Brian Reber

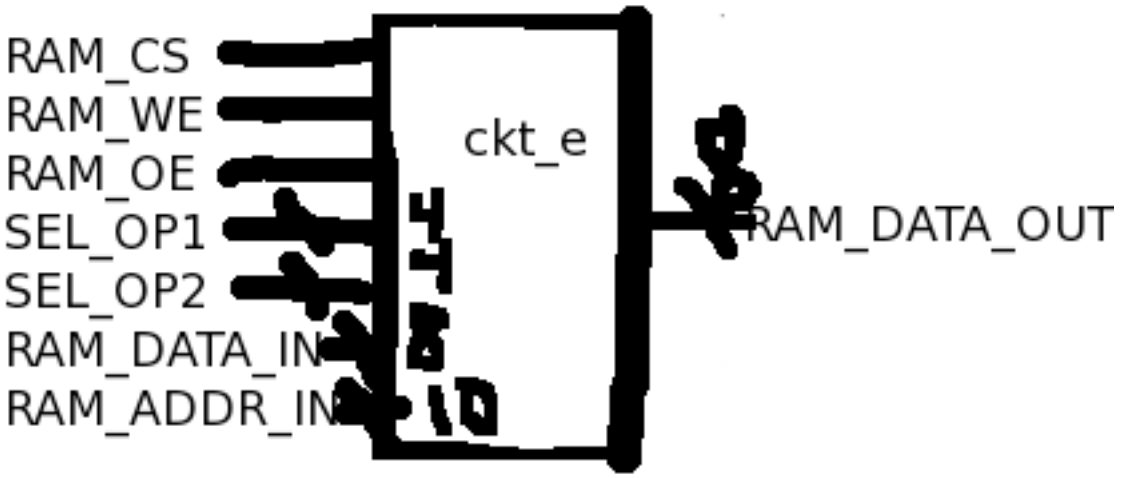
CprE 381

Lab 02

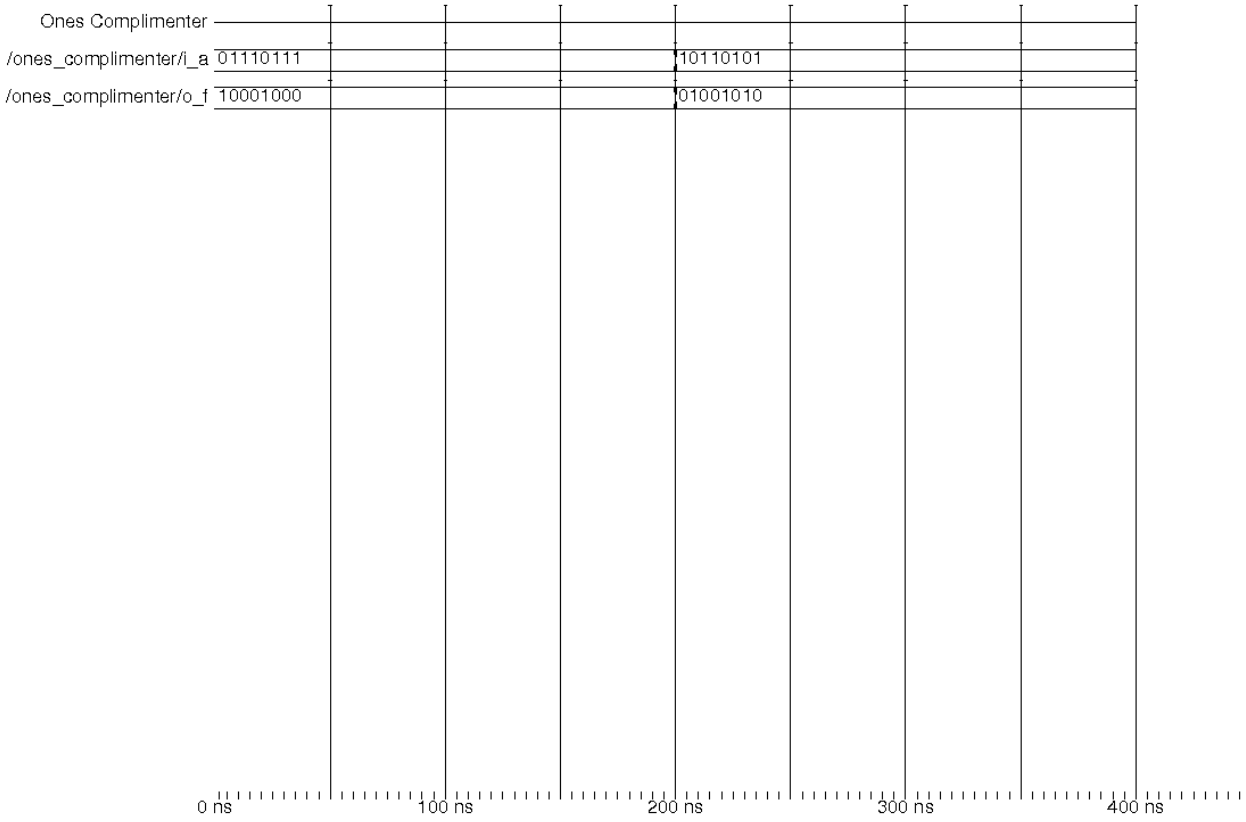
Prelab:

5e)

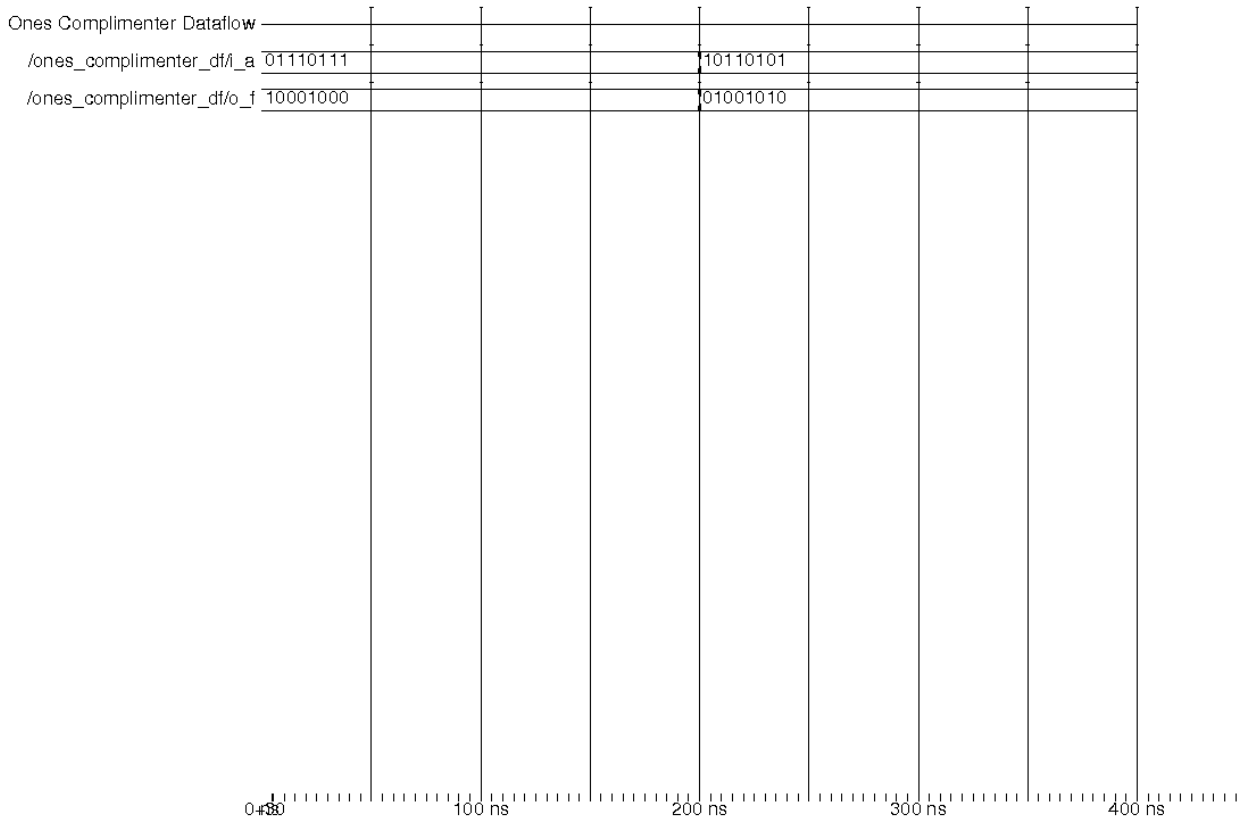
```
entity loader is
     port (big_wank : in std_logic_vector(5 downto 0);
          net_tok  : in std_logic;
          tw_cnt   : in std_logic;
          clk      : in std_logic;
          assert   : out std_logic;
          data     : out std_logic_vector(9 downto 0);
          addr     : out std_logic_vector(7 downto 0);
          spud_out : out std_logic);
end loader;
```



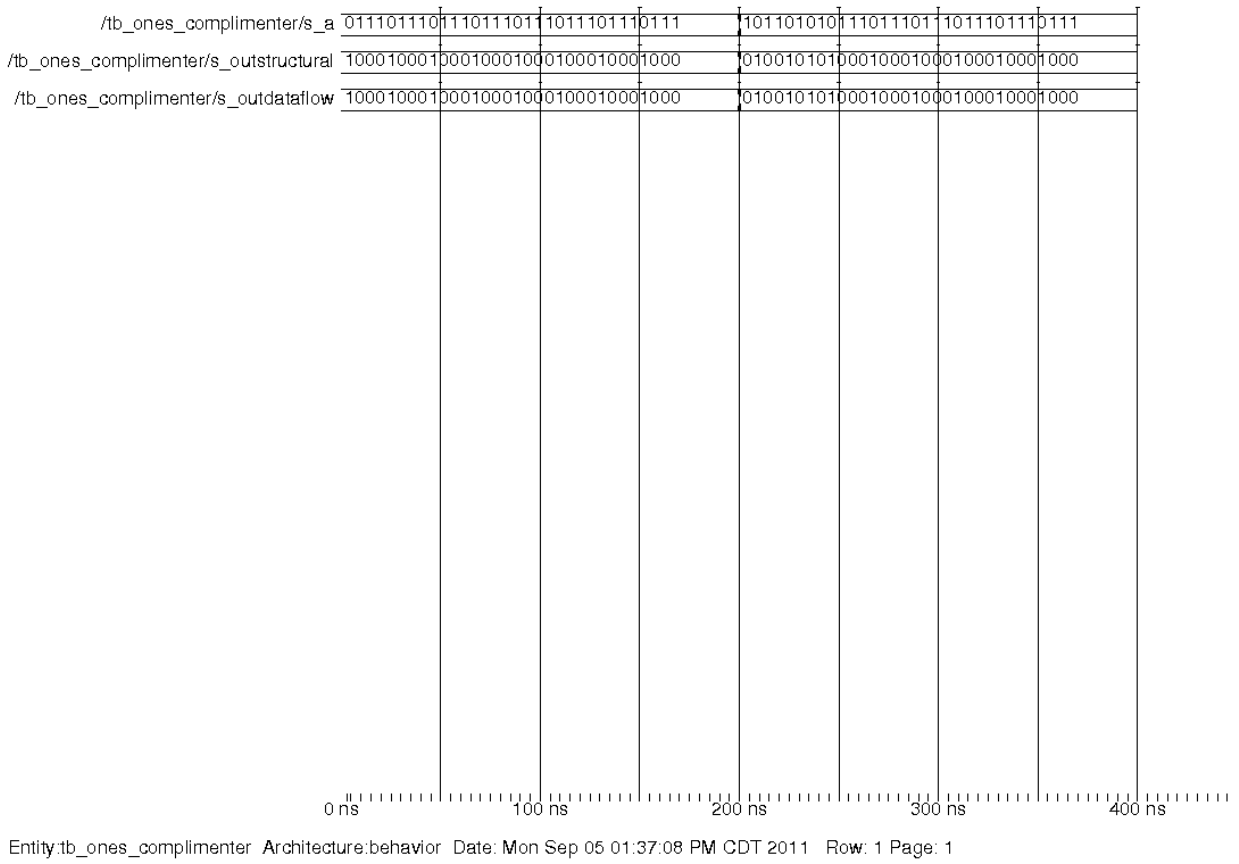
6e)
7) In the entity declaration in part a, there is a missing semicolon at the end of the clk line, and there is also a missing semicolon at the end of the port declaration. In part b, there is a missing closing parenthesis and semicolon for the port declaration.



Entity:ones_complimenter Architecture:structure Date: Wed Aug 31 02:45:52 PM CDT 2011 Row: 1 Page: 1



Entity:ones_complimenter_df Architecture:dataflow Date: Wed Aug 31 02:46:11 PM CDT 2011 Row: 1 Page: 1

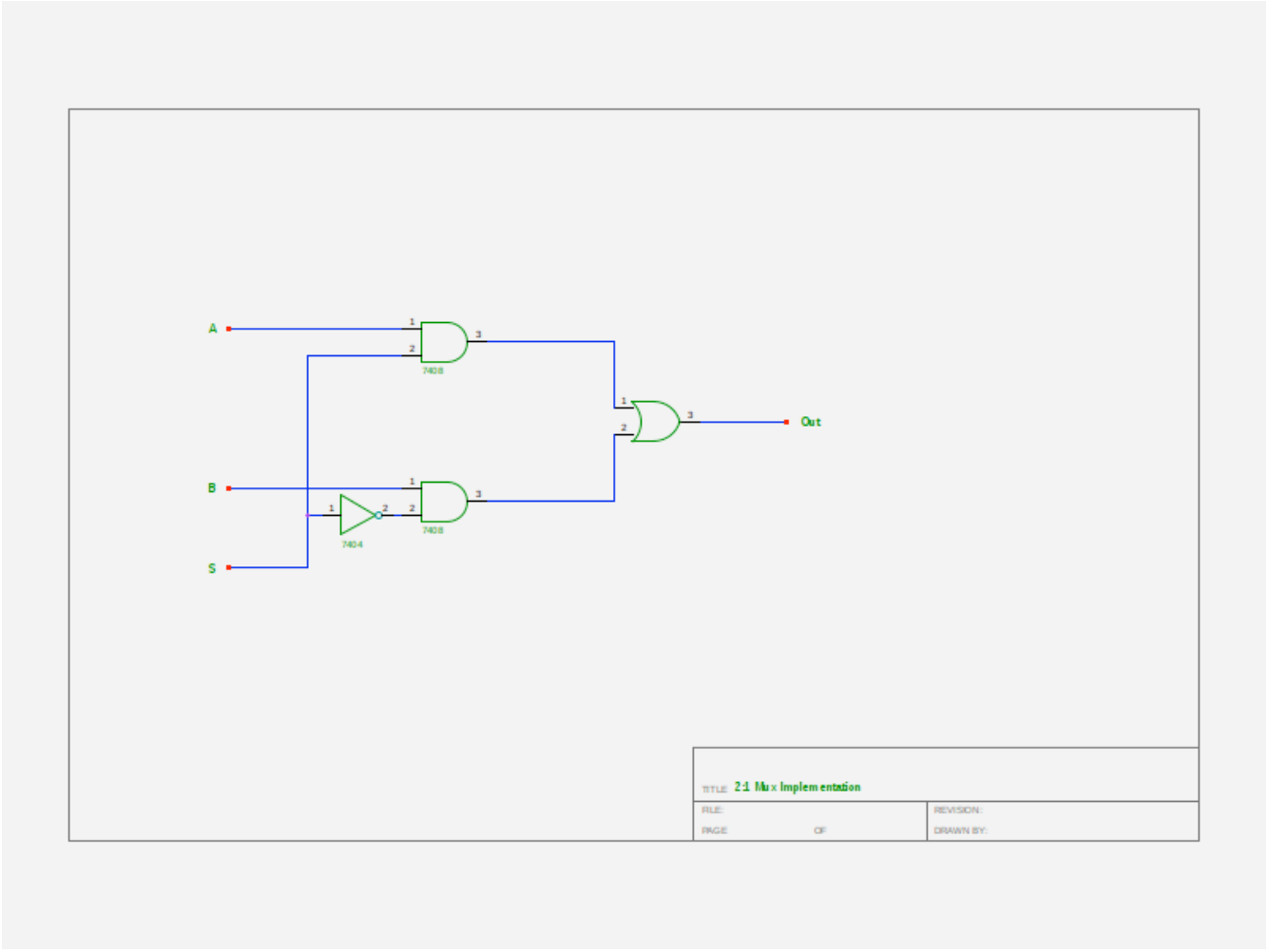


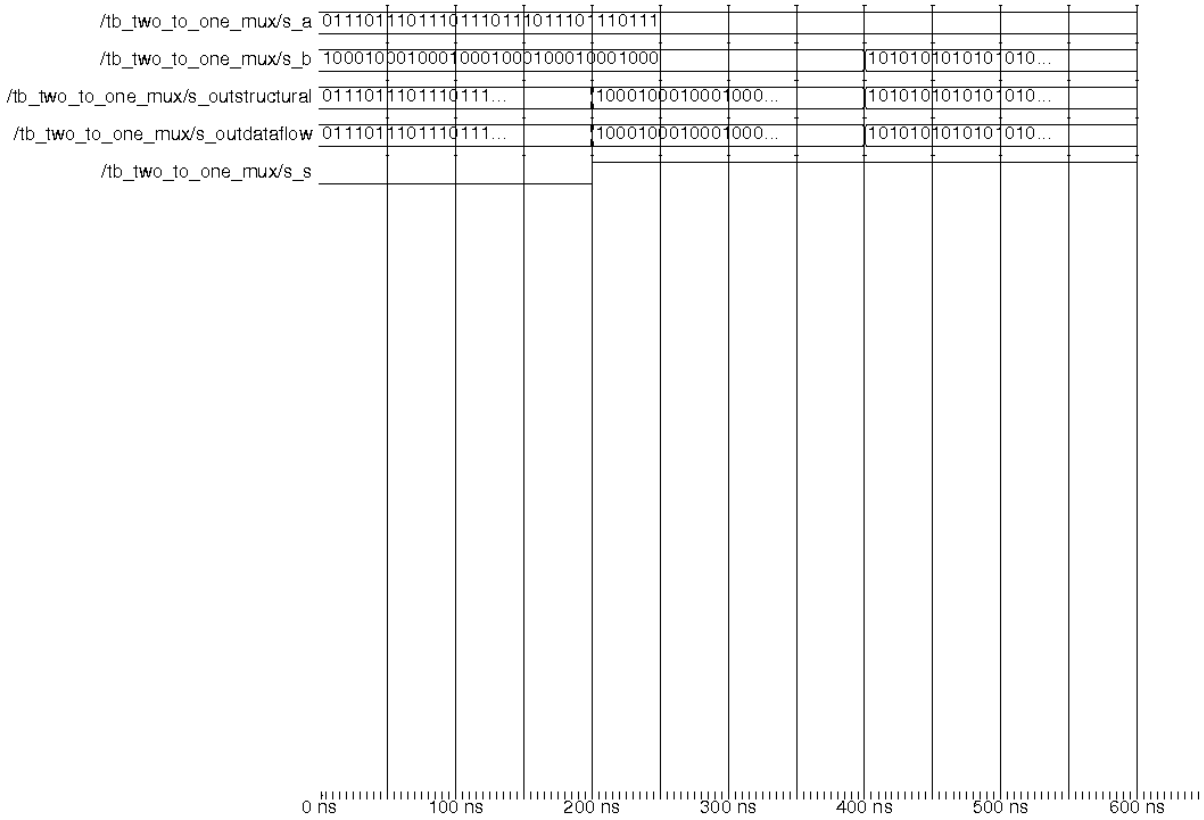
These are a collection of waveforms for the one's complimenter. From looking at the waveforms, you can tell that it is behaving as expected (it is properly inverting bits).

Two-Input Mux

S	A	B	Out
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Expression: S * A + ~S * B



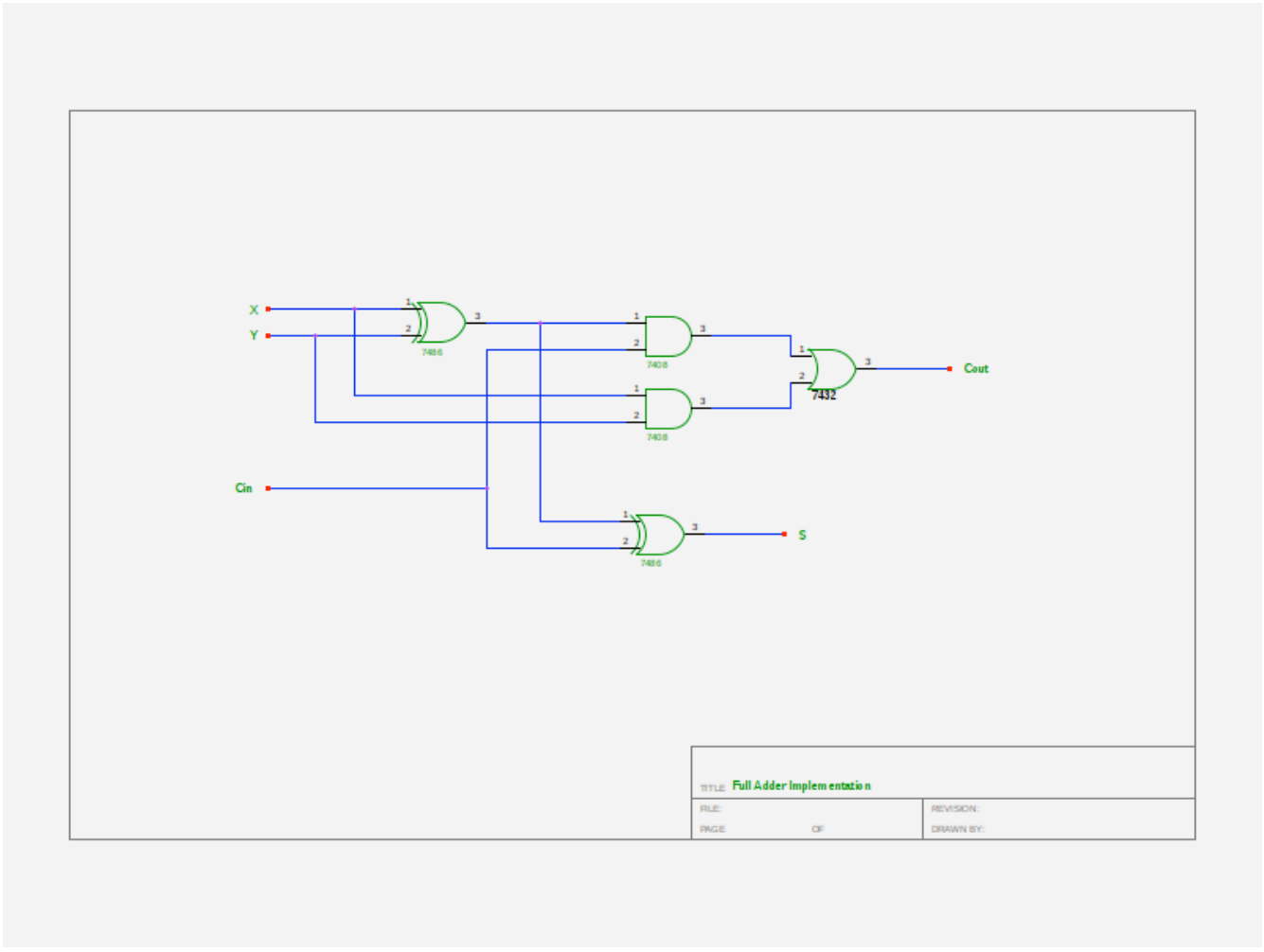


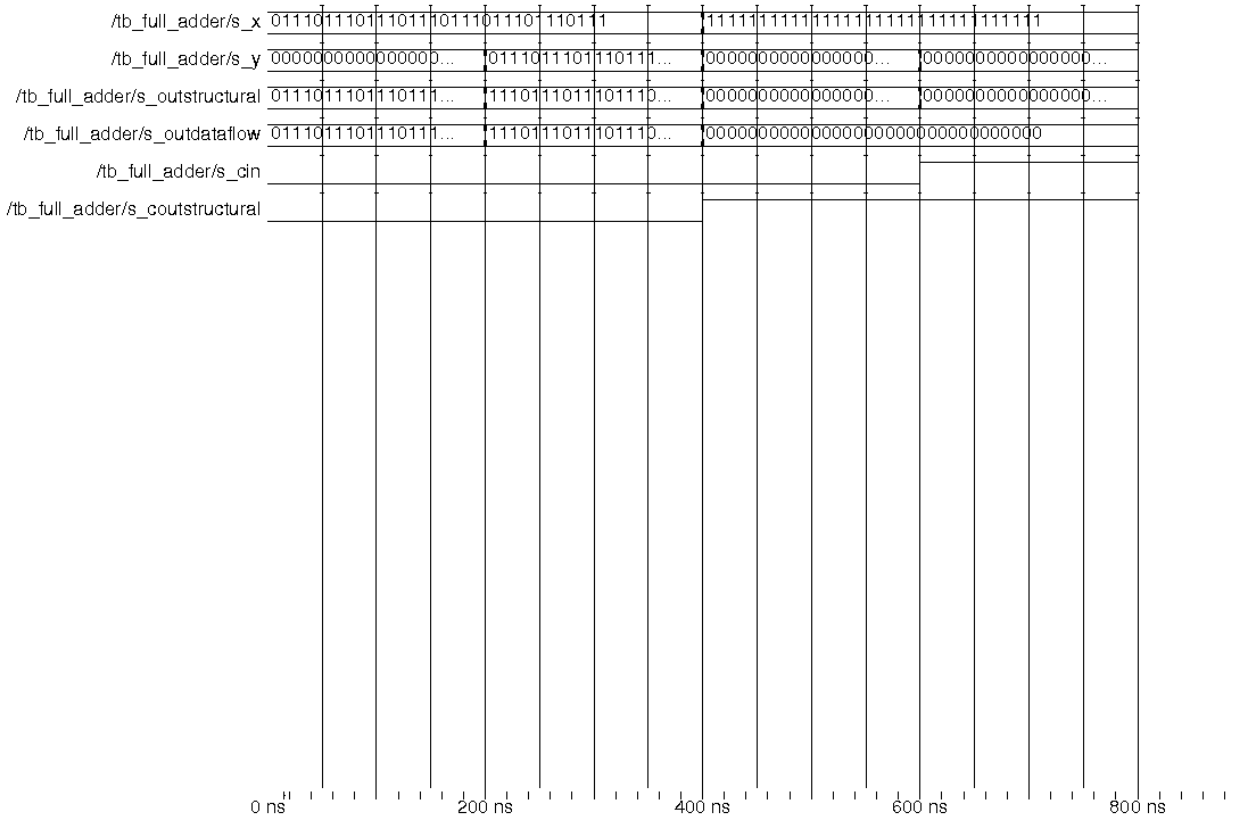
Entity:tb_two_to_one_mux Architecture:behavior Date: Mon Sep 05 03:51:31 PM CDT 2011 Row: 1 Page: 1

Full Adder

X	Y	Cin	Cout	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

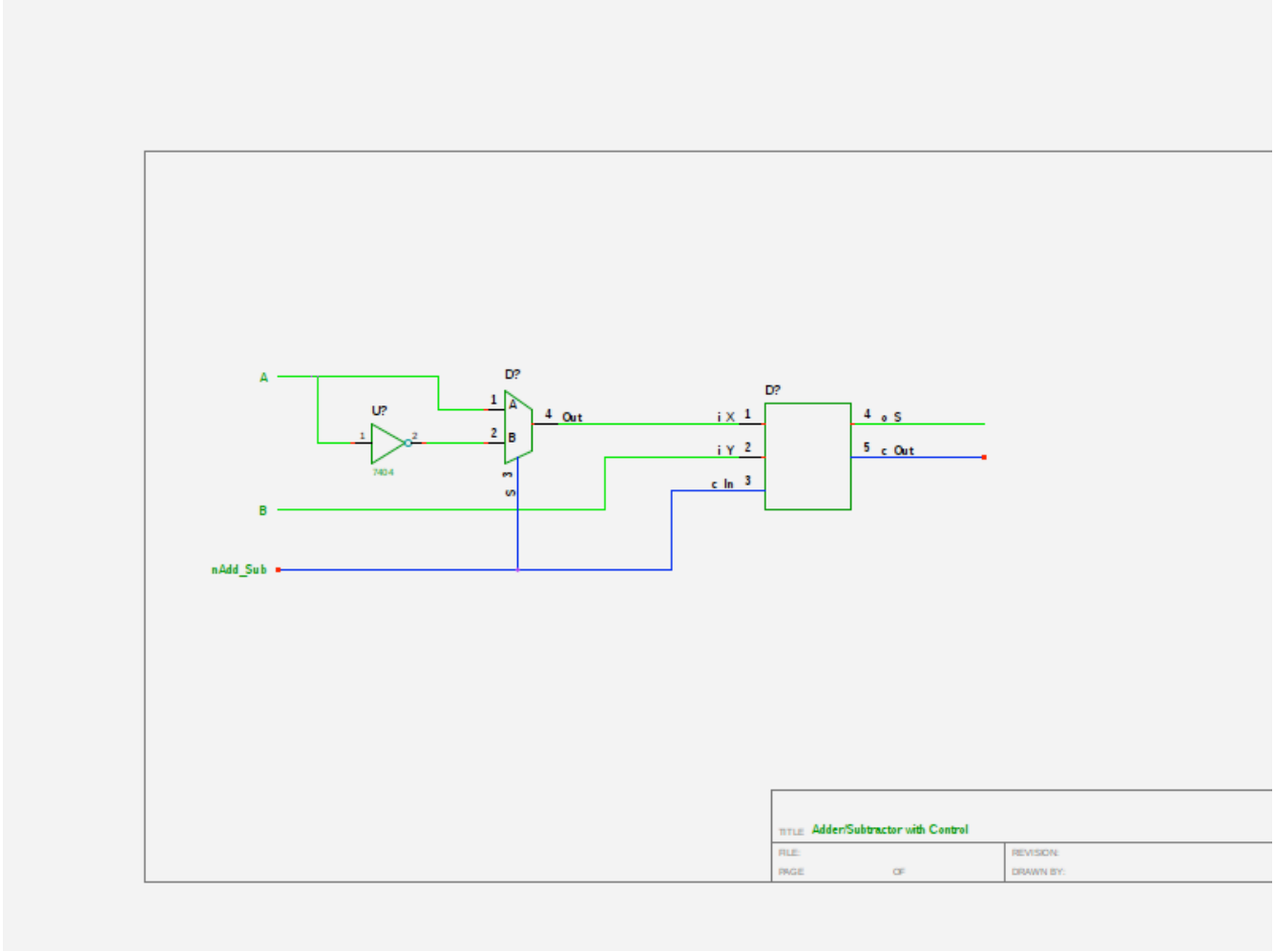
Expression: **S** = ~X * ~Y * C + ~X * Y * ~C + X * ~Y * ~C + X * Y * C = **X xor Y xor C**
Cout = Y * C + X * C + X * Y = **(Y xor X) * C + X * Y**



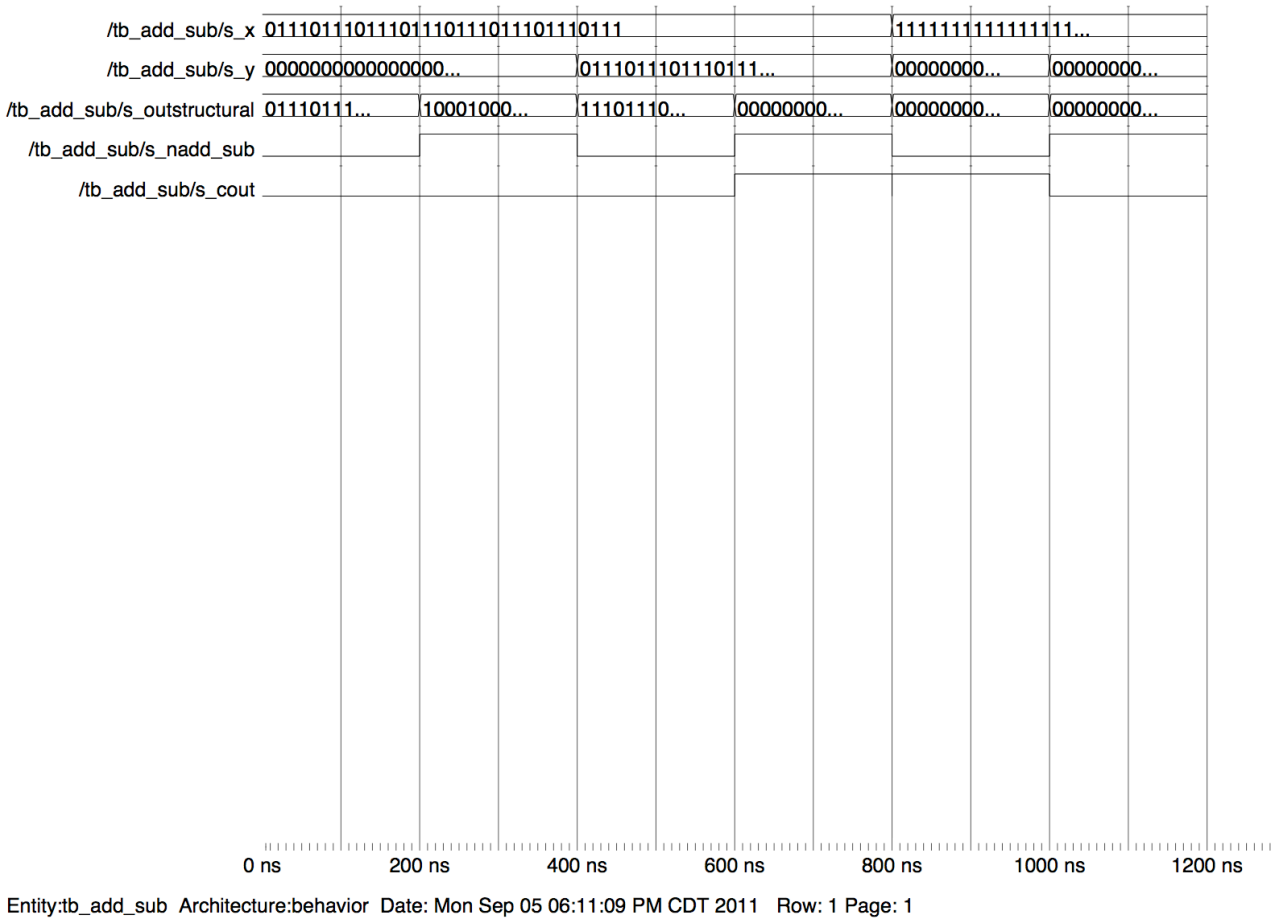


Entity:tb_full_adder Architecture:behavior Date: Mon Sep 05 05:25:51 PM CDT 2011 Row: 1 Page: 1

Adder/Subtracter with Control



This design uses an N-bit Adder, an N-bit 2:1 Mux and an N-bit inverter. If nAdd_Sub = 0, the mux will provide A to the adder, making it perform A + B. If nAdd_Sub = 1, the mux will give the adder the inverse of A (by using the N-bit inverter). It will also pass along 1 to the adder as the carry in bit, which causes it to perform the equivalent of A – B.



In this waveform, I have tested multiple values for the adder/subtracter unit. I tried to make test cases which would test different aspects of addition and subtraction including adding a positive and a negative number, subtracting a value from itself, adding two positive values, adding to make it overflow. See the tb_add_sub.vhd file for all of the values used, and their expected results.