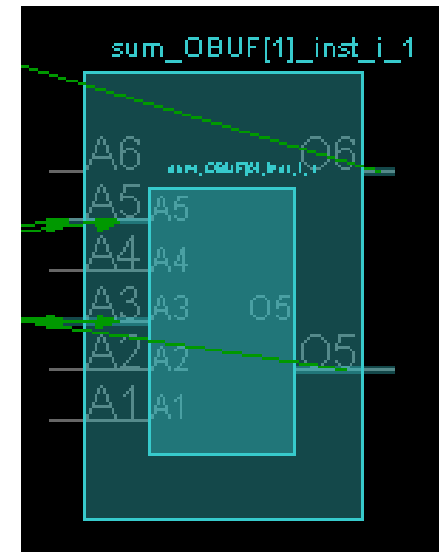
03

АРИФМЕТИКА. МАРШРУТ ПРОЕКТИРОВАНИЯ



03 АРИФМЕТИКА. СЛОЖЕНИЕ

```
module sum(  
    input a, b,  
    output [1:0] sum  
);  
  
    assign sum = a + b;  
  
endmodule
```



Resource	Utilization	Available	Utilization %
LUT	1	63400	0.01
IO	4	210	1.90

03 АРИФМЕТИКА. СЛОЖЕНИЕ

```
module sum(  
    input a, b,  
    output [1:0] sum  
);  
  
    assign sum = a + b;  
  
endmodule
```

The screenshot displays the implementation of a 2-bit adder in a logic synthesizer. The top part shows the hierarchy of the 'sum' module, with leaf cells including input buffers, output buffers, and LUT2s. The bottom part shows the 'Cell Properties' window for two LUTs, displaying their truth tables.

Left Cell Properties (sum_OBUF[0]_inst_i_1):

I1	I0	O=I0 & I1 + I0 & I1
0	0	0
0	1	1
1	0	1
1	1	0

Right Cell Properties (sum_OBUF[1]_inst_i_1):

I1	I0	O=I0 & I1
0	0	0
0	1	0
1	0	0
1	1	1

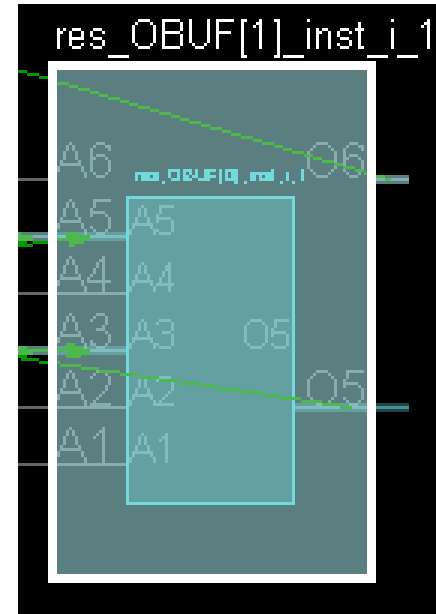
03

АРИФМЕТИКА. ВЫЧИТАНИЕ

```
module sub(
    input a, b,
    output [1:0] res
);

    assign res = a - b;

endmodule
```



Resource	Utilization	Available	Utilization %
LUT	1	63400	0.01
IO	4	210	1.90

03

АРИФМЕТИКА. ВЫЧИТАНИЕ

```
module sub(  
    input a, b,  
    output [1:0] res  
);  
  
    assign res = a - b;  
  
endmodule
```

The figure displays four screenshots from a logic synthesizer interface, illustrating the implementation of a subtraction module.

The top row shows two views of the **Netlist** window. Both windows show a hierarchy with **Nets (8)** and **Leaf Cells (6)**. The leaf cells include **a_IBUF_inst (IBUF)**, **b_IBUF_inst (IBUF)**, **res_OBUF[0]_inst (OBUF)**, **res_OBUF[0]_inst_i_1 (LUT2)**, **res_OBUF[1]_inst (OBUF)**, and **res_OBUF[1]_inst_i_1 (LUT2)**. The **res_OBUF[0]_inst_i_1 (LUT2)** and **res_OBUF[1]_inst_i_1 (LUT2)** cells are highlighted in blue in both screenshots.

The bottom row shows two views of the **Cell Properties** window. The left window shows the properties for **res_OBUF[0]_inst_i_1**, and the right window shows the properties for **res_OBUF[1]_inst_i_1**. Both windows display a truth table with columns **I1**, **I0**, and **O=I0 & !I1**.

The truth table for **res_OBUF[0]_inst_i_1** is:

I1	I0	O=I0 & !I1
0	0	0
0	1	1
1	0	1
1	1	0

The truth table for **res_OBUF[1]_inst_i_1** is:

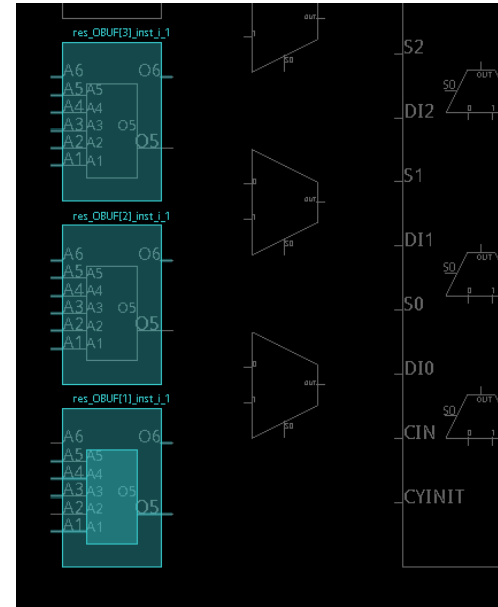
I1	I0	O=I0 & !I1
0	0	0
0	1	1
1	0	0
1	1	0

Both windows also feature an **Edit LUT Equation...** button and a tabbed interface at the bottom with **Properties**, **Power**, **Nets**, **Cell Pins**, and **Truth Table** tabs.

03

АРИФМЕТИКА. СЛОЖЕНИЕ И ВЫЧИТАНИЕ НА БОЛЬШЕЙ РАЗРЯДНОСТИ

```
module add(
    input [2:0] a,
    input [2:0] b,
    output [3:0] res
);
    assign res = a + b;
endmodule
```

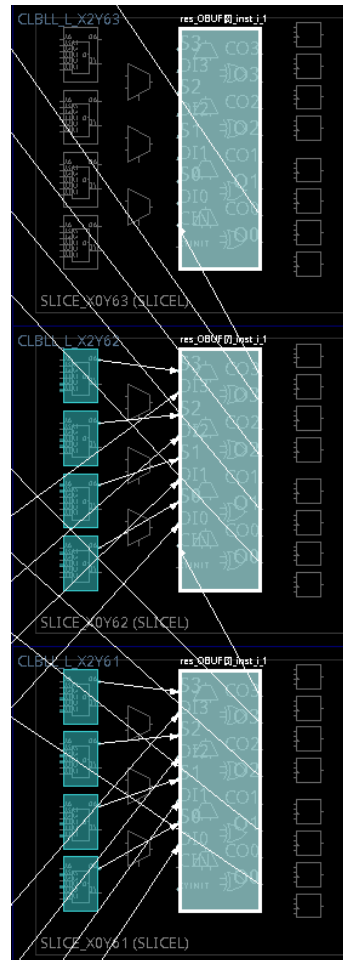


Resource	Estimation	Available	Utilization %
LUT	3	63400	0.01
IO	10	210	4.76

03

АРИФМЕТИКА. СЛОЖЕНИЕ И ВЫЧИТАНИЕ НА БОЛЬШЕЙ РАЗРЯДНОСТИ

```
module add(
    input [7:0] a,
    input [7:0] b,
    output [8:0] res
);
    assign res = a + b;
endmodule
```



Ref Name	Used	Functional Category
IBUF	16	IO
OBUF	9	IO
LUT2	8	LUT
CARRY4	3	CarryLogic

03

АРИФМЕТИКА. СЛОЖЕНИЕ И ВЫЧИТАНИЕ НА БОЛЬШЕЙ РАЗРЯДНОСТИ

```
module add(  
    input [7:0] a,  
    input [7:0] b,  
    output [8:0] res  
);  
assign res = a - b;  
endmodule
```

I1	I0	O=I1 & I0 + I1 & I1
0	0	1
0	1	0
1	0	0
1	1	1

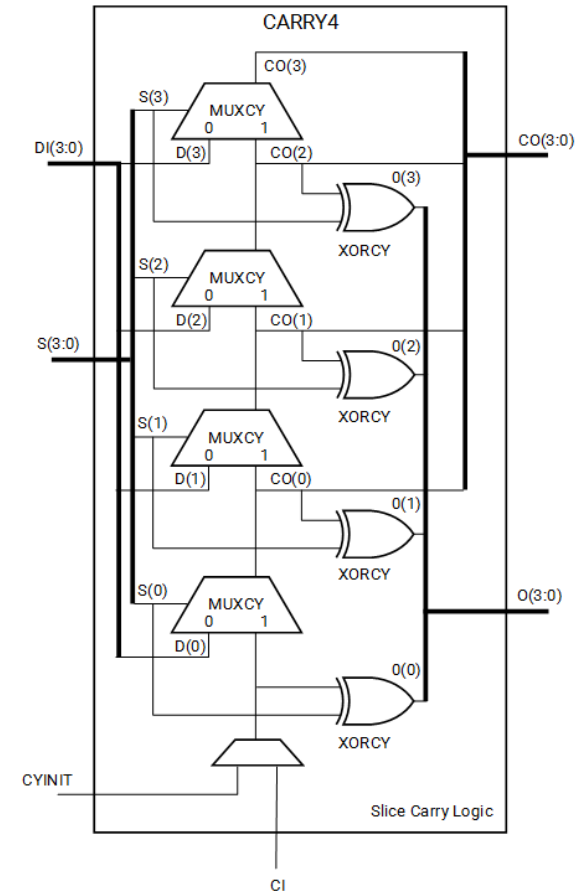
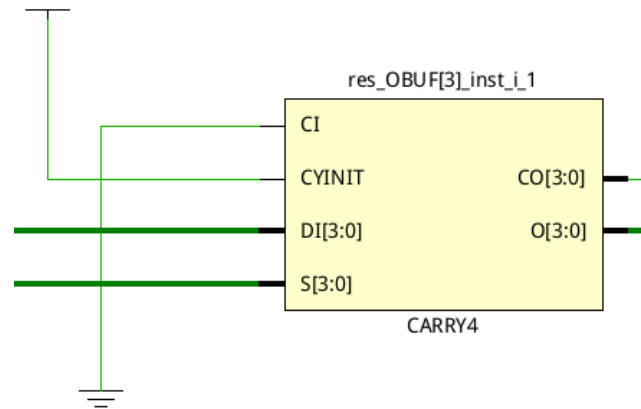
Edit LUT Equation...

Properties Power Nets Cell Pins Truth Table

03

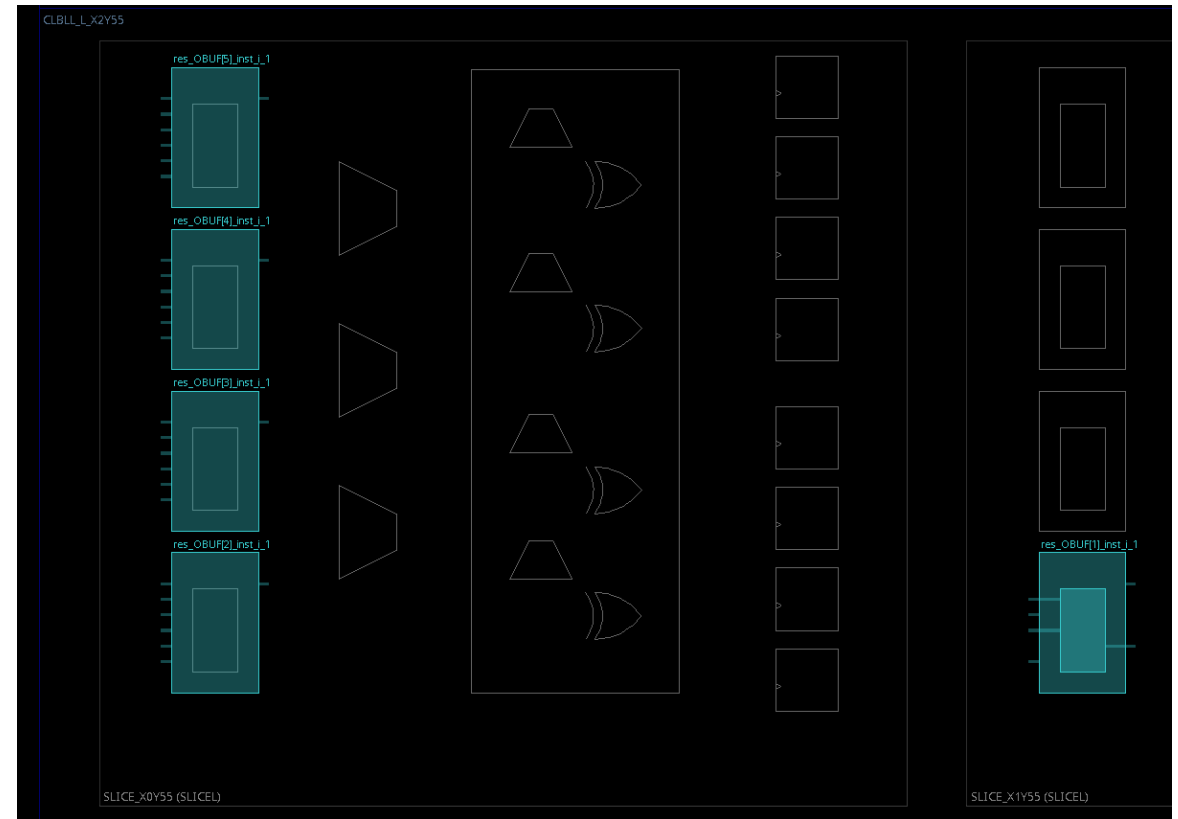
АРИФМЕТИКА. СЛОЖЕНИЕ И ВЫЧИТАНИЕ НА БОЛЬШЕЙ РАЗРЯДНОСТИ

```
module add(
    input [7:0] a,
    input [7:0] b,
    output [8:0] res
);
    assign res = a - b;
endmodule
```



03 АРИФМЕТИКА. УМНОЖЕНИЕ

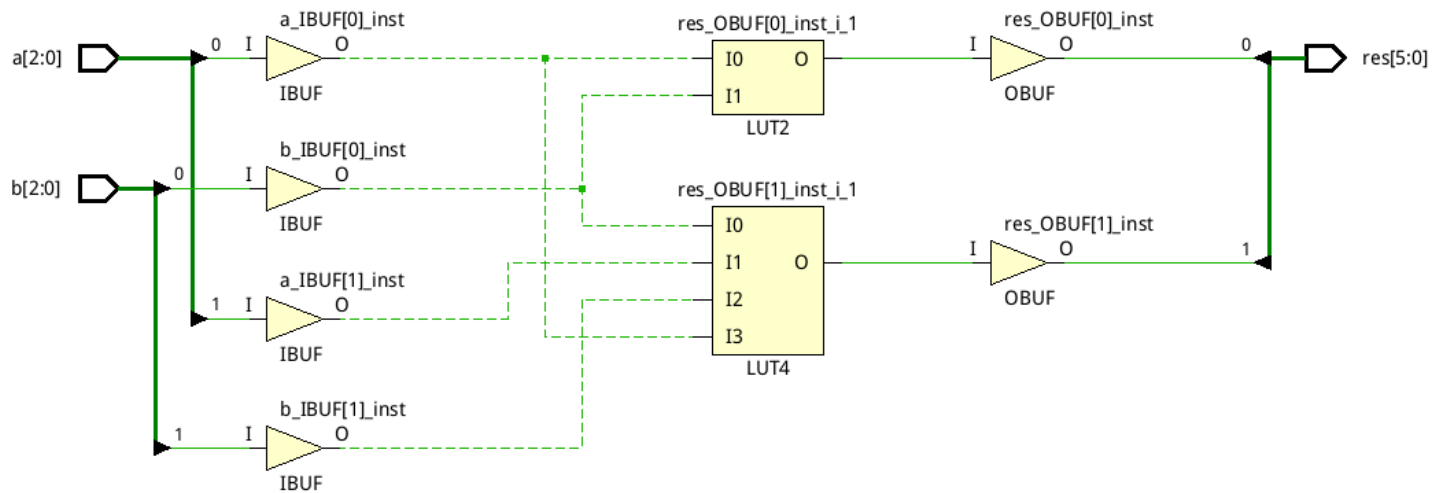
```
module mult(  
    input [2:0] a,  
    input [2:0] b,  
    output [5:0] res  
);  
  
assign res = a * b;  
  
endmodule
```



Resource	Utilization	Available	Utilization %
LUT	5	63400	0.01
IO	12	210	5.71

03

АРИФМЕТИКА. УМНОЖЕНИЕ



$$\begin{array}{r}
 \times 101 \quad a[0] \\
 101 \quad b[0] \\
 \hline
 101 \\
 000 \\
 101 \\
 \hline
 11001 \quad res[0] \\
 \text{LUT2}
 \end{array}$$

$$\begin{array}{r}
 \times 101 \quad a[1:0] \\
 101 \quad b[1:0] \\
 \hline
 101 \\
 000 \\
 101 \\
 \hline
 11001 \quad res[1] \\
 \text{LUT4}
 \end{array}$$

03 АРИФМЕТИКА. УМНОЖЕНИЕ

Sources Netlist Cell Properties x

res_OBUF[1]_inst_i_1

I3	I2	I1	I0	O=I0 & I1 & I3 + I0 & I1 & I2 + I1 & I2 & I3 + I0 & I2 & I3
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

Edit LUT Equation...

General Properties Power Nets Cell Pins Truth Table

res_OBUF[0]_inst_i_1

I1	I0	O=I0 & I1
0	0	0
0	1	0
1	0	0
1	1	1

Edit LUT Equation...

Properties Power Nets Cell Pins Truth Table

03

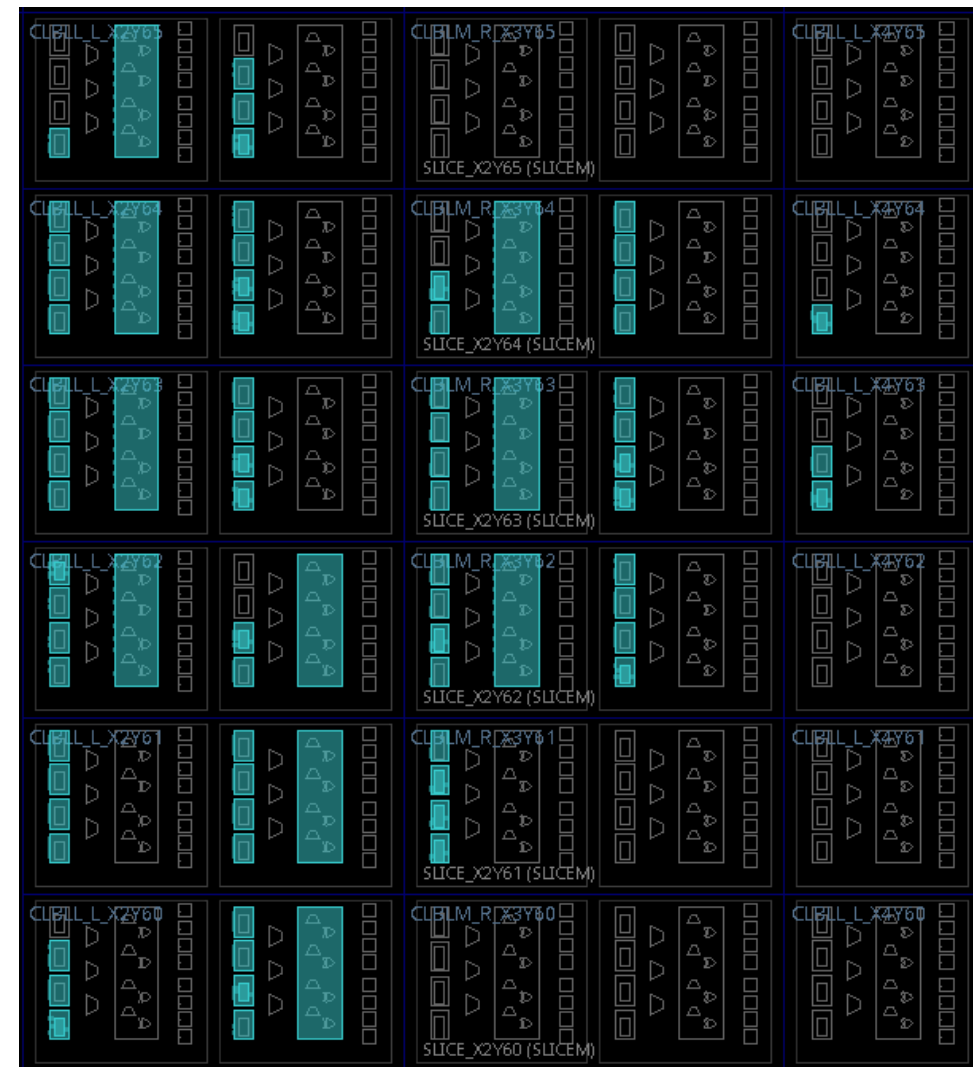
АРИФМЕТИКА. УМНОЖЕНИЕ

```
module mult(
    input [7:0] a,
    input [7:0] b,
    output [15:0] res
);

assign res = a * b;

endmodule
```

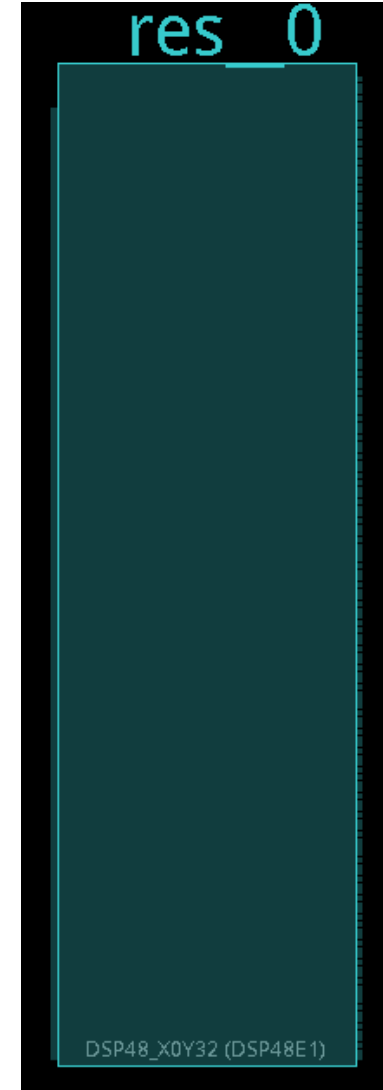
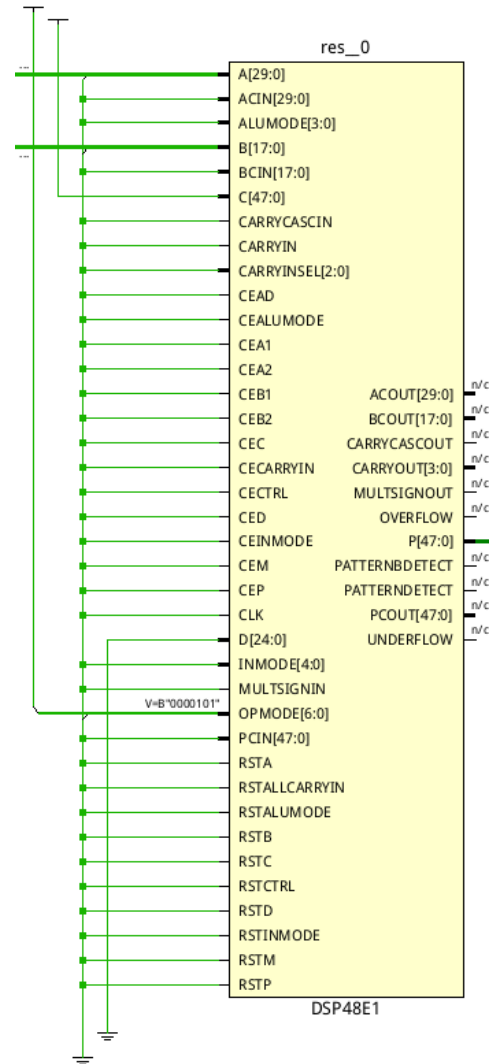
RefName	Used	Functional Category
LUT6	37	LUT
LUT2	25	LUT
LUT4	21	LUT
OBUF	16	IO
IBUF	16	IO
CARRY4	10	CarryLogic
LUT5	3	LUT
LUT3	3	LUT



03 АРИФМЕТИКА. УМНОЖЕНИЕ

```
module mult(
    input [15:0] a,
    input [15:0] b,
    output [31:0] res
);
    assign res = a * b;
endmodule
```

Resource	Utilization	Available	Utilization %
DSP	1	240	0.42
IO	64	210	30.48



03 АРИФМЕТИКА. MAC

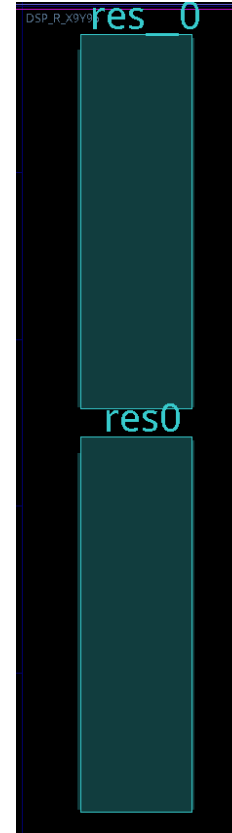
```

module s_mac(
    input signed [15:0] a0, a1,
    input signed [15:0] b0, b1,
    output signed [32:0] res
);

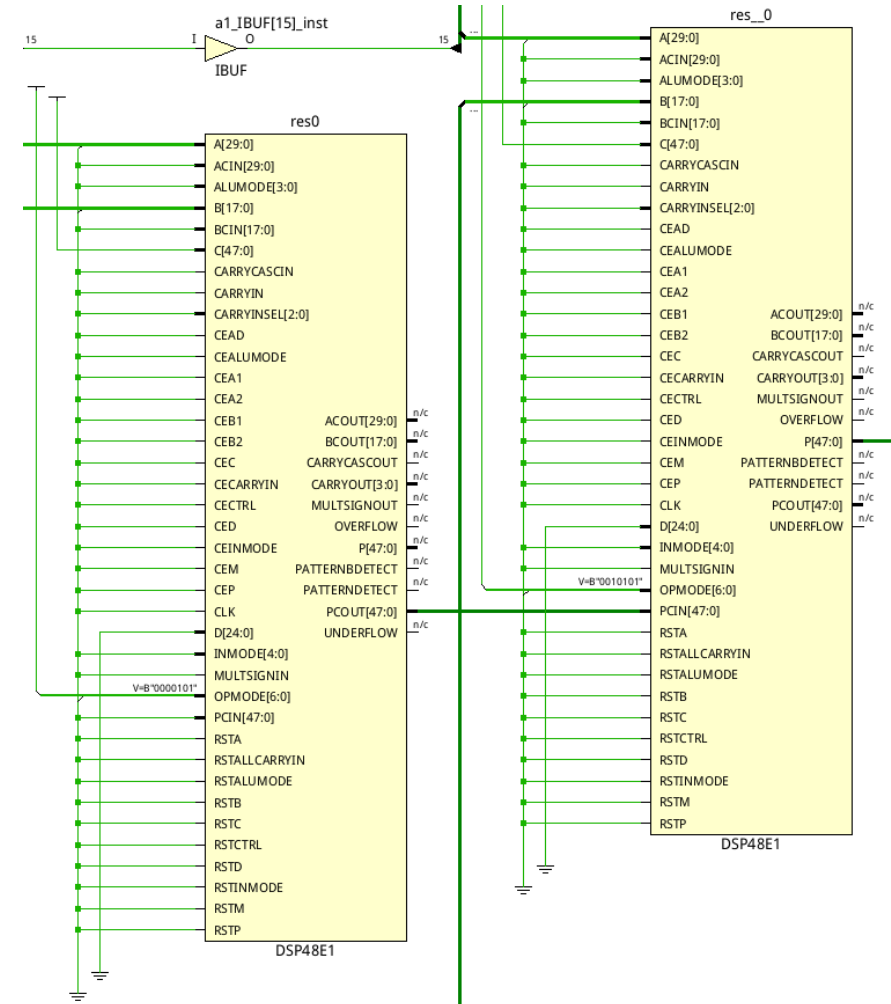
assign res = a0 * b0 + a1 * b1;

endmodule

```



Resource	Utilization	Available	Utilization %
DSP	2	240	0.83
IO	97	210	46.19



03

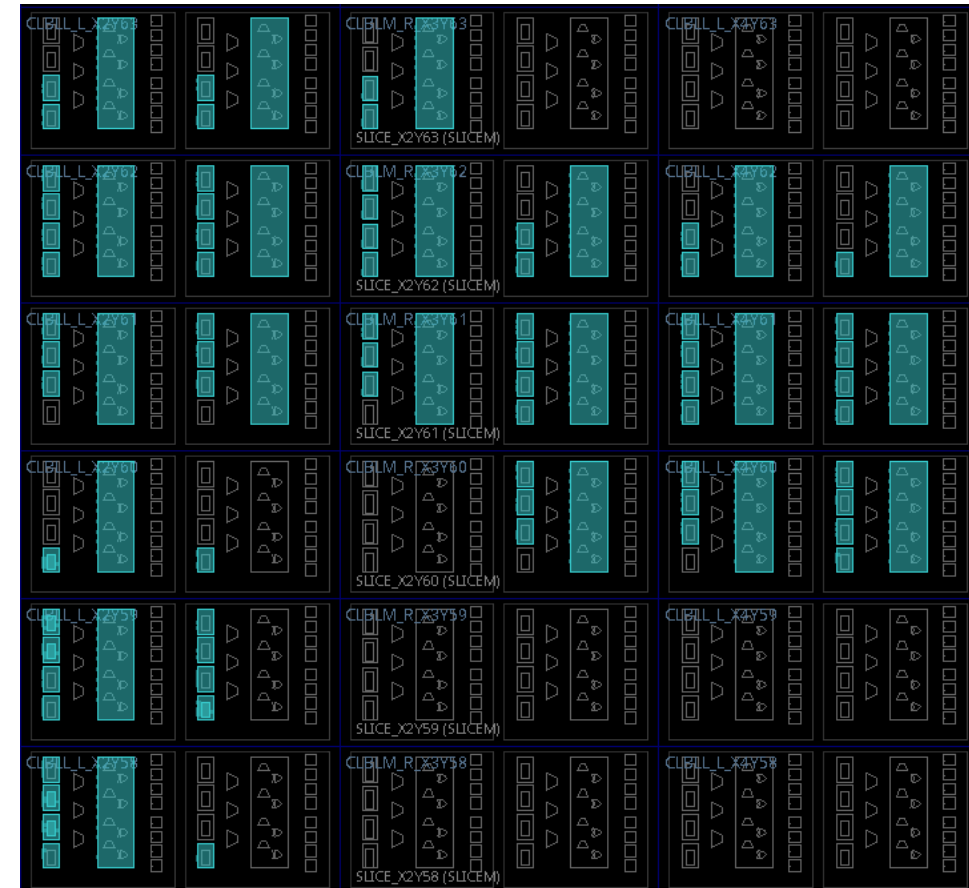
АРИФМЕТИКА. ДЕЛЕНИЕ

```
module div(
    input [7:0] a,
    input [7:0] b,
    output [7:0] res
);

assign res = a / b;

endmodule
```

RefName	Used	Functional Category
LUT3	49	LUT
CARRY4	21	CarryLogic
IBUF	16	IO
LUT2	13	LUT
OBUF	9	IO
LUT4	7	LUT
LUT6	3	LUT
LUT5	3	LUT



03

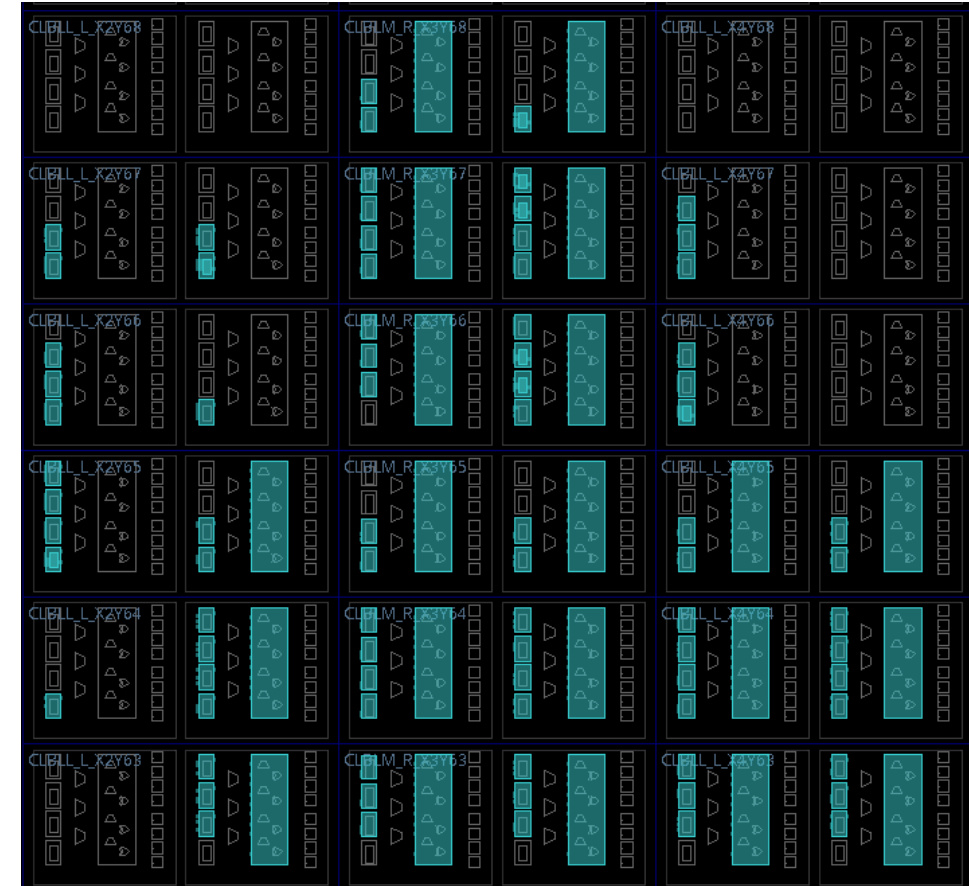
АРИФМЕТИКА. ВЗЯТИЕ ОСТАТКА

```
module mod (
    input [7:0] a,
    input [7:0] b,
    output [7:0] res
);

assign res = a % b;

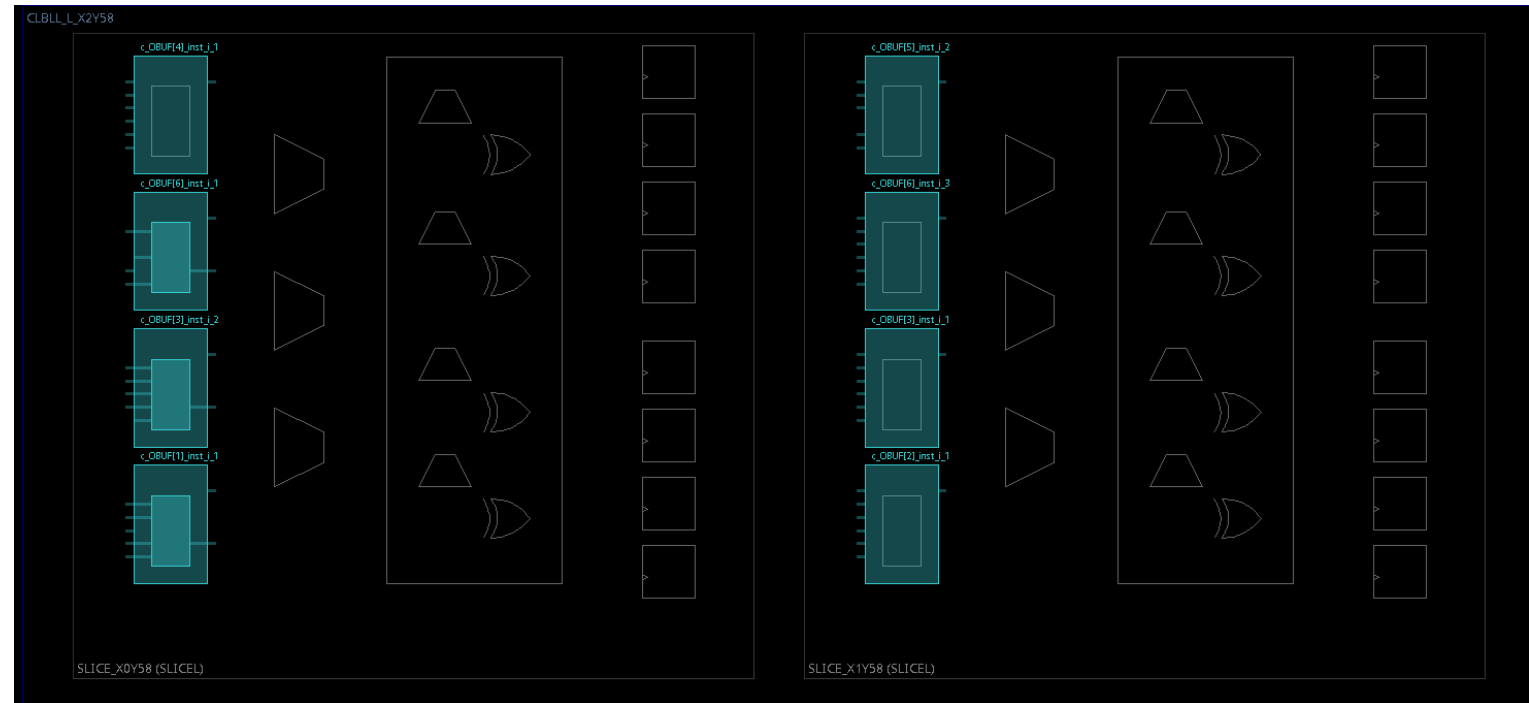
endmodule
```

Ref Name	Used	Functional Category
LUT3	49	LUT
CARRY4	21	CarryLogic
IBUF	16	IO
LUT2	13	LUT
OBUF	9	IO
LUT4	7	LUT
LUT6	3	LUT
LUT5	3	LUT



03 АРИФМЕТИКА. СДВИГ

```
module shift (  
    input [6:0] a,  
    input [2:0] b,  
    output [6:0] c  
);  
  
assign c = a << b;  
  
endmodule
```



Resource	Utilization	Available	Utilization %
LUT	8	63400	0.01
IO	17	210	8.10

03

АРИФМЕТИКА. АРИФМЕТИКА СО ЗНАКОМ

```
module s_add (  
    input signed [7:0] a,  
    input signed [7:0] b,  
    output signed [8:0] res  
);  
  
    assign res = a + b;  
  
endmodule
```

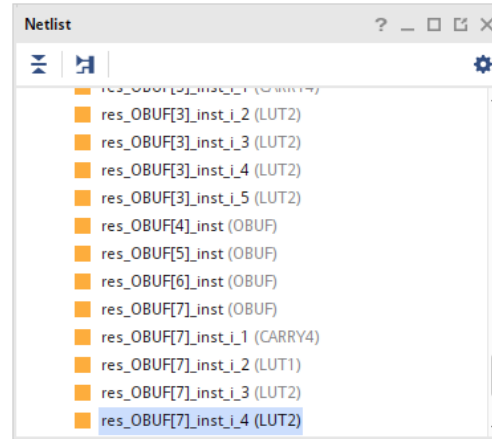
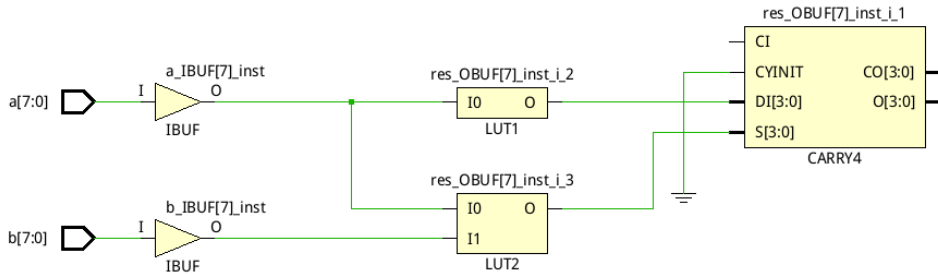
Name	Value	0.000000 us
> a[7:0]	11111011	11111011
> b[7:0]	00000010	00000010
> res[8:0]	11111101	11111101

Name	Value	0.000000 us
> a[7:0]	-5	-5
> b[7:0]	2	2
> res[8:0]	253	253

Name	Value	0.000000 us
> a[7:0]	-5	-5
> b[7:0]	2	2
> res[8:0]	-3	-3

03

АРИФМЕТИКА. АРИФМЕТИКА СО ЗНАКОМ

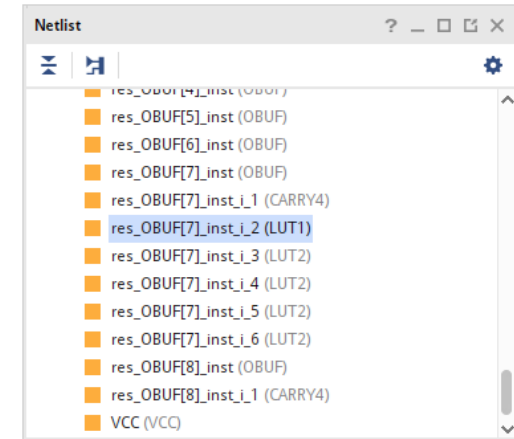


Cell Properties window for `res_OBUF[7]_inst_i_4`. The truth table is as follows:

I1	I0	O=I0 & I1 + !I0 & I1
0	0	0
0	1	1
1	0	1
1	1	0

Buttons: Edit LUT Equation...

Properties Power Nets Cell Pins Truth Table



Cell Properties window for `res_OBUF[7]_inst_i_2`. The truth table is as follows:

I0	O=I0
0	1
1	0

Buttons: Edit LUT Equation...

Properties Power Nets Cell Pins Truth Table

03

АРИФМЕТИКА.

АРИФМЕТИКА СО ЗНАКОМ

```
module s_add(  
    input signed [7:0] a,  
    input signed [7:0] b,  
    output signed [8:0] s_res,  
    output [8:0] u_res  
);  
  
assign s_res = a + b;  
assign u_res = a + b;  
  
endmodule
```

Name	Value	0.000000 us
> a[7:0]	11111011	11111011
> b[7:0]	00000010	00000010
> u_res[8:0]	11111101	11111101
> s_res[8:0]	11111101	11111101

Name	Value	0.000000 us
> a[7:0]	-5	-5
> b[7:0]	2	2
> u_res[8:0]	-3	-3
> s_res[8:0]	-3	-3

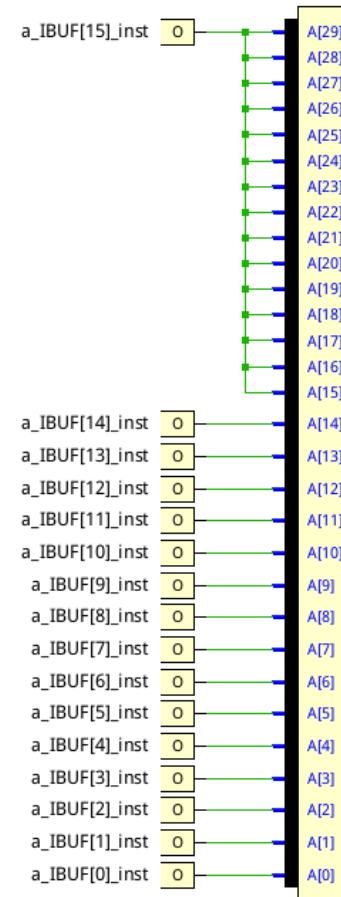
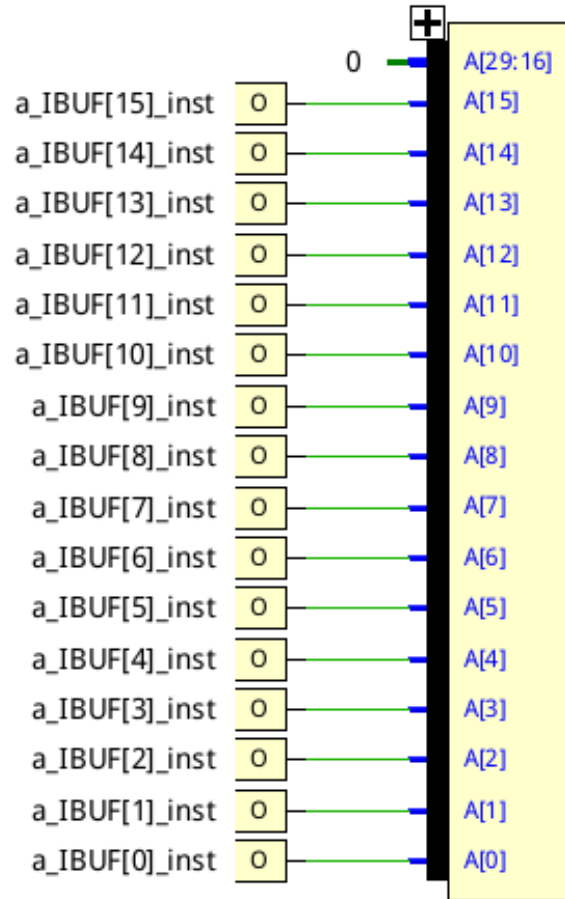
03

АРИФМЕТИКА. АРИФМЕТИКА СО ЗНАКОМ

```
module s_mult(  
    input signed [15:0] a,  
    input signed [15:0] b,  
    output signed [31:0] res  
);  
  
assign res = a * b;  
  
endmodule
```

03

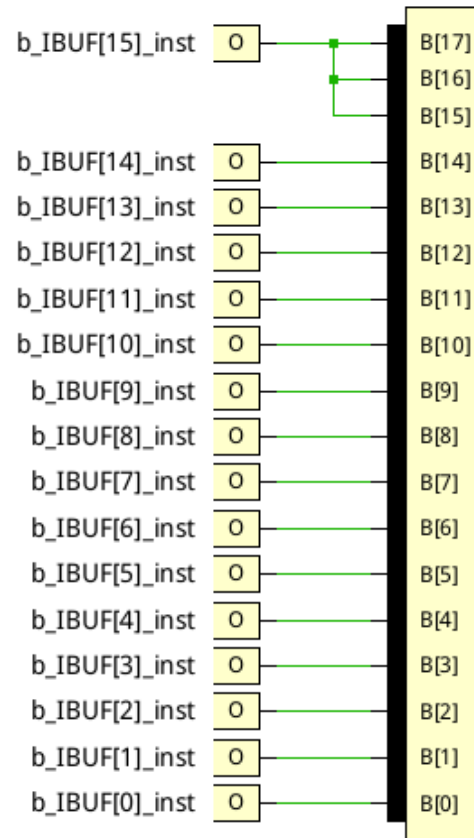
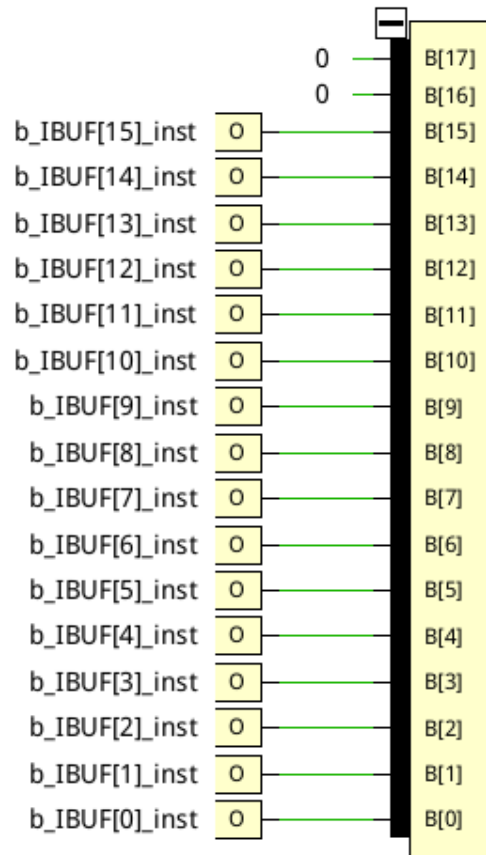
АРИФМЕТИКА. АРИФМЕТИКА СО ЗНАКОМ



03

АРИФМЕТИКА.

АРИФМЕТИКА СО ЗНАКОМ



03

АРИФМЕТИКА.

АРИФМЕТИКА СО ЗНАКОМ

```
module s_add_cin(  
    input signed [3:0] a,  
    input signed [3:0] b,  
    input cin,  
  
    output signed [4:0] res  
);  
  
assign res = a + b + cin;  
  
endmodule
```

Name	Value	0.000000 us
> a[3:0]	1011	1011
> b[3:0]	0010	0010
cin	1	
> res[4:0]	01110	01110

Name	Value	0.000000 us
> a[3:0]	-5	-5
> b[3:0]	2	2
cin	1	
> res[4:0]	14	14

03

АРИФМЕТИКА.

АРИФМЕТИКА СО ЗНАКОМ

```
module s_add_cin(  
    input signed [3:0] a,  
    input signed [3:0] b,  
    input cin,  
  
    output signed [4:0] res  
);  
  
assign res = a + b + $signed(cin);  
  
endmodule
```

Name	Value	0.000000 us
> a[3:0]	1011	1011
> b[3:0]	0010	0010
cin	1	
> res[4:0]	11100	11100

Name	Value	0.000000 us
> a[3:0]	-5	-5
> b[3:0]	2	2
cin	1	
> res[4:0]	-4	-4

03

АРИФМЕТИКА.

АРИФМЕТИКА СО ЗНАКОМ

```
module s_add_cin(  
    input signed [3:0] a,  
    input signed [3:0] b,  
    input cin,  
  
    output signed [4:0] res  
);  
  
assign res = a + b + $signed({1'b0, cin});  
  
endmodule
```

Name	Value	0.000000 us
> a[3:0]	1011	1011
> b[3:0]	0010	0010
cin	1	
> res[4:0]	11110	11110

Name	Value	0.000000 us
> a[3:0]	-5	-5
> b[3:0]	2	2
cin	1	
> res[4:0]	-2	-2

03

АРИФМЕТИКА.

АРИФМЕТИКА СО ЗНАКОМ

```
module s_add_part(  
    input signed [3:0] a,  
    input signed [3:0] b,  
    input signed [3:0] c,  
  
    output signed [6:0] res  
);  
  
assign res = a + b + c[3:0];  
  
endmodule
```

Name	Value	0.000000 us
> a[3:0]	1011	1011
> b[3:0]	1110	1110
> c[3:0]	1110	1110
> res[6:0]	0100111	0100111

Name	Value	0.000000 us
> a[3:0]	-5	-5
> b[3:0]	-2	-2
> c[3:0]	-2	-2
> res[6:0]	39	39

03

АРИФМЕТИКА.

АРИФМЕТИКА СО ЗНАКОМ

```
module s_add_part(  
    input signed [3:0] a,  
    input signed [3:0] b,  
    input signed [3:0] c,  
  
    output signed [6:0] res  
);  
  
assign res = a + b + $signed(c[3:0]);  
  
endmodule
```

Name	Value	0.000000 us
> a[3:0]	1011	1011
> b[3:0]	1110	1110
> c[3:0]	1110	1110
> res[6:0]	1110111	1110111

Name	Value	0.000000 us
> a[3:0]	-5	-5
> b[3:0]	-2	-2
> c[3:0]	-2	-2
> res[6:0]	-9	-9

03

АРИФМЕТИКА. СРАВНЕНИЕ ЧИСЕЛ

«>» — первый операнд больше второго;
«<» — первый операнд меньше второго;
«==» — первый операнд со вторым равны;
«>=» — первый операнд больше или равен второму;
«<=» — первый операнд меньше или равен второму;
«!=» — первый операнд не равен второму.

03

АРИФМЕТИКА. СРАВНЕНИЕ ЧИСЕЛ

```
module more(  
    input [3:0] a, b,  
    output c  
);  
  
assign c = a > b;  
  
endmodule
```

Resource	Utilization
LUT	2
IO	9

Ref Name	Used	Functional Category
IBUF	8	IO
OBUF	1	IO
LUT6	1	LUT
LUT3	1	LUT

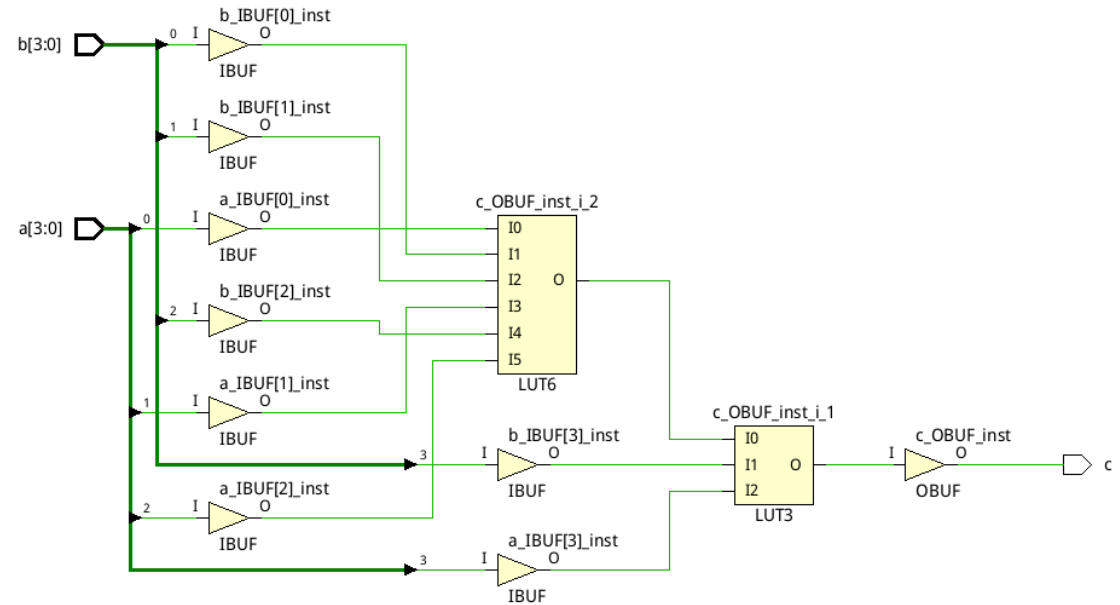
03

АРИФМЕТИКА. СРАВНЕНИЕ ЧИСЕЛ

```
module more(
    input [3:0] a, b,
    output c
);

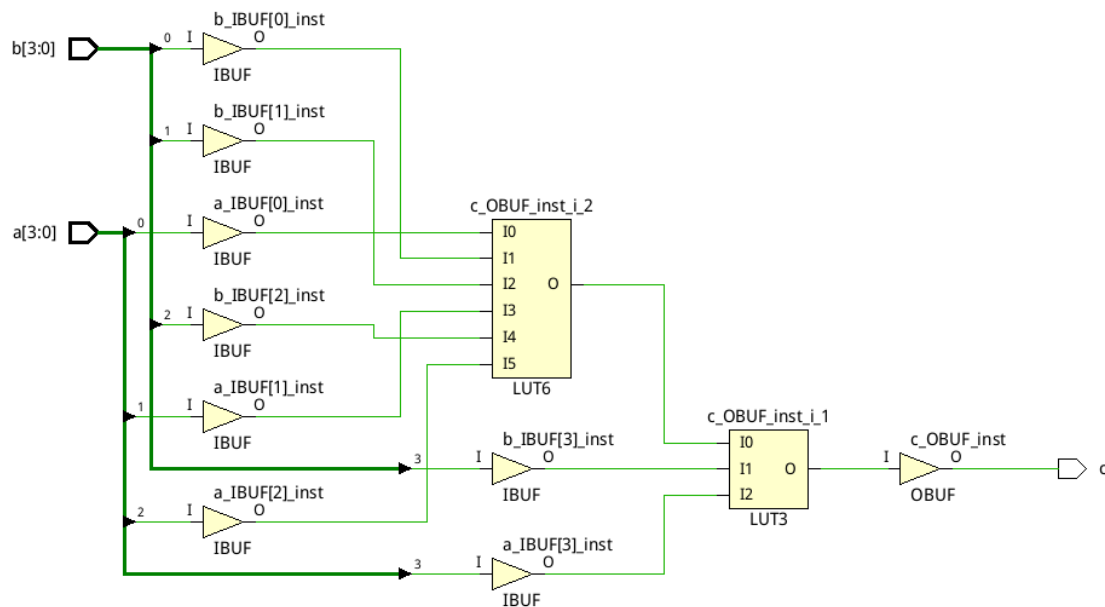
assign c = a > b;

endmodule
```



03

АРИФМЕТИКА. СРАВНЕНИЕ ЧИСЕЛ



Cell Properties

c_OBUF_inst_i_1

I2	I1	I0	O=I0 & I1 + I1 & I2 + I0 & I2
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

Cell Properties

c_OBUF_inst_i_2

I5	I4	I3	I2	I1	I0	O=I0 & I1 & I2
0	0	0	0	0	0	0
0	0	0	0	0	1	1
0	0	0	0	1	0	0
0	0	0	0	1	1	0
0	0	0	1	0	0	0
0	0	0	1	0	1	0
0	0	0	1	1	0	0
0	0	0	1	1	1	0
0	0	1	0	0	0	1
0	0	1	0	0	1	1
0	0	1	0	1	0	1
0	0	1	0	1	1	1
0	0	1	1	0	0	0
0	0	1	1	0	1	1
0	0	1	1	1	1	0
0	1	0	0	0	0	0
0	1	0	0	0	1	0

03

АРИФМЕТИКА. СРАВНЕНИЕ ЧИСЕЛ. УВЕЛИЧЕНИЕ РАЗРЯДНОСТИ

```
module more(  
    input [6:0] a, b,  
    output c  
);  
  
assign c = a > b;  
  
endmodule
```

Resource	Utilization
LUT	4
IO	15

Ref Name	Used	Functional Category
IBUF	14	IO
LUT4	6	LUT
LUT2	2	LUT
OBUF	1	IO
CARRY4	1	CarryLogic

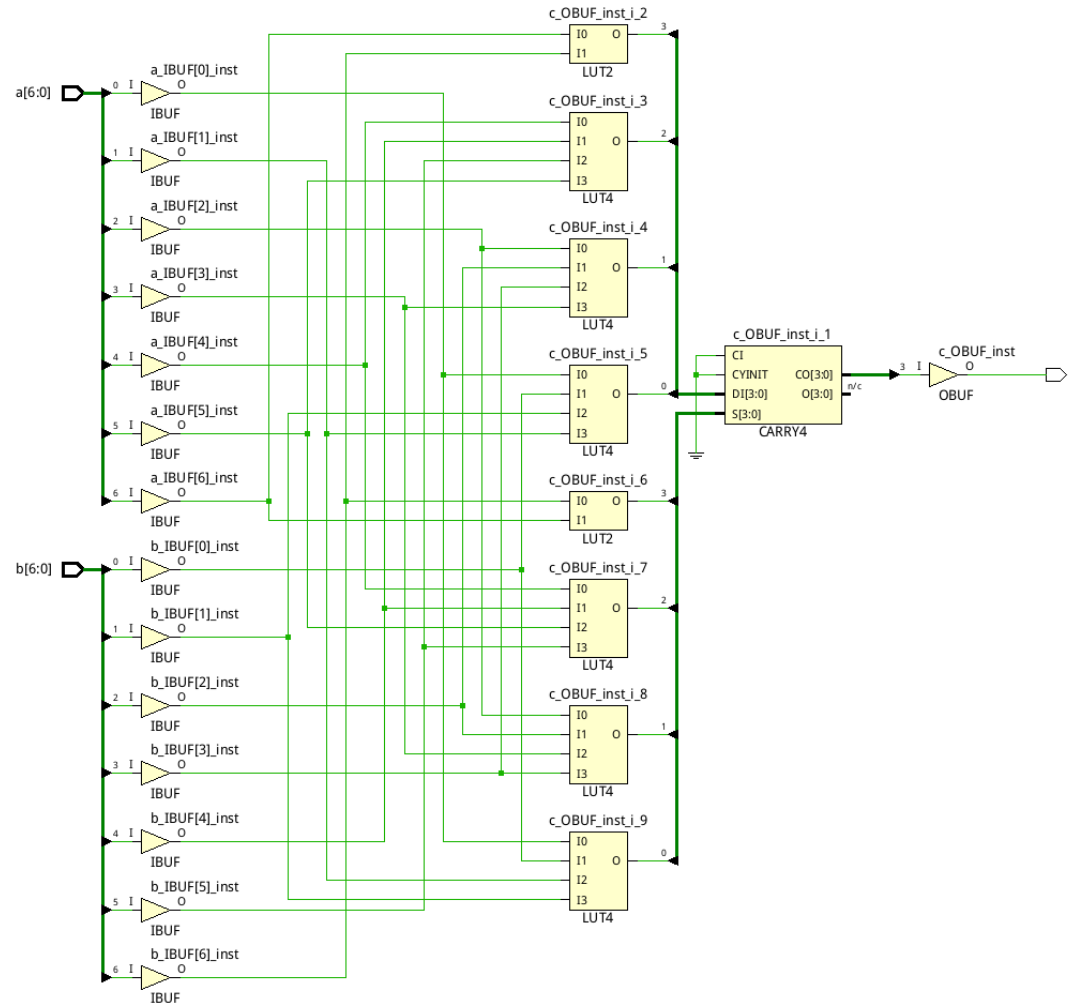
03

АРИФМЕТИКА. СРАВНЕНИЕ ЧИСЕЛ. УВЕЛИЧЕНИЕ РАЗРЯДНОСТИ

```
module more(
    input [6:0] a, b,
    output c
);

assign c = a > b;

endmodule
```



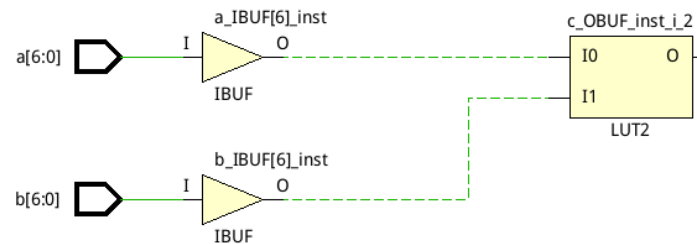
03

АРИФМЕТИКА. СРАВНЕНИЕ ЧИСЕЛ. СТАРШИЙ РАЗРЯД

```
module more(
    input [6:0] a, b,
    output c
);

assign c = a > b;

endmodule
```



Cell Properties

■ c_OBUF_inst_i_2

I1	I0	O=I0 & !I1
0	0	0
0	1	1
1	0	0
1	1	0

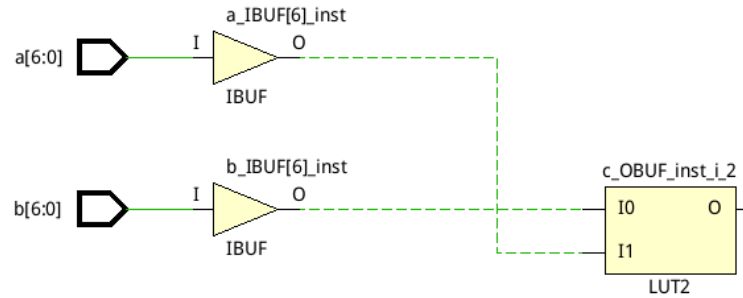
03

АРИФМЕТИКА. СРАВНЕНИЕ ЧИСЕЛ. СТАРШИЙ РАЗРЯД

```
module more(
    input [6:0] a, b,
    output c
);

assign c = a < b;

endmodule
```



Cell Properties

c_OBUF_inst_i_2

I1	I0	O=I0 & I1
0	0	0
0	1	1
1	0	0
1	1	0

03

АРИФМЕТИКА. СРАВНЕНИЕ ЧИСЕЛ. ПРОВЕРКА НА РАВЕНСТВО РАЗРЯДОВ ПРИ «>» И «<»

Cell Properties

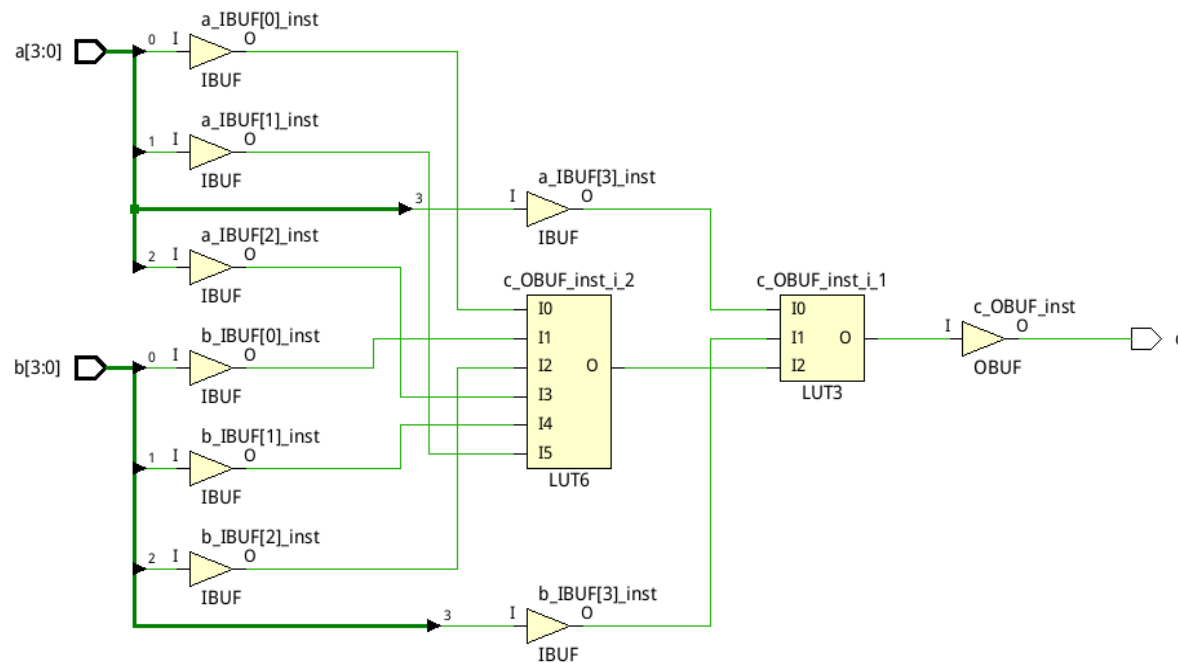
■ c_OBUF_inst_i_6

I1	I0	$O = I0 \& I1 + I0 \& I1$
0	0	1
0	1	0
1	0	0
1	1	1

03

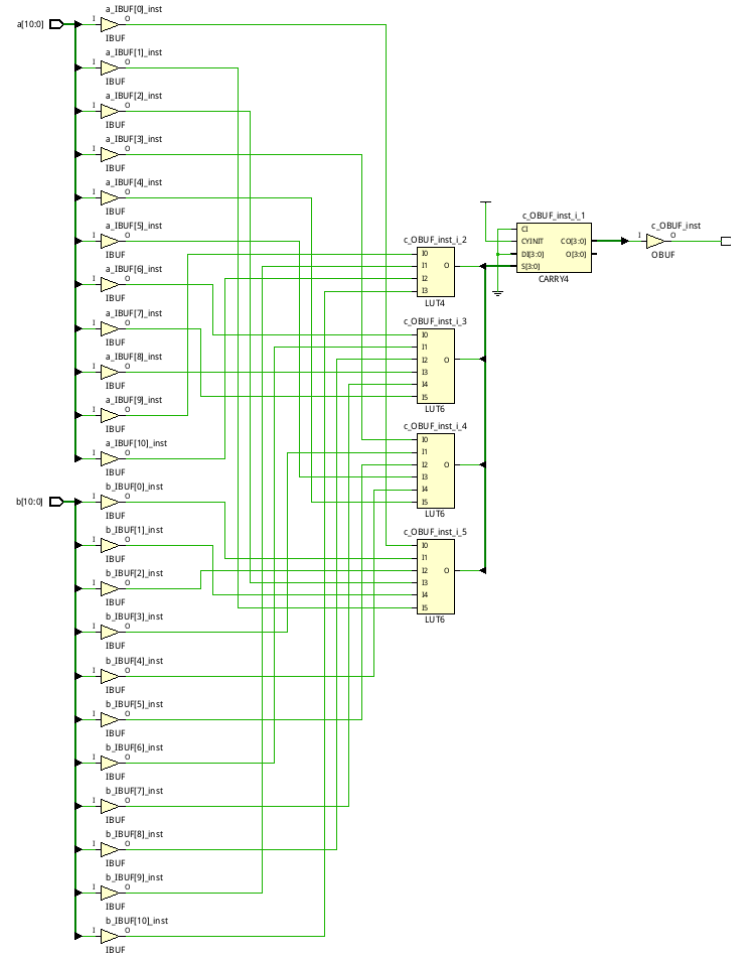
АРИФМЕТИКА. СРАВНЕНИЕ ЧИСЕЛ ЧЕТЫРЁХРАЗРЯДНЫЙ КОМПАРАТОР

«==»



03

АРИФМЕТИКА. СРАВНЕНИЕ ЧИСЕЛ 11-РАЗРЯДНЫЙ КОМПАРАТОР «==»



03

АРИФМЕТИКА. СРАВНЕНИЕ ЧИСЕЛ КОМПАРАТОР «==». ТАБЛИЦЫ ИСТИННОСТИ ПРИ РАЗНОЙ РАЗРЯДНОСТИ

Cell Properties

c_OBUF_inst_i_2

I5	I4	I3	I2	I1	I0	O=I0 & I1 & I2
0	0	0	0	0	0	1
0	0	0	0	0	1	0
0	0	0	0	1	0	0
0	0	0	0	1	1	1
0	0	0	1	0	0	0
0	0	0	1	0	1	0
0	0	0	1	1	0	0
0	0	0	1	1	1	0
0	0	1	0	0	0	0
0	0	1	0	0	1	0
0	0	1	0	1	0	0
0	0	1	0	1	1	0
0	0	1	1	0	0	1
0	0	1	1	0	1	0
0	0	1	1	1	0	0
0	0	1	1	1	1	1
0	1	0	0	0	0	0
0	1	0	0	0	1	0

Cell Properties

c_OBUF_inst_i_1

I2	I1	I0	O=I0 & I1 & I2 + I0 & I1 & I2
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Cell Properties

c_OBUF_inst_i_2

I5	I4	I3	I2	I1	I0	O=I0 & I1 & I2
0	0	0	0	0	0	1
0	0	0	0	0	1	0
0	0	0	0	1	0	0
0	0	0	0	1	1	1
0	0	0	1	0	0	0
0	0	0	1	0	1	0
0	0	0	1	1	0	0
0	0	0	1	1	1	0
0	0	1	0	0	0	0
0	0	1	0	0	1	0
0	0	1	0	1	0	0
0	0	1	0	1	1	0
0	0	1	1	0	0	1
0	0	1	1	0	1	0
0	0	1	1	1	0	0
0	0	1	1	1	1	1
0	1	0	0	0	0	0

Cell Properties

c_OBUF_inst_i_1

I3	I2	I1	I0	O=I0 & I1 & I2 & I3 + I0 & I1 & I2 & I3
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

03

АРИФМЕТИКА.

СРАВНЕНИЕ ЧИСЕЛ СО ЗНАКОМ

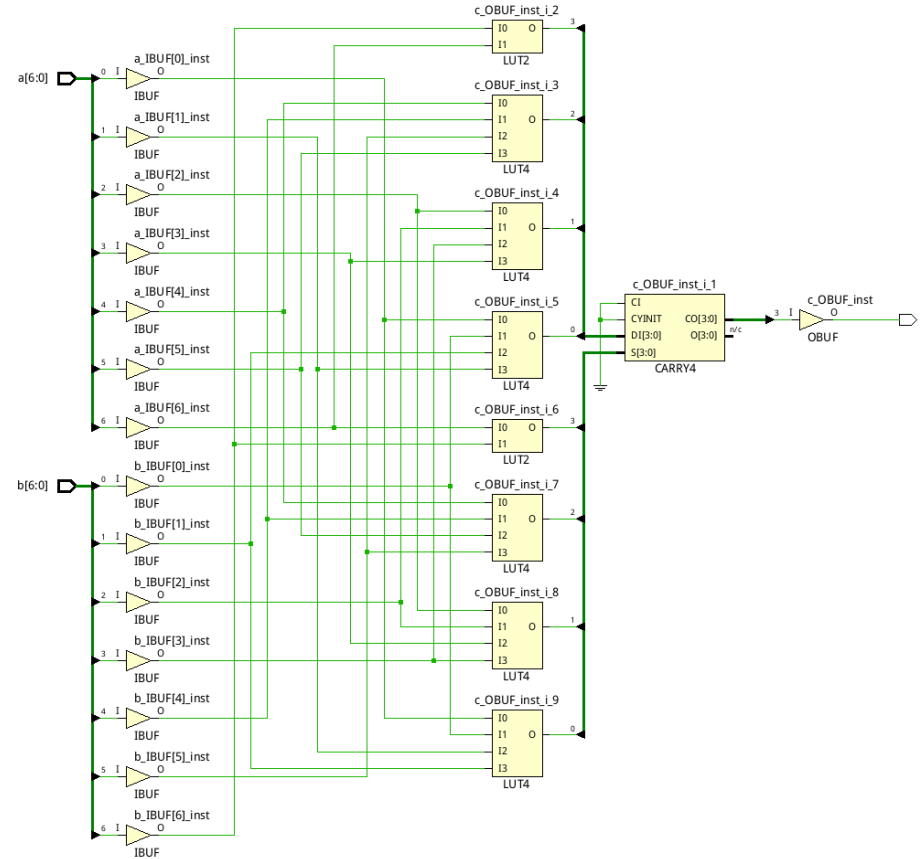
```

module more(
    input signed [6:0] a, b,
    output c
);

assign c = a > b;

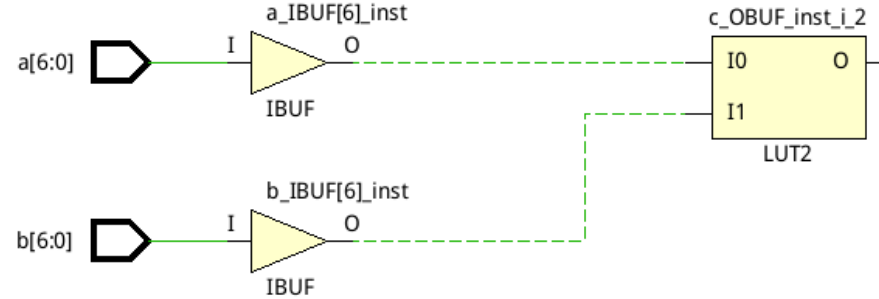
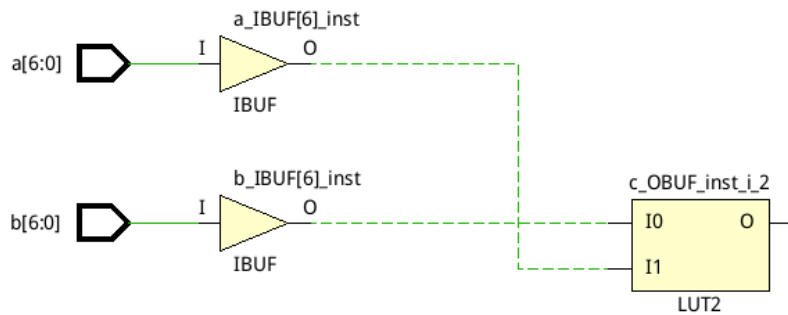
endmodule

```



03

АРИФМЕТИКА. СРАВНЕНИЕ ЧИСЕЛ СО ЗНАКОМ



Cell Properties		
c_OBUF_inst_i_2		
I1	I0	O=I0 & !I1
0	0	0
0	1	1
1	0	0
1	1	0

Знаковый

Беззнаковый

03

АРИФМЕТИКА.

НЕСИНТЕЗИРУЕМЫЕ КОНСТРУКЦИИ

```
module pow(  
    input [7:0] a,  
    input [1:0] b,  
    output [31:0] res  
);  
  
assign res = a ** b;  
  
endmodule
```

❗ [Synth 8-277] exponentiation is not supported [pow.v:11]

03

АРИФМЕТИКА. НЕСИНТЕЗИРУЕМЫЕ КОНСТРУКЦИИ

```
module div_real(  
    input [31:0] a, b,  
    output [31:0] res  
);  
real r;  
always@*  
    r = a / b;  
assign res = r;  
endmodule
```

❗ [Synth 8-27] real number expression not supported [div.v:33]

03

АРИФМЕТИКА. ПЛАВАЮЩАЯ ТОЧКА

Порядок

Знак

Неявная единица
(целая часть)

Мантисса
(дробная часть)

$$\pm 2^k (1 + f)$$

Смещённый
порядок

Исходный
порядок

Величина
смещения

$$k_{\text{смещ}} = k + b$$

03

АРИФМЕТИКА. ПЛАВАЮЩАЯ ТОЧКА

Знак	Порядок (в смещённом коде)	Мантисса (в прямом коде)	
1 бит	$k_{\text{смещ}}$ (m бит)	1,	f (n бит)

$$k_{\text{смещ}} = k + b$$

Смещённый порядок Исходный порядок Величина смещения

$$b = 2^{m-1} - 1$$

03

АРИФМЕТИКА. ПЛАВАЮЩАЯ ТОЧКА

Знак	Порядок (в смещённом коде)	Мантисса (в прямом коде)	
1 бит	5 бит	1,	10 бит

Знак	Порядок (в смещённом коде)	Мантисса (в прямом коде)	
1 бит	8 бит	1,	23 бита

Знак	Порядок (в смещённом коде)	Мантисса (в прямом коде)	
1 бит	11 бит	1,	52 бита

03 АРИФМЕТИКА. ПЛАВАЮЩАЯ ТОЧКА

Тип числа	Знак (Sign)	Порядок (Exponent)	Целая часть (неявная) (Implicit Leading Bit)	Мантисса (Significand)
Нормализованное число (Normal)	+ -	$0 < E < \max$	1	Любой набор битов
Ненормализованное число (Subnormal)	+ -	0	0	Любой ненулевой набор битов
Ноль	+ -	0	0	0
Бесконечность (∞ — Infinity)	+ -	Все 1 (max)	1	0
Не число (NaN — Not a Number)	+ -	Все 1 (max)	1	Любой ненулевой набор битов

03

АРИФМЕТИКА. ПЛАВАЮЩАЯ ТОЧКА

$$0,01011_2 = 1,011_2 * 2^{-2}$$

$$k_{\text{смещ}} = -2_{10} + 15_{10} = 13_{10} = 01101_2$$

Знак	Порядок (в смещённом коде)	Мантисса (в прямом коде)	
0	01101	1,	011000000000000000000000

03

АРИФМЕТИКА. ПЛАВАЮЩАЯ ТОЧКА

1. Выводить порядки чисел.
2. Произвести операцию суммирования или вычитания.

$$1,1101_2 * 2_{10}^{0_{10}} = 0,0111_2 * 2_{10}^{2_{10}}$$

03

АРИФМЕТИКА. ПЛАВАЮЩАЯ ТОЧКА. ИСКЛЮЧЕНИЯ ДЛЯ СЛОЖЕНИЯ

ОП2\ОП1	Не число	$+\infty$	$-\infty$	$+0$	-0	+Норм. число	-Норм. число
Не число	Не число	Не число	Не число	Не число	Не число	Не число	Не число
$+\infty$	Не число	$+\infty$	Не число	$+\infty$	$+\infty$	$+\infty$	$+\infty$
$-\infty$	Не число	Не число	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$
$+0$	Не число	$+\infty$	$-\infty$	$+0$	$+0$	+Норм. число	-Норм. число
-0	Не число	$+\infty$	$-\infty$	-0	-0	+Норм. число	-Норм. число
+Норм. число	Не число	$+\infty$	$-\infty$	+Норм. число	+Норм. число	+Норм. число или $+\infty$ в случае переполнения	\pm Норм. число или $+0$
-Норм. число	Не число	$+\infty$	$-\infty$	-Норм. число	-Норм. число	\pm Норм. число или $+0$	-Норм. число или $-\infty$ в случае переполнения

03

АРИФМЕТИКА. ПЛАВАЮЩАЯ ТОЧКА. ИСКЛЮЧЕНИЯ ДЛЯ ВЫЧИТАНИЯ

ОП2\ОП1	Не число	$+\infty$	$-\infty$	$+0$	-0	+Норм. число	-Норм. число
Не число	Не число	Не число	Не число	Не число	Не число	Не число	Не число
$+\infty$	Не число	Не число	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$
$-\infty$	Не число	$+\infty$	Не число	$+\infty$	$+\infty$	$+\infty$	$+\infty$
$+0$	Не число	$+\infty$	$-\infty$	$+0$	-0	+Норм. число	-Норм. число
-0	Не число	$+\infty$	$-\infty$	$+0$	$+0$	+Норм. число	-Норм. число
+Норм. число	Не число	$+\infty$	$-\infty$	-Норм. число	-Норм. число	\pm Норм. число или $+0$	-Норм. число или $-\infty$ в случае переполнения
-Норм. число	Не число	$+\infty$	$-\infty$	+Норм. число	+Норм. число	+Норм. число или $+\infty$ в случае переполнения	\pm Норм. число или $+0$

03

АРИФМЕТИКА. ПЛАВАЮЩАЯ ТОЧКА. УМНОЖЕНИЕ

При умножении нормализованных чисел порядки сомножителей складываются, а мантиссы перемножаются. Знак результата определяется путём сложения по модулю 2 знаков сомножителей. Результат подвергается операции нормализации.

03

АРИФМЕТИКА. ПЛАВАЮЩАЯ ТОЧКА. ИСКЛЮЧЕНИЯ ДЛЯ УМНОЖЕНИЯ

ОП2\ОП1	Не число	$+\infty$	$-\infty$	$+0$	-0	+Норм. число	-Норм. число
Не число	Не число	Не число	Не число	Не число	Не число	Не число	Не число
$+\infty$	Не число	$+\infty$	$-\infty$	Не число	Не число	$+\infty$	$-\infty$
$-\infty$	Не число	$-\infty$	$+\infty$	Не число	Не число	$-\infty$	$+\infty$
$+0$	Не число	Не число	Не число	$+0$	-0	$+0$	-0
-0	Не число	Не число	Не число	-0	$+0$	-0	$+0$
+Норм. число	Не число	$+\infty$	$-\infty$	$+0$	-0	+Норм. число или $+\infty$ (переполнение) или $+0$ (потеря значимости)	-Норм. число или $+\infty$ (переполнение) или -0 (потеря значимости)
-Норм. число	Не число	$-\infty$	$+\infty$	-0	$+0$	-Норм. число или $-\infty$ (переполнение) или -0 (потеря значимости)	+Норм. число или $+\infty$ (переполнение) или $+0$ (потеря значимости)

03

АРИФМЕТИКА. ПЛАВАЮЩАЯ ТОЧКА. ДЕЛЕНИЕ

При делении нормализованных чисел порядок делителя вычитается из порядка делимого, а мантиссы делимого делится на мантиссу делителя. Знак результата определяется путём сложения по модулю 2 знаков сомножителей. Результат подвергается операции нормализации.

03

АРИФМЕТИКА. ПЛАВАЮЩАЯ ТОЧКА. ИСКЛЮЧЕНИЯ ДЛЯ ДЕЛЕНИЯ

ОП2\ОП1	Не число	$+\infty$	$-\infty$	$+0$	-0	+Норм. число	-Норм. число
Не число	Не число	Не число	Не число	Не число	Не число	Не число	Не число
$+\infty$	Не число	Не число	Не число	$+0$	-0	$+0$	-0
$-\infty$	Не число	Не число	Не число	-0	$+0$	-0	$+0$
$+0$	Не число	$+\infty$	$-\infty$	Не число	Не число	$+\infty$	$-\infty$
-0	Не число	$-\infty$	$+\infty$	Не число	Не число	$-\infty$	$+\infty$
+Норм. число	Не число	$+\infty$	$-\infty$	$+0$	-0	+Норм. число или $+\infty$ (переполнение) или $+0$ (потеря значимости)	-Норм. число или $-\infty$ (переполнение) или -0 (потеря значимости)
-Норм. число	Не число	$-\infty$	$+\infty$	-0	$+0$	-Норм. число или $-\infty$ (переполнение) или -0 (потеря значимости)	+Норм. число или $+\infty$ (переполнение) или $+0$ (потеря значимости)

03

АРИФМЕТИКА. ПЛАВАЮЩАЯ ТОЧКА. СТАНДАРТЫ

IEEE Standard 754 for Binary Floating-Point Arithmetic (1985)

Recognized as an American National Standard (ANSI)

IEEE Std 754-1985

An American National Standard

IEEE Standard for Binary Floating-Point Arithmetic

Sponsor
**Standards Committee
of the
IEEE Computer Society**

Approved March 21, 1985
Reaffirmed December 6, 1990

IEEE Standards Board

Approved July 26, 1985
Reaffirmed May 21, 1991

American National Standards Institute

IEEE Standard for Floating-Point Arithmetic (2008)



IEEE Standard for Floating-Point Arithmetic

IEEE Computer Society

Sponsored by the
Microprocessor Standards Committee

04

ЗАДАНИЕ



ПОСТАНОВКА ЗАДАЧИ

04 ЗАДАНИЕ НА ПРАКТИЧЕСКУЮ РАБОТУ.

1. Создать проект в САПР Vivado для ПЛИС Artix-7 xc7a100tcsg324-1.
2. Повторить рассмотренные в течение занятия примеры.
3. Проанализировать результат, сравнив его с тем, что продемонстрировано в примерах.
4. Согласно выданному варианту:
 1. Создать файл для модуля верхнего уровня с именем main.
 2. В файле описать схему согласно выданному варианту.
 3. Произвести синтез и имплементацию описанного устройства.
 4. Проанализировать задействованные для устройства аппаратные ресурсы.
 5. Произвести серию изменений размеров входных и выходных шин, основываясь на теоретическом введении.
 6. Проанализировать результат синтеза и размещения каждого из вариантов.
 7. Произвести серию изменений правил выполнения арифметических операций, указывая, что операнды/результат знаковые.
 8. Проанализировать результат синтеза и размещения каждого из вариантов.
5. Составить отчёт.

ВОПРОСЫ



СПАСИБО ЗА ВНИМАНИЕ!