CS 223 Digital Design

Bilkent University

Fall 2024-25

Laboratory Assignment 2

Arithmetic Circuits on FPGA

Nabeeha Khan

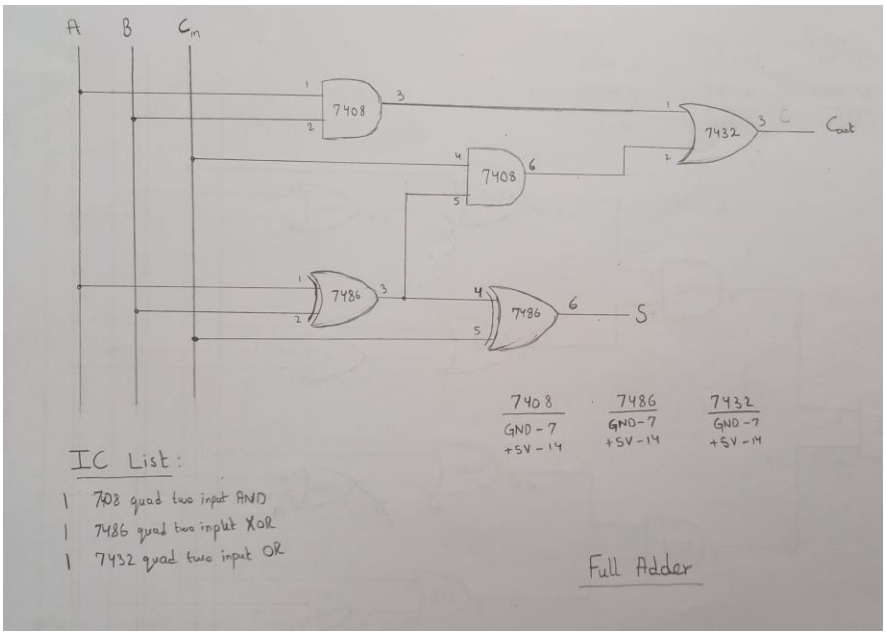
22301304

Section: 03

Date: 4th November 2024

Circuit Schematics:

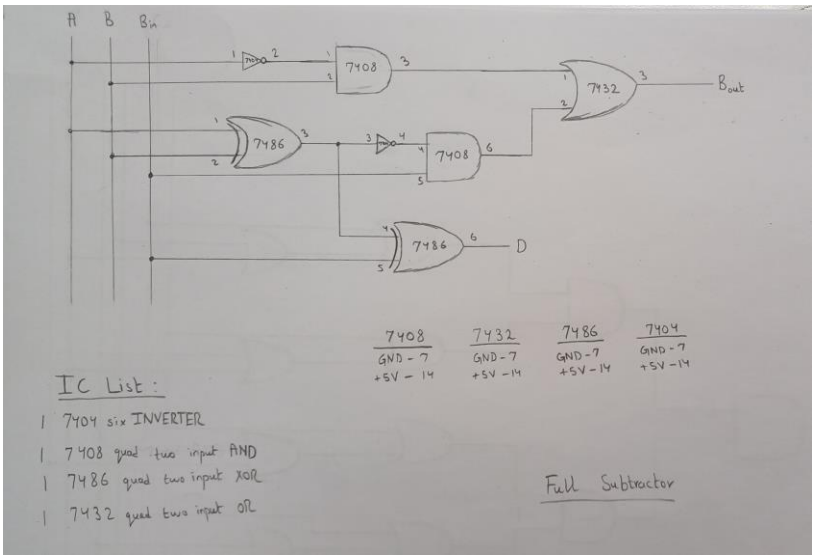
1.Circuit schematic for Full Adder:



IC List:

- 1x 7408 quad two input AND
- 1x 7486 quad two input XOR
- 1x 7432 quad two input OR

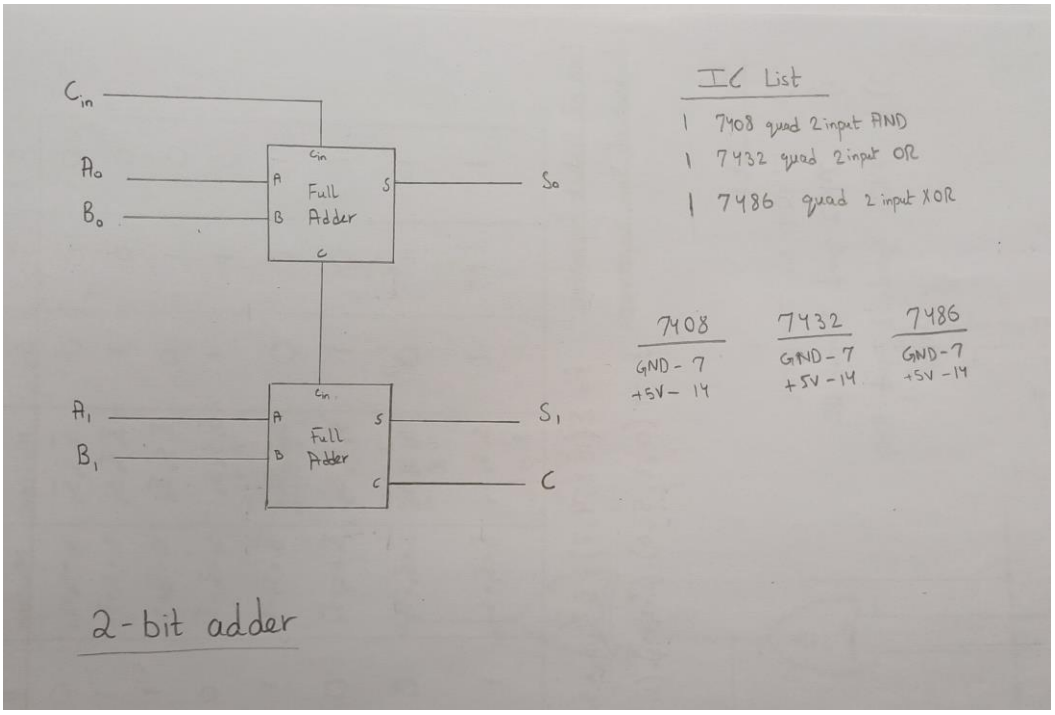
2.Circuit schematic for Full Subtractor:



IC List:

- 1x 7408 quad two input AND
- 1x 7486 quad two input XOR
- 1x 7432 quad two input OR
- 1x 7404 six INVERTER

3. Circuit schematic for 2-bit Adder:



IC List:

- 1x 7408 quad two input AND
- 1x 7486 quad two input XOR
- 1x 7432 quad two input OR

System Verilog Modules:

0. Modules for 2-input AND, OR, and XOR gates:

```
module AND2input(input logic a, b, output logic y);
     assign y = a & b;
endmodule

module OR2input(input logic a, b, output logic y);
     assign y = a | b;
endmodule

module XOR2input(input logic a, b, output logic y);
     assign y = a ^ b;
endmodule

module NOT(input logic a, output logic y);
     assign y = ~a;
endmodule
```

1. Behavioral module for Full Adder and testbench:

```
module fullAdderBeh(input logic a, b, cin, output logic s, c);
     logic l0, l1, l2;
     assign l0 = a ^ b;
     assign s = l0 ^ cin;
     assign l1 = a & b;
     assign l2 = cin & l0;
     assign c = l1 | l2;
```

```
endmodule
```

Testbench:

```
module fullAdderBeh_TB();
     logic a, b, cin, s, c;
     fullAdderBeh dut(a, b, cin, s, c);

     initial begin
          a = 0; b = 0; cin = 0; #10;
          if(s != 0 || c != 0) $display("000 failed.");
          a = 0; b = 0; cin = 1; #10;
          if(s != 1 || c != 0) $display("001 failed.");
          a = 0; b = 1; cin = 0; #10;
          if(s != 1 || c != 0) $display("010 failed.");
          a = 0; b = 1; cin = 1; #10;
          if(s != 0 || c != 1) $display("011 failed.");
          a = 1; b = 0; cin = 0; #10;
          if(s != 1 || c != 0) $display("100 failed.");
          a = 1; b = 0; cin = 1; #10;
          if(s != 0 || c != 1) $display("101 failed.");
          a = 1; b = 1; cin = 0; #10;
          if(s != 0 || c != 1) $display("110 failed.");
          a = 1; b = 1; cin = 1; #10;
          if(s != 1 || c != 1) $display("111 failed.");
     end
endmodule
```

2. Structural module for Full Adder and testbench:

```
module fullAdderStruc(input logic a, b, cin, output logic s, c);
     logic l0, l1, l2;
     XOR2input l0_xor(a, b, l0);
     XOR2input s_xor(l0, cin, s);
     AND2input l1_and(a, b, l1);
     AND2input l2_and(cin, l0, l2);
     OR2input c_or(l1, l2, c);
endmodule
```

Testbench:

```
module fullAdderStruc_TB();
     logic a, b, cin, s, c;
     fullAdderStruc dut(a, b, cin, s, c);

     initial begin
          a = 0; b = 0; cin = 0; #10;
          if(s != 0 || c != 0) $display("000 failed.");
          a = 0; b = 0; cin = 1; #10;
          if(s != 1 || c != 0) $display("001 failed.");
          a = 0; b = 1; cin = 0; #10;
          if(s != 1 || c != 0) $display("010 failed.");
          a = 0; b = 1; cin = 1; #10;
          if(s != 0 || c != 1) $display("011 failed.");
          a = 1; b = 0; cin = 0; #10;
          if(s != 1 || c != 0) $display("100 failed.");
          a = 1; b = 0; cin = 1; #10;
```

```
            if(s != 0 || c != 1) $display("101 failed.");
            a = 1; b = 1; cin = 0; #10;
            if(s != 0 || c != 1) $display("110 failed.");
            a = 1; b = 1; cin = 1; #10;
            if(s != 1 || c != 1) $display("111 failed.");
      end
endmodule
```

3. Behavioral module for Full Subtractor and testbench:

```
module fullSubtractorBeh(input logic a, b, bin, output logic d,
bout);
      logic l0, l1, l2;
      assign l0 = a ^ b;
      assign d = l0 ^ bin;
      assign  l1 = ~a & b;
      assign l2 = bin & ~l0;
      assign bout = l1 | l2;
endmodule
```

Testbench:

```
module fullSubtractorBeh_TB();
      logic a, b, bin, d, bout;
      fullSubtractorBeh dut(a, b, bin, d, bout);

      initial begin
            a = 0; b = 0; bin = 0; #10;
            if(d != 0 || bout != 0) $display("000 failed.");
```

```
        a = 0; b = 0; bin = 1; #10;
        if(d != 1 || bout != 1) $display("001 failed.");
        a = 0; b = 1; bin = 0; #10;
        if(d != 1 || bout != 1) $display("010 failed.");
        a = 0; b = 1; bin = 1; #10;
        if(d != 0 || bout != 1) $display("011 failed.");
        a = 1; b = 0; bin = 0; #10;
        if(d != 1 || bout != 0) $display("100 failed.");
        a = 1; b = 0; bin = 1; #10;
        if(d != 0 || bout != 0) $display("101 failed.");
        a = 1; b = 1; bin = 0; #10;
        if(d != 0 || bout != 0) $display("110 failed.");
        a = 1; b = 1; bin = 1; #10;
        if(d != 1 || bout != 1) $display("111 failed.");
    end
endmodule
```

4. Structural module for Full Subtractor and testbench:

```
module fullSubtractorStruc(input logic a, b, bin, output logic d,
bout);
    logic l0, l1, l2, l3, l4;
    XOR2input l0_xor(a, b, l0);
    XOR2input d_xor(l0, bin, d);
    NOT  l1_not(a, l1);
    AND2input l2_and(b, l1, l2);
    NOT l3_not(l0, l3);
    AND2input l4_and(bin, l3, l4);
```

```
    OR2input bout_or(l2, l4, bout);
endmodule
```

Testbench:

```
module fullSubtractorStruc_TB();
    logic a, b, bin, d, bout;
    fullSubtractorStruc dut(a, b, bin, d, bout);

    initial begin
        a = 0; b = 0; bin = 0; #10;
        if(d != 0 || bout != 0) $display("000 failed.");
        a = 0; b = 0; bin = 1; #10;
        if(d != 1 || bout != 1) $display("001 failed.");
        a = 0; b = 1; bin = 0; #10;
        if(d != 1 || bout != 1) $display("010 failed.");
        a = 0; b = 1; bin = 1; #10;
        if(d != 0 || bout != 1) $display("011 failed.");
        a = 1; b = 0; bin = 0; #10;
        if(d != 1 || bout != 0) $display("100 failed.");
        a = 1; b = 0; bin = 1; #10;
        if(d != 0 || bout != 0) $display("101 failed.");
        a = 1; b = 1; bin = 0; #10;
        if(d != 0 || bout != 0) $display("110 failed.");
        a = 1; b = 1; bin = 1; #10;
        if(d != 1 || bout != 1) $display("111 failed.");
    end
endmodule
```

5. Structural module for 2-bit Adder and testbench:

```
module twobitAdderStruc(input logic cin, a0, b0, a1, b1,output
logic sum0, sum1, cout1);
     logic cout0;
     fullAdderStruc fullAdder0(a0, b0, cin, sum0, cout0);
     fullAdderStruc fullAdder1(a1, b1, cout0, sum1, cout1);
endmodule
```

Testbench:

```
module twobitAdderStruc_TB();
     logic cin, a0, b0, a1, b1, sum0, sum1, cout1;
     twobitAdderStruc dut(cin, a0, b0, a1, b1, sum0, sum1, cout1);

     initial begin
          cin = 0; a0 = 0; b0 = 0; a1 = 0; b1 = 0; #10;
          cin = 0; a0 = 1; b0 = 0; a1 = 0; b1 = 0; #10;
          cin = 0; a0 = 0; b0 = 0; a1 = 1; b1 = 0; #10;
          cin = 0; a0 = 1; b0 = 1; a1 = 0; b1 = 0; #10;
          cin = 0; a0 = 0; b0 = 0; a1 = 1; b1 = 1; #10;
          cin = 1; a0 = 0; b0 = 0; a1 = 1; b1 = 0; #10;
          cin = 1; a0 = 1; b0 = 1; a1 = 0; b1 = 0; #10;
          cin = 1; a0 = 1; b0 = 1; a1 = 1; b1 = 1; #10;
     end
endmodule
```