Bangladesh University of Engineering and Technology

Department of Electrical and Electronics Engineering

Course Number: EEE 304

Course Title: Digital Electronics Laboratory

Experiment Number: 02

Name of the Experiment(s):

Design, Simulation, and Implementation of Arithmetic Circuits using 74 series ICs and VerilogHDL

Prepared by: Tasmin Khan

Student ID: 1906044

Level:3 Term:2

Lab exercise 1

Question

(a) Write the Verilog code for the full adder circuit using gate level primitives (structural code). Simulate the circuit and observe the wave form.

(b) Repeat the above experiment with the behavioral description of Verilog code and name the module as fulladd.

(c) Write the Verilog code of a 4-bit ripple carry adder by instantiating the fulladd module as a subcircuit. Represent one of the 4-bit numbers to be added by the four signals

x3(MSB), x2, x1, x0(LSB), and the other number as y3(MSB) , y2, y1, y0(LSB). Represent the sum as s3, s2, s1, s0 and the carry output as carryout.

1 bit Full Adder

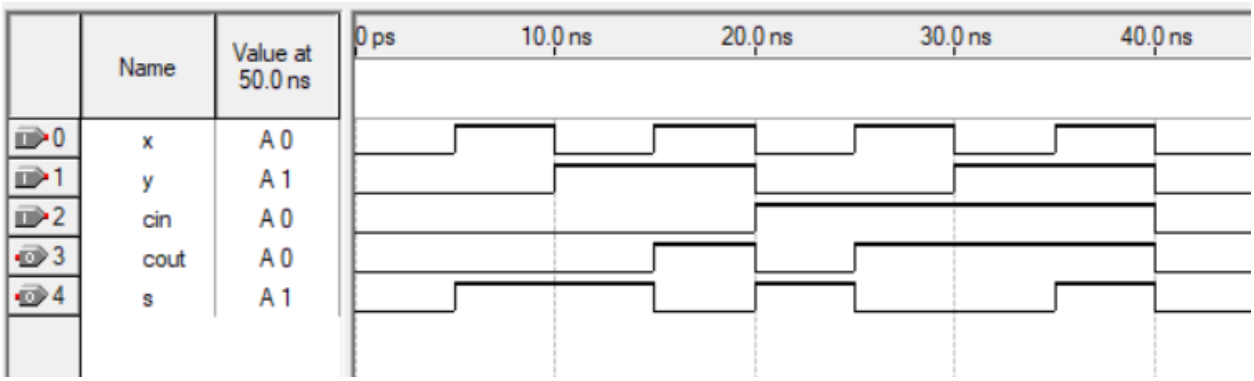
Truth Table:

Inputs			Outputs	
A	B	C_{in}	Sum	Carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

(Structural code)

```
1  module fulladder (cin, x,y,s,cout);
2  input cin,x,y;
3  output s, cout;
4  wire z1,z2,z3;
5  xor (s,x,y,cin);
6  and (z1,x,y);
7  and(z2, x,cin);
8  and (z3,y,cin);
9  or (cout, z1,z2,z3);
10 endmodule
```

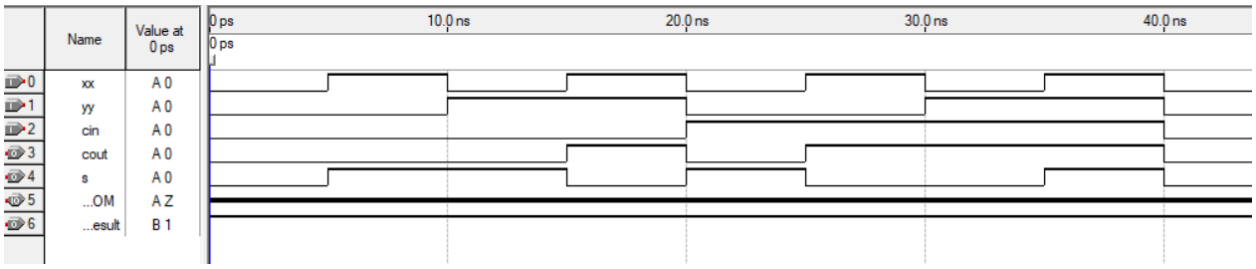
Vector Waveform



(Behavioral code)

```
1  module fulladder(cin,xx,yy,s,cout,LED_COM);
2  input cin,xx,yy;
3  output cout,s;
4  inout LED_COM;
5  assign s=xx^yy^cin;
6  assign cout=(xx&yy)|(xx&cin)|(yy&cin);
7  assign LED_COM=1;
8  endmodule
```

Vector waveform:



4 bit Full Adder

Truth Table:

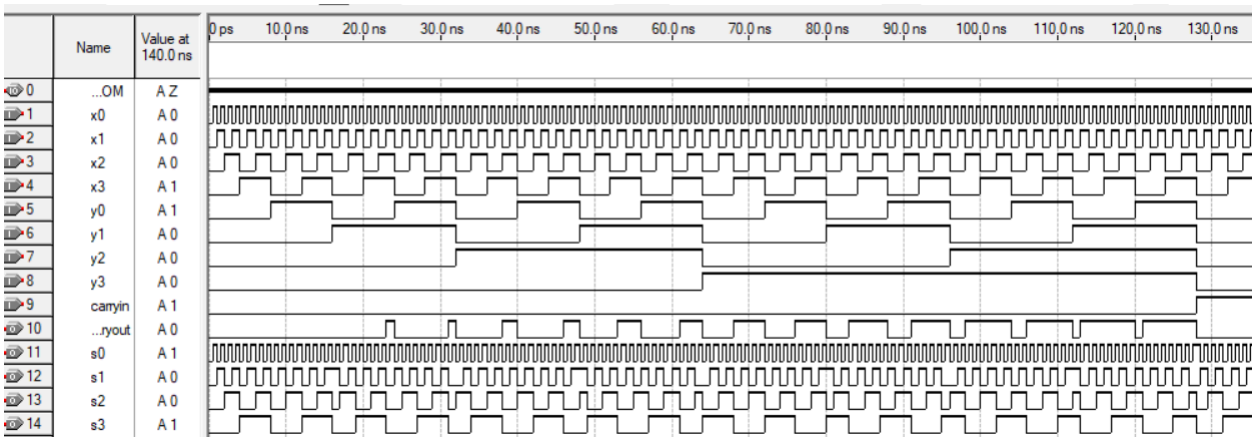
	A				B				Sum				Carry
Cin	A3	A2	A1	A0	B3	B2	B1	B0	S3	S2	S1	S0	Cout
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	1	0	0	1	0	0
0	0	0	1	0	0	0	1	0	0	1	0	0	0
0	0	0	1	1	0	0	1	1	0	1	1	0	0
0	0	1	0	0	0	1	0	0	1	0	0	0	0
0	0	1	0	1	0	1	0	1	1	0	1	0	0
0	0	1	1	0	0	1	1	0	1	1	0	0	0
0	0	1	1	1	0	1	1	1	1	1	1	0	0
0	1	0	0	0	1	0	0	0	0	0	0	0	1
0	1	0	0	1	1	0	0	1	0	0	1	0	1
0	1	0	1	0	1	0	1	0	0	1	0	0	1
0	1	0	1	1	1	0	1	1	0	1	1	0	1
0	1	1	0	0	1	1	0	0	1	0	0	0	1
0	1	1	0	1	1	1	0	1	1	0	1	0	1
0	1	1	1	0	1	1	1	0	1	1	0	0	1
0	1	1	1	1	1	1	1	1	1	1	1	0	1

Verilog code:

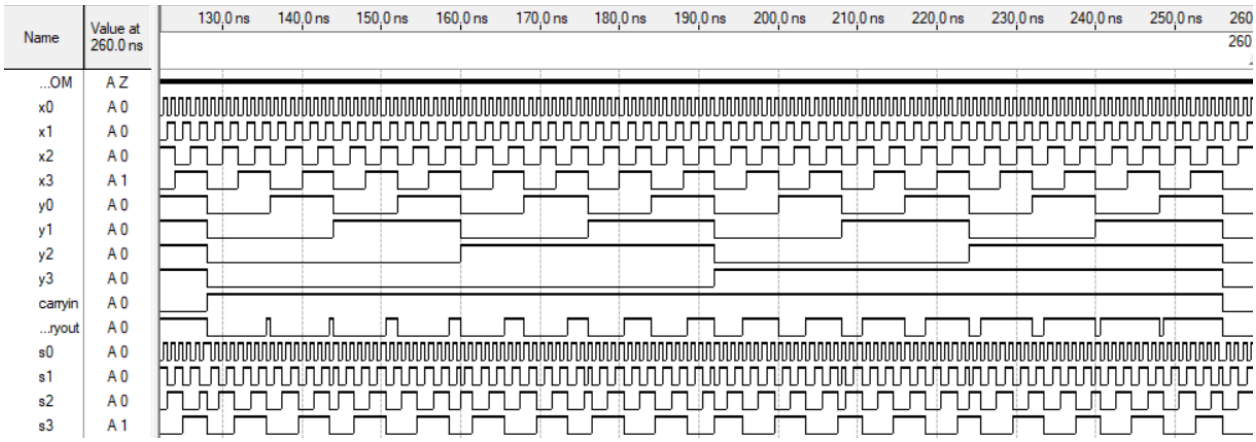
```
1  module adder(carryin, x3,x2,x1,x0,y3,y2,y1,y0,s3,s2,s1,s0,carryout, LED_COM);
2  input carryin, x3,x2,x1,x0,y3,y2,y1,y0;
3  output s3,s2,s1,s0,carryout;
4  inout LED_COM;
5  wire c1,c2,c3;
6  fulladd stage0(carryin, x0,y0,s0,c1);
7  fulladd stage1(c1, x1,y1,s1,c2);
8  fulladd stage2(c2, x2,y2,s2,c3);
9  fulladd stage3(c3, x3,y3,s3,carryout);
10 assign LED_COM=1;
11 endmodule
12
13 module fulladd(cin,xx,yy,s,cout);
14 input cin,xx,yy;
15 output cout,s;
16 assign s=xx^yy^cin;
17 assign cout=(xx&yy)|(xx&cin)|(yy&cin);
18 endmodule
19
```

Vector waveform:

For carryin = 0



For carryin = 1



Lab exercise 2

Question

Write a Verilog code of an adder/subtractor circuit which will add two 4-bit numbers when a control signal ctrl = 0 and subtract the two 4-bit number when the control signal ctrl = 1.

4 bit Adder-subtractor

Verilog code:

```
module adder(carryin, x3,x2,x1,x0,y3,y2,y1,y0,s3,s2,s1,s0,carryout, LED_COM);
input carryin, x3,x2,x1,x0,y3,y2,y1,y0;
output s3,s2,s1,s0,carryout;
inout LED_COM;
wire c1,c2,c3,yy0,yy1,yy2,yy3;
adsb(carryin,yo,y1,y2,y3,yy0,yy1,yy2,yy3);
fulladd stage0(carryin, x0,yy0,s0,c1);
fulladd stage1(c1, x1,yy1,s1,c2);
fulladd stage2(c2, x2,yy2,s2,c3);
fulladd stage3(c3, x3,yy3,s3,carryout);
assign LED_COM=1;
endmodule

module fulladd(cin,xx,yy,s,cout);
input cin,xx,yy;
output cout,s;
assign s=xx^yy^cin;
assign cout=(xx&yy)|(xx&cin)|(yy&cin);
endmodule

module adsb(ctrl,yo,y1,y2,y3,yy0,yy1,yy2,yy3);
input ctrl,yo,y1,y2,y3;
output yy0,yy1,yy2,yy3;
xor xor0(yy0,ctrl,y0);
xor xor1(yy1,ctrl,y1);
xor xor2(yy2,ctrl,y2);
xor xor3(yy3,ctrl,y3);
endmodule
```

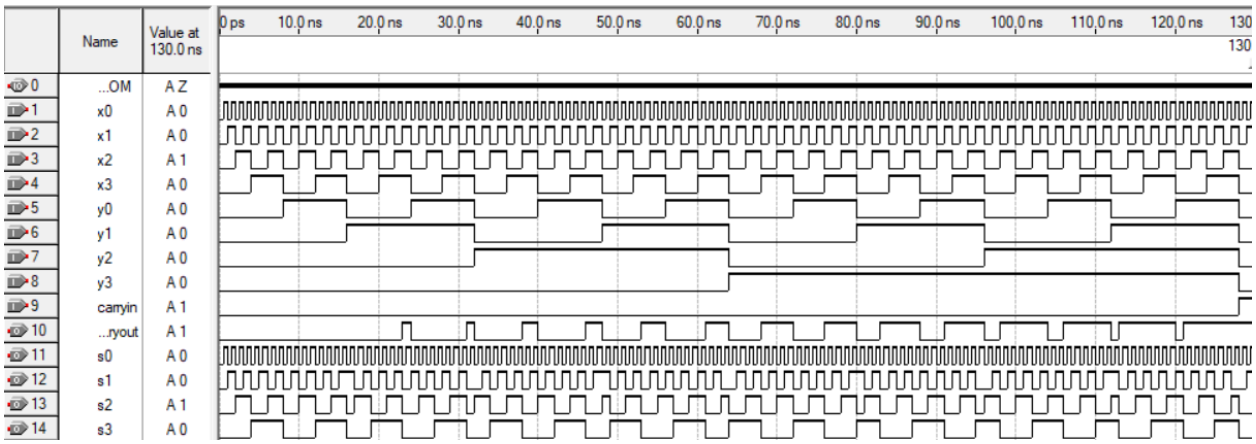
Here we use a four bit full adder circuit to perform subtraction. To do so, we do XOR operation of subtrahend and carryin before subtraction. This way, when the

carry in is 1, it is effectively performing 2's complement operation on the second number to be used as a subtrahend.

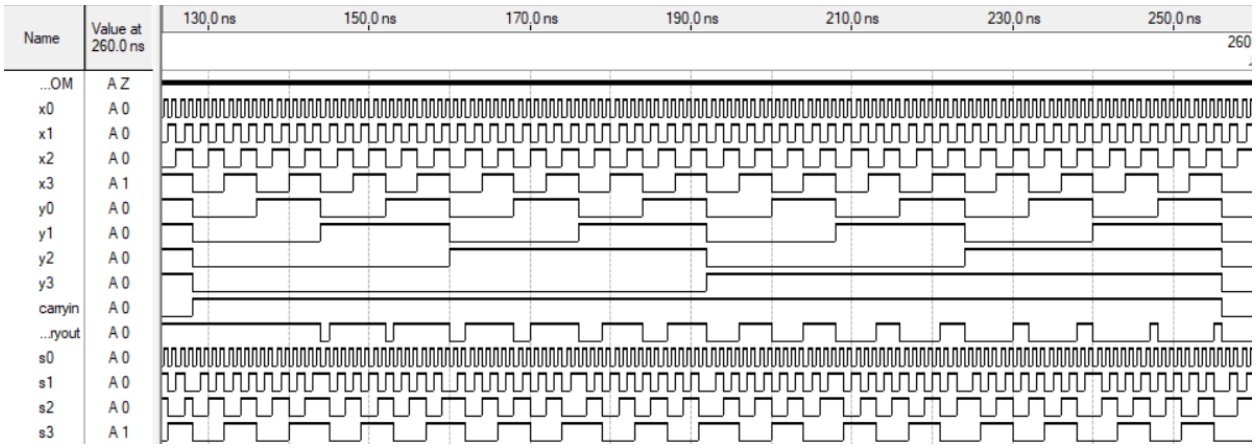
At the same time, we can do regular addition operations when the carry in is zero. Thus we made an adder subtractor.

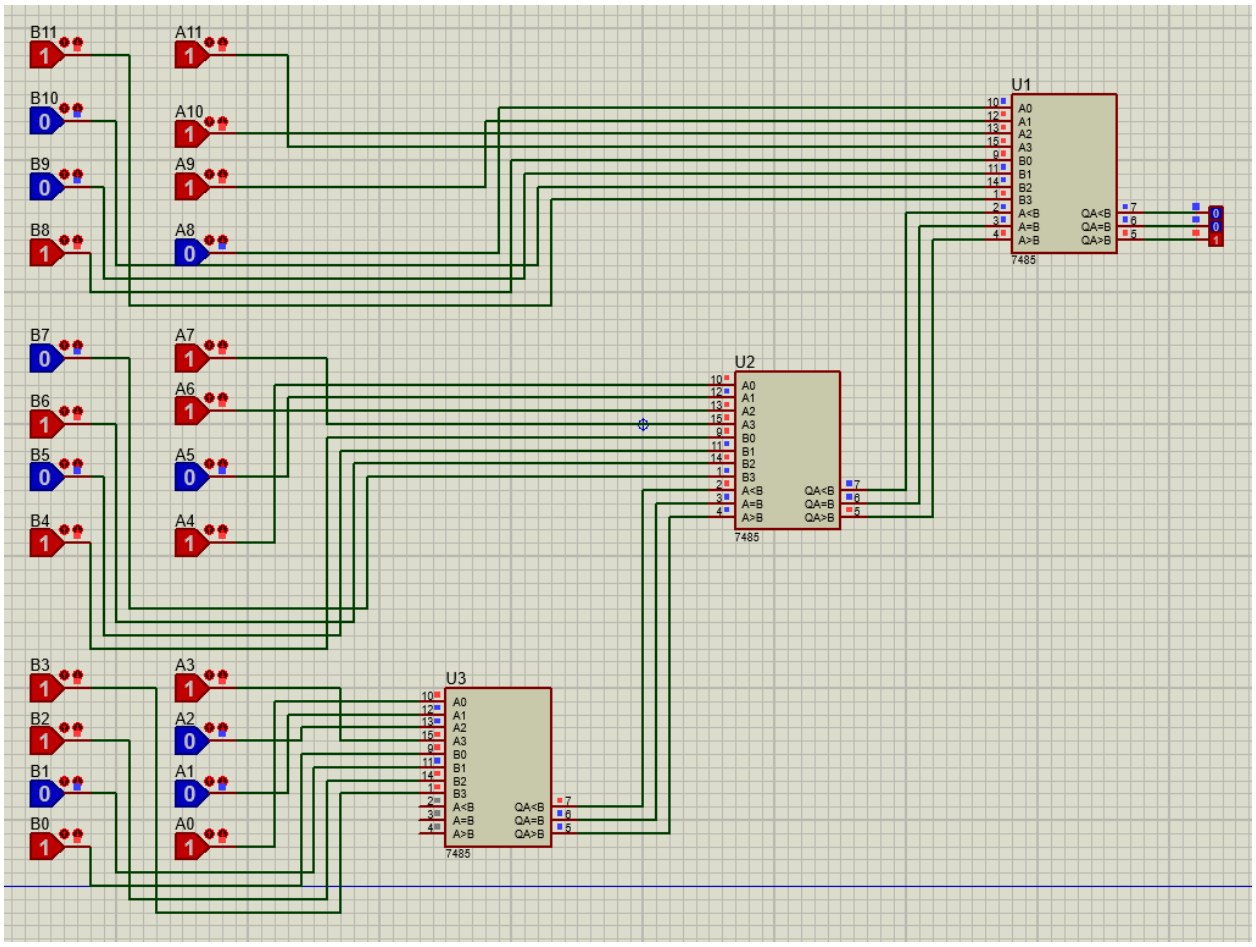
Vector waveform:

Addition for carryin = 0



Subtraction For carryin = 1





Here we implemented a 12 bit comparator circuit to compare A and B where

A=111011011001

B=100101011101

Clearly A>B and we can see that in the finally output from the circuit as well,

Report Problem

In a ternary number system, there are three digits: 0, 1, and 2. The following table defines a ternary half-adder. Write a Verilog code and plot vector waveform that implements this half-adder using binary-encoded signals, such that two bits are used for each ternary digit. Let A = a1a0, B = b1b0, and Sum = s1s0; note that Carry is just a binary signal. Use the following encoding: 00 = (0)3, 01 = (1)3, and 10 = (2)3. Also find Boolean expression of outputs that minimize the cost of the circuit.

A B	Carry	Sum
0 0	0	0
0 1	0	1
0 2	0	2
1 0	0	1
1 1	0	2
1 2	1	0
2 0	0	2
2 1	1	0
2 2	1	1

Ternary Half Adder

In a ternary number system, there are three digits: 0, 1, and 2.

The following table defines a ternary half-adder.

A B	Carry	Sum
0 0	0	0
0 1	0	1
0 2	0	2
1 0	0	1
1 1	0	2
1 2	1	0
2 0	0	2
2 1	1	0
2 2	1	1

Now to implement this half-adder using binary-encoded signals, we will use two bits for each ternary digit.

Let A = a1a0, B = b1b0, and Sum = s1s0

Output carry is just a binary signal. There is no input carry since this is a half adder. We will use the following encoding:

00 = (0)3, 01 = (1)3, and 10 = (2)3

Binary Encoded Ternary Half Adder truth table

A1	A0	B1	B0	S1	S0	C
0	0	0	0	0	0	0
0	0	0	1	0	1	0
0	0	1	0	1	0	0
0	1	0	0	0	1	0
0	1	0	1	1	0	0
0	1	1	0	0	0	1
1	0	0	0	1	0	0
1	0	0	1	0	0	1
1	0	1	0	0	1	1

Karnaugh Map for expression of Sum bit 0:

A1A0 B1B0	00	01	11	10
00	0	1	x	0
01	1	0	x	0
11	x	x	x	x
10	0	0	x	1

Boolean Expression:

A1'A0'B0 + A0*B1'B0'

Karnaugh Map for expression of Sum bit 1:

A1A0 / B1B0	00	01	11	10
00	0	0	x	1
01	0	1	x	0
11	x	x	x	x
10	1	0	x	0

Boolean Expression:

A0*B0 + A1*B1'B0'

Karnaugh Map for expression of Carry out:

A1A0 / B1B0	00	01	11	10
00	0	0	x	0
01	0	0	x	1
11	x	x	x	x
10	0	1	x	1

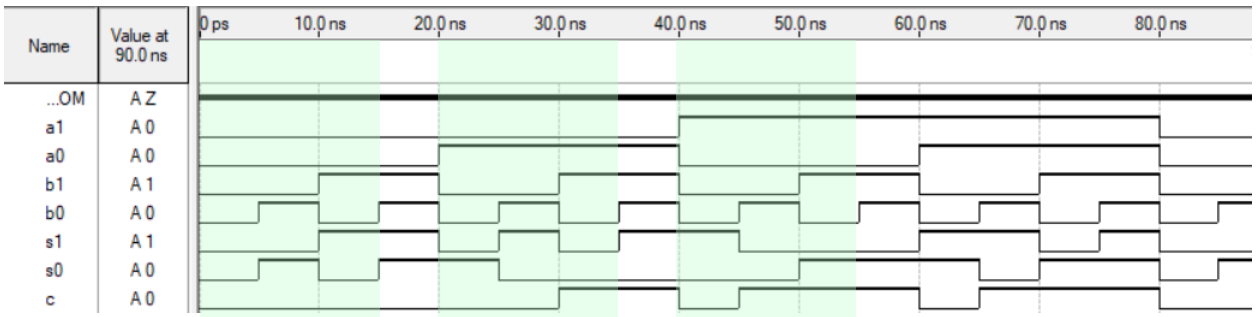
Boolean Expression:

A1* (B0 + B1) + A0*B1

Verilog code:

```
1  module ternary(c,a0,a1,b0,b1,s0,s1,LED_COM);
2  input a0,a1,b0,b1;
3  wire x0,y0,x1,y1;
4  output s0,s1,c;
5  inout LED_COM;
6
7  assign x0=!a0;
8  assign x1=!a1;
9  assign y0=!b0;
10 assign y1=!b1;
11
12 assign s0 = (x1&x0&b0)|(a0&y1&y0)|(a1&b1);
13 assign s1 = (x1&x0&b1)|(a1&y1&y0)|(a0&b0);
14 assign c  = (a0&b1)|(a1&b1)|(a1&b0);
15
16 assign LED_COM=1;
17 endmodule
18
```

Vector Waveform:



Here the parts of the waveform that is needed for ternary adder is darkened. It can be verified matching with the truth table. The rest of the values are don't care values for sum and carry out.

Discussion:

In this experiment, I performed addition and subtraction using digital logic. I also implemented the comparator circuits and ternary half adder circuit. Hardware description language Verilog was used to describe the digital logic circuit.