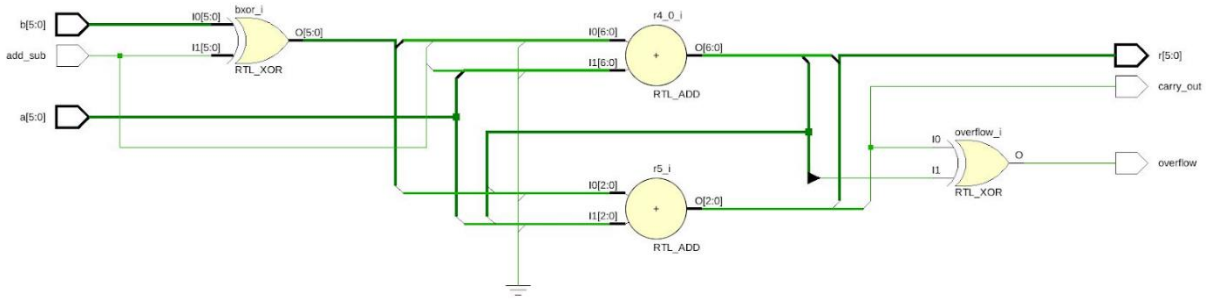
Lab 2: 2's Complement Adder/Subtractor

-Overview-

This lab involves creating a 4-bit adder/subtractor using VHDL, with carry-out and two-complement overflow detection. The task requires manipulating VHDL arrays, using the numeric_std library, and handling addition/subtraction through conditional logic. In two's-complement, subtraction is performed by inverting the second input and adding 1. The lab consists of two parts: first, implementing and simulating the 6-bit design, and second, testing it on the Nexys A7 FPGA board. The lab also offers extra credit for optimizing the VHDL code with minimal statements.

-Design-

Logic Diagram



Main VHDL

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.numeric_std.all;

entity adder_6bit is
    Port ( a : in STD_LOGIC_VECTOR (5 downto 0);
           b : in STD_LOGIC_VECTOR (5 downto 0);
           add_sub : in STD_LOGIC;
           r : out STD_LOGIC_VECTOR (5 downto 0);
           carry_out : out STD_LOGIC;
           overflow : out STD_LOGIC);
end adder_6bit;

architecture Behavioral of adder_6bit is
    signal bxor : std_logic_vector ( 5 downto 0 );

    signal r4_0 : std_logic_vector ( 6 downto 0 );

    signal b5 : std_logic_vector ( 2 downto 0 );
    signal a5 : std_logic_vector ( 2 downto 0 );
    signal r5 : std_logic_vector (2 downto 0);

begin

bxor <= b xor ( add_sub & add_sub & add_sub & add_sub & add_sub & add_sub );

r4_0 <= std_logic_vector ( unsigned( '0' & bxor(4 downto 0) & add_sub ) +  unsigned ( '0' & a(4 downto 0) & add_sub ) );

b5 <= '0' & bxor(5) & r4_0(6);
a5 <= '0' & a(5) & r4_0(6);
r5 <= std_logic_vector ( unsigned( b5 ) +  unsigned ( a5 ) );

r <= r5(1) & r4_0(5 downto 1);
carry_out <= r5(2);
overflow <= r5(2) xor r4_0(6);

end Behavioral;
```

Testbench VHDL

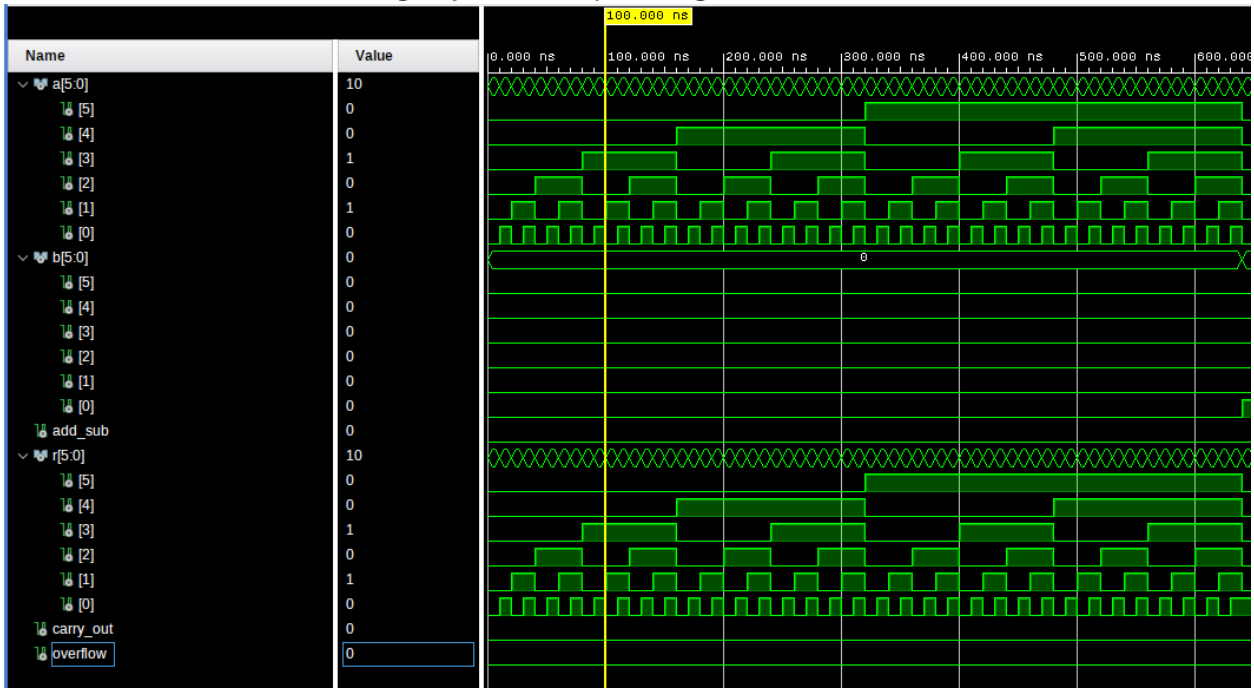
```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;


entity decoder_2x4_testbench is
end decoder_2x4_testbench;

architecture Behavioral of decoder_2x4_testbench is
  signal Y : std_logic_vector ( 3 downto 0 ) := "0000";
  signal A : std_logic_vector ( 1 downto 0 ) := "00";
  signal EN : std_logic := '0';

begin

uut: entity work.decoder_2x4 (decoder)
    port map(
        y => Y,
        a => A,
        en => EN
        );

a(0) <= not a(0) after 10ns;
a(1) <= not a(1) after 20ns;
en <= not en after 40ns;


end Behavioral;
```

Constraint File

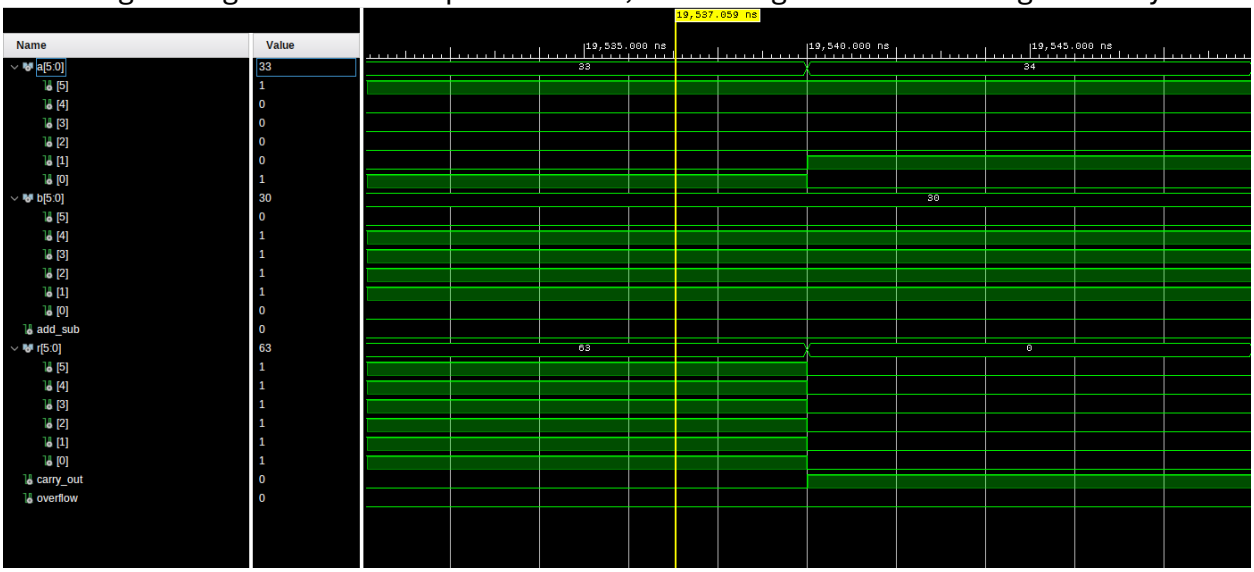
```
##Switches
set_property -dict {PACKAGE_PIN J15 IOSTANDARD LVCMOS33} [get_ports {a[0]}]
set_property -dict {PACKAGE_PIN L16 IOSTANDARD LVCMOS33} [get_ports {a[1]}]
set_property -dict {PACKAGE_PIN M13 IOSTANDARD LVCMOS33} [get_ports {a[2]}]
set_property -dict {PACKAGE_PIN R15 IOSTANDARD LVCMOS33} [get_ports {a[3]}]
set_property -dict {PACKAGE_PIN R17 IOSTANDARD LVCMOS33} [get_ports {a[4]}]
set_property -dict {PACKAGE_PIN T18 IOSTANDARD LVCMOS33} [get_ports {a[5]}]
#set_property -dict {PACKAGE_PIN U18 IOSTANDARD LVCMOS33} [get_ports {}]
set_property -dict {PACKAGE_PIN R13 IOSTANDARD LVCMOS33} [get_ports {add_sub}]
#set_property -dict {PACKAGE_PIN T8 IOSTANDARD LVCMOS18} [get_ports {}]
#set_property -dict {PACKAGE_PIN U8 IOSTANDARD LVCMOS18} [get_ports {}]
set_property -dict {PACKAGE_PIN R16 IOSTANDARD LVCMOS33} [get_ports {b[0]}]
set_property -dict {PACKAGE_PIN T13 IOSTANDARD LVCMOS33} [get_ports {b[1]}]
set_property -dict {PACKAGE_PIN H6 IOSTANDARD LVCMOS33} [get_ports {b[2]}]
set_property -dict {PACKAGE_PIN U12 IOSTANDARD LVCMOS33} [get_ports {b[3]}]
set_property -dict {PACKAGE_PIN U11 IOSTANDARD LVCMOS33} [get_ports {b[4]}]
set_property -dict {PACKAGE_PIN V10 IOSTANDARD LVCMOS33} [get_ports {b[5]}]

## LEDs
set_property -dict {PACKAGE_PIN H17 IOSTANDARD LVCMOS33} [get_ports {r[0]}]
set_property -dict {PACKAGE_PIN K15 IOSTANDARD LVCMOS33} [get_ports {r[1]}]
set_property -dict {PACKAGE_PIN J13 IOSTANDARD LVCMOS33} [get_ports {r[2]}]
set_property -dict {PACKAGE_PIN N14 IOSTANDARD LVCMOS33} [get_ports {r[3]}]
set_property -dict {PACKAGE_PIN R18 IOSTANDARD LVCMOS33} [get_ports {r[4]}]
set_property -dict {PACKAGE_PIN V17 IOSTANDARD LVCMOS33} [get_ports {r[5]}]
#set_property -dict {PACKAGE_PIN U17 IOSTANDARD LVCMOS33} [get_ports {leds[6]}]
#set_property -dict {PACKAGE_PIN U16 IOSTANDARD LVCMOS33} [get_ports {leds[7]}]
#set_property -dict {PACKAGE_PIN V16 IOSTANDARD LVCMOS33} [get_ports {leds[8]}]
#set_property -dict {PACKAGE_PIN T15 IOSTANDARD LVCMOS33} [get_ports {leds[9]}]
#set_property -dict {PACKAGE_PIN U14 IOSTANDARD LVCMOS33} [get_ports {leds[10]}]
#set_property -dict {PACKAGE_PIN T16 IOSTANDARD LVCMOS33} [get_ports {leds[11]}]
#set_property -dict {PACKAGE_PIN V15 IOSTANDARD LVCMOS33} [get_ports {leds[12]}]
#set_property -dict {PACKAGE_PIN V14 IOSTANDARD LVCMOS33} [get_ports {leds[13]}]
set_property -dict {PACKAGE_PIN V12 IOSTANDARD LVCMOS33} [get_ports {carry_out}]
set_property -dict {PACKAGE_PIN V11 IOSTANDARD LVCMOS33} [get_ports {overflow}]
```

-Simulation -

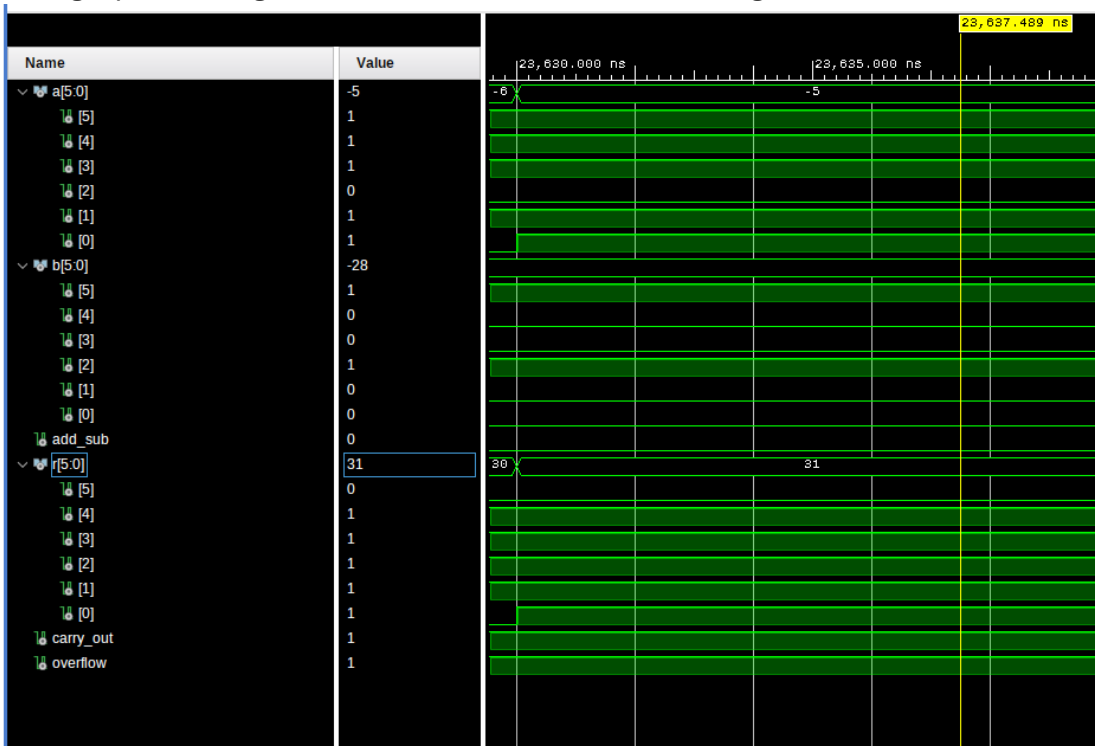
Adding any number plus 0, gives that number



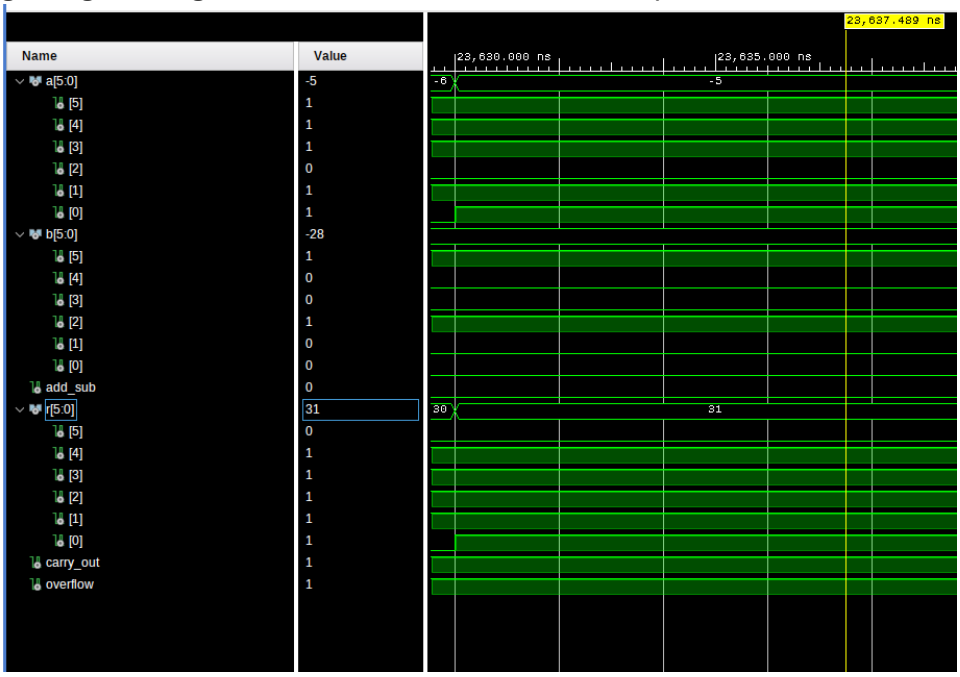
Adding 2 unsigned numbers up to 63 is fine, but adding to 64 and above gives carry out



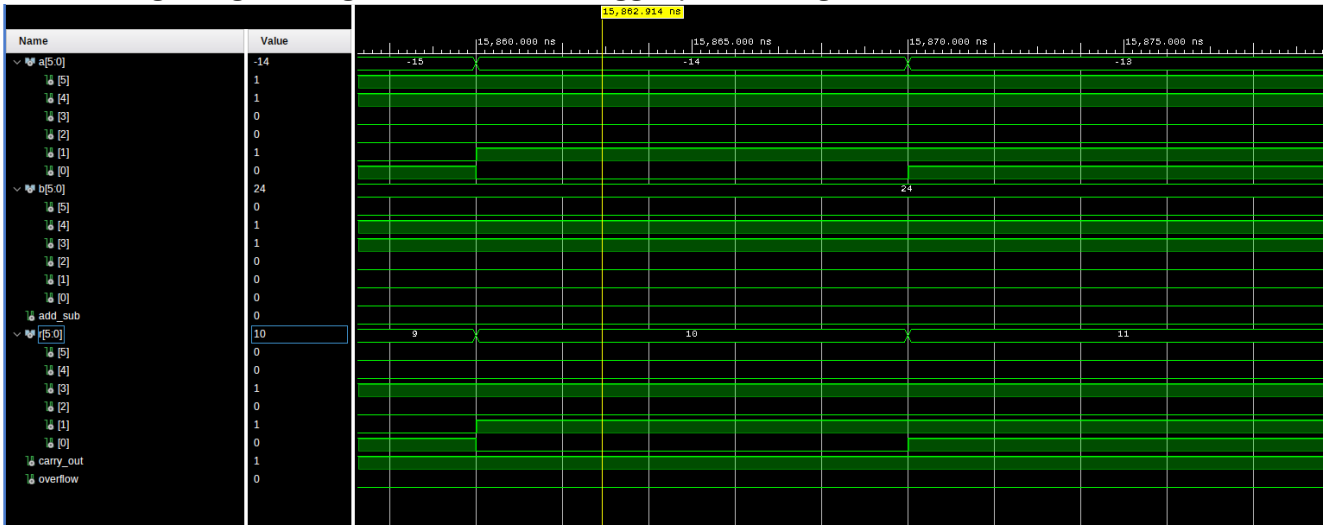
Adding 2 positive signed numbers that overflows to a negative. Overflow asserted.



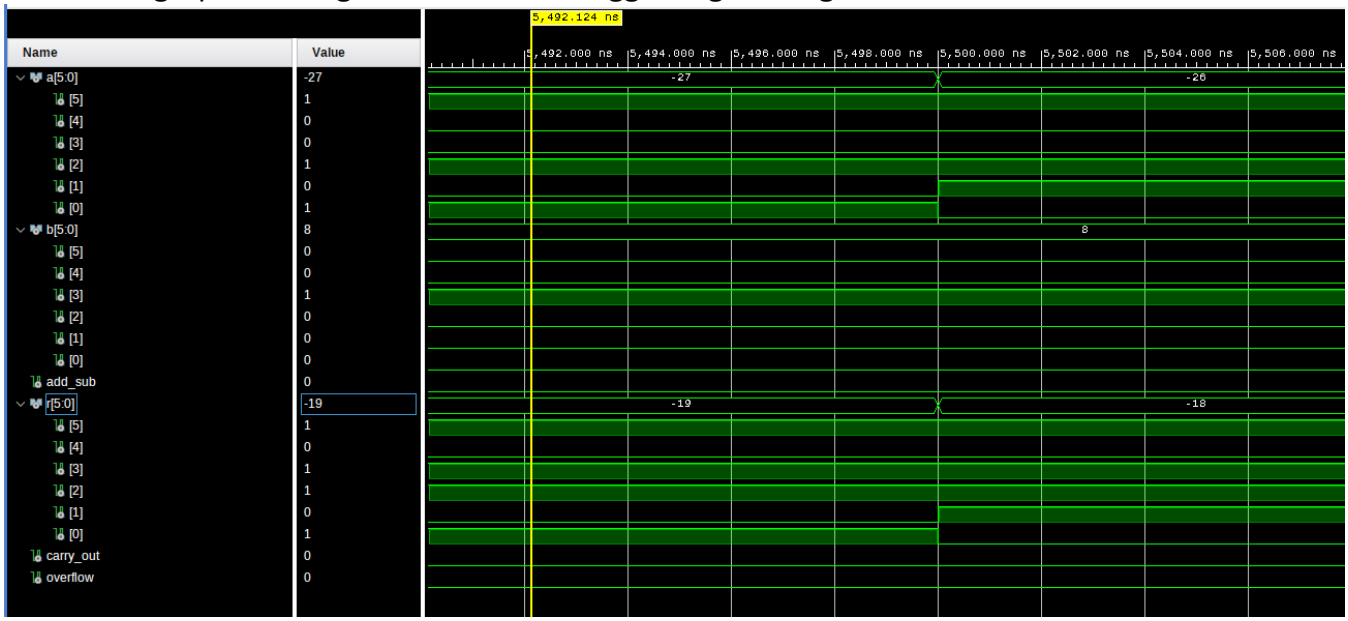
Adding 2 negative signed numbers that overflows to a positive. Overflow asserted.



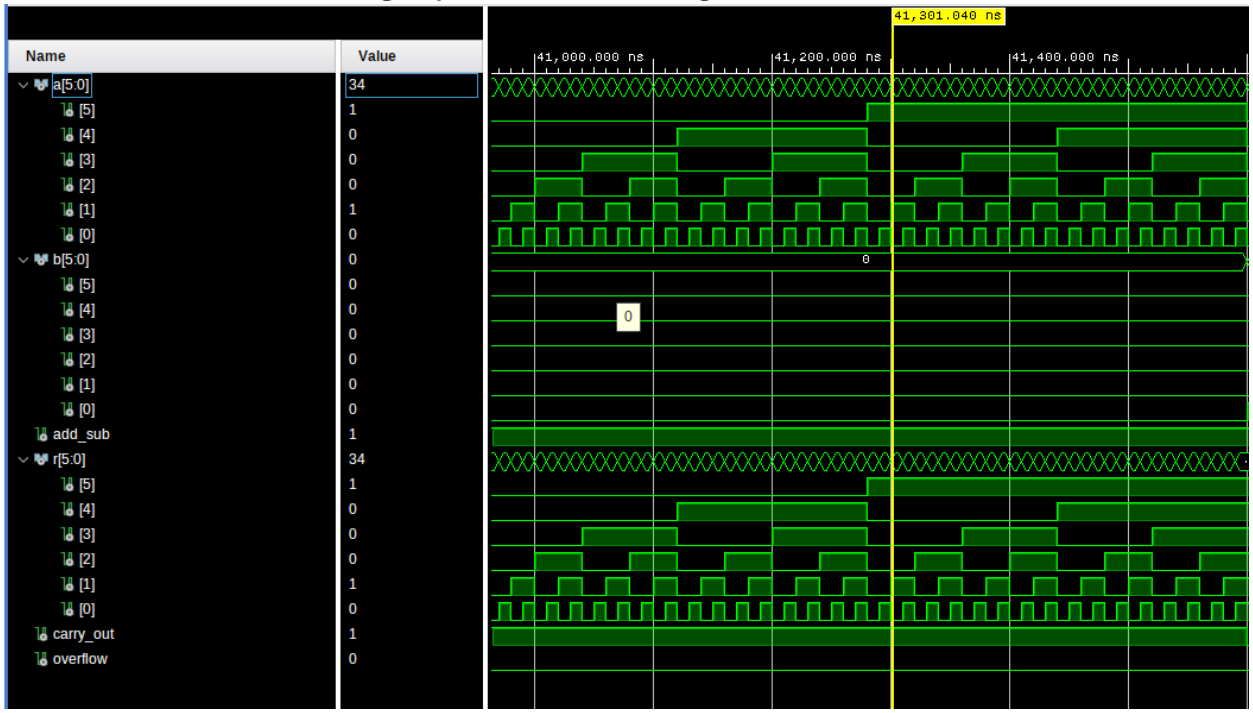
Adding a negative signed number to a bigger positive signed number. No overflow



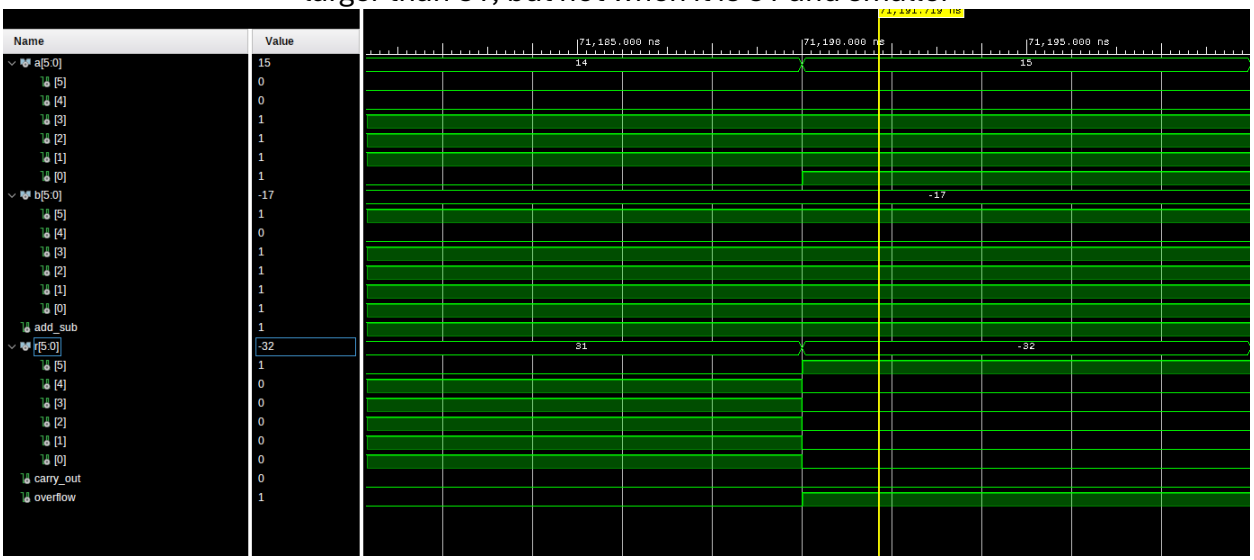
Adding a positive signed number to a bigger negative signed number. No overflow



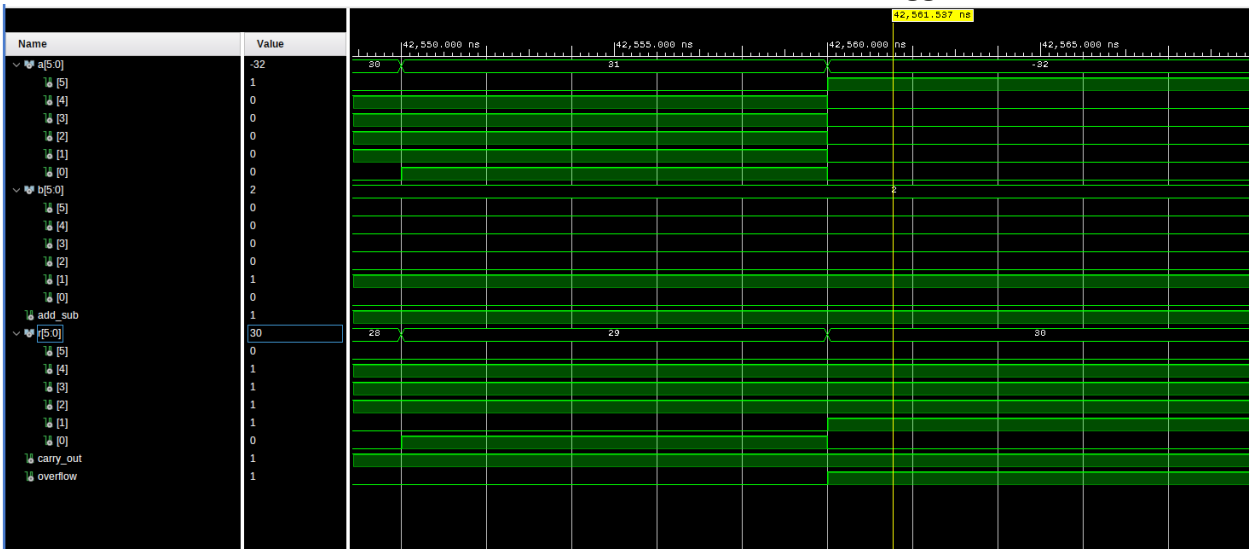
Subtracting any number minus 0, gives that number back



A signed positive number minus a signed negative number overflows when the result is larger than 31, but not when it is 31 and smaller



A signed negative number minus a signed positive number overflows when the result is smaller than -32, but not when it is -32 and bigger



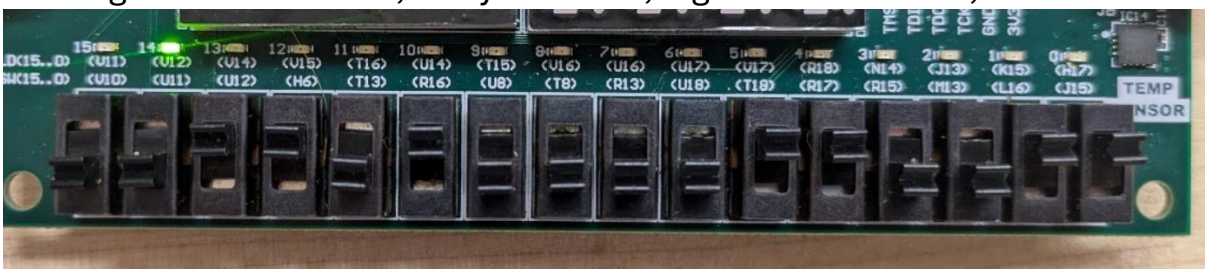
-Implemented Design Verification-

(left 6 switches is B (second term), right 6 switches is A (first term), switch 7 is add_sub)
(right 6 LEDs are result, LED 15 is overflow, LED 14 is carry)

All 0 input, all 0 output

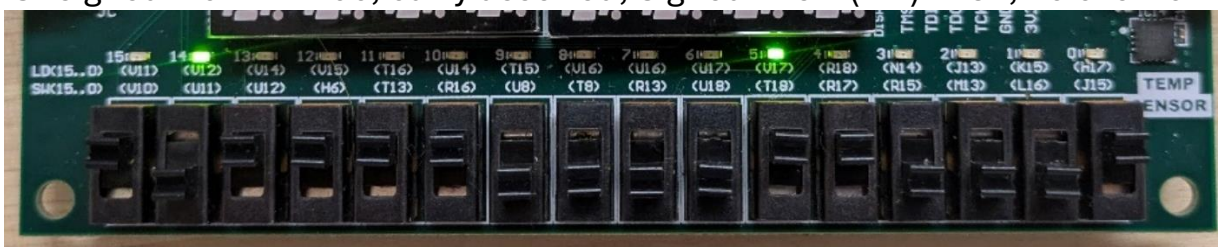
Unsigned: 51 + 12 = 63, no carry; Signed: -13 + 12 = -1, no overflow

Unsigned: 51 + 13 = 64, carry asserted; Signed: -13 + 13 = 0, no overflow



Unsigned: 31 + 1 = 32, no carry; Signed: 31 + 1 = 0, overflow asserted

Unsigned: 49 + 47 = 96, carry asserted; Signed: -15 + (-17) = -32, no overflow

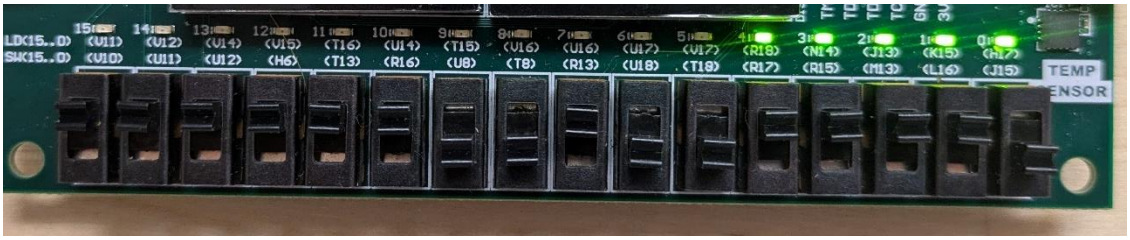


Unsigned: : 48 + 47 = 95, carry asserted; Signed: -16 + (-17) = -33, overflow asserted

0 – 0 = 0, no overflow

4 – 7 = -3, no overflow

30 – (-1) = 31, no overflow



31 – (-1) = 32, overflow asserted

-Conclusion-

In conclusion, the lab was successfully completed with all objectives met. The 6-bit adder/subtractor was designed and simulated correctly, demonstrating accurate addition and subtraction with carry-out and two's-complement overflow detection. T mapping to the Nexys A7 FPGA board was successfully implemented. All functional tests, including the use of switches and the LEDs for output, confirmed the correctness of the design. The project was completed without issues, and the results aligned perfectly with the expected behavior.