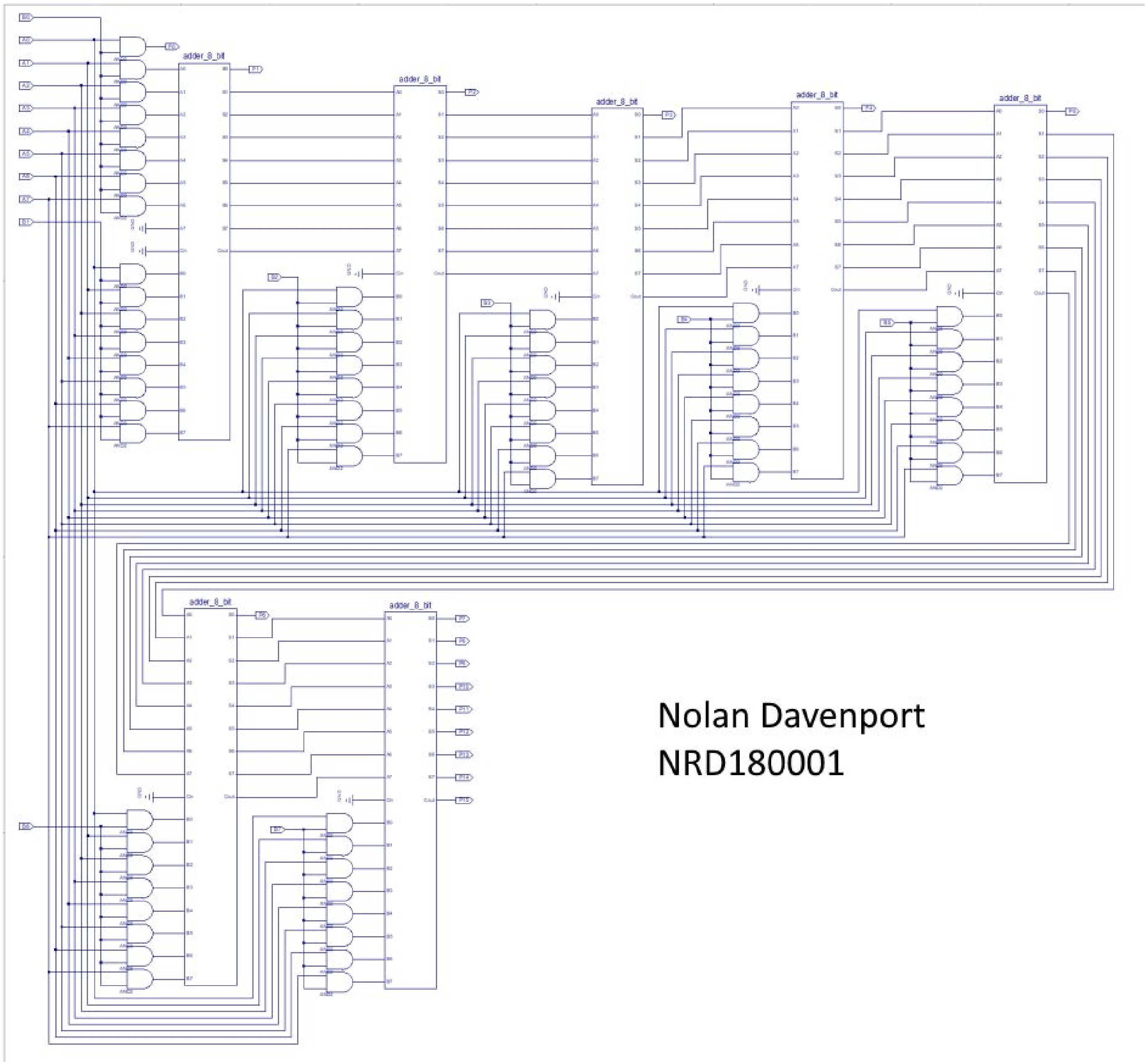
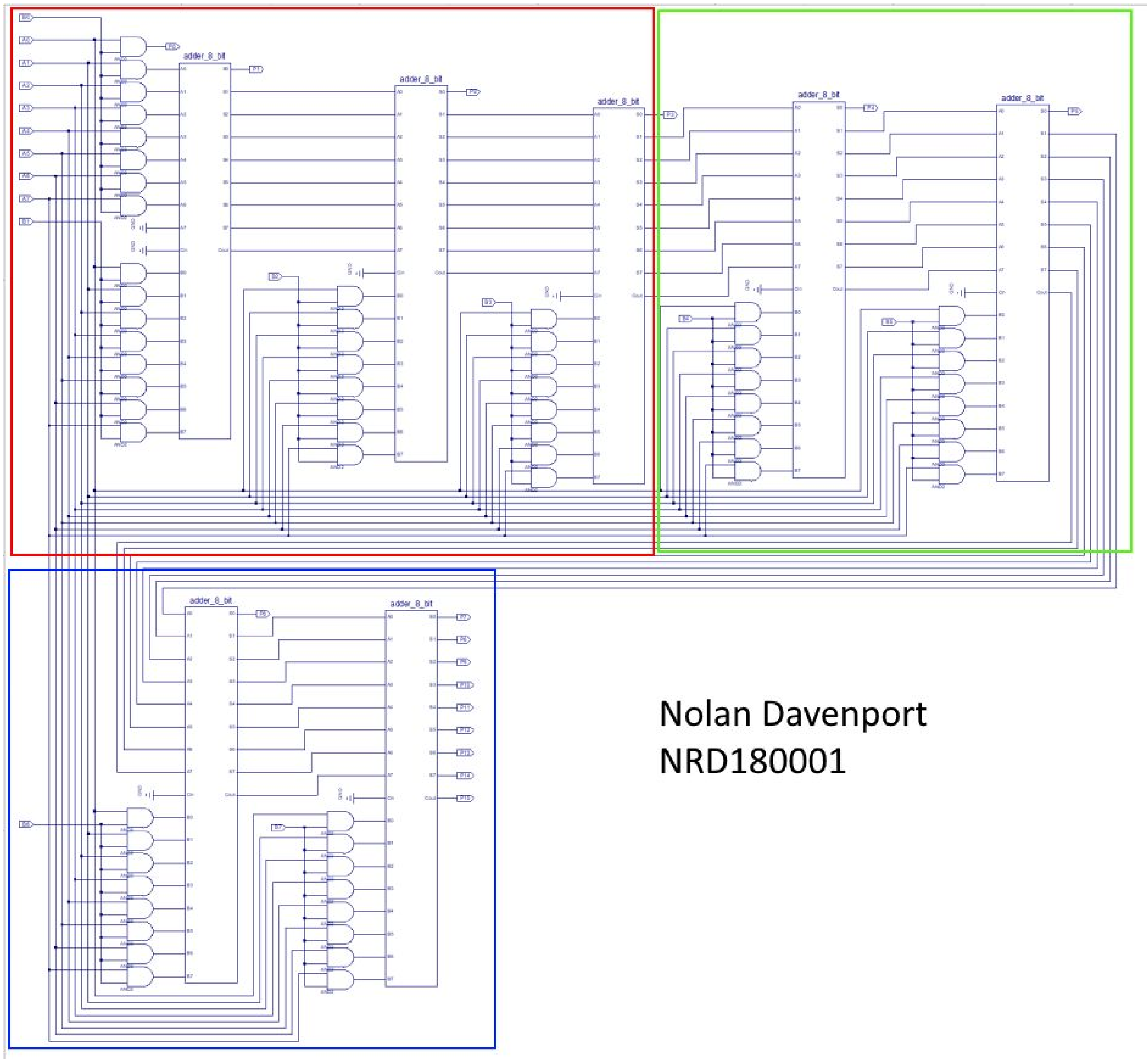
Second Project: 8 Bit Multiplier

Nolan Davenport
CE 3320.002
NRD180001

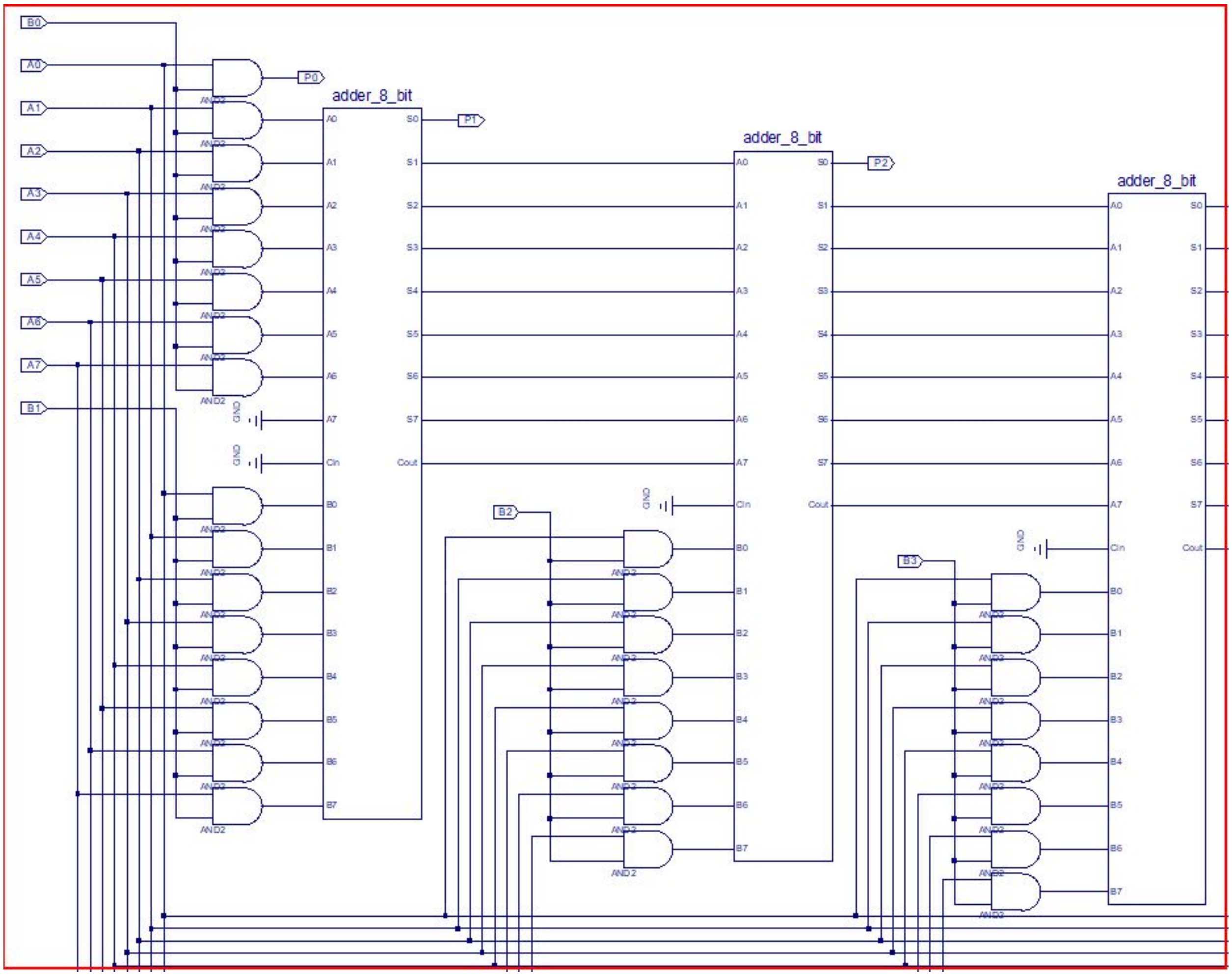
The schematic is too large to fit onto a page and still maintain readability, so I will show a picture of the whole schematic, and then show multiple sections, but zoomed in, so you can see the details.

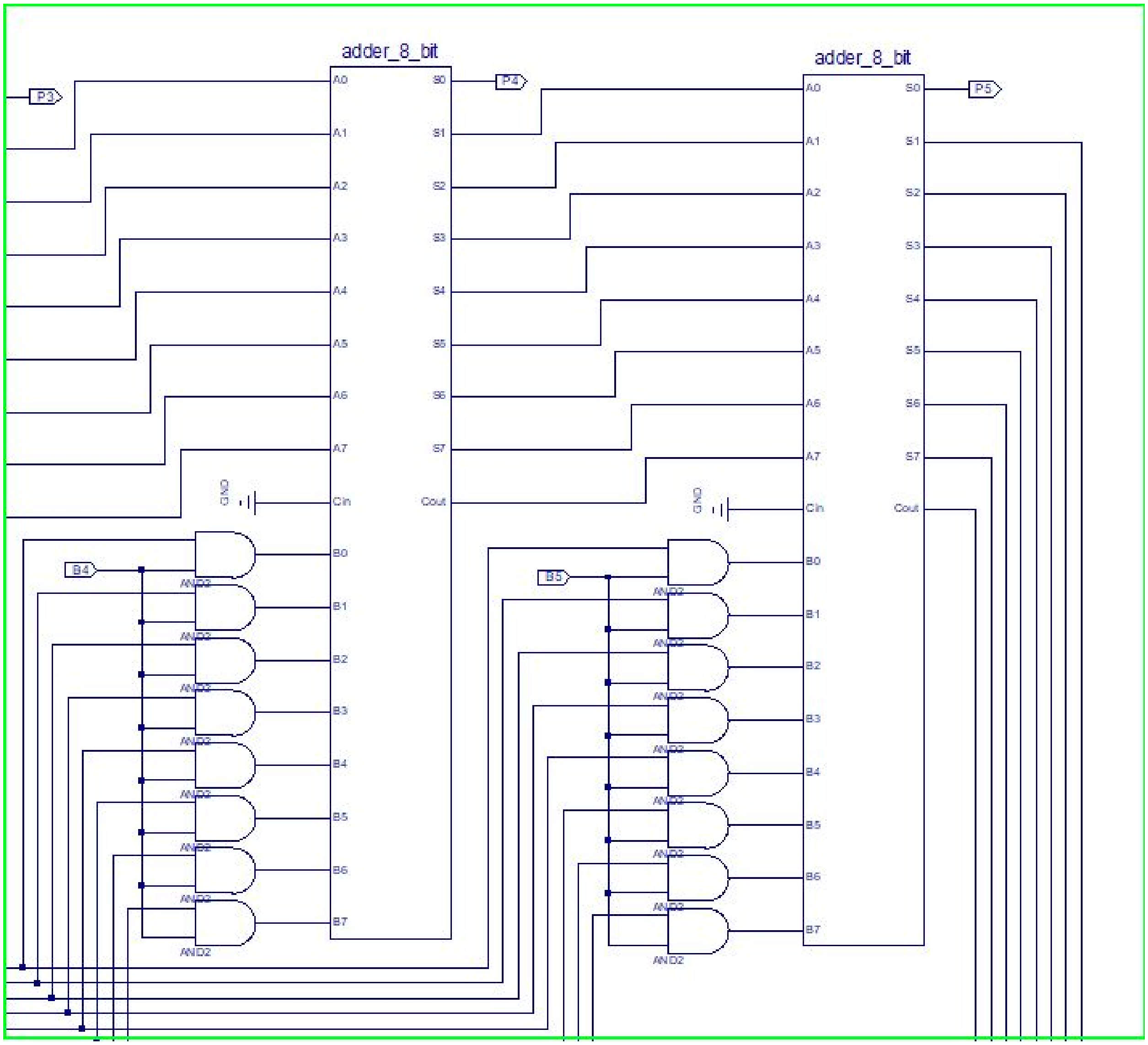


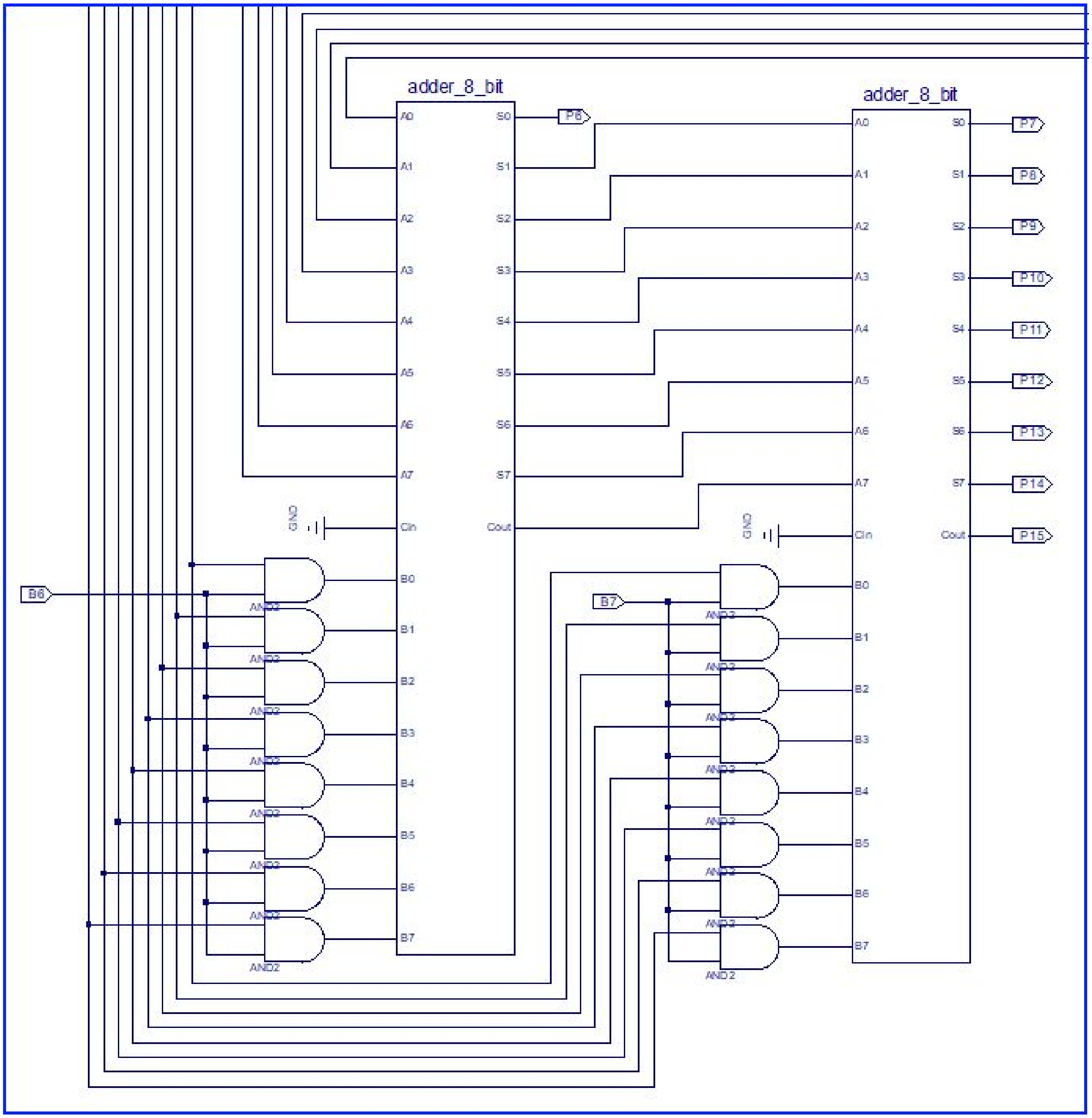
The colored boxes highlight where the sections have been taken from.



Nolan Davenport
NRD180001


B0
A0
A1
A2
A3
A4
A5
A6
A7
B1
B2
B3
P0
P1
P2
adder_8_bit
adder_8_bit
adder_8_bit
AND2
GND
A0
A1
A2
A3
A4
A5
A6
A7
Cin
B0
B1
B2
B3
B4
B5
B6
B7
S0
S1
S2
S3
S4
S5
S6
S7
Cout


adder_8_bit
adder_8_bit
P3
P4
P5
B4
B5
GND
GND
A0
A1
A2
A3
A4
A5
A6
A7
Cin
B0
B1
B2
B3
B4
B5
B6
B7
S0
S1
S2
S3
S4
S5
S6
S7
Cout
AND2


adder_8_bit
A0
A1
A2
A3
A4
A5
A6
A7
Cin
B0
B1
B2
B3
B4
B5
B6
B7
S0
S1
S2
S3
S4
S5
S6
S7
Cout
P6
GND
B6
AND2
AND2
AND2
AND2
AND2
AND2
AND2
AND2
adder_8_bit
A0
A1
A2
A3
A4
A5
A6
A7
Cin
B0
B1
B2
B3
B4
B5
B6
B7
S0
S1
S2
S3
S4
S5
S6
S7
Cout
P7
P8
P9
P10
P11
P12
P13
P14
P15
GND
B7
AND2
AND2
AND2
AND2
AND2
AND2
AND2
AND2

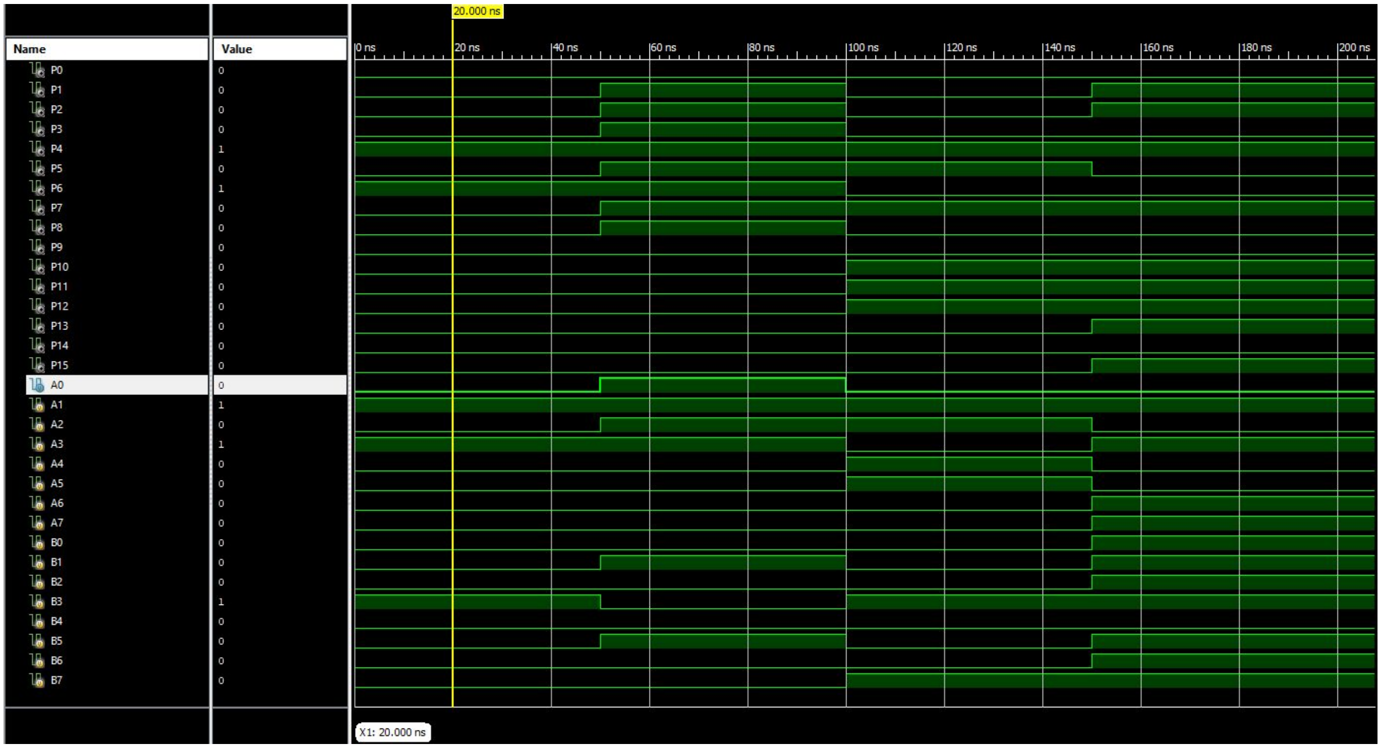
Test File

```
// Verilog test fixture created from schematic
C:\Users\guzzo\Documents\ISE\Multiplier_Project\Multiplier.sch - Thu
Apr 30 15:00:23 2020

`timescale 1ns / 1ps

module Multiplier_Multiplier_sch_tb();

// Inputs
     reg A0;
   reg A1;
   reg A2;
   reg A3;
   reg A4;
   reg A5;
   reg A6;
   reg A7;
   reg B0;
   reg B1;
   reg B2;
   reg B3;
   reg B4;
   reg B5;
   reg B6;
   reg B7;

// Output
   wire P0;
   wire P1;
   wire P2;
   wire P3;
   wire P4;
   wire P5;
   wire P6;
   wire P7;
   wire P8;
   wire P9;
   wire P10;
   wire P11;
   wire P12;
   wire P13;
   wire P14;
   wire P15;


// Bidirs

// Instantiate the UUT
   Multiplier UUT (
           .A1(A1),
           .A2(A2),
           .A3(A3),
           .A4(A4),
           .A5(A5),
           .A6(A6),
           .A7(A7),
           .A0(A0),
           .B0(B0),
           .P0(P0),
           .B1(B1),
           .P1(P1),
           .B2(B2),
           .P2(P2),
           .B3(B3),
           .B4(B4),
           .P3(P3),
           .P4(P4),
           .B5(B5),
           .P5(P5),
           .B6(B6),
           .P6(P6),
           .B7(B7),
           .P7(P7),
           .P8(P8),
           .P9(P9),
           .P10(P10),
           .P11(P11),
           .P12(P12),
           .P13(P13),
           .P14(P14),
           .P15(P15)
   );
// Initialize Inputs
   `ifdef auto_init
       initial begin
```

```
        A0 = 0;
        A1 = 0;
        A2 = 0;
        A3 = 0;
        A4 = 0;
        A5 = 0;
        A6 = 0;
        A7 = 0;
        //
        B0 = 0;
        B1 = 0;
        B2 = 0;
        B3 = 0;
        B4 = 0;
        B5 = 0;
        B6 = 0;
        B7 = 0;
`endif
  initial begin
        A0 = 0;
        A1 = 1;
        A2 = 0;
        A3 = 1;
        A4 = 0;
        A5 = 0;
        A6 = 0;
        A7 = 0;
        //1010000
        B0 = 0;
        B1 = 0;
        B2 = 0;
        B3 = 1;
        B4 = 0;
        B5 = 0;
        B6 = 0;
        B7 = 0;
        #50
        A0 = 1;
        A1 = 1;
        A2 = 1;
        A3 = 1;
        A4 = 0;
        A5 = 0;
```

```
A6 = 0;
A7 = 0;
//111111110
B0 = 0;
B1 = 1;
B2 = 0;
B3 = 0;
B4 = 0;
B5 = 1;
B6 = 0;
B7 = 0;
#50
A0 = 0;
A1 = 1;
A2 = 1;
A3 = 0;
A4 = 1;
A5 = 1;
A6 = 0;
A7 = 0;
//1110010110000
B0 = 0;
B1 = 0;
B2 = 0;
B3 = 1;
B4 = 0;
B5 = 0;
B6 = 0;
B7 = 1;
#50
A0 = 0;
A1 = 1;
A2 = 0;
A3 = 1;
A4 = 0;
A5 = 0;
A6 = 1;
A7 = 1;
//1011110010010110
B0 = 1;
B1 = 1;
B2 = 1;
B3 = 1;
```

```
        B4 = 0;
        B5 = 1;
        B6 = 1;
        B7 = 1;
    end
endmodule
```



Name	Value
P0	0
P1	0
P2	0
P3	0
P4	1
P5	0
P6	1
P7	0
P8	0
P9	0
P10	0
P11	0
P12	0
P13	0
P14	0
P15	0
A0	0
A1	1
A2	0
A3	1
A4	0
A5	0
A6	0
A7	0
B0	0
B1	0
B2	0
B3	1
B4	0
B5	0
B6	0
B7	0

For my functionality test, I input 4 different pairs of numbers with products that are ordered from lowest to highest to test the multiplier. Below are the test multiplications that I performed. The interval between each calculation was 50ms.

1. 00001010 * 00001000 = 0000000000101000
2. 00001111 * 00100010 = 0000000011111110
3. 00110110 * 10001000 = 0001110010110000
4. 11001010 * 11101111 = 1011110010010110

As you can see from the timing diagram, The P values, with **P15** being the MSB and **P0** being the LSB, produced the proper 16 bit product from the given 8 bit **A** and **B** inputs. Being as I put a big range of numbers through the multiplier, with **40** being the lowest product, and **48,278** being the highest product, it is safe to conclude that my completed design for an 8 bit multiplier works properly.