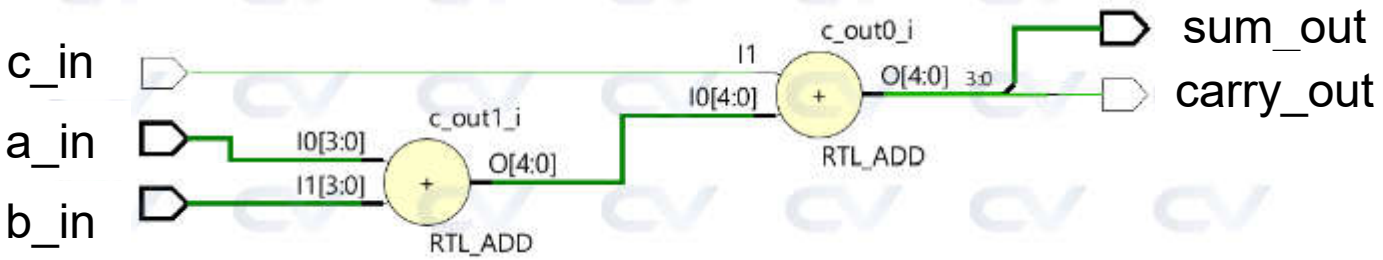
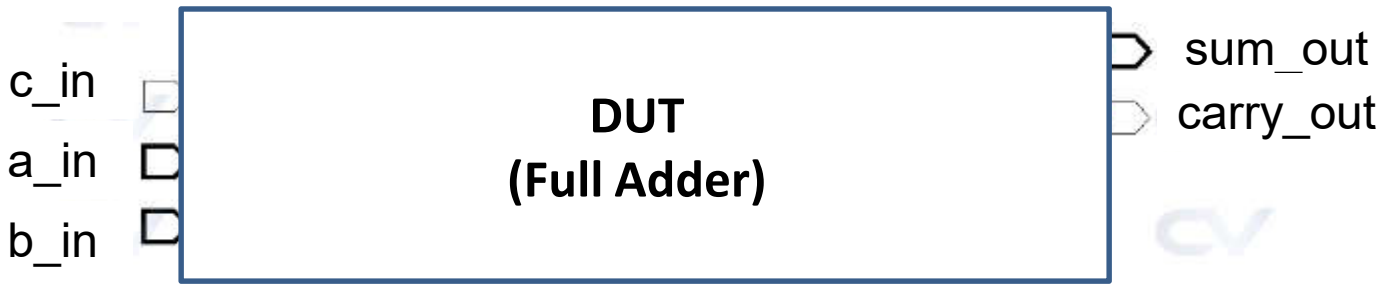
Design and Testbench

Tuan Nguyen-viet

Full Adder

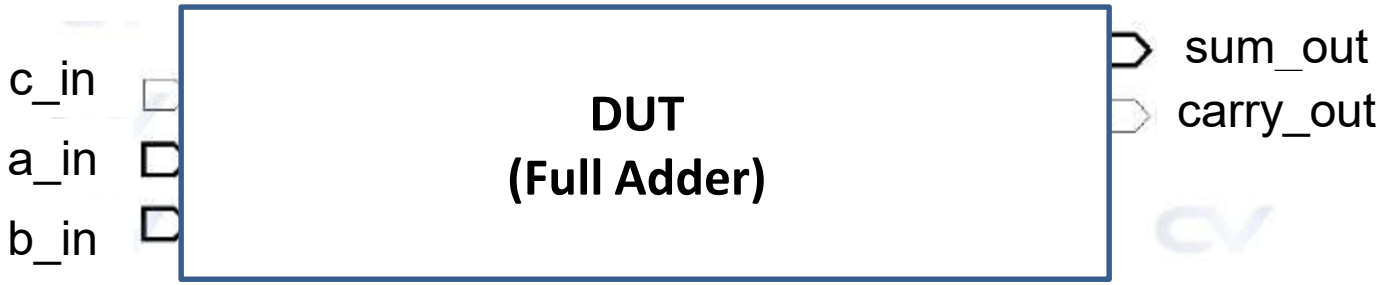

c_in
a_in
b_in
I0[3:0]
I1[3:0]
c_out1_i
O[4:0]
RTL_ADD
I1
I0[4:0]
c_out0_i
O[4:0]
3:0
RTL_ADD
sum_out
carry_out

Full Adder (2)

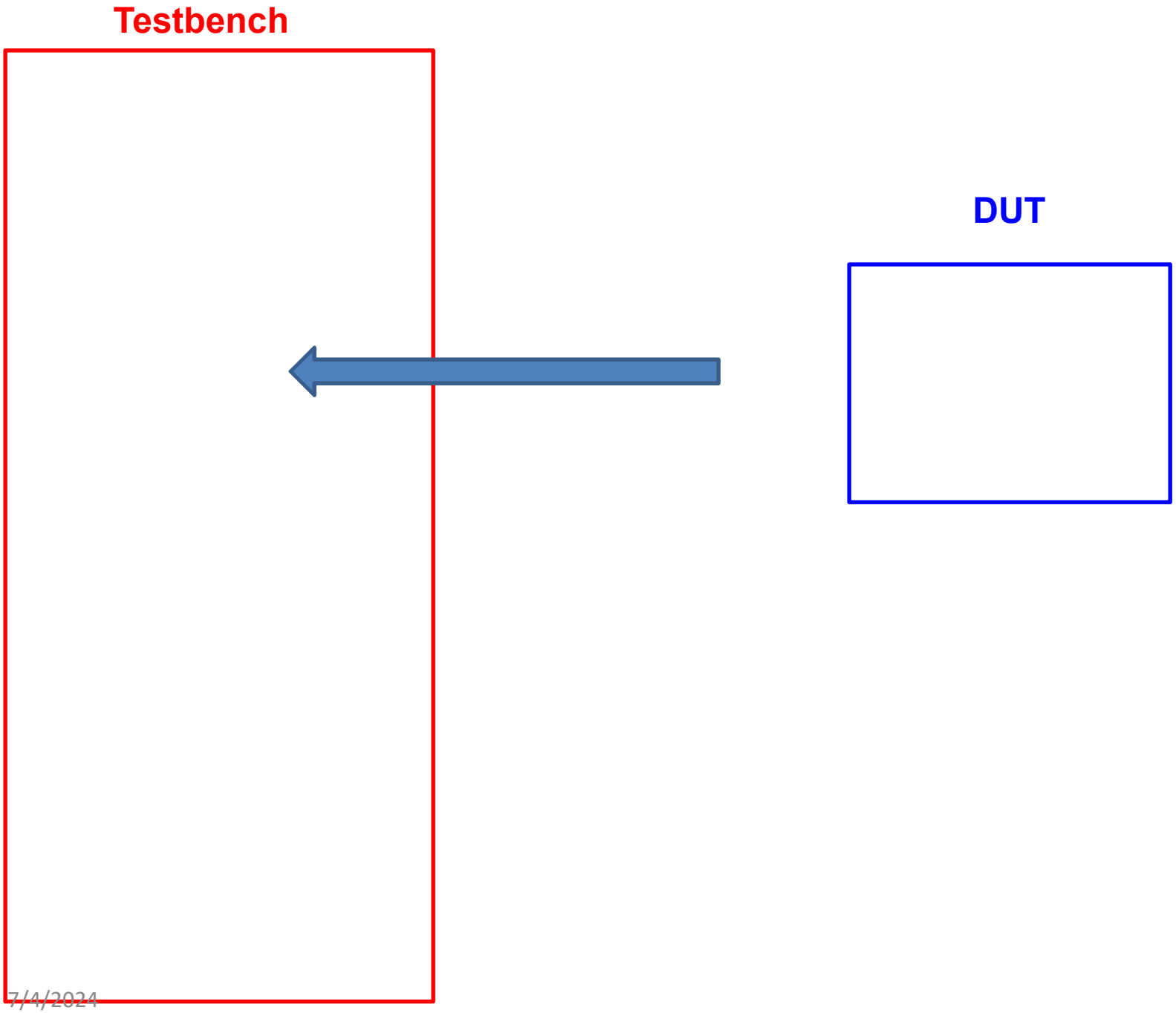


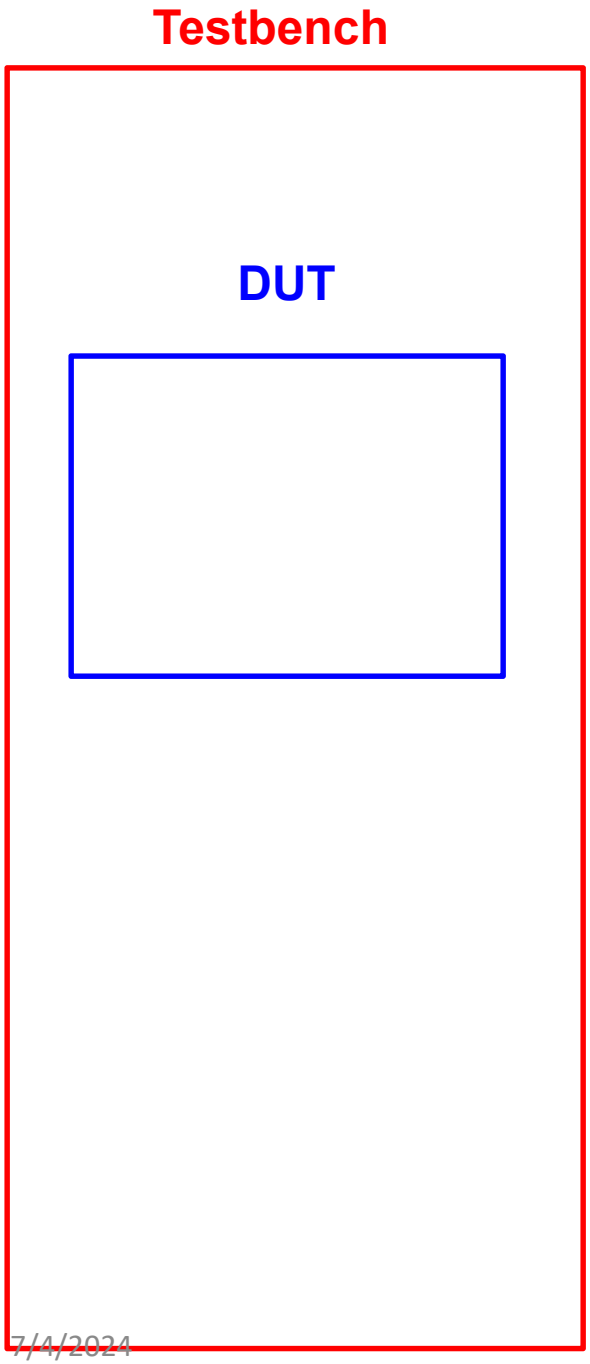

```
module full_adder (a_in, b_in, c_in, c_out, sum_out);

input [3:0] a_in, b_in;
input c_in;
output carry_out;
output [3:0] sum_out;

reg carry_out;
reg [3:0] sum_out;

always @ (a_in or b_in or c_in)
    begin
        {carry_out, sum_out} = a_in + b_in + c_in;
    end

endmodule
```

Full Adder (3)



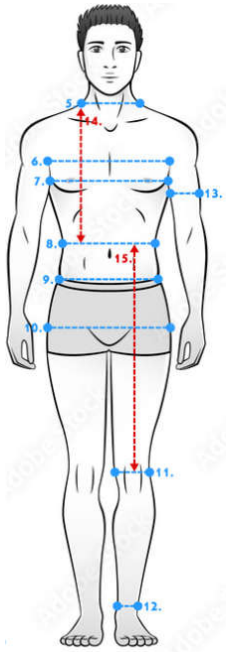

```
full_adder dut (
                .a_in(   ),
                .b_in(   ),
                .c_in(   ),
                .carry_out(   ),
                .sum_out(   )
               );
```


Testbench
DUT

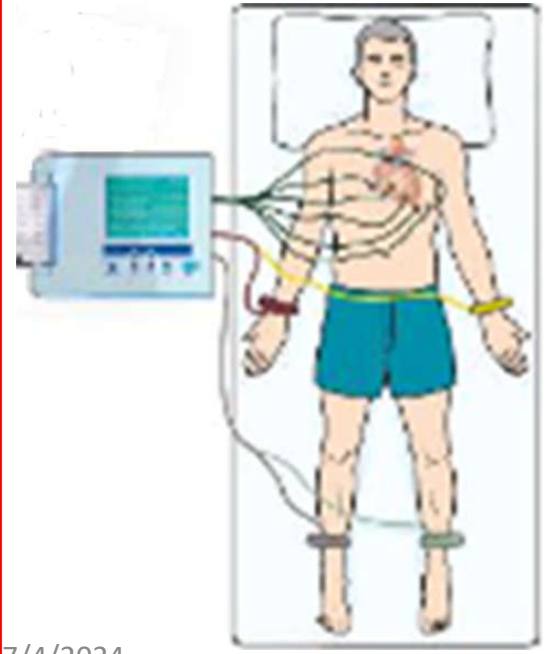

Testbench
DUT

Testbench

DUT



Testbench



```
`timescale 1 ns/10 ps

module full_adder_test; // 4-bit

reg [3:0] a, b;
reg c;
wire [3:0] sum;
wire carry;
```

```
full_adder dut (
                .a_in(   ),
                .b_in(   ),
                .c_in(   ),
                .carry_out(   ),
                .sum_out(   )
                );

initial
    begin
        a = 4'b0000;
        b = 4'b0000;
        c = 1'b0;
        #50;
        a = 4'b0101;
        b = 4'b1010;
        // => sum=1111, carry=0
        #50;
        a = 4'b1111;
        b = 4'b0001;
        // => sum=0000, carry=1
        #50;
        a = 4'b0000;
        b = 4'b1111;
        c = 1'b1;
        // => sum=0000, carry=1
        #50;
        a = 4'b0110;
        b = 4'b0001;
        // => sum=1000, carry=0
    end
endmodule
```

```
`timescale 1 ns/10 ps

module full_adder_test; // 4-bit

reg [3:0] a, b;
reg c;
wire [3:0] sum;
wire carry;

full_adder dut (
                .a_in(   ),
                .b_in(   ),
                .c_in(   ),
                .carry_out(   ),
                .sum_out(   )
                );
initial
    begin
        a = 4'b0000;
        b = 4'b0000;
        c = 1'b0;
        #50;
        a = 4'b0101;
        b = 4'b1010;
        // => sum=1111, carry=0
        #50;
        a = 4'b1111;
        b = 4'b0001;
        // => sum=0000, carry=1
        #50;
        a = 4'b0000;
        b = 4'b1111;
        c = 1'b1;
        // => sum=0000, carry=1
        #50;
        a = 4'b0110;
        b = 4'b0001;
        // => sum=1000, carry=0
    end
endmodule
```

```
`timescale 1 ns/10 ps

module full_adder_test; // 4-bit

reg [3:0] a, b;
reg c;
wire [3:0] sum;
wire carry;

full_adder dut (
                .a_in(a),
                .b_in(b),
                .c_in(c),
                .carry_out(carry),
                .sum_out(sum)
                );
initial
    begin
        a = 4'b0000;
        b = 4'b0000;
        c = 1'b0;
        #50;
        a = 4'b0101;
        b = 4'b1010;
        // => sum=1111, carry=0
        #50;
        a = 4'b1111;
        b = 4'b0001;
        // => sum=0000, carry=1
        #50;
        a = 4'b0000;
        b = 4'b1111;
        c = 1'b1;
        // => sum=0000, carry=1
        #50;
        a = 4'b0110;
        b = 4'b0001;
        // => sum=1000, carry=0
    end
endmodule
```

Thank You