ELEC-311
Project 3
Adder-Subtractor

Charles Pittman

October 29, 2013

1 Objective

- Design a combinational logic circuit that adds, or subtracts, two 4-bit 2's complement numbers
- Describe the circuit using VHDL

2 Discussion

A diagram of a 4-bit adder/subtractor circuit is shown in Fig 1. The adder is identical to the type previously studied: given two 4-bit numbers as input, X and Y, output their 5-bit sum. Subtraction is functionally identical to adding a negative number, so the adder can be converted to a subtractor simply by allowing one input to be negative.

The encoding given for negative numbers is two's-complement, formed by complementing (negating) the binary number and adding 1. A third input is made, SEL, to produce the negative of the second input, Y, to make the circuit a subtractor. When SEL is 1, the XOR gates produce the complement of Y, and C_{in} of the circuit adds 1 to form the two's-complement.

The entire circuit was then implemented in VHDL (See Section 3), using a library consisting of the logic XOR gates and full-adders. After programming the FPGA, the circuit's operation was verified using the values in Table 1.

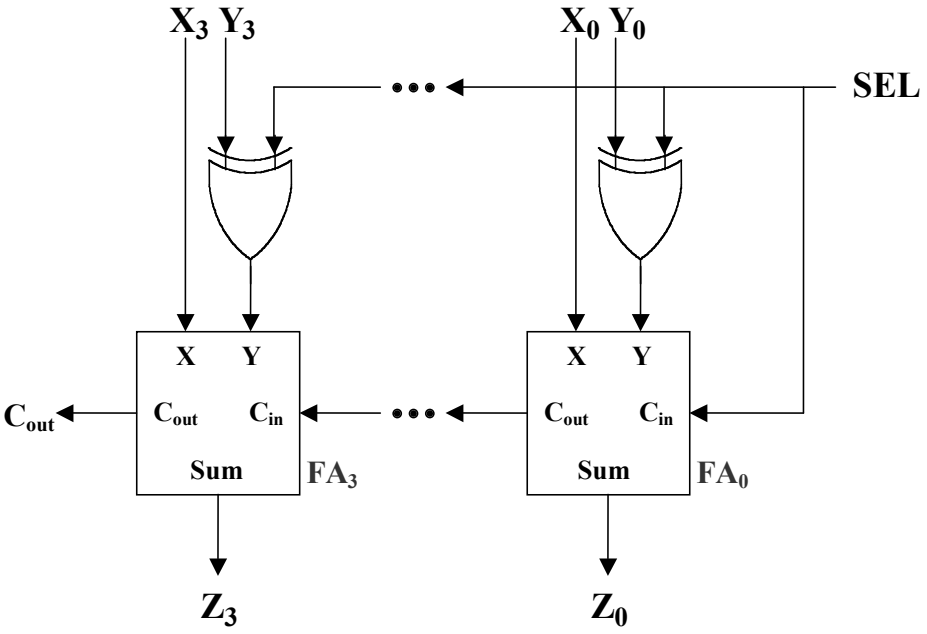

Figure 1: 4-bit adder/subtractor.

X	Y	X	Y	X + Y	X - Y
3	1	0011	0001	0100	0010
1	5	0001	0101	0110	1100
-2	3	1110	0011	0001	1011
-2	-4	1110	1100	1010	0010
4	6	0100	0110	1010	1110

Table 1: Test vectors and expected values.

3 VHDL

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
use work.project3_gates.all;

entity circuit is
  port (X    : in  std_logic_vector(3 downto 0);
        Y    : in  std_logic_vector(3 downto 0);
        Z    : out std_logic_vector(3 downto 0);
        SEL  : in  std_logic;
        Cout : out std_logic;
        OVR  : out std_logic);
end circuit;

architecture Structural of circuit is

  signal C : std_logic_vector(3 downto 0);
  signal S : std_logic_vector(3 downto 0);

begin

  x3 : ExclusiveOR port map(SEL, Y(3), S(3));
  x2 : ExclusiveOR port map(SEL, Y(2), S(2));
  x1 : ExclusiveOR port map(SEL, Y(1), S(1));
  x0 : ExclusiveOR port map(SEL, Y(0), S(0));

  fa3 : FullAdder port map(X(3), S(3), C(2), C(3), Z(3));
  fa2 : FullAdder port map(X(2), S(2), C(1), C(2), Z(2));
  fa1 : FullAdder port map(X(1), S(1), C(0), C(1), Z(1));
  fa0 : FullAdder port map(X(0), S(0), SEL, C(0), Z(0));

  Cout <= C(3);
  OVR <= (((not X(3)) and (not Y(3)) and C(3)) or (X(3) and Y(3) and (not C(3))));

end Structural;
```