

# Цифровые схемы

$$F: \{0, 1\}^N \rightarrow \{0, 1\}^M$$

Комбинационная  
логика

$$\begin{pmatrix} \text{out}^1(t_r) \\ \text{out}^2(t_r) \\ \text{out}^3(t_r) \\ \dots \\ \text{out}^m(t_r) \end{pmatrix} = F \begin{pmatrix} \text{in}^1(t_r) \\ \text{in}^2(t_r) \\ \text{in}^3(t_r) \\ \dots \\ \text{in}^n(t_r) \end{pmatrix}$$

Последовательностная  
логика

$$\begin{pmatrix} \text{out}^1(t_r) \\ \text{out}^2(t_r) \\ \text{out}^3(t_r) \\ \dots \\ \text{out}^m(t_r) \end{pmatrix} = F \begin{pmatrix} \text{in}^1(t_r) \\ \dots \\ \text{in}^n(t_r) \\ \text{in}^1(t_{r-1}) \\ \dots \\ \text{in}^k(t_{r-1}) \end{pmatrix}$$

# Логический элемент AND

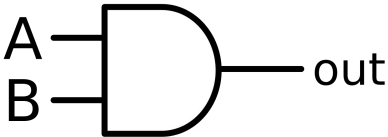
В языке Verilog	На схеме
$\text{OUT} = A \ \& \ B$	

Таблица истинности		
A	B	OUT
0	0	
0	1	
1	0	
1	1	

# Логический элемент XOR

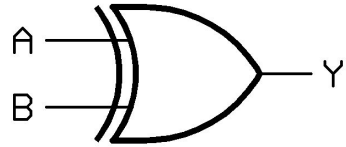
В языке Verilog	На схеме
$\text{OUT} = A \oplus B$	

Таблица истинности		
A	B	OUT
0	0	
0	1	
1	0	
1	1	

# Однобитный сумматор

# Однобитный сумматор

```
module adder(  
    input a,  
    input b,  
    input carry_in,  
    output sum,  
    output carry_out  
);  
assign sum =
```

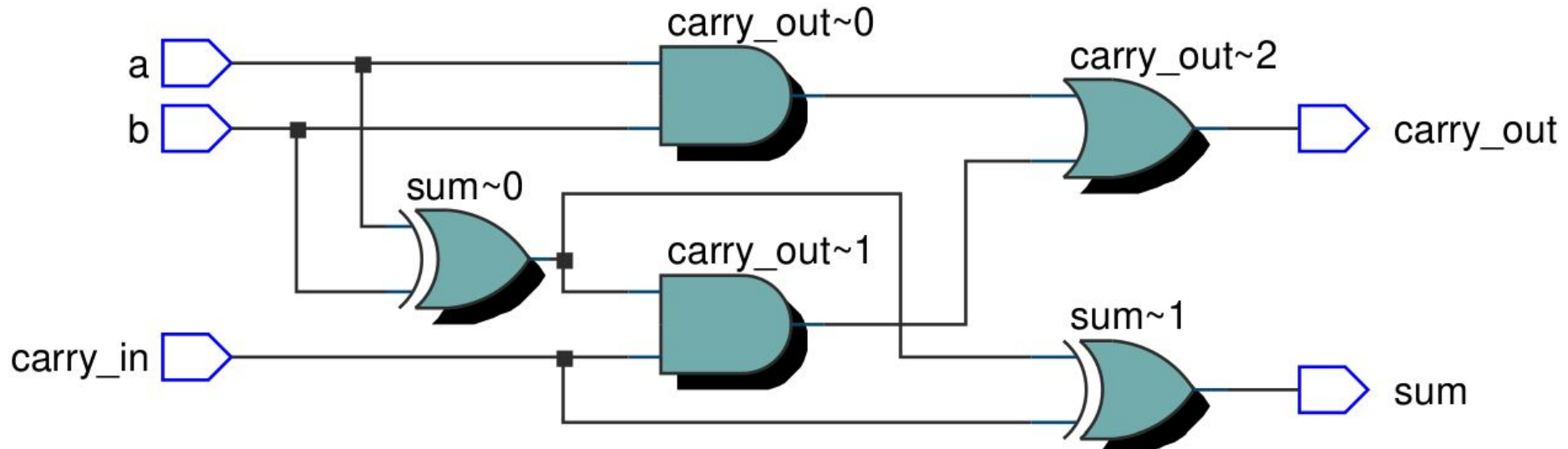
# Однобитный сумматор

```
module adder(  
    input a,  
    input b,  
    input carry_in,  
    output sum,  
    output carry_out  
);  
assign sum = (a ^ b) ^ carry_in;  
assign carry_out =
```

# Однобитный сумматор

```
module adder(  
    input a,  
    input b,  
    input carry_in,  
    output sum,  
    output carry_out  
);  
assign sum = (a ^ b) ^ carry_in;  
assign carry_out = (a & b) | ((a ^ b) & carry_in);  
endmodule
```

# Результат синтеза однобитного сумматора





# Восьмибитный сумматор

```
module adder8(  
    input [7:0]a,  
    input [7:0]b,  
    output [7:0]sum  
);  
adder adder0(.a(a[0]), .b(b[0]), .carry_in(0), .sum(sum[0]), .carry_out(c0));  
adder adder1(.a(a[1]), .b(b[1]), .carry_in(c0), .sum(sum[1]), .carry_out(c1));  
  
.....  
adder adder6(.a(a[6]), .b(b[6]), .carry_in(c5), .sum(sum[6]), .carry_out(c6));  
adder adder7(.a(a[7]), .b(b[7]), .carry_in(c6), .sum(sum[7]));  
endmodule
```

# Восьмибитный сумматор

```
module top();  
  reg [7:0]a;  
  reg [7:0]b;  
  wire [7:0]s;  
  adder8 adder8_inst(.a(a), .b(b), .sum(s));  
  initial begin  
    a = 35; b = 62; #1 $display("a =%d, b =%d, sum =%d", a, b, s);  
    a = 19; b = 14; #1 $display("a =%d, b =%d, sum =%d", a, b, s);  
  end  
endmodule
```

```
~/fpga-intro $ iverilog adder.v; ./a.out
```

```
a = 35, b = 62, sum = 97
```

```
a = 19, b = 14, sum = 33
```