數位邏輯設計-HW5 鄭凱文 104062223

HW5-1:

Decoder: input多少，output的1就往左移多少bit

6-64→3-8:

首先把6-bit的input平分成兩段，分別透過decoder 0 和decoder 1得到兩個8-bit的one-hot，X & Y。對最後的output來說：

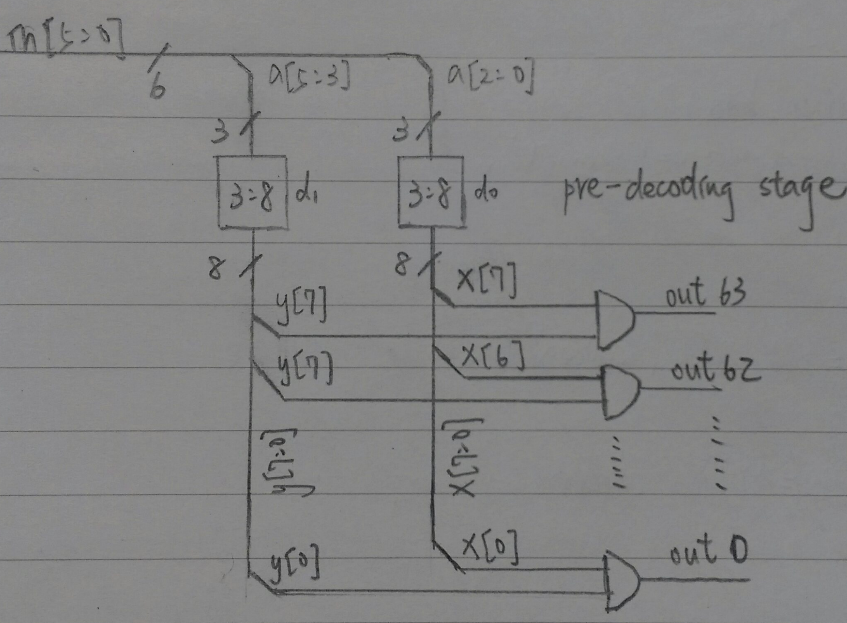
X[j] = j;

Y[i] = 8 \* i;

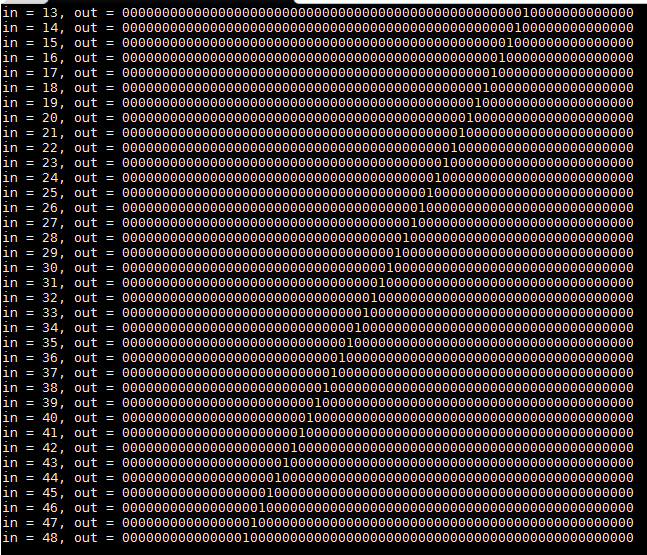
Out[i\*8+j] = Y[i] & X[j]，if input = (i\*8 + j)，則output[i\*8+j] = 1

Testbench選擇：從0~63全部跑一遍

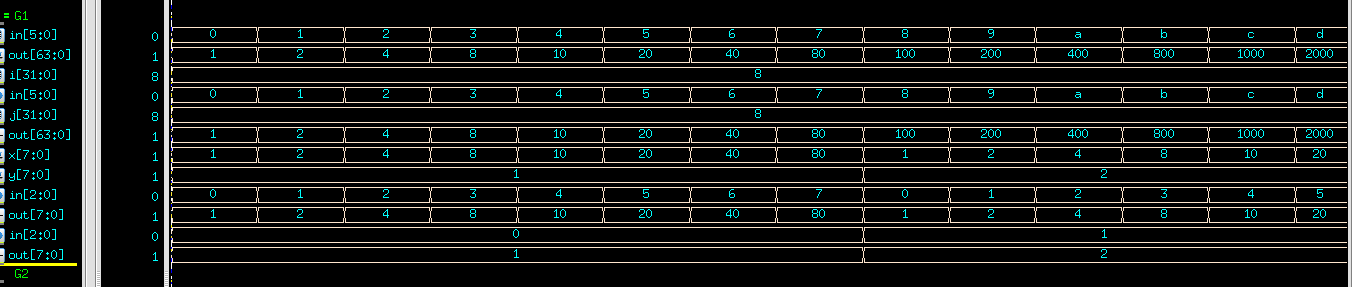
Logic diagram:



Simulation:



波形圖：



HW5-2:

Binary Priority Encoder:

為了rotate回來方便，取in\_d[15:0] & pri\_d[3:0]:

in\_d = {in, in} ←延伸為兩倍的input，如此一來操作時不用rotate回in[0]

pri\_d = {1’b0, pri} ←變成3-bit，方便之後作加法進位

利用for迴圈，從pri + 8(即在延伸的in\_d的原本pri位置)開始，依序向右找第一個出現的1(i=i-1)，若有找到則zero = 0，剩餘的for迴圈將自動跳出

for(i=8;i>=1&&zero!=0;i=i-1) //start from pri, right forward, if zero==0 then break.

if(in\_d[pri\_d+i] == 1'b1)

out = pri + i;

zero = 1'b0;

output即為從pri開始，往右找到的第一個1的位置。

Testbench選擇：

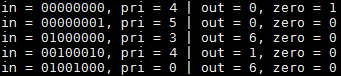
in = 8'b00000000, pri = 4: 測試input==0時，output = 0, zero = 1

in = 8'b00000001, pri = 5: 測試input==1時，output = 0, 但zero = 0(代表其不為0)

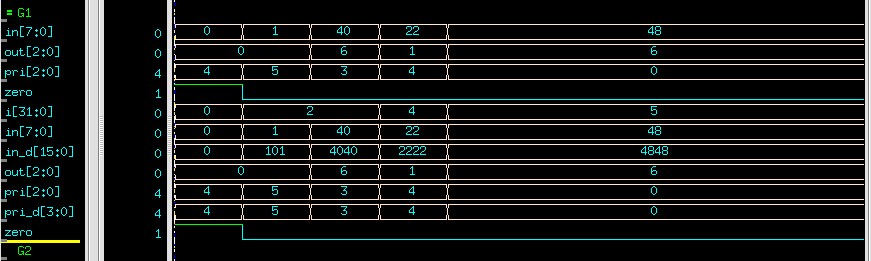
in = 8'b01000000, pri = 3: 測試會不會正確的rotate回來

其他兩個測資為input中有多個1的情況。

Simulation:



波形圖：



HW5-3:

16-bit的input，拆成兩部分放入encoder:

使用兩個encoder，input[7:0]放入enc0，input[15:8]放入enc1，而傳入的pri必須是3-bit。

pri的選擇：如果pri<=7，代表從input[7:0]開始找起，再”rotate”回input[15:8]內找，因此enc1的pri應一律設為”7”(等於從input[15]開始找)；反之若pri>7，代表enc0的pri也應設為”7”(從input[7]開始找)。

wire tmp\_pri = 7 – pri[2:0] ←取7和pri%7的相差值，方便待會補成7

另外zero\_1, zero\_2用來判斷各自部分是否皆為零，zero = zero\_1 & zero\_2。

因此：

hw5\_2 enc0(in[7:0], tmp\_pri[2:0]\*pri[3]+pri[2:0], out\_1, zero\_1) ←若pri[3]=1，代表pri>7，enc0應從in[7]找起(tmp\_pri \* 1 + pri[2:0] = 7)

hw5\_2 enc1(in[15:8], tmp\_pri[2:0]\*~pri[3]+pri[2:0], out\_2, zero\_2) ←若pri[3]=0，代表pri<=7，enc1應從in[15]找起(tmp\_pri \* ~0 + pri[2:0] = 7)

再來是output的判斷：

如果zero==1，代表input為0。如果zero!=1，則有pri<=7 or >7兩種可能。

若If pri<=7，則優先從out\_1考慮：

1. out\_1<=pri && zero\_1!=1: 代表out\_1在pri的”右邊”，output即為out\_1(zero\_1!=1表示排除input[7:0]==0的可能)
2. out\_1>pri && zero\_2==0: 雖然out\_1在pri的”左邊”，代表有rotate過，但由於zero\_2==0 → input[15:8]==0，因此output仍為out\_1
3. out\_1>pri && zero\_2!=0: out\_1在pri的”左邊”，且input[15:8]!=0，代表有rotate過且在input[15:8]內已經碰到1，因此output=out\_2 + 8(out\_2原為對8取的餘數，輸出時應補回8)

反之亦然，if pri>7，只有在out\_2>pri && zero\_1!=0時才考慮out\_1。

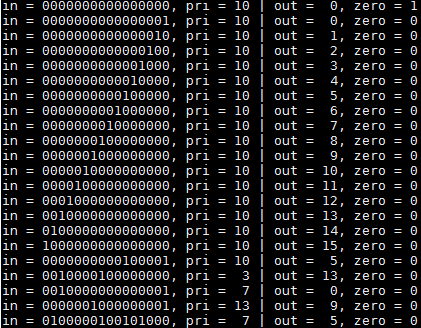
Testbench選擇：

1. Input=0, 16-bit全部的one-hot, pri=10: 全部跑一遍確認程式無誤，以及input=0時zero是否為1。
2. Input=16'b0000000000100001, pri = 10; in = 16'b0010000100000000, pri = 3:

確認如果碰到上述狀況(3)的時候，1集中在其中一側，pri在另一側時，是否可以得出正確結果。

1. 其他為pri介於input內1的中間，或是input中有多個1的情況。

Simulation:



波形圖：

