Exp. 7: Iterative Circuits – Adders and Subtractors

Engineering 357 - Digital Design Lab

Fall 2015

Farnam Adelkhani

Date Experiment Performed: 10-1-2015

Date Report Submitted: 10-2-2015

Farnam Adelkhani
Engineering 357 - Digital Design Lab – 10/1/15
Experiment #7

Exp. 7: Iterative Circuits – Adders and Subtractors

Abstract and Objective:

This lab is an introduction to adders and subractors in circuit logic design and more specifically how to implement and test them with Xilinx software. The lab also introduces iterative circuits and its related design techniques. Typical word lengths are in the powers of 2 and so one typically needs to design the circuit for one bit and then imply interconnect multiple copies of that same adder circuit to form a complete circuit for the word, this approach is known as iterative design. Of course, making copies of a circuit is easy when CAD software is being utilized, the experimenter will utilize it to chain several full adders together in order to create a ripple-carry adder.

This lab is also about the design of basic binary add and/or subtract circuits. An adder/subtractor circuit can be implemented from an adder circuit by the inclusion of an add/sub control signal and a XOR gate (1s complementer). Inputs will be used that can be easily verified mathematically to confirm that the circuit is functioning as intended.

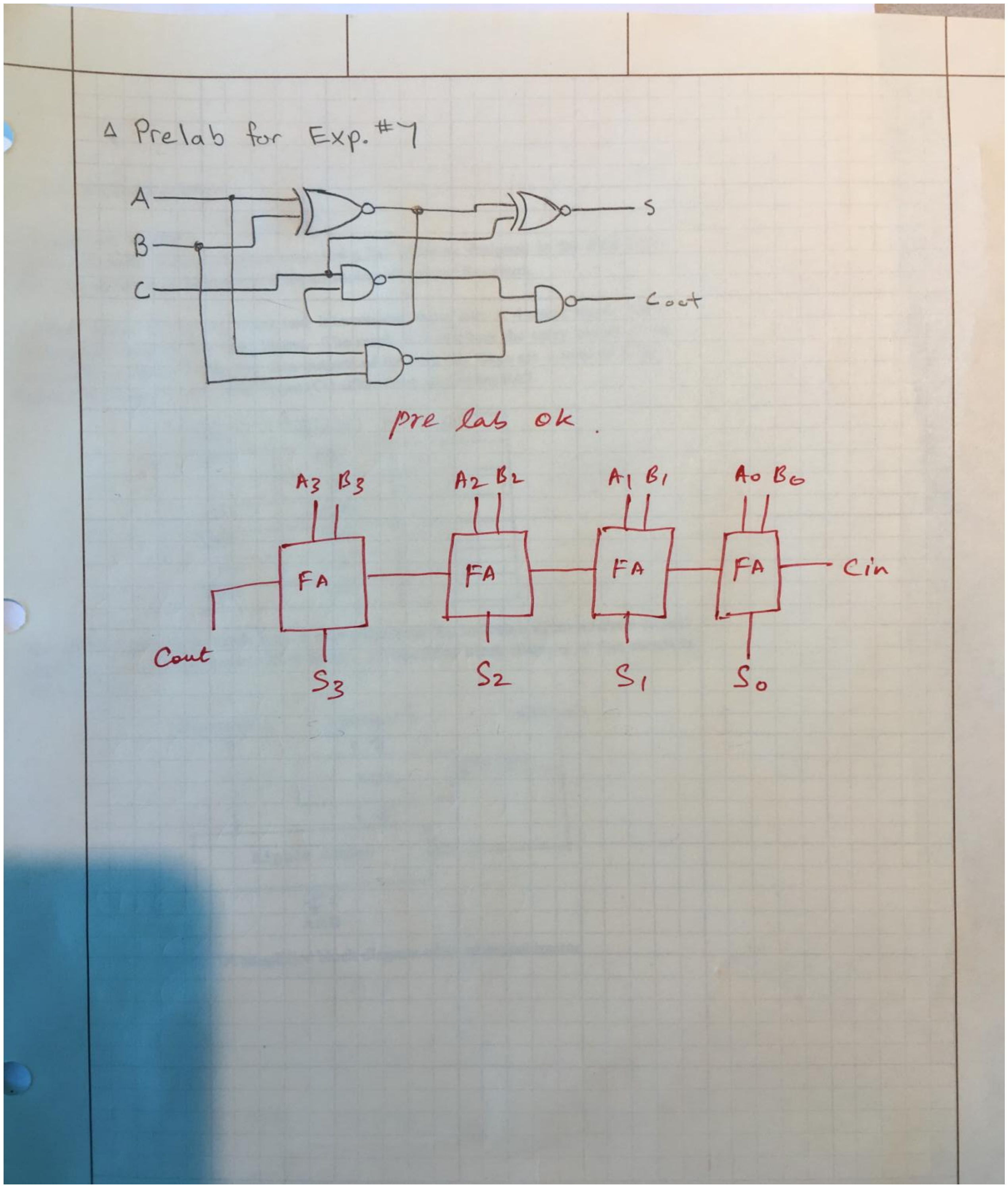
Components Used:

- Xilinx software on PC
- The following gates/integrated circuits:

➢ 3 - nand2	quad 2-input NAND
➢ 3 – XOR2	quad XOR
➢ Other components as needed	

Procedures and result:

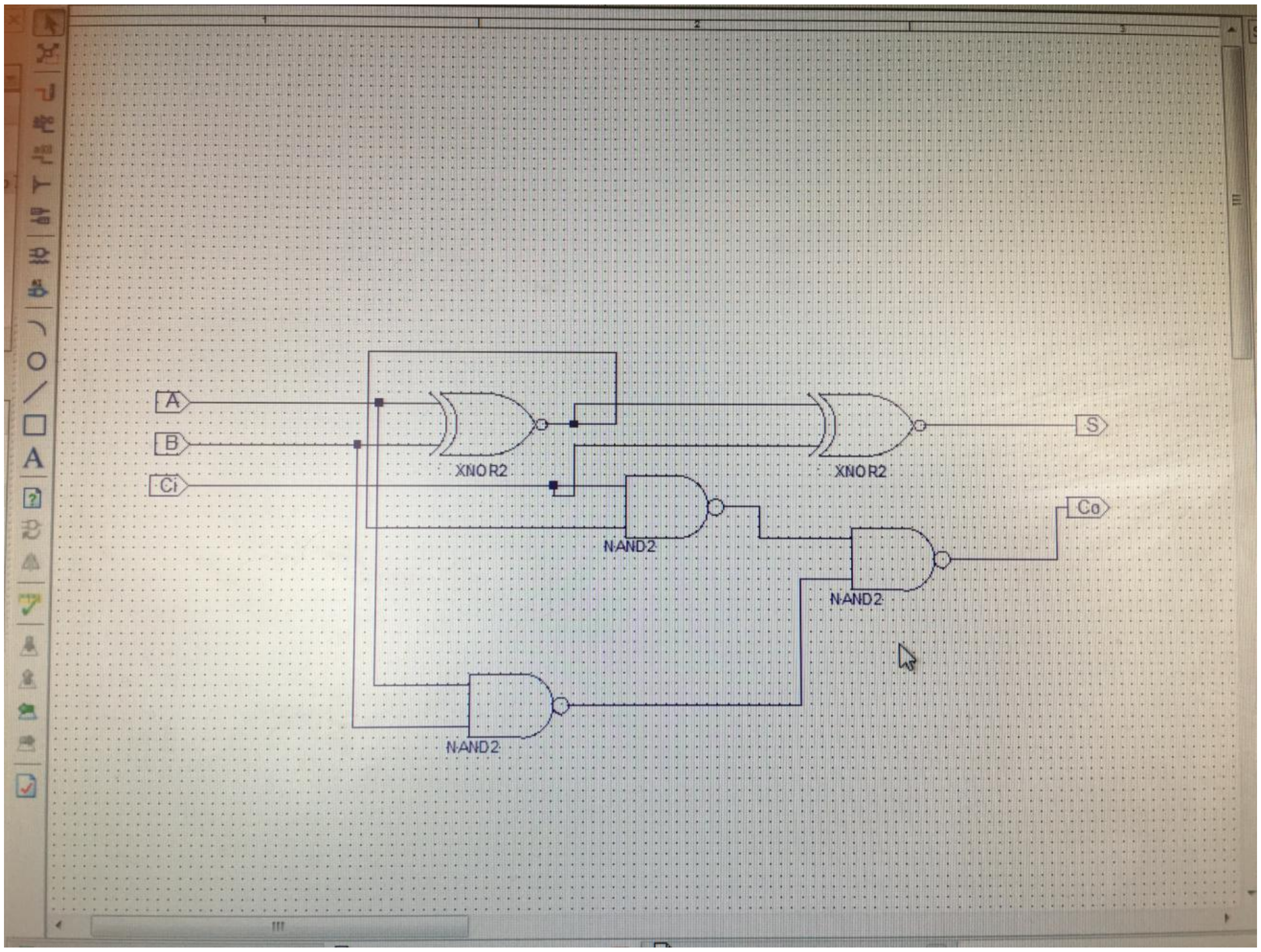
1. The first step is to complete the pre-lab assignment before the lab class. The prelab assignment in this case required the experimenters to design a full adder circuit using as few logic gates as possible. The circuit that was designed is shown on the next page…



Our full adder was approved (see arrow above).

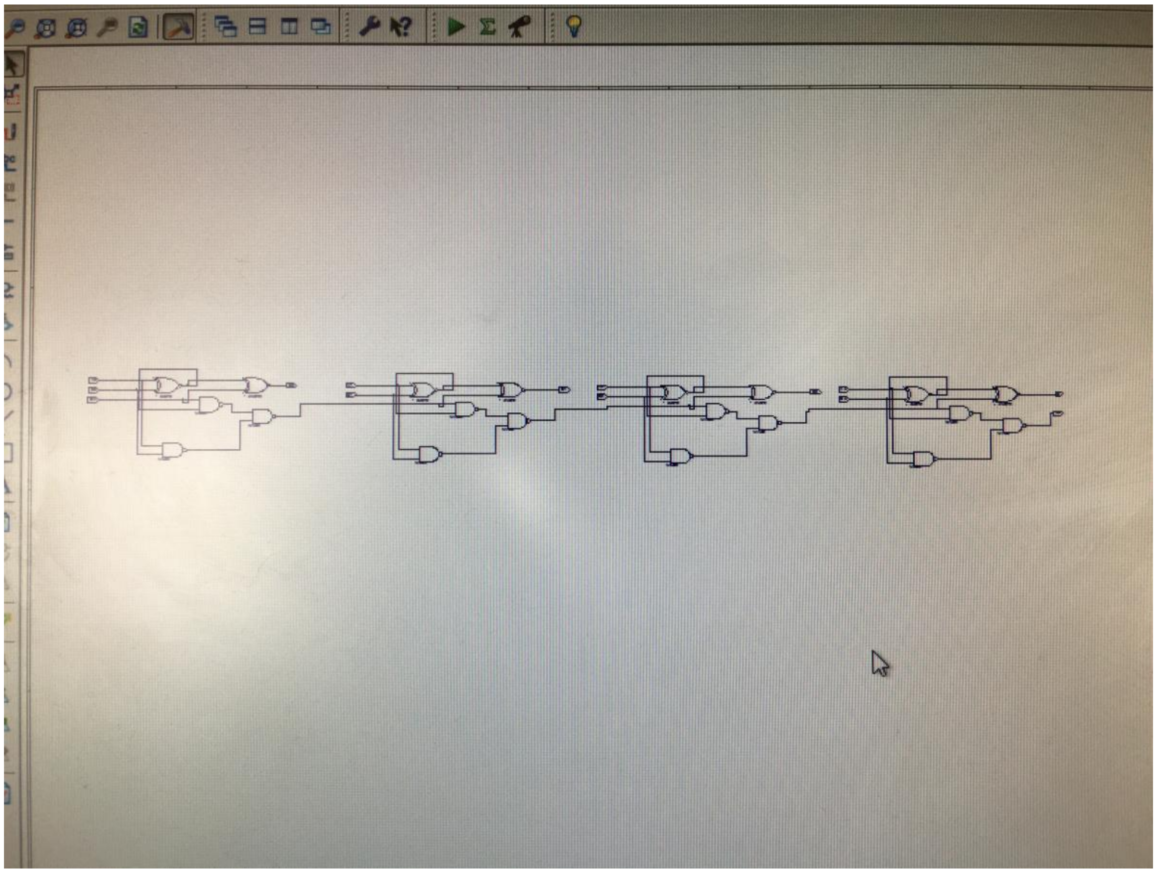
2. Next step is to construct and implement the full adder circuit that has been designed during the prelab, in Xilinx. Which just requires creating a project and

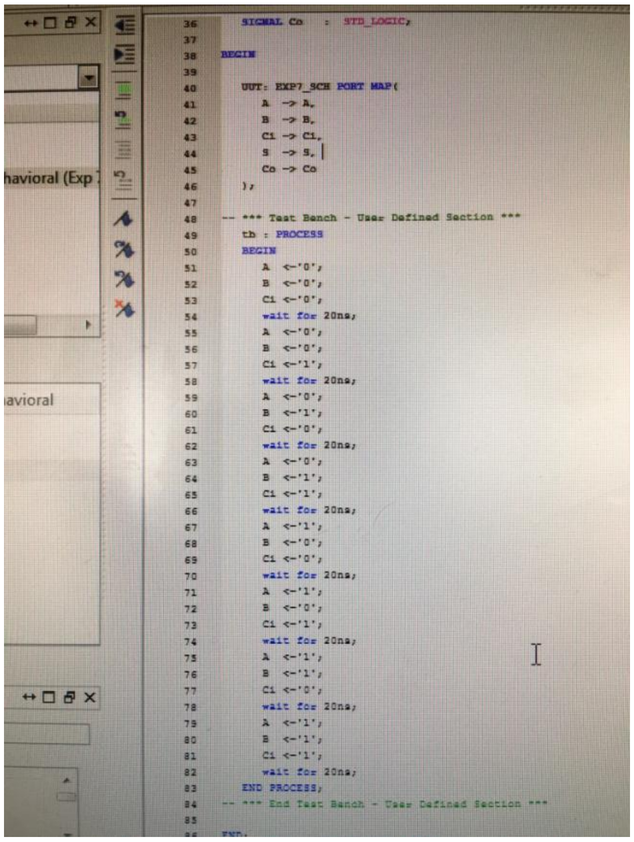
placing the components and wires on the given area. Our adder circuit in Xilinx is pictured below:



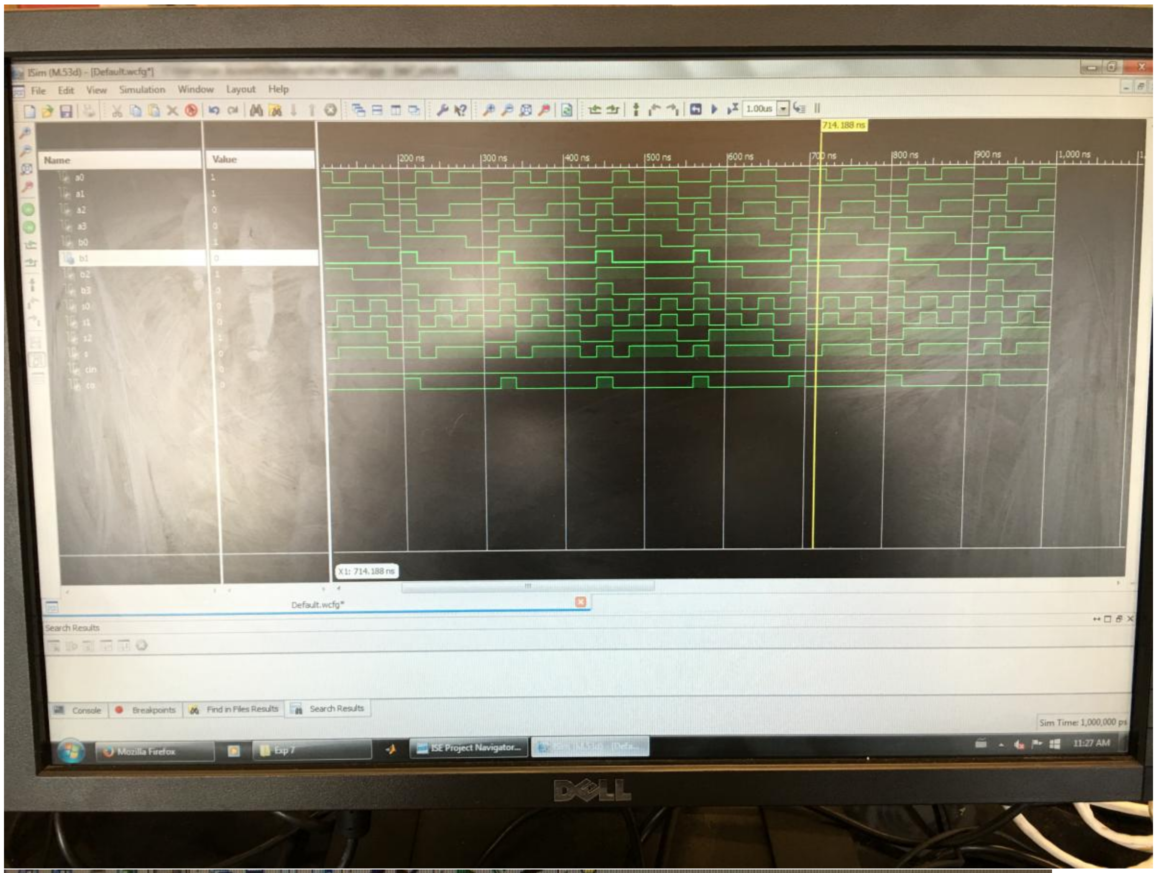
The above circuit was simulated and found to be functional.

3. The next step in this lab is to make four copies of the full adder and interconnect them into a 4-stage ripple adder. The experimenters will test the ripple adder by experimentally completing a table where the inputs can easily be used to confirm the given output in Xilinx is correct. The multi stage ripple adder is shown on the next page in Xilinx:

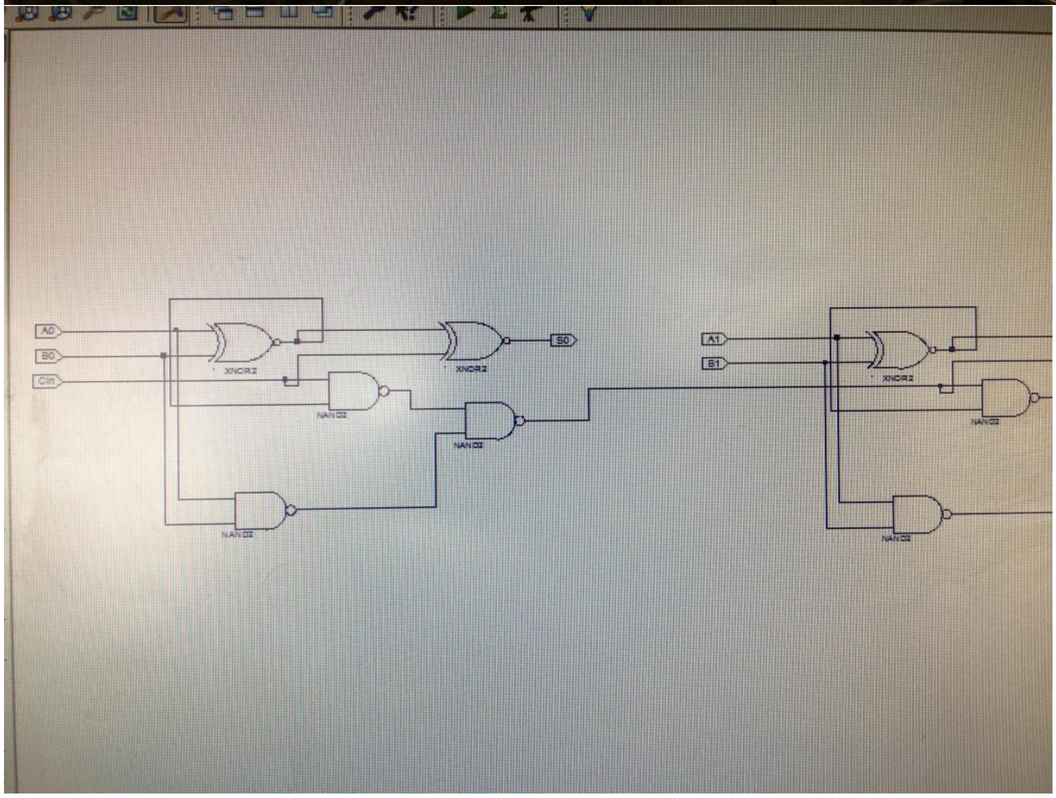




```
    SIGNAL Co   :  STD_LOGIC;

BEGIN

    UUT: EXP7_SCH PORT MAP(
       A  -> A,
       B  -> B,
       Ci -> Ci,
       S  -> S,
       Co -> Co
    );

-- *** Test Bench - User Defined Section ***
    tb : PROCESS
    BEGIN
       A  <-'0';
       B  <-'0';
       Ci <-'0';
       wait for 20ns;
       A  <-'0';
       B  <-'0';
       Ci <-'1';
       wait for 20ns;
       A  <-'0';
       B  <-'1';
       Ci <-'0';
       wait for 20ns;
       A  <-'0';
       B  <-'1';
       Ci <-'1';
       wait for 20ns;
       A  <-'1';
       B  <-'0';
       Ci <-'0';
       wait for 20ns;
       A  <-'1';
       B  <-'0';
       Ci <-'1';
       wait for 20ns;
       A  <-'1';
       B  <-'1';
       Ci <-'0';
       wait for 20ns;
       A  <-'1';
       B  <-'1';
       Ci <-'1';
       wait for 20ns;
    END PROCESS;
-- *** End Test Bench - User Defined Section ***

END;
```



Name
Value
a0
a1
a2
a3
b0
b1
b2
b3
s0
s1
s2
s3
cin
co
714.188 ns
X1: 714.188 ns
Default.wcfg*
Search Results
Console
Breakpoints
Find in Files Results
Search Results
Sim Time: 1,000,000 ps



A0
B0
Cin
XNOR2
XNOR2
S0
NAND2
NAND2
NAND2
A1
B1
XNOR2
NAND2
NAND2

V. LABORATORY WORK

1. Using the schematic capture software construct a full adder as designed in the PRE-LAB. Simulate the circuit to make sure that it is performing all desired functions.

2. Make four copies of the full adder and interconnect them into a 4-stage ripple adder. Experimentally complete the table below. The result is formed by the carry output of the most significant stage and the four sum outputs of the FAs (so there are a total of 5 bits). What should be applied to the carry input (C_0) of the least significant FA?

operand 1	operand 2	result
1010	0101	1111
1111	0001	(1) 0000
1111	0000	1111
0000	0000	0000
1100	0111	(1) 0011
0011	0101	0100

111
1111
0001
10000

111
0011
0101
1000

1100
0111
10011

3. The adder circuit can be made into an adder/subtractor by inclusion of an add/sub control signal and a 1's complementer (XOR gates). A simplified block diagram of this circuit is shown below:

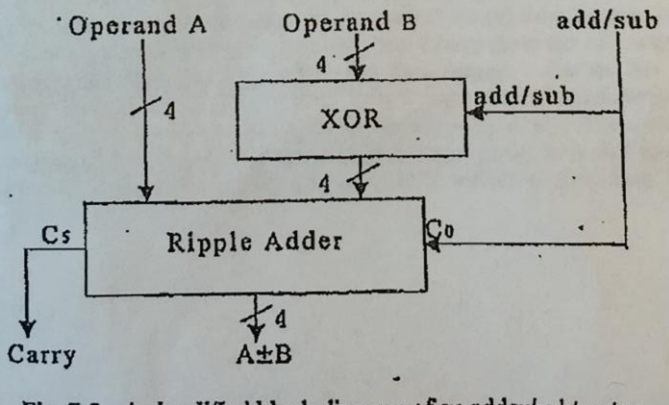
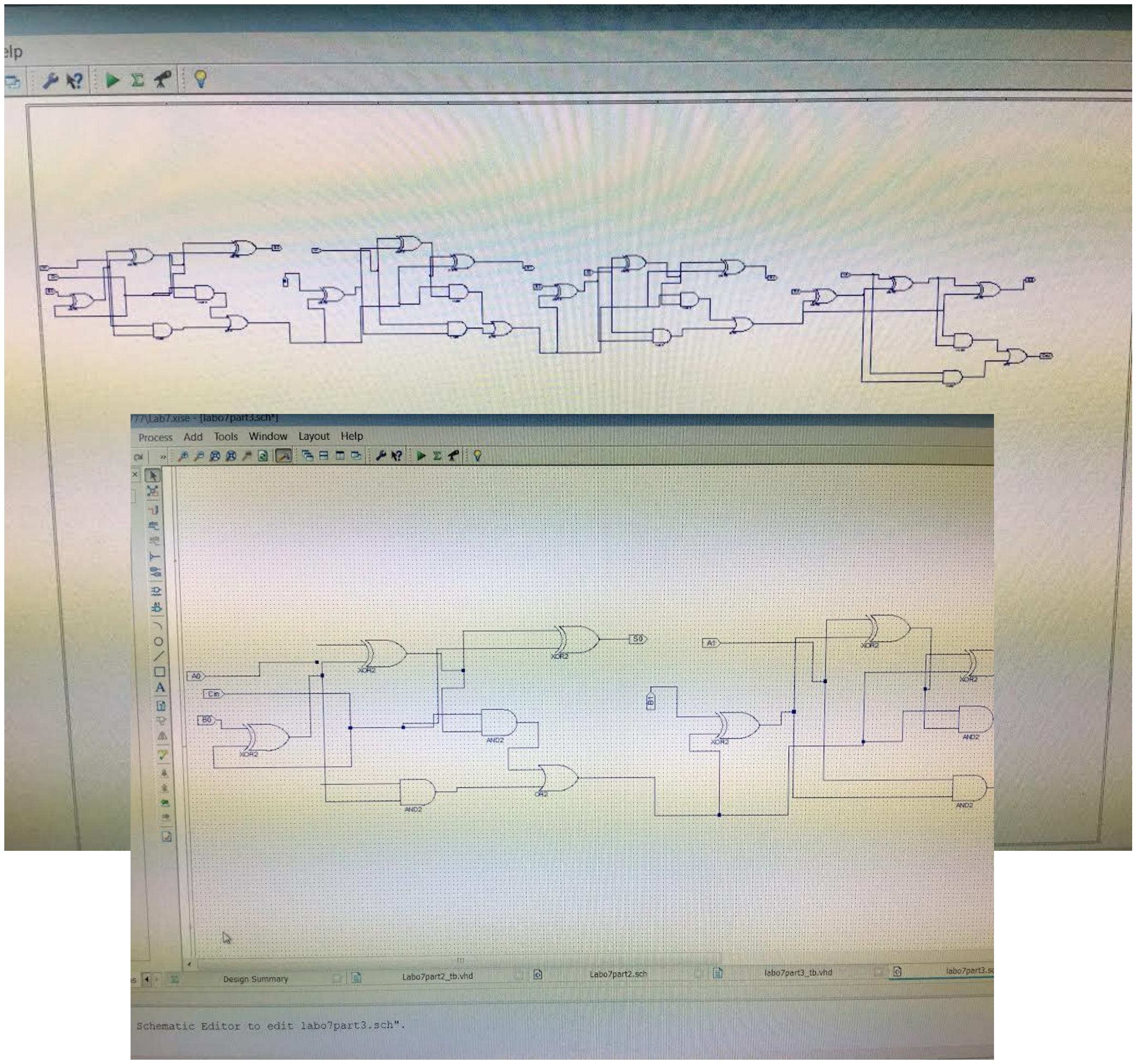
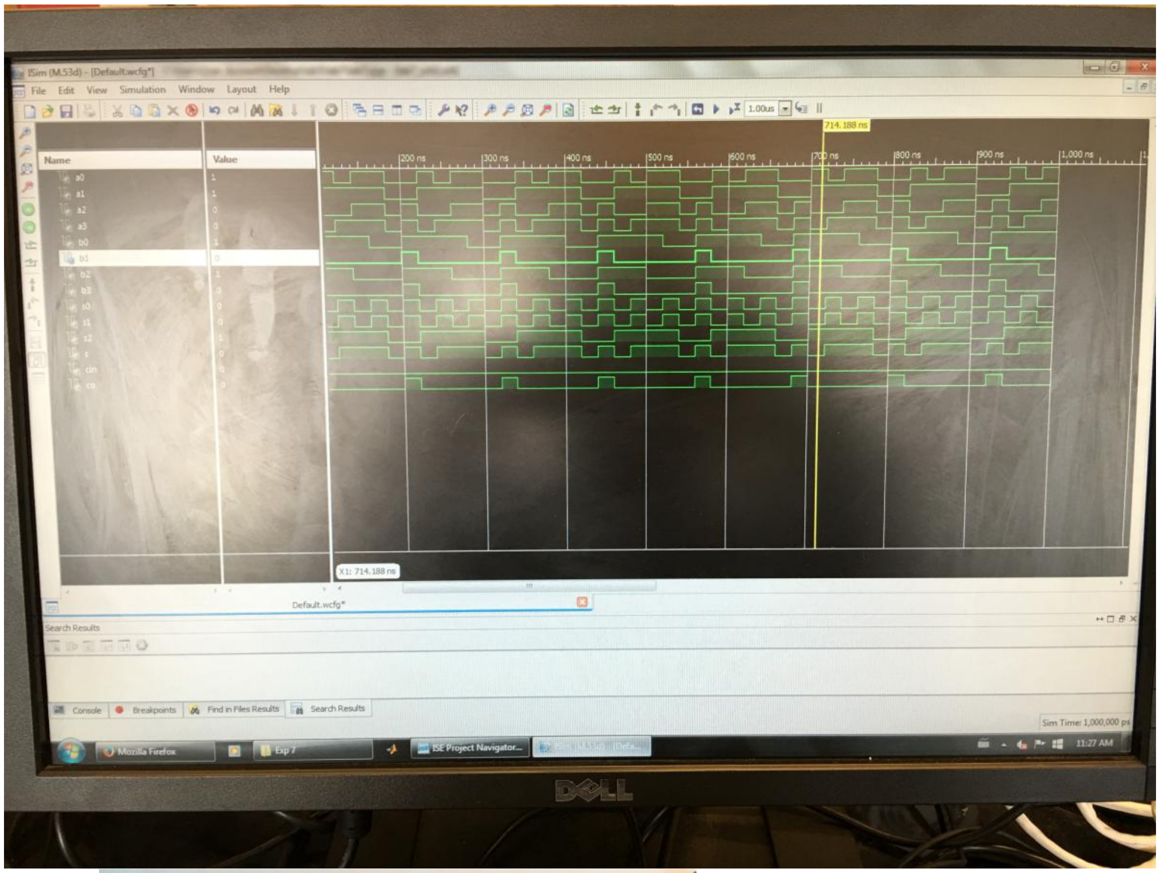


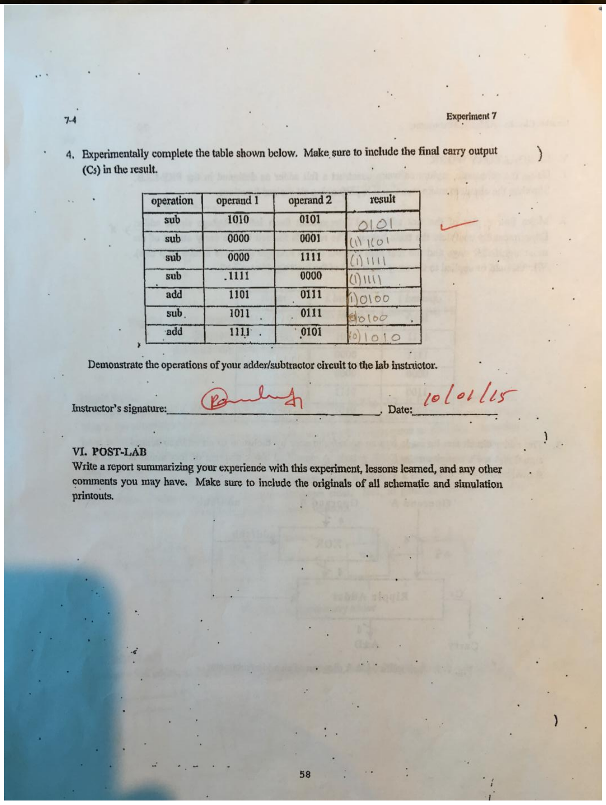
Fig. 7-3. A simplified block diagram of an adder/subtractor

4. The adder circuit can also be made into an adder/subtractor by inclusion of an add/sub control signal and a XOR gate. This circuit was designed and then implementing using the full adder ripple schematics that had already been drawn up. The user implements an XOR gate at each B input with only the first carry bit also feeding into the XOR gate. Every gate thereafter will have the same original carry in value while using the next series of B values. The diagram designed is shown below:



```
Schematic Editor to edit labo7part3.sch".
```





4. Experimentally complete the table shown below. Make sure to include the final carry output (C_3) in the result.

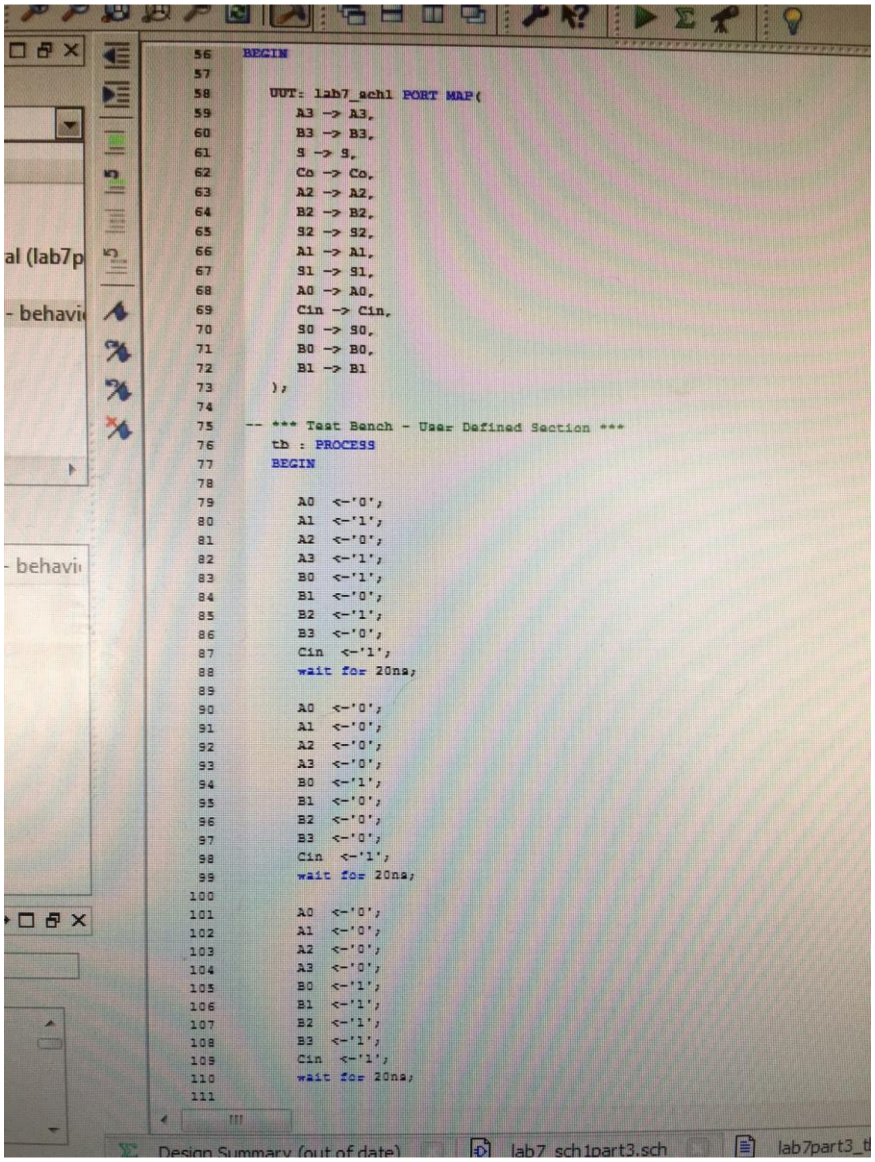
operation	operand 1	operand 2	result
sub	1010	0101	0101 ✓
sub	0000	0001	(1) 1101
sub	0000	1111	(1) 1111
sub	.1111	0000	(1) 1111
add	1101	0111	(1) 0100
sub	1011	0111	[illegible] 0100
add	1111	0101	(0) 1010

Demonstrate the operations of your adder/subtractor circuit to the lab instructor.

Instructor's signature: [illegible] Date: 10/01/15

VI. POST-LAB

Write a report summarizing your experience with this experiment, lessons learned, and any other comments you may have. Make sure to include the originals of all schematic and simulation printouts.



```
BEGIN

   UUT: lab7_sch1 PORT MAP(
      A3 => A3,
      B3 => B3,
      S => S,
      Co => Co,
      A2 => A2,
      B2 => B2,
      S2 => S2,
      A1 => A1,
      S1 => S1,
      A0 => A0,
      Cin => Cin,
      S0 => S0,
      B0 => B0,
      B1 => B1
   );

-- *** Test Bench - User Defined Section ***
   tb : PROCESS
   BEGIN

      A0  <='0';
      A1  <='1';
      A2  <='0';
      A3  <='1';
      B0  <='1';
      B1  <='0';
      B2  <='1';
      B3  <='0';
      Cin  <='1';
      wait for 20ns;

      A0  <='0';
      A1  <='0';
      A2  <='0';
      A3  <='0';
      B0  <='1';
      B1  <='0';
      B2  <='0';
      B3  <='0';
      Cin  <='1';
      wait for 20ns;

      A0  <='0';
      A1  <='0';
      A2  <='0';
      A3  <='0';
      B0  <='1';
      B1  <='1';
      B2  <='1';
      B3  <='1';
      Cin  <='1';
      wait for 20ns;

```

Conclusion and Summary:

Now the experimenters have gained familiarity with adder circuit and ripple circuit design while also learning to implement our circuit to perform as an adder/subtractor. It is useful to be given data points that can be confirmed by hand in order to confirm our system is working as expected. There are often issues that arise in the Xilinx software itself. You might assume a 10 gb program is free of bugs but not so by any means. Often the best fix is to save the circuit and reload it. I do have some concern that these software issues with Xilinx may become overbearing as a problem once the circuit is much more complex. It is important and vital that the user are gaining familiarity with the Xilinx design suite.