Group: Power Puff - Step 1: Verilog & LaTeX Basics

Group Members: Zainab Khan and Jade Wilson

July 7, 2025

Contents

Chapter 1

Introduction

This report summarizes the implementation and simulation of several 1-bit logic gates and a 1x4-bit arithmetic shift circuit using Verilog. The gates implemented include NOT, NAND, and NOR gates. We tested each circuit by running testbenches and verified their behavior using waveform simulations in GTKWave. Screenshots of the waveforms are included in the figures section.

Chapter 2

NOT Gate Implementation

The NOT gate is a fundamental logic gate that outputs the logical negation of its single input.

2.1 Verilog Code

```
module not_1b (
    input wire a,
    output wire y
);
    assign y = ~a;
endmodule
```

2.2 Testbench

```
`timescale 1ns/1ps

module not_1b_test;
    reg a;
    wire y;

    not_1b uut (
        .a(a),
        .y(y)
    );
    initial begin
        $dumpfile("not_1b.vcd");
        $dumpvars(0, not_1b_test);

        a = 0; #10;
        a = 1; #10;

        $finish;
    end
endmodule
```

2.3 Waveform Results

Figure 2.1 shows the waveform output for the NOT gate. At 0 ns, input `a` is 0, resulting in output `y` being 1. This confirms the expected inversion behavior.

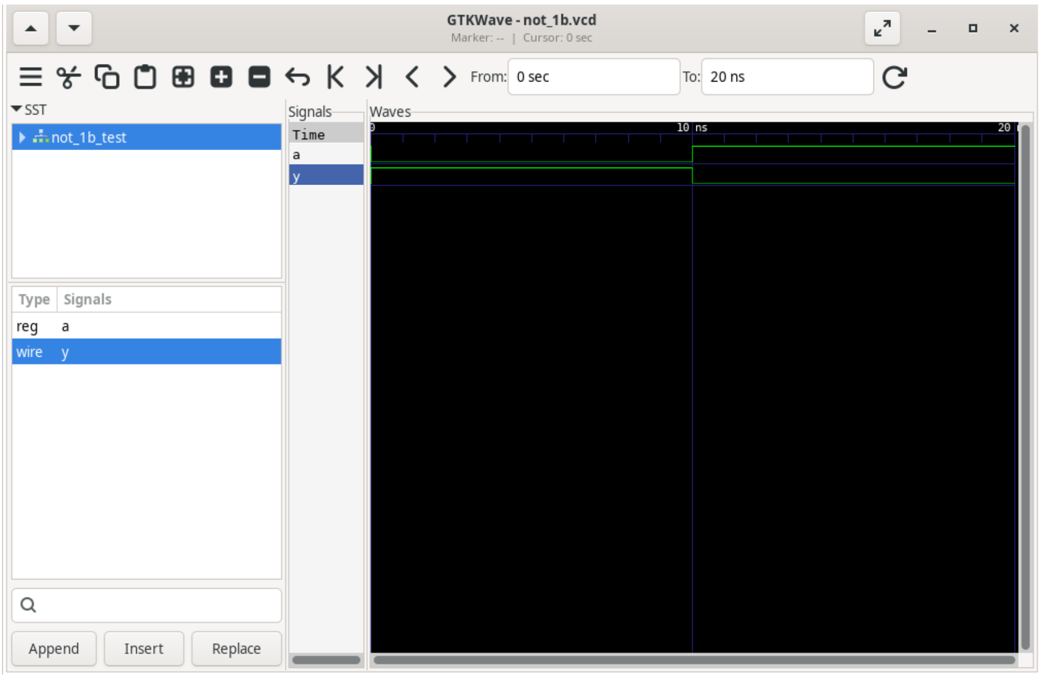

Figure 2.1: NOT Gate Waveform captured in GTKWave

Chapter 3

NAND Gate Implementation

The NAND gate outputs the negation of the AND operation on two inputs.

3.1 Verilog Code

```
module nand_1b (
    input wire a,
    input wire b,
    output wire y
);
    assign y = ~(a & b);
endmodule
```

3.2 Testbench

```
`timescale 1ns/1ps

module nand_1b_test;
    reg a, b;
    wire y;

    nand_1b uut (
        .a(a),
        .b(b),
        .y(y)
    );

    initial begin
```

```
        $dumpfile("nand_1b.vcd");
        $dumpvars(0, nand_1b_test);

        a = 0; b = 0; #10;
        a = 0; b = 1; #10;
        a = 1; b = 0; #10;
        a = 1; b = 1; #10;

        $finish;
    end
endmodule
```

3.3 Waveform Results

Figure 3.1 displays the simulation waveform for the NAND gate. The output correctly reflects the NAND logic for all input combinations.

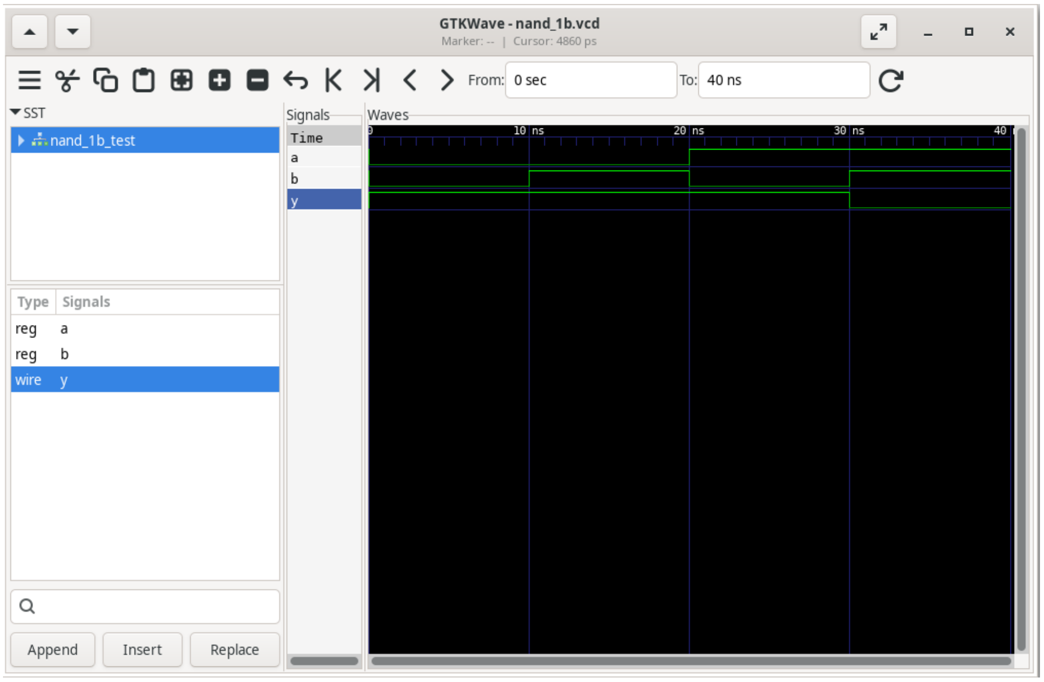

Figure 3.1: NAND Gate Waveform captured in GTKWave

Chapter 4

NOR Gate Implementation

The NOR gate outputs the negation of the OR operation on two inputs.

4.1 Verilog Code

```
module nor_1b (
    input wire a,
    input wire b,
    output wire y
);
    assign y = ~(a | b);
endmodule
```

4.2 Testbench

```
`timescale 1ns/1ps

module nor_1b_test;
    reg a, b;
    wire y;

    nor_1b uut (
        .a(a),
        .b(b),
        .y(y)
    );

    initial begin
```

```
        $dumpfile("nor_1b.vcd");
        $dumpvars(0, nor_1b_test);

        a = 0; b = 0; #10;
        a = 0; b = 1; #10;
        a = 1; b = 0; #10;
        a = 1; b = 1; #10;

        $finish;
    end
endmodule
```

4.3 Waveform Results

Figure 4.1 shows the waveform simulation for the NOR gate. The output confirms the NOR truth table.

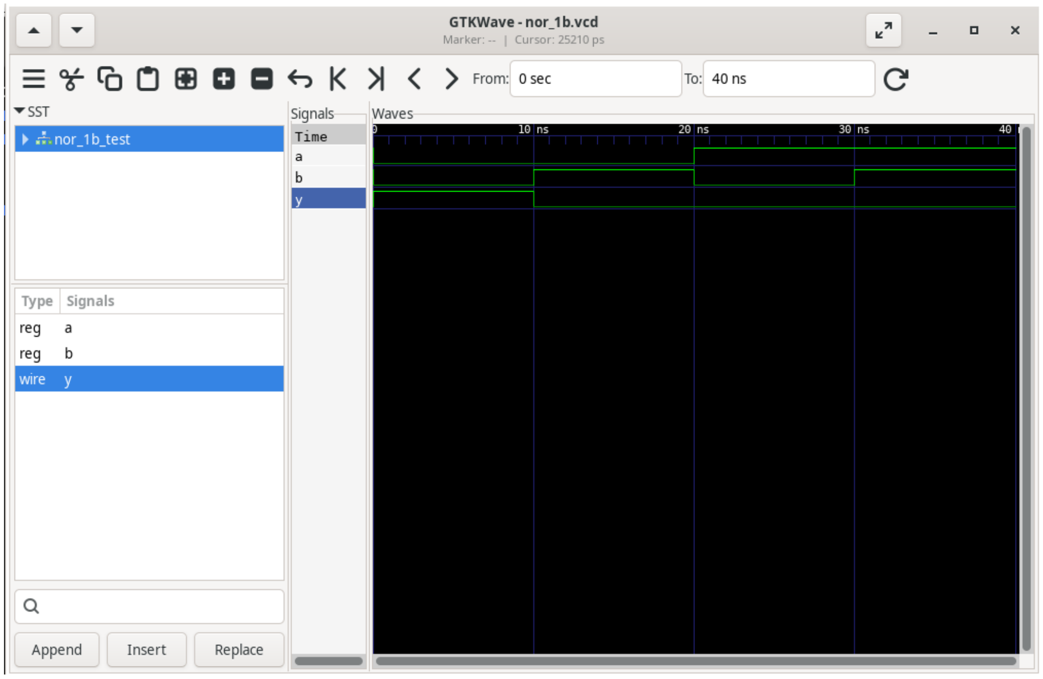

Figure 4.1: NOR Gate Waveform captured in GTKWave

Chapter 5

1x4-bit Arithmetic Shift Circuit

This circuit performs an arithmetic shift operation on a 4-bit input, shifting bits either left or right.

5.1 Verilog Code

```
module shift_4b (
    input wire [3:0] a,
    output wire [3:0] y
);
    assign y = a << 1;
endmodule
```

5.2 Testbench

```
`timescale 1ns/1ps

module shift_4b_test;
    reg [3:0] a;
    wire [3:0] y;

    shift_4b uut (
        .a(a),
        .y(y)
    );

    initial begin
        $dumpfile("shift_4b.vcd");
        $dumpvars(0, shift_4b_test);

        a = 4'b0001; #10;    // 1 << 1 = 2
        a = 4'b0011; #10;    // 3 << 1 = 6
        a = 4'b1010; #10;    // 10 << 1 = 4 (overflow bit lost)
        a = 4'b1111; #10;    // 15 << 1 = 14 (overflow bit lost)

        $finish;
    end
endmodule
```

5.3 Waveform Results

Figure 5.1 shows the waveform simulation for the 1x4-bit arithmetic shift circuit.

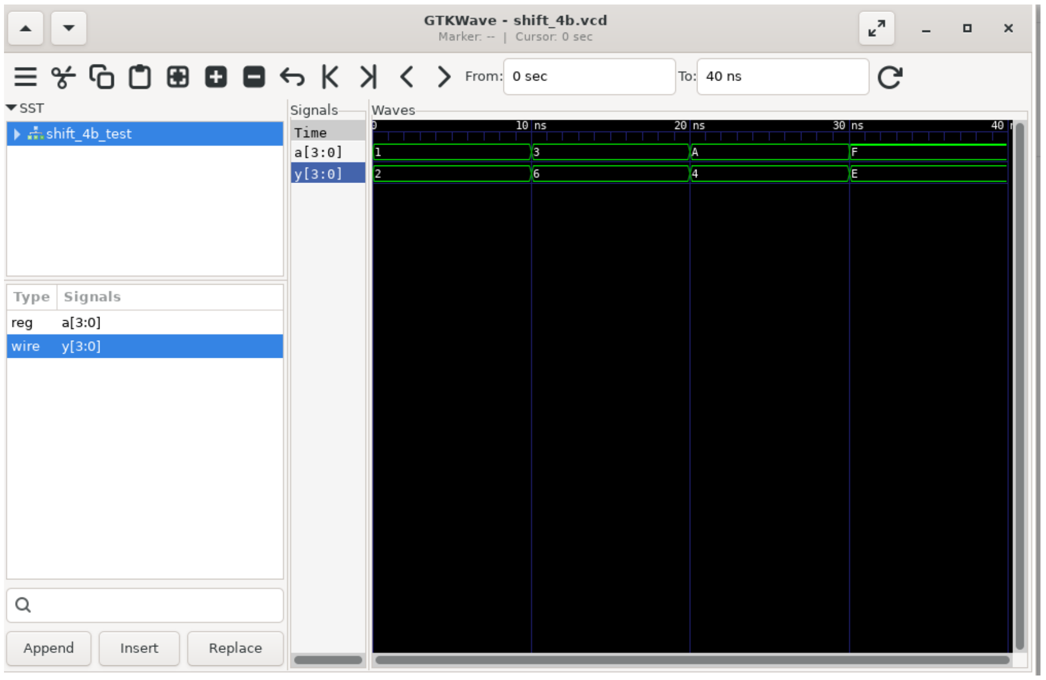

Figure 5.1: 1x4-bit Arithmetic Shift Circuit Waveform captured in GTK-Wave

Chapter 6

Conclusion

We have successfully implemented and tested basic logic gates (NOT, NAND, NOR) and a 1x4-bit arithmetic shift circuit using Verilog. Testbenches were used to verify functionality, and waveform simulations confirmed correct operation of each circuit. This step establishes the foundation for more complex designs in future project steps.