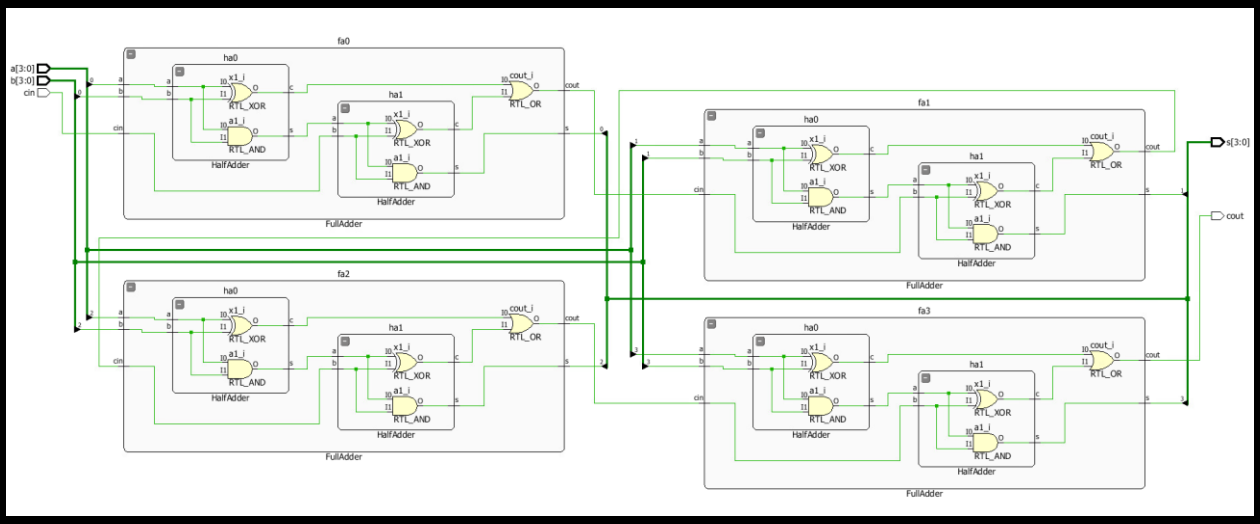
Ripple Adder

RippleCarry4.v

```
`timescale 1ns / 1ps

module HalfAdder(
    input a,
    input b,
    output c,
    output s
    );
    xor x1(s, a, b);
    and a1(c, a, b);
endmodule

module FullAdder(
    input a,
    input b,
    input cin,
    output s,
    output cout
    );
    wire carryGenerate, carryPropagate, sum1;
    HalfAdder ha0(.a(a), .b(b), .c(carryGenerate), .s(sum1));
    HalfAdder ha1(.a(sum1), .b(cin), .c(carryPropagate), .s(s));
    or o1(cout, carryGenerate, carryPropagate);
endmodule

module RippleCarry4(
```

```
    input cin,
    input [3:0] a,
    input [3:0] b,
    output [3:0] s,
    output cout
    );
    wire cout0, cout1, cout2;
    FullAdder fa0(a[0], b[0], cin, s[0], cout0);
    FullAdder fa1(a[1], b[1], cout0, s[1], cout1);
    FullAdder fa2(a[2], b[2], cout1, s[2], cout2);
    FullAdder fa3(a[3], b[3], cout2, s[3], cout);
Endmodule
```

Schematic

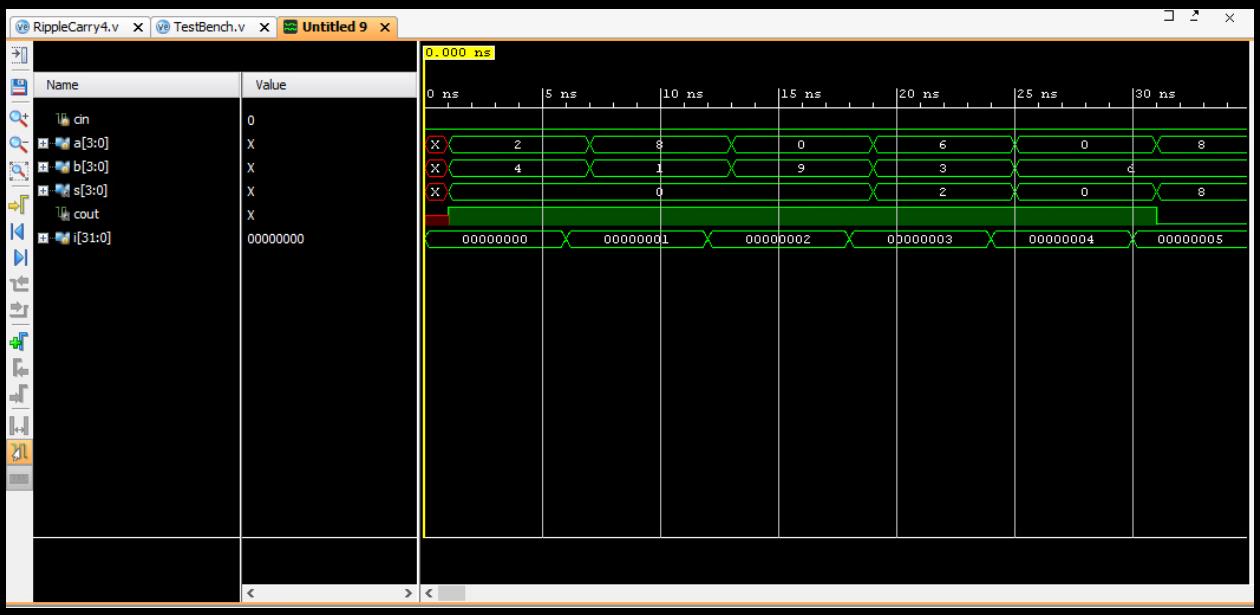


TestBench

```
`timescale 1ns / 1ps
module TestBench();
    reg cin;
    reg [3:0] a, b;
    wire [3:0] s;
    wire cout;
```

```
    RippleCarry4 rut(cin, a, b, s, cout);


    integer i;
    initial
    begin
        $display ("4-bit Ripple Carry");
        cin = 0;
        for (i = 0; i < 8; i = i+1)
            begin
                #1 {a, b} = $random;
                #5 $display ("a = %d, b = %d, s= %d, cout = %b", a, b, s, cout);
            end
    end
endmodule
```

Timing Diagram



Youtube link:

https://youtu.be/eHOL4aypPaw