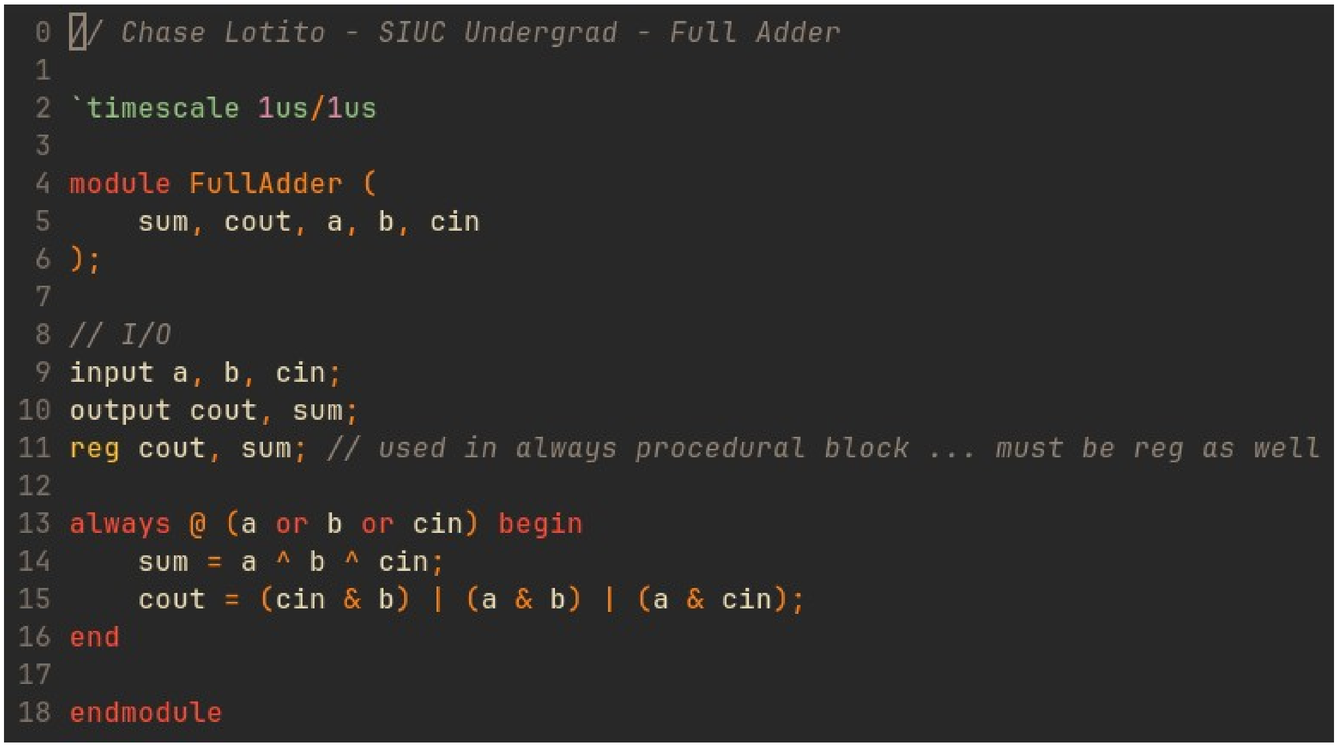
Chase Lotito
ECE426 - Chao Lu

Lab 4: 4-bit Subtractor Design

FullAdder.v



```verilog
// Chase Lotito - SIUC Undergrad - Full Adder

`timescale 1us/1us

module FullAdder (
    sum, cout, a, b, cin
);

// I/O
input a, b, cin;
output cout, sum;
reg cout, sum; // used in always procedural block ... must be reg as well

always @ (a or b or cin) begin
    sum = a ^ b ^ cin;
    cout = (cin & b) | (a & b) | (a & cin);
end

endmodule
```

adder-subtractor-struct-4b.v

```verilog
// Chase Lotito - SIUC - SP2024
// ECE426 - VLSI Lab
// Lab 4: Subtractor Design
// MODULE: 4-bit Adder-Subractor

`timescale 1us/1us
`include "FullAdder.v"

module adder_subtractor_struct_4b (
    a, b, m, rslt, cout, ovf
);

// I/O
input [3:0] a, b;           // 4-bit inputs
input m;                    // mode control
output [3:0] rslt, cout;    // 4-bit result and carrys
output ovf;                 // overflow bit
```

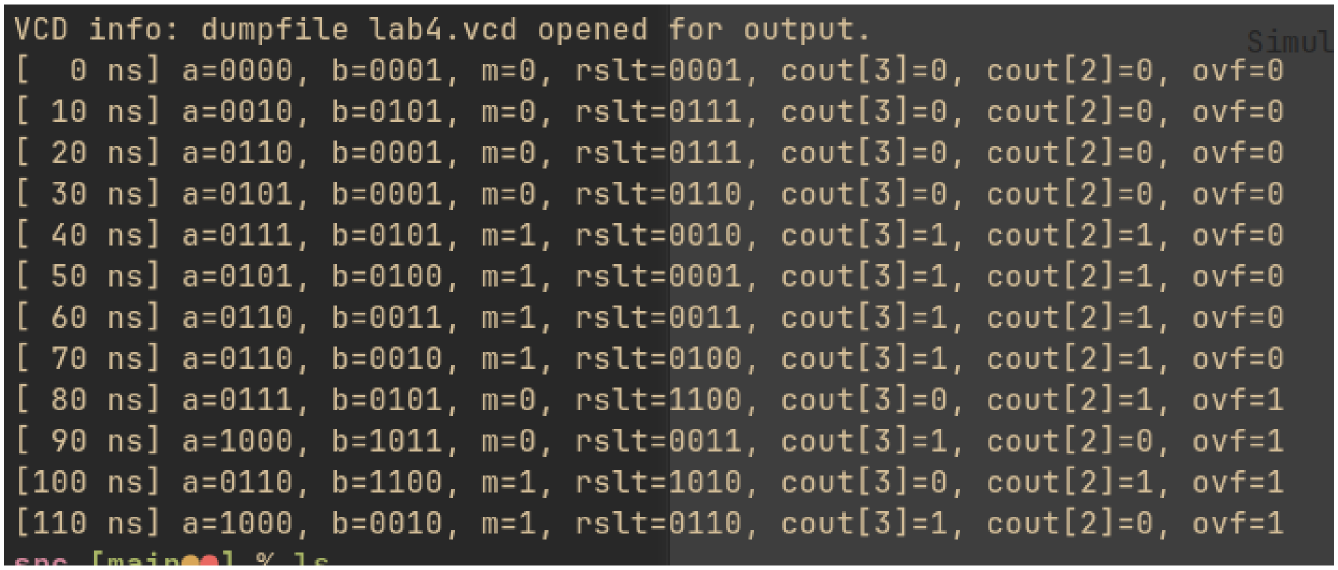
```
// Intermediary Wires
wire net0, net1, net2, net3;

////////////////
// GATE-LEVEL //

// overflow
xor xor0 (ovf, cout[2], cout[3]);

// FA0 section
xor xor1 (net0, m, b[0]);
FullAdder FA0 (
    .sum(rslt[0]),
    .cout(cout[0]),
    .a(a[0]),
    .b(net0),
    .cin(m)
);

// FA0 section
xor xor2 (net1, m, b[1]);
FullAdder FA1 (
    .sum(rslt[1]),
    .cout(cout[1]),
    .a(a[1]),
    .b(net1),
    .cin(cout[0])
);

// FA0 section
xor xor3 (net2, m, b[2]);
FullAdder FA2 (
    .sum(rslt[2]),
    .cout(cout[2]),
    .a(a[2]),
    .b(net2),
    .cin(cout[1])
);

// FA0 section
xor xor4 (net3, m, b[3]);
FullAdder FA3 (
    .sum(rslt[3]),
    .cout(cout[3]),
    .a(a[3]),
    .b(net3),
    .cin(cout[2])
);
endmodule
```

tb.v

```
// Chase Lotito - ECE426 Lab 4 - Testbench

`include "adder-subtractor-struct-4b.v"
`timescale 1us/1us

module tb();

// GTKWAVE
initial begin : GTKWAVE
    $dumpfile("lab4.vcd");
    $dumpvars(0, tb);
end

reg [3:0] a, b;
reg m;
wire [3:0] rslt, cout;
wire ovf;

initial //display variables

$monitor ("[%3t ns] a=%b, b=%b, m=%b, rslt=%b, cout[3]=%b, cout[2]=
%b, ovf=%b", $time, a, b, m, rslt, cout[3], cout[2], ovf);

initial //apply input vectors

begin

//addition
#0  a = 4'b0000; b = 4'b0001; m = 1'b0;
#10 a = 4'b0010; b = 4'b0101; m = 1'b0;
#10 a = 4'b0110; b = 4'b0001; m = 1'b0;
#10 a = 4'b0101; b = 4'b0001; m = 1'b0;

//subtraction
```

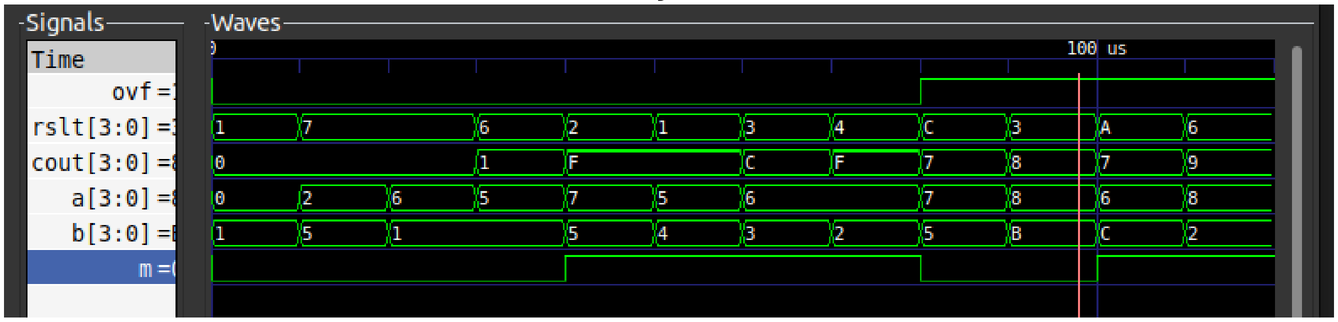
```
#10 a = 4'b0111; b = 4'b0101; m = 1'b1;
#10 a = 4'b0101; b = 4'b0100; m = 1'b1;
#10 a = 4'b0110; b = 4'b0011; m = 1'b1;
#10 a = 4'b0110; b = 4'b0010; m = 1'b1;

//overflow
#10 a = 4'b0111; b = 4'b0101; m = 1'b0;
#10 a = 4'b1000; b = 4'b1011; m = 1'b0;
#10 a = 4'b0110; b = 4'b1100; m = 1'b1;
#10 a = 4'b1000; b = 4'b0010; m = 1'b1;

#10 $finish;

end

adder_subtractor_struct_4b inst1 ( //instantiate the module
.a(a),
.b(b),
.m(m),
.rslt(rslt),
.cout(cout),
.ovf(ovf)
);

endmodule
```

Simulation Output Log



```
VCD info: dumpfile lab4.vcd opened for output.
[  0 ns] a=0000, b=0001, m=0, rslt=0001, cout[3]=0, cout[2]=0, ovf=0
[ 10 ns] a=0010, b=0101, m=0, rslt=0111, cout[3]=0, cout[2]=0, ovf=0
[ 20 ns] a=0110, b=0001, m=0, rslt=0111, cout[3]=0, cout[2]=0, ovf=0
[ 30 ns] a=0101, b=0001, m=0, rslt=0110, cout[3]=0, cout[2]=0, ovf=0
[ 40 ns] a=0111, b=0101, m=1, rslt=0010, cout[3]=1, cout[2]=1, ovf=0
[ 50 ns] a=0101, b=0100, m=1, rslt=0001, cout[3]=1, cout[2]=1, ovf=0
[ 60 ns] a=0110, b=0011, m=1, rslt=0011, cout[3]=1, cout[2]=1, ovf=0
[ 70 ns] a=0110, b=0010, m=1, rslt=0100, cout[3]=1, cout[2]=1, ovf=0
[ 80 ns] a=0111, b=0101, m=0, rslt=1100, cout[3]=0, cout[2]=1, ovf=1
[ 90 ns] a=1000, b=1011, m=0, rslt=0011, cout[3]=1, cout[2]=0, ovf=1
[100 ns] a=0110, b=1100, m=1, rslt=1010, cout[3]=0, cout[2]=1, ovf=1
[110 ns] a=1000, b=0010, m=1, rslt=0110, cout[3]=1, cout[2]=0, ovf=1
```

Waveforms


Signals
Waves
Time
ovf =
rslt[3:0] =
cout[3:0] =
a[3:0] =
b[3:0] =
m =
100 us
1 7 6 2 1 3 4 C 3 A 6
0 1 F C F 7 8 7 9
0 2 6 5 7 5 6 7 8 6 8
1 5 1 5 4 3 2 5 B C 2