

Review Lab : N-Bit Adder

Chue Zhang

Csc343 Fall 2021

Professor Gertner

Table of contents

| | |
|---|----|
| Task 1: Half Adder..... | 3 |
| Created Half Adder in Quartus and simulated in Modelsim | |
| Task 2: Full Adder..... | 5 |
| Created Full Adder in Quartus and simulated in Modelsim | |
| Task 3: 4-bit Adder..... | 7 |
| Created 4-bit Adder in Quartus and simulated in Modelsim | |
| Task 4: 4-bit Adder/Sub..... | 9 |
| Created 4-bit Adder/Subtractor in Quartus and simulated in Modelsim | |
| Task 5: Adder Package..... | 12 |
| Created Adder package, used in Task 10 | |
| Task 6: N-bit Adder/Sub..... | 13 |
| Created N-bit Adder/Subtractor in Quartus and simulated in Modelsim | |
| Task 7: N-Bit Adder/Subtractor with Flags..... | 16 |
| Created 4-bit Adder/Subtractor with flags in Quartus and simulated in Modelsim | |
| Task 8: Testing design of N-bit Adder/Subtractor with flags..... | 19 |
| Further simulating and testing N-Bit Adder/Subtractor with flags in Modelsim | |
| Task 9: LPM N-bit Adder/subtractor with flags..... | 33 |
| Created LPM N-bit Adder/Subtractor in Quartus and simulated in Modelsim | |
| Task 10: Test bench..... | 49 |
| Created testbench for N-bit Adder/Subtractor in Quartus and simulated in Modelsim | |

Chue Zhang , Task 1: Half-Adder

The screenshot shows the Quartus Prime Lite Edition interface with the following details:

- Title Bar:** Quartus Prime Lite Edition - C:/Users/czhan/Desktop/Schoolwork/Gertner/Zhang_CS343_FA21/ZHANG_SEPTMBER19_ADDERS/ZHANG_SEPTMBER19_ADDERS - ZHANG_SEPTMBER19_ADDERS
- Project Navigator:** Shows files including ZHANG_SEPTMBER19_LPM_ADDSUB.qip, ZHANG_SEPTMBER19_LPM_ADDSUB.vhd, ZHANG_SEPTMBER19_N_Bit_AddSub.vhd, ZHANG_SEPTMBER19_HalfAdder.vhd, ZHANG_SEPTMBER19_fulladd_package.vhd, ZHANG_SEPTMBER19_FullAdder.vhd, ZHANG_SEPTMBER19_Four_Bit_AddSub.vhd, ZHANG_SEPTMBER19_Four_Bit_Adder.vhd, and ZHANG_SEPTMBER19_N_Bit_AddSub_Flag.vhd.
- Editor:** Displays the VHDL code for ZHANG_SEPTMBER19_HalfAdder.vhd. The code defines an entity ZHANG_SEPTMBER19_HalfAdder with a port (X, Y, S, C) and an architecture ZHANG_SEPTMBER19_arch with two processes (P1, P2) performing XOR and AND operations respectively.
- IP Catalog:** Shows sections for Installed IP, Project Directory (No Selection Available), Library (Basic Functions, DSP, Interface Protocols, Memory Interfaces and Controllers, Processors and Peripherals, University Program), and a search bar for Partner IP.
- Tasks:** A table showing the compilation tasks: Compile Design (00:01:00), Analysis & Synthesis (00:00:09), Fit (Place & Route) (00:00:40), Assembler (Generate programming files) (00:00:07), Timing Analysis (00:00:04), EDA Netlist Writer, Edit Settings, and Program Device (Open Programmer).
- Messages:** A list of messages with IDs and descriptions. It includes 332140 entries for recovery and removal paths, 332102 entries for design constraints, and 293000 entries for successful compilation.

Half-Adder coded in VHDL using two processes, we XOR inputs X and Y because '0' XOR '1' = 1 which is essentially the sum of 0 and 1. Furthermore, '1' XOR '1' gives 0 which is processed properly in P2 where we do '1' AND '1' which determines the sum as a value with a carry out or excess value.

Chue Zhang , Task 1: Half-Adder

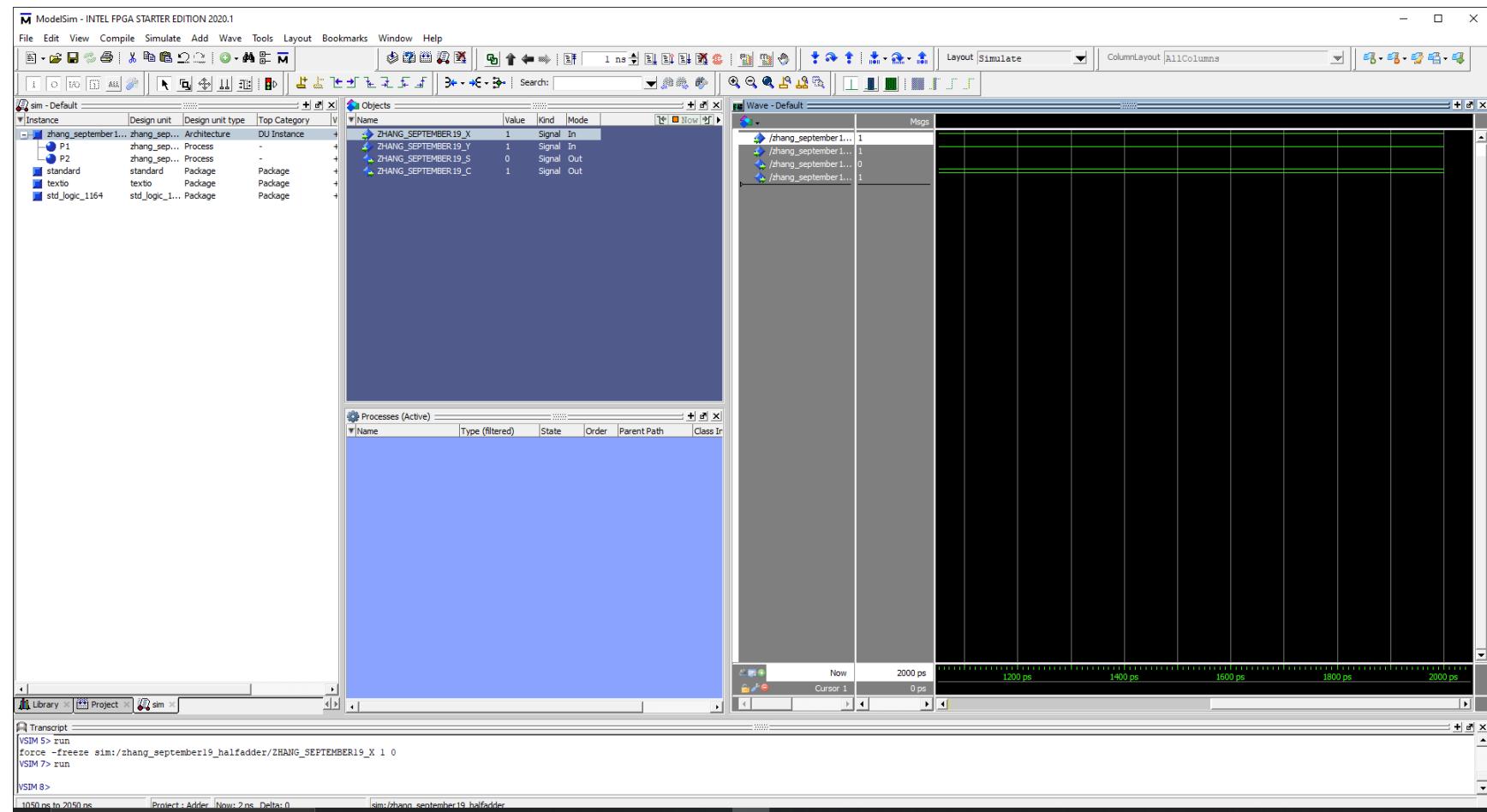


Figure above showcases input values of 1 and 1 which gives a sum value of '0' and a carry out value of '1'

Chue Zhang , Task 2: Full Adder

The screenshot shows the Quartus Prime Lite Edition interface with the following details:

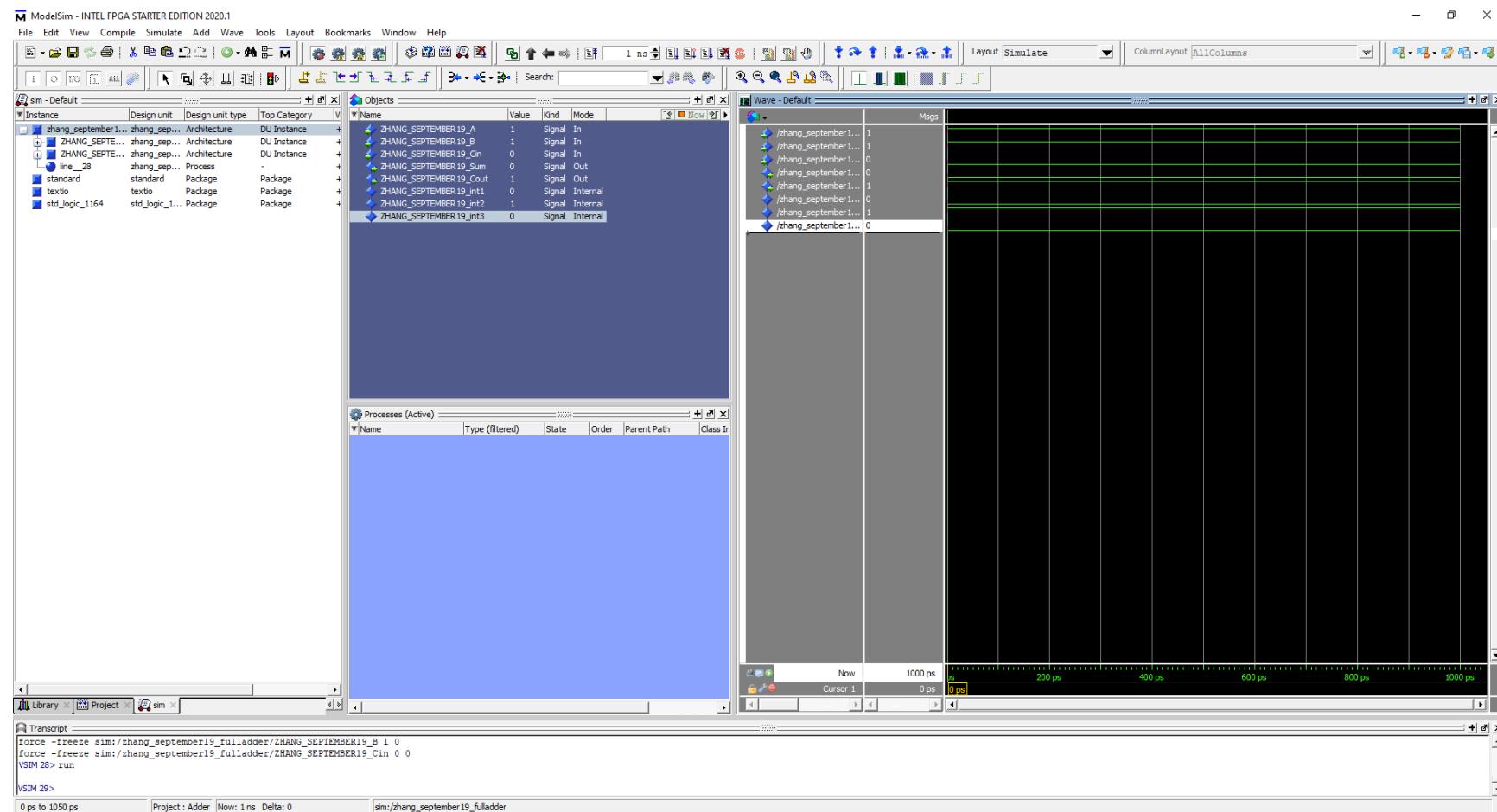
- Title Bar:** Quartus Prime Lite Edition - C:/Users/czhan/Desktop/Schoolwork/Gertner/Zhang_CS43_FA21/ZHANG_SEPTMBER19_ADDERS/ZHANG_SEPTMBER19_ADDERS - ZHANG_SEPTMBER19_ADDERS
- File Menu:** File Edit View Project Assignments Processing Tools Window Help
- Project Navigator:** Shows files including ZHANG_SEPTMBER19_LPM_ADDSUB.vhd, ZHANG_SEPTMBER19_LPM_ADDSUB.vhd, ZHANG_SEPTMBER19_HalfAdder.vhd, ZHANG_SEPTMBER19_fulladd_package.vhd, ZHANG_SEPTMBER19_FullAdder.vhd, ZHANG_SEPTMBER19_Four_Bit_AddSub.vhd, ZHANG_SEPTMBER19_Four_Bit_Adder.vhd, and ZHANG_SEPTMBER19_N_Bit_AddSub_Flag.vhd.
- Editor:** Displays the VHDL code for ZHANG_SEPTMBER19_FullAdder.vhd. The code defines a full adder using two half adders and a 4-bit adder package.
- IP Catalog:** Shows installed IP components and a search bar for partner IP.
- Tasks:** A table showing the compilation tasks:

| Task | Time |
|--|----------|
| Compile Design | 00:01:00 |
| Analysis & Synthesis | 00:00:09 |
| Fitter (Place & Route) | 00:00:40 |
| Assembler (Generate programming files) | 00:00:07 |
| Timing Analysis | 00:00:04 |
| EDA Netlist Writer | |
| Edit Settings | |
| Program Device (Open Programmer) | |
- Messages:** A table showing compilation messages:

| Type | ID | Message |
|--------|--|---------|
| 332140 | No Recovery paths to report | |
| 332140 | No Removal paths to report | |
| 332140 | No Minimum Pulse Width paths to report | |
| 332102 | Design is not fully constrained for setup requirements | |
| 332102 | Design is not fully constrained for hold requirements | |
| > | quartus Prime Timing Analyzer was successful. 0 errors, 6 warnings | |
| 293000 | quartus Prime Full Compilation was successful. 0 errors, 14 warnings | |

Full 1-bit Adder VHDL code using two half adders in a structural model

Chue Zhang , Task 2: Full Adder



Full Adder simulated with input values of '1' and '1' and a carry in value of '0'. Sum is '0' with a carry out of '1' which is correct

Chue Zhang , Task 3: 4-bit Full Adder

The screenshot shows the Quartus Prime Lite Edition interface with the following details:

- Title Bar:** Quartus Prime Lite Edition - C:/Users/czhan/Desktop/Schoolwork/Gertner/Zhang_CS343_FA21/ZHANG_SEPTMBER19_ADDERS/ZHANG_SEPTMBER19_ADDERS - ZHANG_SEPTMBER19_ADDERS
- File Menu:** File Edit View Project Assignments Processing Tools Window Help
- Project Navigator:** Shows files including ZHANG_SEPTMBER19_LPM_ADDSUB.qip, ZHANG_SEPTMBER19_LPM_ADDSUB.vhd, ZHANG_SEPTMBER19_N_Bit_AddSub.vhd, ZHANG_SEPTMBER19_HalfAdder.vhd, ZHANG_SEPTMBER19_fulladd_package.vhd, ZHANG_SEPTMBER19_FullAdder.vhd, ZHANG_SEPTMBER19_Four_Bit_AddSub.vhd, ZHANG_SEPTMBER19_Four_Bit_Adder.vhd, and ZHANG_SEPTMBER19_N_Bit_AddSub_Flag.vhd.
- Editor:** Displays the VHDL code for ZHANG_SEPTMBER19_Four_Bit_Adder.vhd. The code defines a 4-bit full adder using four 1-bit full adders (FA1 to FA4) connected in a chain. The code includes declarations for libraries, entity, architecture, components, signals, and the main process body.
- IP Catalog:** Shows the installed IP catalog with sections for Project Directory, Library, and Search for Partner IP.
- Tasks:** A table showing compilation tasks:

| Task | Time |
|--|----------|
| Compile Design | 00:01:00 |
| Analysis & Synthesis | 00:00:09 |
| Fitter (Place & Route) | 00:00:40 |
| Assembler (Generate programming files) | 00:00:07 |
| Timing Analysis | 00:00:04 |
| EDA Netlist Writer | |
| Edit Settings | |
| Program Device (Open Programmer) | |
- Messages:** A table showing compilation messages:

| Type | ID | Message |
|--------|--|---------|
| 332140 | NO Recovery paths to report | |
| 332140 | NO Removal paths to report | |
| 332140 | No Minimum pulse width paths to report | |
| 332102 | Design is not fully constrained for setup requirements | |
| 332102 | Design is not fully constrained for hold requirements | |
| > | quartus Prime Timing Analyzer was successful. 0 errors, 6 warnings | |
| 293000 | Quartus Prime Full Compilation was successful. 0 errors, 14 warnings | |

Figure above showcases a Full 4-bit Adder coded in VHDL. The process goes as follows, the first 1-bit Full Adder processes the initial values which then has a carry out value and a sum which is inputted to the next Full Adder and is then processed.

Chue Zhang , Task 3: 4-bit Full Adder

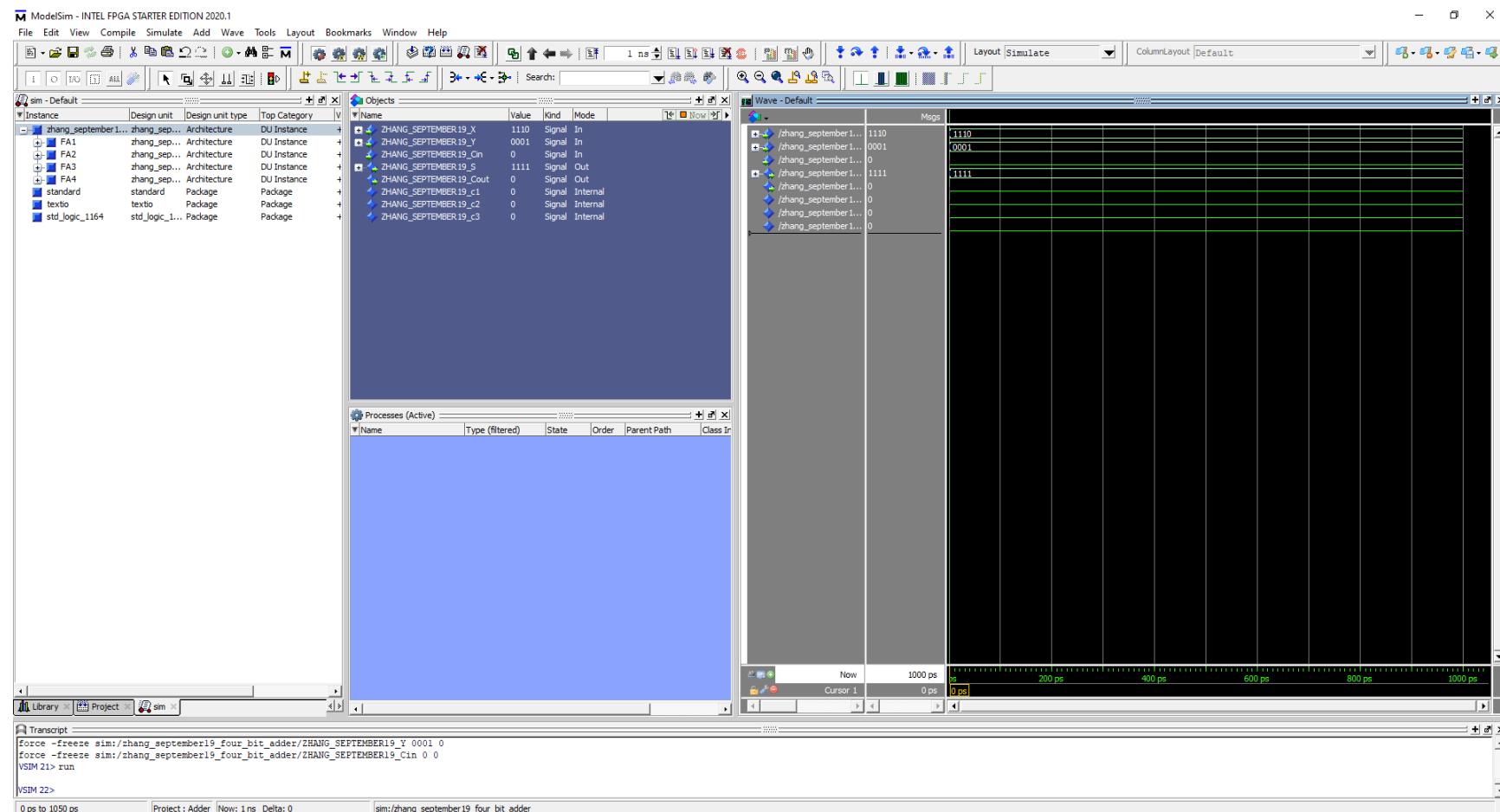


Figure above showcases Full 4-bit Adder simulation using ModelSim with input values of '1101' and '0001' with '0' as cin. The sum output for these two input values are 1111 with a carry out of 0 which is correct.

Chue Zhang , Task 4 : Full 4-bit Adder/Sub

The screenshot shows the Quartus Prime Lite Edition interface with the following details:

- Title Bar:** Quartus Prime Lite Edition - C:/Users/czhan/Desktop/Schoolwork/Gertner/Zhang_CS343_FA21/ZHANG_SEPTMBER19_ADDERS/ZHANG_SEPTMBER19_ADDERS - ZHANG_SEPTMBER19_ADDERS
- File Menu:** File Edit View Project Assignments Processing Tools Window Help
- Project Navigator:** Shows files including ZHANG_SEPTMBER19_LPM_ADDSUB.qip, ZHANG_SEPTMBER19_LPM_ADDSUB.vhd, ZHANG_SEPTMBER19_N_Bit_AddSub.vhd, ZHANG_SEPTMBER19_HalfAdder.vhd, ZHANG_SEPTMBER19_fulladd_package.vhd, ZHANG_SEPTMBER19_FullAdder.vhd, ZHANG_SEPTMBER19_Four_Bit_AddSub.vhd, ZHANG_SEPTMBER19_Four_Bit_Adder.vhd, and ZHANG_SEPTMBER19_N_Bit_AddSub_Flag.vhd.
- Editor:** Displays the VHDL code for ZHANG_SEPTMBER19_Four_Bit_AddSub.vhd. The code defines a full 4-bit adder/subtractor with an extra OP input. It includes components for a four-bit adder, four full adders (FA1-FA4), and various signals for inputs X, Y, OP, Cin, and outputs S, Cout, and bxor.
- Tasks:** A table showing the compilation status of various tasks:

| Task | Time |
|--|----------|
| Compile Design | 00:01:00 |
| Analysis & Synthesis | 00:00:09 |
| Fitter (Place & Route) | 00:00:40 |
| Assembler (Generate programming files) | 00:00:07 |
| Timing Analysis | 00:00:04 |
| EDA Netlist Writer | |
| Edit Settings | |
| Program Device (Open Programmer) | |
- Messages:** A table showing messages from the system:

| Type | ID | Message |
|------|--------|--|
| 1 | 332140 | No Recovery paths to report |
| 1 | 332140 | No Removal paths to report |
| 1 | 332140 | No Minimum Pulse width paths to report |
- IP Catalog:** Shows the installed IP catalog with sections for Project Directory, Library, and Search for Partner IP.

Figure above showcases the VHDL code of a Full 4-bit Adder/Subtractor which is similar to the 4-bit Adder but has an extra input variable that helps decide whether the machine should add or subtract. If OP = '1' then we subtract if OP = '0' then we add.

Chue Zhang , Task 4 : Full 4-bit Adder/Sub

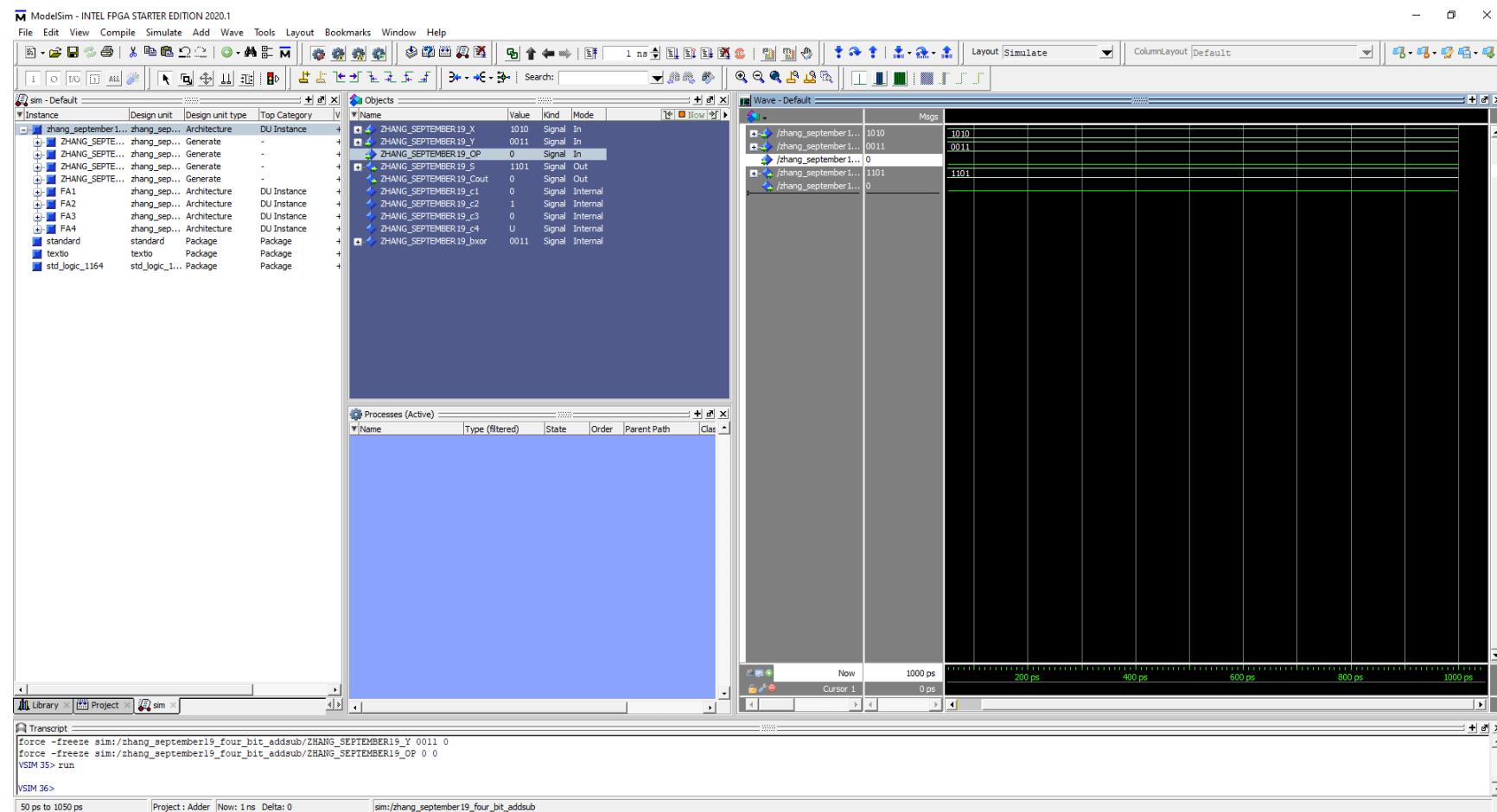


Figure above showcases OP = '0' which means we add the two input values of '1010' and '0011' which gives a sum output of 1101 with a carry out of 0, which is correct.

Chue Zhang , Task 4 : Full 4-bit Adder/Sub

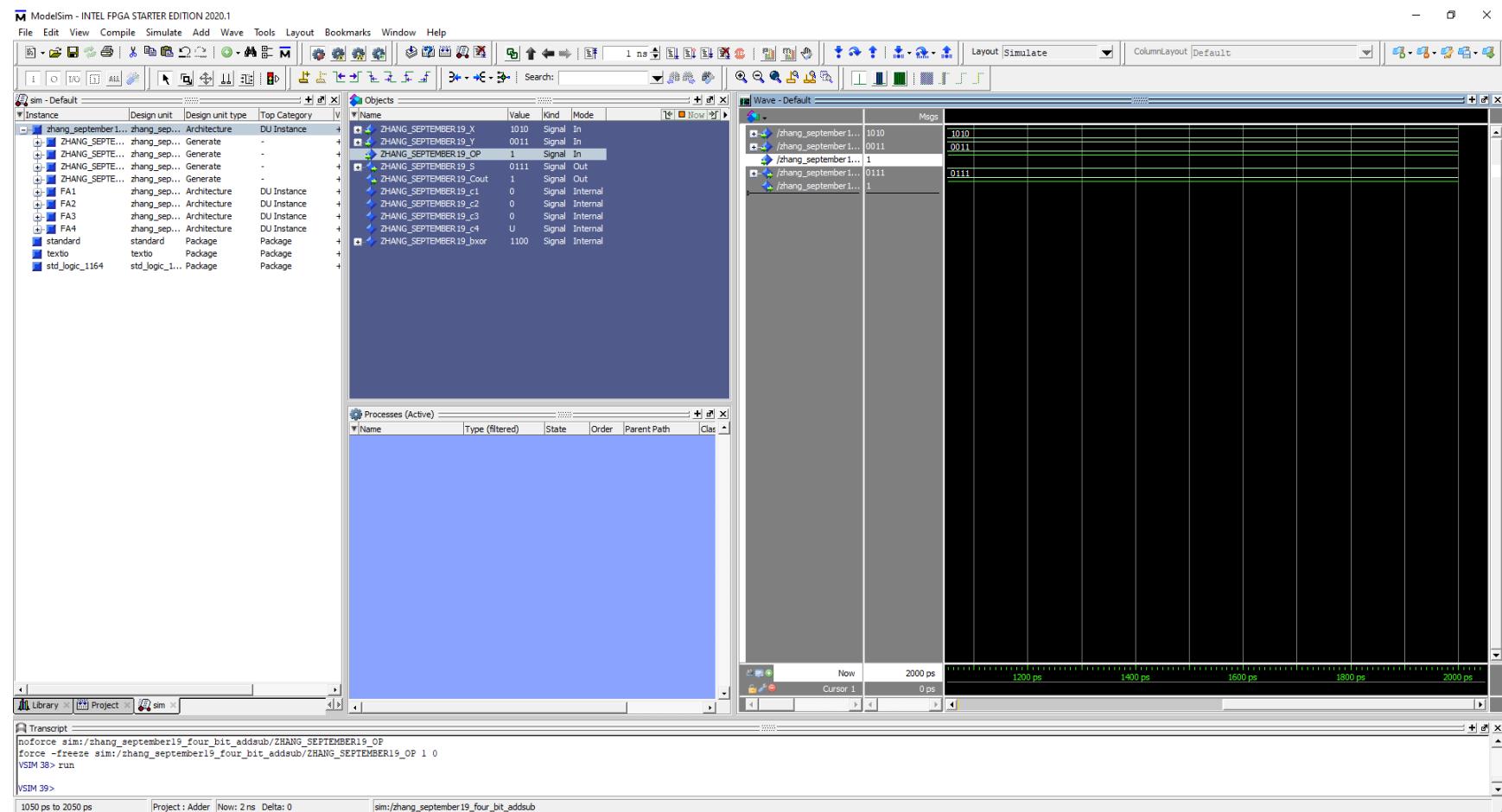


Figure above showcases OP = '1' which means we subtract the two values '1010' and '0011' which gives a sum output of '0111' which is correct

Chue Zhang , Task 5 : Full 4-bit Adder Package

```

1  library IEEE;
2  use ieee.std_logic_1164.all;
3
4  package ZHANG_SEPTMBER19_fulladd_package is
5      component ZHANG_SEPTMBER19_HalfAdder
6          port (ZHANG_SEPTMBER19_X, ZHANG_SEPTMBER19_Y : in std_logic;
7                ZHANG_SEPTMBER19_S, ZHANG_SEPTMBER19_C : out std_logic);
8      end component;
9
10     component ZHANG_SEPTMBER19_FullAdder is
11         port (ZHANG_SEPTMBER19_A, ZHANG_SEPTMBER19_B, ZHANG_SEPTMBER19_Cin : in std_logic;
12               ZHANG_SEPTMBER19_Sum, ZHANG_SEPTMBER19_Cout : out std_logic);
13     end component;
14
15     component ZHANG_SEPTMBER19_Four_Bit_Adder is
16         port (ZHANG_SEPTMBER19_A, ZHANG_SEPTMBER19_B, ZHANG_SEPTMBER19_Cin : in std_logic_vector(3 downto 0);
17               ZHANG_SEPTMBER19_Cout : out std_logic;
18               ZHANG_SEPTMBER19_S : out std_logic_vector(3 downto 0);
19               ZHANG_SEPTMBER19_Cout : out std_logic);
20     end component;
21
22     component ZHANG_SEPTMBER19_Four_Bit_Addsub is
23         port (ZHANG_SEPTMBER19_X, ZHANG_SEPTMBER19_Y : in std_logic_vector(3 downto 0);
24               ZHANG_SEPTMBER19_OP : in std_logic;
25               ZHANG_SEPTMBER19_S : out std_logic_vector(3 downto 0);
26               ZHANG_SEPTMBER19_Cout : out std_logic);
27     end component;
28
29 end ZHANG_SEPTMBER19_fulladd_package;

```

Tasks

| Task | Time |
|--|----------|
| Compile Design | 00:01:00 |
| Analysis & Synthesis | 00:00:09 |
| Fitter (Place & Route) | 00:00:40 |
| Assembler (Generate programming files) | 00:00:07 |
| Timing Analysis | 00:00:04 |
| EDA Netlist Writer | |
| Edit Settings | |
| Program Device (Open Programmer) | |

Messages

| Type | ID | Message |
|--------|--|---------|
| 332140 | No Recovery paths to report | |
| 332140 | No Removal paths to report | |
| 332140 | No Minimum Pulse width paths to report | |

Full Adder Package which houses the 4-bit Adder/Subtractor, 4-bit Adder, 1-bit Full Adder and the Half Adder. Further note is that package CANNOT be compiled as it has no entity but can still perform its function as long as it exists within the files.

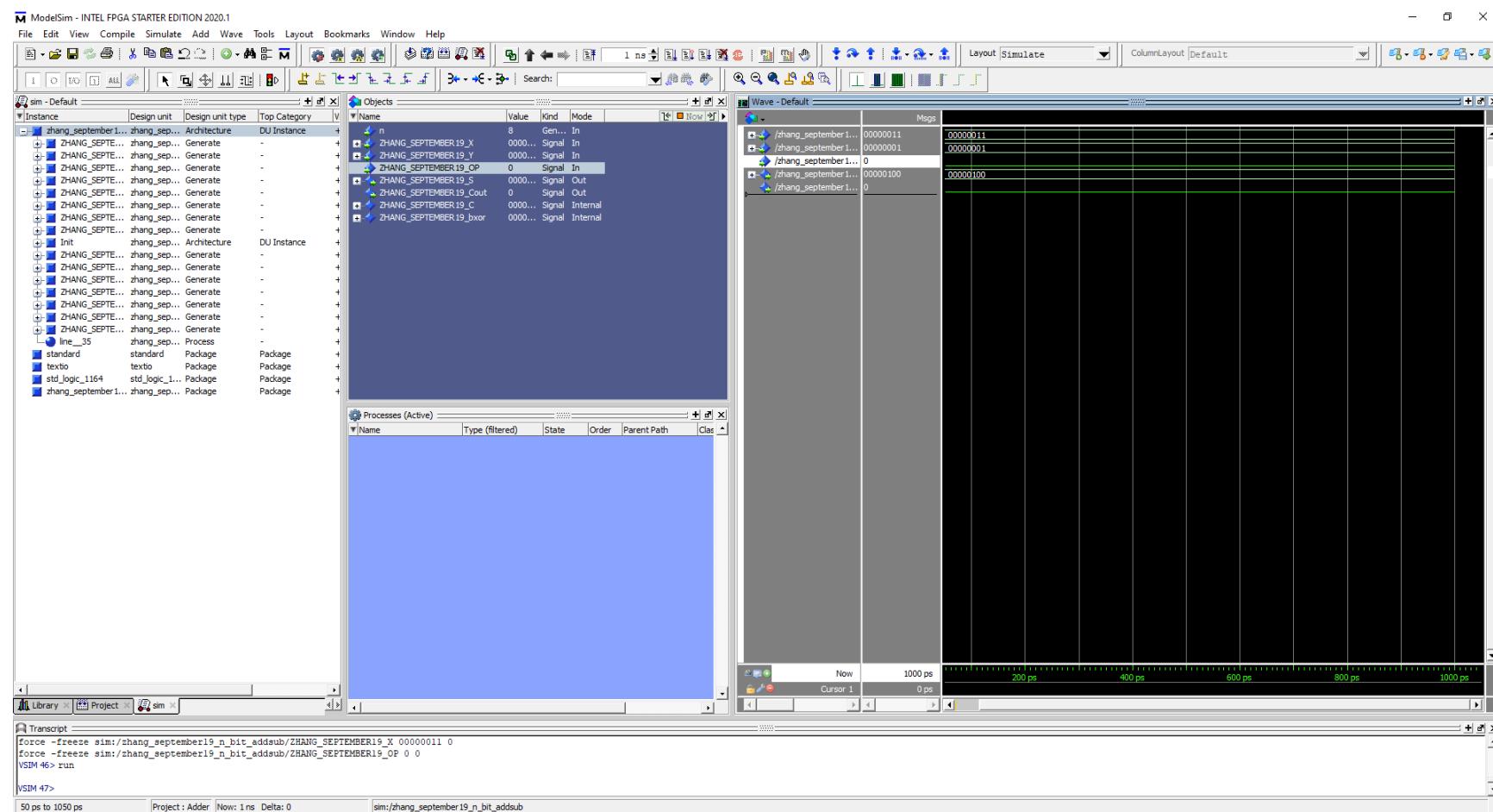
Chue Zhang , Task 6 : Full N-bit Adder/Sub

The screenshot shows the Quartus Prime Lite Edition interface with the following details:

- Project Navigator:** Shows files including ZHANG_SEPTMBER19_LPM_ADDSUB.qip, ZHANG_SEPTMBER19_LPM_ADDSUB.vhd, ZHANG_SEPTMBER19_N_Bit_AddSub.vhd, ZHANG_SEPTMBER19_HalfAdder.vhd, ZHANG_SEPTMBER19_fulladd_package.vhd, ZHANG_SEPTMBER19_FullAdder.vhd, ZHANG_SEPTMBER19_Four_Bit_AddSub.vhd, ZHANG_SEPTMBER19_Four_Bit_Adder.vhd, and ZHANG_SEPTMBER19_N_Bit_AddSub_Flag.vhd.
- Editor:** Displays the VHDL code for ZHANG_SEPTMBER19_N_Bit_AddSub.vhd. The code defines an entity ZHANG_SEPTMBER19_N_Bit_AddSub with a generic input 'n' (set to 8) and ports for X, OP, Y, and Cout. It uses a process to calculate the sum and carry bits through bit-level operations (XOR and AND).
- Tasks:** Shows a list of completed tasks under 'Compile Design', 'Analysis & Synthesis', 'Analysis & Elaboration', 'Partition Merge', 'Netlist Viewers', and 'Design Assistant (Post-Mapping)'.
- Messages:** Displays a list of messages with IDs and descriptions, including:
 - 332140 No Removal paths to report
 - 332140 No Minimum Pulse width paths to report
 - 332102 Design is not fully constrained for setup requirements
 - 332102 Design is not fully constrained for hold requirements
 - Quartus Prime Timing Analyzer was successful. 0 errors, 6 warnings
 - 293000 Quartus Prime Full Compilation was successful. 0 errors, 14 warnings

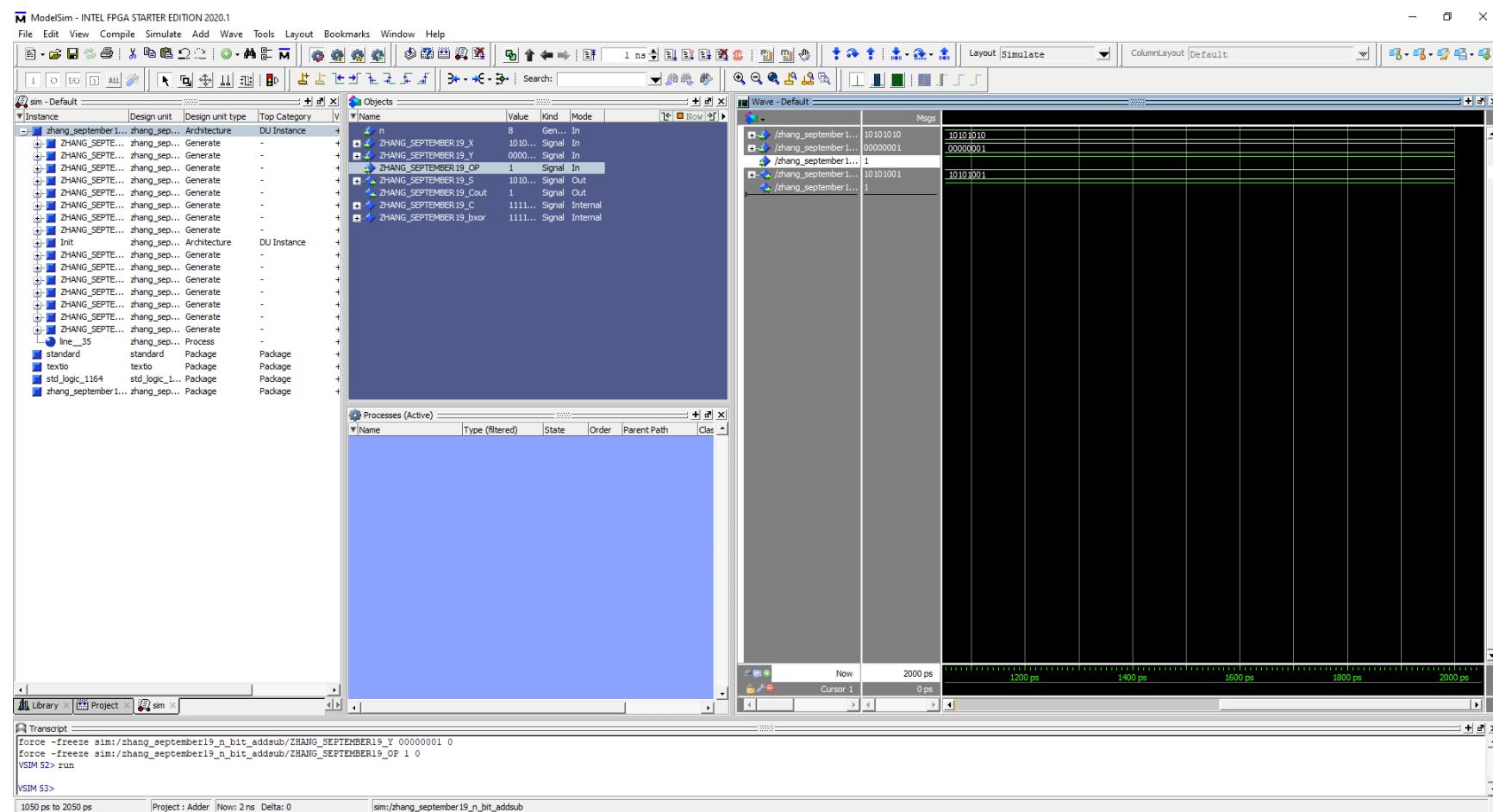
Figure above showcases the VHDL code for N-bit Adder/Subtractor which uses a generic input 'n' which we can set to any value. What happens next is the n value helps set the bit count of appropriate variables. Here we are using n = 8.

Chue Zhang , Task 6 : Full N-bit Adder/Sub



The figure above showcases the N-bit Adder/Subtractor with an N value of 8. We use input values of '00000011' and '00000001' with an OP value of '0' so we add the two inputs together. The sum output is '00000100' with no carry out which is correct.

Chue Zhang , Task 6 : Full N-bit Adder/Sub



The figure above showcases the N-bit Adder/Subtractor with an N value of 8. We use input values of '10101010' and '00000001' with an OP value of '1' so we subtract the two inputs together. The sum output is '10101001' with no carry out which is correct.

Chue Zhang , Task 7 : Full N-bit Adder/Sub with Flags

Quartus Prime Lite Edition - C:/Users/czhan/Desktop/Schoolwork/Gertner/Zhang_CS343_FA21/ZHANG_SEPTMBER19_ADDERS/ZHANG_SEPTMBER19_ADDERS - ZHANG_SEPTMBER19_ADDERS

File Edit View Project Assignments Processing Tools Window Help

Search altera.com

Project Navigator

- ZHANG_SEPTMBER19_ADDERS
- Files
 - ZHANG_SEPTMBER19_LPM_ADDSUB.qip
 - ZHANG_SEPTMBER19_LPM_ADDSUB.vhd
 - ZHANG_SEPTMBER19_N_Bit_AddSub.vhd
 - ZHANG_SEPTMBER19_HalfAdder.vhd
 - ZHANG_SEPTMBER19_fulladd_package.vhd
 - ZHANG_SEPTMBER19_FullAdder.vhd
 - ZHANG_SEPTMBER19_Four_Bit_AddSub.vhd
 - ZHANG_SEPTMBER19_Four_Bit_Adder.vhd
 - ZHANG_SEPTMBER19_N_Bit_AddSub_Flag.vhd
 - delete_me.qip
 - delete_me.vhd

ZHANG_SEPTMBER19_N_Bit_AddSub_Flag.vhd

```

1 library ieee;
2 use ieee.std_logic_1164.all;
3
4 entity ZHANG_SEPTMBER19_N_Bit_AddSub_Flag is
5   generic (n : integer:= 32); -- we can set this to whatever we want to test, for example 32
6   port ( ZHANG_SEPTMBER19_X : in std_logic_vector(n-1 downto 0);
7          ZHANG_SEPTMBER19_Op : in std_logic;
8          ZHANG_SEPTMBER19_Op : out std_logic_vector(n-1 downto 0);
9          ZHANG_SEPTMBER19_Cout : out std_logic;
10         ZHANG_SEPTMBER19_Overflow, ZHANG_SEPTMBER19_Zero, ZHANG_SEPTMBER19_Negative: out std_logic);
11
12 end ZHANG_SEPTMBER19_N_Bit_AddSub_Flag;
13
14 architecture ZHANG_SEPTMBER19_behav of ZHANG_SEPTMBER19_N_Bit_AddSub_Flag is
15
16   signal ZHANG_SEPTMBER19_C, ZHANG_SEPTMBER19_SUM_BUFFER, ZHANG_SEPTMBER19_bxor, ZHANG_SEPTMBER19_zeroes : std_logic_vector(n-1 downto 0);
17
18 begin
19
20   ZHANG_SEPTMBER19_bxor_allocate: for i in 0 to n-1 generate -- OP xor Y, this to determine if we subtracting or not
21     ZHANG_SEPTMBER19_bxor(i) <= ZHANG_SEPTMBER19_Op xor ZHANG_SEPTMBER19_Y(i);
22   end generate;
23
24   ZHANG_SEPTMBER19_zero_allocate: for i in 0 to n-1 generate -- for zero flag, generates 'N' number of 0's
25     ZHANG_SEPTMBER19_zeroes(i) <= '0';
26   end generate;
27
28   process (ZHANG_SEPTMBER19_X, ZHANG_SEPTMBER19_bxor, ZHANG_SEPTMBER19_SUM_BUFFER, ZHANG_SEPTMBER19_Op, ZHANG_SEPTMBER19_C)
29   begin
30
31     ZHANG_SEPTMBER19_SUM_BUFFER(0) <= ZHANG_SEPTMBER19_X(0) xor ZHANG_SEPTMBER19_bxor(0) xor ZHANG_SEPTMBER19_Op;
32     ZHANG_SEPTMBER19_C(0) <= (ZHANG_SEPTMBER19_X(0) and ZHANG_SEPTMBER19_bxor(0)) or ((ZHANG_SEPTMBER19_X(0) xor ZHANG_SEPTMBER19_bxor(0)) and ZHANG_SEPTMBER19_Op);
33
34     for i in 1 to n-1 loop
35       ZHANG_SEPTMBER19_SUM_BUFFER(i) <= ZHANG_SEPTMBER19_X(i) xor ZHANG_SEPTMBER19_bxor(i) xor ZHANG_SEPTMBER19_C(i-1);
36       ZHANG_SEPTMBER19_C(i) <= (ZHANG_SEPTMBER19_X(i) and ZHANG_SEPTMBER19_bxor(i)) or ((ZHANG_SEPTMBER19_X(i) xor ZHANG_SEPTMBER19_bxor(i)) and ZHANG_SEPTMBER19_C(i-1));
37     end loop;
38
39   end process;
40
41   ZHANG_SEPTMBER19_Cout <- ZHANG_SEPTMBER19_C(n-1);
42   ZHANG_SEPTMBER19_S <- ZHANG_SEPTMBER19_SUM_BUFFER;
43   ZHANG_SEPTMBER19_Overflow <- ZHANG_SEPTMBER19_C(n-1) xor ZHANG_SEPTMBER19_C(n-2);
44   ZHANG_SEPTMBER19_Zero <- '1' when ZHANG_SEPTMBER19_SUM_BUFFER = ZHANG_SEPTMBER19_zeroes else '0'; -- if sum == only zeroes, 1 else 0
45   ZHANG_SEPTMBER19_Negative <- ZHANG_SEPTMBER19_SUM_BUFFER(n-1); -- left most bit determines if it is negative or not
46
47 end ZHANG_SEPTMBER19_behav;

```

Tasks

- Task
- Compile Design
- Analysis & Synthesis
 - Edit Settings
 - View Report
- Analysis & Elaboration
- Partition Merge
- Netlist Viewers
- Design Assistant (Post-Mapping)

Messages

| Type | ID | Message |
|--------|--|--|
| 332140 | No Removal paths to report | |
| 332140 | No Minimum Pulse width paths to report | |
| 332102 | Design is not fully constrained for setup requirements | |
| 332102 | Design is not fully constrained for hold requirements | |
| > | 332102 | quartus Prime Timing Analyzer was successful. 0 errors, 6 warnings |
| > | 293000 | quartus Prime Full Compilation was successful. 0 errors, 14 warnings |

System (1) Processing (138)

100% 00:00:54

VHDL code for an N-bit Adder/Subtractor with Overflow, Negative and Zero flags.

Chue Zhang , Task 7 : Full N-bit Adder/Sub with Flags

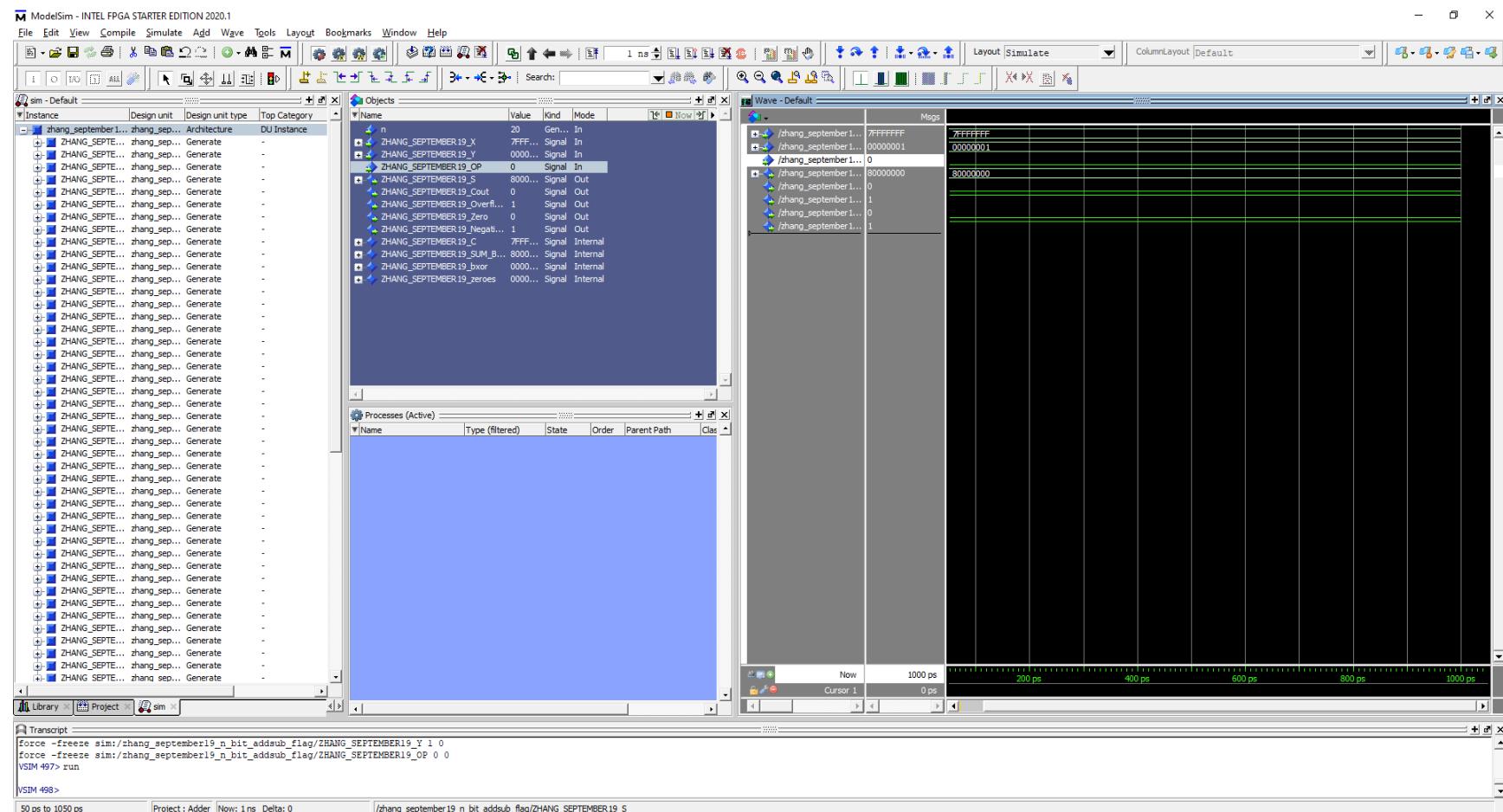


Figure above demonstrates the overflow flag and the negative flag triggering.

Chue Zhang , Task 7 : Full N-bit Adder/Sub with Flags

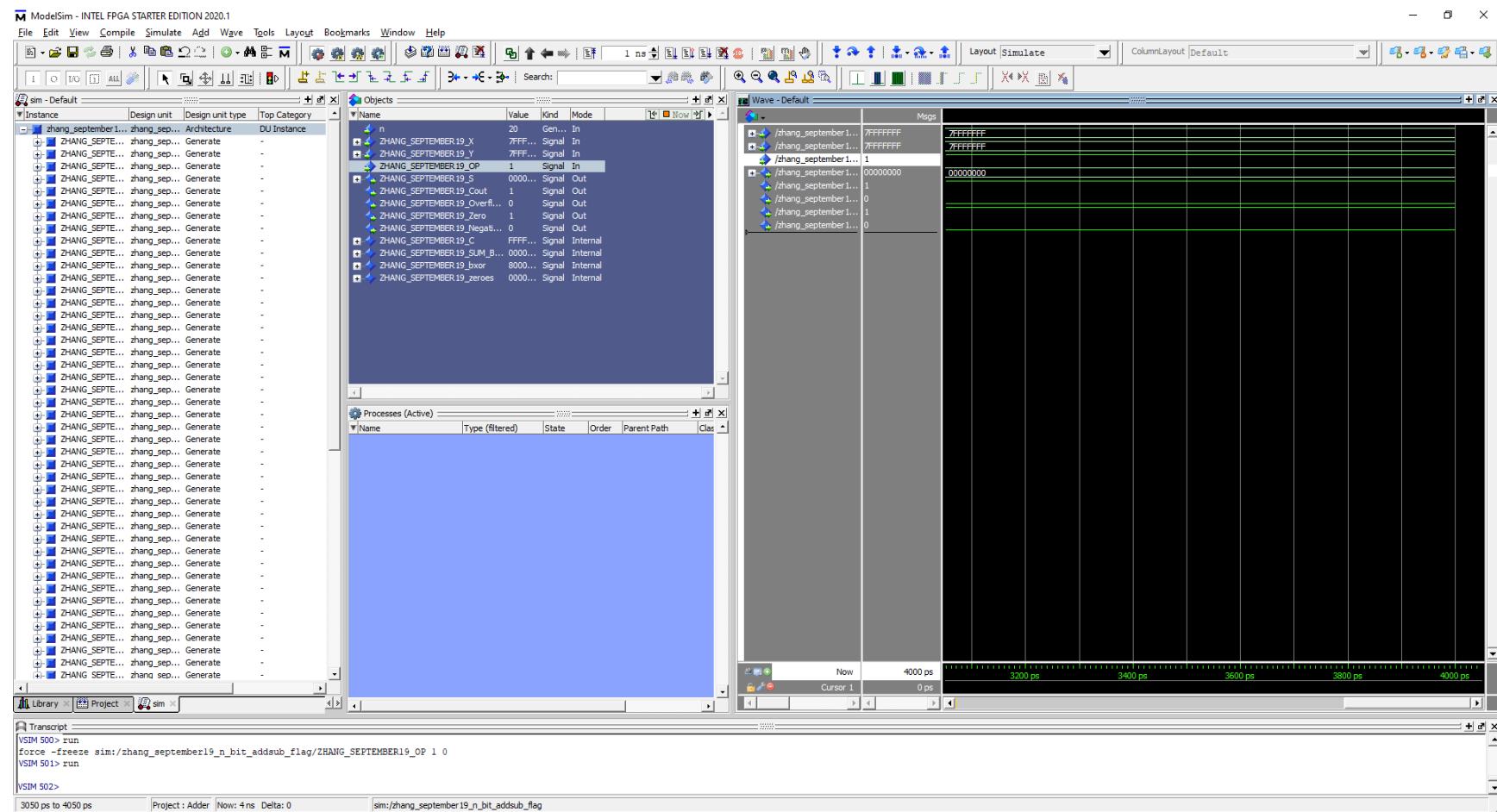
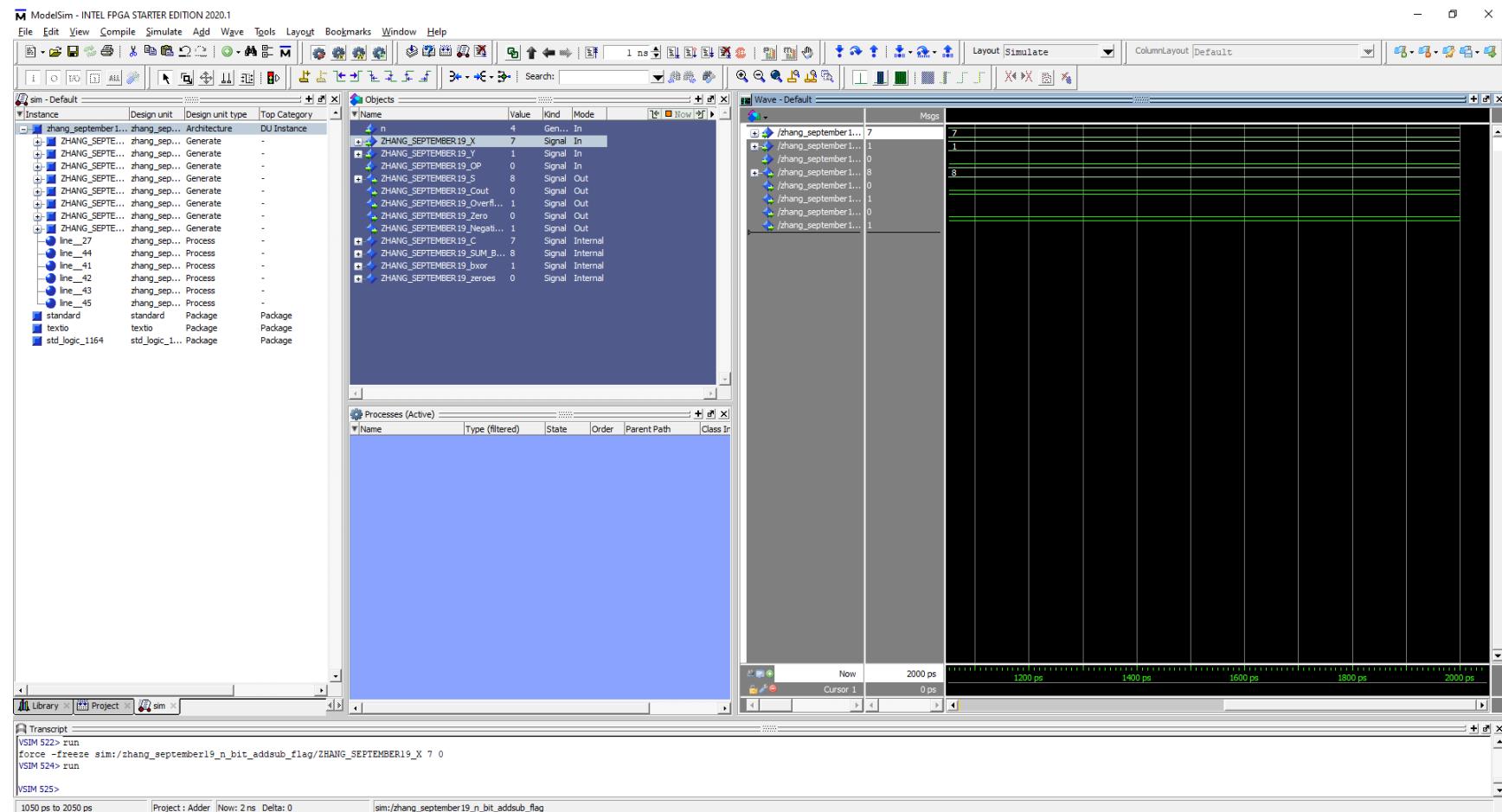


Figure above demonstrates the zero flag being triggered when two values of equal size cancel each other out through subtraction

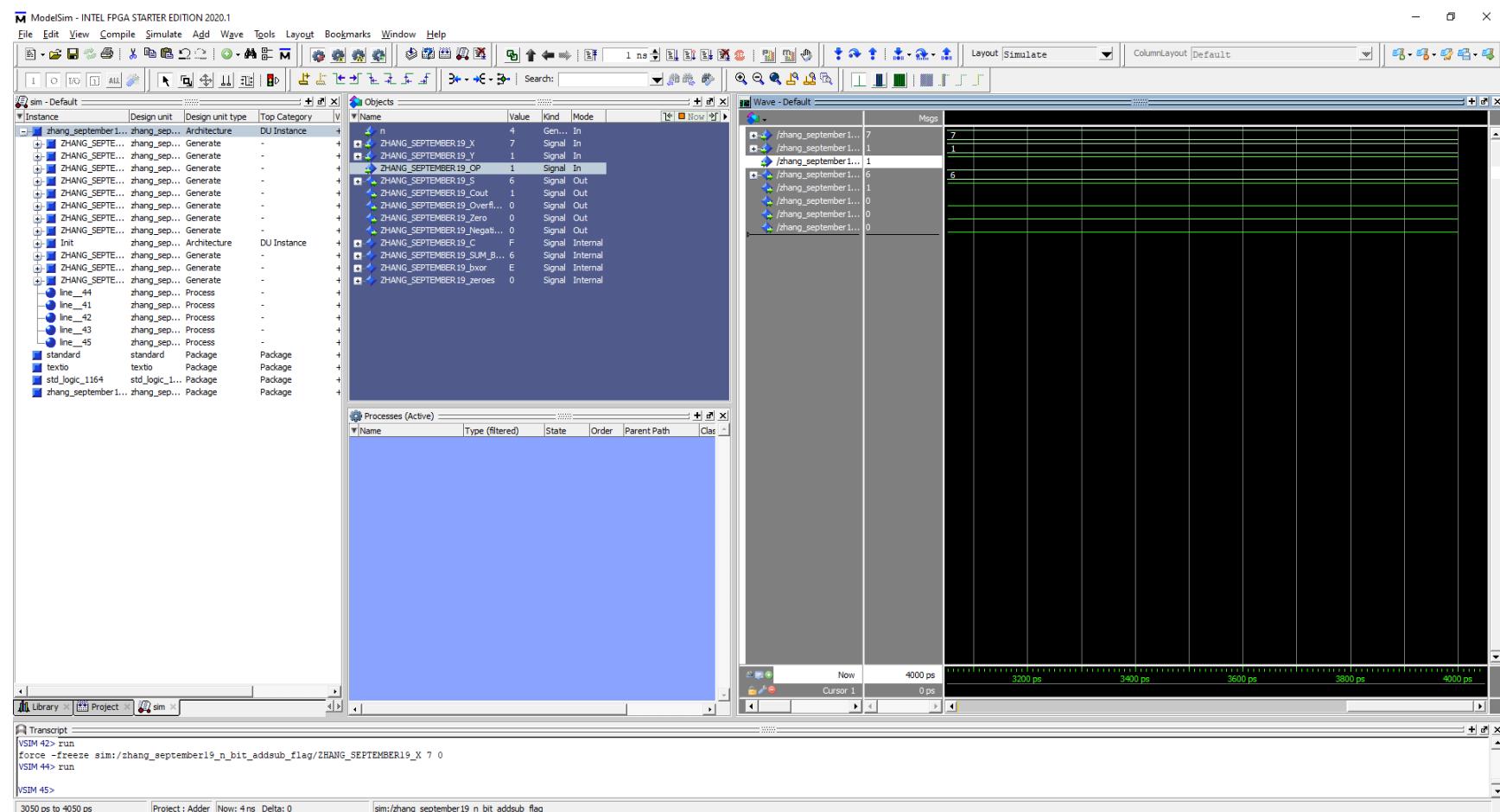
Chue Zhang , Task 8 : Overflow, Zero and Negative Flags verify n=4 and n=32



Task 8a. Most positive N bit + 1, N = 4

Most positive N bit = 7, $7 + 1 = 8$ which ends up becoming a negative value because the most significant bit is now '1'. Overflow has also been detected due to positive + positive = negative

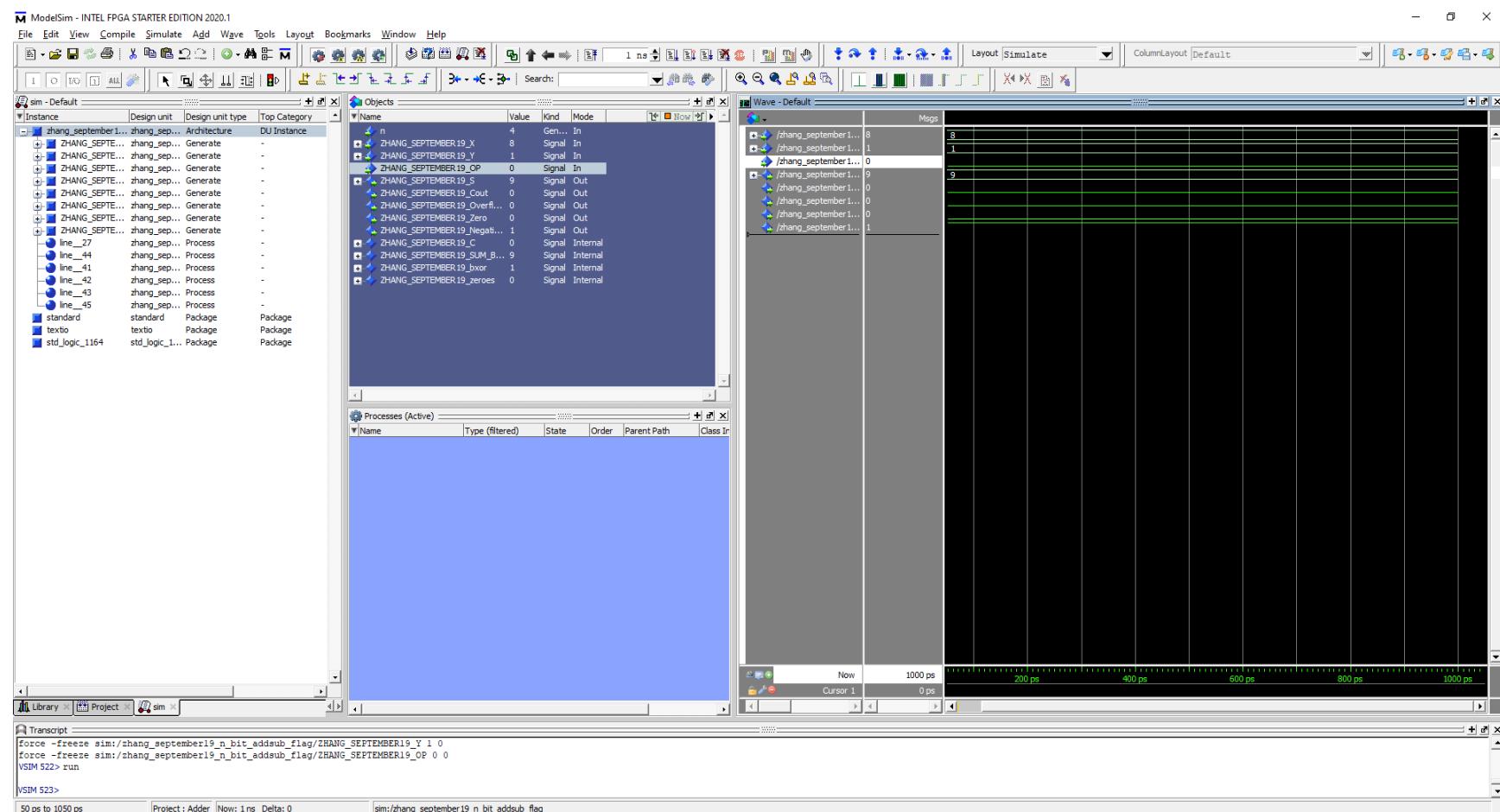
Chue Zhang , Task 8 : Overflow, Zero and Negative Flags verify n=4 and n=32



Task 8b. Most positive N bit - 1, N = 4

$7 - 1 = 6$ which does not trigger any of the flags. 6 is not negative, not zero and does not have overflow because two positives did not become a negative

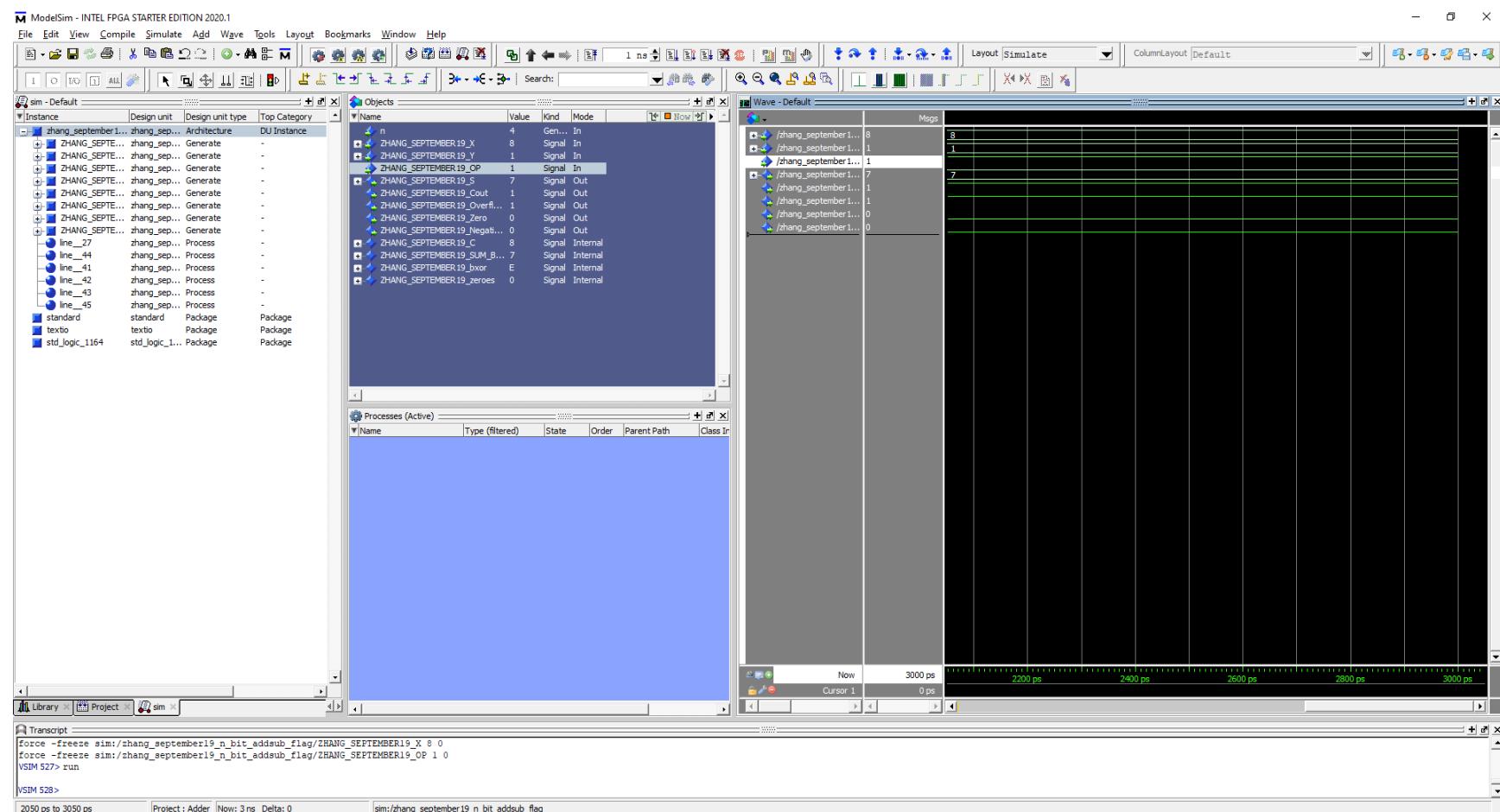
Chue Zhang , Task 8 : Overflow, Zero and Negative Flags verify n=4 and n=32



Task 8c. Most Negative N bit + 1, N = 4

Most negative bit + 1 is 9 which triggers only the negative flag and nothing else

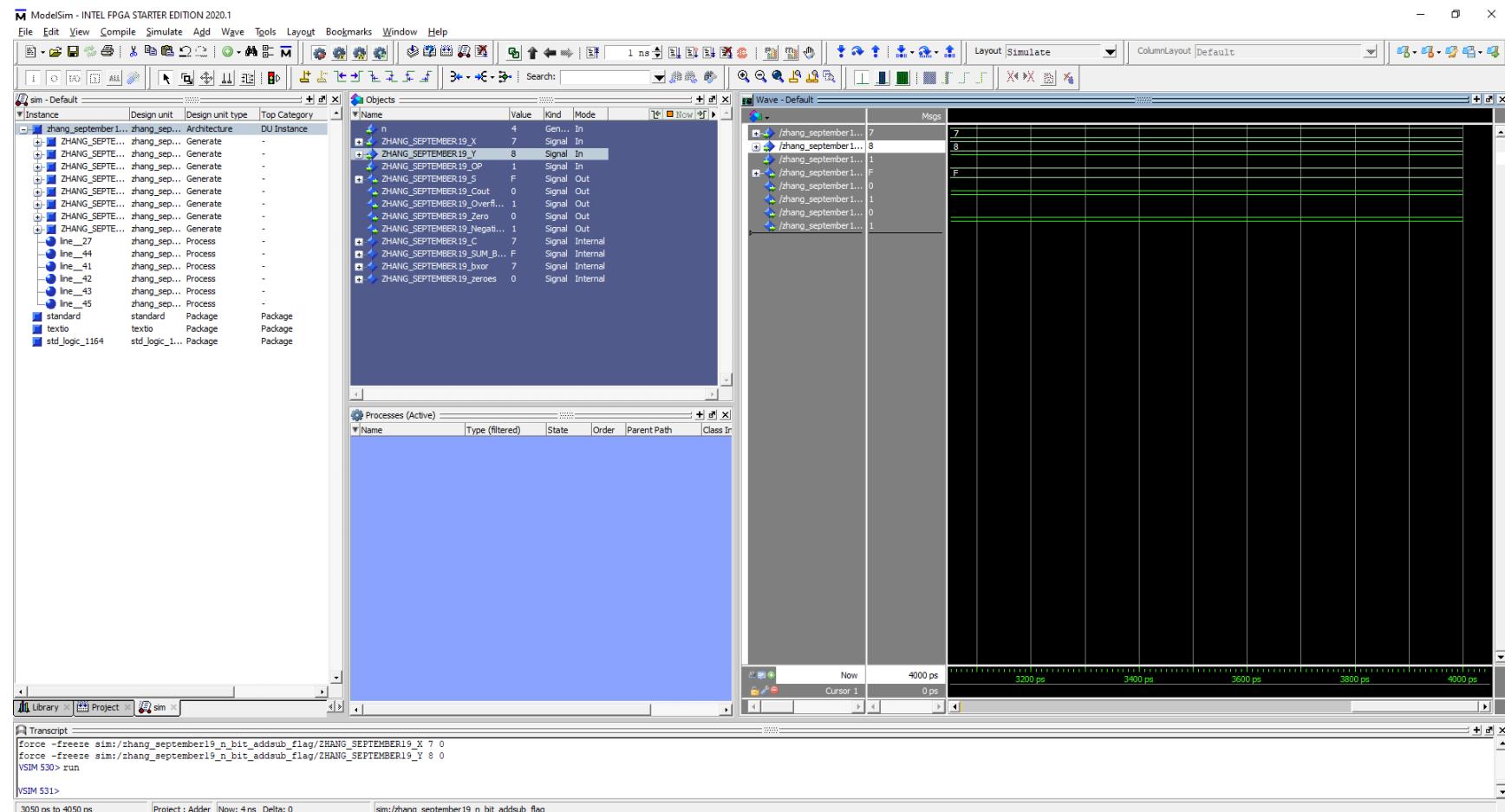
Chue Zhang , Task 8 : Overflow, Zero and Negative Flags verify n=4 and n=32



Task 8d. Most Negative N bit - 1, N = 4

$8-1 = 7$ which is positive but a negative just became positive so the overflow flag triggered

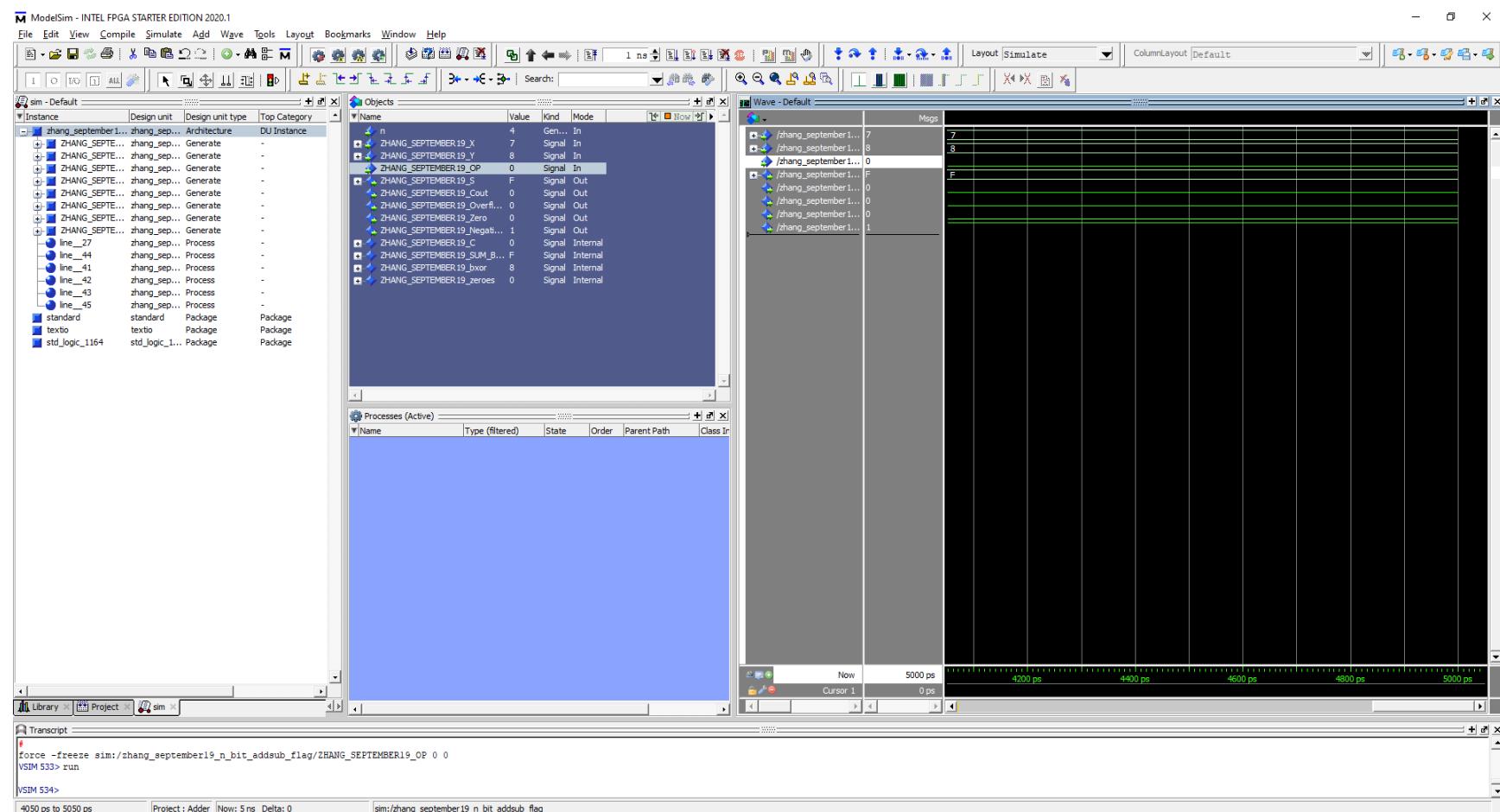
Chue Zhang , Task 8 : Overflow, Zero and Negative Flags verify n=4 and n=32



Task 8e. Most Positive N bit - Most Negative N bit, N = 4

$7 - 8 = -1$ which loops back to F(15). Overflow is triggered because of the state change from negative to positive to negative and negative flag is also triggered because it is negative.

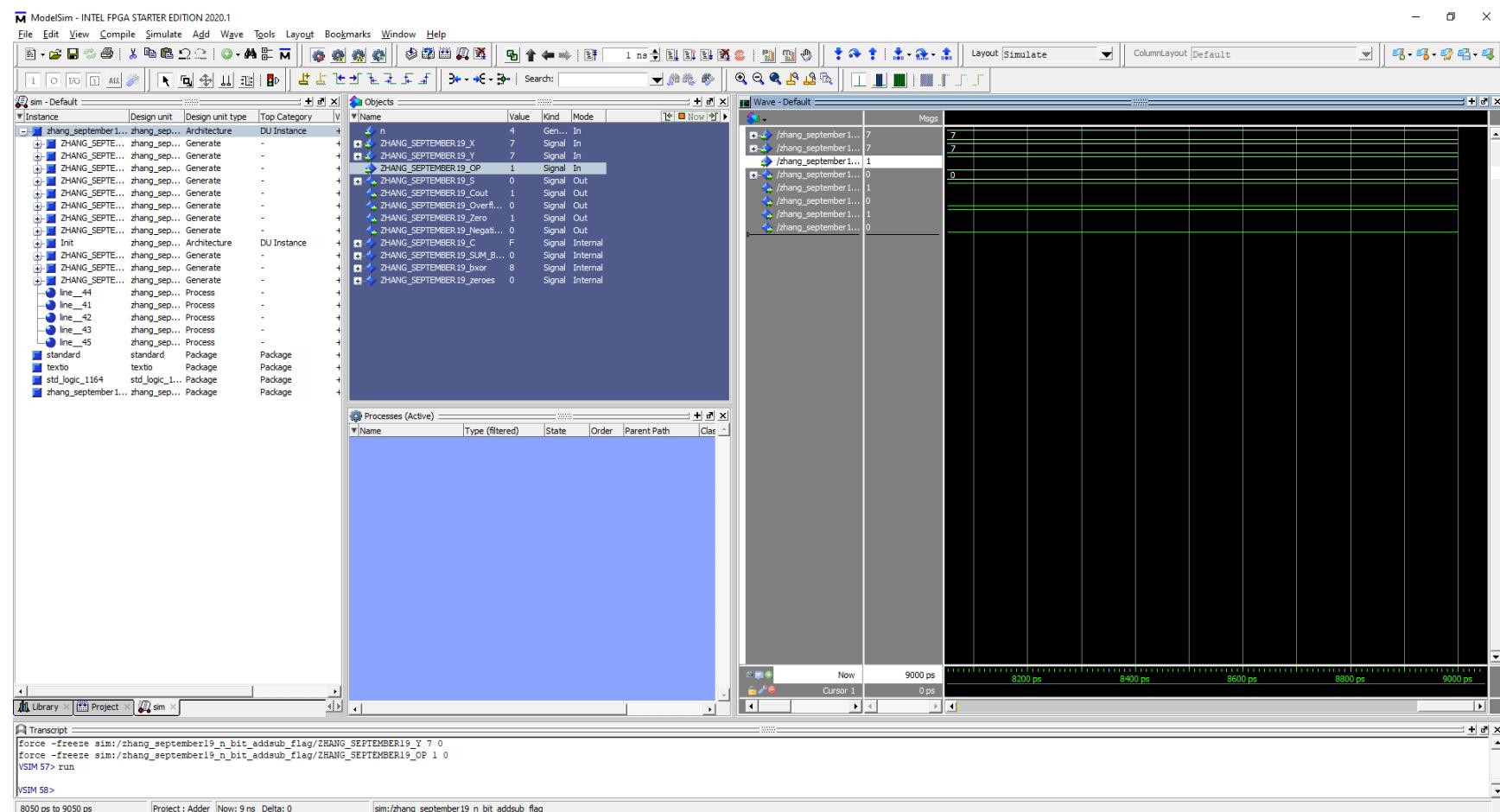
Chue Zhang , Task 8 : Overflow, Zero and Negative Flags verify n=4 and n=32



Task 8f. Most Positive N bit + Most Negative N bit, N = 4

$7 + 8 = 15$ which goes from negative to positive, therefore no overflow flag is triggered but negative flag is still triggered.

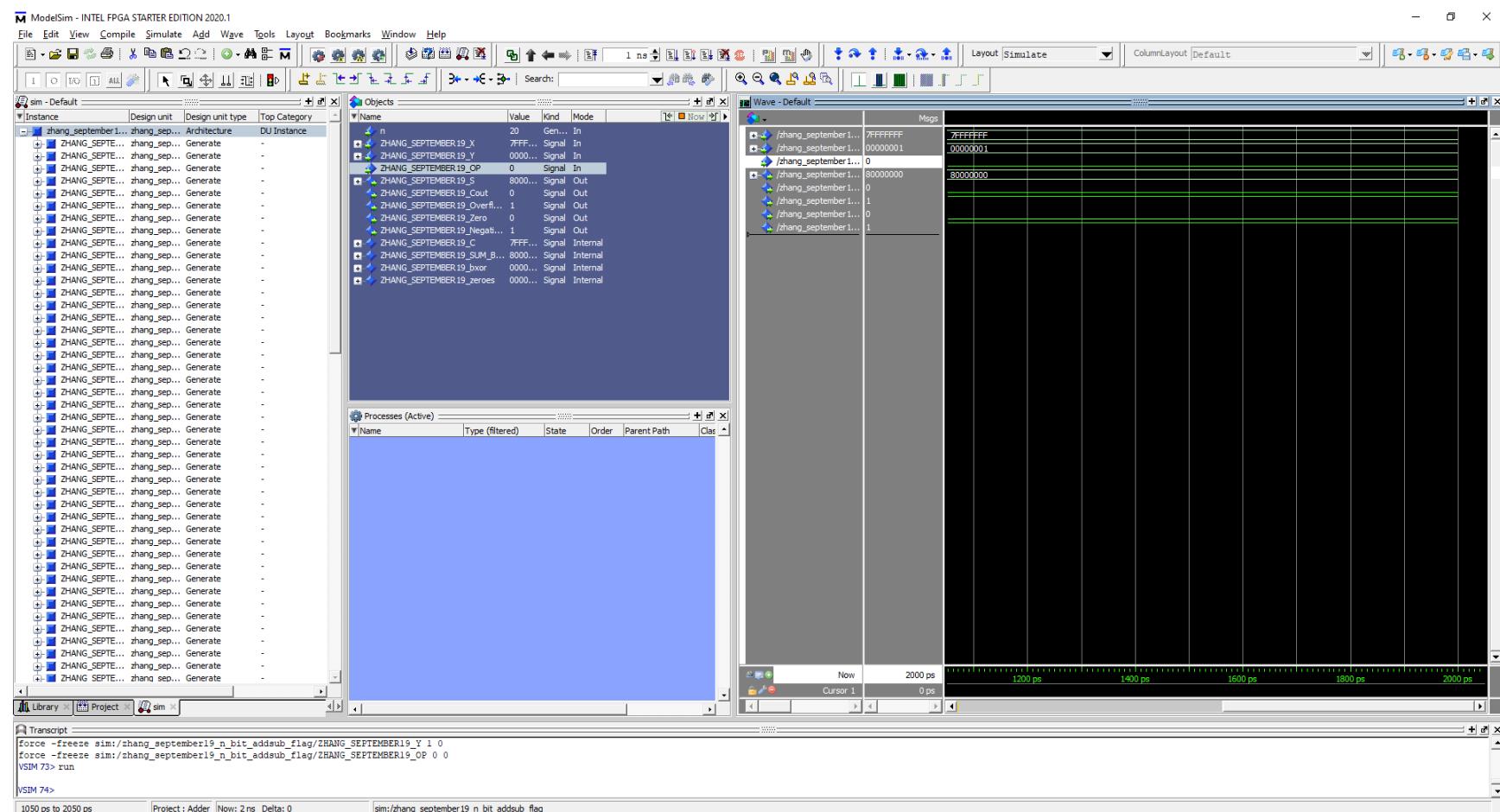
Chue Zhang , Task 8 : Overflow, Zero and Negative Flags verify n=4 and n=32



Task 8g. Most Positive N bit - Most Negative N bit, N = 4

7-7 = 0 which triggers only the zero flag and nothing else.

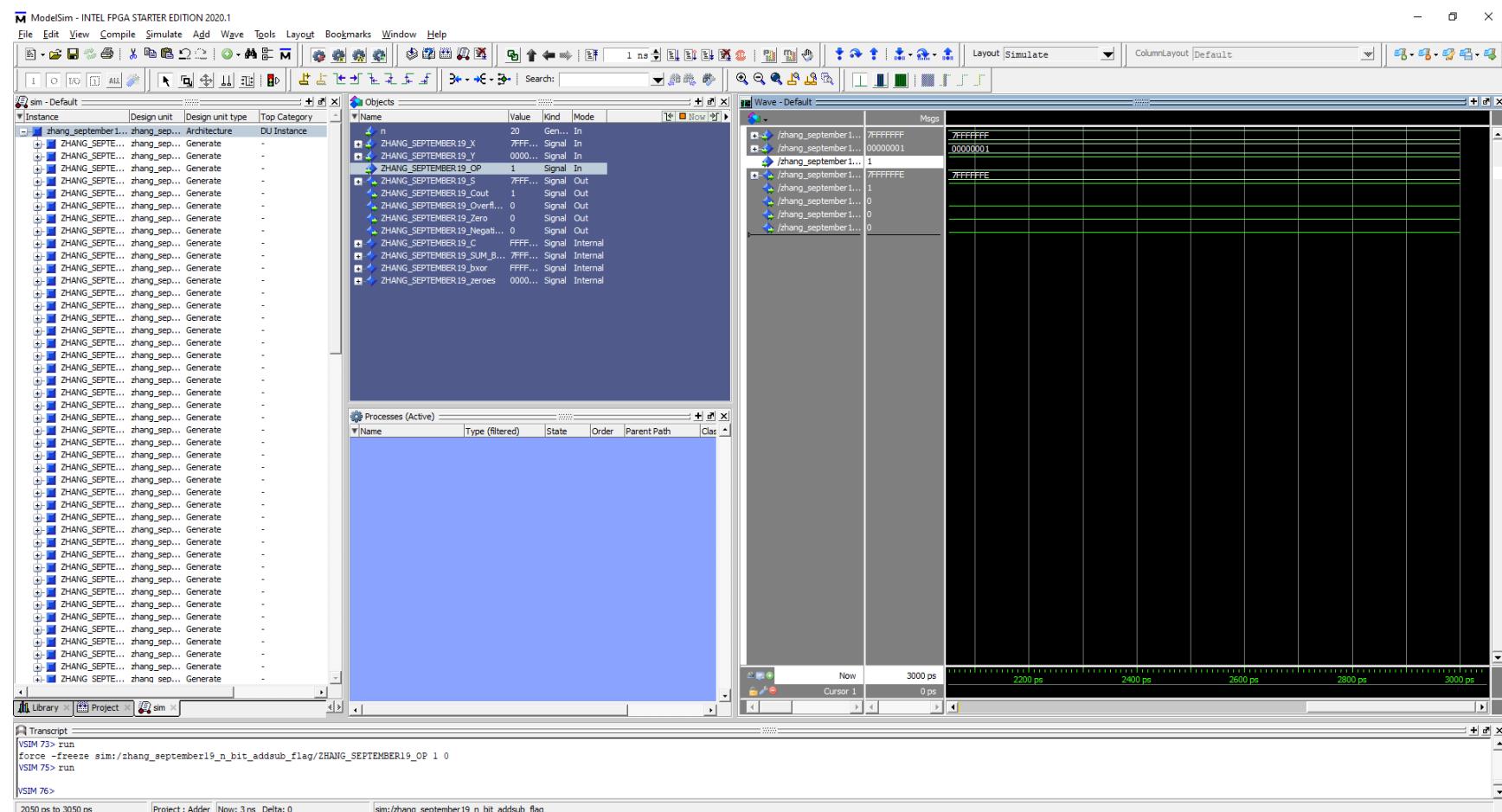
Chue Zhang , Task 8 : Overflow, Zero and Negative Flags verify n=4 and n=32



Task 8a. Most Positive N bit + 1, N = 32

The most positive N bit integer add 1 means that the most significant bit now will have a 1 which triggers negative flag and overflow flag

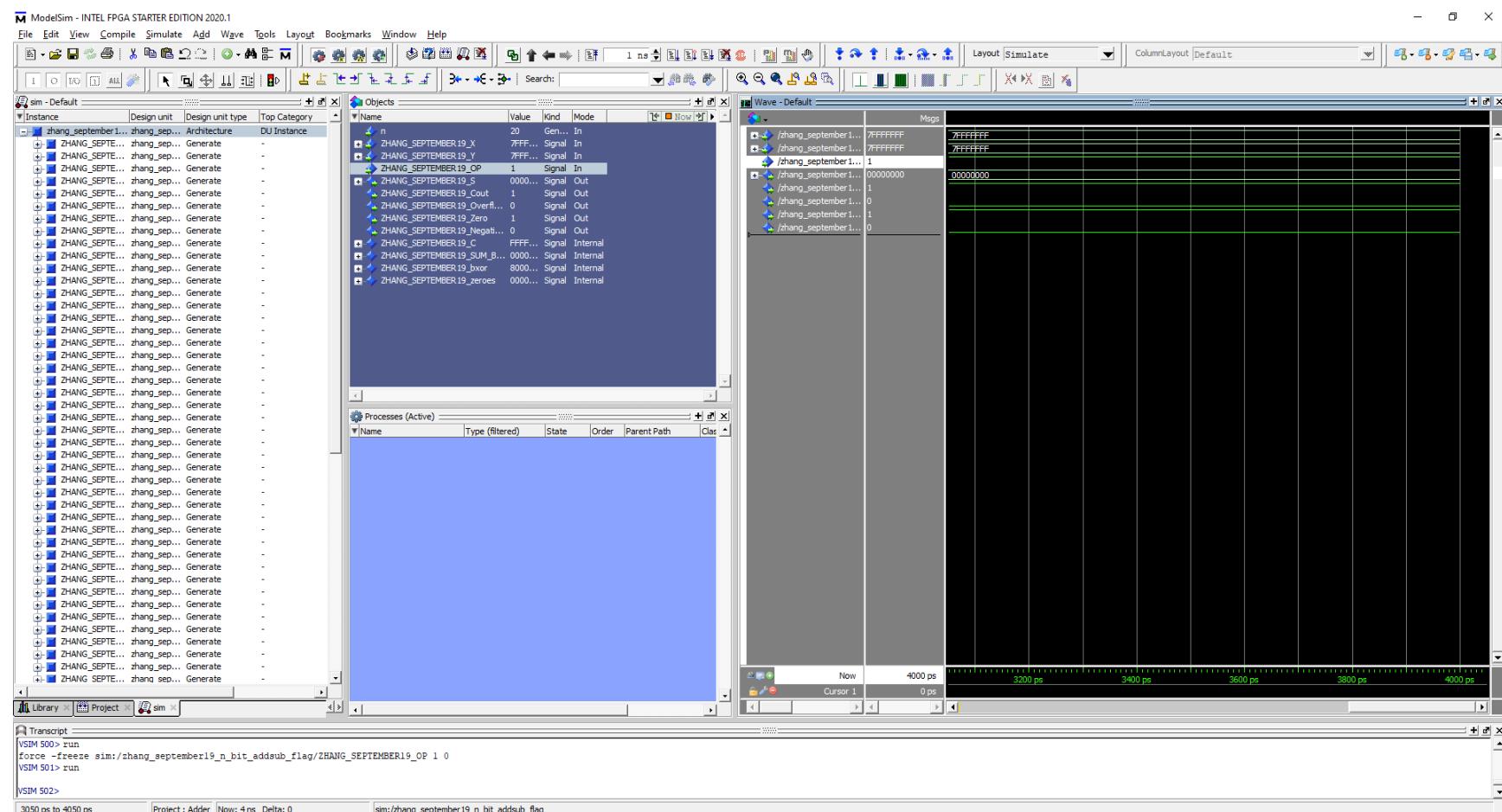
Chue Zhang , Task 8 : Overflow, Zero and Negative Flags verify n=4 and n=32



Task 8b. Most Positive N bit - 1, N = 32

No flags triggered as two positives subtracting each other is positive, there are no zeroes and the most positive – 1 is still positive

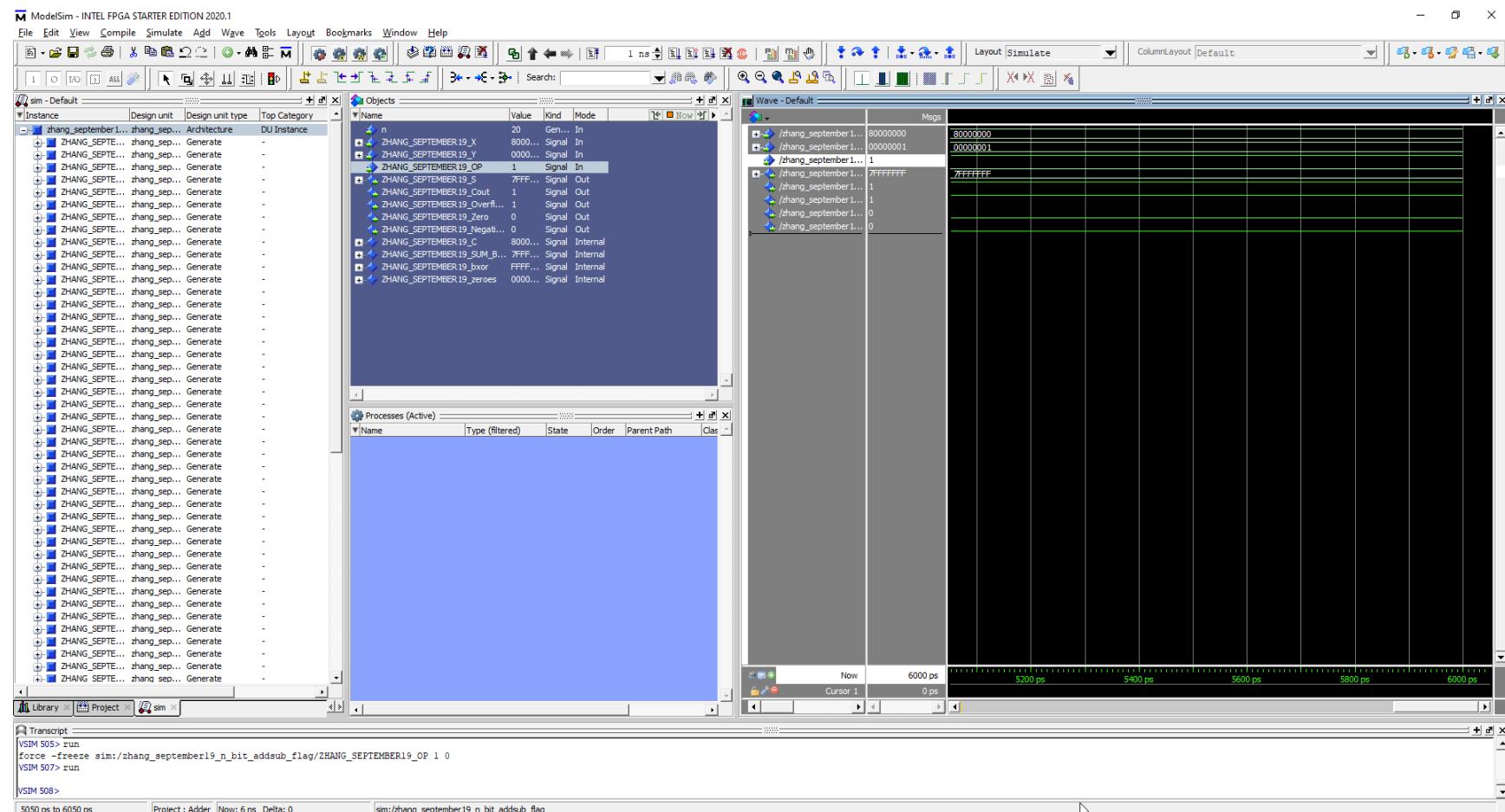
Chue Zhang , Task 8 : Overflow, Zero and Negative Flags verify n=4 and n=32



Task 8c. Most Negative N bit + 1, N = 32

The most negative N bit + 1 is 8000 0001 which is negative, which also triggers the negative flag

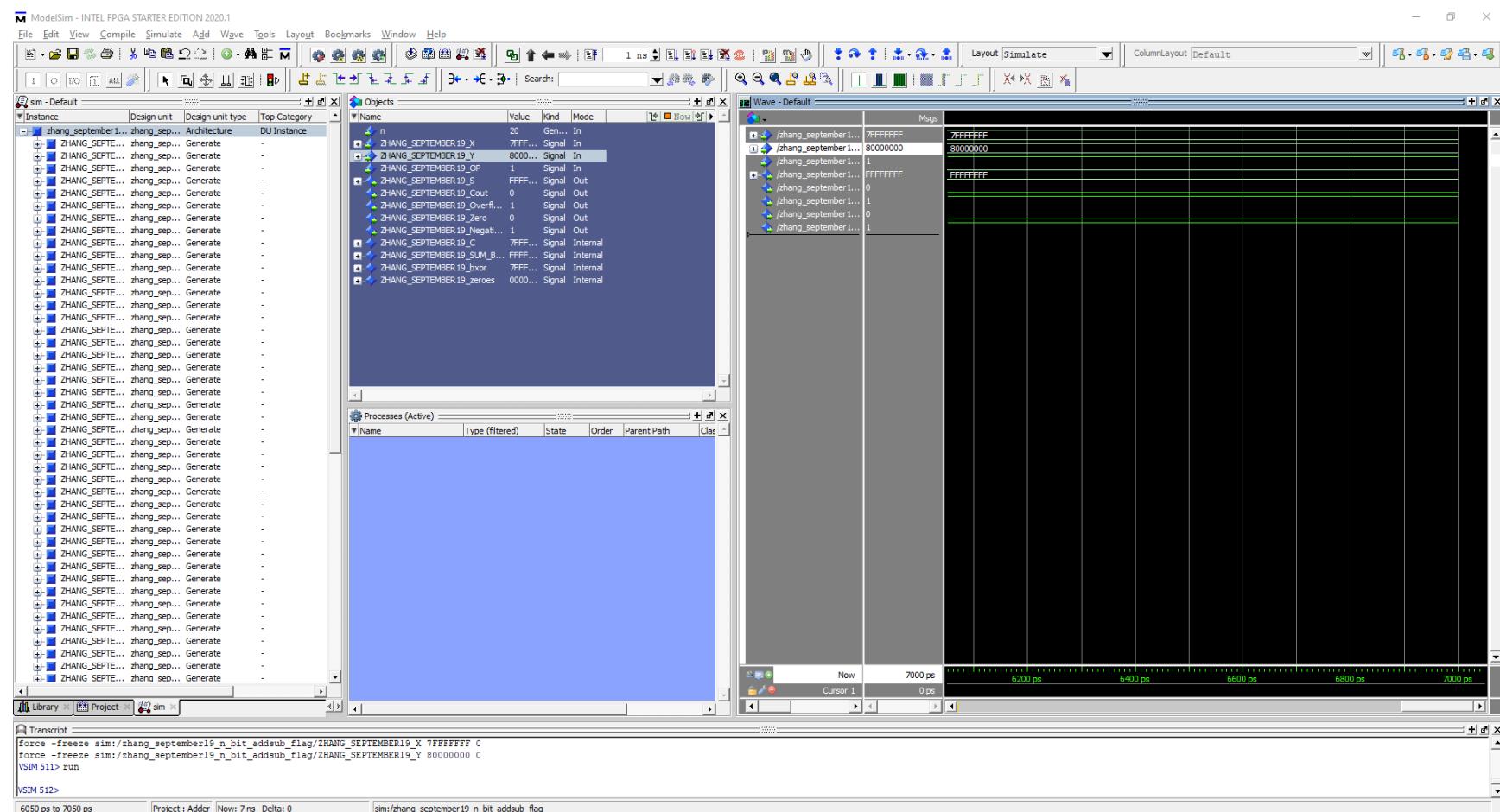
Chue Zhang , Task 8 : Overflow, Zero and Negative Flags verify n=4 and n=32



Task 8d. Most Negative N bit - 1, N = 32

Most negative N bit – 1 becomes a positive N bit therefore overflow flag is triggered

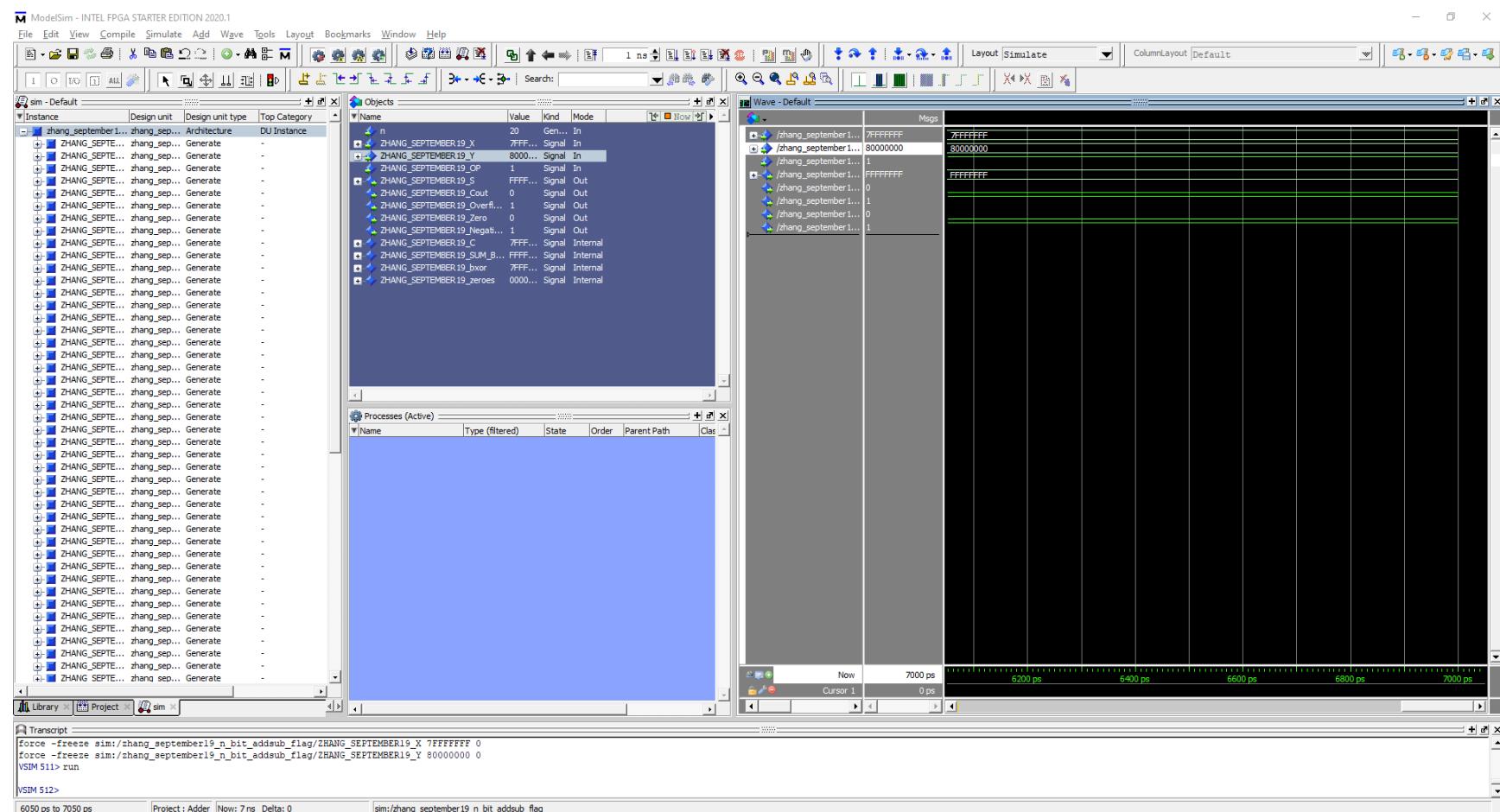
Chue Zhang , Task 8 : Overflow, Zero and Negative Flags verify n=4 and n=32



Task 8e. Most Positive N bit - Most Negative N bit, N = 32

The Most positive N bit – Most negative N bit results in FFFFFFFF which is the least negative N bit and also triggers the overflow flag and the negative flag. Overflow is triggered because double negative negates and a positive and a positive just became a negative

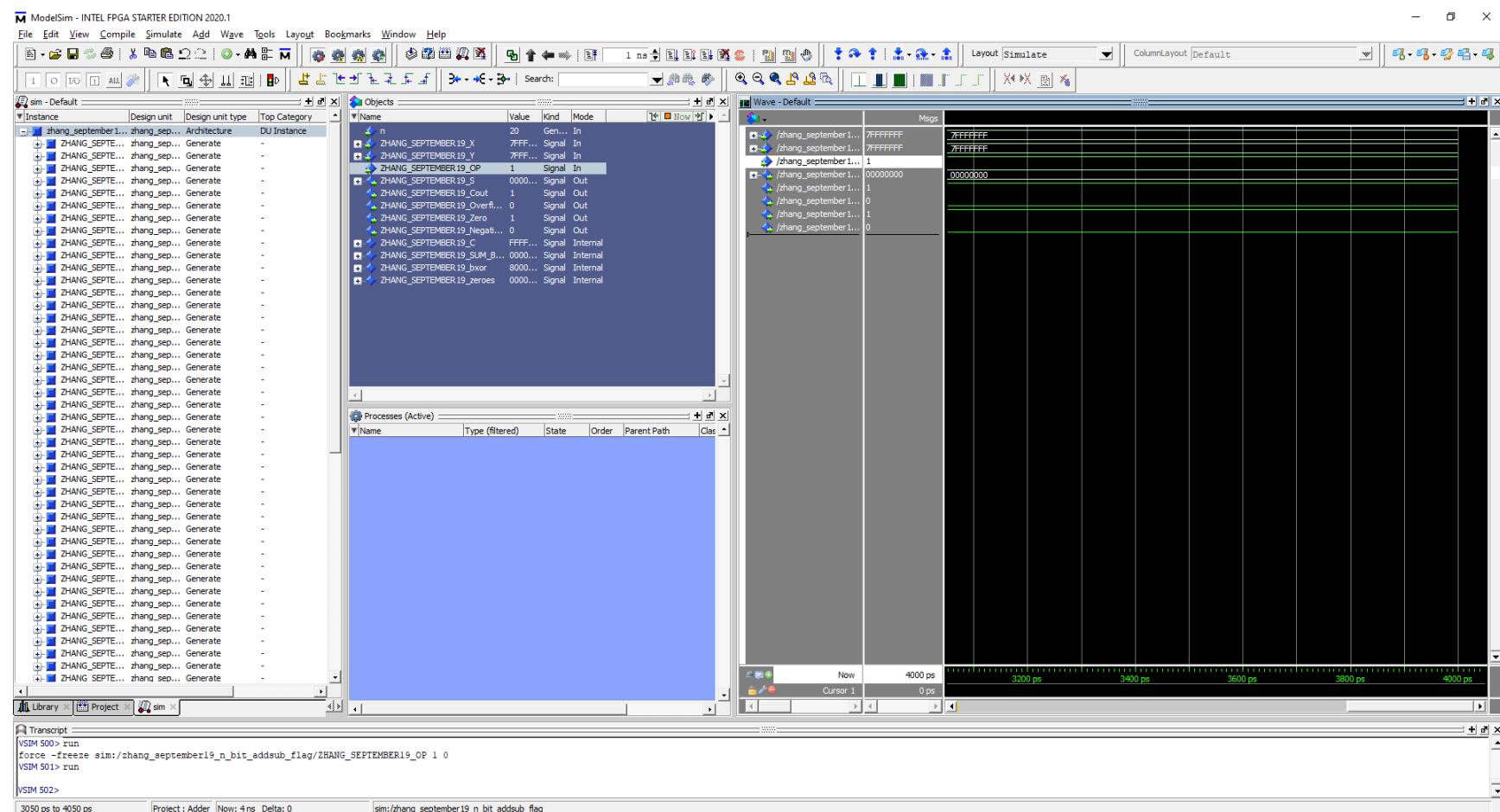
Chue Zhang , Task 8 : Overflow, Zero and Negative Flags verify n=4 and n=32



Task 8f. Most Positive N bit + Most Negative N bit, N = 32

The result becomes the same as above except no negative negation therefore no overflow flag but there is a negative flag triggered.

Chue Zhang , Task 8 : Overflow, Zero and Negative Flags verify n=4 and n=32



Task 8f. Most Positive N bit - Most Positive N bit, N = 32

Only Zero flag is triggered as the two most positive N bit integers subtracting each other is 0.

Chue Zhang , Task 9 : LPM N-bit Adder/Subtractor

The screenshot shows the Quartus Prime Lite Edition software interface. The top menu bar includes File, Edit, View, Project, Assignments, Processing, Tools, Window, and Help. A search bar at the top right says "Search altera.com". The main window has tabs for "ZHANG_SEPTMBER19_ADDERS", "ZHANG_SEPTMBER19_N_Bit_AddSub.vhd", "ZHANG_SEPTMBER19_N_Bit_AddSub_Flag.vhd", and "ZHANG_SEPTMBER19_LPM_ADDSUB.vhd".

The left sidebar shows a "Project Navigator" with files like ZHANG_SEPTMBER19_LPM_ADDSUB.qip, ZHANG_SEPTMBER19_LPM_ADDSUB.vhd, ZHANG_SEPTMBER19_N_Bit_AddSub.vhd, ZHANG_SEPTMBER19_HalfAdder.vhd, ZHANG_SEPTMBER19_FullAdder.vhd, ZHANG_SEPTMBER19_FullAdder.vhd, ZHANG_SEPTMBER19_Four_Bit_AddSub.vhd, ZHANG_SEPTMBER19_Four_Bit_Adder.vhd, ZHANG_SEPTMBER19_N_Bit_AddSub_Flag.vhd, delete_me.qip, and delete_me.vhd.

The central pane displays VHDL code for a 4-bit adder:

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_unsigned.all;
LIBRARY lpm;
USE lpm.all;
ENTITY ZHANG_SEPTMBER19_LPM_ADDSUB IS
generic (n : integer := 4);
PORT
(
    add_sub : IN STD_LOGIC ;
    cin : IN STD_LOGIC ;
    dataaa : IN STD_LOGIC_VECTOR (n-1 DOWNTO 0);
    datab : IN STD_LOGIC_VECTOR (n-1 DOWNTO 0);
    cout : OUT STD_LOGIC ;
    overflow : OUT STD_LOGIC ;
    result : OUT STD_LOGIC_VECTOR (n-1 DOWNTO 0);
    zero : out std_logic;
    negative : out std_logic
);
END ZHANG_SEPTMBER19_LPM_ADDSUB;
ARCHITECTURE SYN OF ZHANG_SEPTMBER19_LPM_ADDSUB IS
SIGNAL sub_wire0 : STD_LOGIC ;
SIGNAL sub_wire1 : STD_LOGIC ;
SIGNAL sub_wire2 : STD_LOGIC_VECTOR (n-1 DOWNTO 0);
SIGNAL zeroes : STD_LOGIC_VECTOR (n-1 DOWNTO 0);
SIGNAL Test : STD_LOGIC_VECTOR (n-1 DOWNTO 0);
COMPONENT lpm_add_sub
GENERIC (
    lpm_direction : STRING;
    lpm_hint : STRING;
    lpm_representation : STRING;
    lpm_type : STRING;
    lpm_width : NATURAL
);
PORT (
    add_sub : IN STD_LOGIC ;
    cin : IN STD_LOGIC ;
    dataaa : IN STD_LOGIC_VECTOR (n-1 DOWNTO 0);
    datab : IN STD_LOGIC_VECTOR (n-1 DOWNTO 0);
    cout : OUT STD_LOGIC ;
    overflow : OUT STD_LOGIC ;
    result : OUT STD_LOGIC_VECTOR (n-1 DOWNTO 0)
);
```

The bottom pane shows a "Messages" window with a table of errors and warnings, and a "System" tab indicating 1 processing task.

LPM N-Bit Adder/Subtractor VHDL code with flags part 1

Chue Zhang , Task 9 : LPM N-bit Adder/Subtractor

Quartus Prime Lite Edition - C:/Users/czhan/Desktop/Schoolwork/Gertner/Zhang_CS343_FA21/ZHANG_SEPTMBER19_ADDERS/ZHANG_SEPTMBER19_ADDERS - ZHANG_SEPTMBER19_ADDERS

File Edit View Project Assignments Processing Tools Window Help Search altera.com

ZHANG_SEPTMBER19_ADDERS

Project Navigator Files ZHANG_SEPTMBER19_N_Bit_AddSub.vhd ZHANG_SEPTMBER19_N_Bit_AddSub_Flag.vhd ZHANG_SEPTMBER19_LPM_ADDSUB.vhd

```

51 BEGIN
52
53
54
55    zero_allocate: for i in 0 to n-1
56        generate -- for zero flag, generates 'N' number of 0's
57            zeroes(i) <= '0';
58        end generate;
59
60    cout <= sub_wire0;
61    overflow <= sub_wire1;
62    result <= sub_wire2(n-1 downto 0);
63    zero <= '1' when sub_wire2 = zeroes else '0';
64
65    negative <= sub_wire2(n-1);
66
67
68
69 LPM_ADD_SUB_component : LPM_ADD_SUB
70 GENERIC MAP (
71     lpm_direction => "UNUSED",
72     lpm箅术操作=>"LPM_ADD_SUB",
73     lpm_representation => "SIGNED",
74     lpm_type => "LPM_ADD_SUB",
75     lpm_width => n
76 )
77 PORT MAP (
78     add_sub => add_sub,
79     cin => cin,
80     dataa => dataa,
81     datab => datab,
82     cout => sub_wire0,
83     overflow => sub_wire1,
84     result => sub_wire2
85 );
86
87
88 END SYN;
89
90 -- CNX file retrieval info
91 -- Retrieval info: PRIVATE: Carryin NUMERIC "1"
92 -- Retrieval info: PRIVATE: Carryout NUMERIC "1"
93 -- Retrieval info: PRIVATE: Constanta NUMERIC "0"
94 -- Retrieval info: PRIVATE: Function NUMERIC "1"
95 -- Retrieval info: PRIVATE: Function NUMERIC "2"
96 -- Retrieval info: PRIVATE: INTENDED_DEVICE_FAMILY STRING "Cyclone V"
97 -- Retrieval info: PRIVATE: INTENDED_DEVICE_FAMILY STRING "Cyclone V"
98 -- Retrieval info: PRIVATE: INTENDED_DEVICE_FAMILY STRING "Cyclone V"

```

Tasks Compilation

- Task
 - Compile Design
 - Analysis & Synthesis
 - Edit Settings
 - View Report
 - Analysis & Elaboration
 - Partition Merge
 - Netlist Viewers
 - Design Assistant (Post-Mapping)

Messages

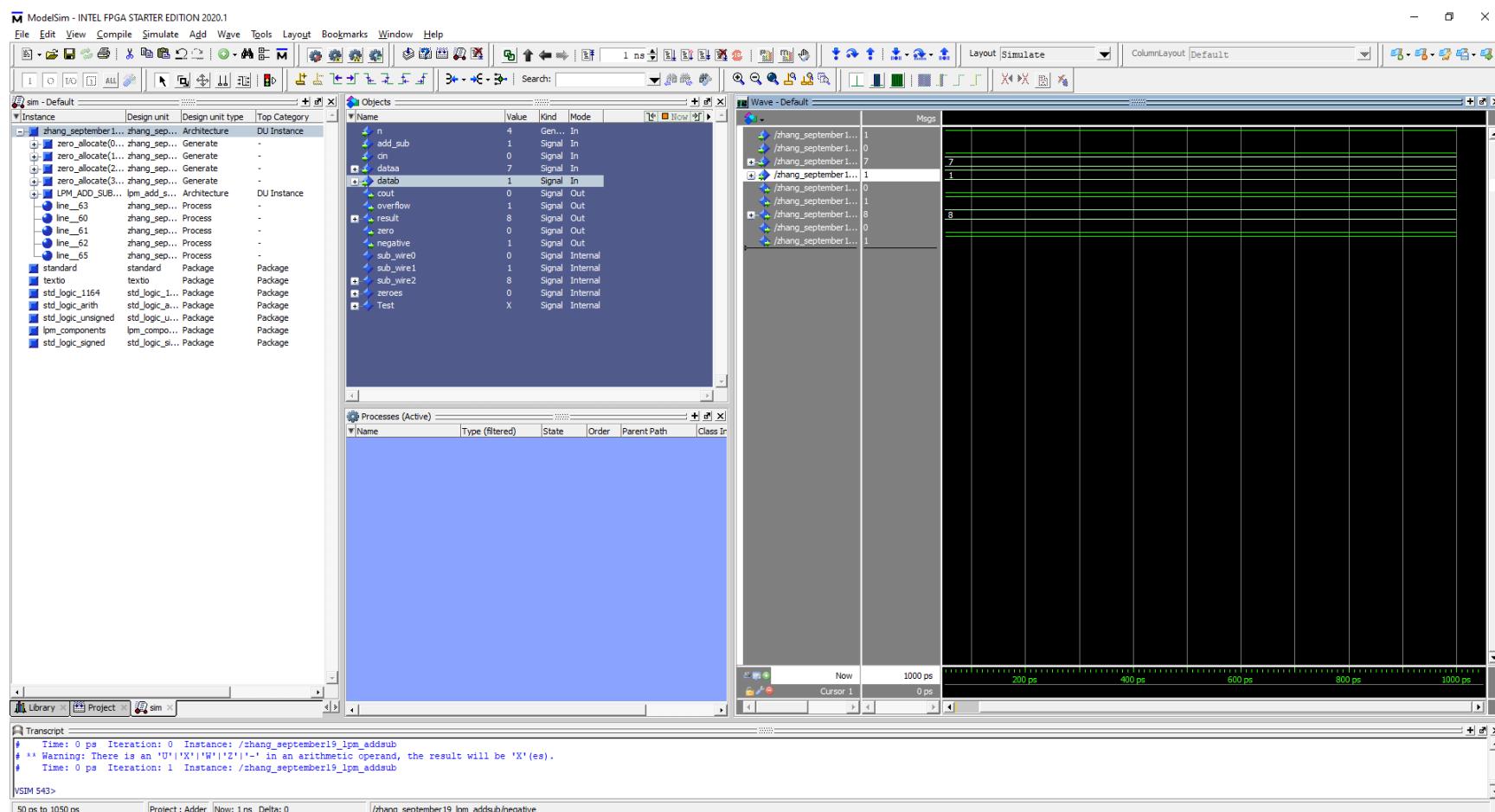
| Type | ID | Message |
|----------|--|---------|
| 332140 | No Removal paths to report | |
| 332140 | No Minimum Pulse width paths to report | |
| 332102 | Design is not fully constrained for setup requirements | |
| 332102 | Design is not fully constrained for hold requirements | |
| > 293000 | quartus Prime Timing Analyzer was successful. 0 errors, 6 warnings | |
| > 293000 | quartus Prime Full Compilation was successful. 0 errors, 14 warnings | |

System (1) Processing (138)

100% 00:00:54

LPM N-B it Adder/Subtractor VHDL code with flags part 2

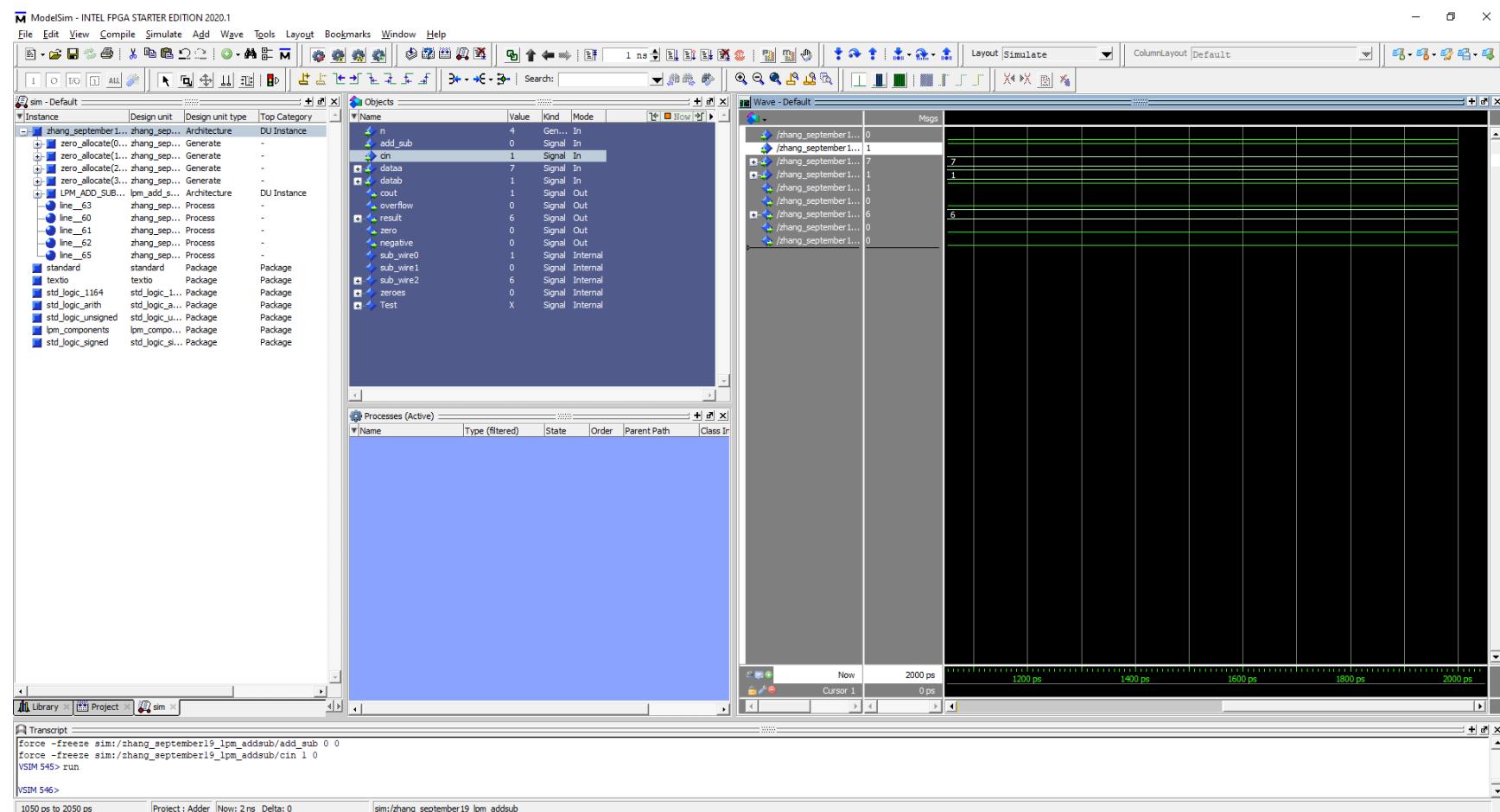
Chue Zhang , Task 9 : LPM N-bit Adder/Subtractor



Task 8a. Most Positive N bit + 1, N = 4

Most positive N = 7 which when added with 1 triggers the overflow flag and the negative flag. Two positives became a negative so overflow flag and its negative so negative flag

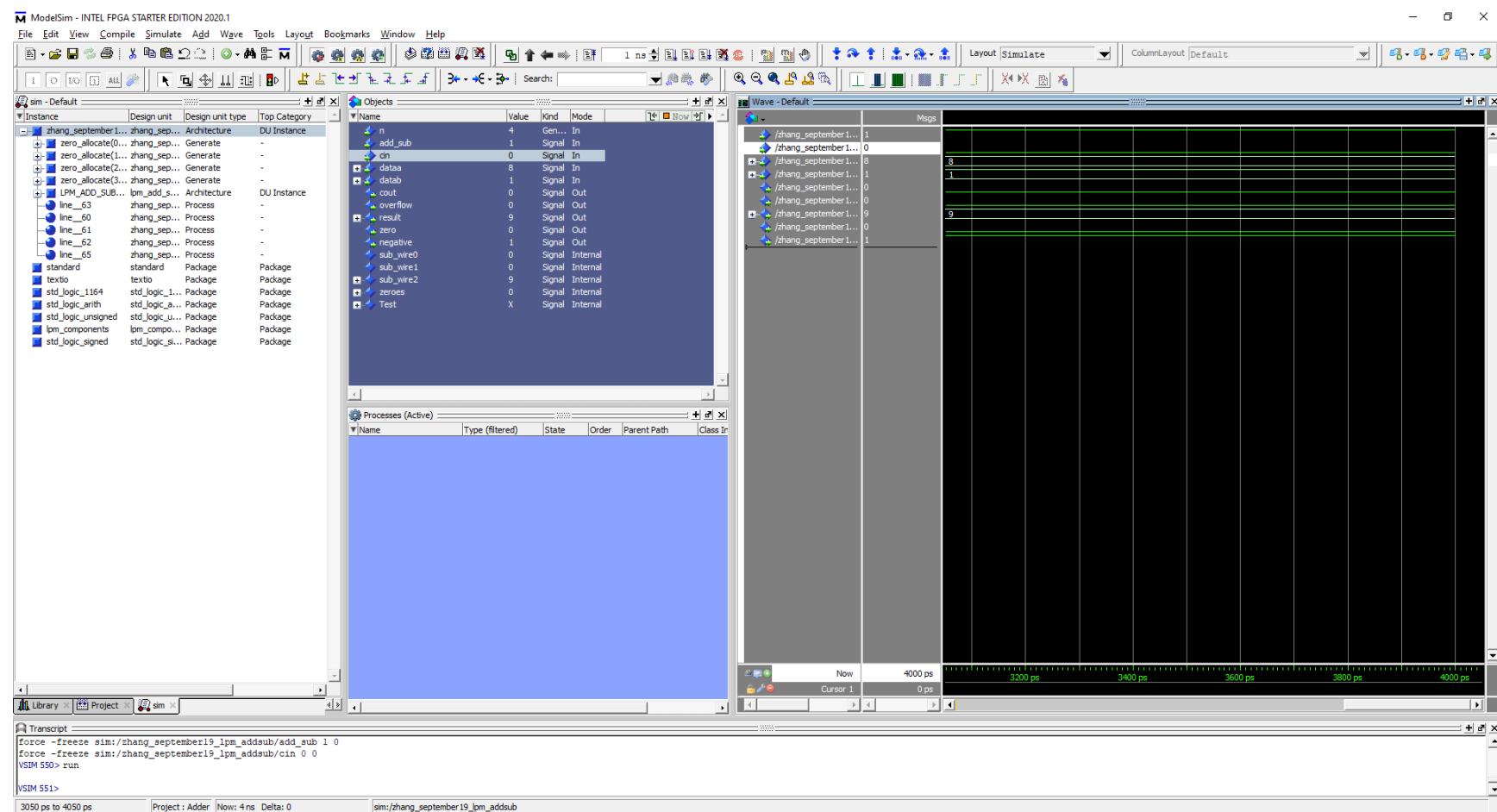
Chue Zhang , Task 9 : LPM N-bit Adder/Subtractor



Task 8b. Most Positive N bit - 1, N = 4

$7 - 1 = 6$ which leaves it as a positive number and no other flags trigger

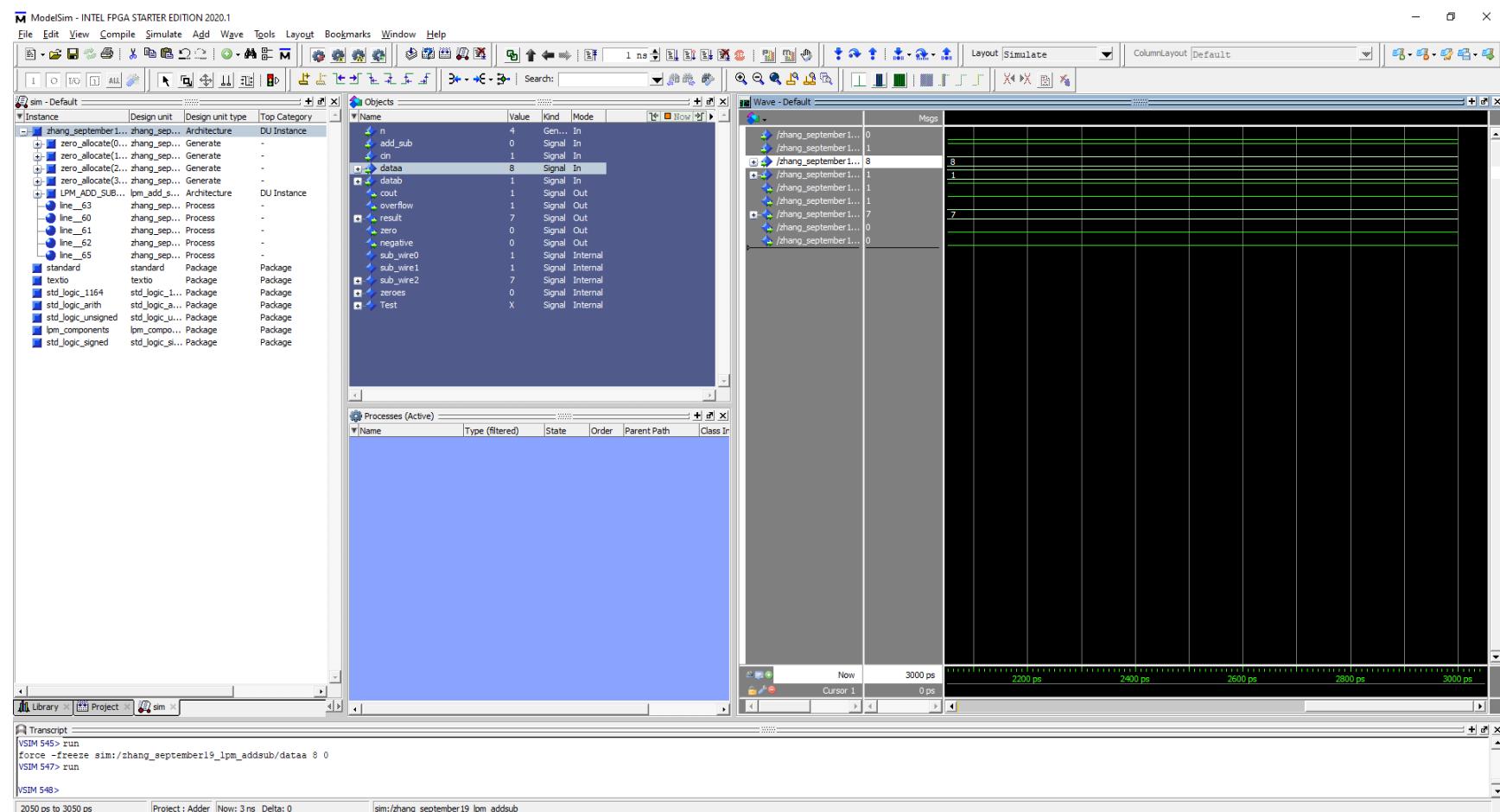
Chue Zhang , Task 9 : LPM N-bit Adder/Subtractor



Task 8c. Most Negative N bit + 1, N = 4

8 is the most negative N bit and when added with 1, it becomes 9 which triggers the negative flag only

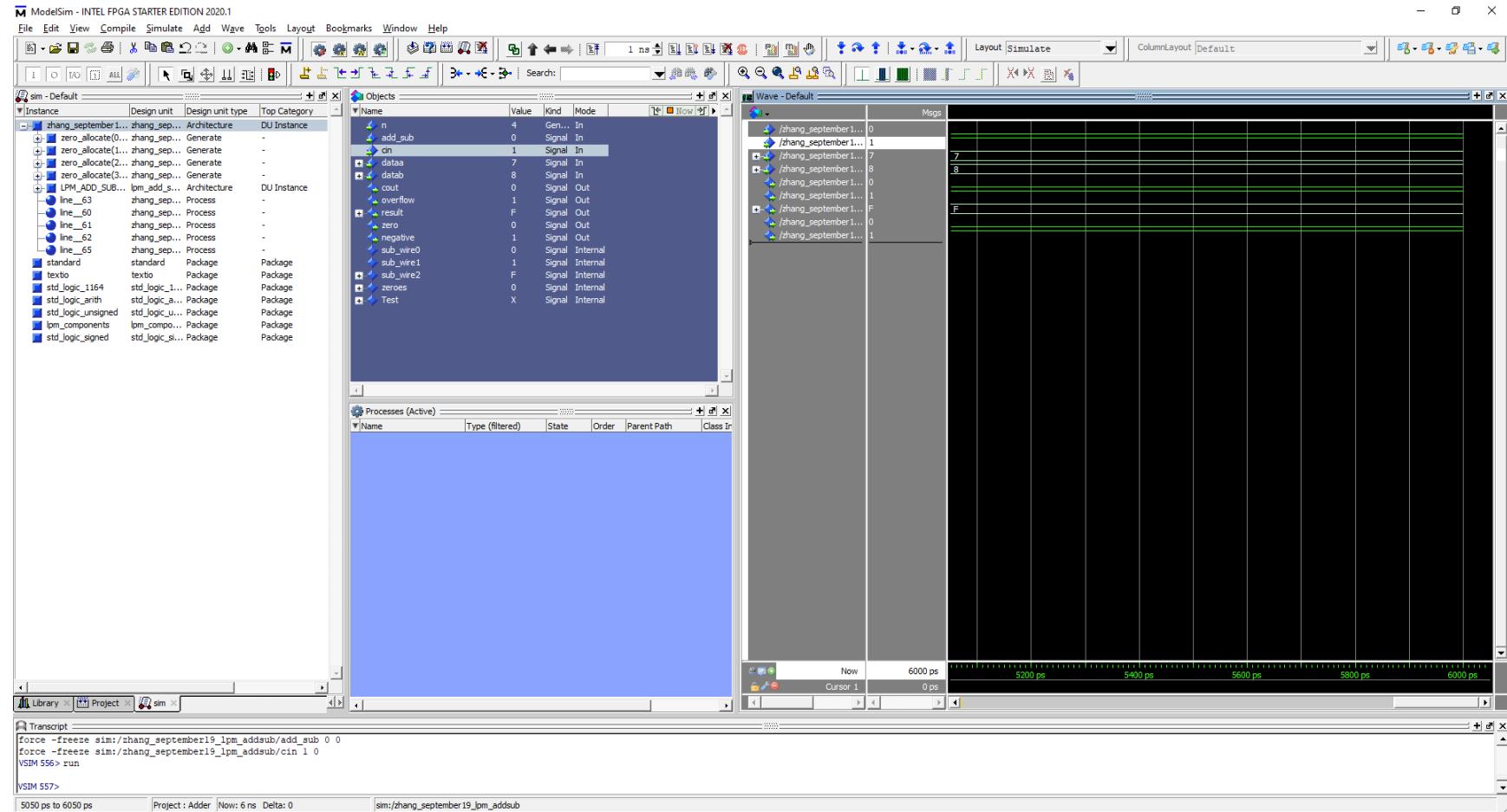
Chue Zhang , Task 9 : LPM N-bit Adder/Subtractor



Task 8d. Most Negative N bit - 1, N = 4

$8 - 1 = 7$ which triggers the overflow flag because of the sign change

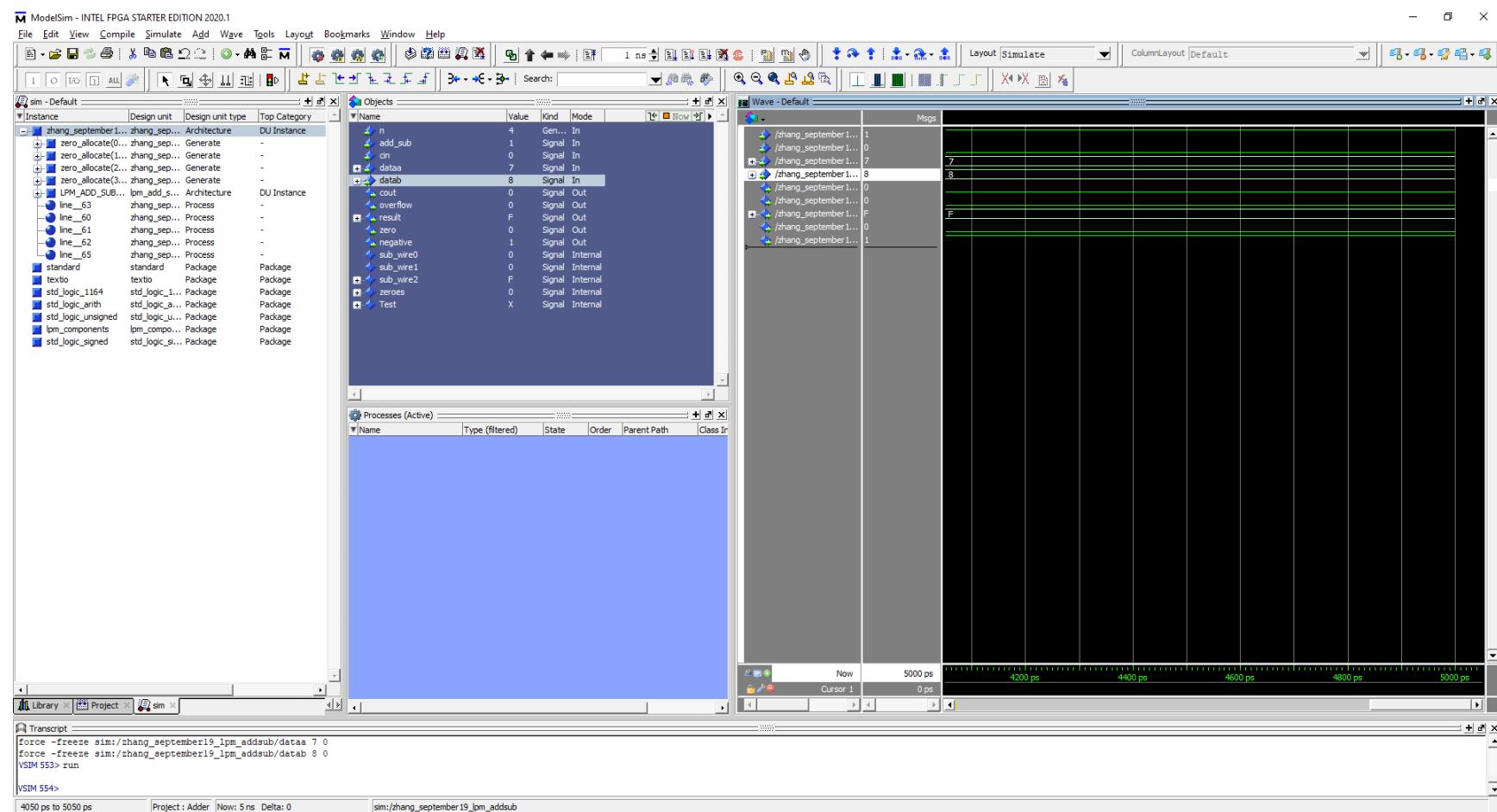
Chue Zhang , Task 9 : LPM N-bit Adder/Subtractor



Task 8e. Most Positive N bit - Most Negative N bit , N = 4

$7 - 8 = -1$ which means it goes from negative to positive to negative which triggers the overflow flag as well as the negative flag

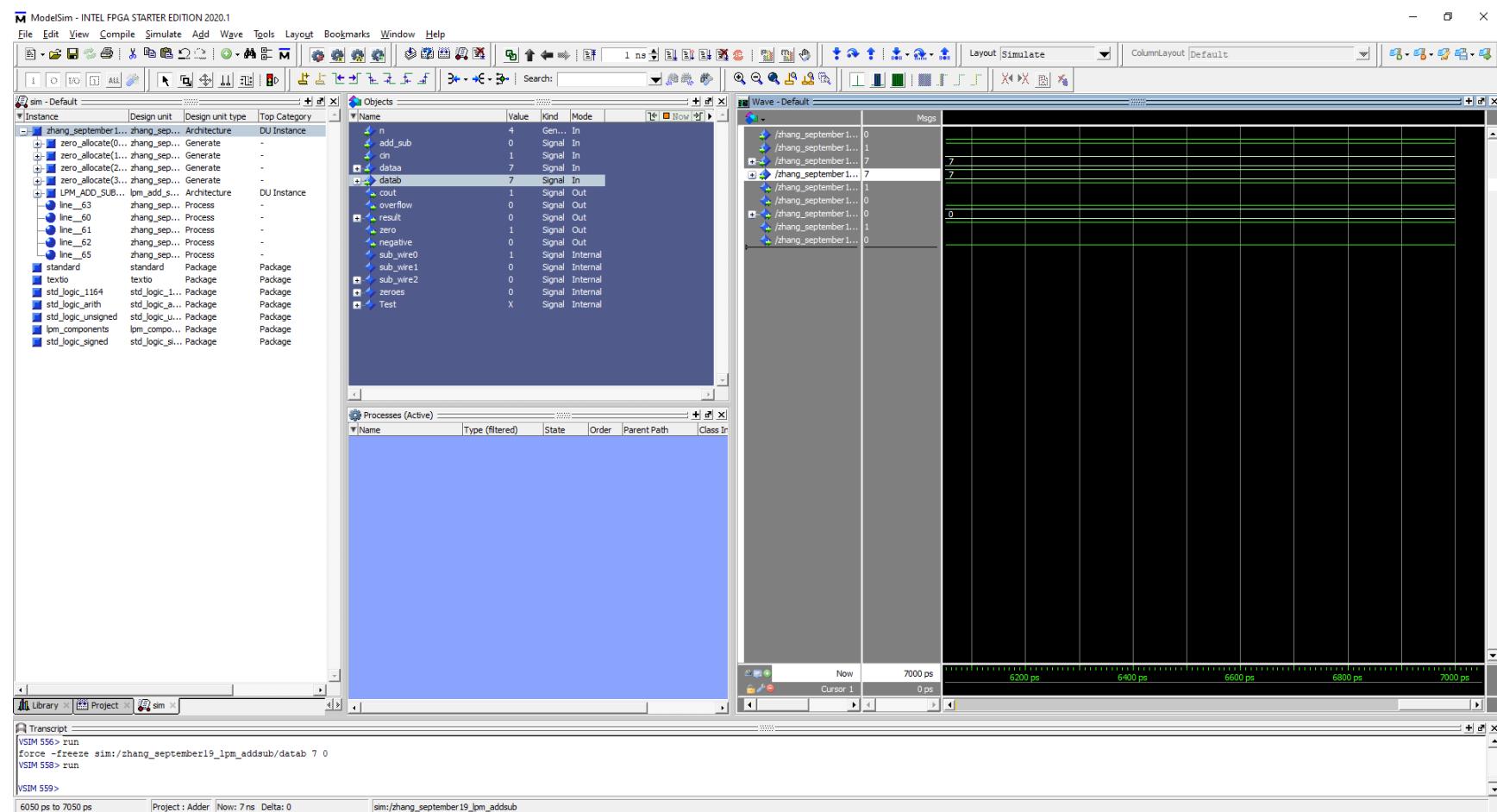
Chue Zhang , Task 9 : LPM N-bit Adder/Subtractor



Task 8e. Most Positive N bit + Most Negative N bit , N = 4

$7+8 = 15$ which is negative so only the negative flag is triggered, no overflow occurs

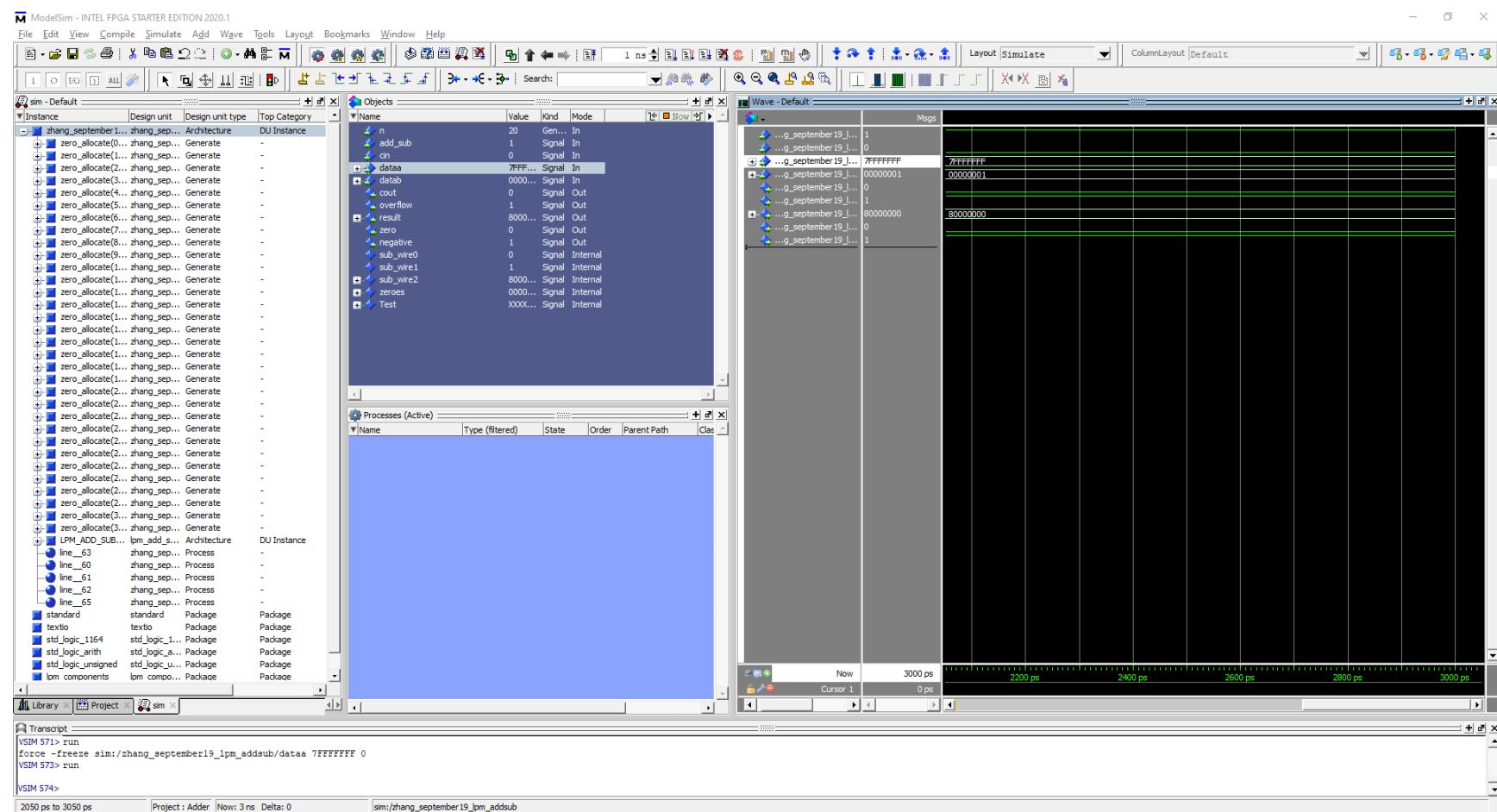
Chue Zhang , Task 9 : LPM N-bit Adder/Subtractor



Task 8f. Most Positive N bit - Most Positive N bit , N = 4

$7 - 7 = 0$ which only triggers the zero flag

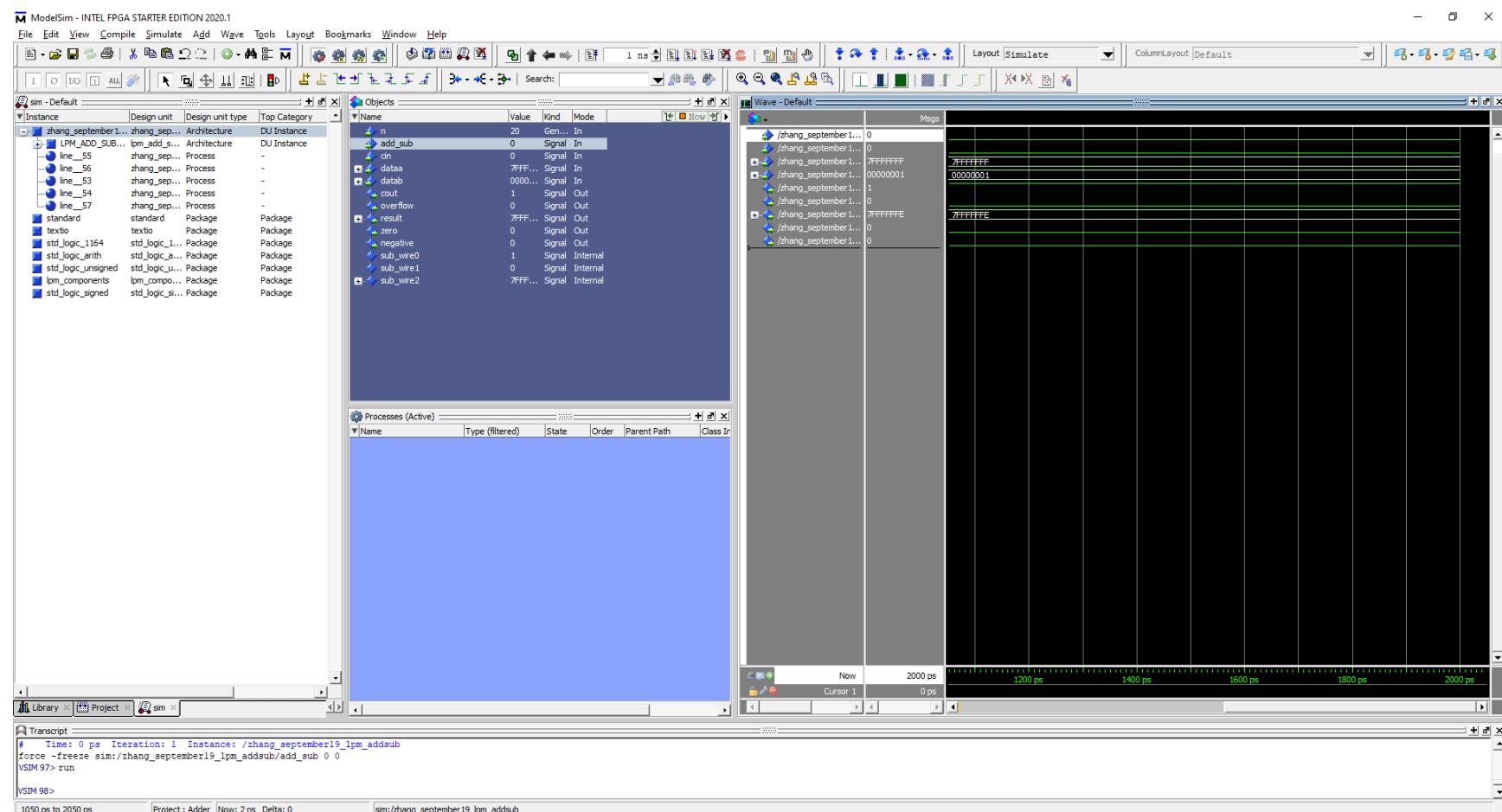
Chue Zhang , Task 9 : LPM N-bit Adder/Subtractor



Task 8a. Most Positive N bit + 1, N = 32

Most positive N bit integer + 1 becomes negative afterwards so negative flag triggered alongside overflow

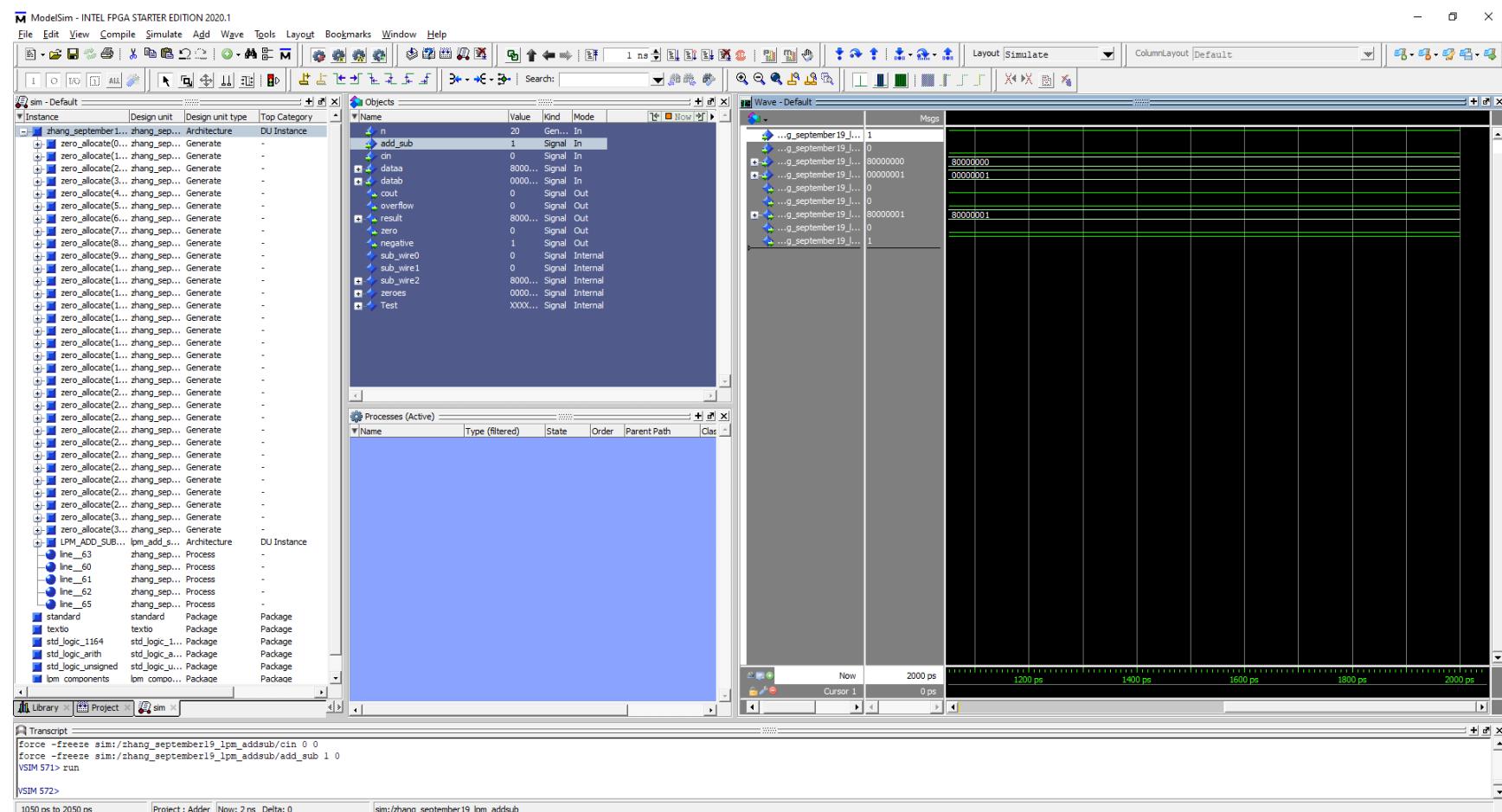
Chue Zhang , Task 9 : LPM N-bit Adder/Subtractor



Task 8b. Most Positive N bit - 1, N = 32

Most positive N bit – 1 is still a positive integer value therefore no flags are triggered

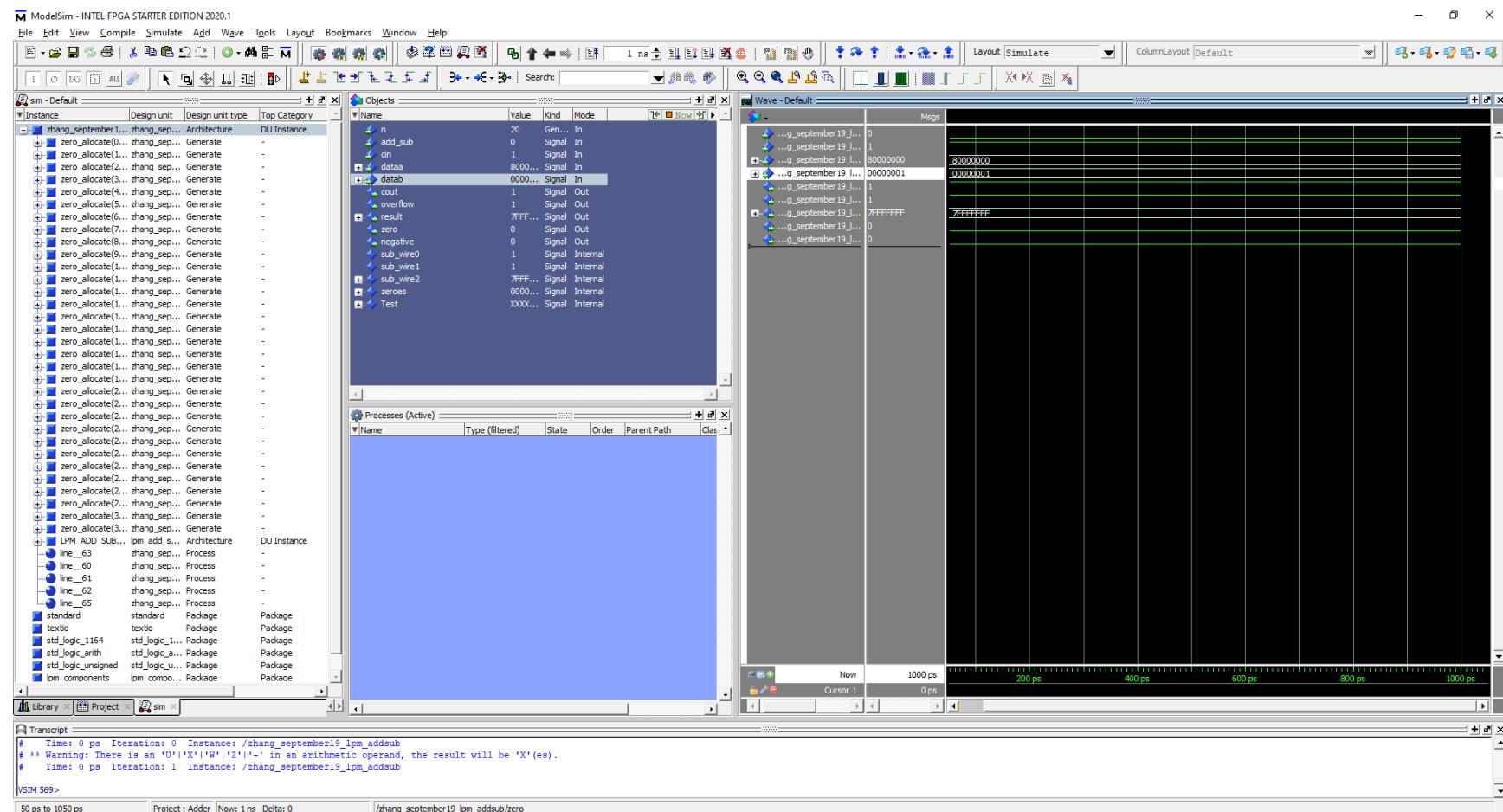
Chue Zhang , Task 9 : LPM N-bit Adder/Subtractor



Task 8c. Most Negative N bit + 1, N = 32

Most Negative bit + 1 because '80000001' which is still negative therefore negative flag is triggered

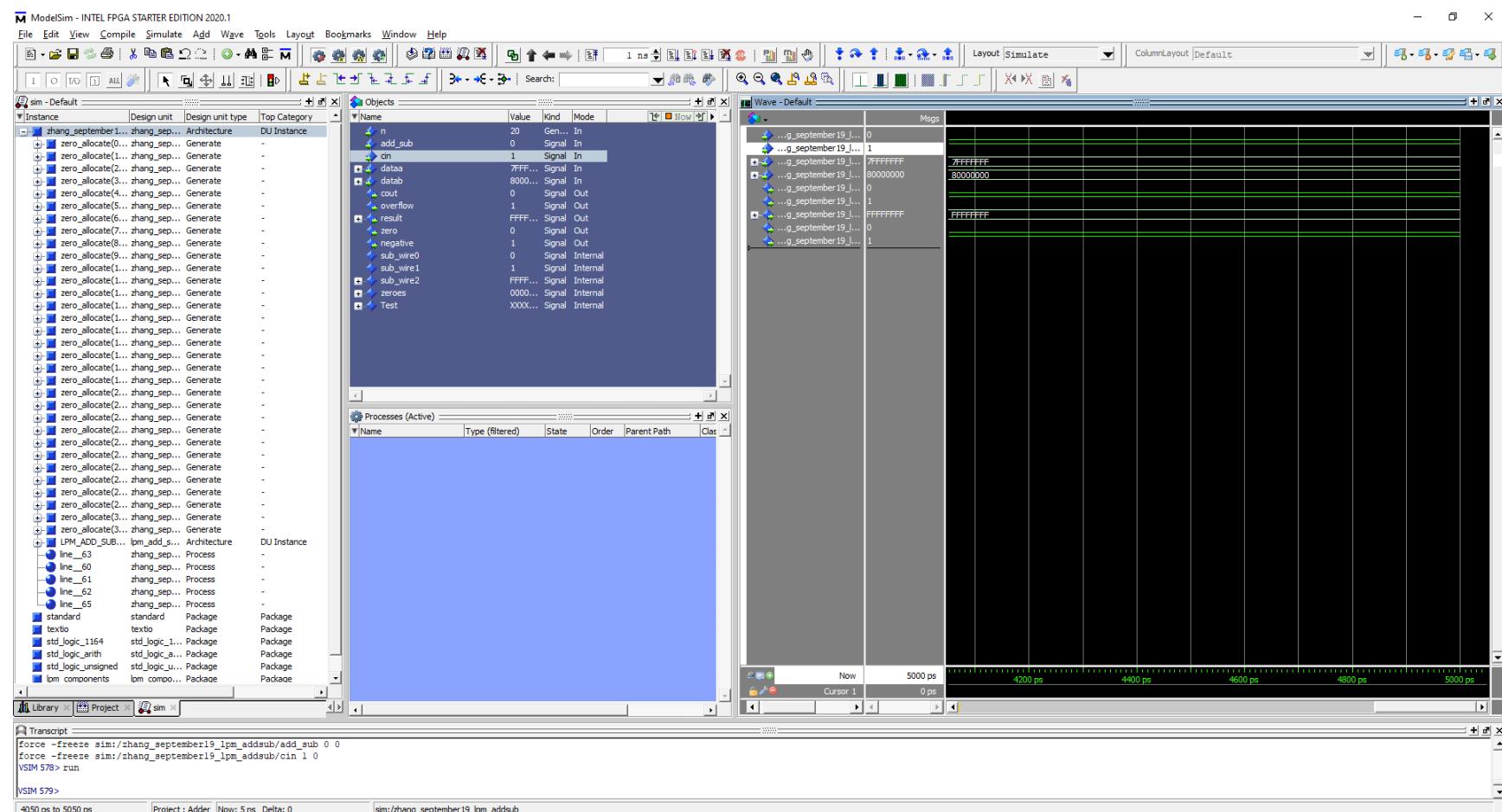
Chue Zhang , Task 9 : LPM N-bit Adder/Subtractor



Task 8d. Most Negative N bit - 1, N = 32

Most Negative N bit – 1 becomes positive therefore overflow flag is triggered only.

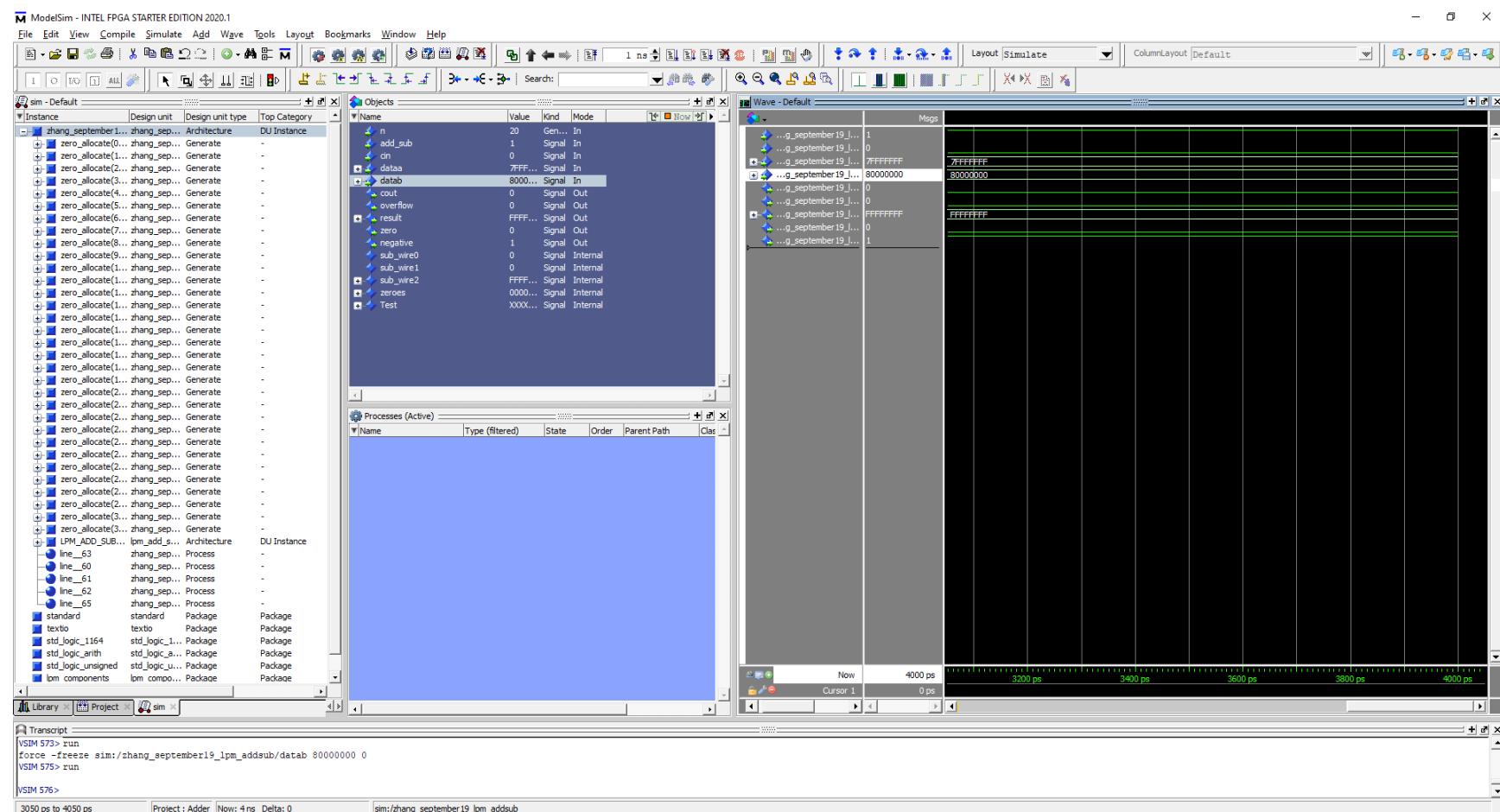
Chue Zhang , Task 9 : LPM N-bit Adder/Subtractor



Task 8e. Most Positive N bit - Most Negative N bit , N = 32

The output goes from negative to positive to negative therefore overflow flag is triggered alongside the negative flag. FFFFFFFF is the smallest negative bit and is also the limit.

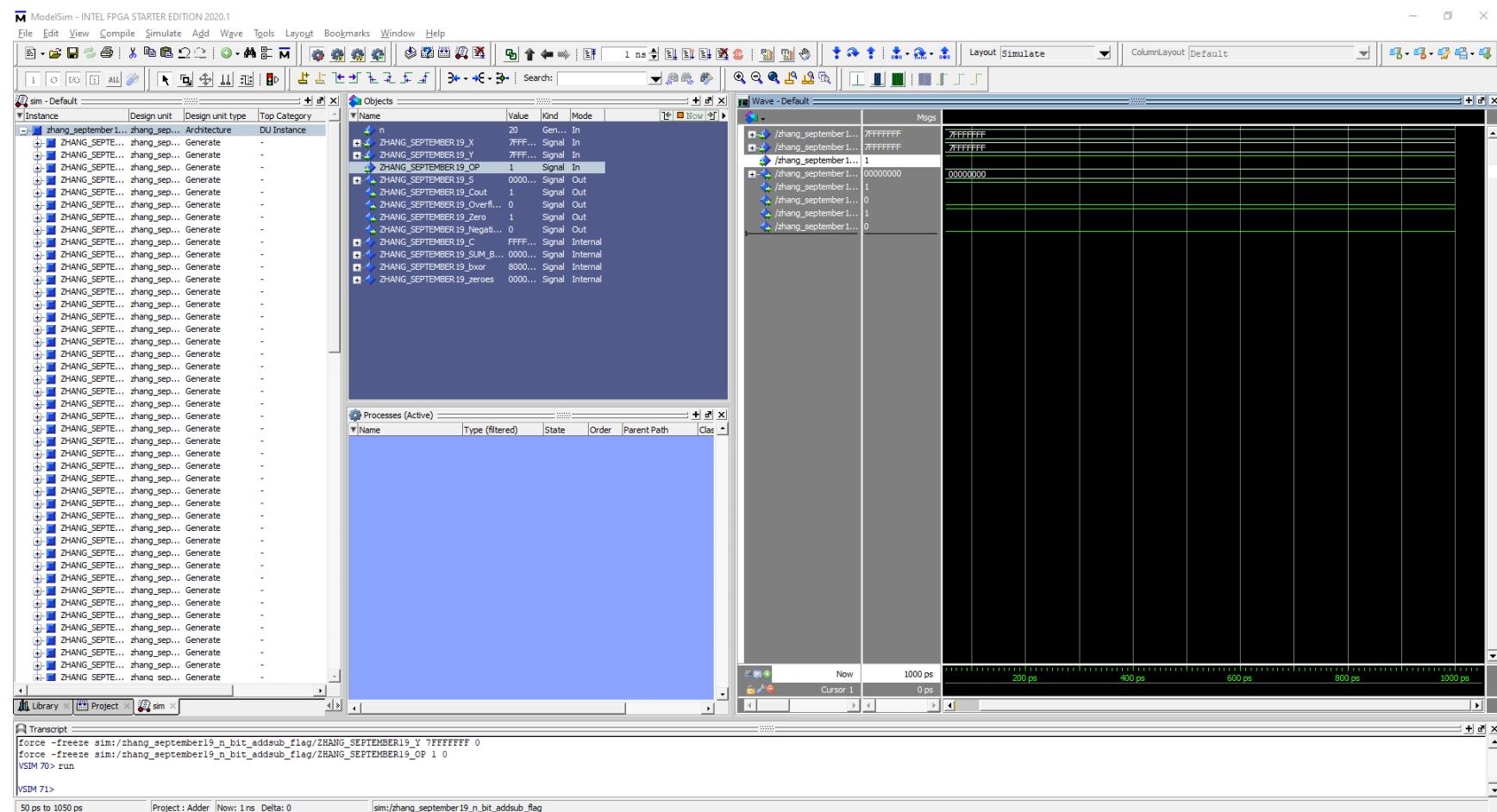
Chue Zhang , Task 9 : LPM N-bit Adder/Subtractor



Task 8f. Most Positive N bit + Most Negative N bit , N = 32

Nothing is triggered except for negative flag and there is no overflow because it goes from negative to negative

Chue Zhang , Task 9 : LPM N-bit Adder/Subtractor



Task 8g. Most Positive N bit - Most Positive N bit , N = 32

Two identical values subtracting each other results in zero thus triggering the zero flag

Chue Zhang , Task 10 : Testbench NO FLAGS

```

use ieee.numeric_std.all;
use ieee.std_logic_unsigned.all;

entity ZHANG_SEPTMBER19_16bit_AddSub_tb is
    generic (n : integer := 16);
end ZHANG_SEPTMBER19_16bit_AddSub_tb;

architecture ZHANG_SEPTMBER19_arch of ZHANG_SEPTMBER19_16bit_AddSub_tb is
begin
    ZHANG_SEPTMBER19_flag: ZHANG_SEPTMBER19_flag
        generic map (n)
        port map (ZHANG_SEPTMBER19_X, ZHANG_SEPTMBER19_Y, ZHANG_SEPTMBER19_OP, ZHANG_SEPTMBER19_S, ZHANG_SEPTMBER19_Cout, ZHANG_SEPTMBER19_Overflow);
end architecture;

```

Image above is the testbench code

Chue Zhang , Task 10 : Testbench NO FLAGS

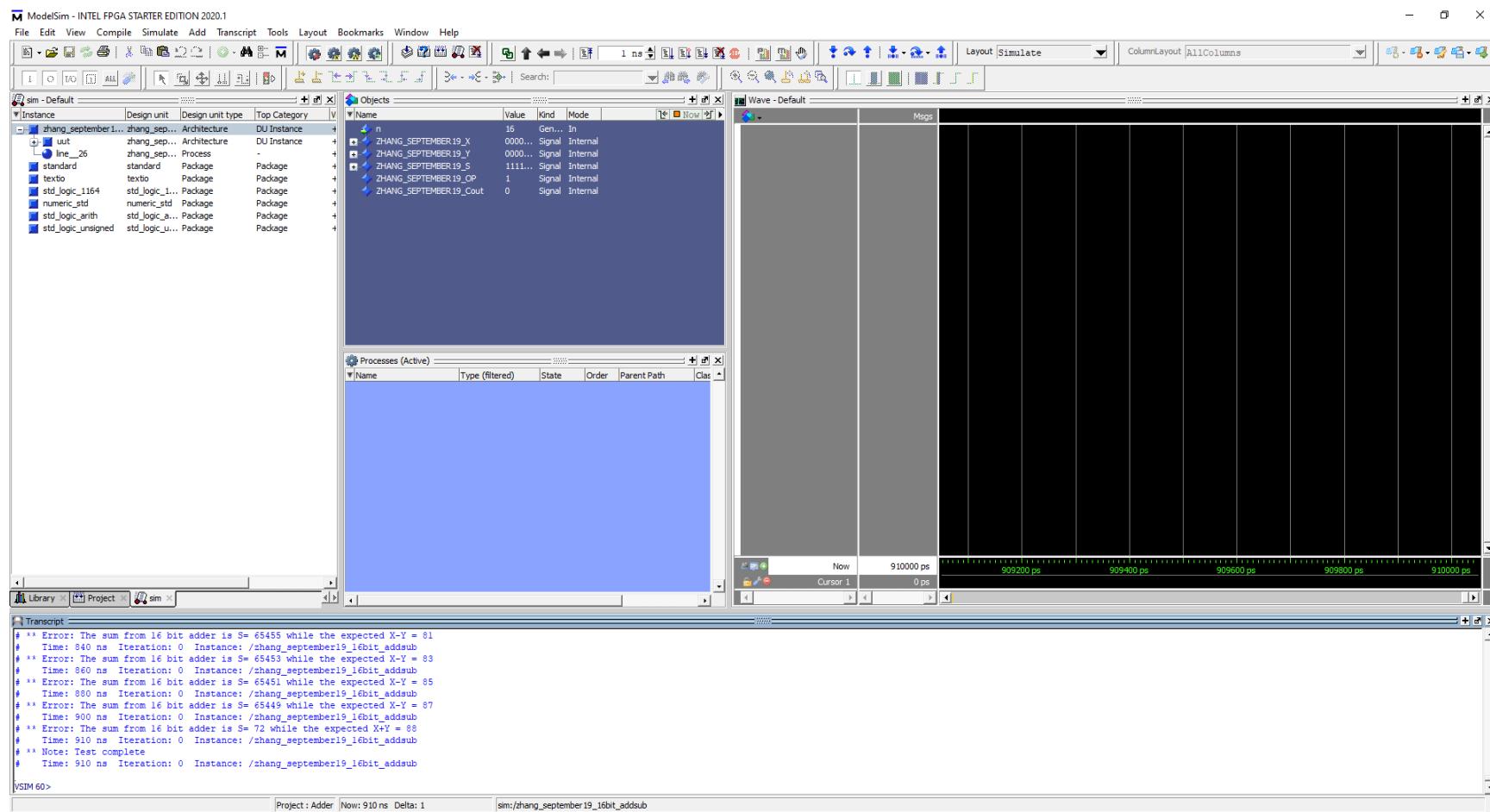
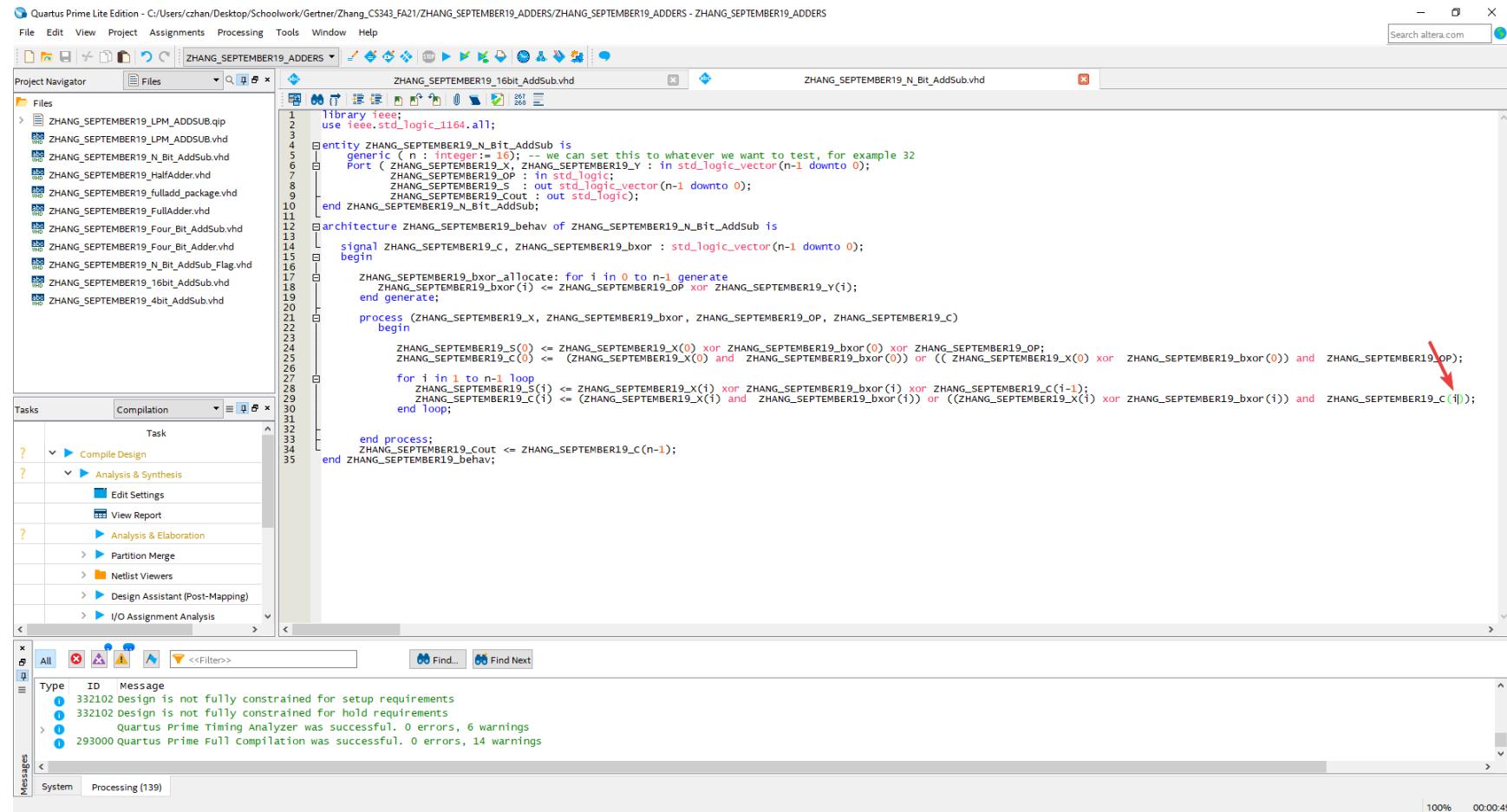


Figure above showcases the testbench and the errors if there is a fault in the code

Chue Zhang , Task 10 : Testbench NO FLAGS



The screenshot shows the Quartus Prime Lite Edition interface with the following details:

- File Menu:** File, Edit, View, Project, Assignments, Processing, Tools, Window, Help.
- Toolbar:** Standard icons for opening files, saving, running simulations, and more.
- Project Navigator:** Shows the project structure with files like ZHANG_SEPTMBER19_LPM_ADDSUB.qip, ZHANG_SEPTMBER19_N_Bit_AddSub.vhd, etc.
- Code Editor:** Displays the VHDL code for ZHANG_SEPTMBER19_16bit_AddSub.vhd. The code defines an entity ZHANG_SEPTMBER19_N_Bit_AddSub with a generic parameter n and ports for X, Y, OP, and Cout. It includes a behavioral architecture ZHANG_SEPTMBER19_behav with a process that calculates the sum and carry bits using a loop from i=0 to n-1. A red arrow highlights a syntax error in the assignment statement for ZHANG_SEPTMBER19_C(i) on line 29.
- Tasks Panel:** Shows compilation tasks: Compile Design, Analysis & Synthesis, View Report, Analysis & Elaboration, Partition Merge, Netlist Viewers, Design Assistant (Post-Mapping), I/O Assignment Analysis.
- Messages Panel:** Shows compilation messages:

| Type | ID | Message |
|------|--------|--|
| 1 | 332102 | Design is not fully constrained for setup requirements |
| 1 | 332102 | Design is not fully constrained for hold requirements |
| > | 1 | Quartus Prime Timing Analyzer was successful. 0 errors, 6 warnings |
| 1 | 293000 | quartus prime Full Compilation was successful. 0 errors, 14 warnings |

This was the code that we tested on our testbench, the red arrow points to where the fault is so we correct it.

Chue Zhang , Task 10 : Testbench NO FLAGS

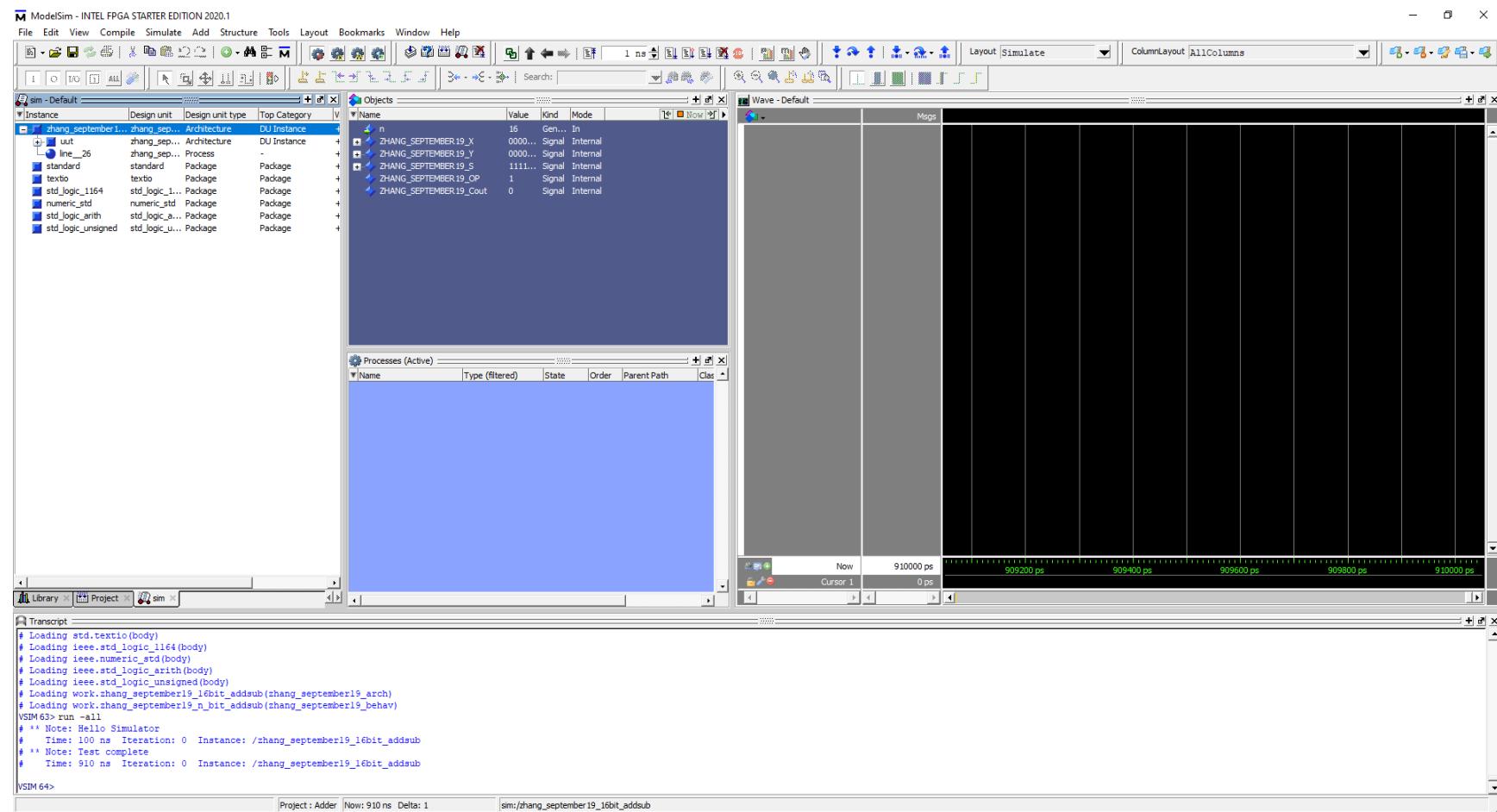
The screenshot shows the Quartus Prime Lite Edition interface with the following details:

- File Menu:** File, Edit, View, Project, Assignments, Processing, Tools, Window, Help.
- Search Bar:** Search altera.com
- Project Navigator:** Shows files related to the project, including ZHANG_SEPTMBER19_LPM_ADDSUB.qip, ZHANG_SEPTMBER19_LPM_ADDSUB.vhd, ZHANG_SEPTMBER19_N_Bit_AddSub.vhd, ZHANG_SEPTMBER19_HalfAdder.vhd, ZHANG_SEPTMBER19_fulladd_package.vhd, ZHANG_SEPTMBER19_FullAdder.vhd, ZHANG_SEPTMBER19_Four_Bit_AddSub.vhd, ZHANG_SEPTMBER19_Four_Bit_Adder.vhd, ZHANG_SEPTMBER19_N_Bit_AddSub_Flag.vhd, ZHANG_SEPTMBER19_16bit_AddSub.vhd, and ZHANG_SEPTMBER19_4bit_AddSub.vhd.
- Code Editor:** Displays the VHDL code for ZHANG_SEPTMBER19_16bit_AddSub.vhd. The code defines an entity ZHANG_SEPTMBER19_N_Bit_AddSub with a generic parameter n (ranging from 1 to 16) and ports for X, Y, OP, and Cout. It uses a process to implement the addition logic using XOR operations.
- Tasks Panel:** Shows compilation tasks:
 - Compile Design (highlighted)
 - Analysis & Synthesis
 - Edit Settings
 - View Report
 - Analysis & Elaboration
 - Partition Merge
 - Netlist Viewers
 - Design Assistant (Post-Mapping)
 - I/O Assignment Analysis
- Messages Panel:** Displays compilation messages:

| Type | ID | Message |
|------|--------|--|
| 1 | 332102 | Design is not fully constrained for setup requirements |
| 1 | 332102 | Design is not fully constrained for hold requirements |
| > | 1 | Quartus Prime Timing Analyzer was successful. 0 errors, 6 warnings |
| 1 | 293000 | Quartus Prime Full Compilation was successful. 0 errors, 14 warnings |
- Bottom Status:** Ln 29 Col 185, VHDL File, 100%, 00:00:49.

Corrected N bit Add/Sub code to be simulated on

Chue Zhang , Task 10 : Testbench NO FLAGS



New test runs and we see that there are no errors created.