

2016 Logic Design HW6

1.

	1's complement	2's complement
1110_0001	0001_1110	0001_1111
1010_1011	0101_0100	0101_0101
0111_0011	1000_1100	1000_1101
1110_0111	0001_1000	0001_1001
0101_0101	1010_1010	1010_1011

2.

(a) 2 補數加法

```

0101
+1100
-----

```

10011 (左一丟掉) A: 0011 (2 補數表示法)

(b) 2 補數加法

```

0111010
+1101011
-----

```

10100101 (左一丟掉) A: 0100101 (2 補數表示法)

3.

(a) 0101-0110

先將-0110(unsigned)轉 2's comp

Step1. 0110=>1010

Step2. 0101+1010=1111

Ans: 1111(2's comp)=-0001(unsigned)

(b)10110-1100

先將兩邊補至同 bit 數，1100=>01100，再兩邊數值加上 extend bits，
10110=>010110，01100=>001100。式子變成:010110-001100(unsigned)

Step1. -001100 轉成(2's comp) ; 110100(2's comp)

Step2. 010110+110100=1001010(最前面 1 是 overflow 捨去，計算結果應為
6 bits)

Ans:001010=>1010(2's comp or unsigned)

(c)1011110-1111110

補上 extend bits，1011110=>01011110，1111110=>01111110

Step1. 01111110 轉成 2's comp ; -01111110=>10000010

Step2. 01011110+10000010=11100000(2's comp)

Ans: 100000(2's comp) = - 100000 (unsigned)

(d)101001-101

先將兩邊補至同 bit 數，101=>000101，再兩邊數值加上 extend bits，
101001=>0101001，000101=>0000101。式子變成:0101001-0000101(unsigned)

Step1. -0000101 轉成 2's comp ; -0000101=>1111011

Step2. 0101001+1111011=10100100(最前面 1 是 overflow 捨去，計算結果應
為 7 bits)

Ans: 0100100

4.

(a)

2's complement of B = $\sim B + 1$

$$-(0110)_2 = (\underline{0110})_2 + (1)_2 = (1001)_2 + (1)_2 = (1010)_{2's\ complement}$$

$$(0101)_{2's\ complement} - (0110)_{2's\ complement} = (0101)_2 + (1010)_2 =$$

$(1111)_2$

No overflow

$$(5)_{10} - (6)_{10} = (-1)_{10}$$

(b)

Sign extension: $(1100)_{2's\ complement} \rightarrow (11100)_{2's\ complement}$

$$-(11100)_2 = (\underline{11100})_2 + (1)_2 = (00100)_2$$

$$(10110)_2 - (11100)_2 = (10110)_2 + (00100)_2 = (11010)_{2's\ complement}$$

No overflow

$$(-10)_{10} - (-4)_{10} = (-6)_{10}$$

(c)

$$-(1111110)_2 = (\underline{1111110})_2 + (1)_2 = (0000010)_{2's\ complement}$$

$$(1011110)_2 - (1111110)_2 = (1011110)_2 + (0000010)_2 = (1100000)_2$$

No overflow

$$(-34)_{10} - (-2)_{10} = (-32)_{10}$$

(d)

Sign extension: $(101)_2 \rightarrow (111101)_{2's\ complement}$

$$-(111101)_2 = (\underline{111101})_2 + (1)_2 = (000010)_2 + (1)_2 = (000011)_{2's\ c}$$

No overflow

$$(101001)_2 - (111101)_2 = (101001)_2 + (000011)_2 = (101100)_{2's\ c}$$

$$(-23)_{10} - (-3)_{10} = (-20)_{10}$$

5.

(a)

$0_101 - 0_110$ ($5 - 6$)
0110 轉 2's complement $\rightarrow 1010$
 $0101 + 1010 = 1111 \rightarrow -1$
 \rightarrow No overflow

(b)

$1_0110 - 1_100$ [$-6 - (-4)$]
6 (0110)轉 2's complement $\rightarrow 1010$
 $1010 + 0100 = 1110 \rightarrow -2$
 \rightarrow No overflow

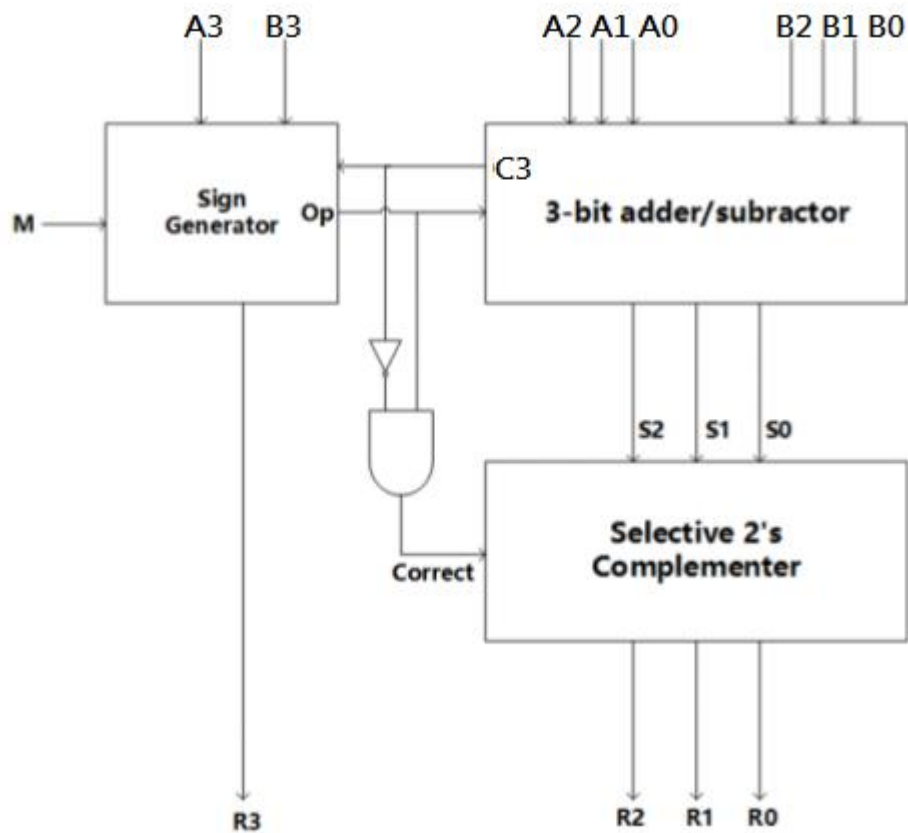
(c)

$1_011110 - 1_111110$ [$-30 - (-62)$]
30 (011110)轉 2's complement $\rightarrow 100010$
 $1100010 + 0111110 = 1_0100000 \rightarrow 32$
 \rightarrow No overflow

(d)

$1_01001 - 1_01$ [$-9 - (-1)$]
9 (01001)轉 2's complement $\rightarrow 10111$
 $10111 + 00001 = 11000 \rightarrow -8$
 \rightarrow No overflow

6.



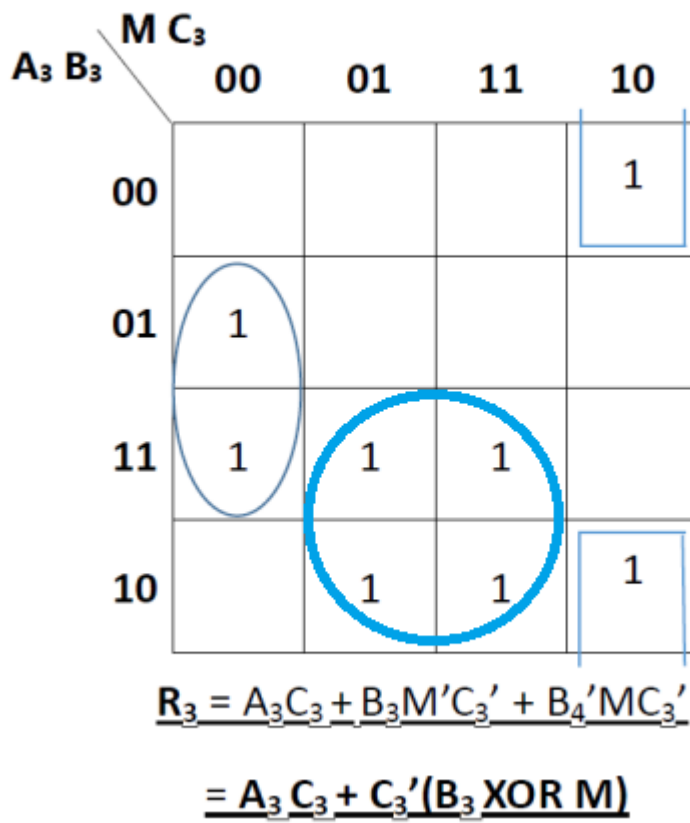
- M is the operation we want to do ($M=1$ is subtraction, $M=0$ is addition).
- A_3 and B_3 are the sign bits of the signed numbers. C_3 is the carry out from circuit.
- Op is the signal that determines the real operation (add or subtract) depending on the signs A_3 and B_3 as well as from the input M . After we have checked the signs and carry we can determine the final sign (R_3) for the addition/subtraction

A ₃	B ₃	M	C ₃	Op	R ₃
0	0	0	0	0	0
0	0	0	1	0	0
0	0	1	0	1	1
0	0	1	1	1	0
0	1	0	0	1	1
0	1	0	1	1	0
0	1	1	0	0	0
0	1	1	1	0	0
1	0	0	0	1	0
1	0	0	1	1	1
1	0	1	0	0	1
1	0	1	1	0	1
1	1	0	0	0	1
1	1	0	1	0	1
1	1	1	0	1	0
1	1	1	1	1	1

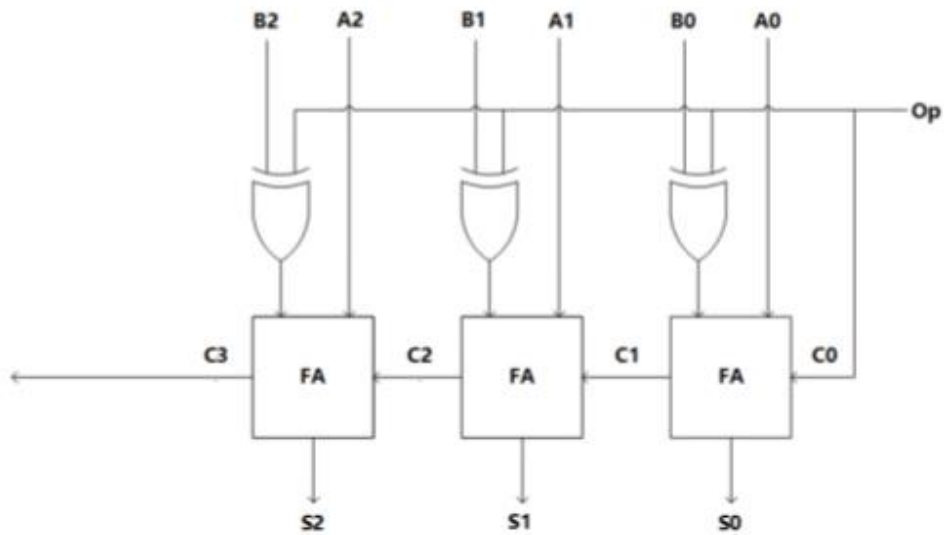
A ₃ B ₃		M C ₃			
		00	01	11	10
00			1	1	
01	1	1			
11			1	1	
10	1	1			

$$\underline{\text{Op} = A_3 B_3' M' + A_3 B_3 M + A_3' B_3 M' + A_3' B_3' M}$$

$$\underline{= A_3 \text{ XOR } B_3 \text{ XOR } M}$$

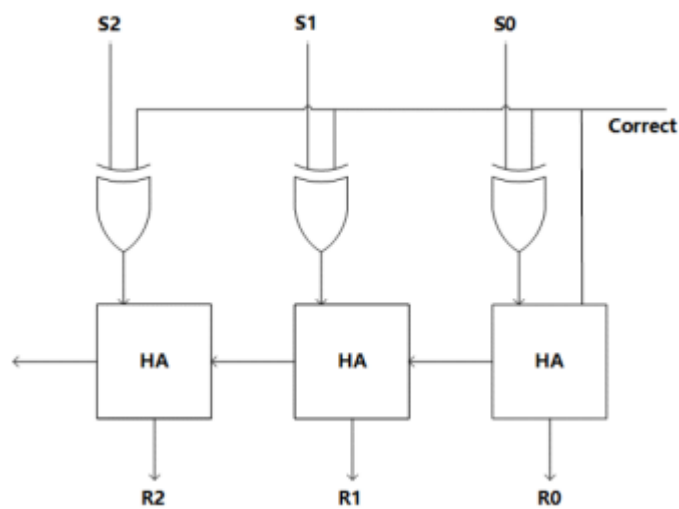


3-bit Adder/subtractor



Selective 2's complementer

- We only want to take the 2's complement when needed. So we use the correct line which is 1 if the carry is zero and a subtract operation is used. If correct is 1 it complements and adds 1 to the input, else it does nothing and the output equals the input.



7.

先畫出真值表

z ₃	z ₂	z ₁	z ₀	Z ₃	Z ₂	Z ₁	Z ₀
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	1
0	1	0	0	0	1	0	0
0	1	0	1	0	1	0	1
0	1	1	0	0	1	1	0
0	1	1	1	0	1	1	1
1	0	0	0	x	x	x	x
1	0	0	1	0	1	1	1
1	0	1	0	0	1	1	0
1	0	1	1	0	1	0	1
1	1	0	0	0	1	0	0
1	1	0	1	0	0	1	1
1	1	1	0	0	0	1	0
1	1	1	1	0	0	0	1

由真值表畫出 K-map，並求出 logic function。

$$Z_3 = 0$$

$$Z_2 = z_3'z_2 + z_3z_2'$$

z ₁ z ₀ \z ₃ z ₂	00	01	11	10
00		1		1
01		1		1
11		1		1
10		1		1

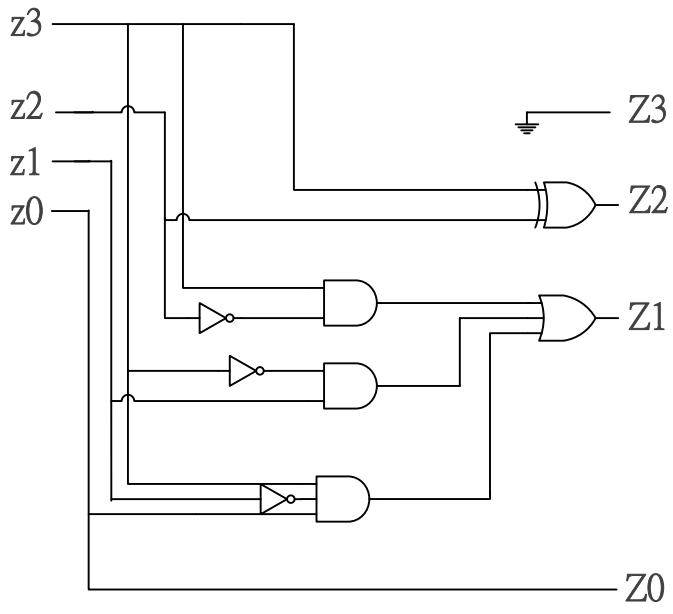
$$Z_1 = z_3z_2' + z_3'z_1 + z_3z_1'z_0$$

z ₁ z ₀ \z ₃ z ₂	00	01	11	10
00				
01			1	1
11	1	1		
10	1	1	1	1

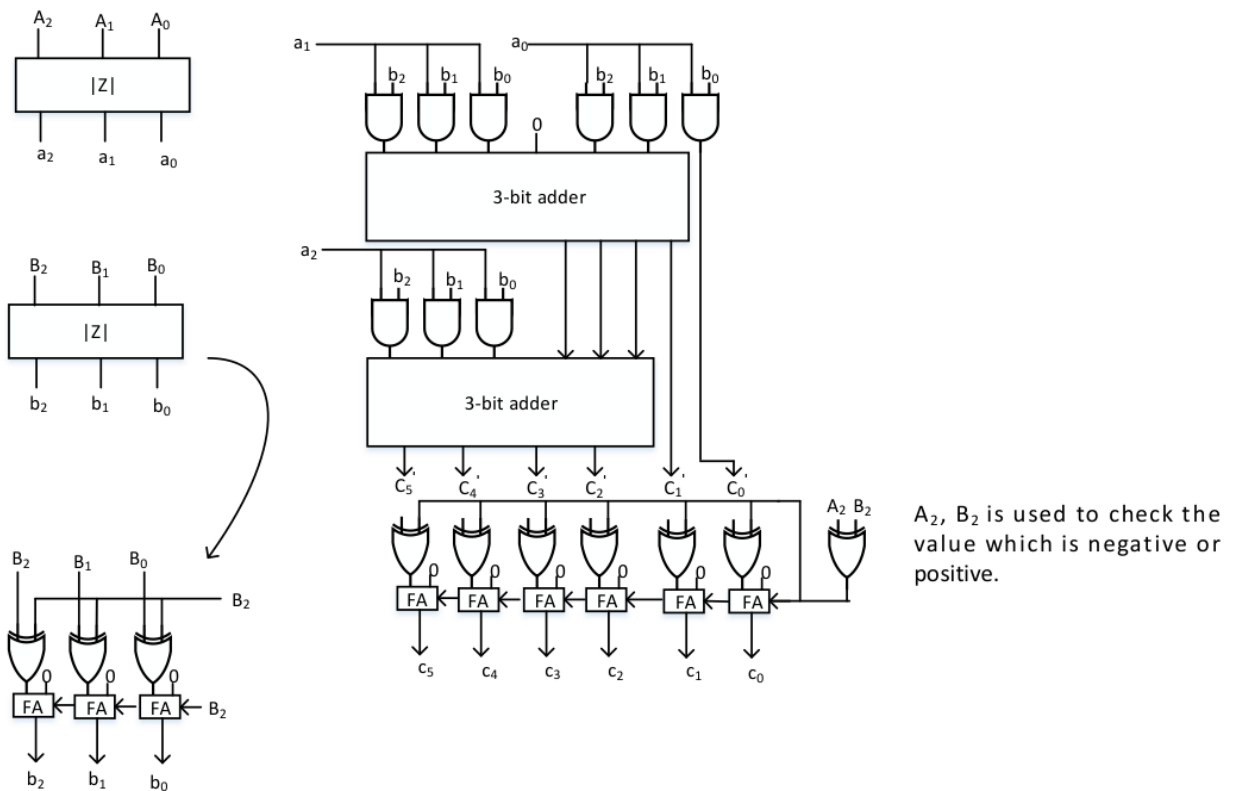
$$Z_0 = z_0$$

z ₁ z ₀ \z ₃ z ₂	00	01	11	10
00				
01	1	1	1	1
11	1	1	1	1
10				

畫出 logic diagram

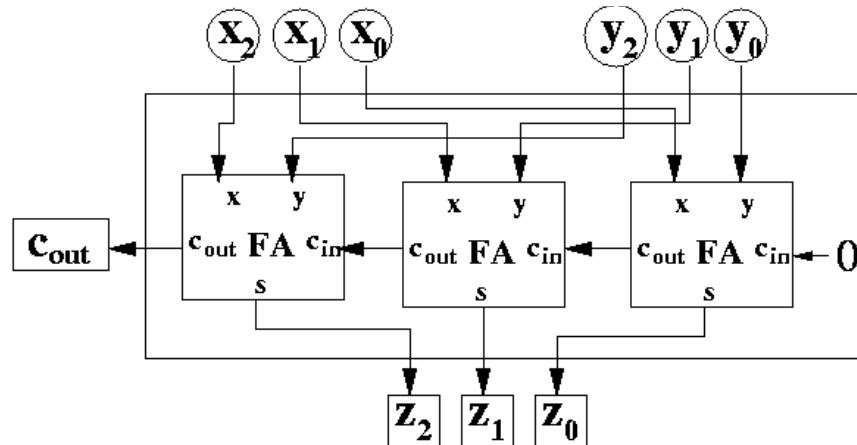


8.



A_2, B_2 is used to check the value which is negative or positive.

其中 3-bit adder



首先 先將 A 和 B 取絕對值，變成 unsigned numbers $a_0a_1a_2$ 及 $b_0b_1b_2$ ，利用加法做乘法運算後得到 c，最後再利用 $A_2 \text{ xor } B_2$ 判斷為正負值。

9.

比較大小取 4bits unsigned $A > B$ 的結果:

先看 1 bit 比較大小， $A_0 > B_0$ 則 $A_0 \& (B_0)' = 1$

然後可知:

$$\text{Equation: } A_3(B_3)' + x_3A_2(B_2)' + x_3x_2A_1(B_1)' + x_3x_2x_1A_0(B_0)'$$

Logic diagram:

Solution 取 $A > B$ 的位置。

