Hacettepe University

Computer Engineering Department

BM233 Logic Design Lab - 2022 Fall

Experiment 4 - Combinational Circuits in Verilog

December 10, 2022

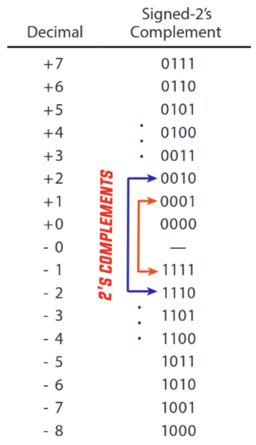
Student name:
İsmail Ulaş ÜNAL

Student Number:
b2200356001

1 Problem Definition

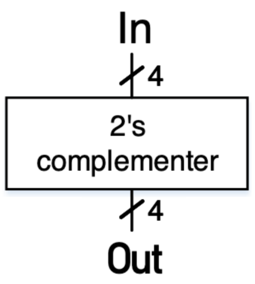
In this homework, we have to fulfill the tasks given in 3 different parts. Part 0 is about a implementing 4-bit 2's complementer with dataflow design approach. And test it with 16 different input combinations. In part 1 we need to implement a 4-bit 2x1 multiplexer with any design approach we want and create a testbench for all possible inputs. In part 2 we need to implement a single bit full adder with dataflow design approach and 4 bit-full adder with using single bit full adder by structural design approach with explicit association and test it with testbench In last part we need to make an adder / subtractor by using other part's circuits like 2's complementer, 2x1 multiplexer and 4 bit full adder. While doing this we need to use structural desgin approach and explicit association

Part 0:

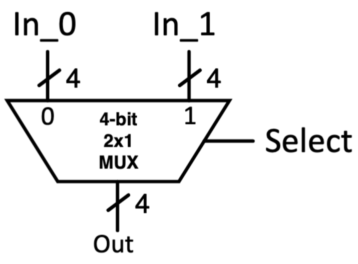


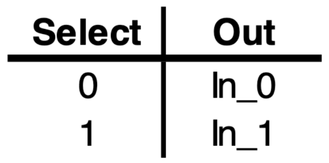
Decimal	Signed-2's Complement
+7	0111
+6	0110
+5	0101
+4	0100
+3	0011
+2	0010
+1	0001
+0	0000
- 0	—
- 1	1111
- 2	1110
- 3	1101
- 4	1100
- 5	1011
- 6	1010
- 7	1001
- 8	1000

2'S COMPLEMENTS



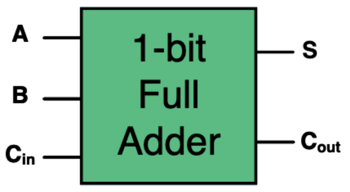
Part 1:

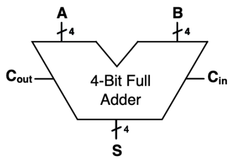


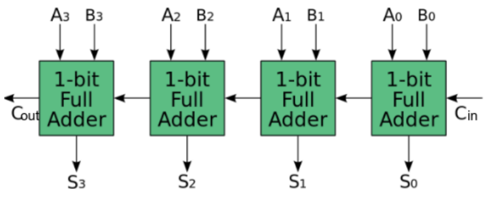


Select	Out
0	In_0
1	In_1

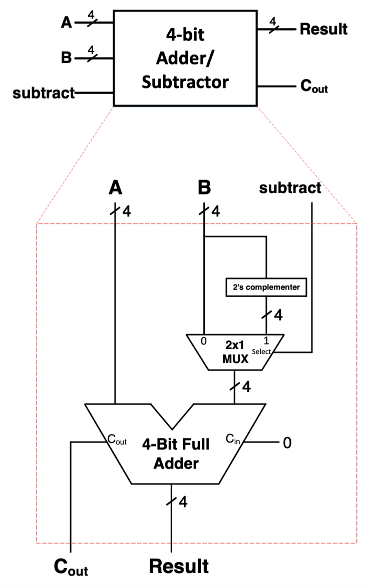
Part 2:







Part 3:



2 Solution Implementation

```
module two_s_complement(In,Out);
    input [3:0] In;
    output [3:0] Out;

    wire [3:0] not_In;

    assign not_In = ~In;
    assign Out = not_In + 1'b1;
endmodule
```

```
module four_bit_2x1_mux(In_1, In_0, Select, Out);
        input [3:0] In_1;
        input [3:0] In_0;
        input Select;
        output [3:0] Out;

        assign Out = Select ? In_1 : In_0;
endmodule
```

```
module full_adder(
    input A,
    input B,
    input Cin,
    output S,
    output Cout
);
    assign S = A ^ B ^ Cin;
    assign Cout = (A & B) | (Cin & (A ^ B));
endmodule
```

```
module four_bit_rca(
    input [3:0] A,
    input [3:0] B,
    input Cin,
    output [3:0] S,
    output Cout
);
    wire [2:0] Carries;

    full_adder A1(.A(A[0]), .B(B[0]), .Cin(Cin), .S(S[0]), .Cout(Carries[0]));
    full_adder A2(.A(A[1]), .B(B[1]), .Cin(Carries[0]), .S(S[1]), .Cout(Carries[1]));
    full_adder A3(.A(A[2]), .B(B[2]), .Cin(Carries[1]), .S(S[2]), .Cout(Carries[2]));
    full_adder A4(.A(A[3]), .B(B[3]), .Cin(Carries[2]), .S(S[3]), .Cout(Cout));
endmodule
```

```
module four_bit_adder_subtractor(A, B, subtract, Result, Cout);
    input [3:0] A;
    input [3:0] B;
    input subtract;
    output [3:0] Result;
    output Cout;

    wire [3:0] minus_B;
    wire [3:0] ready_B_for_adder_subtractor;
    wire Cin = 0;

    two_s_complement minus_version(.In(B), .Out(minus_B));
    four_bit_2x1_mux decide_plus_or_minus(.In_1(minus_B), .In_0(B), .Select(subtract)
    four_bit_rca make_calculations(.A(A), .B(ready_B_for_adder_subtractor), .Cin(Cin)



endmodule
```

3 Testbench Implementation

```
`timescale 1ns/10ps

module two_s_complement_tb;
```

```
    reg [3:0] In;
    wire [3:0] Out;

    integer In_int;

    two_s_complement test_two_s_complement (.In(In), .Out(Out));

    initial begin
        $dumpfile("two_s_complement.vcd");
        $dumpvars;

        for (In_int = 0; In_int < 16; In_int++) begin
            In = In_int;
            #50;
        end
    end
endmodule
```

```
`timescale 1ns/10ps
module four_bit_2x1_mux_tb;

        reg[3:0] In_1;
        reg[3:0] In_0;
        reg Select;
        wire[3:0] Out;

        four_bit_2x1_mux test_2x1_mux(.In_1(In_1), .In_0(In_0), .Select(Select), .Out

        initial begin
        $dumpfile("four_bit_2x1_mux.vcd");
        $dumpvars;

                for (integer select_int = 0; select_int < 2; select_int++) begin
                        Select = select_int;
                        for (integer In_1_int = 0; In_1_int < 16; In_1_int++) begin
                                In_1 = In_1_int;
                                for (integer In_0_int = 0; In_0_int < 16; In_0_int++)
                                        In_0 = In_0_int;
                                        #50;
                                end
                        end
                end

                $finish;


        end
endmodule
```

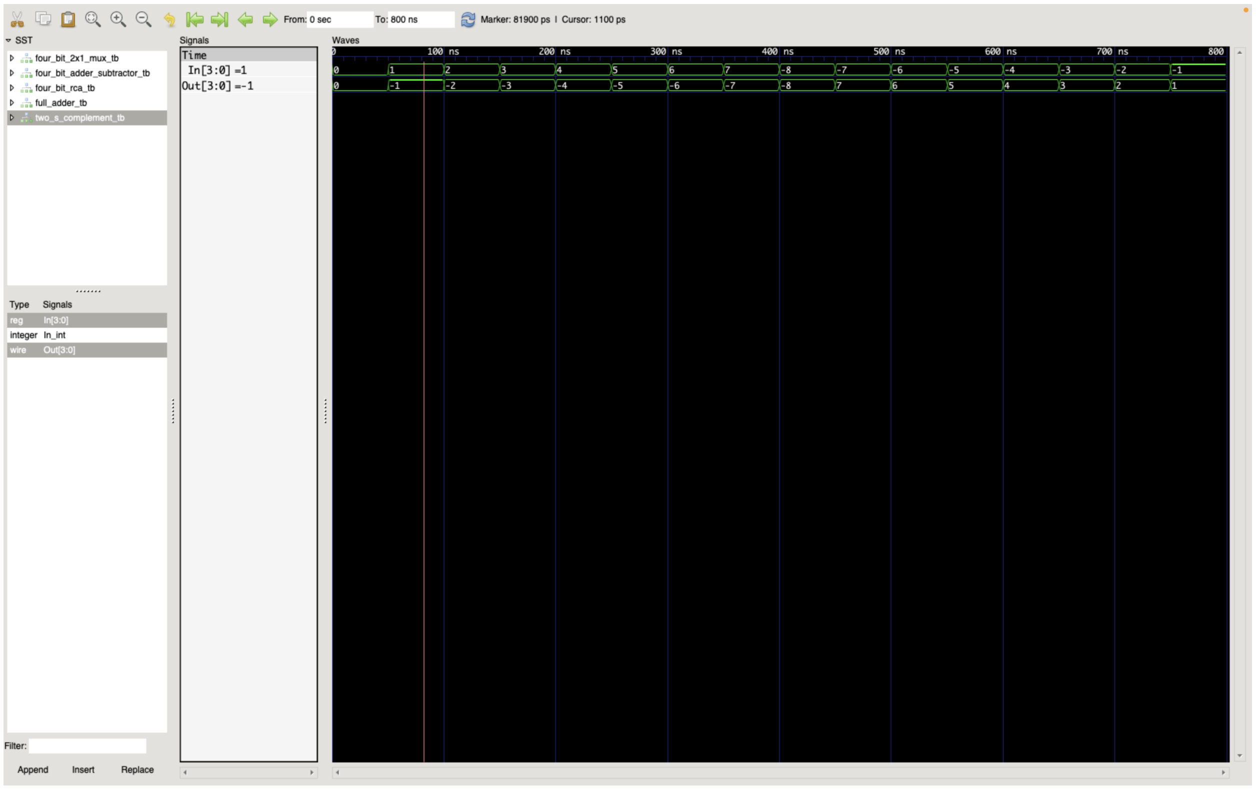
```verilog
`timescale 1 ns/10 ps
module full_adder_tb;

    reg A, B, Cin;
    wire S, Cout;

    full_adder test_full_adder(.A(A), .B(B), .Cin(Cin), .S(S), .Cout(Cout));

    initial begin
        $dumpfile("full_adder.vcd");
        $dumpvars;

        for (integer A_int = 0; A_int < 2; A_int++) begin
            A = A_int;
            for (integer B_int = 0; B_int < 2; B_int++) begin
                B = B_int;
                for (integer Cin_int = 0; Cin_int < 2; Cin_int++) begin
                    Cin = Cin_int;
                    #50;
                end
            end
        end
    end

endmodule
```

```verilog
`timescale 1 ns/10 ps

module four_bit_rca_tb;

  reg[3:0] A, B;
  reg Cin;

  wire[3:0] S;
  wire Cout;

  integer A_int, B_int, Cin_int;

  four_bit_rca test_four_bit_rca(.A(A), .B(B), .Cin(Cin), .S(S), .Cout(Cout));

  initial begin
    $dumpfile("four_bit_rca.vcd");
    $dumpvars;

    for (Cin_int = 0; Cin_int < 2; Cin_int++) begin
      Cin = Cin_int;
      for (A_int = 0; A_int < 16; A_int++) begin
        A = A_int;
        for (B_int = 0; B_int < 16; B_int++) begin
          B = B_int;
          #50;
        end
      end
    end



  end

endmodule
```

```
`timescale 1ns/1ps
module four_bit_adder_subtractor_tb;

    reg[3:0] A, B;
    reg subtract;

    output [3:0] Result;
    output Cout;

    integer A_int, B_int, subtract_int;

    four_bit_adder_subtractor test_four_bit_adder_subtractor(.A(A), .B(B), .subtract(


    initial begin
        $dumpfile("four_bit_adder_subtractor.vcd");
        $dumpvars;

        for (subtract_int = 0; subtract_int < 2; subtract_int++) begin
            subtract = subtract_int;
            for (A_int = 0; A_int < 16; A_int++) begin
                A = A_int;
                for (B_int = 0; B_int < 16; B_int++) begin
                    B = B_int;
                    #50;
                end
            end
        end


    end

endmodule
```

4 Results

Your explanations, results, screenshots...



From: 0 sec To: 800 ns Marker: 81900 ps | Cursor: 1100 ps

SST
- four_bit_2x1_mux_tb
- four_bit_adder_subtractor_tb
- four_bit_rca_tb
- full_adder_tb
- two_s_complement_tb

Type	Signals
reg	In[3:0]
integer	In_int
wire	Out[3:0]

```
Time
 In[3:0] =1
Out[3:0] =-1
```

Figure 1: Two's complement

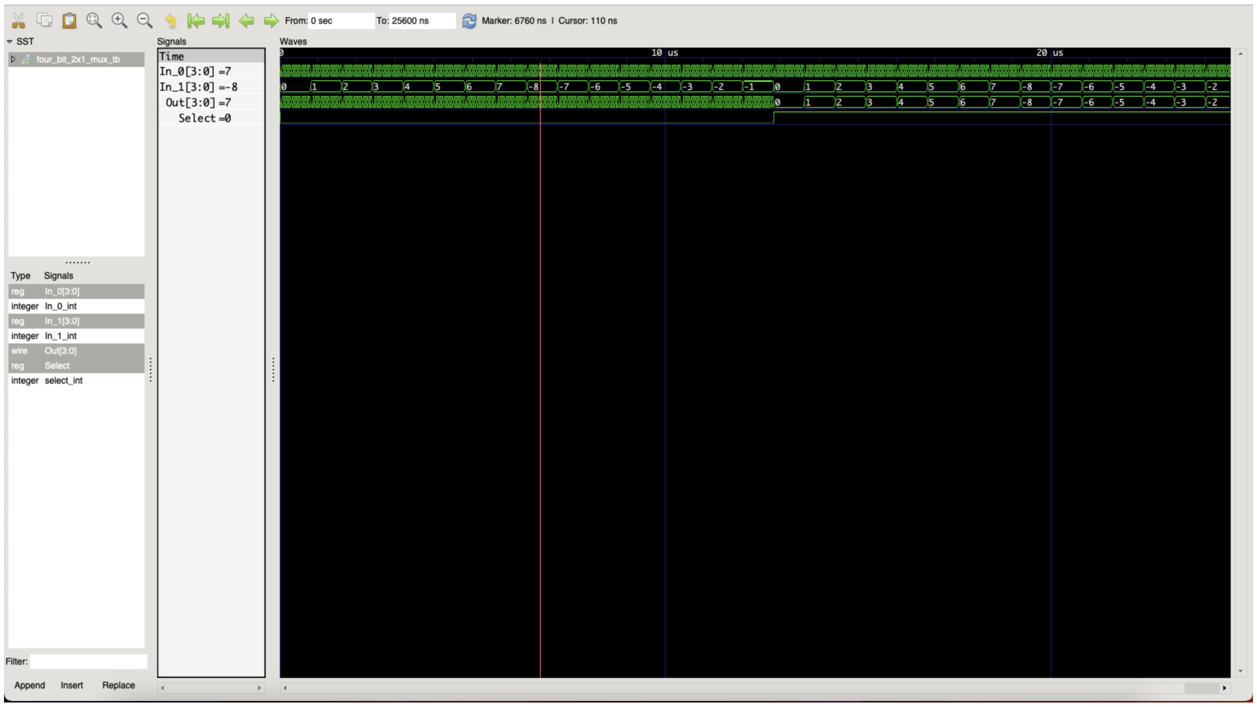


Figure 2: 4-bit 2x1 multiplexer

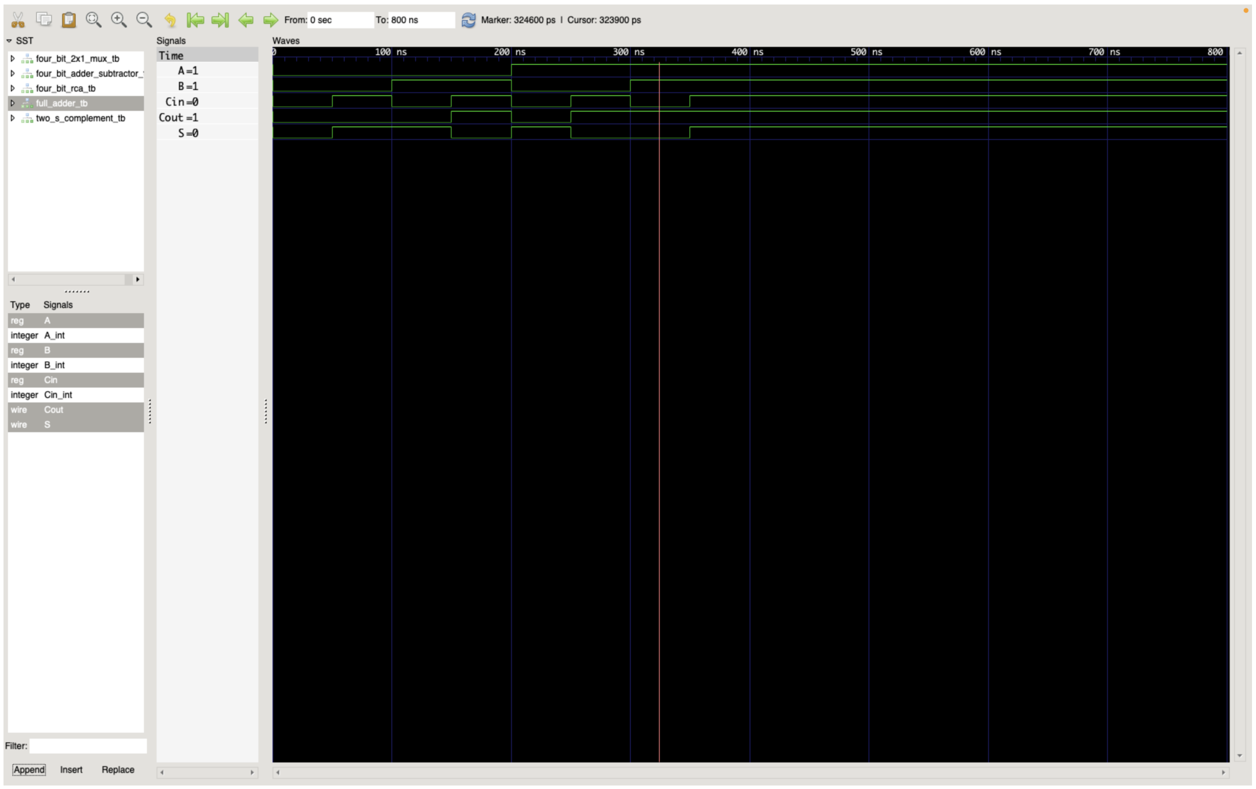
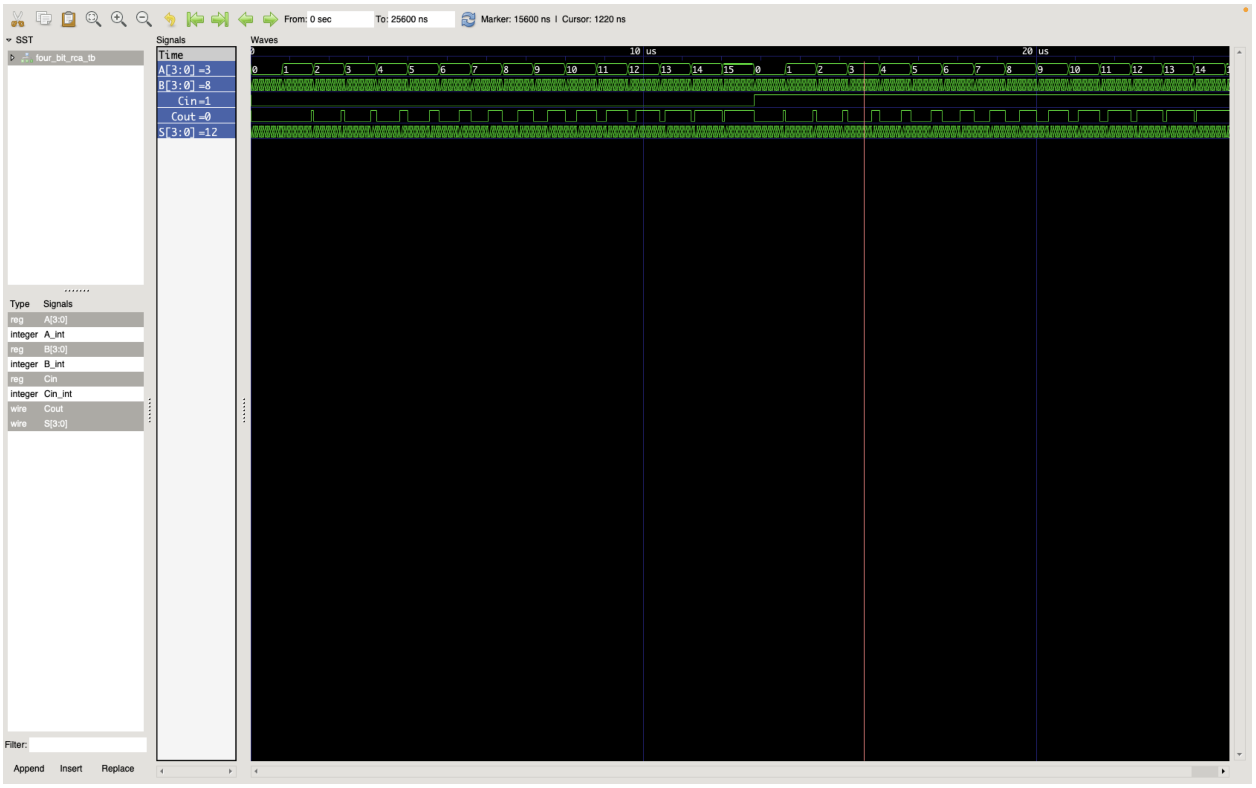


Figure 3: 1-bit Full adder

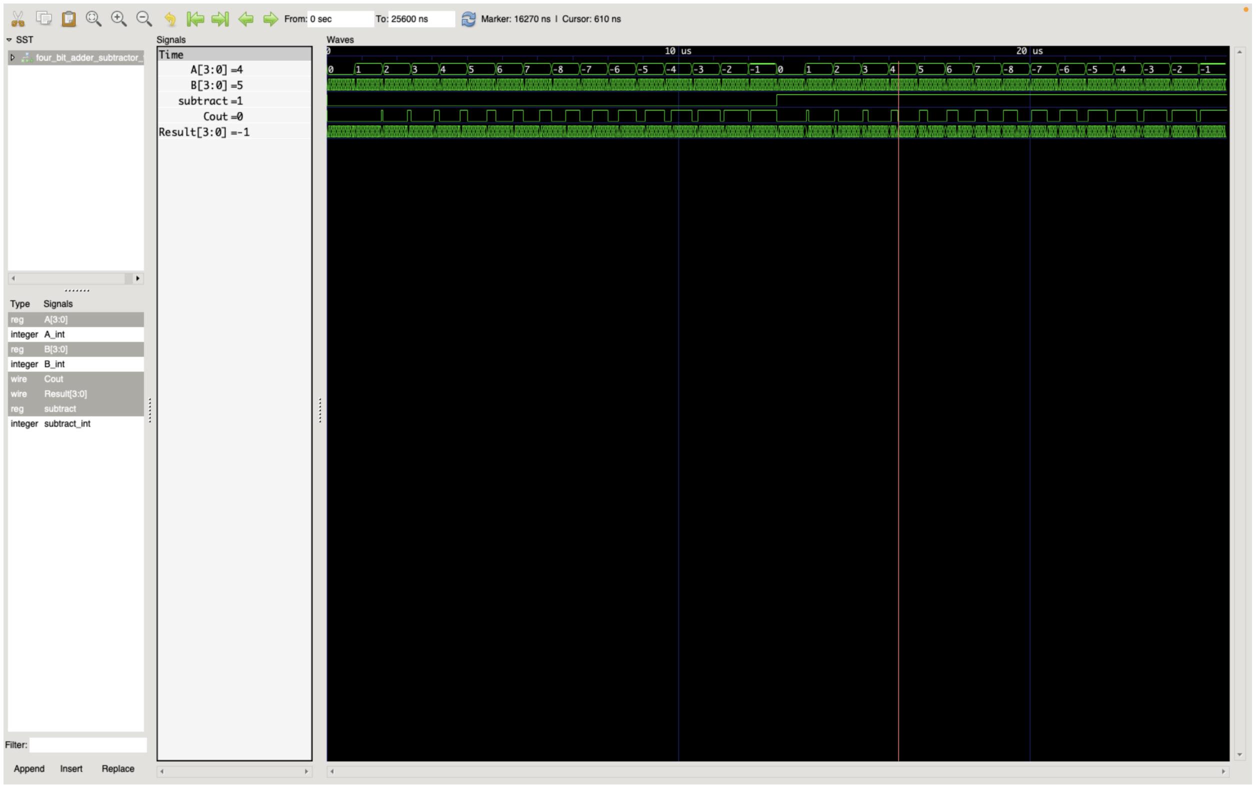


From: 0 sec To: 25600 ns Marker: 15600 ns | Cursor: 1220 ns

Type	Signals
reg	A[3:0]
integer	A_int
reg	B[3:0]
integer	B_int
reg	Cin
integer	Cin_int
wire	Cout
wire	S[3:0]

```
Time
A[3:0] =3
B[3:0] =8
Cin =1
Cout =0
S[3:0] =12
```

Figure 4: Four bit RCA



From: 0 sec To: 25600 ns Marker: 16270 ns | Cursor: 610 ns

```
Time
      A[3:0] =4
      B[3:0] =5
  subtract =1
      Cout =0
Result[3:0] =-1
```

Type	Signals
reg	A[3:0]
integer	A_int
reg	B[3:0]
integer	B_int
wire	Cout
wire	Result[3:0]
reg	subtract
integer	subtract_int

Figure 5: 4-bit adder subtractor