Lab Report
CSE4010 Computer Architecture

Name: Zachary A. Hampton **Score:** /10
Student ID: 008339494 **Due:** 02-23-2025
Lab: Lab 3 - Full Subtractor Implementation

Report

- What is a full subtractor? How does it differ from a half subtractor?
  - A full subtractor is a combinational circuit that performs subtraction of three binary digits: minuend (A), subtrahend (B), and borrow-in (Bin), producing difference (diff) and borrow-out (Bout) outputs.
  - Unlike a half subtractor which only handles two inputs (A and B), a full subtractor can process a borrow from a previous stage, making it suitable for multi-bit subtraction operations.
- How can a full subtractor be constructed using half subtractors?
  - A full subtractor can be built using two half subtractors and an OR gate:
    - The first half subtractor subtracts B from A
    - The second half subtractor subtracts Bin from the difference of the first stage
    - The final borrow-out is the OR of both half subtractor borrows

Source Code

halfSubtractor.v

```verilog
module halfSubtractor(
    input op1,
    input op2,
    output diff,
    output borrow
);

    assign diff = op1 ^ op2;
    assign borrow = !op1 & op2;

endmodule
```

fullSubtractor.v

```verilog
module fullSubtractor(
    input A,
    input B,
    input Bin,
    output diff,
    output Bout
);

    wire diff1, borrow1, borrow2;

    halfSubtractor hs1(
        .op1(A),
        .op2(B),
        .diff(diff1),
        .borrow(borrow1)
    );

    halfSubtractor hs2(
        .op1(diff1),
        .op2(Bin),
        .diff(diff),
        .borrow(borrow2)
    );

    assign Bout = borrow1 | borrow2;

endmodule
```

fullSubtractor_tb.v

```verilog
`timescale 1ns/1ns
`include "halfSubtractor.v"
`include "fullSubtractor.v"

module fullSubtractor_tb;
    reg A, B, Bin;
    wire diff, Bout;

    fullSubtractor uut(
        .A(A),
        .B(B),
        .Bin(Bin),
        .diff(diff),
        .Bout(Bout)
    );

    initial begin
        $dumpfile("fullSubtractor_tb.vcd");
        $dumpvars(0, fullSubtractor_tb);

        // Test all input combinations
        A = 0; B = 0; Bin = 0; #20;
        A = 0; B = 0; Bin = 1; #20;
        A = 0; B = 1; Bin = 0; #20;
        A = 0; B = 1; Bin = 1; #20;
        A = 1; B = 0; Bin = 0; #20;
        A = 1; B = 0; Bin = 1; #20;
        A = 1; B = 1; Bin = 0; #20;
        A = 1; B = 1; Bin = 1; #20;

        $display("Test␣Complete!");
        $finish;
    end

    initial begin
        $monitor("Time=%0d␣A=%b␣B=%b␣Bin=%b␣diff=%b␣Bout=%b",
                 $time, A, B, Bin, diff, Bout);
    end
endmodule
```

Truth Table and Results

A	B	Bin	diff	Bout
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

[Simulation results and waveform analysis will be added after running the testbench]