

```
1
2 // 64-bit adder with 4 adder_16b instances. This is a simple extension of adder_32b, thus
  no testbench was required for validation.
3
4 module adder_64b (
5     input cin,
6     input [63:0] x, y,
7     output cout,
8     output [63:0] s
9 );
10
11 // carry-in signal for each 16-bit sub-adder. we use 'h' to denote a 'hierarchical' carry.
12 wire [4:0] hc;
13 assign hc[0] = cin;
14
15 // 'hierachical' Generate and Propagate signals.
16 wire [3:0] hP, hG;
17
18 // 4 adder_16b instances for each 16-bit subset of x and y.
19 genvar i;
20 generate
21     for (i=0; i<4; i = i+1) begin : subadders
22         adder_16b subadder (hc[i], x[16*i+15: 16*i], y[16*i+15: 16*i], hP[i], hG[i], s[16*i+15
23 : 16*i]);
24     end
25 endgenerate
26
27 // Hierarchical carries according to the lookahead framework.
28 assign hc[1] = hG[0] | hP[0] & cin;
29 assign hc[2] = hG[1] | hP[1] & hG[0] | hP[1] & hP[0] & cin;
30 assign hc[3] = hG[2] | hP[2] & hG[1] | hP[2] & hP[1] & hG[0] | hP[2] & hP[1] & hP[0] &
31 cin;
32 assign hc[4] = hG[3] | hP[3] & hG[2] | hP[3] & hP[2] & hG[1] | hP[3] & hP[2] & hP[1] & hG
33 [0] | hP[3] & hP[2] & hP[1] & hP[0] & cin;
34 assign cout = hc[4];
35
36 endmodule
37
```