Lab Exercise 7

7.1 Structural Description of 1 Bit Full Adder

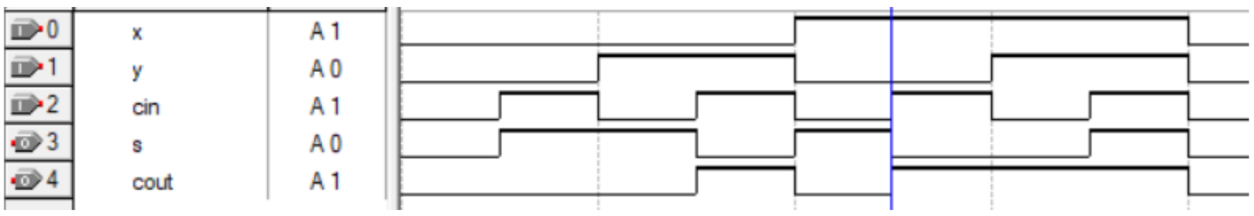
7.1.1 Truth Table

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

7.1.2 Verilog Code:

```
module fulladd (cin, x,y,s,cout);
input cin,x,y;
        output s, cout;
        wire z1,z2,z3;
        xor (s,x,y,cin);
        and (z1,x,y);
        and(z2, x,cin);
        and (z3,y,cin);
        or (cout, z1,z2,z3);
    endmodule
```

7.1.3 Vector Waveform

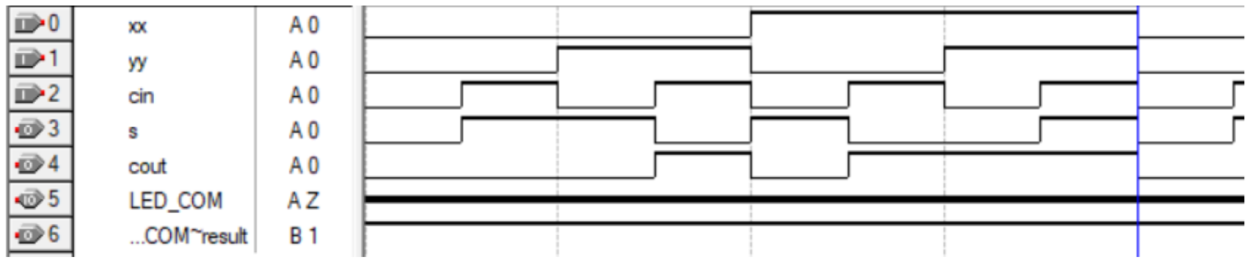


7.2 Behavioral Description of 1-bit Full Adder

7.2.1 Verilog Code:

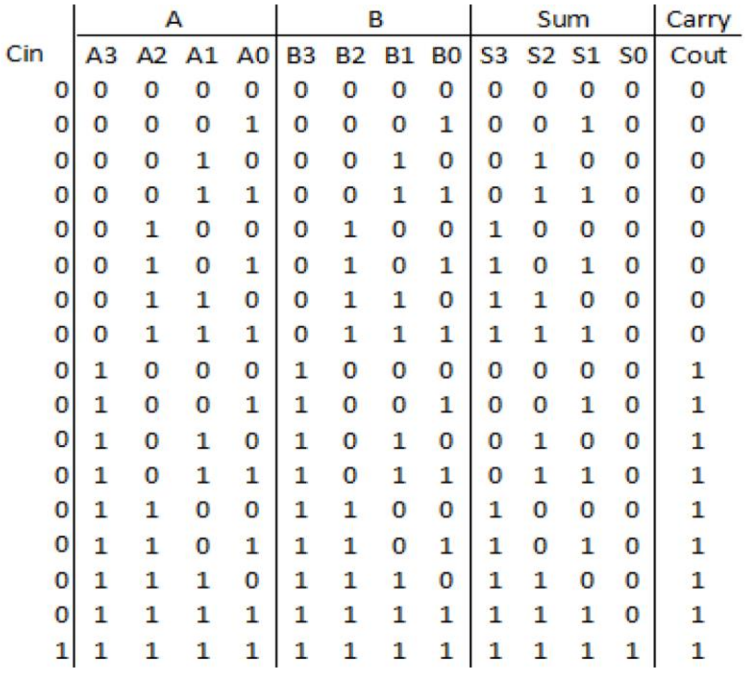
```
module fulladd_behav(cin,xx,yy,s,cout,LED_COM);
        input cin,xx,yy;
        output cout,s;
        inout LED_COM;
        assign s=xx^yy^cin;
        assign cout=(xx&yy)|(xx&cin)|(yy&cin);
        assign LED_COM=1;
endmodule
```

7.2.2 Vector Waveform



7.3 Behavioral Description of 4 Bit Full Adder

7.3.1 Truth Table

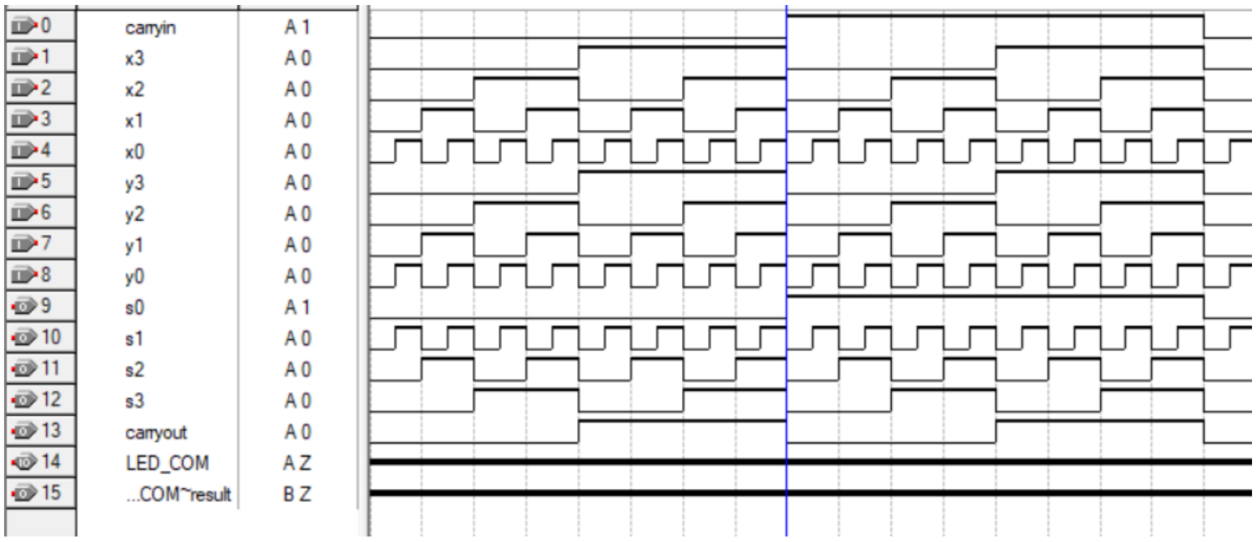


	A				B				Sum				Carry
Cin	A3	A2	A1	A0	B3	B2	B1	B0	S3	S2	S1	S0	Cout
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	1	0	0	1	0	0
0	0	0	1	0	0	0	1	0	0	1	0	0	0
0	0	0	1	1	0	0	1	1	0	1	1	0	0
0	0	1	0	0	0	1	0	0	1	0	0	0	0
0	0	1	0	1	0	1	0	1	1	0	1	0	0
0	0	1	1	0	0	1	1	0	1	1	0	0	0
0	0	1	1	1	0	1	1	1	1	1	1	0	0
0	1	0	0	0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

7.3.2 Verilog Code

```
module adder4(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;
        fulladd stage0 (carryin, x0,y0,s0,c1);
        fulladd stage1 (c1, x1,y1,s1,c2);
        fulladd stage2 (c2, x2,y2,s2,c3);
        fulladd stage3 (c3, x3,y3,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
```

7.3.3 Vector Waveform

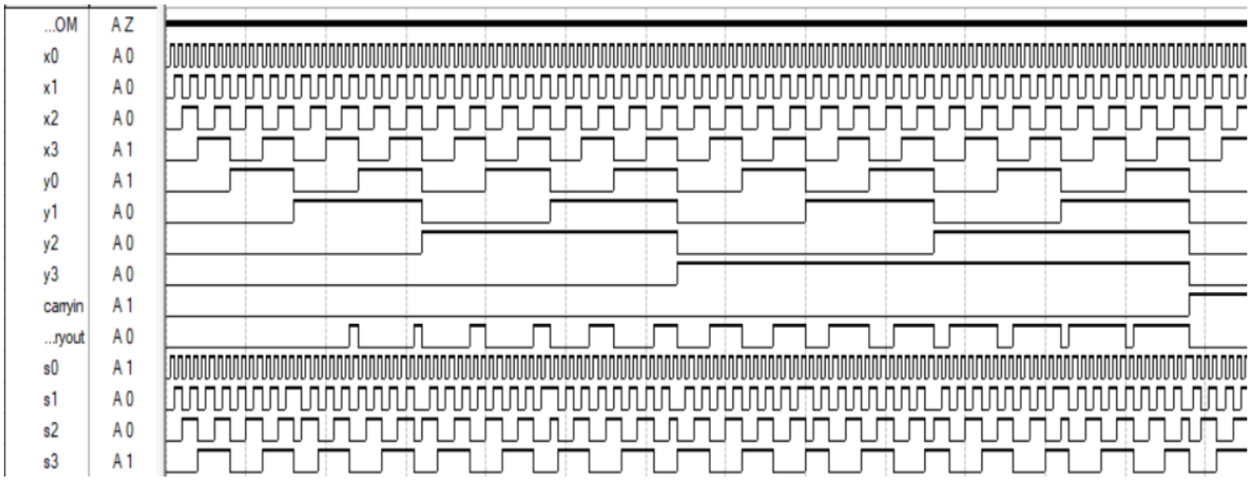


Lab Exercise 8

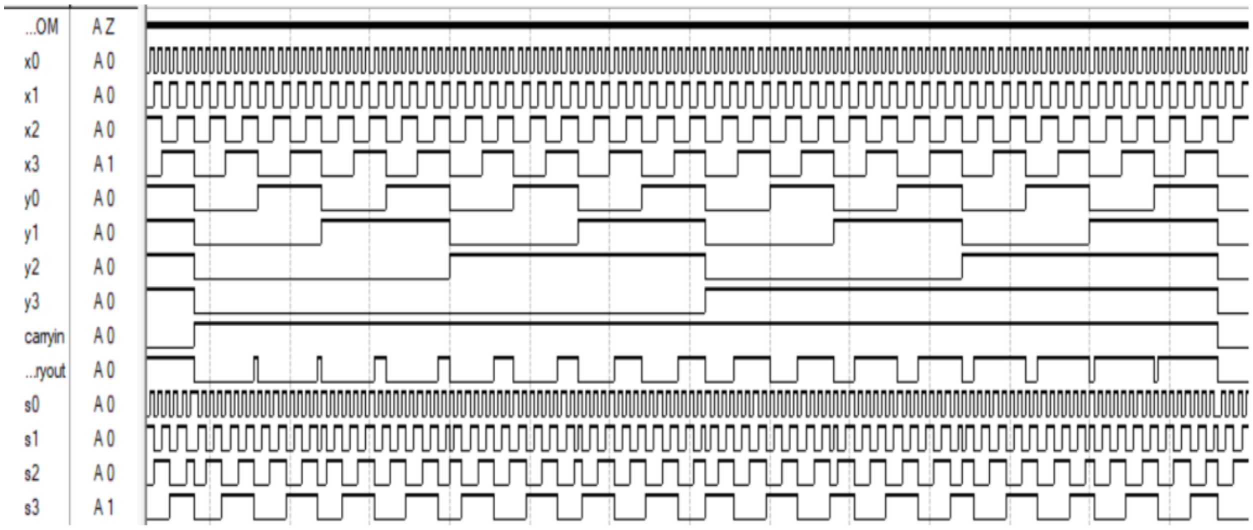
8.1 Verilog Code

```
module addsub(cin, x3,x2,x1,x0,y3,y2,y1,y0,s3,s2,s1,s0,cout, LED_COM);

input cin, x3,x2,x1,x0,y3,y2,y1,y0;

output s3,s2,s1,s0,cout;

inout LED_COM;

wire c1,c2,c3,yy0, yy1, yy2,yy3;

adsub(cin, y3,y2,y1,y0,yy0, yy1, yy2,yy3);

fulladd stage0 (cin, x0,y0,s0,c1);

fulladd stage1 (c1, x1,y1,s1,c2);

fulladd stage2 (c2, x2,y2,s2,c3);

fulladd stage3 (c3, x3,y3,s3,cout);

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 adsub(cin, y3,y2,y1,y0,yy0, yy1, yy2,yy3);

input cin, y0, y1, y2, y3;

output yy0, yy1, yy2, yy3;

assign yy0 = cin^y0;

assign yy1 = cin^y1;
assign yy2 = cin^y2;
assign yy3 = cin^y3;
endmodule
```

8.2 Vector Waveform for Addition (Cin = 0)



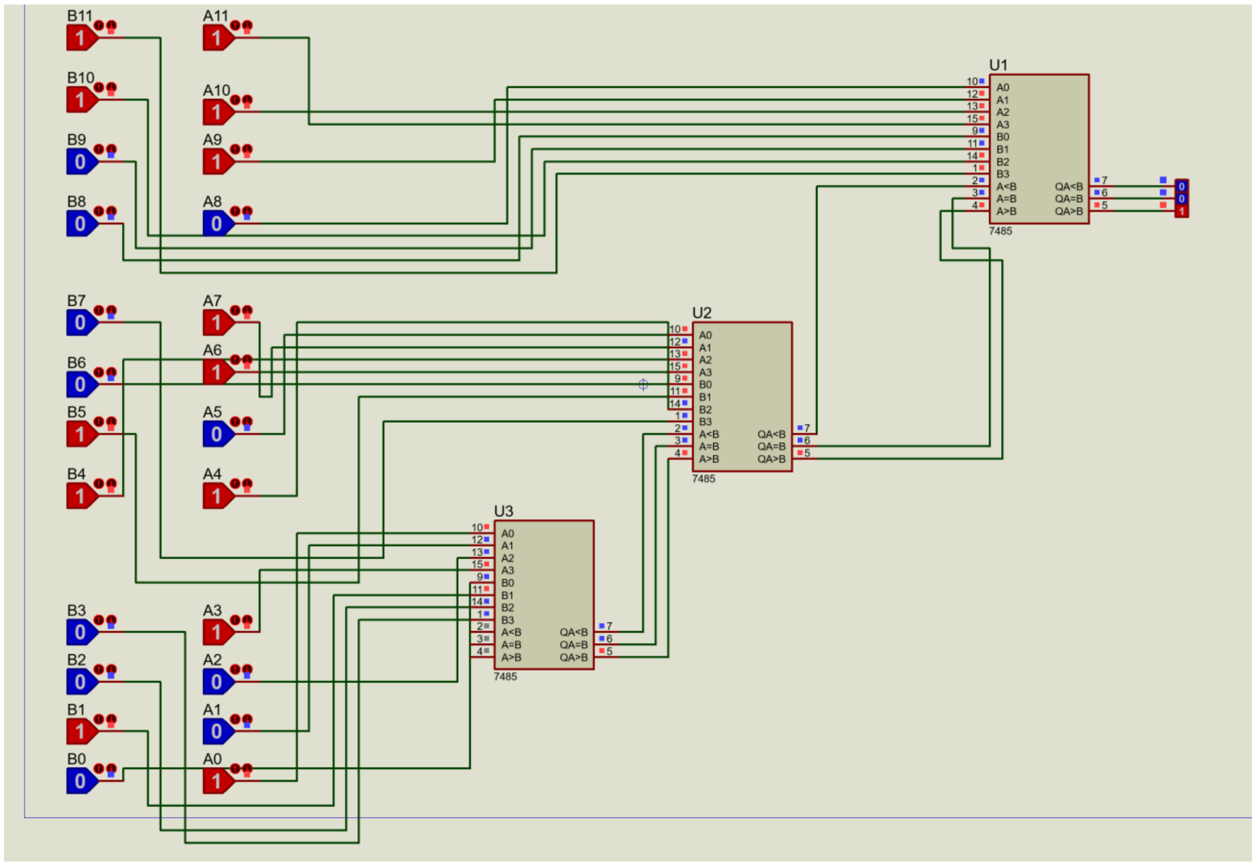
8.3 Vector Waveform Subtraction (Cin = 1)



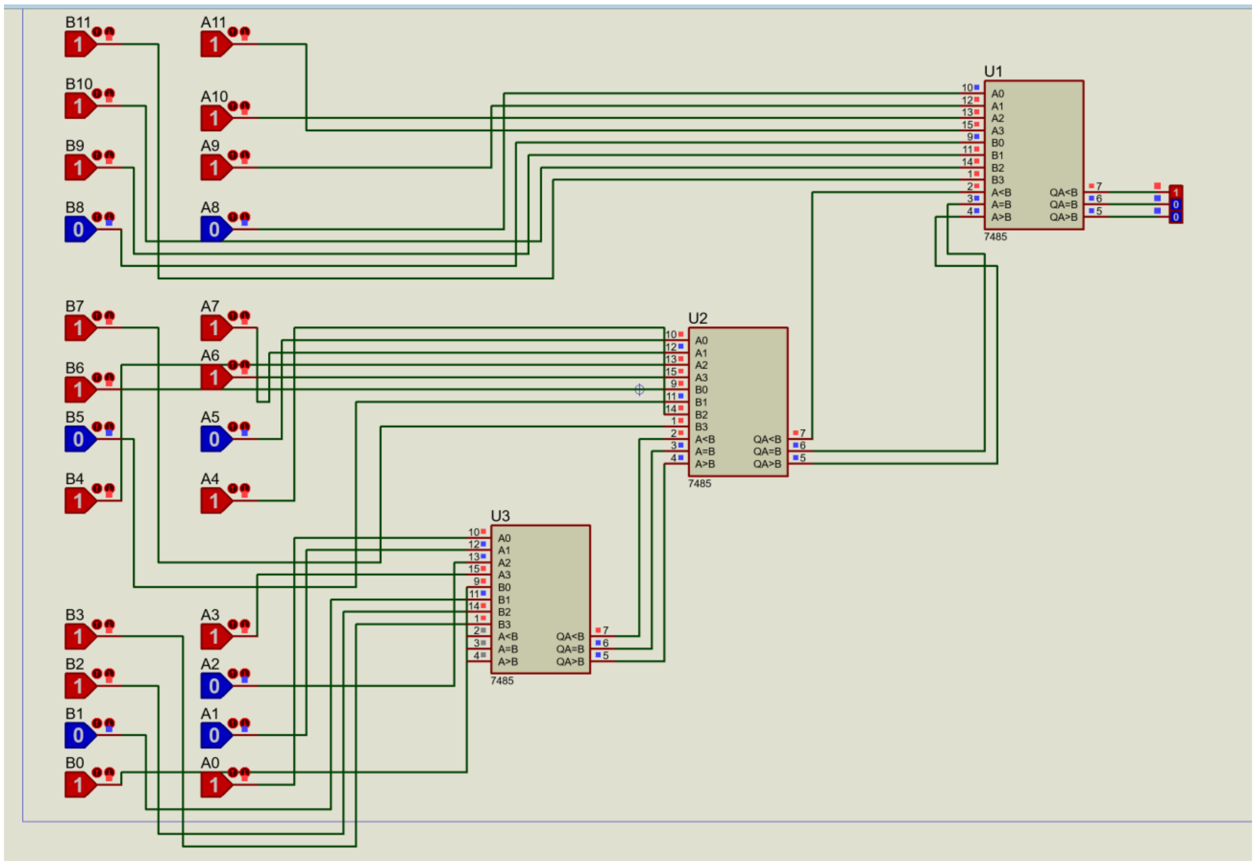
Report Problem 1

12 Bit Comparator

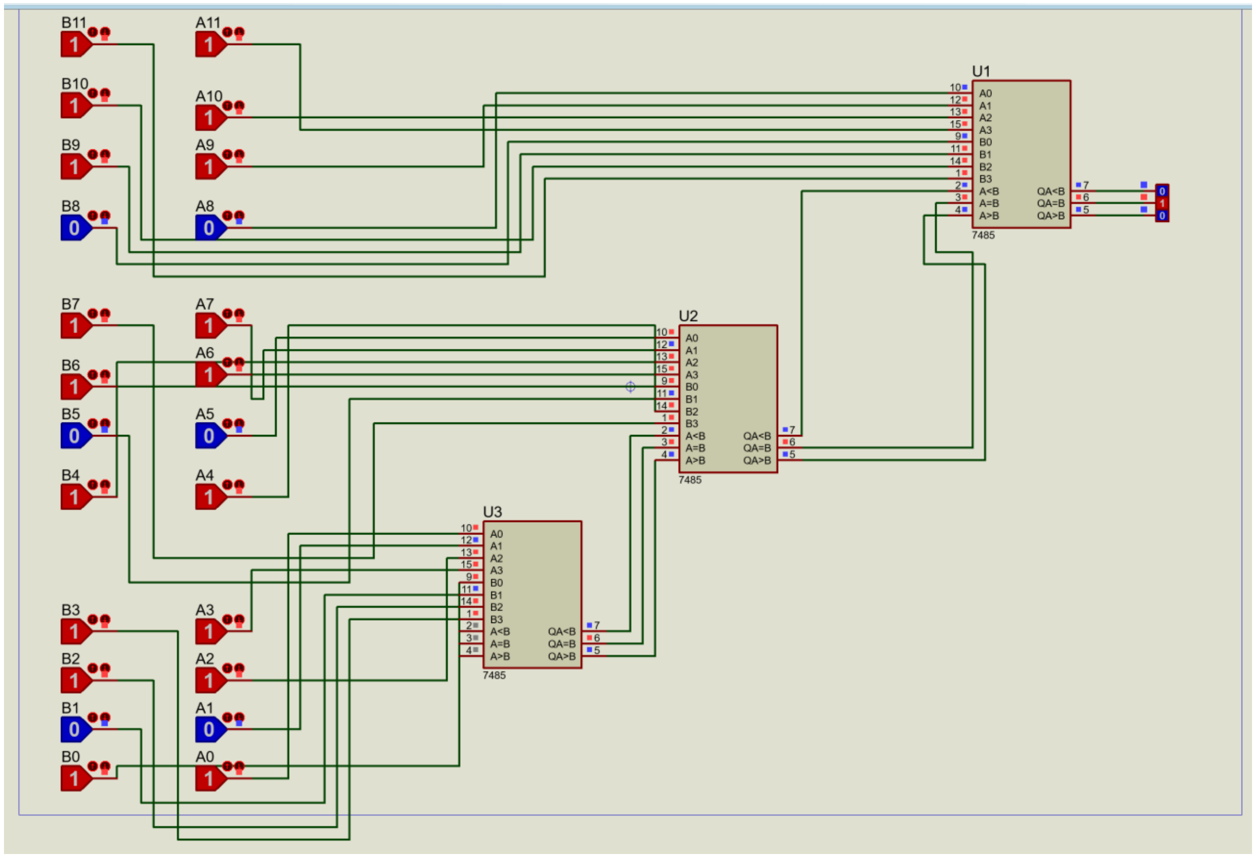
R.1.1 A> B



R.1.2 A< B


B11
A11
B10
A10
B9
A9
B8
A8
U1
7485
B7
A7
B6
A6
B5
A5
B4
A4
U2
7485
U3
7485
B3
A3
B2
A2
B1
A1
B0
A0
A0
A1
A2
A3
B0
B1
B2
B3
A<B
A=B
A>B
QA<B
QA=B
QA>B

R.1.3 A = B



B11
A11
B10
A10
B9
A9
B8
A8
B7
A7
B6
A6
B5
A5
B4
A4
B3
A3
B2
A2
B1
A1
B0
A0
U1
U2
U3
7485
A0
A1
A2
A3
B0
B1
B2
B3
A<B
A=B
A>B
QA<B
QA=B
QA>B

Report Problem 2

Ternary Adder

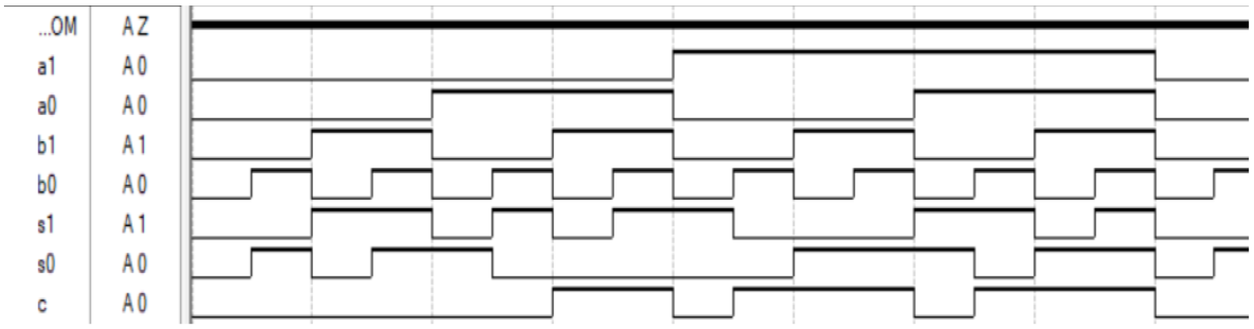
R.2.1 Binary Encoded Truth Table

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

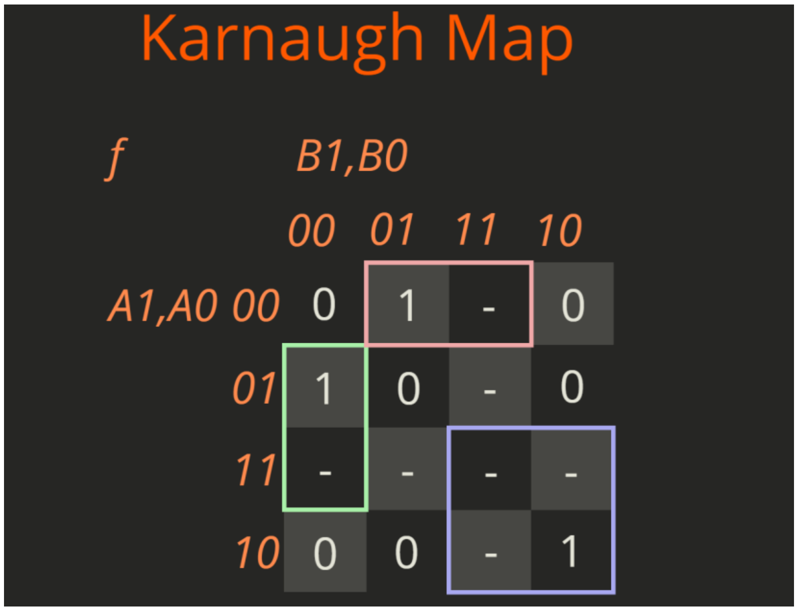
R.2.2 Verilog Code

```
module ternaryadd(c, a0, a1, b0, b1, s0, s1, LED_COM);

input a0, a1, b0, b1;

wire x0, y0, x1, y1;

output s0, s1, c;

inout LED_COM;

assign x0 = !a0;

assign x1 = !a1;

assign y0 = !b0;

assign y1 = !b1;

assign s0 = (x1 & x0 & b0) | (a0 & y1 & y0) | (a1 & b1);

assign s1 = (x1 & x0 & b1) | (a1 & y1 & y0) | (a0 & b0);

assign c = (a0 & b1) | (a1 & b1) | (a1 & b0);

assign LED_COM = 1;

endmodule
```

R.2.3 Vector Waveform



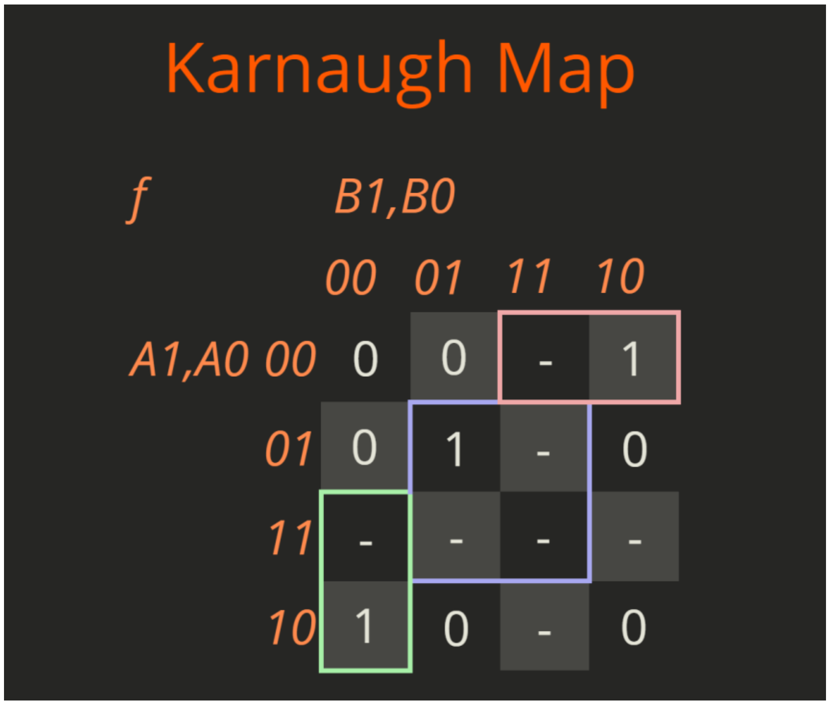
R.2.4 K-Map and Boolean Expression for S0:



A1,A0 \ B1,B0	00	01	11	10
00	0	1	-	0
01	1	0	-	0
11	-	-	-	-
10	0	0	-	1

f(A1, A0, B1, B0) = A1'A0'B0 + A0B1'B0' + A1B1

R.2.5 K-map and Boolean Expression for S1:

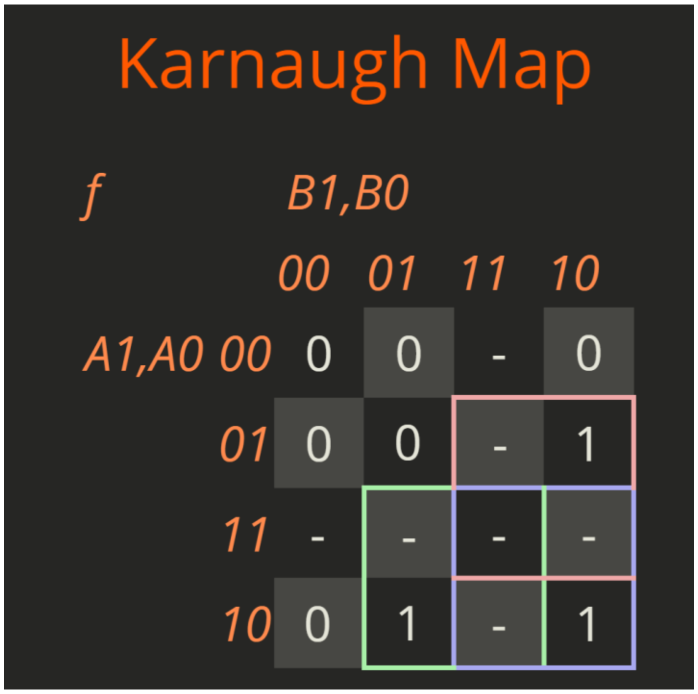


Karnaugh Map

f	B1,B0 00	01	11	10
A1,A0 00	0	0	-	1
01	0	1	-	0
11	-	-	-	-
10	1	0	-	0

f(A1, A0, B1, B0) = A1'A0'B1 + A0B0 + A1B1'B0'

R.2.6 K-map and Boolean Expression for C:



Karnaugh Map

f		B1,B0			
		00	01	11	10
A1,A0	00	0	0	-	0
	01	0	0	-	1
	11	-	-	-	-
	10	0	1	-	1

f(A1, A0, B1, B0) = A0B1 + A1B0 + A1B1