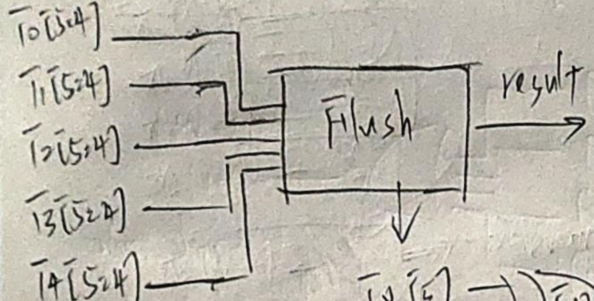
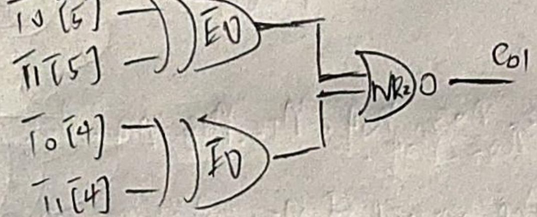
module : Flush



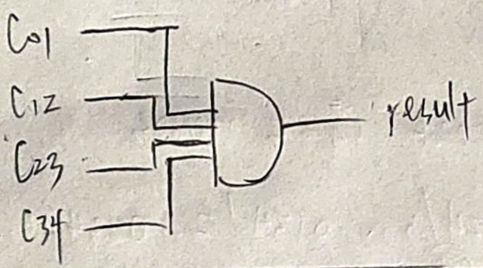
result=1 ⇒ suit is the same

result=0 ⇒ suit is different

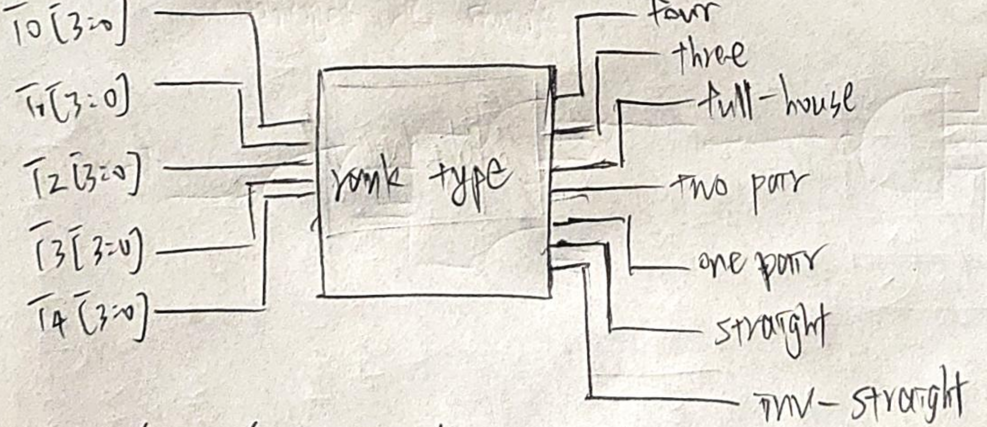
for suit of T0.T1



{ 1: same suit
0: different suit



module : rank type.

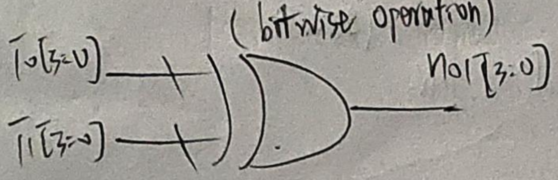


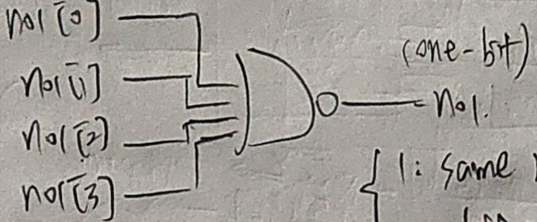
if four/three/full-house/two pair/one pair/straight =1
⇒ denoting the corresponding type

inv-straight = (straight)'

first need to check $C_2^5=10$ pair, if the rank of each pair is same

for example: check if $T_0[3:0]$, $T_1(3:0)$ have the same rank.





{ 1: same rank
  0: different rank.

⇒ total have n_{01}, n_{02}, n_{03}, n_{04}, n_{12}, n_{13}, n_{14}, n_{23}, n_{24}, n_{34} = 10 result

⇒ inv-n_{ij} denoting the inverse of n_{ij}. that is inv-n_{01} { 1: different rank
0: same rank.

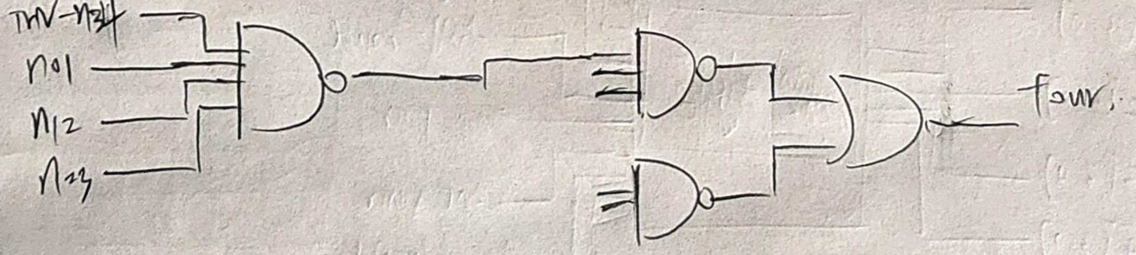
(n_{ij} —▷o— inv-n_{ij})

※ for output "four" ⇒ check $C_4^5=5$ cases

that is: 0123/4, 0124/3, 0134/2, 0234/1, 1234/0

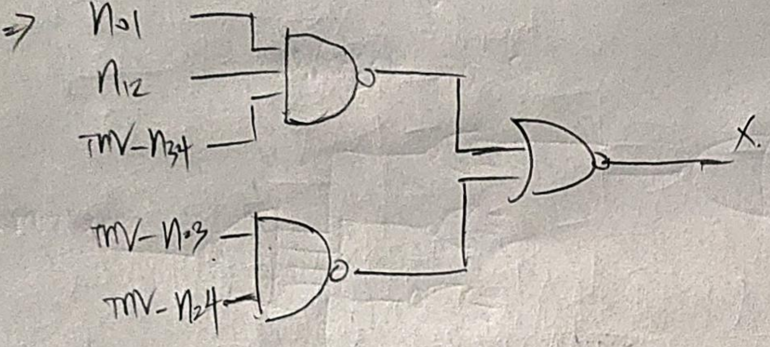
for example: 0123/4:

after get all result of 5 cases
⇒ OR together.

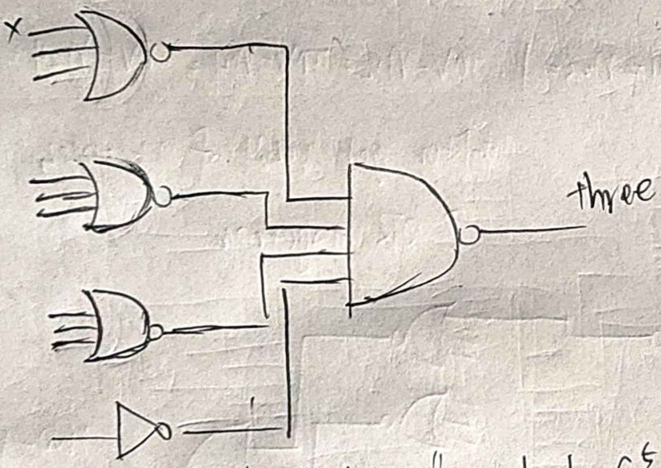


※: for output "three" ⇒ check $C_3^5 = 10$ case

⇒ for example : 012/3/4 which representing $n_{01} = n_{12} = 1$: inv-n_{34} = 1.
inv-n_{23} = inv-n_{24} = 1



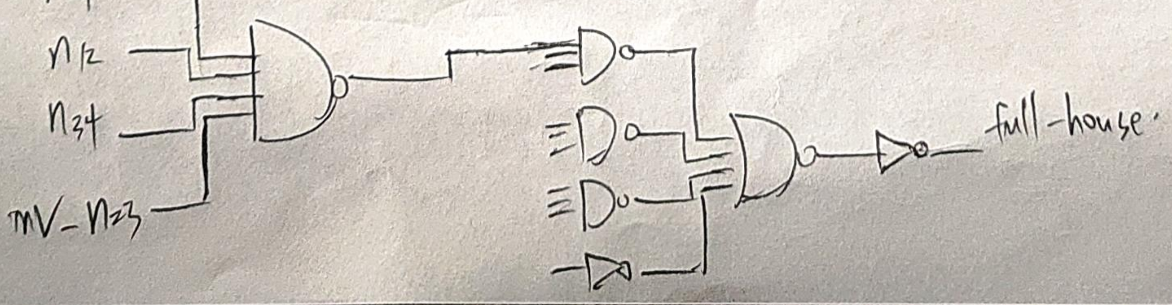
after get all result of 10 cases ⇒ OR together.



※: for output "full-house" ⇒ check $C_3^5 = 10$ cases

for example : 012/34 ⇒ $n_{01} = n_{12} = n_{34} = 1$. inv-$n_{23}$ = 1

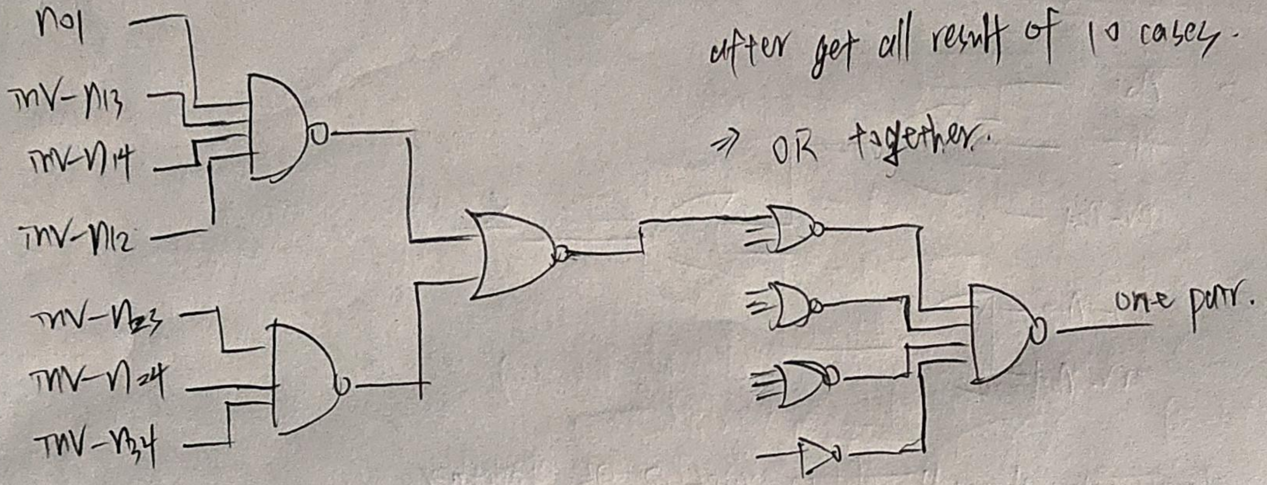
after get all result of 10 cases ⇒ OR together



※: for output "one-pair" ⇒ check $C^5_2 = 10$ cases.

for example.: 01/2/3/4 ⇒ $n_{01} = 1$. INV-n_{12} = INV-n_{13} = INV-n_{14} = 1

INV-n_{23} = INV-n_{24} = INV-n_{34} = 1



after get all result of 10 cases.

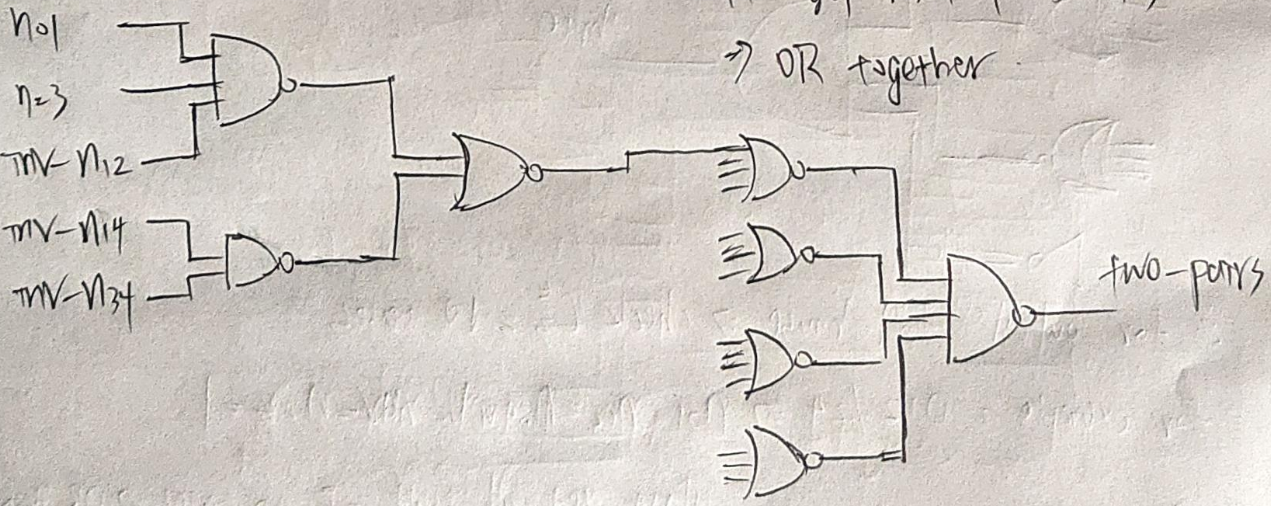
⇒ OR together.

※: for output "two-pair" ⇒ check $C^5_2 \times C^3_2 / 2 = 15$ cases.

for example: 01/23/4 ⇒ $n_{01} = n_{23} = 1$. INV-n_{12} = INV-n_{14} = INV-n_{34} = 1

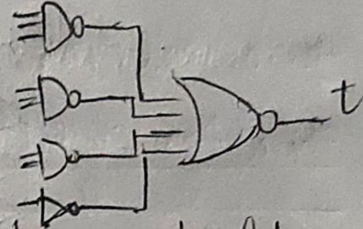
after get result of 15 cases

⇒ OR together



※ for output "straight" ⇒ 12345 → 10 11 12 13 1 have 10 cases

First need to check total 10 inv-nij = 1 ⇒
(because five cards have different rank).



for example {1 2 3 4 5}. check if every cards is in the list

truth table : if T0 [3=0] is in the list {1-2-3-4-5}

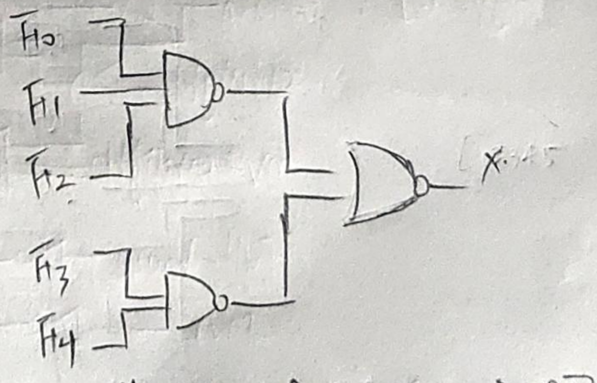
	X3	X2	X1	X0	Ti = 1
(0)	0	0	0	0	X
(1)	0	0	0	1	1
(2)	0	0	1	0	1
(3)	0	0	1	1	1
(4)	0	1	0	0	1
(5)	0	1	0	1	1
(6)	0	1	1	0	0
(7)	0	1	1	1	0
(8)	1	0	0	0	0
(9)	1	0	0	1	0
(10)	1	0	1	0	0
(11)	1	0	1	1	0
(12)	1	1	0	0	0
(13)	1	1	0	1	0
(14)	1	1	1	0	X
(15)	1	1	1	1	X

K-map ⇒

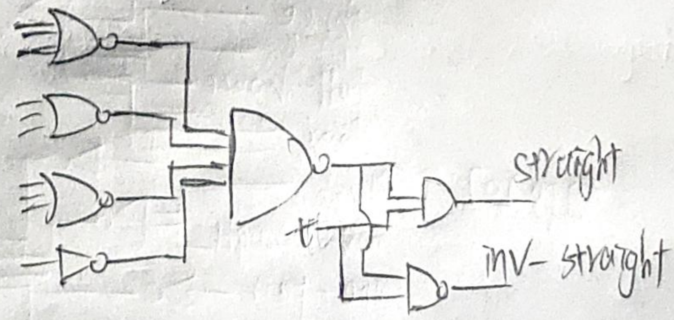
X1X0 \ X3X2	00	01	11	10
00	X	1	0	0
01	1	1	0	0
11	1	0	X	0
10	1	0	X	0

$\Rightarrow Ti = X_3' X_2' + X_3' X_1'$

for every T0 . T1 . T2 . T3 . T4 . result



after get all result of 10 cases ⇒ OR.

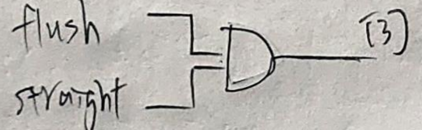


module : card type

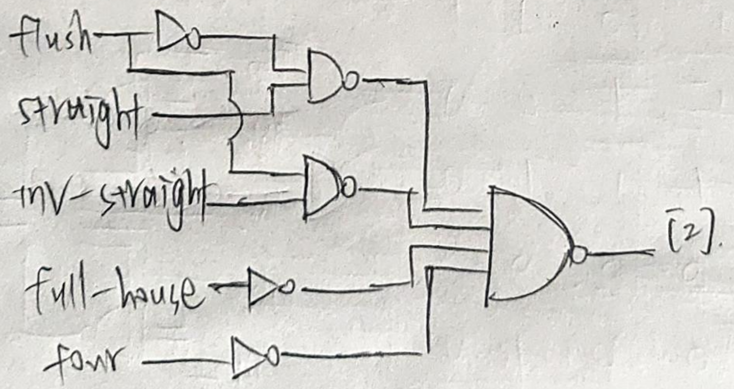
truth table : (not complete)

							type			
Flush.	four	three	full-house	two-pair	one-pair	straight	[3]	[2]	[1]	[0]
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	1
0	0	0	0	1	0	0	0	0	1	0
0	0	1	0	0	0	0	0	0	1	1
0	0	0	0	0	0	1	0	1	0	0
1	0	0	0	0	0	0	0	1	0	1
0	0	0	1	0	0	0	0	1	1	0
0	1	0	0	0	0	0	0	1	1	1
1	0	0	0	0	0	1	1	0	0	0

⇒ type [3] :

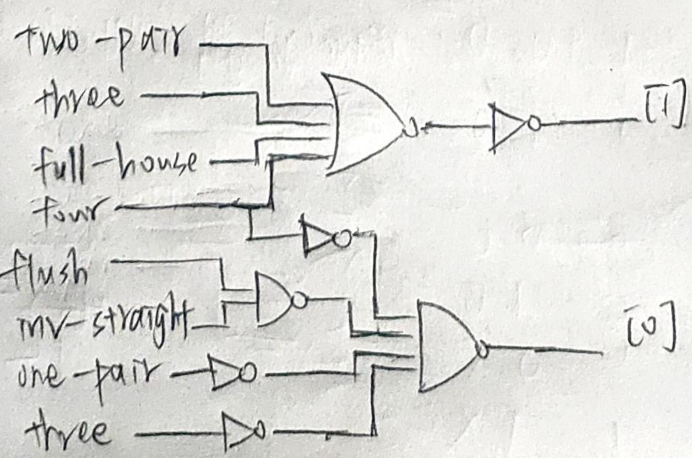


type [2] :



type [1] :

type [0]



Discussion: how to improve critical path

1. 盡量把 AND 和 OR gate 轉換成 NAND 和 NOR gate。

2. 因為 delay 取決於路徑上 gate 數量，盡量不要疊太多的 gate。

3. 把所有可能的 case 都攤開來平行的做，可以減少 delay。

4. 如果 AND 或 OR 的 input 太多的話，拆成多級，每個 gate input 平均且少

   Ex OR 15 -> NOR4 * 3 + NOR3 + NAND4

   OR 10 -> NOR3 * 3 + NAND4