Lab1-1.

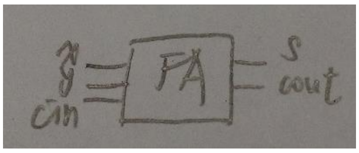
1. Design Specification

a 1-bit full adder

Input : x, y, cin

Output : s, cout

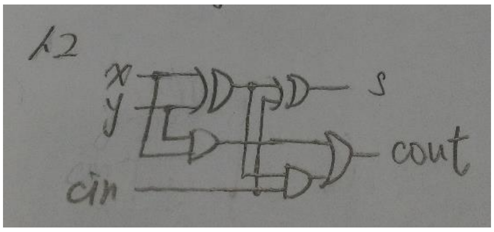
s + cout = x + y + cin



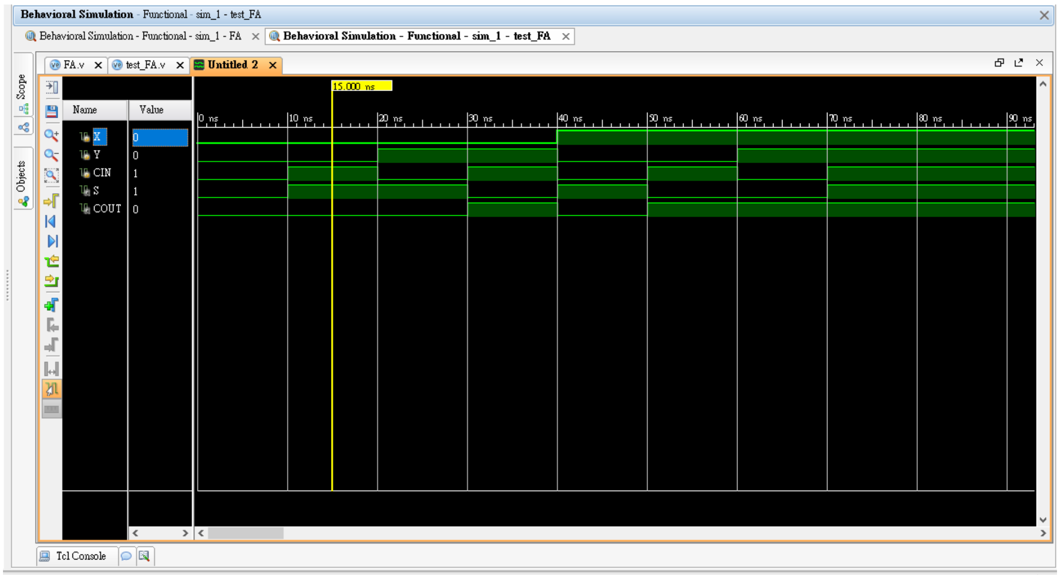
2. Design Implementation

s = x ^ y ^ z

cout = xy + (x ^ y)cin (xy : carry generate, x^y : carry generate)



3. Stimulation


Behavioral Simulation - Functional - sim_1 - test_FA
Behavioral Simulation - Functional - sim_1 - FA
Behavioral Simulation - Functional - sim_1 - test_FA
FA.v
test_FA.v
Untitled 2
Scope
Objects
15.000 ns
Name
Value
X 0
Y 0
CIN 1
S 1
COUT 0
0 ns
10 ns
20 ns
30 ns
40 ns
50 ns
60 ns
70 ns
80 ns
90 ns
Tcl Console

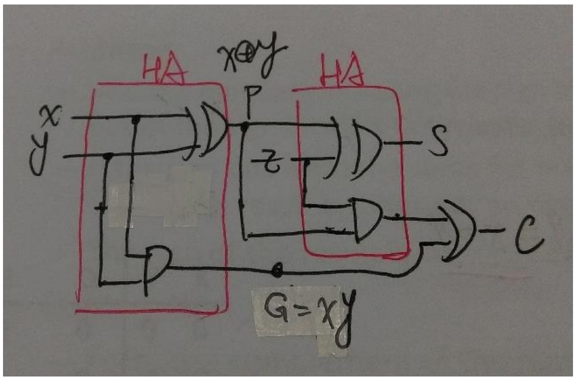
x	y	cin	Z	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0

1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

4. References

Logic Design – Lecture 6 Arithmetic Circuits

It can also implement by half adder and other logic gates



Lab1-2.

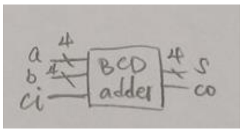
1. Design Specification

a BCD adder

Input : a[3:0], b[3:0], ci

Output : s[3:0], co

{Co, S} = A + B + Ci

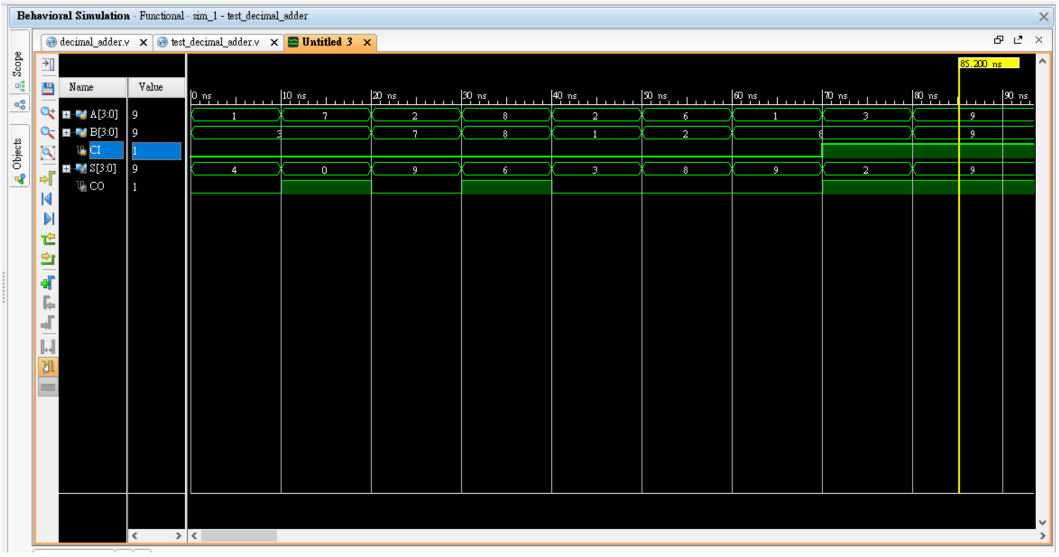


2. Design Implementation

If A+B+Ci <= 9, {Co, S} = A+B+Ci

If A+B+Ci > 9, {Co, S} = A+B+Ci+{0110}

3. Stimulation



a[3:0]	b[3:0]	ci	Z[3:0]	Co
0001(1)	0011(3)	0	0100(4)	0
0111(7)	0011(3)	0	0000(0)	1
0010(2)	0111(7)	0	1001(9)	0
1000(8)	1000(8)	0	0110(6)	1
0010(2)	0001(1)	0	0011(3)	0
0110(6)	0010(2)	0	1000(8)	0
0001(1)	1000(8)	0	1001(9)	0
0011(3)	1000(8)	1	0010(2)	1
1001(9)	1001(9)	1	1001(9)	1

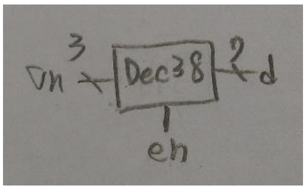
Lab1-3.

1. Design Specification

A 3-to-8 decoder

Input : in[3:0], en

Output : d[7:0]



2. Design Implementation

d[0] = en * ~in[2] * ~in[1] * ~in[0]

d[1] = en * ~in[2] * ~in[1] * in[0]

d[2] = en * ~in[2] * in[1] * ~in[0]

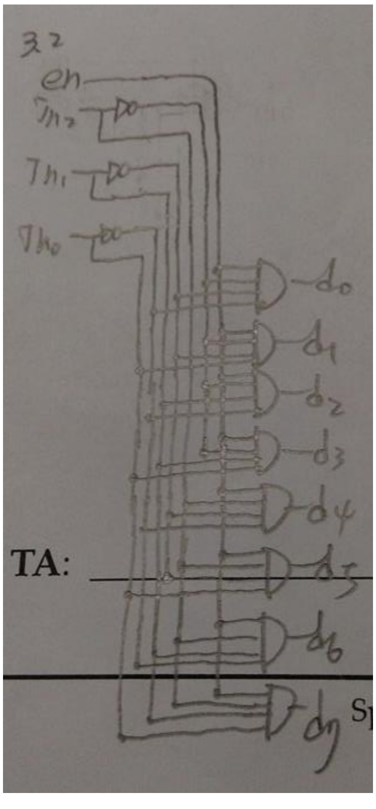
d[3] = en * ~in[2] * in[1] * in[0]

d[4] = en * in[2] * ~in[1] * ~in[0]

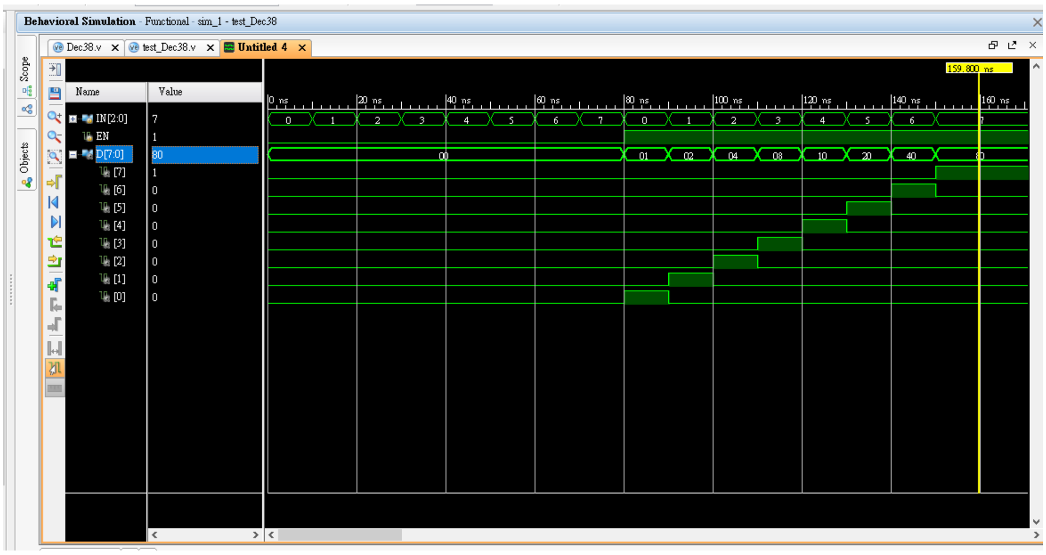
d[5] = en * in[2] * ~in[1] * in[0]

d[6] = en * in[2] * in[1] * ~in[0]

d[7] = en * in[2] * in[1] * in[0]



3. Stimulation



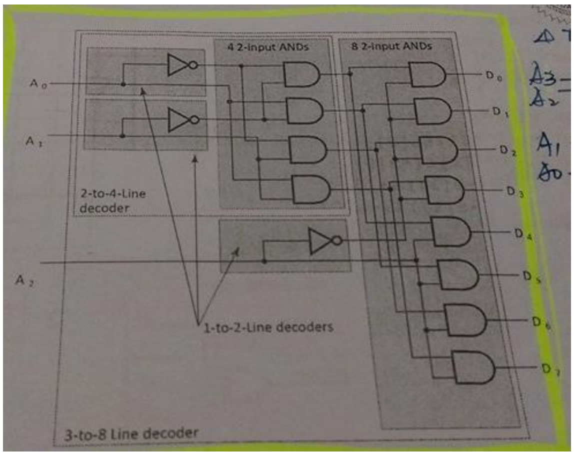
Name	Value
IN[2:0]	7
EN	1
D[7:0]	80
[7]	1
[6]	0
[5]	0
[4]	0
[3]	0
[2]	0
[1]	0
[0]	0

In[3:0]	en	D[7:0]
000(0)	0	00000000
001(1)	0	00000000
010(2)	0	00000000
011(3)	0	00000000
100(4)	0	00000000
101(5)	0	00000000
110(6)	0	00000000
111(7)	0	00000000
000(0)	1	00000001
001(1)	1	00000010
010(2)	1	00000100
011(3)	1	00001000
100(4)	1	00010000
101(5)	1	00100000
110(6)	1	01000000
111(7)	1	10000000

4. References

Logic Design – Lecture 5 Combinational Building Blocks

It can also implement by 2-to-4 decoder and AND gates


4 2-input ANDs
8 2-input ANDs
A0
A1
A2
D0
D1
D2
D3
D4
D5
D6
D7
2-to-4-Line decoder
1-to-2-Line decoders
3-to-8 Line decoder
A3
A2
A1
A0