2020-2021 Spring Semester

Lab-2 Preliminary Report

Course Name:CS223

Section: 1

Lab: 2

Name: Alper

Surname: Mumcular

Student ID: 21902740

Date: 22.02.2021

1) Schematic for a 1-bit fulladder

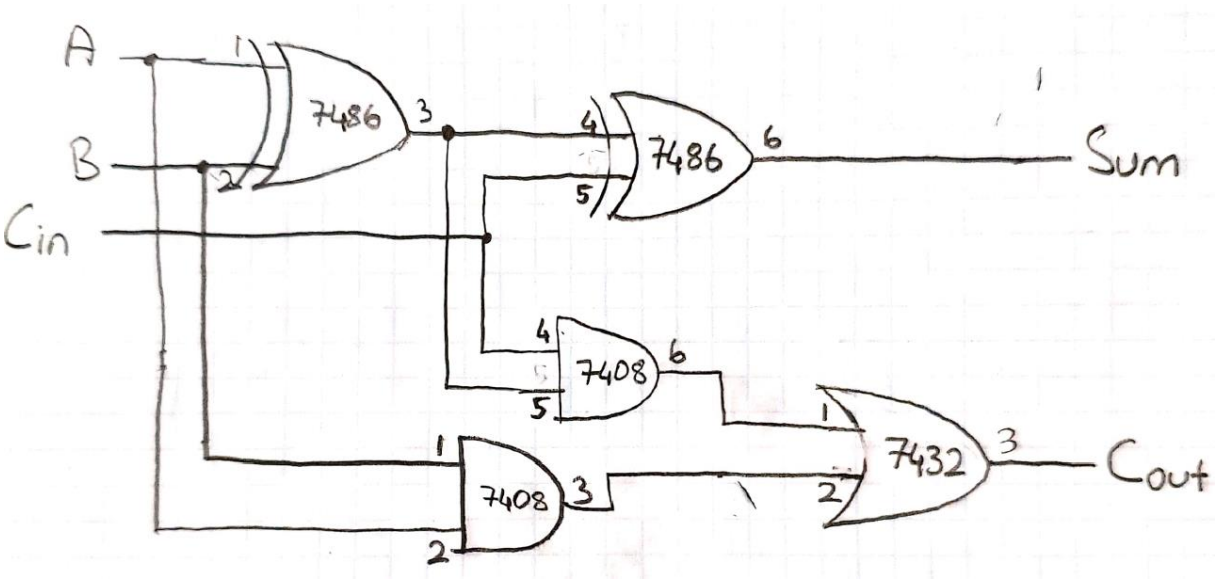

Figure 1: Schematic for a 1-bit fulladder

IC List:

- One 7408 Quad 2-Input AND gate
- One 7432 Quad 2-Input OR gate
- One 7486 Quad 2-Input XOR gate

7408: GND → 7, +5V → 14
7432: GND → 7, +5V → 14
7486: GND → 7, +5V → 14

2) Schematic for a 2-bit adder

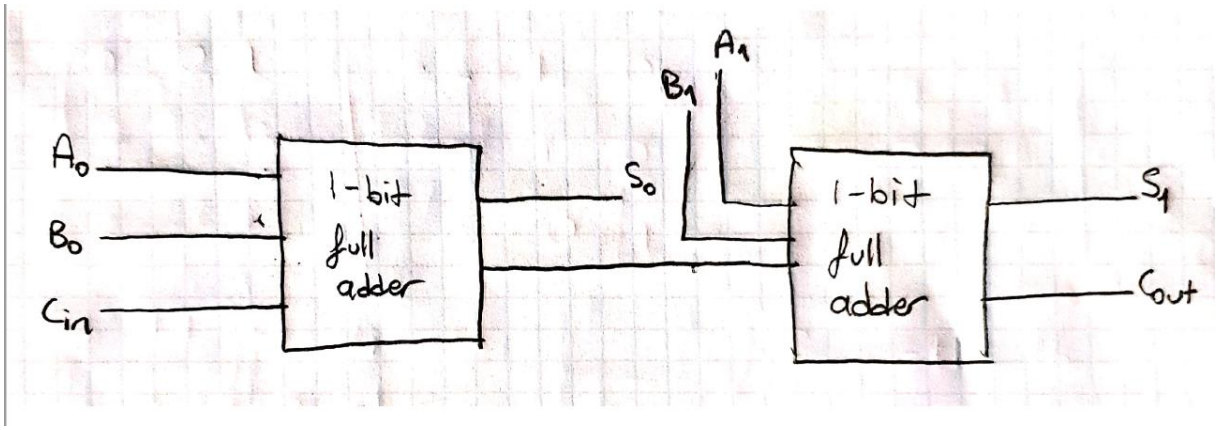

Figure 2: Schematic for a 2-bit adder

3.1) Dataflow Systemverilog module for a 1-bit fulladder

```
module onebitfulladder (input logic a, b, cin, output logic sum, cout);
	assign sum = a ^ b ^ cin;
	assign cout = (a ^ b) & cin | (a & b);
endmodule
```

3.2) Testbench module for a dataflow 1-bit fulladder module

```
module testbench1bitfulladder();
	logic a, b, cin;
	logic sum, cout;
	onebitfulladder dut(a, b, cin, sum, cout);
	initial begin
		a = 0; b = 0; cin = 0; #10;
		cin = 1; #10;
		b = 1; cin = 0; #10;
		cin = 1; #10;
		a = 1; b = 0; cin = 0; #10;
		cin = 1; #10;
		b = 1; cin = 0; #10;
		cin = 1; #10;
	end
endmodule
```

4.1) Structural Systemverilog module for a 1-bit fulladder

```
module and2(input logic a, b, output logic c);
	assign c = a & b;
endmodule

module or2(input logic a, b, output logic c);
	assign c = a | b;
endmodule

module xor2(input logic a, b, output logic c);
	assign c = a ^ b;
endmodule

module onebitfulladderstructural(input logic a, b, cin, output logic sum, cout);
	logic o1, o2, o3;
	xor2 xor_gate (a, b, o1);
	xor2 xor_gate2 (o1, cin, sum);
	and2 and_gate (a, b, o2);
	and2 and_gate2 (o1, cin, o3);
	or2 or_gate (o2, o3, cout);
endmodule
```

4.2) Testbench module for a structural 1-bit fulladder module

```
module testbench1bitfulladderstructural();
	logic a, b, cin;
	logic sum, cout;
	onebitfulladderstructural dut(a, b, cin, sum, cout);
	initial begin
		a = 0; b = 0; cin = 0; #20;
		cin = 1; #20;
		b = 1; cin = 0; #20;
		cin = 1; #20;
		a = 1; b = 0; cin = 0; #20;
		cin = 1; #20;
		b = 1; cin = 0; #20;
		cin = 1; #20;
	end
endmodule
```

5.1) Structural Systemverilog module for a 2-bit adder

```
module twobitadder(input logic a0, a1, b0, b1, cin, output logic sum0, sum1, cout);
	logic o;
	onebitfulladderstructural fulladder(a0, b0, cin, sum0, o);
	onebitfulladderstructural fulladder2(a1, b1, o, sum1, cout);
endmodule
```

5.2) Testbench module for a 2-bit adder module

```
module testbench2bitadderstructural();
	logic a0, a1, b0, b1, cin;
	logic sum0, sum1, cout;
	twobitadder dut(a0, a1, b0, b1, cin, sum0, sum1, cout);
	initial begin
		a0 = 0; a1 = 0; b0 = 0; b1 = 0; cin = 0; #20;
		cin = 1; #20;
		b1 = 1; cin = 0; #20;
		cin = 1; #20;
		b0 = 1; b1 = 0; cin = 0; #20;
		cin = 1; #20;
		b1 = 1; cin = 0; #20;
		cin = 1; #20;
		a1 = 1; b0 = 0; b1 = 0; cin = 0; #20;
		cin = 1; #20;
		b1 = 1; cin = 0; #20;
```

```
        cin = 1; #20;
        b0 = 1; b1 = 0; cin = 0; #20;
        cin = 1; #20;
        b1 = 1; cin = 0; #20;
        cin = 1; #20;
        a0 = 1; a1 = 0; b0 = 0; b1 = 0; cin = 0; #20;
        cin = 1; #20;
        b1 = 1; cin = 0; #20;
        cin = 1; #20;
        b0 = 1; b1 = 0; cin = 0; #20;
        cin = 1; #20;
        b1 = 1; cin = 0; #20;
        cin = 1; #20;
        a1 = 1; b0 = 0; b1 = 0; cin = 0; #20;
        cin = 1; #20;
        b1 = 1; cin = 0; #20;
        cin = 1; #20;
        b0 = 1; b1 = 0; cin = 0; #20;
        cin = 1; #20;
        b1 = 1; cin = 0; #20;
        cin = 1; #20;
    end
endmodule
```