Project NO. - 6

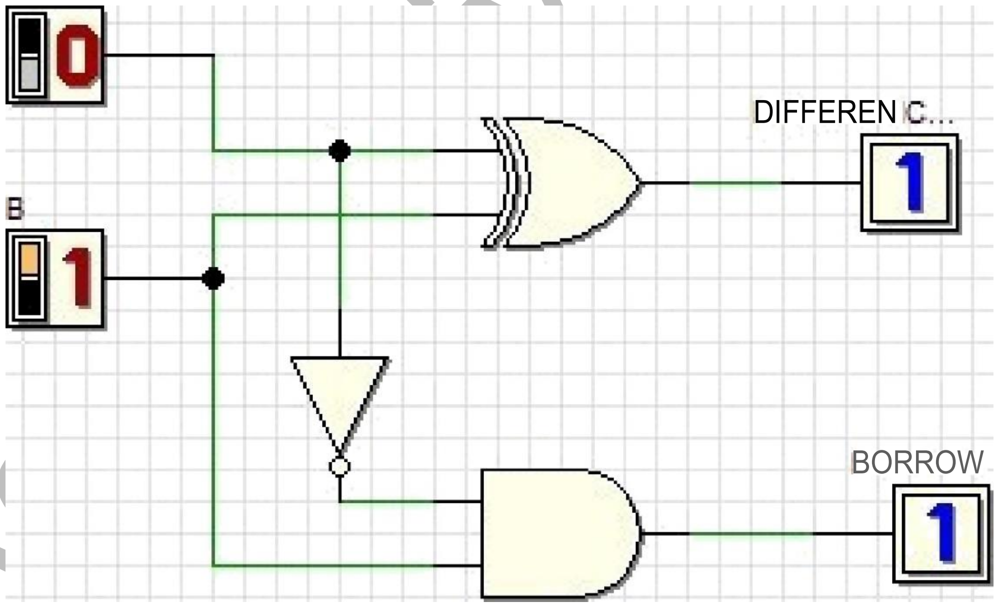
AIM : To design a half subtractor and full subtractor using basic gate and NAND gate and to verify its truth table .

APPARATUS : Input and output switches ; wires for connection ; XOR , NOT,AND, NAND,NOR Gates ; DEEDS simulator

TRUTH TABLE FOR HALF SUBTRACTOR:

INPUT		OUTPUT	
A	B	DIFFERENCE (A-B)	BORROW
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

HALF SUBTRACTOR



TRUTH TABLE FOR FULL SUBTRACTOR :

INPUT			OUTPUT	
A	B	Bin	DIFFERENCE (A-B-Bin)	BORROW
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

KMAP FOR FULL SUBTRACTOR (DIFFERENCE)

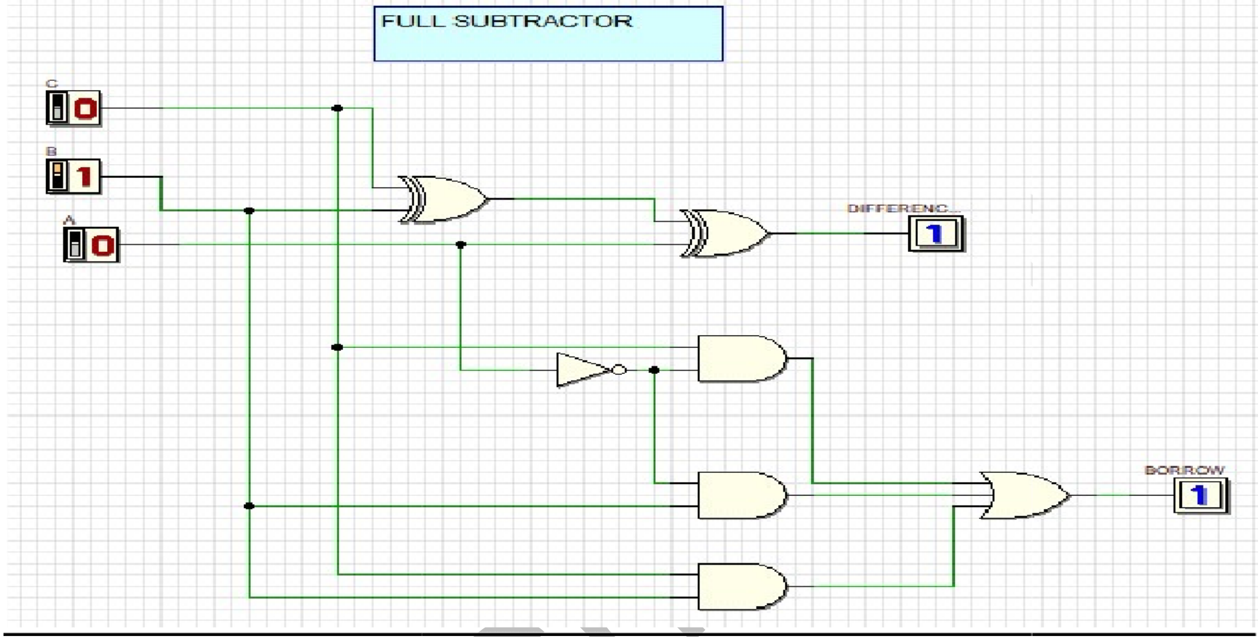
		B’*(Bin)’	B’*(Bin)	B*(Bin)	B*(Bin)’
A’		0	1	0	1
A		1	0	1	0

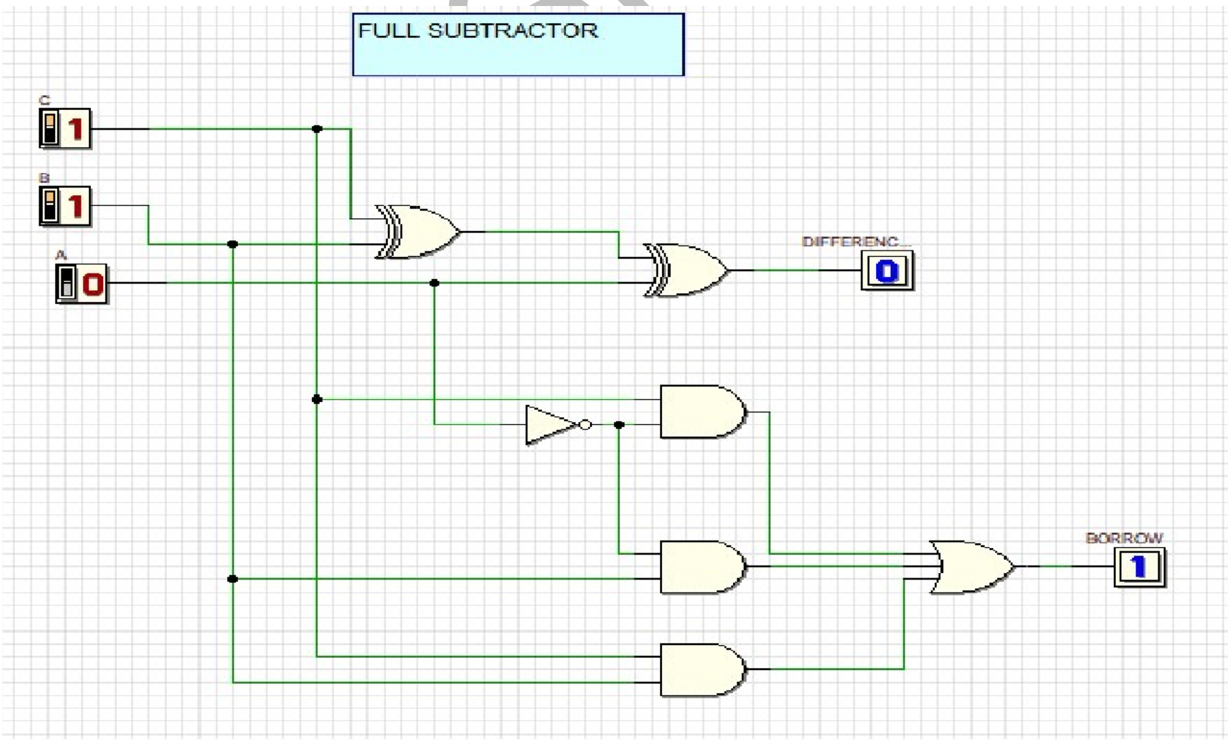
DIFFERENCE = (A ‘)*(B’*(Bin)+ B*(Bin)’) + A*(B’*(Bin)’+ B*(Bin)) .

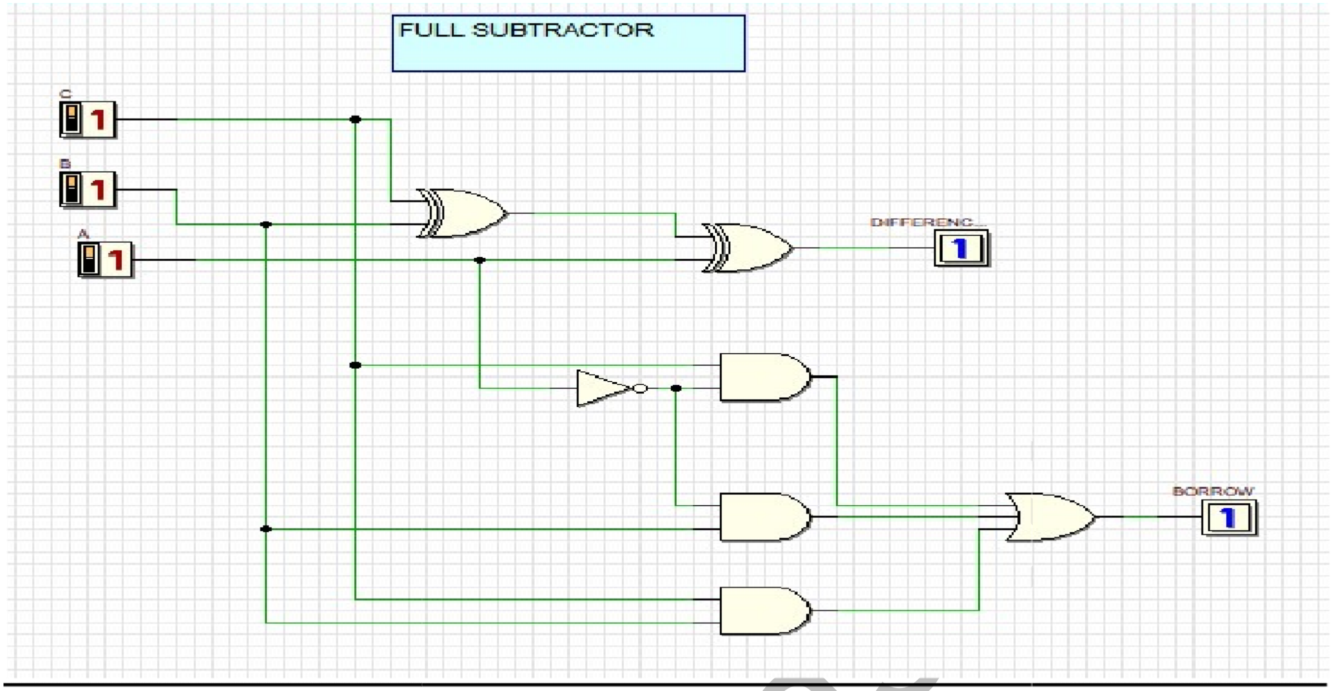
$=A\oplus B\oplus Bin$

KMAP FOR FULL SUBTRACTOR (BORROW)

	B’*(Bin)’	B’*(Bin)	B*(Bin)	B*(Bin)’	
A’	0	1	1	1	
A	0	0	1	0	


FULL SUBTRACTOR
C
0
B
1
A
0
DIFFERENC...
1
BORROW
1

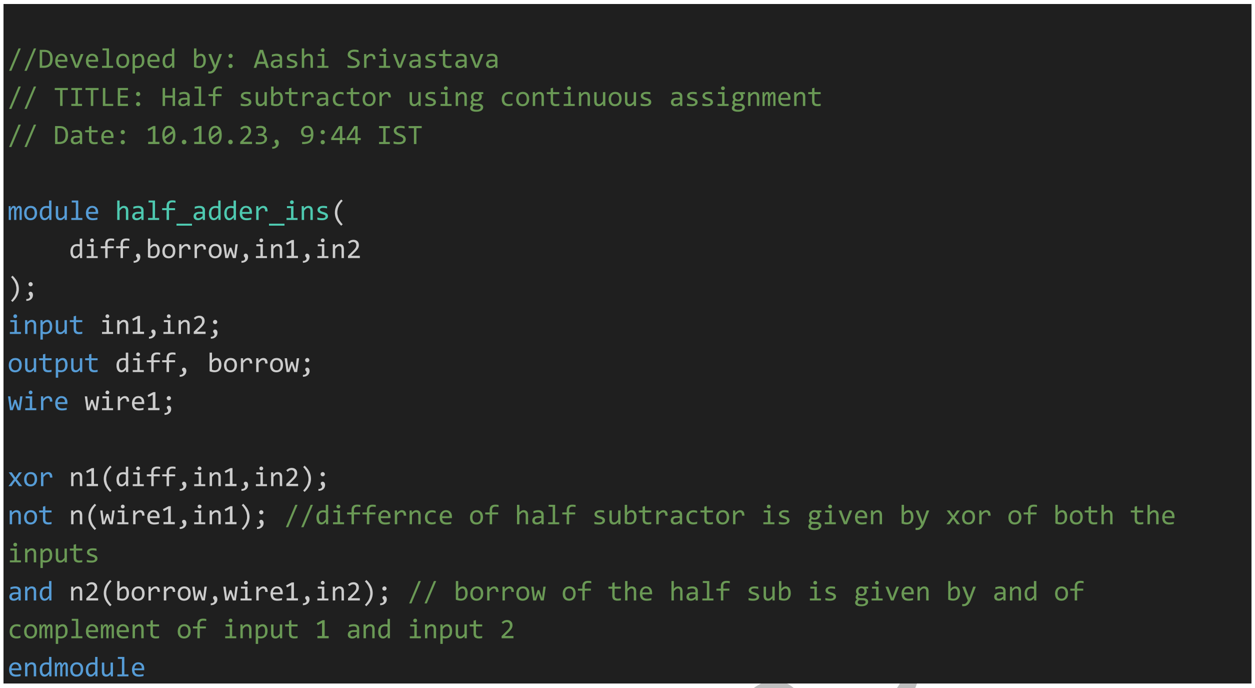

FULL SUBTRACTOR
C
1
B
1
A
0
DIFFERENC...
0
BORROW
1



Verilog Code:

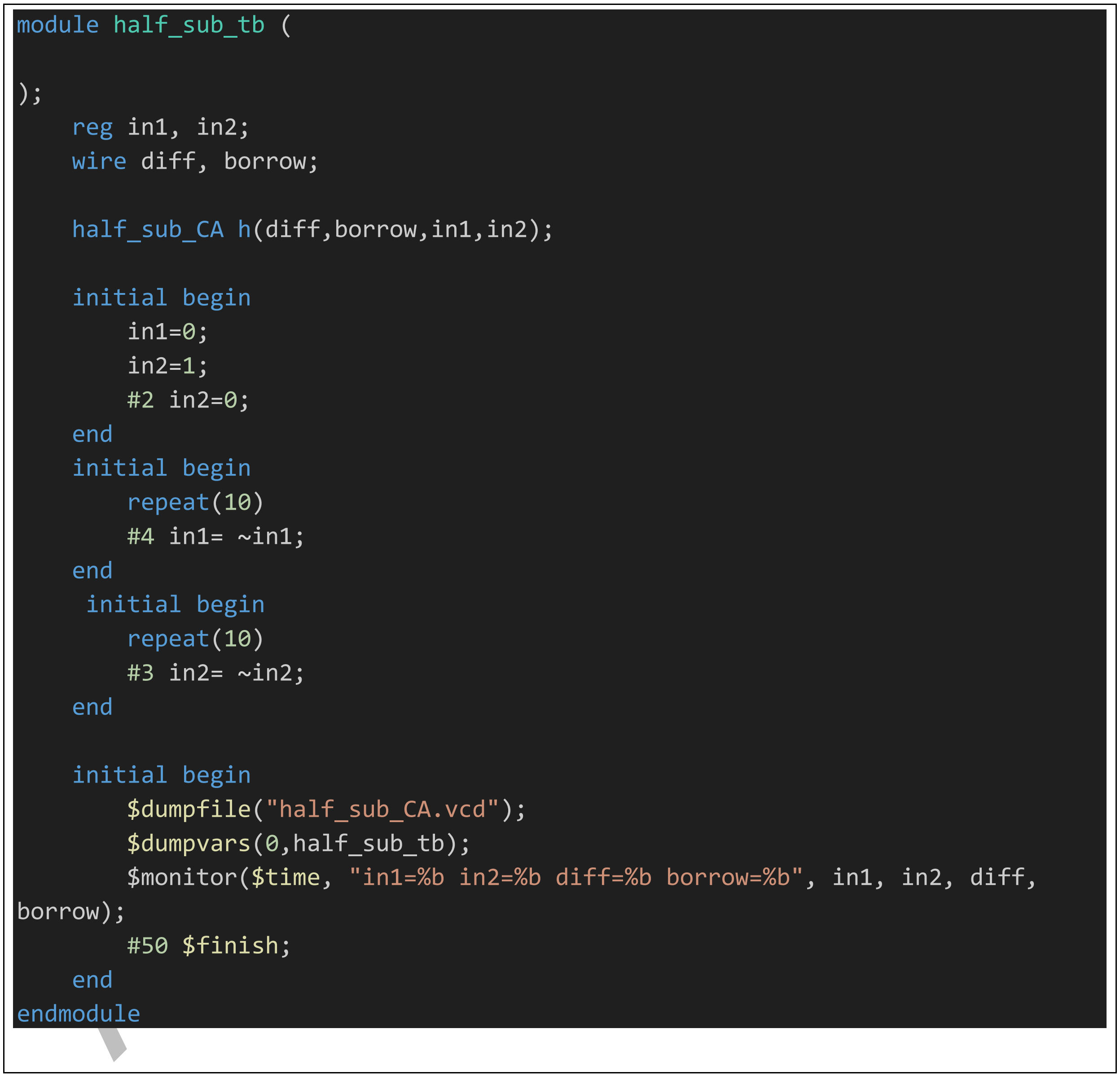
Half Subtractor

```
//Developed by: Aashi Srivastava
// TITLE: Half Subtractor using continuous assignment
// Date: 10.10.23, 19:53 IST

module half_sub_CA(
    diff,borrow,in1,in2
);
input in1,in2;
output diff, borrow;

assign diff=in1^in2; //differnce of half subtractor is given by xor of both
the inputs
assign borrow=(~in1 )& (in2); // borrow of the half sub is given by and of
complement of input 1 and input 2
endmodule
```



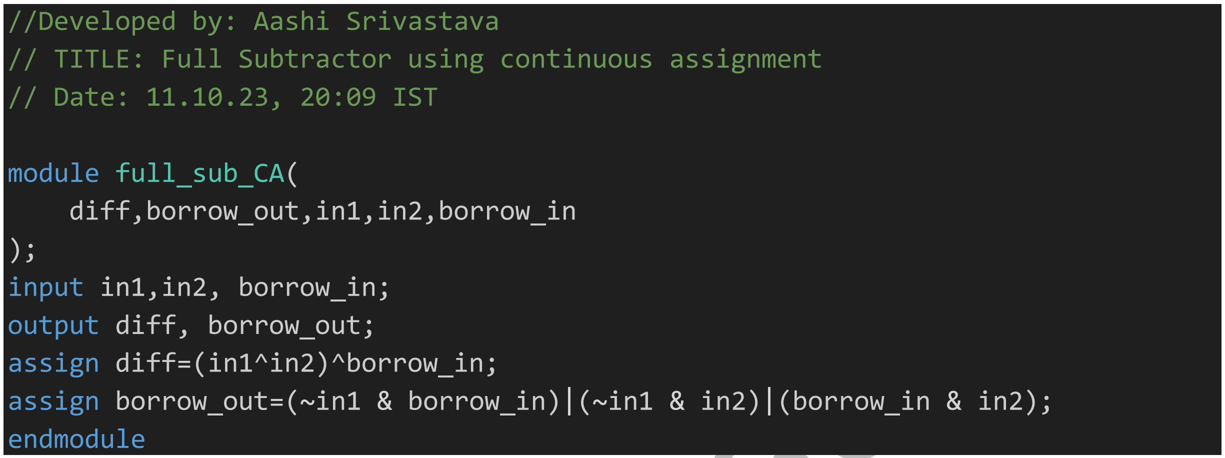
```
//Developed by: Aashi Srivastava
// TITLE: Half subtractor using continuous assignment
// Date: 10.10.23, 9:44 IST

module half_adder_ins(
    diff,borrow,in1,in2
);
input in1,in2;
output diff, borrow;
wire wire1;

xor n1(diff,in1,in2);
not n(wire1,in1); //differnce of half subtractor is given by xor of both the inputs
and n2(borrow,wire1,in2); // borrow of the half sub is given by and of complement of input 1 and input 2
endmodule
```

Test-Bench:



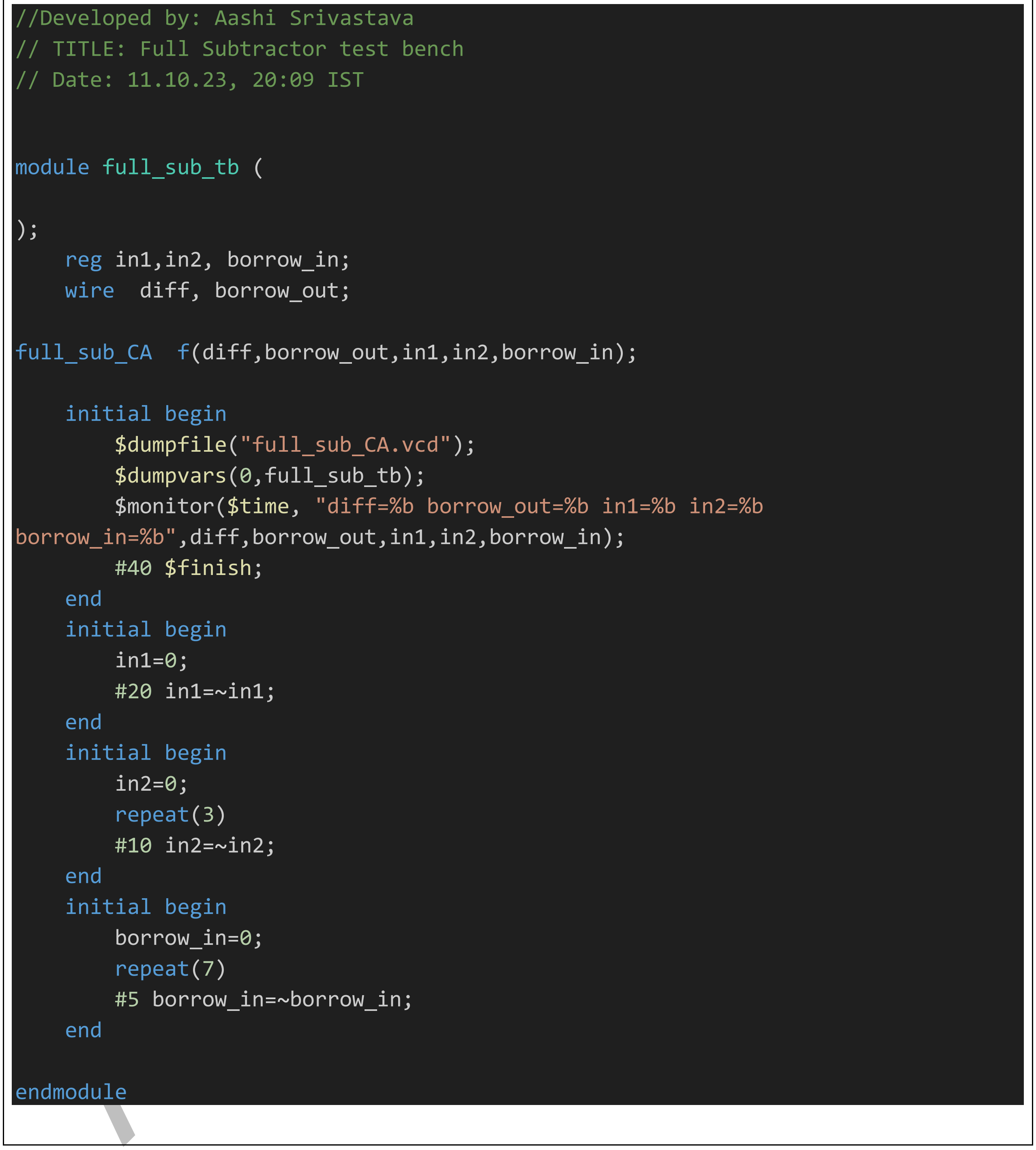
```
module half_sub_tb (

);
    reg in1, in2;
    wire diff, borrow;

    half_sub_CA h(diff,borrow,in1,in2);

    initial begin
        in1=0;
        in2=1;
        #2 in2=0;
    end
    initial begin
        repeat(10)
        #4 in1= ~in1;
    end
     initial begin
        repeat(10)
        #3 in2= ~in2;
    end

    initial begin
        $dumpfile("half_sub_CA.vcd");
        $dumpvars(0,half_sub_tb);
        $monitor($time, "in1=%b in2=%b diff=%b borrow=%b", in1, in2, diff,
borrow);
        #50 $finish;
    end
endmodule
```

Full Subtractor



```
//Developed by: Aashi Srivastava
// TITLE: Full Subtractor using continuous assignment
// Date: 11.10.23, 20:09 IST

module full_sub_CA(
    diff,borrow_out,in1,in2,borrow_in
);
input in1,in2, borrow_in;
output diff, borrow_out;
assign diff=(in1^in2)^borrow_in;
assign borrow_out=(~in1 & borrow_in)|(~in1 & in2)|(borrow_in & in2);
endmodule
```

Test-Bench:



```
//Developed by: Aashi Srivastava
// TITLE: Full Subtractor test bench
// Date: 11.10.23, 20:09 IST


module full_sub_tb (

);
    reg in1,in2, borrow_in;
    wire  diff, borrow_out;

full_sub_CA  f(diff,borrow_out,in1,in2,borrow_in);

    initial begin
        $dumpfile("full_sub_CA.vcd");
        $dumpvars(0,full_sub_tb);
        $monitor($time, "diff=%b borrow_out=%b in1=%b in2=%b
borrow_in=%b",diff,borrow_out,in1,in2,borrow_in);
        #40 $finish;
    end
    initial begin
        in1=0;
        #20 in1=~in1;
    end
    initial begin
        in2=0;
        repeat(3)
        #10 in2=~in2;
    end
    initial begin
        borrow_in=0;
        repeat(7)
        #5 borrow_in=~borrow_in;
    end

endmodule
```