Implementing a 64-bit Signed Binary Multiplier & Divider Circuit

Name	Naveen Kumar Bhuthakatanahalli Ramalingaiah	SJSU ID	011439792
Email	naveenkumar.bhuthakatanahalliramalingaiah@sjsu.edu	Phone	6692659782

Executive Summary (50 to 80 words)

The multiplier and divider plays an important role in CPU unit. The Signed Binary multiplier and divider circuit calculates multiplication or division for 32 bit operands. 64-bit dividend is used while calculating quotient and reminder. The multiplication and division is operated using different states. There are 4 modules which are involved for design, three are designed for 32-bit addition. The control unit is responsible for performing operation related to shifting and providing input to adder circuit. The test inputs are provided using testbench. A combination of different positive and negative values are given as input through testbench. This design is analyzed for power, area and delay. It is simulated, designed and synthesized on Synopsys Design compiler. The tool works on 250nm technology and post synthesis which is observed using NCVerilog provided by Cadence.

I. General Project Information

Table I.1: List of EDA Tools Used

EDA Tool Name	Company	You Used it for
VCS	Synopsys	Simulation & test
NCVerilog	Cadence	Simulation & test
Design Vision	Synopsys	Synthesis & optimization

Table I.2: List of Libraries Used

Library file name	Used with (EDA tool name)	The library is at (directories on eecad systems)
WCCOM	Synopsys Design Compiler	/apps/Toshiba/sjsu/synopsys/tc240c/ tc240c.db_WCCOM25
BCCOM	Synopsys Design Compiler	/apps/Toshiba/sjsu/synopsys/tc240c/ tc240c.db_BCCOM25
NOMIN	Synopsys Design Compiler	/apps/Toshiba/sjsu/synopsys/tc240c/ tc240c.db_NOMIN25

Table I.3: List of Verilog Modules (both design and test modules)

Module Name	Input port	Output Port	Short Description
multiplier	opera1,opera2,clock, muordi, start, reset	valid,result	Signed Multiplier module
divider	opera1,opera2,clock, muordi, start, reset	valid, result	Signed Divider module
add32	a,b,cin	sum,cout	32 bit adder module
fulladd	a,b,cin	sum,cout	1 bit full adder module
adder	a,b	sum,carry	Half adder module
Test	valid,result	opera1,opera2,clock, muordi, start, reset	Testbench for multiplier and divider circuit

II. Implementation Overview

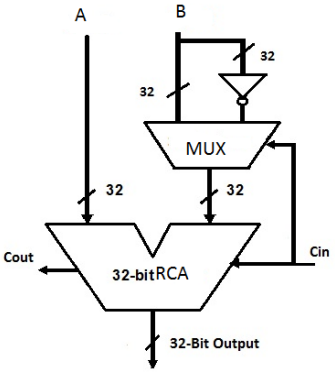


Figure II.1: Block Diagram of 32-bit ALU unit

The ALU input operands are of 32-bit. One input is from Multiplicand register and the other input is upper half of product/multiplier register. 32-bit multiplier or 64-bit dividend is stored in 64-bit register. The control unit is responsible for shifting the register and providing values for addition or subtraction. At any given time only two 32 bit operands are given to ALU unit. The ALU unit consists of 3 modules, 32-bit adder which is further designed with full adder and half adder. The ALU can perform 32-bit addition and 32-bit subtraction. The carry input pin of adder module controls the subtraction or addition operation. The result is written back to the Product/Multiplier register.

Figure II.2: State Transition Diagram of Sequential Multiplier

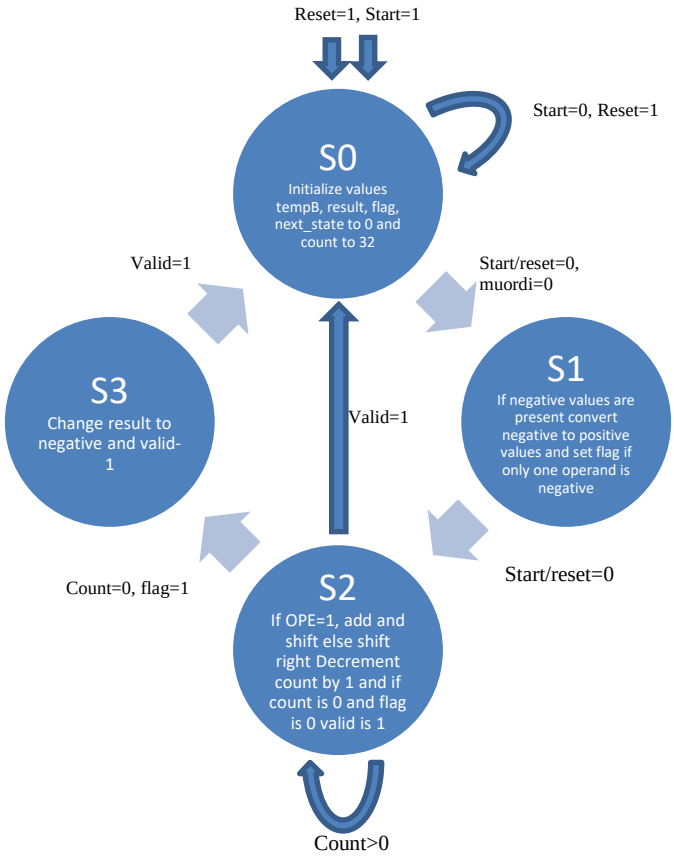
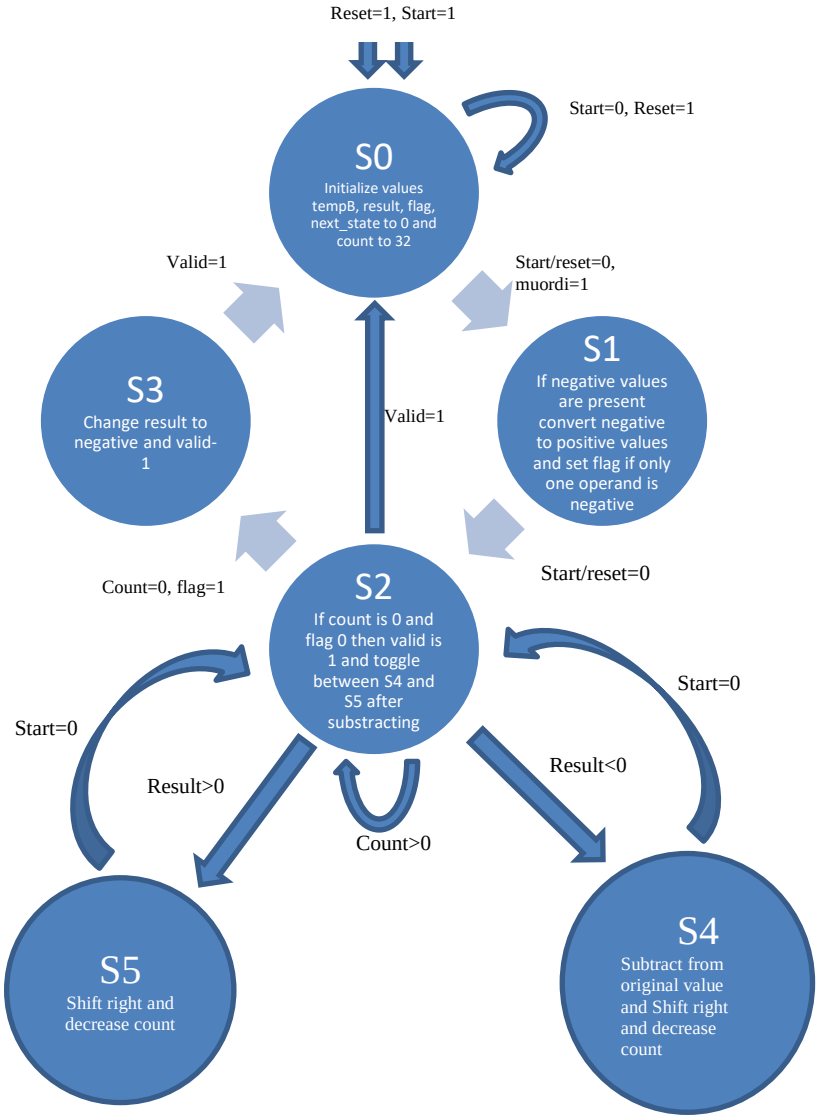


Figure II.3: State Transition Diagram of Sequential Divider



Discuss the timing characteristics of the multiplier and divider (less than 100 words)

The number of clock cycles needed for multiplication is less when compared to divider design. When subtracting the divisor register from upper half of the remainder/quotient register, if result is negative then the value should be restored. This operation is done in state S4. Thus time for division includes more clock cycles. When inputs or operands are negative, the number of clock cycles would increase further. Converting these negative operands to positive would consume

more clock cycles. When the reset bit or start bit is set control will move to state S0, the design would start operation when start bit turns to 0 and the control shifts to next state. The final result would be in result register after the valid pin is high.

III. RTL-Level (Pre-synthesis) Simulations/Tests

Short descriptions of testbench(es) *that were used to test your modules during the project implementation process (less than 50 words)*

Multiplier Testbench:
The multiplier module testbench tests for all 4 operands types. Both positive, both negative or any one operand negative. The testbench consists of a display command which displays the output in decimal format when valid pin is set. Integer register is used to read data from quotient/remainder register and display in decimal format. For each input value, a delay is provided for the operation to complete. The clock is generated in the testbench. The test case includes reset and start bit functionalities.

Divider Testbench:
The result from divider is stored in both the upper and lower half of 64-bit result register. Integers are used for reading data from quotient and remainder part of result register. The divider testbench includes reset, start bits which are sensed at the positive edge of the clock. If reset is 1 then the system would move to initial state. Different operands are given as inputs, both operands positive, negative and any one operand negative. The result is read when valid pin is high and this is displayed through display command. The clock for design is generated in testbench.

For each table in this section, you need to show simulation waveforms that can demonstrate major coverage in testing of your implementation and to support your expected results listed in the tables. Try to justify yourself if what parts of the tests and the changes from one value to other value that are important and should be included here in the report. ***You should make some notes on the waveforms to show and explain your results and analyses.***

Table III.1 – Four Selected Test Data for Multiplier Circuit

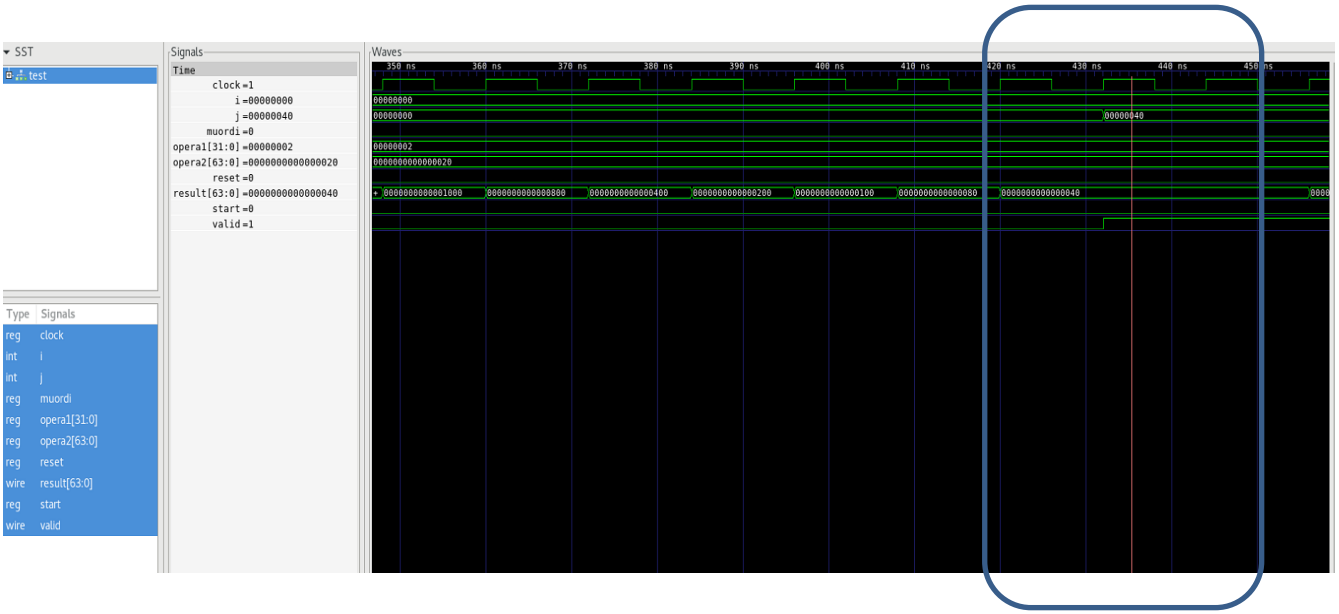
Test Case	`opera1` (hex)	`opra2` (hex)	`result (hex)`
1	00000002	0000000000000020	0000000000000040
2	FFFFFFFE	0000000000000020	00000000FFFFFFC0
3	00000002	FFFFFFFFFFFFFFE0	00000000FFFFFFC0
4	FFFFFFFE	FFFFFFFFFFFFFFE0	0000000000000040

Figure III.1a: RTL simulation waveform that contains test case #1

INPUT

The input values which are given through the testbench are displayed in the waveform. This is followed by operations of multiplication using adder module. The result register stores result from adder module.

OUTPUT



The output is present in result register. The lower half of register consists of the result value. The integer j will display the output in decimal.

Figure III.1b: RTL simulation waveform that contains test case #2

INPUT

Opera1 is set with negative value and opera2 is set with positive value. The opera1 value will be converted to positive value for operation and when count is 0, 2's complement of the output will be taken.

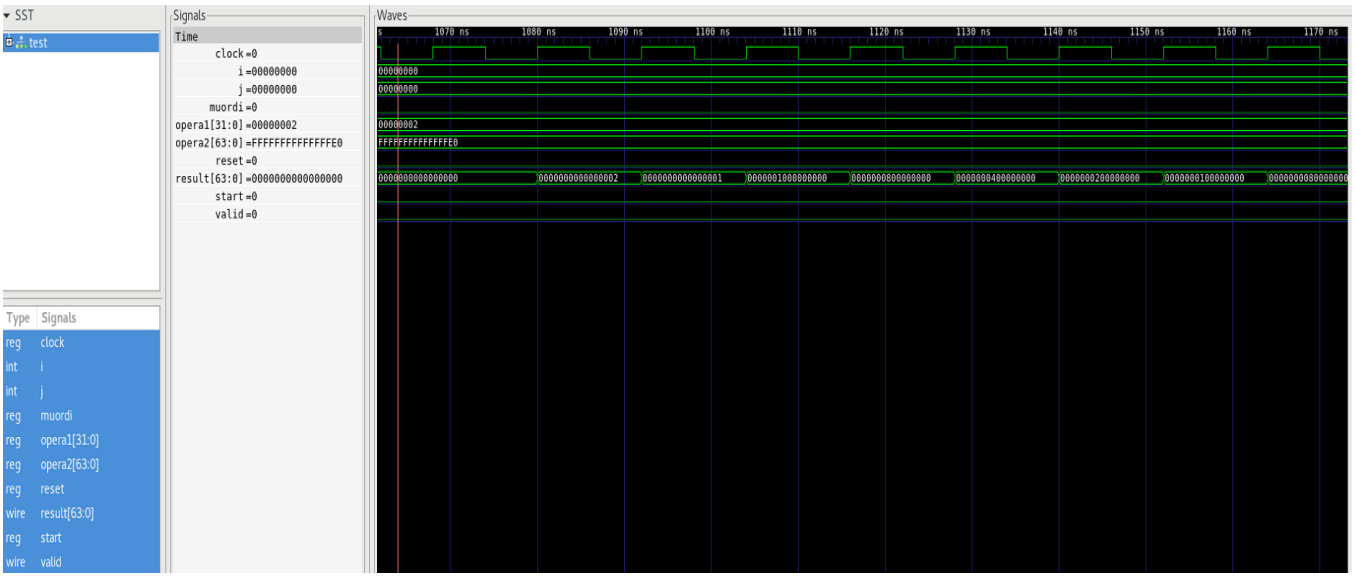
OUTPUT

The output register result contains the negative result as one input operand was negative.

Figure III.1c: RTL simulation waveform that contains test case #3

INPUT

Opera2 is set to negative and Opera1 to positive.



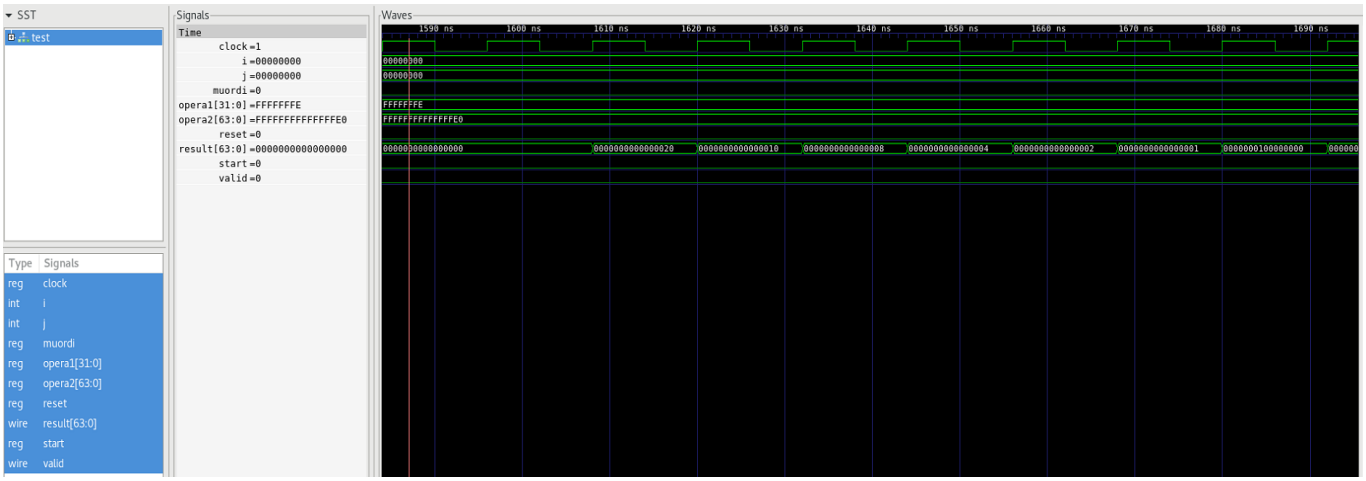
OUTPUT

The output value is present in result register. The lower half of the result register consists of 2's complement as one input operand is negative.

Figure III.1d: RTL simulation waveform that contains test case #4

INPUT

This test case consists of operands when both are negative. 2 clock cycles are required for converting the values to positive. One clock cycle for each operand



OUTPUT

The output consists of a positive value as multiplication of 2 negative numbers is positive. Additional clock cycles are required for this operation.

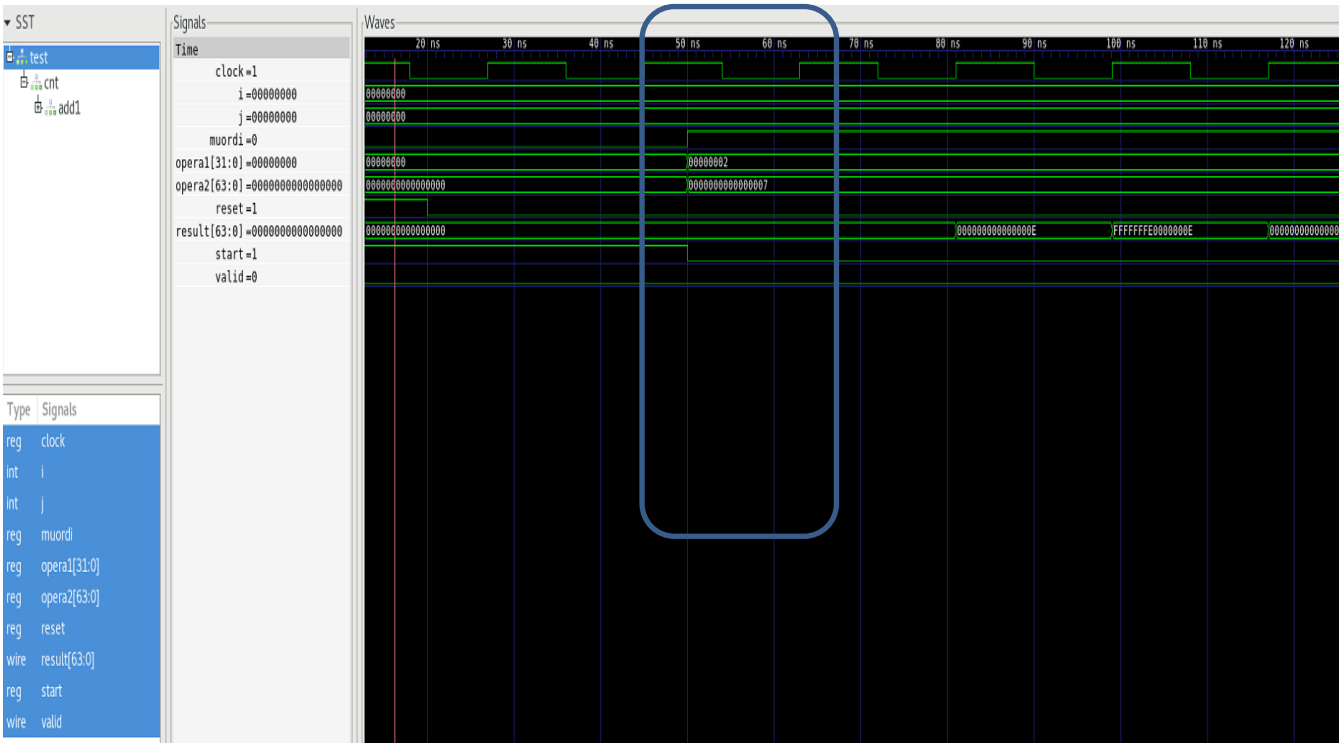
Table III.2 – Four Selected Test Data for Divider Circuit

Test Case	`opera1` (hex)	`opra2` (hex)	`result (hex)`
1	00000002	0000000000000007	0000000100000003
2	00000002	FFFFFFFFFFFFFFF9	00000001FFFFFFFD
3	FFFFFFFE	0000000000000007	00000001FFFFFFFD
4	FFFFFFFE	FFFFFFFFFFFFFFF9	0000000100000003

Figure III.2a: RTL simulation waveform that contains test case #1

INPUT

When both input operands are positive, the state 0 is activated and all values are assigned to 0. The division operation starts when start bit is set to 0.



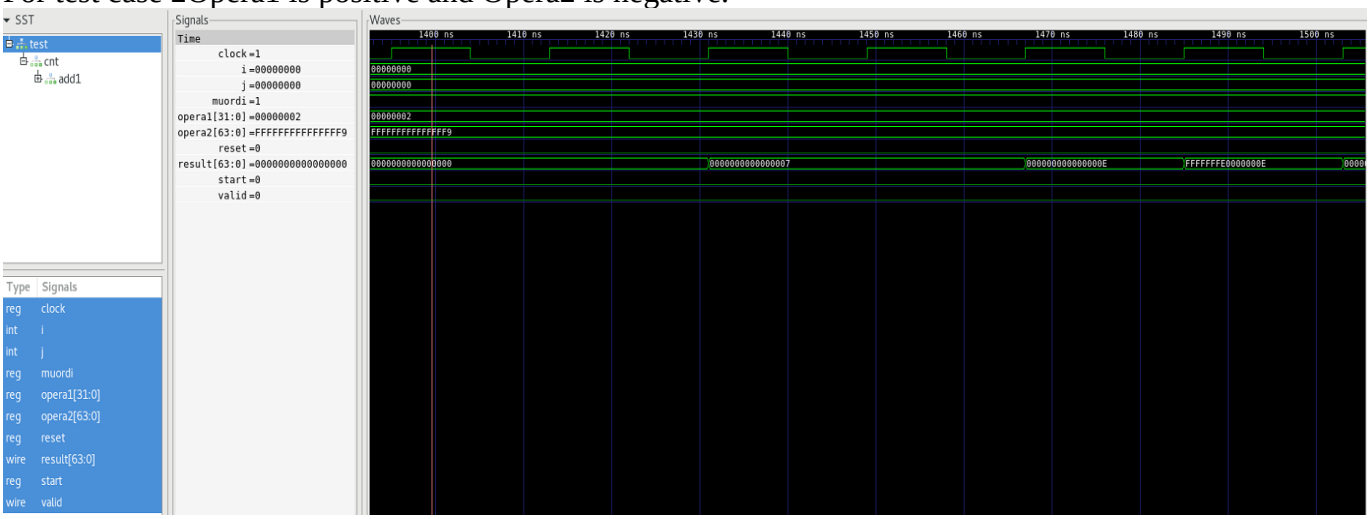
OUTPUT

The output of this particular division is stored in result register. The lower half consists of quotient and the upper half consists of reminder. This will later move to idle state.

Figure III.2b: RTL simulation waveform that contains test case #2

INPUT

For test case 2Opera1 is positive and Opera2 is negative.



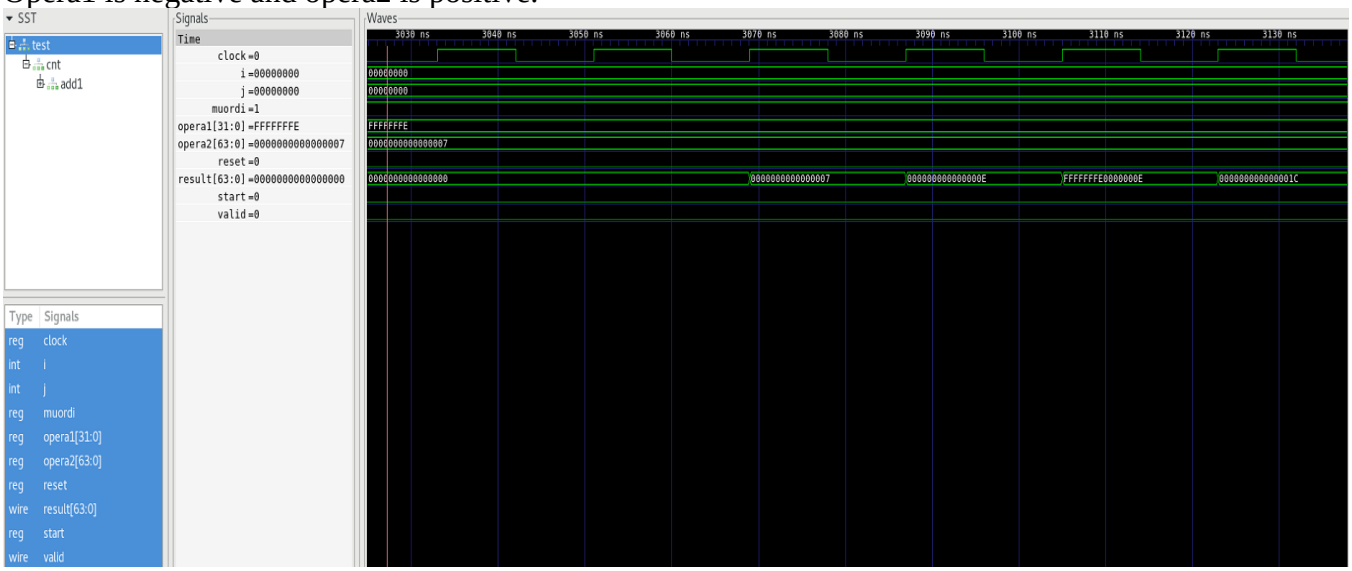
OUTPUT

The output consists of a 2's compliment as one of the input is negative.

Figure III.2c: RTL simulation waveform that contains test case #3

INPUT

Opera1 is negative and opera2 is positive.



OUTPUT

The output consists of a 2's compliment as one input is negative while the other is positive and lower half consists of quotient and upper half consists of a reminder.

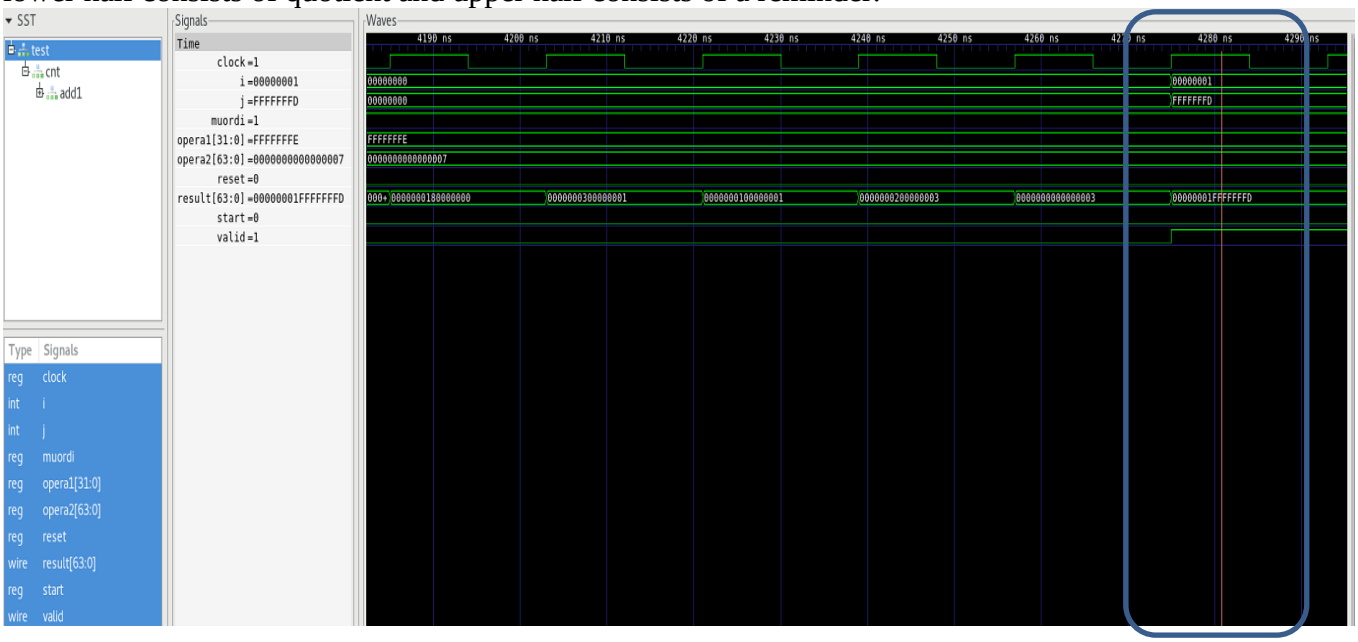
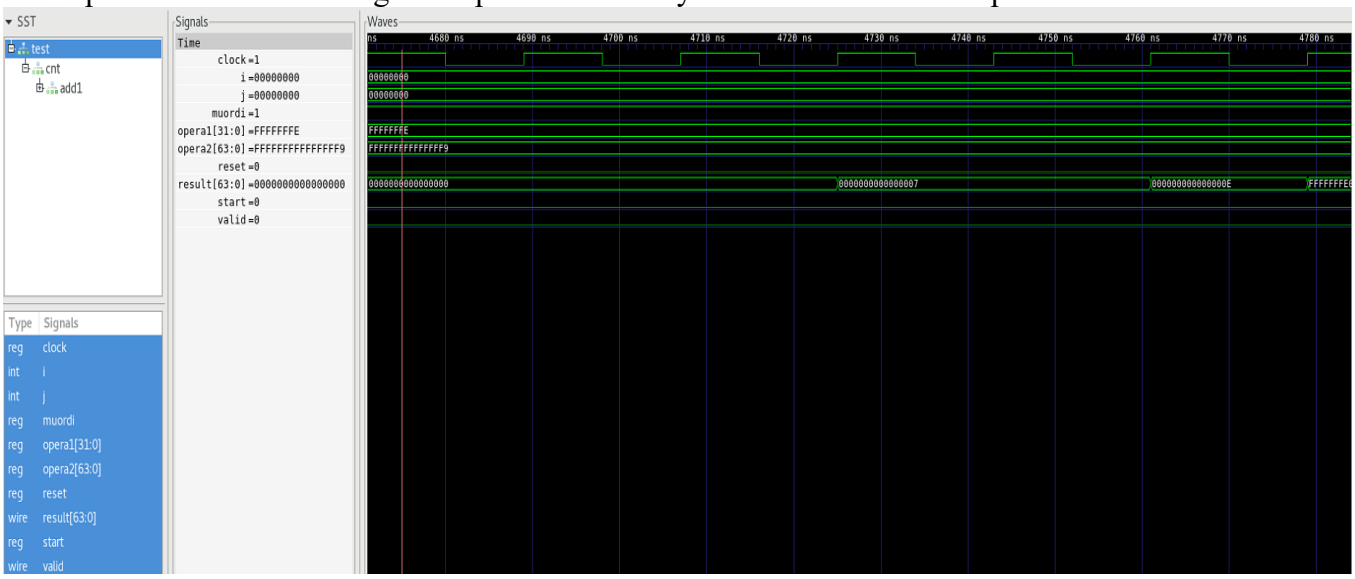


Figure III.2d: RTL simulation waveform that contains test case #4

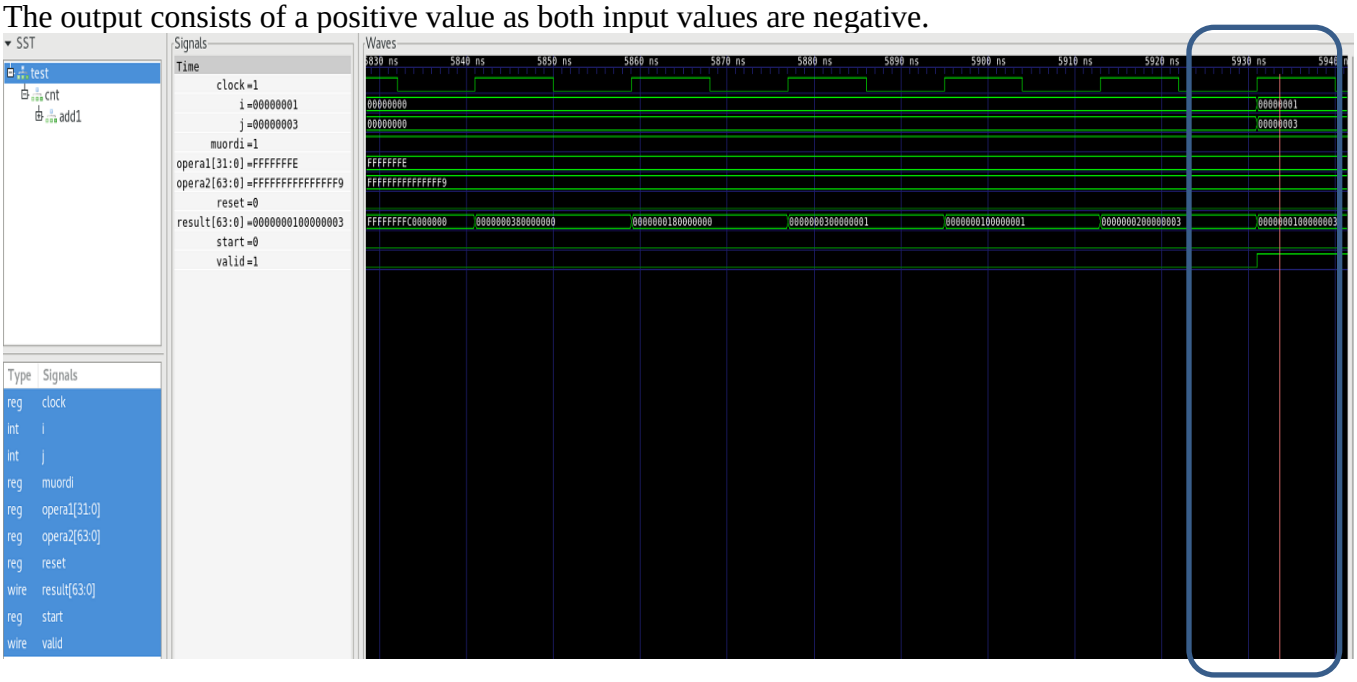
INPUT

The input consists of both negative operands and they are stored in a 2's complement format.



OUTPUT

The output consists of a positive value as both input values are negative.



IV. Synthesis and Optimizations

In each table below, show 6 selected trials that you have gone through in synthesizing and optimizing your implementations. ***You may try tens of trials, but you need to select 6 to show in the tables****. Select the trials that can represent your synthesizing and optimizing efforts so that you can conclude that the last trial (trial #6) is the one you finally used as the most optimized circuit.* ***Save the data from your (many) trials and use them to plot the "Area vs. Delay" curves in figures IV, which need many more data points.***

Table IV.1: Synthesis Constraints and Results for Multiplier

Trial #	**Your design constraint settings** (such as area, clock, delay, etc.)	**Results after synthesis** (such as area, time slack, power, etc.)
1	Area = 0 Clock = 12ns	Area = 2069.5 Time Slack = 0 Power = 2.6117 mW
2	Area = 0 Clock = 15ns	Area = 1781.5 Time Slack = 0 Power = 2.0153 mW
3	Area = 0 Clock = 20ns	Area = 1793 Time Slack = 0.99 Power = 1.5361 mW
4	Area = 2000 Clock = 25ns	Area = 1854.5 Time Slack = 7.85 Power = 1.1673 mW

5	Area = 2000 Clock = 10ns	Area = 2208.5 Time Slack = -0.91(VIOLATED) Power = 3.7803 mW
6	Area = 0 Clock = 13ns	Area = 1875 Time Slack = .01 Power = 2.4192 mW

Figure IV.1: Area vs. Delay Curve of Multiplier

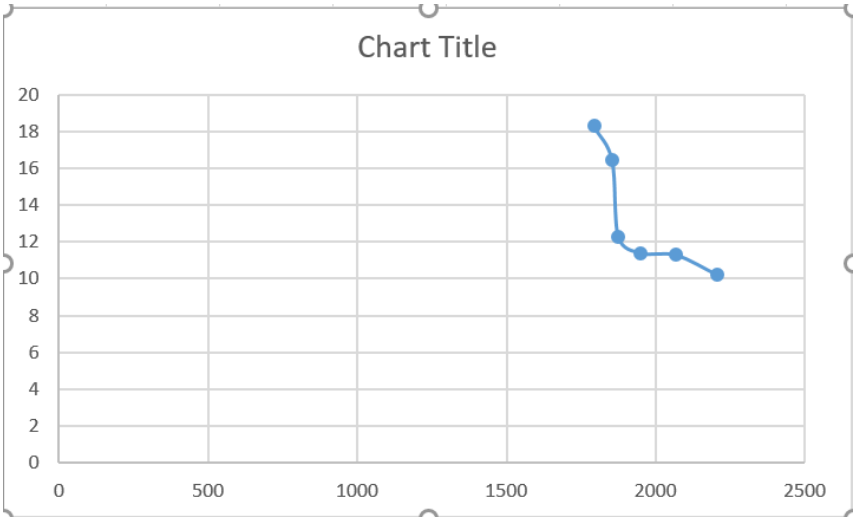
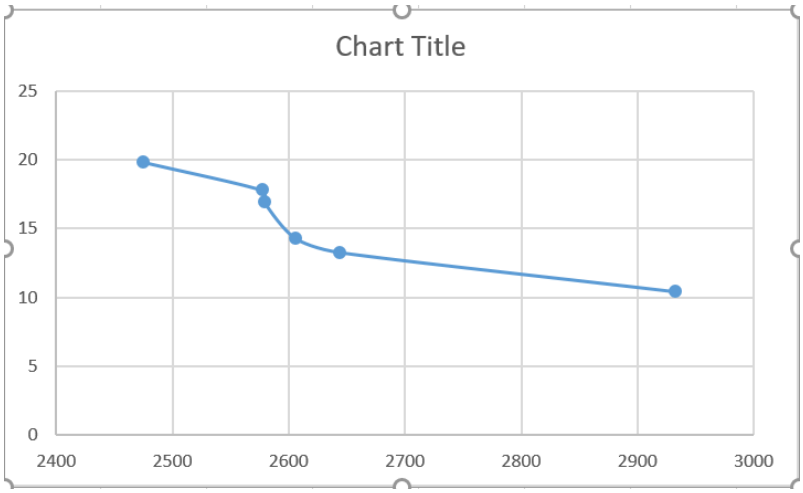


Table IV.2: Synthesis Constraints and Results for Divider

Trial #	**Your design constraint settings** (such as area, clock, delay, etc.)	**Results after synthesis** (such as area, time slack, power, etc.)
1	Area = 0 Clock = 18ns	Area = 2579 Time Slack = 0.38 Power = 2.3461 mW
2	Area = 0 Clock = 20ns	Area = 2577.5 Time Slack = 1.5 Power = 2.0939 mW
3	Area = 2000 Clock = 25ns	Area = 2475 Time Slack = 4.65 Power = 1.8282 mW
4	Area = 0 Clock = 15ns	Area = 2606.5 Time Slack = 0.01 Power = 2.7838 mW
5	Area = 1000 Clock = 10ns	Area = 2932.5 Time Slack = -1.15 (VIOLATED) Power = 4.9679 mW
6	Area = 0 Clock = 14ns	Area = 2644 Time Slack = 0.01 Power = 3.0844 mW

Figure IV.2: Area vs. Delay Curve of Divider



Briefly discuss and analyze the relationships in timings, areas, and other related constraints of your optimized multiplier and divider circuits (less than 100 words)

The area required for divider circuit is more when compared to multiplier module. The time taken for result to be present in product multiplier register is more for divider circuit and it is less for multiplier circuit. This is due to the clock cycles divider operation is encountering. The clock time for synthesis is less for multiplier circuit and it is high for divider circuit. As shown in the graph, the delay is decreasing for increased area. Both multiplier and divider circuit delay will decrease with increasing area. The area and power is optimized with maximum effort.

V. Gate-Level (Post-synthesis) Dynamic Simulations/Tests

For each table in this section, you need to show simulation waveforms that can demonstrate major coverage in testing of your implementation and to support your measured performance shown in the tables. The test data in these tables should be the same as test data in Section III tables, however for post-synthesis simulation, you need to select data points and adjust your timing scale of the displayed waveforms such that you are able to show the fluctuation of the immediate data through the circuits based on the values of the input operands and the correctness of the logic functions. ***You should make some notes on the waveforms to show and explain the timing delays that result in data fluctuations as you observed from the simulation result****.*

Table V.1 – Four Selected Test Data for Multiplier Circuit

Test Case	`opera1` (hex)	`opra2` (hex)	`Time Delay (in time unit)`
1	**00000002**	**0000000000000020**	**35 clock+0.6ns 35 * 12+0.6ns 420+0.6ns**
2	**FFFFFFFE**	**0000000000000020**	**35 clock+0.6ns 35 * 12+0.6ns 420+0.6ns**
3	**00000002**	**FFFFFFFFFFFFFFE0**	**35 clock+0.6ns 35 * 12+0.6ns 420+0.6ns**
4	**FFFFFFFE**	**FFFFFFFFFFFFFFE0**	**35 clock+0.6ns 35 * 12+0.6ns 420+0.6ns**

Figure V.1a: Gate-level simulation waveform that contains test case #1

INPUT

The inputs to circuit is given after a specified delay as propagation delay of gates will vary with different gates.

OUTPUT

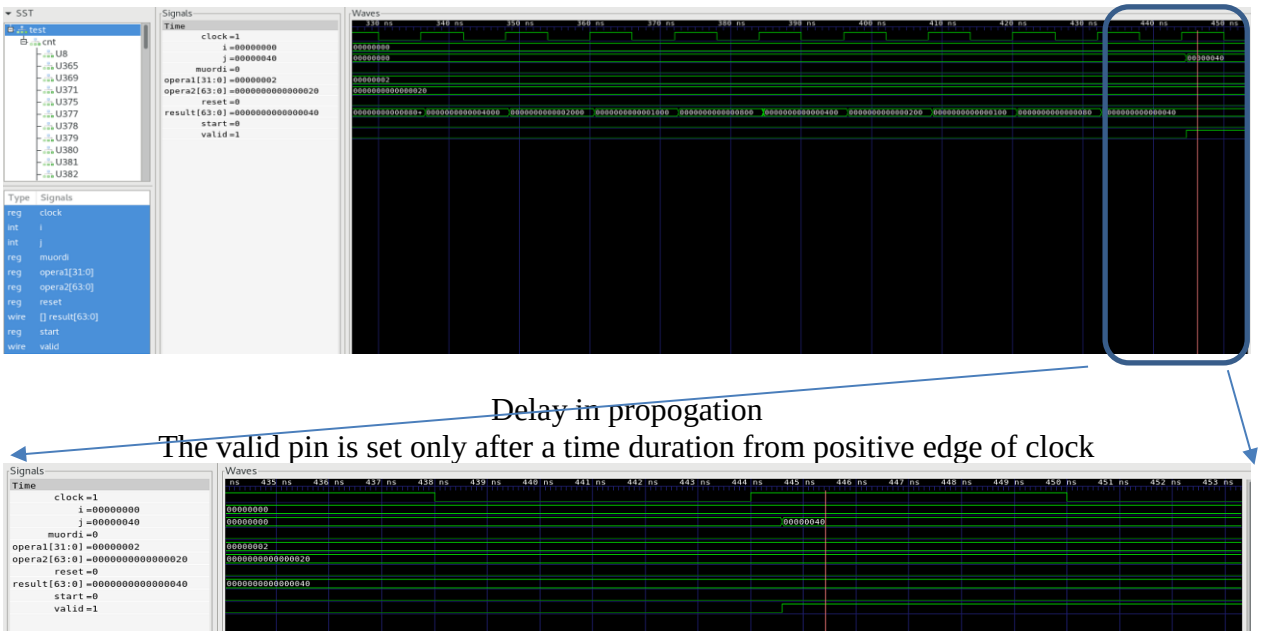
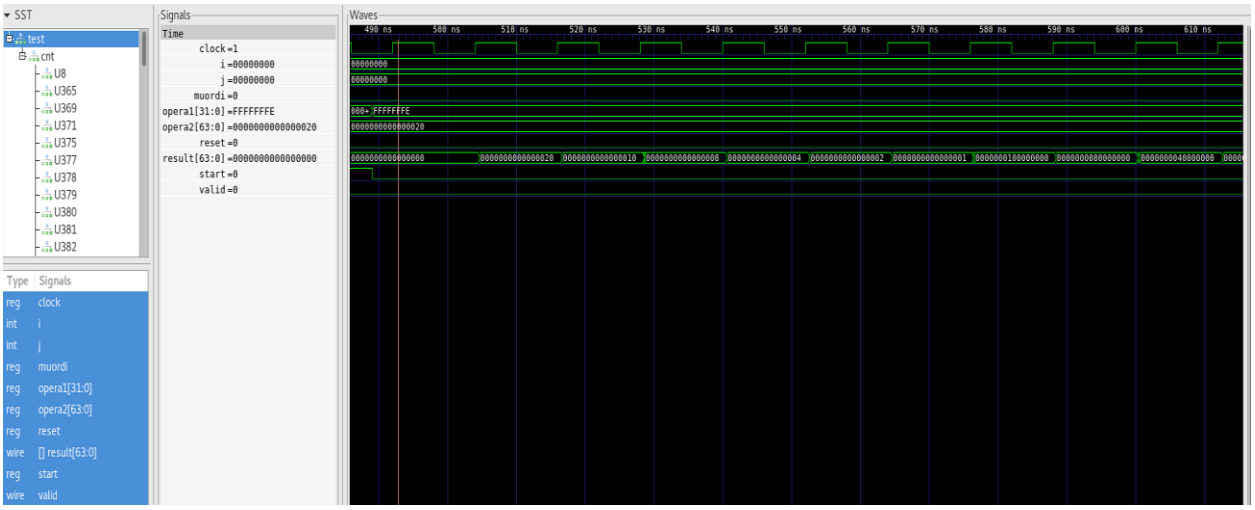


Figure V.1b: Gate-level simulation waveform that contains test case #2

INPUT

Test case 2 consists of one operand positive and the other operand negative.



OUTPUT

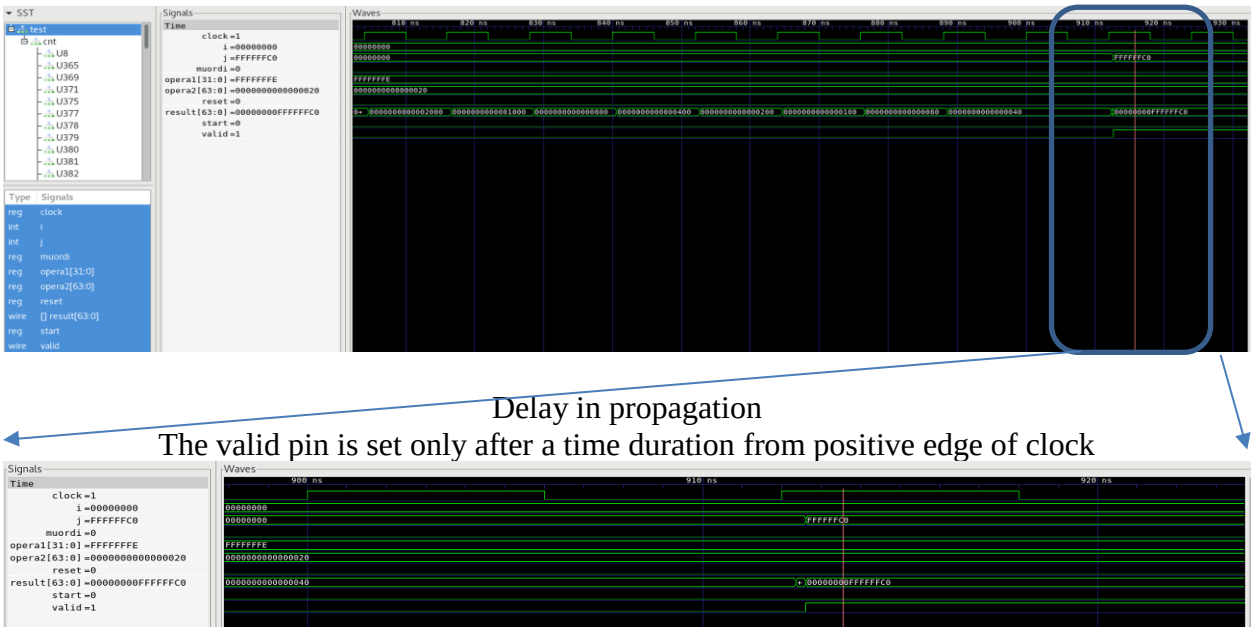
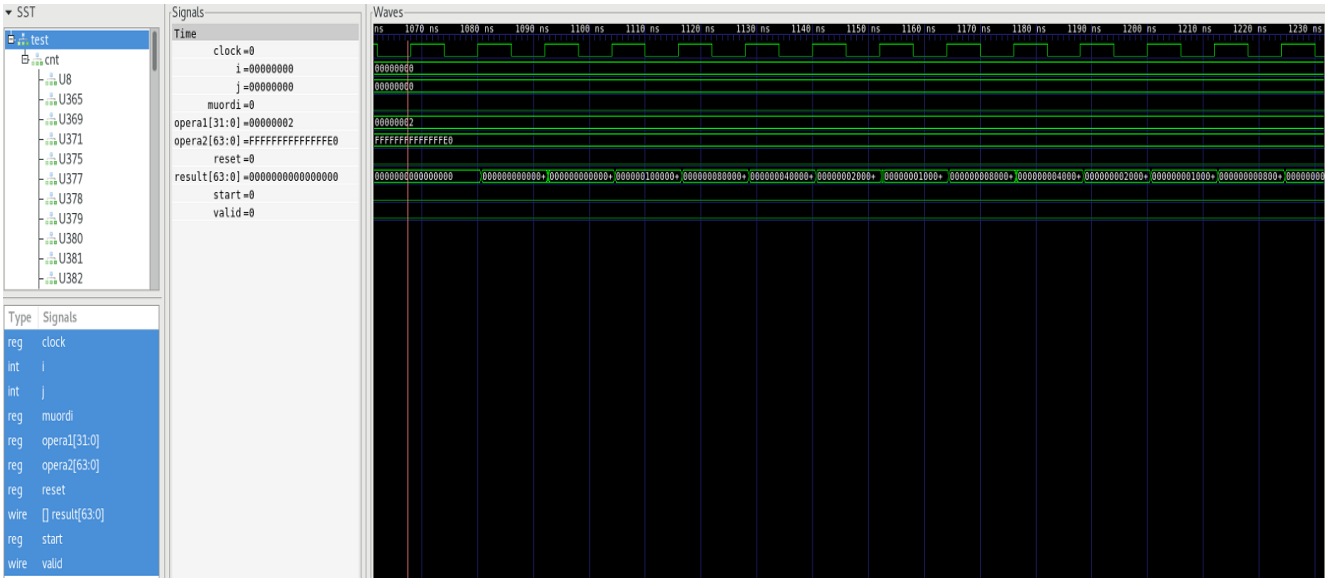


Figure V.1c: Gate-level simulation waveform that contains test case #3

INPUT



OUTPUT

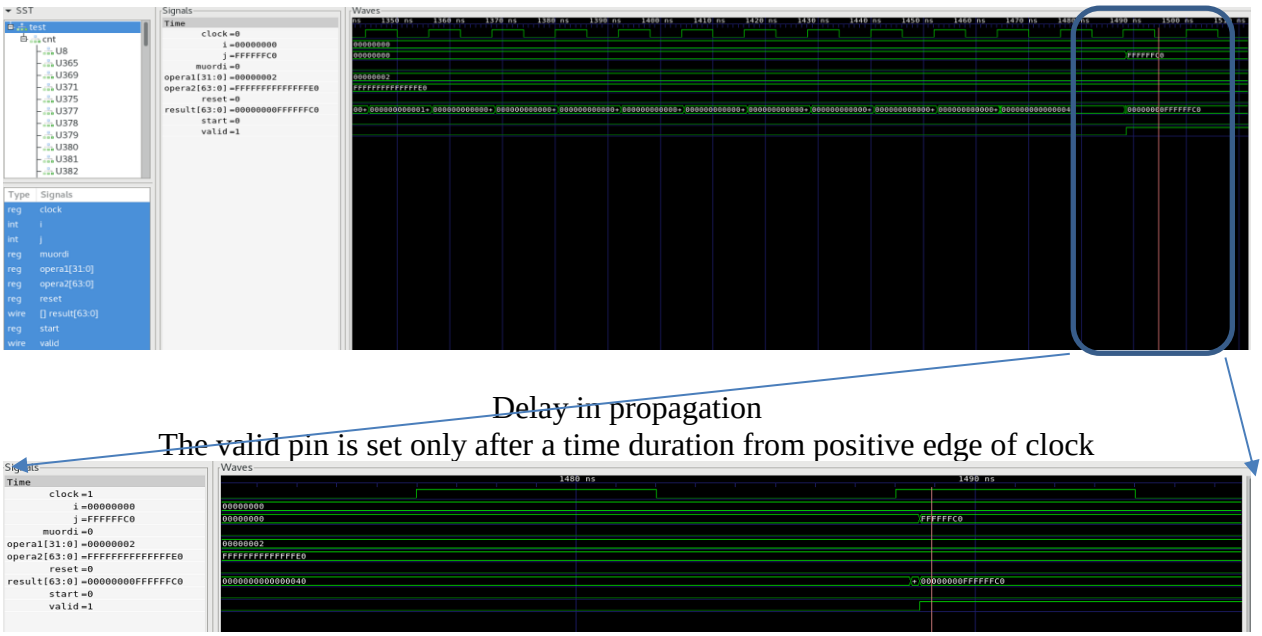
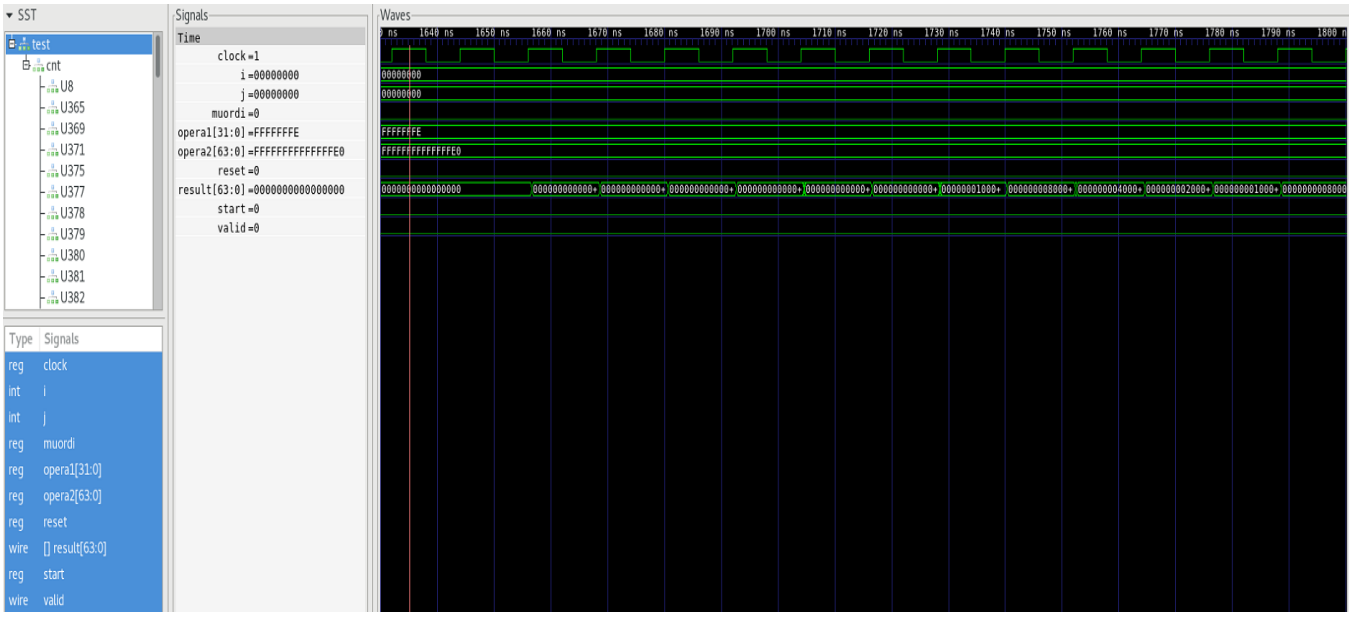
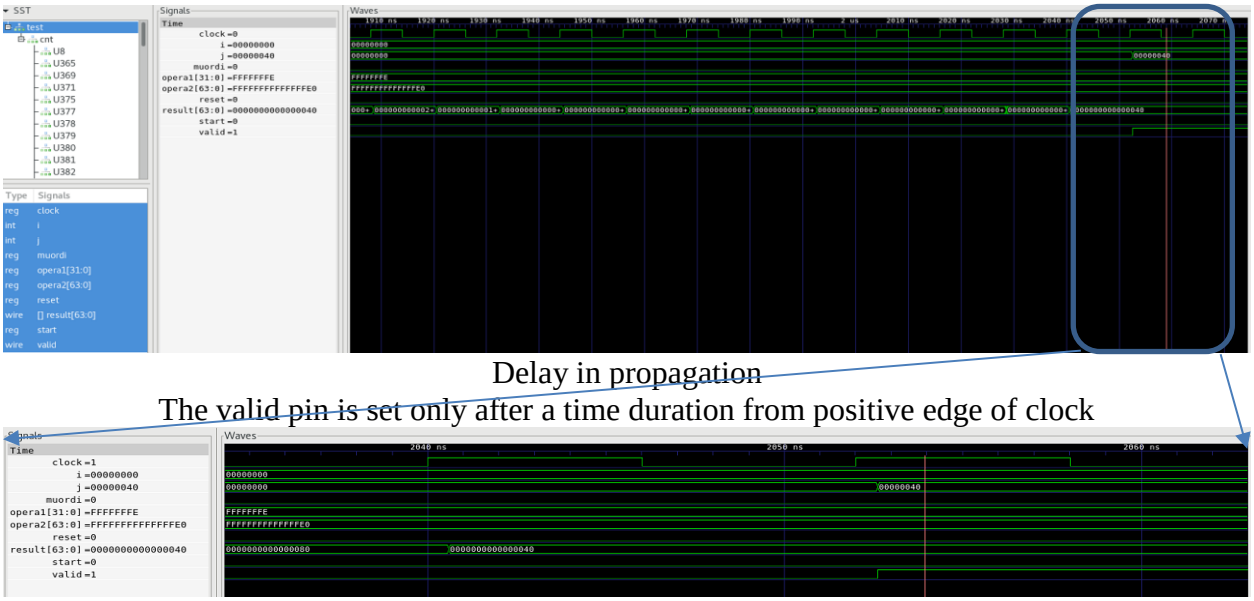


Figure V.1d: Gate-level simulation waveform that contains test case #4

INPUT



OUTPUT



Delay in propagation

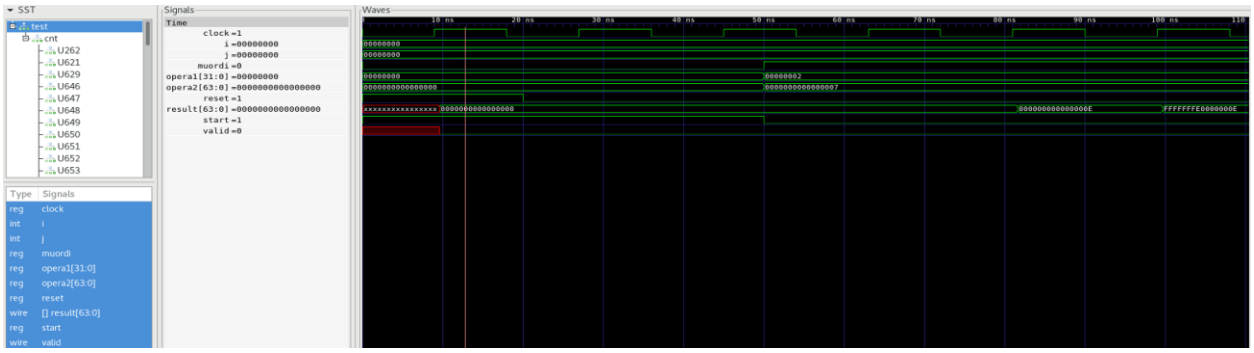
The valid pin is set only after a time duration from positive edge of clock

Table V.2 – Four Selected Test Data for Divider Circuit

Test Case	**opera1** (hex)	**opra2** (hex)	**Time Delay (in time unit)**
1	**00000002**	**000000000000007**	**66 clock+0.66ns** **66 * 18+0.66ns** **1188+0.66ns**
2	**00000002**	**FFFFFFFFFFFFFFF9**	**70 clock+0.66ns** **35 * 18+0.66ns** **1260+0.66ns**
3	**FFFFFFFE**	**0000000000000007**	**69 clock+0.66ns** **69 * 18+0.66ns** **1242+0.66ns**
4	**FFFFFFFE**	**FFFFFFFFFFFFFFF9**	**70 clock+0.66ns** **70 * 18+0.66ns** **1260+0.66ns**

Figure V.2a: Gate-level simulation waveform that contains test case #1

INPUT



OUTPUT

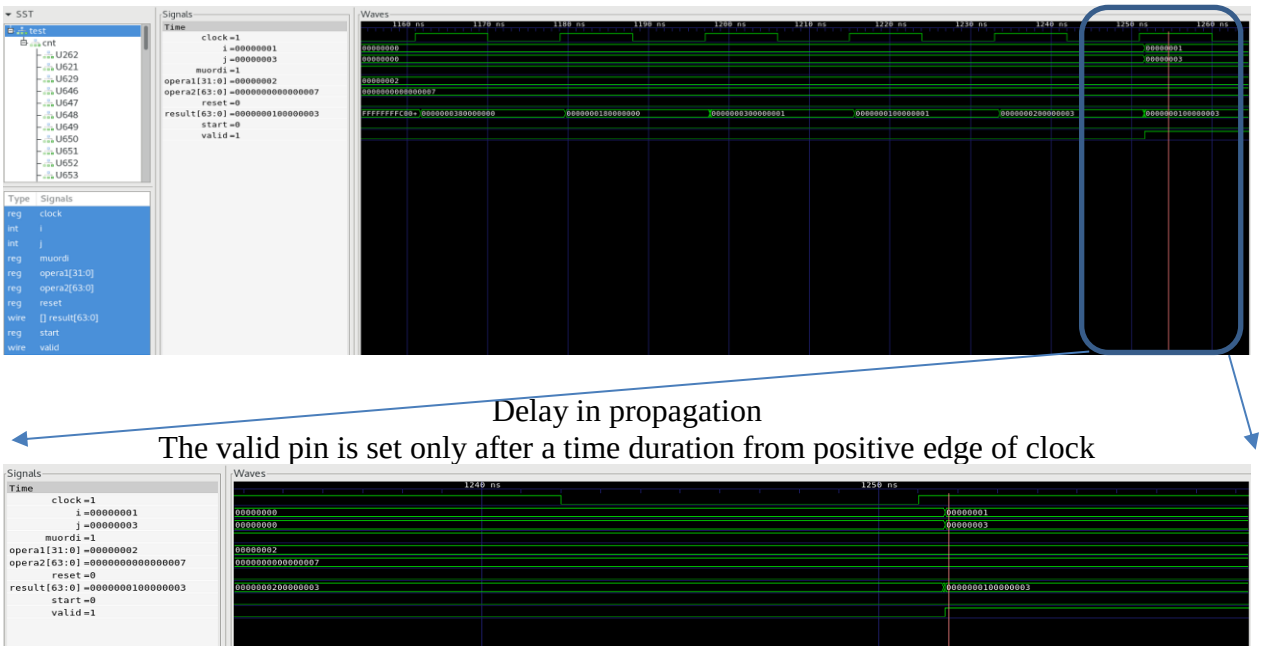
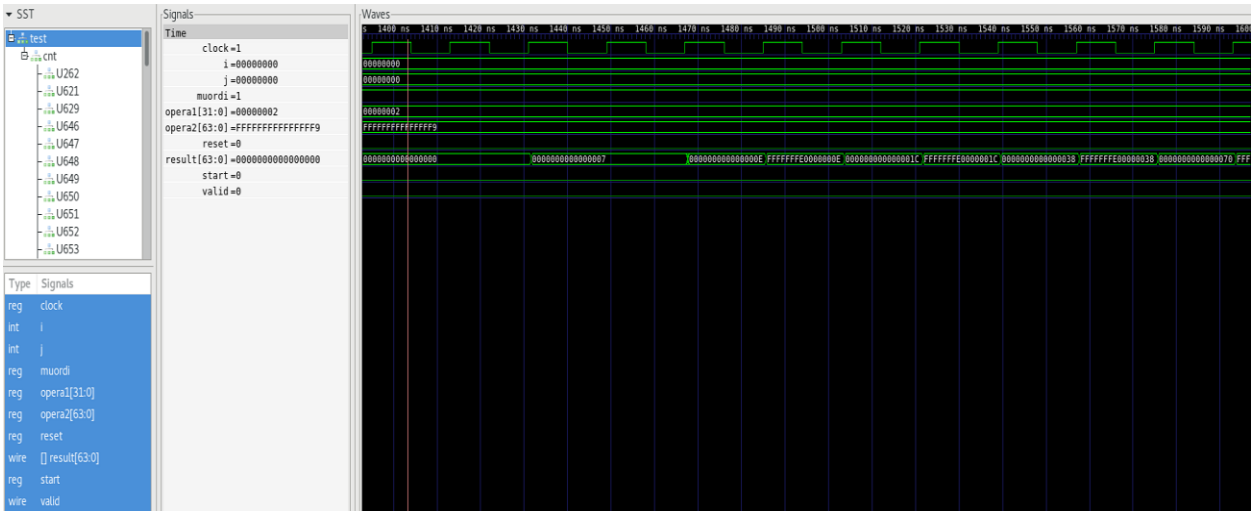


Figure V.2b: Gate-level simulation waveform that contains test case #2

INPUT



OUTPUT

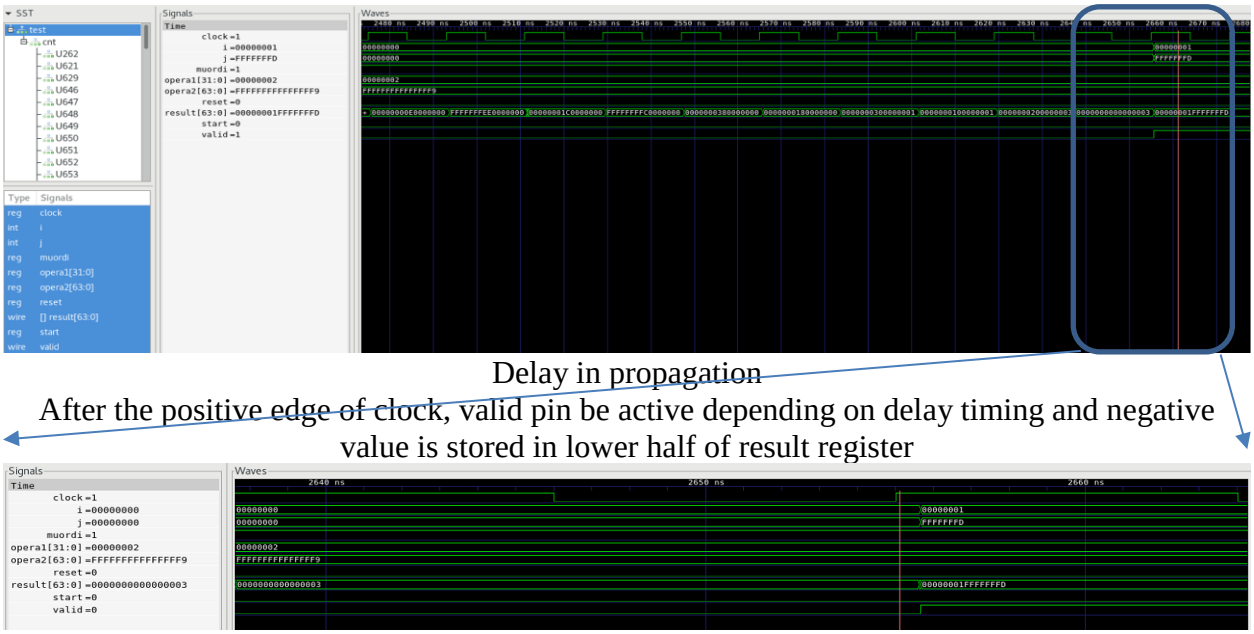
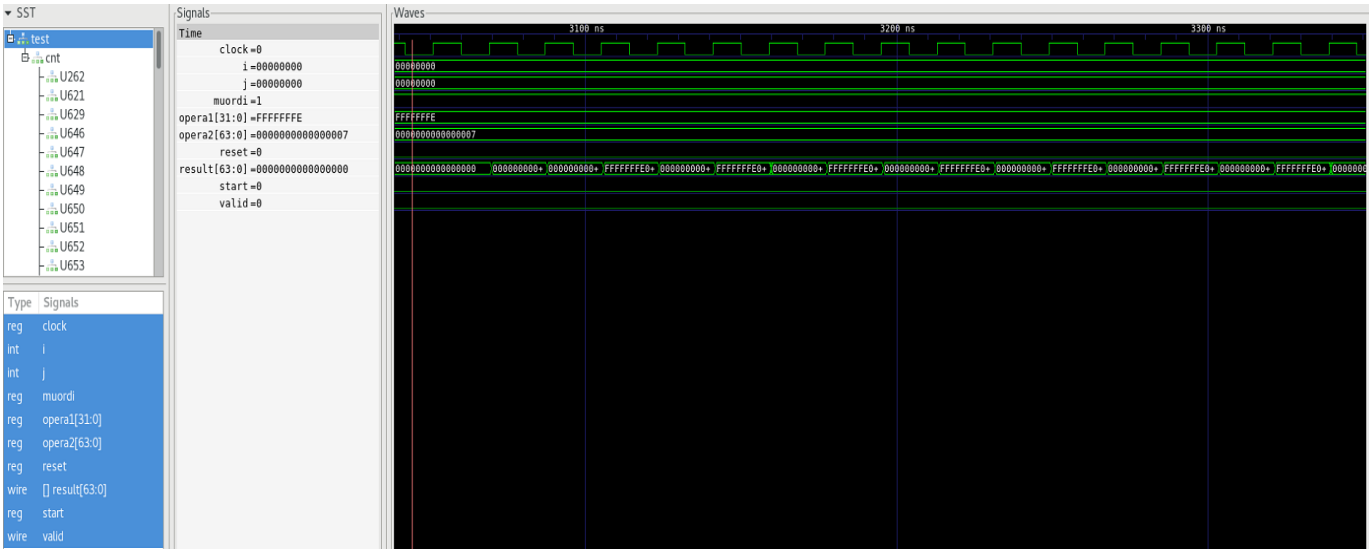


Figure V.2c: Gate-level simulation waveform that contains test case #3

INPUT



OUTPUT

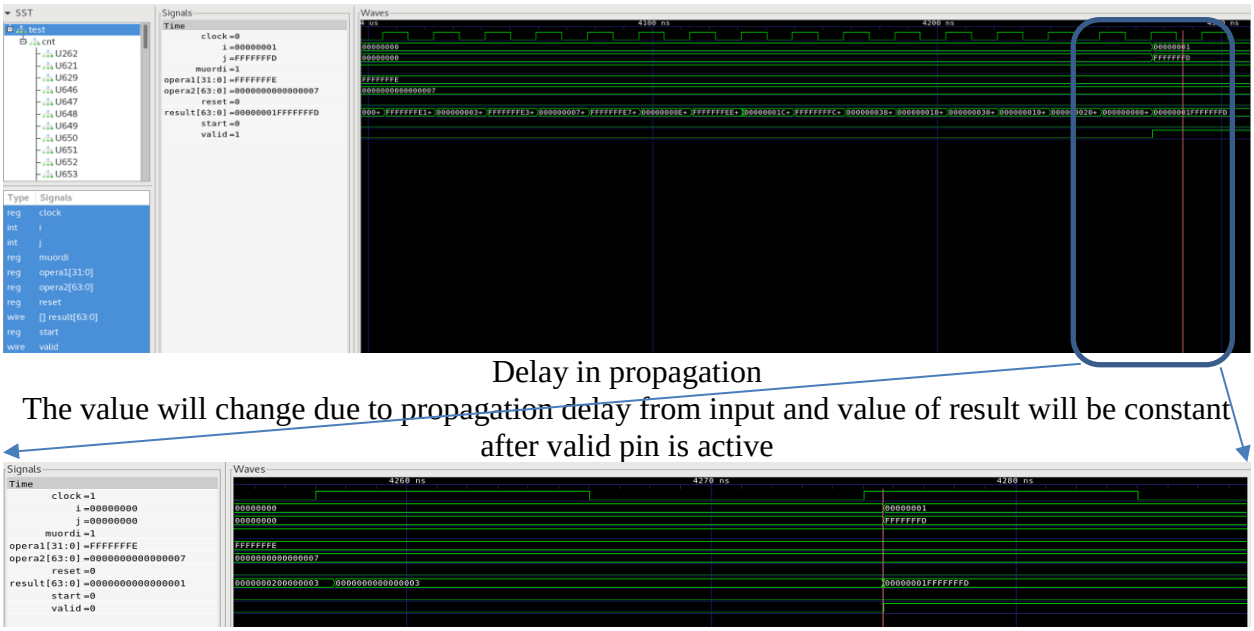
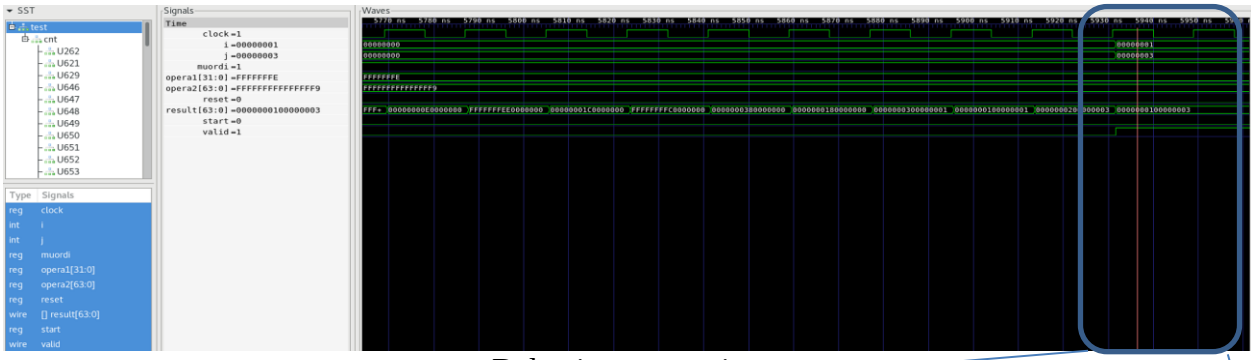
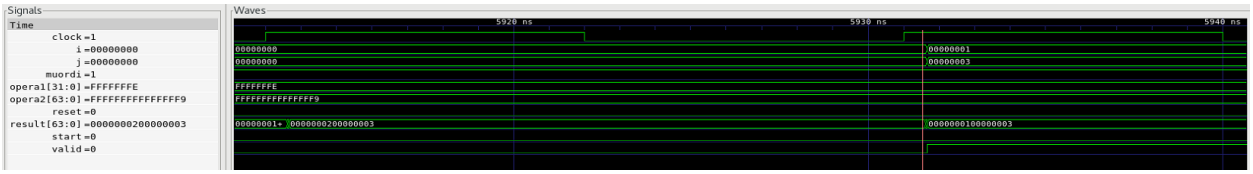


Figure V.2d: Gate-level simulation waveform that contains test case #4

INPUT

OUTPUT





Based on the gate-level simulation with different data, briefly discuss the timing characteristics at each operation step throughout the multiplication process and the division process related to the data (operands) (less than 150 words)

The two operands considered are first checked for any negative value. This negative value should be converted to positive value. The clock cycles would increase in test case 4 as both operands are negative. As in test case 3, if anyone operand is negative, then the flag is set and number of clock cycles required to convert to positive will be one less when compared to test case 4. Least number of clock cycles are consumed when both operands are positive as in case 1. The test case 3 and 2 will consume additional clock cycles to convert the output to negative. The synthesis clock time should be set appropriately to avoid any setup and hold violations.

VI. Conclusion

Give conclusion about the entire project (**less than 100 words**)

The project helped in understanding synthesis tools like design complier and analyzing the design. Multiple states were used for designing multiplier and divider modules. Synthesizing the result helped in understanding timing constraints and optimization. Latches from design were removed from design using the synthesis report. Individual modules can be used for marketing purpose and reusability is also increased. Area, delay and power optimization are performed in design. Different adder styles can be used for faster processing, lower area consumption like carry look ahead adder, ripple carry adder or parallel adders.

Appendix A

A.1 Contents from EDA Tool Configurations and Setup Files

```
set search_path {/apps/synopsys/SYNTH/libraries/syn \
/apps/synopsys/CORE/libraries/syn ./src}

set link_library {* class.db and_or.db dw_foundation.sldb}

set target_library {class.db and_or.db}

set symbol_library {class.sdb generic.sdb}
set synthetic_library {dw_foundation.sldb standard.sldb}
```

A.2 Commands and/or Scripts Used for Simulation and Synthesis

Compile Command:- `vcs +v2k <filename.v> <testbench.v>`

Simulation Command:- `./simv`

Synthesis Command:- `dc_shell -xg -f synthesis.script | tee <synres.txt>`

Postsynthesis Simulation Command:-

```
ncverilog -y /apps/toshiba/sjsu/verilog/tc240c +libext+.tsbvlip
+access +r testbench.v design_netlist.v
```

Synthesis script for divider module

```
set link_library {/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db_WCCOM25}
set target_library {/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db_WCCOM25}
set symbol_library {/apps/toshiba/sjsu/synopsys/tc240c/tc240c.workview.sdb}
set synthetic_library {dw_foundation.sldb standard.sldb}

set_min_library /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db_WCCOM25 -min_version
/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db_BCCOM25

read_verilog {divider.v}
current_design divider
link
check_design
create_clock clock -name clock -period 18
```

```
set_propagated_clock clock
set_clock_uncertainty .25 clock

set_max_delay 2 from [all_inputs]
set_max_delay 2 -to [all_outputs]
set_max_area 1000
set_fix_hold clock
compile -map_effort high
report_cell
report_net
update_timing
report_timing -max_paths 1
report_timing
report_area
report_power
write -hierarchy -format verilog -output seq_netlist.v
quit
```

Synthesis Script for multiplier module

```
set link_library {/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db_WCCOM25}
set target_library {/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db_WCCOM25}
set symbol_library {/apps/toshiba/sjsu/synopsys/tc240c/tc240c.workview.sdb}
set synthetic_library {dw_foundation.sldb standard.sldb}

set_min_library /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db_WCCOM25 -min_version
/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db_BCCOM25

read_verilog {multiplier.v}
current_design multipier
link
check_design
create_clock clock -name clock -period 12

set_propagated_clock clock
set_clock_uncertainty .25 clock

set_max_delay 2 from [all_inputs]
set_max_delay 2 -to [all_outputs]
set_max_area 0
set_fix_hold clock
compile -map_effort high
report_cell
report_net
update_timing
```

```
report_timing -max_paths 5
report_timing
report_area
report_power
write -hierarchy -format verilog -output seq_netlist.v
quit
```

Appendix B
Completed Verilog Source Codes and Testbenches
(Use Courier New 10 font)

Multiplication Module

```
`timescale 1ns/10ps
module multiplier(result,opera1,opera2,clock,valid,muordi,start,reset);
parameter s0=3'd0,s1=3'd1,s2=3'd2,s3=3'd3;
output [63:0]result;
input [63:0]opera2;
input [31:0]opera1;
input clock;
output valid;
input muordi;
input start,reset;
wire [31:0]res,opera1;
reg [31:0]tempB;
wire reset,start,cout;
reg flag,flag1;
reg [2:0] state;
reg [2:0] next_state;
reg OPE,neg,valid;
reg [31:0]count;
reg [63:0]result;
reg divres,divneg;
reg [31:0]tempdiv;
reg [1:0] multi,chk;
reg [2:0] st2;



add32 add1(.sum(res),.cout(cout),.a(tempB),.b(result[63:32]),.cin(neg));

always @ (posedge clock) begin
      chk={reset,start};
      casex(chk)
      2'b1x:
            state=s0;
      2'b01:
            state=s0;
      default:
            state=next_state;
      endcase
      case(state)
      s0: begin
            case (start)
            1'b1:begin
                  next_state=s0;
                  result=0;
                  valid=0;
                  flag=0;
                  OPE=0;
```

```
                neg=0;
                tempB=0;
                count=32'hFFFFFFFF;
                end
          1'b0:next_state=s1;
          endcase
          end

    s1: begin

          result[63:32]=32'b0;
          //                MULTIPLICATION

                multi={opera2[31],opera1[31]};
                case(multi)
                2'b10:begin
                      flag=1;
                      tempB=opera2[31:0];
                      result[31:0]=opera1;
                      neg=1;
                      next_state=s3;
                end
                2'b01:begin
                      tempB=opera1;
                      flag=1;
                      result[31:0]=opera2[31:0];
                      neg=1;
                      next_state=s3;
                end
                2'b11:begin
                      neg=1;
                      case (flag)
                      1'b0:begin
                            tempB=opera2[31:0];
                            flag=1;
                            next_state=s1;
                      end
                      1'b1:begin
                            tempB=opera1;
                            result[31:0]=res;
                            next_state=s3;
                            flag=0;
                      end
                      endcase
                      end
                default:begin
                result[31:0]=opera1;
                tempB=opera2[31:0];
                neg=0;
                next_state=s3;
                end
                endcase

    end
    s2:begin

          result[31:0]=res;
```

```
            valid=1;
         end
      s3:begin
            case(count[0])
            1'b1:begin
                  OPE=result[0];
                        case(OPE)
                        1'b1:begin
                        result[63:32]=res;
                        end
                        endcase
                  result=result>>1;
                  count=count>>1;

                  next_state=s3;
            end
            default:begin
                  chk={flag,count[0]};
                  case(chk)
                  2'b10:begin
                        tempB=result[31:0];
                        neg=1;
                        next_state=s2;
                  end
                  default:begin
                        valid=1;
                        next_state=s0;
                  end
                  endcase
            end
            endcase
      end
      endcase

      end

endmodule
```

MULTIPLIER TESTBENCH

```
`timescale 1ns/10ps

module test();
wire [63:0]result;
reg [31:0]opera1;
reg [63:0]opera2;
reg clock;
wire valid;
reg muordi;
reg reset;
reg start;
multiplier
cnt(.result(result),.opera1(opera1),.opera2(opera2),.clock(clock),.valid(vali
d),.muordi(muordi),.start(start),.reset(reset));
integer i,j;
initial begin

reset=1;start=1;muordi=0;opera1=0;opera2=0;i=0;j=0;#9
```

```
reset=0;start=1;#15
start=0;
opera1=2;opera2=32;muordi=0; #450;
start=1;#15
start=0;
opera1=-2;opera2=32;#550;
start=1;#20
start=0;
opera1=2;opera2=-32;#550;
start=1;#20
start=0;
opera1=-2;opera2=-32;#500;
#5 $finish;
end


always @(valid) begin
      i=result[63:32];
      j=result[31:0];
      if(valid==1)
      $display("Multiplicand=",opera1,"| Multiplier=",opera2,"|| Result--
",j);

end

initial begin
clock=1;
forever #6 clock=~clock;
end

initial begin

$dumpfile("multiplier.vcd");
$dumpvars(0,test);

end
endmodule
```

DIVIDER MODULE

```
`timescale 1ns/10ps
module divider(result,opera1,opera2,clock,valid,muordi,start,reset);
parameter s0=3'd0,s1=3'd1,s2=3'd2,s3=3'd3,s4=3'd4,s5=3'd5;
output [63:0]result;
input [63:0]opera2;
input [31:0]opera1;
input clock;
output valid;
input muordi;
input start,reset;
wire [31:0]res,opera1;
reg [31:0]tempB;
wire reset,start,cout;
reg sign,flag,flag1;
reg [2:0] state;
reg [2:0] next_state;
reg OPE,neg,valid;
```

```
reg [31:0]count;
reg [63:0]result;
reg divres,divneg;
reg [31:0]tempdiv;
reg [1:0] multi,chk;
reg [2:0] st2;


add32 add1(.sum(res),.cout(cout),.a(tempB),.b(result[63:32]),.cin(neg));

always @ (posedge clock) begin
      chk={reset,start};
      casex(chk)
      2'b1x:
            state=s0;
      2'b01:
            state=s0;
      default:
            state=next_state;
      endcase
      case(state)
      s0: begin
            case (start)
            1'b1:begin
                  neg=0;
                  next_state=s0;
                  result=0;
                  valid=0;
                  flag=0;
                  divneg=0;
                  divres=0;
                  sign=0;
                  flag1=0;
                  OPE=0;
                  tempB=0;
                  tempdiv=0;
                  count=32'hFFFFFFFF;
                  end
            default:next_state=s1;
            endcase
            end

      s1: begin

            result[63:32]=32'b0;

            //                DIVISION

            //1'b1:begin
                  multi={opera2[63],opera1[31]};
                  case(multi)
                  2'b10:begin
                        divneg=1;
                        chk={flag,flag1};
                        case(chk)
                        2'b11:begin
                              result[63:32]=res;
```

```
                tempB=opera1;
                next_state=s2;
                flag=0;
                neg=1;
          end
          2'b01:begin
                neg=0;
                result[31:0]=res;
                tempB=~opera2[63:32];
                result[32]=cout;
                flag=1;
                next_state=s1;
          end
          default:begin
                tempB=opera2[31:0];
                neg=1;
                flag1=1;
                next_state=s1;
          end
          endcase
    end
    2'b01:begin
          case(flag)
          1'b1:begin
                result[31:0]=opera2[31:0];
                divneg=1;
                tempB=res;
                next_state=s2;
                flag=0;
          end
          default:begin

                tempB=opera1;
                flag=1;
                neg=1;
                next_state=s1;
          end
          endcase
    end
    2'b11:begin
          case(sign)
          1'b1:begin
                chk={flag,flag1};
                case(chk)
                2'b11:begin
                      result[63:32]=res;
                      tempB=tempdiv;
                      next_state=s2;
                      flag=0;

                      neg=1;
                end
                2'b01:begin

                      neg=0;
                      result[31:0]=res;
                      tempB=~opera2[63:32];
```

```
                              result[32]=cout;
                              flag=1;
                              next_state=s1;
                        end
                        default:begin
                              tempdiv=res;
                              tempB=opera2[31:0];
                              neg=1;
                              flag1=1;
                              next_state=s1;
                        end
                        endcase
                  end
                  default:begin
                        tempB=opera1;
                        sign=1;
                        next_state=s1;
                        neg=1;
                  end
                  endcase
            end
            default:begin
                  result=opera2;
                  tempB=opera1;
                  OPE=1;
                  neg=1;
                  result=result<<1;
                  next_state=s3;
            end
            endcase
      end
s2:begin

      st2={muordi,divres,count[0]};
      case(st2)
      3'b110:begin
            result[31:0]=res;
            result[63:32]=tempdiv;
            valid=1;
            next_state=s0;
            end
      default:begin
            result=result<<1;
            next_state=s3;
            end
      endcase
   end

s3:begin
      case(count[0])
      1'b1:result[63:32]=res;
      endcase

      case(result[63])
      1'b0:begin
            next_state=s4;
      end
```

```
            default:begin
                  neg=0;
                  next_state=s5;
            end
            endcase
            case(count[0])
            1'b0:begin
                  case(divneg)
                  1'b1:begin
                        divres=1;
                        neg=1;
                        tempB=result[31:0];
                        tempdiv=result[63:32]>>1;
                        result[63:32]=0;
                        next_state=s2;
                  end
                  default:begin
                        valid=1;
                        next_state=s0;
                        result[63:32]=result[63:32]>>1;
                  end
                  endcase
            end
            endcase
           end
      s4:begin
            result=result<<1;
            result[0]=1;
            OPE=1;
            neg=1;
            next_state=s3;
            count=count>>1;
      end
      s5:begin
            result[63:32]=res;
            result=result<<1;
            result[0]=0;
            OPE=1;
            neg=1;
            next_state=s3;
            count=count>>1;
      end

      endcase
      end

endmodule
```

DIVIDER TESTBENCH

```
`timescale 1ns/10ps

module test();
wire [63:0]result;
reg [31:0]opera1;
reg [63:0]opera2;
reg clock;
wire valid;
```

```
reg muordi;
reg reset;
reg start;
divider
cnt(.result(result),.opera1(opera1),.opera2(opera2),.clock(clock),.valid(vali
d),.muordi(muordi),.start(start),.reset(reset));
integer i,j;
initial begin

reset=1;start=1;muordi=0;opera1=0;opera2=0;i=0;j=0;#20
reset=0;start=1;#30
start=0;
opera1=2;opera2=7;muordi=1; #1300;
reset=0;start=1;#35
start=0;
opera1=2;opera2=-7;muordi=1; #1600;
reset=0;start=1;#35
start=0;
opera1=-2;opera2=7;muordi=1; #1600;
reset=0;start=1;#35
start=0;
opera1=-2;opera2=-7;muordi=1; #1600;
#5 $finish;
end


always @(valid) begin
      i=result[63:32];
      j=result[31:0];
      if(valid==1)
      $display("Divisor=",opera1,"| Dividend=",opera2,"|| Remainder---",i,"
|| Quotient--",j);

end

initial begin
clock=0;
forever #9 clock=~clock;
end

initial begin

$dumpfile("divider.vcd");
$dumpvars(0,test);

end
endmodule
```

ADDER MODULE

```
//*****************************ADD32 MODULE*********************************

module add32(sum,cout,a,b,cin);
output [31:0]sum;
output cout;
input [31:0]a;
input [31:0]b;
input cin;
```

```
wire [32:0]cout1;


reg [31:0]temp;


assign cout1[0]=cin;
assign cout=cout1[32];


always @ (cin or a) begin
if (cin==1)
       temp=~a;
else
       temp=a;
end

genvar i;
generate


            for(i=0;i<=31;i=i+1) begin : generate_block
                  fulladd
f1(.sum(sum[i]),.cin(cout1[i]),.a(temp[i]),.b(b[i]),.cout(cout1[i+1]));


            end
endgenerate
endmodule
```

FULL ADDER MODULE

```
//*********************************FULL ADDER*******************************

module fulladd(sum,cout,a,b,cin);
output sum,cout;
input a,b,cin;
wire a,b,sum1,carry2,carry1,cin;

adder a1(.sum(sum1),.carry(carry1),.a(a),.b(b));
adder a2(.sum(sum),.carry(carry2),.a(cin),.b(sum1));

or o1(cout,carry1,carry2);

endmodule
```

HALF ADDER MODULE

```
//*******************************HALF ADDER*****************************


module adder(sum,carry,a,b);
output sum,carry;
input a,b;

assign sum=a^b;
assign carry=a&b;
endmodule
```

Appendix C
Reports and Circuits from EDA Tools
(Use Courier New 10 font)

C.1 Contents of Selected Reports from RTL (Pre-synthesis) Simulations (VCS or NCVERILOG)

PRESYNTHESIS SIMULATION REPORT FOR MULTIPLIER

```
Chronologic VCS simulator copyright 1991-2014
Contains Synopsys proprietary information.
Compiler version I-2014.03-2; Runtime version I-2014.03-2;  Dec  9 22:22 2016
Multiplicand=         2| Multiplier=                  32|| Result--
64
Multiplicand=4294967294| Multiplier=                  32|| Result--        -
64
Multiplicand=         2| Multiplier=18446744073709551584|| Result--        -
64
Multiplicand=4294967294| Multiplier=18446744073709551584|| Result--
64
$finish called from file "test.v", line 29.
$finish at simulation time              213400
           V C S   S i m u l a t i o n   R e p o r t
Time: 2134000 ps
CPU Time:      0.210 seconds;       Data structure size:   0.0Mb
Fri Dec  9 22:22:44 2016
```

PRESYNTHESIS SIMULATION REPORT FOR DIVIDER

```
Chronologic VCS simulator copyright 1991-2014
Contains Synopsys proprietary information.
Compiler version I-2014.03-2; Runtime version I-2014.03-2;  Dec  9 22:24 2016
Divisor=         2| Dividend=                   7|| Remainder---          1
|| Quotient--          3
Divisor=         2| Dividend=18446744073709551609|| Remainder---          1
|| Quotient--         -3
Divisor=4294967294| Dividend=                   7|| Remainder---          1
|| Quotient--         -3
Divisor=4294967294| Dividend=18446744073709551609|| Remainder---          1
|| Quotient--          3
$finish called from file "test.v", line 29.
$finish at simulation time              626000
           V C S   S i m u l a t i o n   R e p o r t
Time: 6260000 ps
CPU Time:      0.210 seconds;       Data structure size:   0.0Mb
Fri Dec  9 22:24:19 2016
```

C.2 Contents of Selected Reports from Netlist (Post-synthesis) Simulations (VCS or NCVERILOG)

POSTSYNTHESIS SIMULATION REPORT FOR MULTIPLIER

```
ncverilog: 14.10-p001: (c) Copyright 1995-2014 Cadence Design Systems, Inc.
Loading snapshot worklib.test:v ..................... Done
ncsim> source /apps/cadence/INCISIV141/tools/inca/files/ncsimrc
ncsim> run
Multiplicand=         2| Multiplier=                  32|| Result--
64
Multiplicand=4294967294| Multiplier=                  32|| Result--        -
64
Multiplicand=         2| Multiplier=18446744073709551584|| Result--        -
64
Multiplicand=4294967294| Multiplier=18446744073709551584|| Result--
64
Simulation complete via $finish(1) at time 2134 NS + 0
./test.v:29 #5 $finish;
ncsim> exit
```

POSTSYNTHESIS SIMULATION REPORT FOR DIVIDER

```
ncverilog: 14.10-p001: (c) Copyright 1995-2014 Cadence Design Systems, Inc.
Loading snapshot worklib.test:v ..................... Done
ncsim> source /apps/cadence/INCISIV141/tools/inca/files/ncsimrc
ncsim> run
Divisor=         2| Dividend=                    7|| Remainder---          1
|| Quotient--           3
Divisor=         2| Dividend=18446744073709551609|| Remainder---          1
|| Quotient--          -3
Divisor=4294967294| Dividend=                    7|| Remainder---          1
|| Quotient--          -3
Divisor=4294967294| Dividend=18446744073709551609|| Remainder---          1
|| Quotient--           3
Simulation complete via $finish(1) at time 6260 NS + 0
./test.v:29 #5 $finish;
ncsim> exit
```

C.3 Contents of Selected Reports from Synthesis (Design Compiler)

SYNTHESIS REPORT FOR MULTIPLIER

```
ncverilog: 14.10-p001: (c) Copyright 1995-2014 Cadence Design Systems, Inc.
Recompiling... reason: file './seq_netlist.v' is newer than expected.
      expected: Fri Dec  9 16:46:05 2016
      actual:   Fri Dec  9 22:52:47 2016
            Caching library 'tc240c' ....... Done
            Caching library 'worklib' ....... Done
      Elaborating the design hierarchy:
  add32 add1 ( .sum(res), .a(tempB), .b({1'b0, result[62:33],
net4586}), .cin(
           |
ncelab: *W,CUVWSP (./seq_netlist.v,1297|11): 1 output port was not connected:
ncelab: (./seq_netlist.v,1102): cout

      Building instance overlay tables: ..................... Done
      Building instance specific data structures.
      Loading native compiled code:     ..................... Done
      Design hierarchy summary:
                            Instances  Unique
            Modules:             1359     155
            UDPs:                 133       3
            Primitives:          4511       9
            Timing outputs:      1131      29
            Registers:            140      17
            Scalar wires:        1287       -
            Expanded wires:        96       2
            Always blocks:          1       1
            Initial blocks:         3       3
            Cont. assignments:      0      12
            Pseudo assignments:     2       2
            Timing checks:        804     273
            Simulation timescale:  10ps
```

MAXIMUM PATH SHOWING SLACK MET FOR MULTIPLIER

```
report_timing

****************************************
Report : timing
        -path full
        -delay max
        -max_paths 1
Design : multiplier
Version: C-2009.06-SP5
Date   : Fri Dec  9 22:52:46 2016
****************************************

Operating Conditions: WCCOM25   Library: tc240c
Wire Load Model Mode: top

  Startpoint: neg_reg (rising edge-triggered flip-flop clocked by clock)
```

```
Endpoint: result_reg[31]
          (rising edge-triggered flip-flop clocked by clock)
Path Group: clock
Path Type: max

Point                                                    Incr       Path
-----------------------------------------------------------------------------
clock clock (rise edge)                                  0.00       0.00
clock network delay (propagated)                         0.00       0.00
neg_reg/CP (CFD1QX4)                                     0.00       0.00 r
neg_reg/Q (CFD1QX4)                                      0.43       0.43 f
add1/cin (add32)                                         0.00       0.43 f
add1/U3/Z (CIVX2)                                        0.06       0.49 r
add1/U9/Z (CND2X2)                                       0.09       0.59 f
add1/U6/Z (CND2X2)                                       0.10       0.68 r
add1/generate_block[0].f1/a (fulladd_0)                  0.00       0.68 r
add1/generate_block[0].f1/a1/a (adder_0)                 0.00       0.68 r
add1/generate_block[0].f1/a1/U4/Z (CIVX2)                0.07       0.76 f
add1/generate_block[0].f1/a1/U3/Z (CND2X2)               0.07       0.82 r
add1/generate_block[0].f1/a1/U2/Z (CND2X2)               0.08       0.91 f
add1/generate_block[0].f1/a1/sum (adder_0)               0.00       0.91 f
add1/generate_block[0].f1/a2/b (adder_63)                0.00       0.91 f
add1/generate_block[0].f1/a2/U2/Z (CAN2X1)               0.15       1.06 f
add1/generate_block[0].f1/a2/carry (adder_63)            0.00       1.06 f
add1/generate_block[0].f1/U1/Z (COR2X1)                  0.23       1.29 f
add1/generate_block[0].f1/cout (fulladd_0)               0.00       1.29 f
add1/generate_block[1].f1/cin (fulladd_31)               0.00       1.29 f
add1/generate_block[1].f1/a2/a (adder_61)                0.00       1.29 f
add1/generate_block[1].f1/a2/U2/Z (CAN2X1)               0.17       1.47 f
add1/generate_block[1].f1/a2/carry (adder_61)            0.00       1.47 f
add1/generate_block[1].f1/U2/Z (CIVX2)                   0.06       1.53 r
add1/generate_block[1].f1/U1/Z (CND2IX2)                 0.08       1.61 f
add1/generate_block[1].f1/cout (fulladd_31)              0.00       1.61 f
add1/generate_block[2].f1/cin (fulladd_30)               0.00       1.61 f
add1/generate_block[2].f1/a2/a (adder_59)                0.00       1.61 f
add1/generate_block[2].f1/a2/U2/Z (CAN2X1)               0.16       1.76 f
add1/generate_block[2].f1/a2/carry (adder_59)            0.00       1.76 f
add1/generate_block[2].f1/U2/Z (CND2IX1)                 0.17       1.94 f
add1/generate_block[2].f1/cout (fulladd_30)              0.00       1.94 f
add1/generate_block[3].f1/cin (fulladd_29)               0.00       1.94 f
add1/generate_block[3].f1/a2/a (adder_57)                0.00       1.94 f
add1/generate_block[3].f1/a2/U2/Z (CAN2X1)               0.16       2.09 f
add1/generate_block[3].f1/a2/carry (adder_57)            0.00       2.09 f
add1/generate_block[3].f1/U1/Z (COR2X1)                  0.23       2.32 f
add1/generate_block[3].f1/cout (fulladd_29)              0.00       2.32 f
add1/generate_block[4].f1/cin (fulladd_28)               0.00       2.32 f
add1/generate_block[4].f1/a2/a (adder_55)                0.00       2.32 f
add1/generate_block[4].f1/a2/U2/Z (CAN2X1)               0.16       2.48 f
add1/generate_block[4].f1/a2/carry (adder_55)            0.00       2.48 f
add1/generate_block[4].f1/U2/Z (CND2IX1)                 0.17       2.66 f
add1/generate_block[4].f1/cout (fulladd_28)              0.00       2.66 f
add1/generate_block[5].f1/cin (fulladd_27)               0.00       2.66 f
add1/generate_block[5].f1/a2/a (adder_53)                0.00       2.66 f
add1/generate_block[5].f1/a2/U2/Z (CAN2X1)               0.16       2.81 f
add1/generate_block[5].f1/a2/carry (adder_53)            0.00       2.81 f
add1/generate_block[5].f1/U1/Z (COR2X1)                  0.23       3.04 f
add1/generate_block[5].f1/cout (fulladd_27)              0.00       3.04 f
```

```
add1/generate_block[6].f1/cin (fulladd_26)              0.00       3.04 f
add1/generate_block[6].f1/a2/a (adder_51)               0.00       3.04 f
add1/generate_block[6].f1/a2/U2/Z (CAN2X1)              0.16       3.20 f
add1/generate_block[6].f1/a2/carry (adder_51)           0.00       3.20 f
add1/generate_block[6].f1/U1/Z (COR2X1)                 0.23       3.43 f
add1/generate_block[6].f1/cout (fulladd_26)             0.00       3.43 f
add1/generate_block[7].f1/cin (fulladd_25)              0.00       3.43 f
add1/generate_block[7].f1/a2/a (adder_49)               0.00       3.43 f
add1/generate_block[7].f1/a2/U2/Z (CAN2X1)              0.16       3.59 f
add1/generate_block[7].f1/a2/carry (adder_49)           0.00       3.59 f
add1/generate_block[7].f1/U1/Z (COR2X1)                 0.23       3.82 f
add1/generate_block[7].f1/cout (fulladd_25)             0.00       3.82 f
add1/generate_block[8].f1/cin (fulladd_24)              0.00       3.82 f
add1/generate_block[8].f1/a2/a (adder_47)               0.00       3.82 f
add1/generate_block[8].f1/a2/U2/Z (CAN2X1)              0.17       4.00 f
add1/generate_block[8].f1/a2/carry (adder_47)           0.00       4.00 f
add1/generate_block[8].f1/U2/Z (CIVX2)                  0.06       4.06 r
add1/generate_block[8].f1/U1/Z (CND2IX2)                0.08       4.14 f
add1/generate_block[8].f1/cout (fulladd_24)             0.00       4.14 f
add1/generate_block[9].f1/cin (fulladd_23)              0.00       4.14 f
add1/generate_block[9].f1/a2/a (adder_45)               0.00       4.14 f
add1/generate_block[9].f1/a2/U2/Z (CAN2X1)              0.17       4.31 f
add1/generate_block[9].f1/a2/carry (adder_45)           0.00       4.31 f
add1/generate_block[9].f1/U2/Z (CIVX2)                  0.06       4.37 r
add1/generate_block[9].f1/U1/Z (CND2X2)                 0.08       4.46 f
add1/generate_block[9].f1/cout (fulladd_23)             0.00       4.46 f
add1/generate_block[10].f1/cin (fulladd_22)             0.00       4.46 f
add1/generate_block[10].f1/a2/a (adder_43)              0.00       4.46 f
add1/generate_block[10].f1/a2/U1/Z (CAN2X1)             0.17       4.63 f
add1/generate_block[10].f1/a2/carry (adder_43)          0.00       4.63 f
add1/generate_block[10].f1/U2/Z (CIVX2)                 0.06       4.69 r
add1/generate_block[10].f1/U1/Z (CND2X2)                0.08       4.77 f
add1/generate_block[10].f1/cout (fulladd_22)            0.00       4.77 f
add1/generate_block[11].f1/cin (fulladd_21)             0.00       4.77 f
add1/generate_block[11].f1/a2/a (adder_41)              0.00       4.77 f
add1/generate_block[11].f1/a2/U2/Z (CAN2X1)             0.17       4.94 f
add1/generate_block[11].f1/a2/carry (adder_41)          0.00       4.94 f
add1/generate_block[11].f1/U2/Z (CIVX2)                 0.06       5.01 r
add1/generate_block[11].f1/U1/Z (CND2X2)                0.08       5.09 f
add1/generate_block[11].f1/cout (fulladd_21)            0.00       5.09 f
add1/generate_block[12].f1/cin (fulladd_20)             0.00       5.09 f
add1/generate_block[12].f1/a2/a (adder_39)              0.00       5.09 f
add1/generate_block[12].f1/a2/U2/Z (CAN2X1)             0.17       5.26 f
add1/generate_block[12].f1/a2/carry (adder_39)          0.00       5.26 f
add1/generate_block[12].f1/U2/Z (CIVX2)                 0.06       5.32 r
add1/generate_block[12].f1/U1/Z (CND2X2)                0.08       5.41 f
add1/generate_block[12].f1/cout (fulladd_20)            0.00       5.41 f
add1/generate_block[13].f1/cin (fulladd_19)             0.00       5.41 f
add1/generate_block[13].f1/a2/a (adder_37)              0.00       5.41 f
add1/generate_block[13].f1/a2/U1/Z (CAN2X1)             0.17       5.58 f
add1/generate_block[13].f1/a2/carry (adder_37)          0.00       5.58 f
add1/generate_block[13].f1/U2/Z (CIVX2)                 0.06       5.64 r
add1/generate_block[13].f1/U1/Z (CND2X2)                0.08       5.72 f
add1/generate_block[13].f1/cout (fulladd_19)            0.00       5.72 f
add1/generate_block[14].f1/cin (fulladd_18)             0.00       5.72 f
add1/generate_block[14].f1/a2/a (adder_35)              0.00       5.72 f
add1/generate_block[14].f1/a2/U2/Z (CAN2X1)             0.17       5.89 f
```

```
add1/generate_block[14].f1/a2/carry (adder_35)          0.00       5.89 f
add1/generate_block[14].f1/U2/Z (CIVX2)                 0.06       5.96 r
add1/generate_block[14].f1/U1/Z (CND2X2)                0.08       6.04 f
add1/generate_block[14].f1/cout (fulladd_18)            0.00       6.04 f
add1/generate_block[15].f1/cin (fulladd_17)             0.00       6.04 f
add1/generate_block[15].f1/a2/a (adder_33)              0.00       6.04 f
add1/generate_block[15].f1/a2/U2/Z (CAN2X1)             0.17       6.21 f
add1/generate_block[15].f1/a2/carry (adder_33)          0.00       6.21 f
add1/generate_block[15].f1/U2/Z (CIVX2)                 0.06       6.27 r
add1/generate_block[15].f1/U1/Z (CND2X2)                0.08       6.35 f
add1/generate_block[15].f1/cout (fulladd_17)            0.00       6.35 f
add1/generate_block[16].f1/cin (fulladd_16)             0.00       6.35 f
add1/generate_block[16].f1/a2/a (adder_31)              0.00       6.35 f
add1/generate_block[16].f1/a2/U2/Z (CAN2X1)             0.17       6.52 f
add1/generate_block[16].f1/a2/carry (adder_31)          0.00       6.52 f
add1/generate_block[16].f1/U2/Z (CIVX2)                 0.06       6.59 r
add1/generate_block[16].f1/U1/Z (CND2X2)                0.08       6.66 f
add1/generate_block[16].f1/cout (fulladd_16)            0.00       6.66 f
add1/generate_block[17].f1/cin (fulladd_15)             0.00       6.66 f
add1/generate_block[17].f1/a2/a (adder_29)              0.00       6.66 f
add1/generate_block[17].f1/a2/U2/Z (CAN2X1)             0.17       6.84 f
add1/generate_block[17].f1/a2/carry (adder_29)          0.00       6.84 f
add1/generate_block[17].f1/U2/Z (CIVX2)                 0.06       6.90 r
add1/generate_block[17].f1/U1/Z (CND2X2)                0.08       6.98 f
add1/generate_block[17].f1/cout (fulladd_15)            0.00       6.98 f
add1/generate_block[18].f1/cin (fulladd_14)             0.00       6.98 f
add1/generate_block[18].f1/a2/a (adder_27)              0.00       6.98 f
add1/generate_block[18].f1/a2/U3/Z (CAN2X1)             0.17       7.15 f
add1/generate_block[18].f1/a2/carry (adder_27)          0.00       7.15 f
add1/generate_block[18].f1/U2/Z (CIVX2)                 0.06       7.21 r
add1/generate_block[18].f1/U1/Z (CND2X2)                0.09       7.30 f
add1/generate_block[18].f1/cout (fulladd_14)            0.00       7.30 f
add1/generate_block[19].f1/cin (fulladd_13)             0.00       7.30 f
add1/generate_block[19].f1/a2/a (adder_25)              0.00       7.30 f
add1/generate_block[19].f1/a2/U3/Z (CIVX2)              0.07       7.37 r
add1/generate_block[19].f1/a2/U2/Z (CNR2IX2)            0.07       7.44 f
add1/generate_block[19].f1/a2/carry (adder_25)          0.00       7.44 f
add1/generate_block[19].f1/U2/Z (CIVX2)                 0.06       7.50 r
add1/generate_block[19].f1/U1/Z (CND2X2)                0.08       7.58 f
add1/generate_block[19].f1/cout (fulladd_13)            0.00       7.58 f
add1/generate_block[20].f1/cin (fulladd_12)             0.00       7.58 f
add1/generate_block[20].f1/a2/a (adder_23)              0.00       7.58 f
add1/generate_block[20].f1/a2/U2/Z (CAN2X1)             0.17       7.75 f
add1/generate_block[20].f1/a2/carry (adder_23)          0.00       7.75 f
add1/generate_block[20].f1/U2/Z (CIVX2)                 0.06       7.82 r
add1/generate_block[20].f1/U1/Z (CND2X2)                0.08       7.89 f
add1/generate_block[20].f1/cout (fulladd_12)            0.00       7.89 f
add1/generate_block[21].f1/cin (fulladd_11)             0.00       7.89 f
add1/generate_block[21].f1/a2/a (adder_21)              0.00       7.89 f
add1/generate_block[21].f1/a2/U2/Z (CAN2X1)             0.17       8.07 f
add1/generate_block[21].f1/a2/carry (adder_21)          0.00       8.07 f
add1/generate_block[21].f1/U2/Z (CIVX2)                 0.06       8.13 r
add1/generate_block[21].f1/U1/Z (CND2X2)                0.08       8.21 f
add1/generate_block[21].f1/cout (fulladd_11)            0.00       8.21 f
add1/generate_block[22].f1/cin (fulladd_10)             0.00       8.21 f
add1/generate_block[22].f1/a2/a (adder_19)              0.00       8.21 f
add1/generate_block[22].f1/a2/U2/Z (CAN2X1)             0.17       8.38 f
```

```
add1/generate_block[22].f1/a2/carry (adder_19)          0.00       8.38 f
add1/generate_block[22].f1/U2/Z (CIVX2)                 0.06       8.44 r
add1/generate_block[22].f1/U1/Z (CND2X2)                0.08       8.53 f
add1/generate_block[22].f1/cout (fulladd_10)            0.00       8.53 f
add1/generate_block[23].f1/cin (fulladd_9)              0.00       8.53 f
add1/generate_block[23].f1/a2/a (adder_17)              0.00       8.53 f
add1/generate_block[23].f1/a2/U3/Z (CAN2X1)             0.17       8.70 f
add1/generate_block[23].f1/a2/carry (adder_17)          0.00       8.70 f
add1/generate_block[23].f1/U1/Z (CIVX2)                 0.06       8.76 r
add1/generate_block[23].f1/U2/Z (CND2X2)                0.08       8.85 f
add1/generate_block[23].f1/cout (fulladd_9)             0.00       8.85 f
add1/generate_block[24].f1/cin (fulladd_8)              0.00       8.85 f
add1/generate_block[24].f1/a2/a (adder_15)              0.00       8.85 f
add1/generate_block[24].f1/a2/U3/Z (CAN2X1)             0.17       9.02 f
add1/generate_block[24].f1/a2/carry (adder_15)          0.00       9.02 f
add1/generate_block[24].f1/U2/Z (CIVX2)                 0.06       9.08 r
add1/generate_block[24].f1/U1/Z (CND2X2)                0.08       9.16 f
add1/generate_block[24].f1/cout (fulladd_8)             0.00       9.16 f
add1/generate_block[25].f1/cin (fulladd_7)              0.00       9.16 f
add1/generate_block[25].f1/a2/a (adder_13)              0.00       9.16 f
add1/generate_block[25].f1/a2/U2/Z (CAN2X1)             0.17       9.33 f
add1/generate_block[25].f1/a2/carry (adder_13)          0.00       9.33 f
add1/generate_block[25].f1/U1/Z (CIVX2)                 0.06       9.40 r
add1/generate_block[25].f1/U2/Z (CND2X2)                0.09       9.49 f
add1/generate_block[25].f1/cout (fulladd_7)             0.00       9.49 f
add1/generate_block[26].f1/cin (fulladd_6)              0.00       9.49 f
add1/generate_block[26].f1/a2/a (adder_11)              0.00       9.49 f
add1/generate_block[26].f1/a2/U5/Z (CND2X2)             0.07       9.56 r
add1/generate_block[26].f1/a2/U4/Z (CIVX2)              0.06       9.62 f
add1/generate_block[26].f1/a2/carry (adder_11)          0.00       9.62 f
add1/generate_block[26].f1/U2/Z (CIVX2)                 0.06       9.69 r
add1/generate_block[26].f1/U1/Z (CND2X2)                0.09       9.78 f
add1/generate_block[26].f1/cout (fulladd_6)             0.00       9.78 f
add1/generate_block[27].f1/cin (fulladd_5)              0.00       9.78 f
add1/generate_block[27].f1/a2/a (adder_9)               0.00       9.78 f
add1/generate_block[27].f1/a2/U5/Z (CND2X2)             0.07       9.85 r
add1/generate_block[27].f1/a2/U6/Z (CIVX2)              0.06       9.91 f
add1/generate_block[27].f1/a2/carry (adder_9)           0.00       9.91 f
add1/generate_block[27].f1/U1/Z (CIVX2)                 0.06       9.98 r
add1/generate_block[27].f1/U2/Z (CND2X2)                0.09      10.07 f
add1/generate_block[27].f1/cout (fulladd_5)             0.00      10.07 f
add1/generate_block[28].f1/cin (fulladd_4)              0.00      10.07 f
add1/generate_block[28].f1/a2/a (adder_7)               0.00      10.07 f
add1/generate_block[28].f1/a2/U2/Z (CND2X2)             0.07      10.14 r
add1/generate_block[28].f1/a2/U3/Z (CIVX2)              0.06      10.20 f
add1/generate_block[28].f1/a2/carry (adder_7)           0.00      10.20 f
add1/generate_block[28].f1/U2/Z (CIVX2)                 0.06      10.27 r
add1/generate_block[28].f1/U1/Z (CND2X2)                0.09      10.36 f
add1/generate_block[28].f1/cout (fulladd_4)             0.00      10.36 f
add1/generate_block[29].f1/cin (fulladd_3)              0.00      10.36 f
add1/generate_block[29].f1/a2/a (adder_5)               0.00      10.36 f
add1/generate_block[29].f1/a2/U2/Z (CND2X2)             0.07      10.43 r
add1/generate_block[29].f1/a2/U3/Z (CIVX2)              0.06      10.49 f
add1/generate_block[29].f1/a2/carry (adder_5)           0.00      10.49 f
add1/generate_block[29].f1/U2/Z (CIVX2)                 0.06      10.56 r
add1/generate_block[29].f1/U1/Z (CND2X2)                0.09      10.65 f
add1/generate_block[29].f1/cout (fulladd_3)             0.00      10.65 f
```

```
  add1/generate_block[30].f1/cin (fulladd_2)              0.00      10.65 f
  add1/generate_block[30].f1/a2/a (adder_3)               0.00      10.65 f
  add1/generate_block[30].f1/a2/U4/Z (CND2X2)             0.07      10.72 r
  add1/generate_block[30].f1/a2/U5/Z (CIVX2)              0.06      10.78 f
  add1/generate_block[30].f1/a2/carry (adder_3)           0.00      10.78 f
  add1/generate_block[30].f1/U2/Z (CIVX2)                 0.06      10.84 r
  add1/generate_block[30].f1/U1/Z (CND2IX2)               0.11      10.96 f
  add1/generate_block[30].f1/cout (fulladd_2)             0.00      10.96 f
  add1/generate_block[31].f1/cin (fulladd_1)              0.00      10.96 f
  add1/generate_block[31].f1/a2/a (adder_1)               0.00      10.96 f
  add1/generate_block[31].f1/a2/U1/Z (CENX2)              0.24      11.19 f
  add1/generate_block[31].f1/a2/sum (adder_1)             0.00      11.19 f
  add1/generate_block[31].f1/sum (fulladd_1)              0.00      11.19 f
  add1/sum[31] (add32)                                    0.00      11.19 f
  U754/Z (CNR2IX2)                                        0.12      11.32 r
  U509/Z (CNR2IX2)                                        0.08      11.40 f
  result_reg[31]/D (CFD3QX2)                              0.00      11.40 f
  data arrival time                                                 11.40

  clock clock (rise edge)                                12.00      12.00
  clock network delay (propagated)                        0.00      12.00
  clock uncertainty                                      -0.25      11.75
  result_reg[31]/CP (CFD3QX2)                             0.00      11.75 r
  library setup time                                     -0.35      11.40
  data required time                                                11.40
  --------------------------------------------------------------------------
  data required time                                                11.40
  data arrival time                                                -11.40
  --------------------------------------------------------------------------
  slack (MET)                                                       0.00
```

AREA REPORT FOR MULTIPLIER

```
report_area

****************************************
Report : area
Design : multiplier
Version: C-2009.06-SP5
Date   : Fri Dec  9 22:52:46 2016
****************************************

Library(s) Used:

    tc240c (File: /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db_WCCOM25)

Number of ports:              165
Number of nets:               936
Number of cells:              832
Number of references:          42

Combinational area:       1470.000000
Noncombinational area:     480.000000
Net Interconnect area:      undefined  (No wire load specified)

Total cell area:          1950.000000
Total area:                 undefined
```

POWER REPORT FOR MULTIPLIER

```
report_power
Loading db file '/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db_BCCOM25'
Loading db file '/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db_WCCOM25'
Warning: Main library 'tc240c' does not specify the following unit required
for power: 'Leakage Power'. (PWR-424)
Information: Propagating switching activity (low effort zero delay
simulation). (PWR-6)
Warning: Design has unannotated primary inputs. (PWR-414)
Warning: Design has unannotated sequential cell outputs. (PWR-415)

****************************************
Report : power
        -analysis_effort low
Design : multiplier
Version: C-2009.06-SP5
Date   : Fri Dec  9 22:52:47 2016
****************************************


Library(s) Used:

    tc240c (File: /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db_WCCOM25)


Operating Conditions: WCCOM25   Library: tc240c
Wire Load Model Mode: top


Global Operating Voltage = 2.3
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000ff
    Time Units = 1ns
    Dynamic Power Units = 1uW    (derived from V,C,T units)
    Leakage Power Units = Unitless


  Cell Internal Power  =   2.3949 mW   (89%)
  Net Switching Power  = 281.9348 uW   (11%)
                         ---------
Total Dynamic Power    =   2.6769 mW  (100%)

Cell Leakage Power     =   0.0000
```

SYNTHESIS REPORT FOR DIVIDER

```
ncverilog: 14.10-p001: (c) Copyright 1995-2014 Cadence Design Systems, Inc.
   Recompiling... reason: file './seq_netlist.v' is newer than expected.
                       expected: Fri Dec  9 17:08:49 2016
                       actual:   Fri Dec  9 22:57:18 2016
                            file: seq_netlist.v
                              module worklib.add32:v
                                  errors: 0, warnings: 0
                             module worklib.divider:v
                                  errors: 0, warnings: 0
          file: /apps/toshiba/sjsu/verilog/tc240c/CMXI2X2.tsbvlibp
                        module tc240c.CMXI2X2:tsbvlibp
                                  errors: 0, warnings: 0
           file: /apps/toshiba/sjsu/verilog/tc240c/CNR4X1.tsbvlibp
                         module tc240c.CNR4X1:tsbvlibp
                                  errors: 0, warnings: 0
         file: /apps/toshiba/sjsu/verilog/tc240c/TFDPNOprim.tsbvlibp
                       primitive tc240c.TFDPNOprim:tsbvlibp
                                  errors: 0, warnings: 0
                         Caching library 'tc240c' ....... Done
                         Caching library 'worklib' ....... Done
                        Elaborating the design hierarchy:
              CFD1XL sign_reg ( .D(n818), .CP(clock), .QN(n47) );
                                              |
ncelab: *W,CUVWSP (./seq_netlist.v,1273|16): 1 output port was not connected:
       ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): Q

             CFD1XL divres_reg ( .D(n814), .CP(clock), .Q(divres) );
                                             |
ncelab: *W,CUVWSP (./seq_netlist.v,1276|18): 1 output port was not connected:
       ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

    CFD1XL \next_state_reg[2]  ( .D(n849), .CP(clock), .Q(next_state[2]) );
                                                |
ncelab: *W,CUVWSP (./seq_netlist.v,1277|27): 1 output port was not connected:
       ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

              CFD1XL valid_reg ( .D(n813), .CP(clock), .Q(valid) );
                                             |
ncelab: *W,CUVWSP (./seq_netlist.v,1279|17): 1 output port was not connected:
       ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

    CFD1XL \next_state_reg[1]  ( .D(n811), .CP(clock), .Q(next_state[1]) );
                                                |
ncelab: *W,CUVWSP (./seq_netlist.v,1312|27): 1 output port was not connected:
       ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

         CFD1XL \tempB_reg[1]  ( .D(n713), .CP(clock), .Q(tempB[1]) );
                                            |
ncelab: *W,CUVWSP (./seq_netlist.v,1313|22): 1 output port was not connected:
       ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

         CFD1XL \tempB_reg[0]  ( .D(n714), .CP(clock), .Q(tempB[0]) );
                                            |
```

```
ncelab: *W,CUVWSP (./seq_netlist.v,1314|22): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

         CFD1XL \tempB_reg[2]  ( .D(n712), .CP(clock), .Q(tempB[2]) );
                                                       |
ncelab: *W,CUVWSP (./seq_netlist.v,1315|22): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

         CFD1XL \tempB_reg[3]  ( .D(n711), .CP(clock), .Q(tempB[3]) );
                                                       |
ncelab: *W,CUVWSP (./seq_netlist.v,1324|22): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

         CFD1XL \tempB_reg[4]  ( .D(n710), .CP(clock), .Q(tempB[4]) );
                                                       |
ncelab: *W,CUVWSP (./seq_netlist.v,1326|22): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

         CFD1XL \tempB_reg[5]  ( .D(n709), .CP(clock), .Q(tempB[5]) );
                                                       |
ncelab: *W,CUVWSP (./seq_netlist.v,1336|22): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

         CFD1XL \tempB_reg[6]  ( .D(n708), .CP(clock), .Q(tempB[6]) );
                                                       |
ncelab: *W,CUVWSP (./seq_netlist.v,1340|22): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

         CFD1XL \tempB_reg[7]  ( .D(n707), .CP(clock), .Q(tempB[7]) );
                                                       |
ncelab: *W,CUVWSP (./seq_netlist.v,1344|22): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

         CFD1XL \tempB_reg[8]  ( .D(n706), .CP(clock), .Q(tempB[8]) );
                                                       |
ncelab: *W,CUVWSP (./seq_netlist.v,1348|22): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

         CFD1XL \tempB_reg[9]  ( .D(n705), .CP(clock), .Q(tempB[9]) );
                                                       |
ncelab: *W,CUVWSP (./seq_netlist.v,1352|22): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

        CFD1XL \tempB_reg[10]  ( .D(n704), .CP(clock), .Q(tempB[10]) );
                                                        |
ncelab: *W,CUVWSP (./seq_netlist.v,1357|23): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

        CFD1XL \tempB_reg[11]  ( .D(n703), .CP(clock), .Q(tempB[11]) );
                                                       |
ncelab: *W,CUVWSP (./seq_netlist.v,1362|23): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

        CFD1XL \tempB_reg[12]  ( .D(n702), .CP(clock), .Q(tempB[12]) );
                                                       |
ncelab: *W,CUVWSP (./seq_netlist.v,1366|23): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN
```

```
        CFD1XL \tempB_reg[13]  ( .D(n701), .CP(clock), .Q(tempB[13]) );
                                                      |
ncelab: *W,CUVWSP (./seq_netlist.v,1374|23): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

        CFD1XL \tempB_reg[14]  ( .D(n700), .CP(clock), .Q(tempB[14]) );
                                                      |
ncelab: *W,CUVWSP (./seq_netlist.v,1377|23): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

        CFD1XL \tempB_reg[15]  ( .D(n699), .CP(clock), .Q(tempB[15]) );
                                                      |
ncelab: *W,CUVWSP (./seq_netlist.v,1382|23): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

        CFD1XL \tempB_reg[16]  ( .D(n698), .CP(clock), .Q(tempB[16]) );
                                                      |
ncelab: *W,CUVWSP (./seq_netlist.v,1387|23): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

        CFD1XL \tempB_reg[17]  ( .D(n697), .CP(clock), .Q(tempB[17]) );
                                                      |
ncelab: *W,CUVWSP (./seq_netlist.v,1391|23): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

        CFD1XL \tempB_reg[18]  ( .D(n696), .CP(clock), .Q(tempB[18]) );
                                                      |
ncelab: *W,CUVWSP (./seq_netlist.v,1399|23): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

        CFD1XL \tempB_reg[19]  ( .D(n695), .CP(clock), .Q(tempB[19]) );
                                                      |
ncelab: *W,CUVWSP (./seq_netlist.v,1402|23): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

        CFD1XL \tempB_reg[20]  ( .D(n694), .CP(clock), .Q(tempB[20]) );
                                                      |
ncelab: *W,CUVWSP (./seq_netlist.v,1407|23): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

        CFD1XL \tempB_reg[21]  ( .D(n693), .CP(clock), .Q(tempB[21]) );
                                                      |
ncelab: *W,CUVWSP (./seq_netlist.v,1412|23): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

        CFD1XL \tempB_reg[22]  ( .D(n692), .CP(clock), .Q(tempB[22]) );
                                                      |
ncelab: *W,CUVWSP (./seq_netlist.v,1416|23): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

        CFD1XL \tempB_reg[23]  ( .D(n691), .CP(clock), .Q(tempB[23]) );
                                                      |
ncelab: *W,CUVWSP (./seq_netlist.v,1424|23): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

        CFD1XL \tempB_reg[24]  ( .D(n690), .CP(clock), .Q(tempB[24]) );
```

```
                                                    |
ncelab: *W,CUVWSP (./seq_netlist.v,1427|23): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

        CFD1XL \tempB_reg[25]  ( .D(n689), .CP(clock), .Q(tempB[25]) );
                                                    |
ncelab: *W,CUVWSP (./seq_netlist.v,1432|23): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

        CFD1XL \tempB_reg[26]  ( .D(n688), .CP(clock), .Q(tempB[26]) );
                                                    |
ncelab: *W,CUVWSP (./seq_netlist.v,1437|23): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

        CFD1XL \tempB_reg[27]  ( .D(n687), .CP(clock), .Q(tempB[27]) );
                                                    |
ncelab: *W,CUVWSP (./seq_netlist.v,1442|23): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

        CFD1XL \tempB_reg[28]  ( .D(n686), .CP(clock), .Q(tempB[28]) );
                                                    |
ncelab: *W,CUVWSP (./seq_netlist.v,1453|23): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

        CFD1XL \tempB_reg[29]  ( .D(n685), .CP(clock), .Q(tempB[29]) );
                                                    |
ncelab: *W,CUVWSP (./seq_netlist.v,1458|23): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

        CFD1XL \tempB_reg[30]  ( .D(n684), .CP(clock), .Q(tempB[30]) );
                                                    |
ncelab: *W,CUVWSP (./seq_netlist.v,1463|23): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

       CFD1XL \result_reg[63]  ( .D(n1015), .CP(clock), .Q(result[63]) );
                                                     |
ncelab: *W,CUVWSP (./seq_netlist.v,1464|24): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

      CFD1XL \tempdiv_reg[30]  ( .D(n715), .CP(clock), .Q(tempdiv[30]) );
                                                    |
ncelab: *W,CUVWSP (./seq_netlist.v,1465|25): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

        CFD1XL \tempB_reg[31]  ( .D(n683), .CP(clock), .Q(tempB[31]) );
                                                    |
ncelab: *W,CUVWSP (./seq_netlist.v,1466|23): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

    CFD1XL \next_state_reg[0]  ( .D(n852), .CP(clock), .Q(next_state[0]) );
                                                     |
ncelab: *W,CUVWSP (./seq_netlist.v,1467|27): 1 output port was not connected:
      ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

               CFD1XL neg_reg ( .D(n810), .CP(clock), .Q(neg) );
                                                   |
ncelab: *W,CUVWSP (./seq_netlist.v,1470|15): 1 output port was not connected:
```

```
       ncelab: (/apps/toshiba/sjsu/verilog/tc240c/CFD1XL.tsbvlibp,7): QN

             Building instance overlay tables: ..................... Done
                          Generating native compiled code:
                           tc240c.CMXI2X2:tsbvlibp <0x53019784>
                                   streams:   0, words:     0
                           tc240c.COND1X1:tsbvlibp <0x1ed76f35>
                                   streams:   0, words:     0
                        Building instance specific data structures.
             Loading native compiled code:     ..................... Done
                            Design hierarchy summary:
                                          Instances  Unique
                           Modules:            1519     140
                           UDPs:                173       2
                           Primitives:         7653       8
                           Timing outputs:     1420      29
                           Registers:           178      10
                           Scalar wires:       1595       -
                           Expanded wires:       96       2
                           Always blocks:         1       1
                           Initial blocks:        3       3
                           Pseudo assignments:    2       2
                           Timing checks:      1020     341
                Simulation timescale:  10ps
```

MAXIMUM PATH IN DIVIDER

```
report_timing

****************************************
Report : timing
        -path full
        -delay max
        -max_paths 1
Design : divider
Version: C-2009.06-SP5
Date   : Fri Dec  9 22:57:16 2016
****************************************

Operating Conditions: WCCOM25   Library: tc240c
Wire Load Model Mode: top

  Startpoint: neg_reg (rising edge-triggered flip-flop clocked by clock)
  Endpoint: neg_reg (rising edge-triggered flip-flop clocked by clock)
  Path Group: clock
  Path Type: max

  Point                                                   Incr       Path
  ---------------------------------------------------------------------------
  clock clock (rise edge)                                 0.00       0.00
  clock network delay (propagated)                        0.00       0.00
  neg_reg/CP (CFD1XL)                                     0.00       0.00 r
  neg_reg/Q (CFD1XL)                                      0.72       0.72 f
  add1/cin (add32)                                        0.00       0.72 f
  add1/U2/Z (CNIVX1)                                      0.66       1.39 f
  add1/U5/Z (CEOX1)                                       0.40       1.78 f
```

```
add1/generate_block[0].f1/a (fulladd_0)                 0.00       1.78 f
add1/generate_block[0].f1/a1/a (adder_0)                0.00       1.78 f
add1/generate_block[0].f1/a1/U1/Z (CEOX1)               0.32       2.10 f
add1/generate_block[0].f1/a1/sum (adder_0)              0.00       2.10 f
add1/generate_block[0].f1/a2/b (adder_63)               0.00       2.10 f
add1/generate_block[0].f1/a2/U1/Z (CAN2X1)              0.16       2.26 f
add1/generate_block[0].f1/a2/carry (adder_63)           0.00       2.26 f
add1/generate_block[0].f1/U1/Z (COR2X1)                 0.26       2.52 f
add1/generate_block[0].f1/cout (fulladd_0)              0.00       2.52 f
add1/generate_block[1].f1/cin (fulladd_31)              0.00       2.52 f
add1/generate_block[1].f1/a2/a (adder_61)               0.00       2.52 f
add1/generate_block[1].f1/a2/U2/Z (CAN2X1)              0.18       2.70 f
add1/generate_block[1].f1/a2/carry (adder_61)           0.00       2.70 f
add1/generate_block[1].f1/U1/Z (COR2X1)                 0.24       2.94 f
add1/generate_block[1].f1/cout (fulladd_31)             0.00       2.94 f
add1/generate_block[2].f1/cin (fulladd_30)              0.00       2.94 f
add1/generate_block[2].f1/a2/a (adder_59)               0.00       2.94 f
add1/generate_block[2].f1/a2/U2/Z (CAN2X1)              0.17       3.11 f
add1/generate_block[2].f1/a2/carry (adder_59)           0.00       3.11 f
add1/generate_block[2].f1/U1/Z (COR2X1)                 0.24       3.35 f
add1/generate_block[2].f1/cout (fulladd_30)             0.00       3.35 f
add1/generate_block[3].f1/cin (fulladd_29)              0.00       3.35 f
add1/generate_block[3].f1/a2/a (adder_57)               0.00       3.35 f
add1/generate_block[3].f1/a2/U2/Z (CAN2X1)              0.17       3.53 f
add1/generate_block[3].f1/a2/carry (adder_57)           0.00       3.53 f
add1/generate_block[3].f1/U1/Z (COR2X1)                 0.26       3.78 f
add1/generate_block[3].f1/cout (fulladd_29)             0.00       3.78 f
add1/generate_block[4].f1/cin (fulladd_28)              0.00       3.78 f
add1/generate_block[4].f1/a2/a (adder_55)               0.00       3.78 f
add1/generate_block[4].f1/a2/U2/Z (CAN2X1)              0.18       3.96 f
add1/generate_block[4].f1/a2/carry (adder_55)           0.00       3.96 f
add1/generate_block[4].f1/U1/Z (COR2X1)                 0.26       4.22 f
add1/generate_block[4].f1/cout (fulladd_28)             0.00       4.22 f
add1/generate_block[5].f1/cin (fulladd_27)              0.00       4.22 f
add1/generate_block[5].f1/a2/a (adder_53)               0.00       4.22 f
add1/generate_block[5].f1/a2/U2/Z (CAN2X1)              0.18       4.40 f
add1/generate_block[5].f1/a2/carry (adder_53)           0.00       4.40 f
add1/generate_block[5].f1/U1/Z (COR2X1)                 0.26       4.65 f
add1/generate_block[5].f1/cout (fulladd_27)             0.00       4.65 f
add1/generate_block[6].f1/cin (fulladd_26)              0.00       4.65 f
add1/generate_block[6].f1/a2/a (adder_51)               0.00       4.65 f
add1/generate_block[6].f1/a2/U2/Z (CAN2X1)              0.18       4.83 f
add1/generate_block[6].f1/a2/carry (adder_51)           0.00       4.83 f
add1/generate_block[6].f1/U1/Z (COR2X1)                 0.26       5.09 f
add1/generate_block[6].f1/cout (fulladd_26)             0.00       5.09 f
add1/generate_block[7].f1/cin (fulladd_25)              0.00       5.09 f
add1/generate_block[7].f1/a2/a (adder_49)               0.00       5.09 f
add1/generate_block[7].f1/a2/U2/Z (CAN2X1)              0.18       5.26 f
add1/generate_block[7].f1/a2/carry (adder_49)           0.00       5.26 f
add1/generate_block[7].f1/U1/Z (COR2X1)                 0.26       5.52 f
add1/generate_block[7].f1/cout (fulladd_25)             0.00       5.52 f
add1/generate_block[8].f1/cin (fulladd_24)              0.00       5.52 f
add1/generate_block[8].f1/a2/a (adder_47)               0.00       5.52 f
add1/generate_block[8].f1/a2/U2/Z (CAN2X1)              0.18       5.70 f
add1/generate_block[8].f1/a2/carry (adder_47)           0.00       5.70 f
add1/generate_block[8].f1/U1/Z (COR2X1)                 0.26       5.96 f
add1/generate_block[8].f1/cout (fulladd_24)             0.00       5.96 f
```

```
add1/generate_block[9].f1/cin (fulladd_23)              0.00       5.96 f
add1/generate_block[9].f1/a2/a (adder_45)               0.00       5.96 f
add1/generate_block[9].f1/a2/U2/Z (CAN2X1)              0.18       6.13 f
add1/generate_block[9].f1/a2/carry (adder_45)           0.00       6.13 f
add1/generate_block[9].f1/U1/Z (COR2X1)                 0.26       6.39 f
add1/generate_block[9].f1/cout (fulladd_23)             0.00       6.39 f
add1/generate_block[10].f1/cin (fulladd_22)             0.00       6.39 f
add1/generate_block[10].f1/a2/a (adder_43)              0.00       6.39 f
add1/generate_block[10].f1/a2/U2/Z (CAN2X1)             0.18       6.57 f
add1/generate_block[10].f1/a2/carry (adder_43)          0.00       6.57 f
add1/generate_block[10].f1/U1/Z (COR2X1)                0.26       6.83 f
add1/generate_block[10].f1/cout (fulladd_22)            0.00       6.83 f
add1/generate_block[11].f1/cin (fulladd_21)             0.00       6.83 f
add1/generate_block[11].f1/a2/a (adder_41)              0.00       6.83 f
add1/generate_block[11].f1/a2/U2/Z (CAN2X1)             0.18       7.00 f
add1/generate_block[11].f1/a2/carry (adder_41)          0.00       7.00 f
add1/generate_block[11].f1/U1/Z (COR2X1)                0.26       7.26 f
add1/generate_block[11].f1/cout (fulladd_21)            0.00       7.26 f
add1/generate_block[12].f1/cin (fulladd_20)             0.00       7.26 f
add1/generate_block[12].f1/a2/a (adder_39)              0.00       7.26 f
add1/generate_block[12].f1/a2/U2/Z (CAN2X1)             0.18       7.44 f
add1/generate_block[12].f1/a2/carry (adder_39)          0.00       7.44 f
add1/generate_block[12].f1/U1/Z (COR2X1)                0.24       7.68 f
add1/generate_block[12].f1/cout (fulladd_20)            0.00       7.68 f
add1/generate_block[13].f1/cin (fulladd_19)             0.00       7.68 f
add1/generate_block[13].f1/a2/a (adder_37)              0.00       7.68 f
add1/generate_block[13].f1/a2/U2/Z (CAN2X1)             0.17       7.85 f
add1/generate_block[13].f1/a2/carry (adder_37)          0.00       7.85 f
add1/generate_block[13].f1/U1/Z (COR2X1)                0.26       8.11 f
add1/generate_block[13].f1/cout (fulladd_19)            0.00       8.11 f
add1/generate_block[14].f1/cin (fulladd_18)             0.00       8.11 f
add1/generate_block[14].f1/a2/a (adder_35)              0.00       8.11 f
add1/generate_block[14].f1/a2/U2/Z (CAN2X1)             0.18       8.29 f
add1/generate_block[14].f1/a2/carry (adder_35)          0.00       8.29 f
add1/generate_block[14].f1/U1/Z (COR2X1)                0.26       8.54 f
add1/generate_block[14].f1/cout (fulladd_18)            0.00       8.54 f
add1/generate_block[15].f1/cin (fulladd_17)             0.00       8.54 f
add1/generate_block[15].f1/a2/a (adder_33)              0.00       8.54 f
add1/generate_block[15].f1/a2/U2/Z (CAN2X1)             0.18       8.72 f
add1/generate_block[15].f1/a2/carry (adder_33)          0.00       8.72 f
add1/generate_block[15].f1/U1/Z (COR2X1)                0.26       8.98 f
add1/generate_block[15].f1/cout (fulladd_17)            0.00       8.98 f
add1/generate_block[16].f1/cin (fulladd_16)             0.00       8.98 f
add1/generate_block[16].f1/a2/a (adder_31)              0.00       8.98 f
add1/generate_block[16].f1/a2/U2/Z (CAN2X1)             0.18       9.15 f
add1/generate_block[16].f1/a2/carry (adder_31)          0.00       9.15 f
add1/generate_block[16].f1/U1/Z (COR2X1)                0.26       9.41 f
add1/generate_block[16].f1/cout (fulladd_16)            0.00       9.41 f
add1/generate_block[17].f1/cin (fulladd_15)             0.00       9.41 f
add1/generate_block[17].f1/a2/a (adder_29)              0.00       9.41 f
add1/generate_block[17].f1/a2/U2/Z (CAN2X1)             0.18       9.59 f
add1/generate_block[17].f1/a2/carry (adder_29)          0.00       9.59 f
add1/generate_block[17].f1/U1/Z (COR2X1)                0.24       9.83 f
add1/generate_block[17].f1/cout (fulladd_15)            0.00       9.83 f
add1/generate_block[18].f1/cin (fulladd_14)             0.00       9.83 f
add1/generate_block[18].f1/a2/a (adder_27)              0.00       9.83 f
add1/generate_block[18].f1/a2/U2/Z (CAN2X1)             0.17      10.00 f
```

```
add1/generate_block[18].f1/a2/carry (adder_27)          0.00      10.00 f
add1/generate_block[18].f1/U1/Z (COR2X1)                0.26      10.26 f
add1/generate_block[18].f1/cout (fulladd_14)            0.00      10.26 f
add1/generate_block[19].f1/cin (fulladd_13)             0.00      10.26 f
add1/generate_block[19].f1/a2/a (adder_25)              0.00      10.26 f
add1/generate_block[19].f1/a2/U2/Z (CAN2X1)             0.18      10.44 f
add1/generate_block[19].f1/a2/carry (adder_25)          0.00      10.44 f
add1/generate_block[19].f1/U1/Z (COR2X1)                0.26      10.70 f
add1/generate_block[19].f1/cout (fulladd_13)            0.00      10.70 f
add1/generate_block[20].f1/cin (fulladd_12)             0.00      10.70 f
add1/generate_block[20].f1/a2/a (adder_23)              0.00      10.70 f
add1/generate_block[20].f1/a2/U2/Z (CAN2X1)             0.18      10.87 f
add1/generate_block[20].f1/a2/carry (adder_23)          0.00      10.87 f
add1/generate_block[20].f1/U1/Z (COR2X1)                0.26      11.13 f
add1/generate_block[20].f1/cout (fulladd_12)            0.00      11.13 f
add1/generate_block[21].f1/cin (fulladd_11)             0.00      11.13 f
add1/generate_block[21].f1/a2/a (adder_21)              0.00      11.13 f
add1/generate_block[21].f1/a2/U2/Z (CAN2X1)             0.18      11.31 f
add1/generate_block[21].f1/a2/carry (adder_21)          0.00      11.31 f
add1/generate_block[21].f1/U1/Z (COR2X1)                0.26      11.57 f
add1/generate_block[21].f1/cout (fulladd_11)            0.00      11.57 f
add1/generate_block[22].f1/cin (fulladd_10)             0.00      11.57 f
add1/generate_block[22].f1/a2/a (adder_19)              0.00      11.57 f
add1/generate_block[22].f1/a2/U2/Z (CAN2X1)             0.18      11.74 f
add1/generate_block[22].f1/a2/carry (adder_19)          0.00      11.74 f
add1/generate_block[22].f1/U1/Z (COR2X1)                0.24      11.98 f
add1/generate_block[22].f1/cout (fulladd_10)            0.00      11.98 f
add1/generate_block[23].f1/cin (fulladd_9)              0.00      11.98 f
add1/generate_block[23].f1/a2/a (adder_17)              0.00      11.98 f
add1/generate_block[23].f1/a2/U2/Z (CAN2X1)             0.17      12.16 f
add1/generate_block[23].f1/a2/carry (adder_17)          0.00      12.16 f
add1/generate_block[23].f1/U1/Z (COR2X1)                0.26      12.42 f
add1/generate_block[23].f1/cout (fulladd_9)             0.00      12.42 f
add1/generate_block[24].f1/cin (fulladd_8)              0.00      12.42 f
add1/generate_block[24].f1/a2/a (adder_15)              0.00      12.42 f
add1/generate_block[24].f1/a2/U2/Z (CAN2X1)             0.18      12.59 f
add1/generate_block[24].f1/a2/carry (adder_15)          0.00      12.59 f
add1/generate_block[24].f1/U1/Z (COR2X1)                0.26      12.85 f
add1/generate_block[24].f1/cout (fulladd_8)             0.00      12.85 f
add1/generate_block[25].f1/cin (fulladd_7)              0.00      12.85 f
add1/generate_block[25].f1/a2/a (adder_13)              0.00      12.85 f
add1/generate_block[25].f1/a2/U2/Z (CAN2X1)             0.18      13.03 f
add1/generate_block[25].f1/a2/carry (adder_13)          0.00      13.03 f
add1/generate_block[25].f1/U1/Z (COR2X1)                0.26      13.28 f
add1/generate_block[25].f1/cout (fulladd_7)             0.00      13.28 f
add1/generate_block[26].f1/cin (fulladd_6)              0.00      13.28 f
add1/generate_block[26].f1/a2/a (adder_11)              0.00      13.28 f
add1/generate_block[26].f1/a2/U2/Z (CAN2X1)             0.18      13.46 f
add1/generate_block[26].f1/a2/carry (adder_11)          0.00      13.46 f
add1/generate_block[26].f1/U1/Z (COR2X1)                0.26      13.72 f
add1/generate_block[26].f1/cout (fulladd_6)             0.00      13.72 f
add1/generate_block[27].f1/cin (fulladd_5)              0.00      13.72 f
add1/generate_block[27].f1/a2/a (adder_9)               0.00      13.72 f
add1/generate_block[27].f1/a2/U2/Z (CAN2X1)             0.18      13.90 f
add1/generate_block[27].f1/a2/carry (adder_9)           0.00      13.90 f
add1/generate_block[27].f1/U1/Z (COR2X1)                0.24      14.14 f
add1/generate_block[27].f1/cout (fulladd_5)             0.00      14.14 f
```

```
  add1/generate_block[28].f1/cin (fulladd_4)              0.00      14.14 f
  add1/generate_block[28].f1/a2/a (adder_7)               0.00      14.14 f
  add1/generate_block[28].f1/a2/U2/Z (CAN2X1)             0.17      14.31 f
  add1/generate_block[28].f1/a2/carry (adder_7)           0.00      14.31 f
  add1/generate_block[28].f1/U1/Z (COR2X1)                0.26      14.57 f
  add1/generate_block[28].f1/cout (fulladd_4)             0.00      14.57 f
  add1/generate_block[29].f1/cin (fulladd_3)              0.00      14.57 f
  add1/generate_block[29].f1/a2/a (adder_5)               0.00      14.57 f
  add1/generate_block[29].f1/a2/U2/Z (CAN2X1)             0.18      14.74 f
  add1/generate_block[29].f1/a2/carry (adder_5)           0.00      14.74 f
  add1/generate_block[29].f1/U1/Z (COR2X1)                0.26      15.00 f
  add1/generate_block[29].f1/cout (fulladd_3)             0.00      15.00 f
  add1/generate_block[30].f1/cin (fulladd_2)              0.00      15.00 f
  add1/generate_block[30].f1/a2/a (adder_3)               0.00      15.00 f
  add1/generate_block[30].f1/a2/U2/Z (CAN2X1)             0.18      15.18 f
  add1/generate_block[30].f1/a2/carry (adder_3)           0.00      15.18 f
  add1/generate_block[30].f1/U1/Z (COR2X1)                0.26      15.44 f
  add1/generate_block[30].f1/cout (fulladd_2)             0.00      15.44 f
  add1/generate_block[31].f1/cin (fulladd_1)              0.00      15.44 f
  add1/generate_block[31].f1/a2/a (adder_1)               0.00      15.44 f
  add1/generate_block[31].f1/a2/U1/Z (CEOX1)              0.35      15.79 f
  add1/generate_block[31].f1/a2/sum (adder_1)             0.00      15.79 f
  add1/generate_block[31].f1/sum (fulladd_1)              0.00      15.79 f
  add1/sum[31] (add32)                                    0.00      15.79 f
  U849/Z (CAOR2X1)                                        0.40      16.19 f
  U1265/Z (CANR3X1)                                       0.23      16.42 r
  U1674/Z (COND3X1)                                       0.27      16.69 f
  U677/Z (CIVX2)                                          0.07      16.76 r
  U1673/Z (CANR11X1)                                      0.11      16.88 f
  U1672/Z (CAOR1XL)                                       0.36      17.24 f
  neg_reg/D (CFD1XL)                                      0.00      17.24 f
  data arrival time                                                 17.24

  clock clock (rise edge)                                18.00      18.00
  clock network delay (propagated)                        0.00      18.00
  clock uncertainty                                      -0.25      17.75
  neg_reg/CP (CFD1XL)                                     0.00      17.75 r
  library setup time                                     -0.48      17.27
  data required time                                                17.27
  --------------------------------------------------------------------------
  data required time                                                17.27
  data arrival time                                                -17.24
  --------------------------------------------------------------------------
  slack (MET)                                                       0.03
```

AREA REPORT FOR DIVIDER

```
report_area

****************************************
Report : area
Design : divider
Version: C-2009.06-SP5
Date   : Fri Dec  9 22:57:16 2016
****************************************
```

```
Library(s) Used:

    tc240c (File: /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db_WCCOM25)

Number of ports:              165
Number of nets:              1318
Number of cells:             1057
Number of references:          38

Combinational area:       1797.500000
Noncombinational area:     680.000000
Net Interconnect area:      undefined  (No wire load specified)

Total cell area:          2477.500000
Total area:                 undefined
```

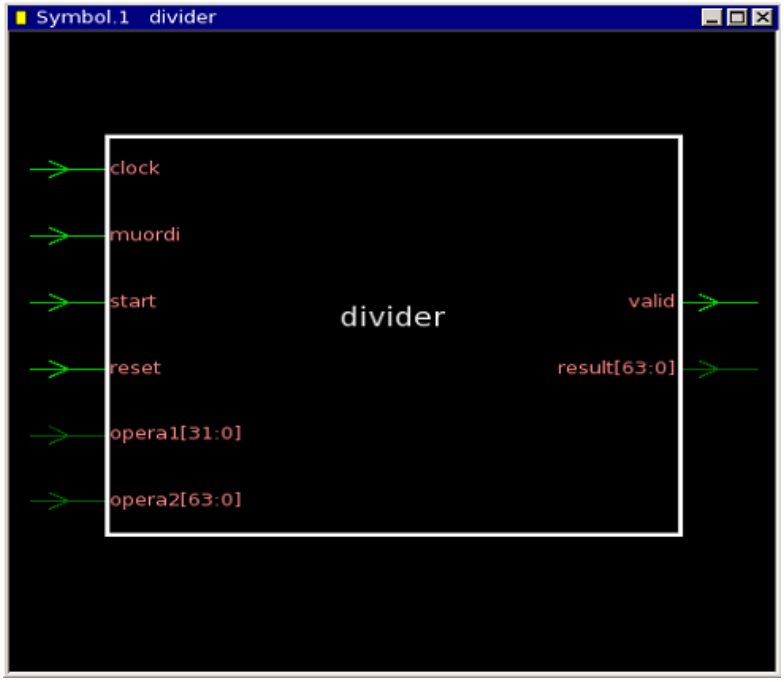
POWER REPORT FOR DIVIDER

```
report_power
Loading db file '/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db_BCCOM25'
Loading db file '/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db_WCCOM25'
Warning: Main library 'tc240c' does not specify the following unit required
for power: 'Leakage Power'. (PWR-424)
Information: Propagating switching activity (low effort zero delay
simulation). (PWR-6)
Warning: Design has unannotated primary inputs. (PWR-414)
Warning: Design has unannotated sequential cell outputs. (PWR-415)

****************************************
Report : power
        -analysis_effort low
Design : divider
Version: C-2009.06-SP5
Date   : Fri Dec  9 22:57:18 2016
****************************************


Library(s) Used:

    tc240c (File: /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db_WCCOM25)


Operating Conditions: WCCOM25   Library: tc240c
Wire Load Model Mode: top


Global Operating Voltage = 2.3
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000ff
    Time Units = 1ns
    Dynamic Power Units = 1uW    (derived from V,C,T units)
    Leakage Power Units = Unitless
```

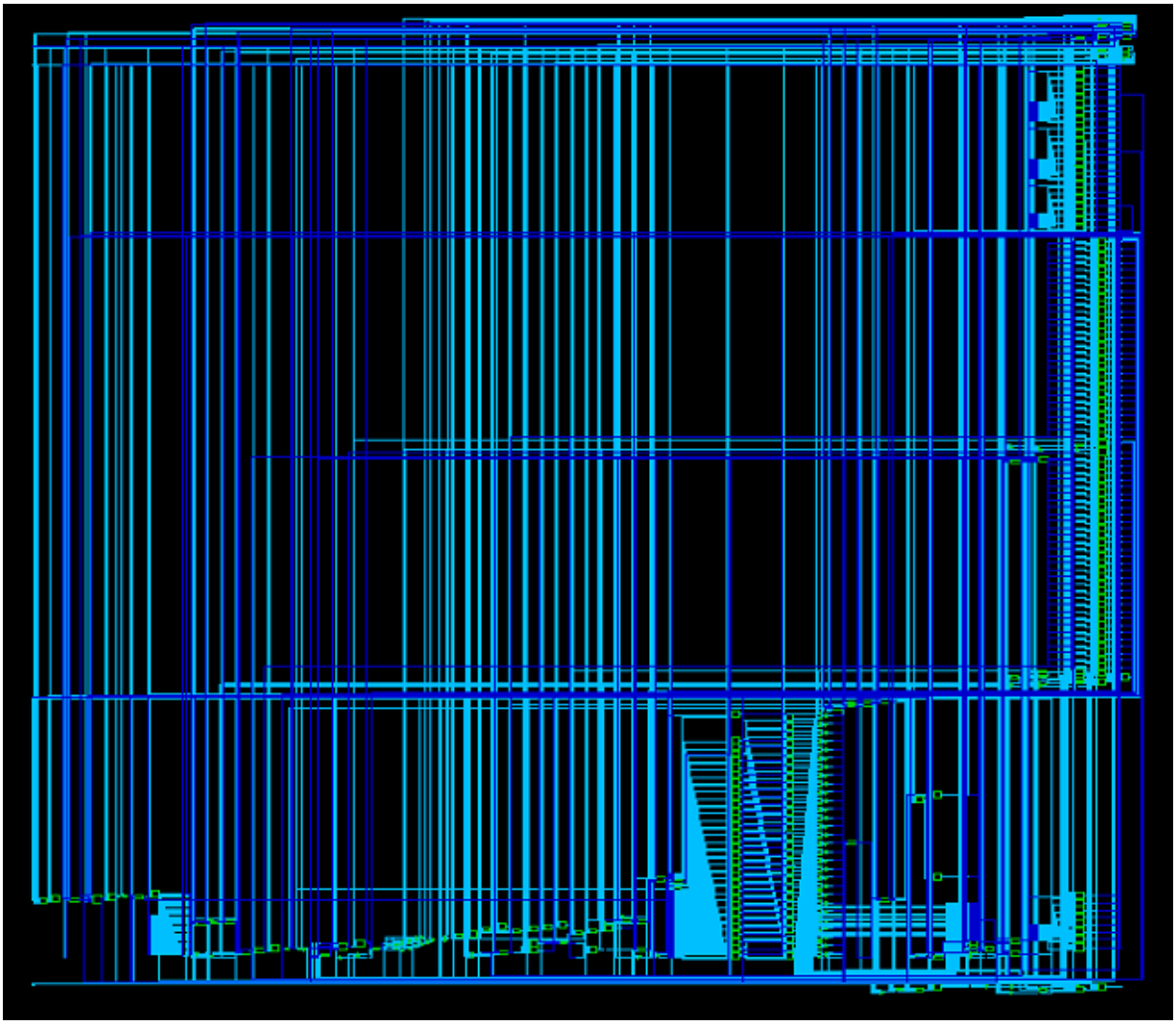
```
  Cell Internal Power  =   2.1447 mW   (87%)
  Net Switching Power  = 308.6608 uW   (13%)
                         ---------
Total Dynamic Power    =   2.4533 mW  (100%)

Cell Leakage Power     =   0.0000
```

C.4 Selected Screenshot Circuits from Synthesis (Design Compiler)

Divider block diagram

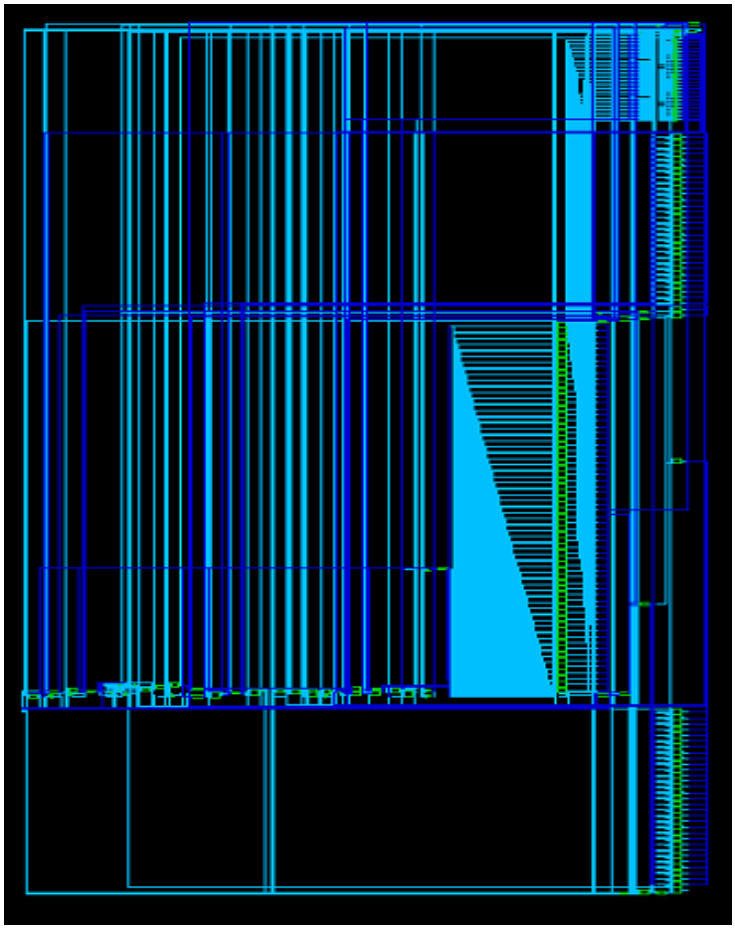


Schematic view

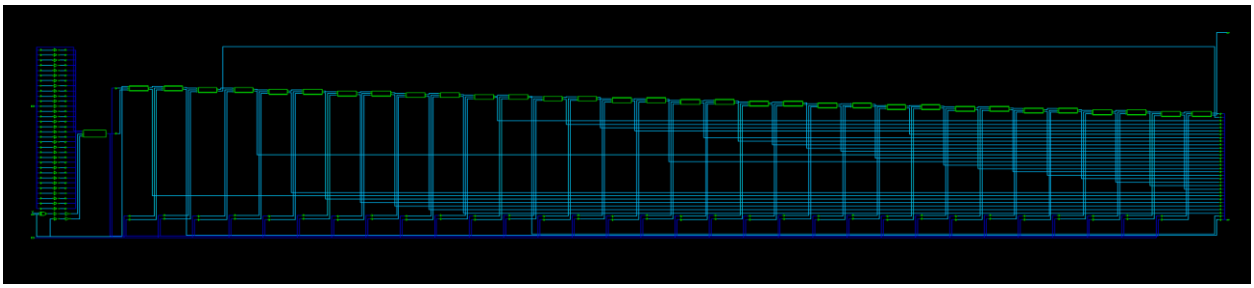


Multiplier Block diagram

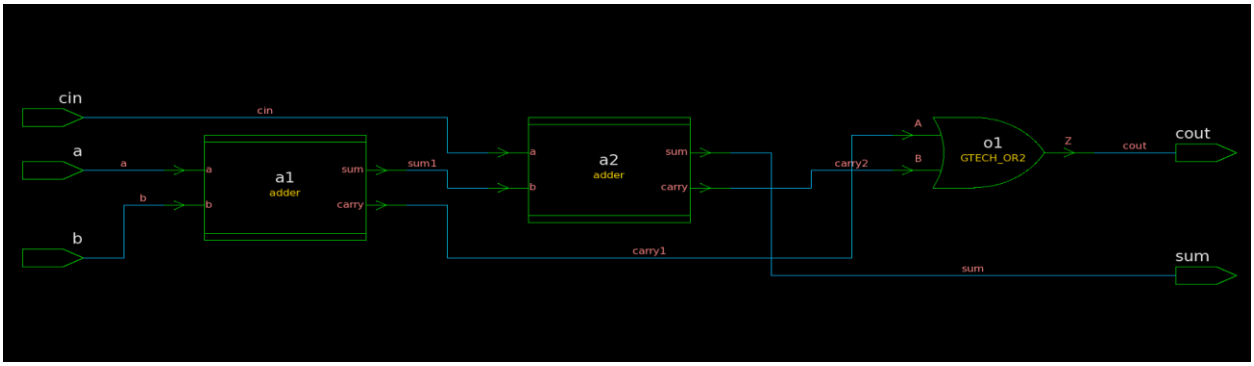
Multiplier schematic diagram



32 bit Adder block diagram



Full adder block diagram



Half adder block diagram