1、 $F = (A'B' + AC')(A+D) = AA'B' + AC' + A'B'D + AC'D = AA'B' + AC' + A'B'D$

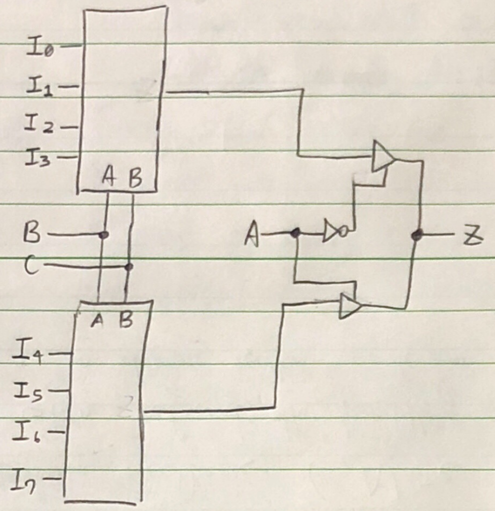
static-1 hazard: (0001) ↔ (1001)

static-0 hazard: (0010) ↔ (1010)

dynamic hazard: (0000) ↔ (1000)

AB / CD	00	01	11	10
00			1	1
01	1		1	1
11	1			-
10	-			

2.



$I_0 \sim I_7$ is the 8 inputs to the 8-to-1 MUX.

A, B, C is the control signal of the 8-to-1 MUX (A is the most significant bit)

Z is the output.

3、1) R=1 and H=0 cannot occur at the same time.

2)

R	H	Q	Q^+
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	X
1	0	1	X
1	1	0	1
1	1	1	1

$Q^+ = R + HQ$

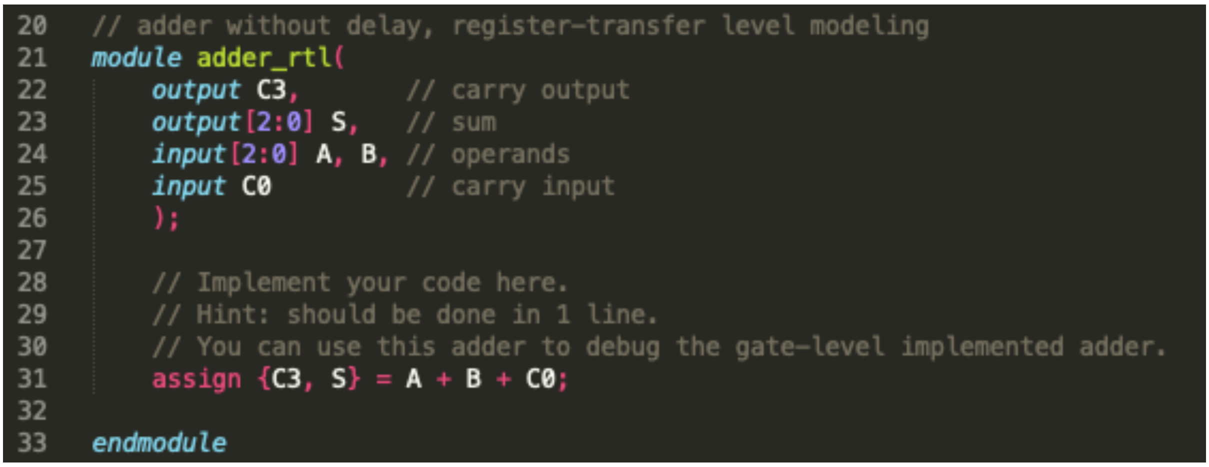
7. A half adder needs 2 gate.

A full adder needs 2 half adder and a OR gate, total of 5 gates

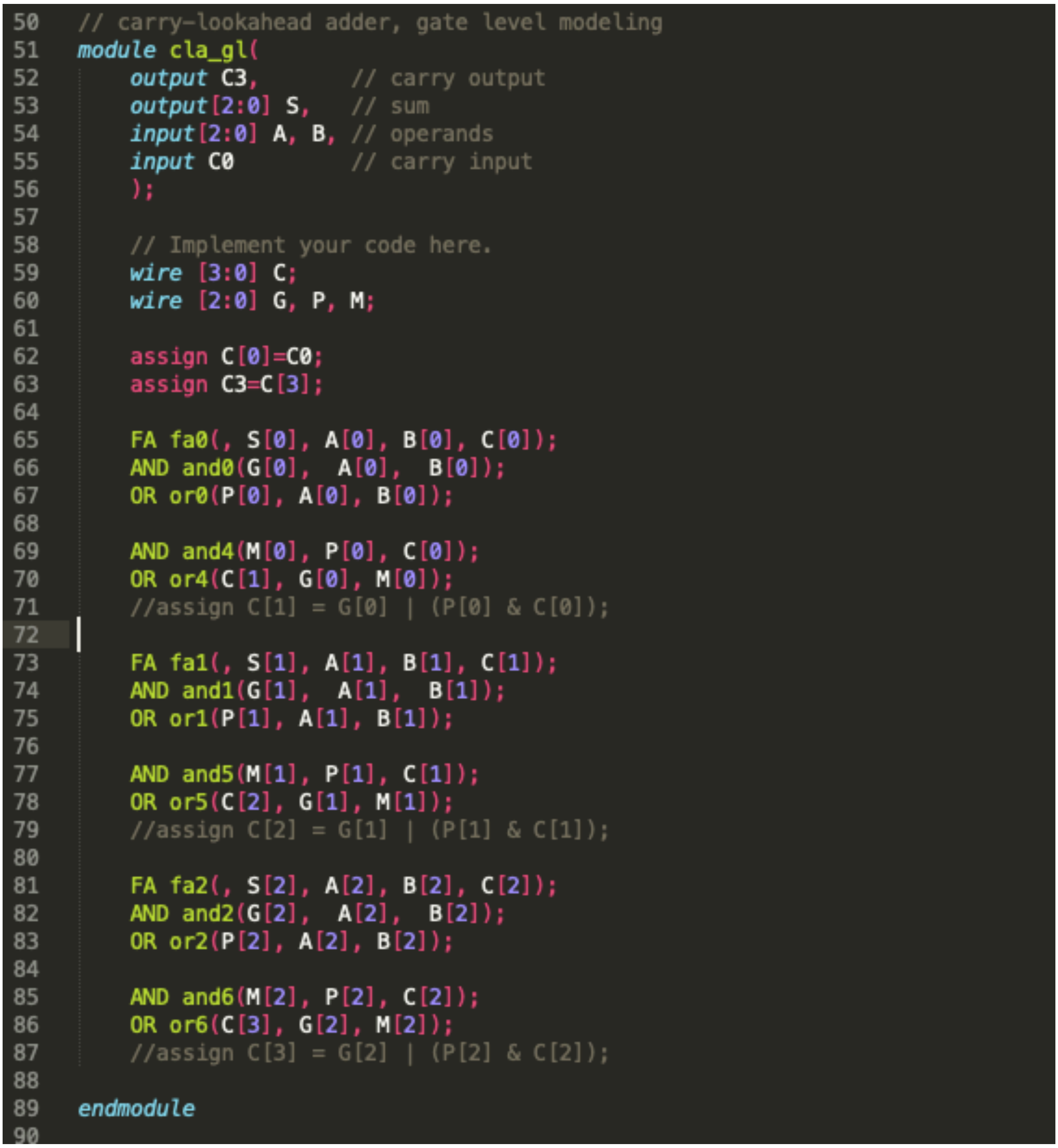
A n-bit CLA adder needs

- 1 full adder and 4 logic gates for a bit, total of 9 gates per bit

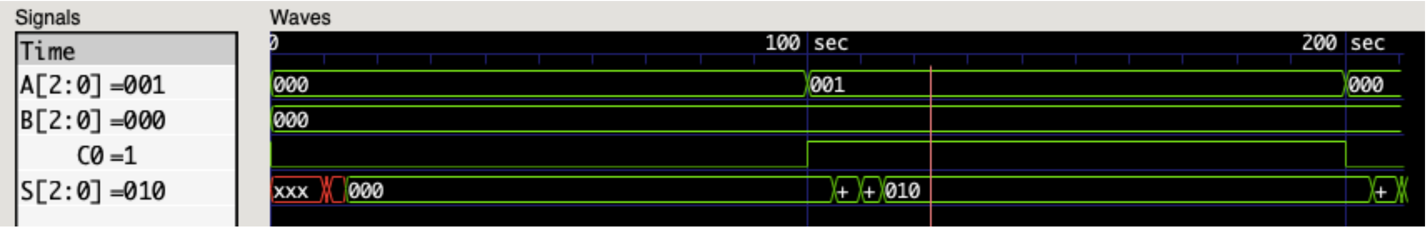
Number of Logic Gate = $9n$ #

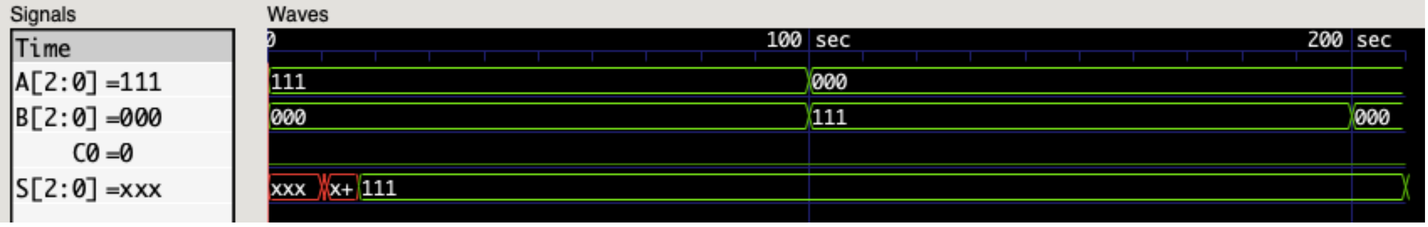


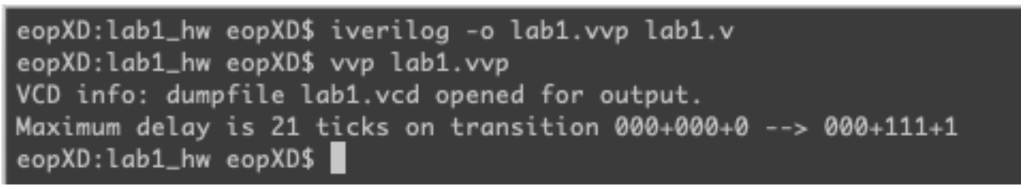
```verilog
// adder without delay, register-transfer level modeling
module adder_rtl(
    output C3,        // carry output
    output[2:0] S,    // sum
    input[2:0] A, B, // operands
    input C0          // carry input
    );

    // Implement your code here.
    // Hint: should be done in 1 line.
    // You can use this adder to debug the gate-level implemented adder.
    assign {C3, S} = A + B + C0;

endmodule
```



```verilog
// carry-lookahead adder, gate level modeling
module cla_gl(
    output C3,        // carry output
    output[2:0] S,    // sum
    input[2:0] A, B, // operands
    input C0          // carry input
    );

    // Implement your code here.
    wire [3:0] C;
    wire [2:0] G, P, M;

    assign C[0]=C0;
    assign C3=C[3];

    FA fa0(, S[0], A[0], B[0], C[0]);
    AND and0(G[0],  A[0],  B[0]);
    OR or0(P[0], A[0], B[0]);

    AND and4(M[0], P[0], C[0]);
    OR or4(C[1], G[0], M[0]);
    //assign C[1] = G[0] | (P[0] & C[0]);

    FA fa1(, S[1], A[1], B[1], C[1]);
    AND and1(G[1],  A[1],  B[1]);
    OR or1(P[1], A[1], B[1]);

    AND and5(M[1], P[1], C[1]);
    OR or5(C[2], G[1], M[1]);
    //assign C[2] = G[1] | (P[1] & C[1]);

    FA fa2(, S[2], A[2], B[2], C[2]);
    AND and2(G[2],  A[2],  B[2]);
    OR or2(P[2], A[2], B[2]);

    AND and6(M[2], P[2], C[2]);
    OR or6(C[3], G[2], M[2]);
    //assign C[3] = G[2] | (P[2] & C[2]);

endmodule

```







```
eopXD:lab1_hw eopXD$ iverilog -o lab1.vvp lab1.v
eopXD:lab1_hw eopXD$ vvp lab1.vvp
VCD info: dumpfile lab1.vcd opened for output.
Maximum delay is 21 ticks on transition 000+000+0 --> 000+111+1
eopXD:lab1_hw eopXD$ 
```