EPWave

Languages & Libraries

Tools & Simulators (http://eda-playground.readthedocs.org/en/latest/intro.html#tools-simulators)

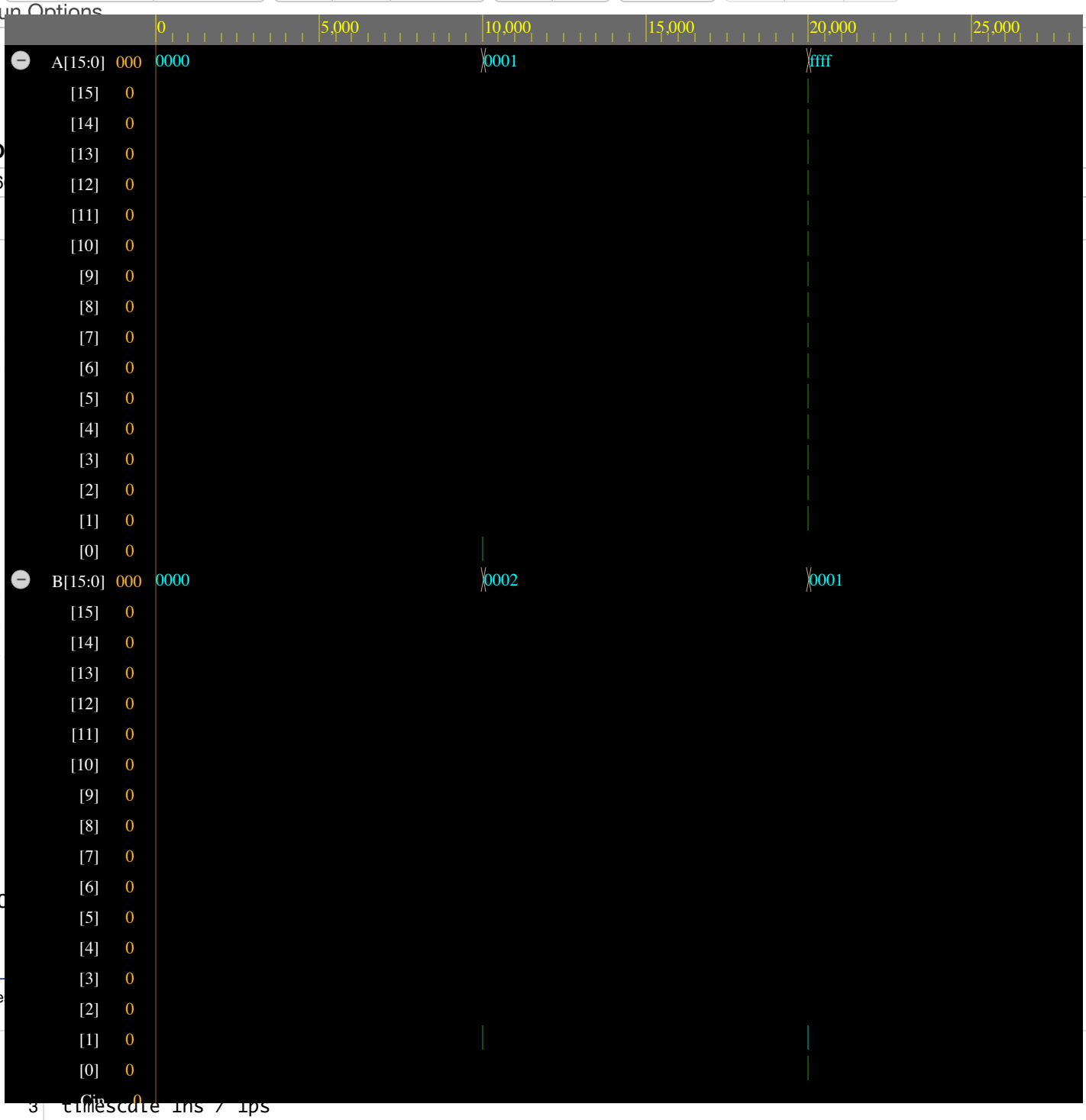
Icarus Verilog 12.0

Compile Options

-Wall -g2012

Run Options

Get Signals | Radix | 100% | 0ps




```
3 `timescale 1ns / 1ps
```

EPWave

```
4
5  module tb_PPA;
6      reg [15:0] A, B;
7      reg Cin;
8      wire [15:0] S;
9      wire Cout;
10
11     PPA uut (
12         .A(A),
13         .B(B),
14         .Cin(Cin),
15         .S(S),
16         .Cout(Cout)
17     );
18
19
20     initial begin
21         $dumpfile("waveform.vcd");
22         $dumpvars(0, tb_PPA);
23     end
24
25     initial begin
26         $monitor("A=%h B=%h Cin=%b | S=%h Cout=%b", A, B, Cin, S, Cout);
27
28         // Test cases
29         A = 16'h0000; B = 16'h0000; Cin = 0; #10;
30         A = 16'h0001; B = 16'h0002; Cin = 0; #10;
31         A = 16'hFFFF; B = 16'h0001; Cin = 0; #10;
32         A = 16'h1234; B = 16'h5678; Cin = 0; #10;
33         A = 16'hAAAA; B = 16'h5555; Cin = 1; #10;
34         A = 16'hFFFF; B = 16'hFFFF; Cin = 1; #10;
35
36         $finish;
37     end
38 endmodule
39
```

design.sv

```
1  // Code your design here
2  `timescale 1ns / 1ps
3
4  // 16-bit Parallel Prefix Adder (PPA)
5  module PPA(A, B, Cin, S, Cout);
6      input [15:0] A, B;
7      input Cin;
8      output [15:0] S;
9      output Cout;
10
11     wire [15:0] P, G;
12     wire [15:0] C;
13
14     // Step 1: Compute Generate (G) and Propagate (P) signals
15     assign G = A & B; // Carry Generate
16     assign P = A ^ B; // Carry Propagate
17
18     // Step 2: Prefix processing to compute carries
19     assign C[0] = Cin;
20     assign C[1] = G[0] | (P[0] & C[0]);
21     assign C[2] = G[1] | (P[1] & C[1]);
22     assign C[3] = G[2] | (P[2] & C[2]);
23     assign C[4] = G[3] | (P[3] & C[3]);
24     assign C[5] = G[4] | (P[4] & C[4]);
25     assign C[6] = G[5] | (P[5] & C[5]);
26     assign C[7] = G[6] | (P[6] & C[6]);
27     assign C[8] = G[7] | (P[7] & C[7]);
28     assign C[9] = G[8] | (P[8] & C[8]);
```

```
    assign C[10] = G[9] | (P[9] & C[9]);
    assign C[11] = G[10] | (P[10] & C[10]);
    assign C[12] = G[11] | (P[11] & C[11]);
    assign C[13] = G[12] | (P[12] & C[12]);
    assign C[14] = G[13] | (P[13] & C[13]);
    assign C[15] = G[14] | (P[14] & C[14]);
    assign Cout = G[15] | (P[15] & C[15]);

    // Step 3: Compute Sum bits
    assign S = P ^ C;

endmodule

```

EPWave

Log Share

```
[2025-03-13 03:27:13 UTC] iverilog '-Wall' '-g2012' design.sv testbench.sv  && unbuffer vvp a.out
VCD info: dumpfile waveform.vcd opened for output.
A=0000 B=0000 Cin=0 | S=0000 Cout=0
A=0001 B=0002 Cin=0 | S=0003 Cout=0
A=ffff B=0001 Cin=0 | S=0000 Cout=1
A=1234 B=5678 Cin=0 | S=68ac Cout=0
A=aaaa B=5555 Cin=1 | S=0000 Cout=1
A=ffff B=ffff Cin=1 | S=ffff Cout=1
testbench.sv:36: $finish called at 60000 (1ps)
Finding VCD file...
./waveform.vcd
[2025-03-13 03:27:14 UTC] Opening EPWave...
Done
```