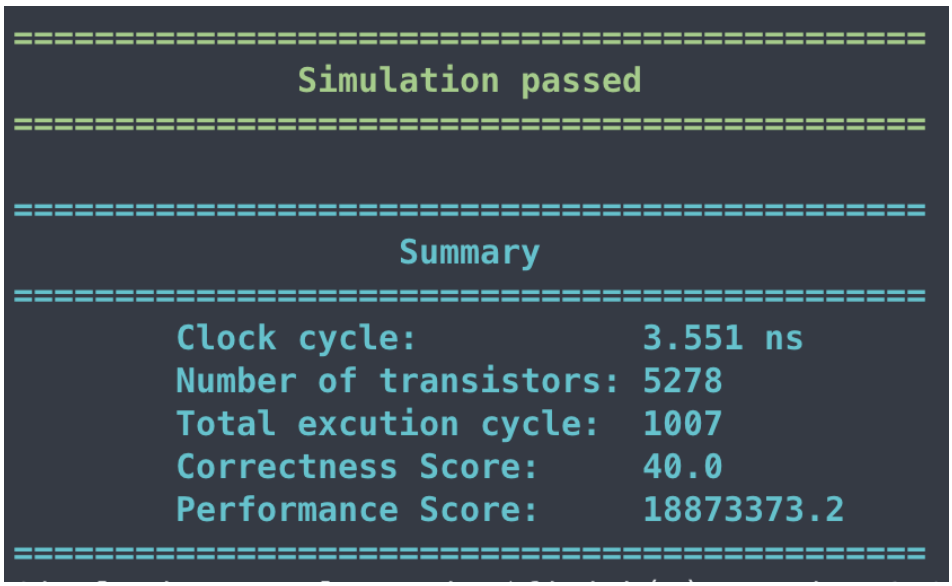
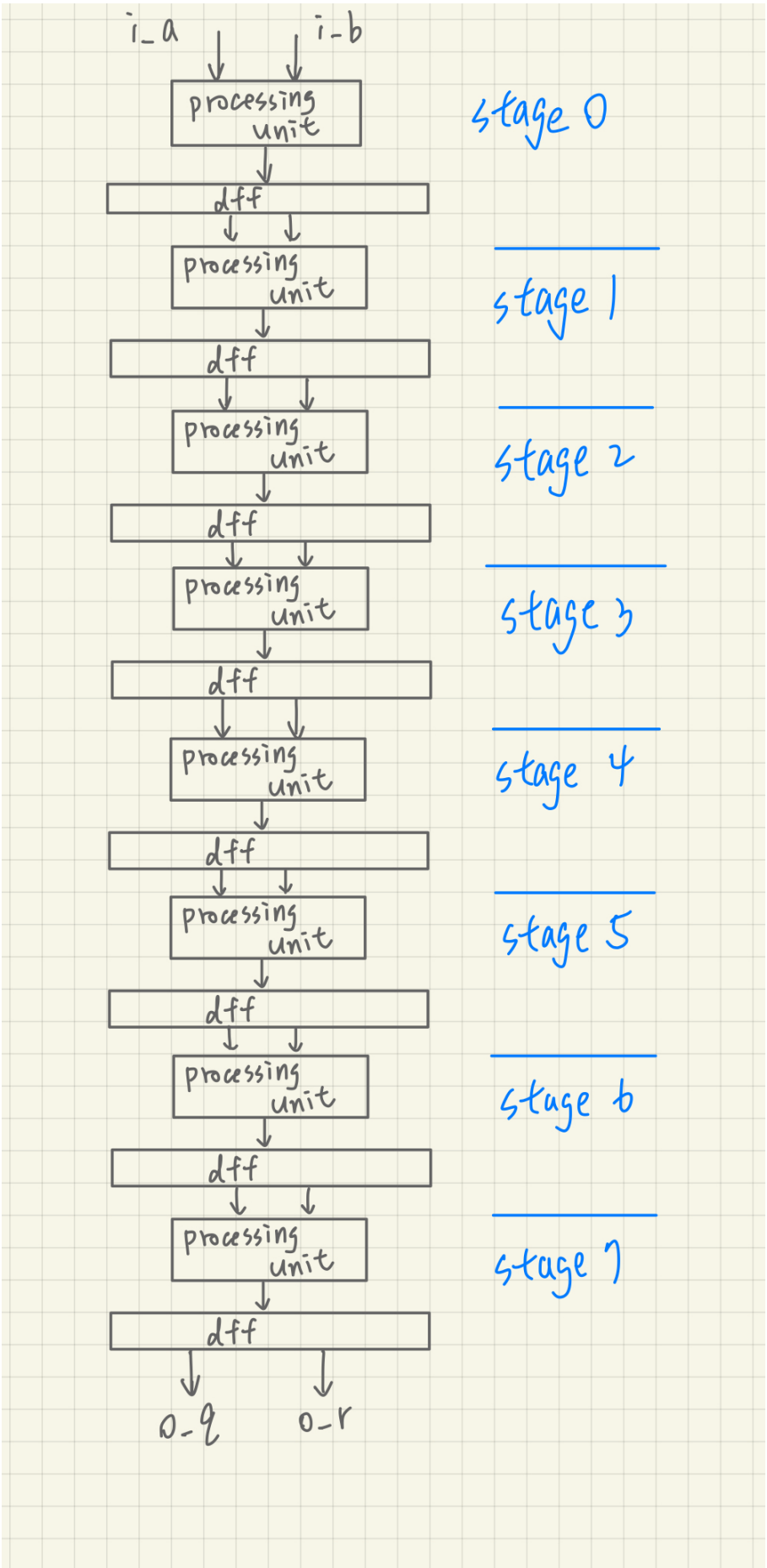
IC Design hw4

電機三 b08901048 陳宥辰

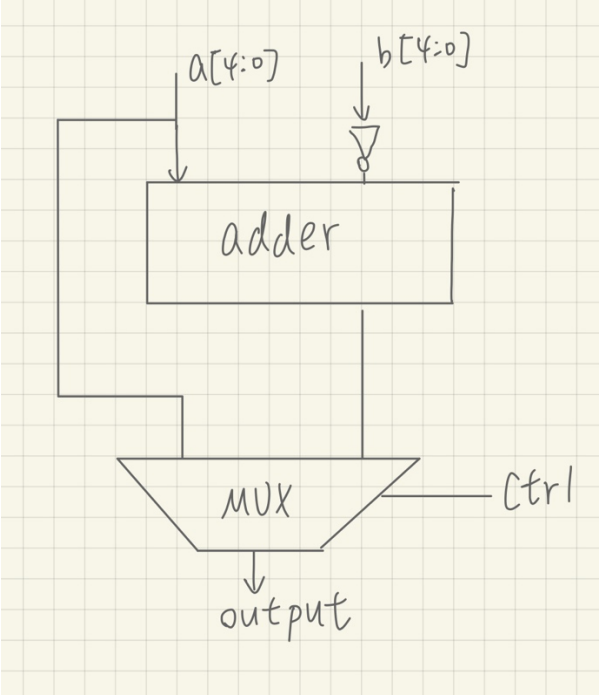


(b) circuit diagram

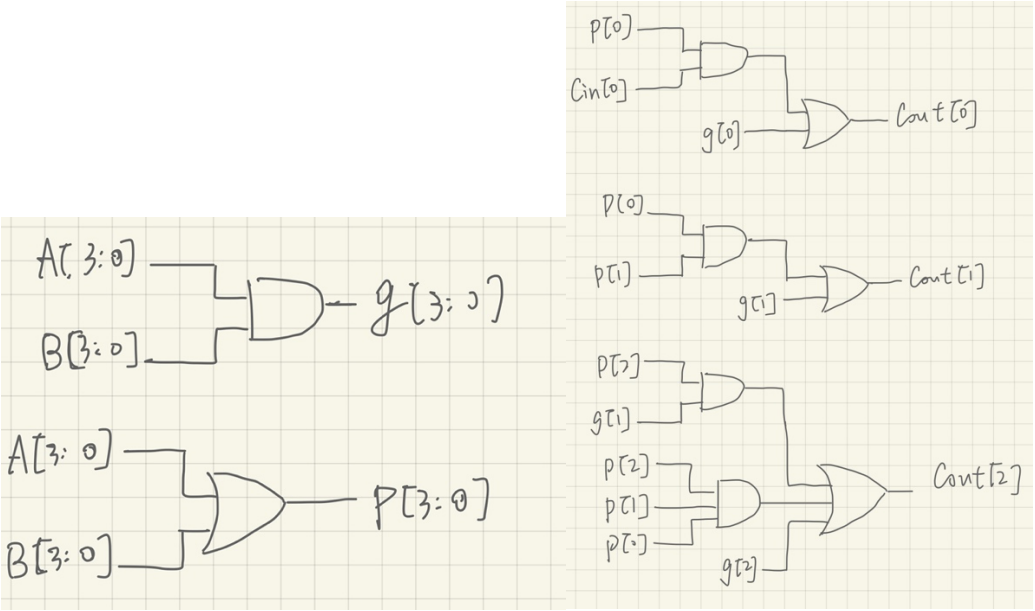
此次我用的是 PIPELINE 的策略，而我的電路主架構如下，分成 8 個 stage，每個 stage 是由一個 processing unit + d-flip flop 所組成，input 為 i_a 和 i_b，經過這 8 個 stage 最後會 output o_q 和 o_r

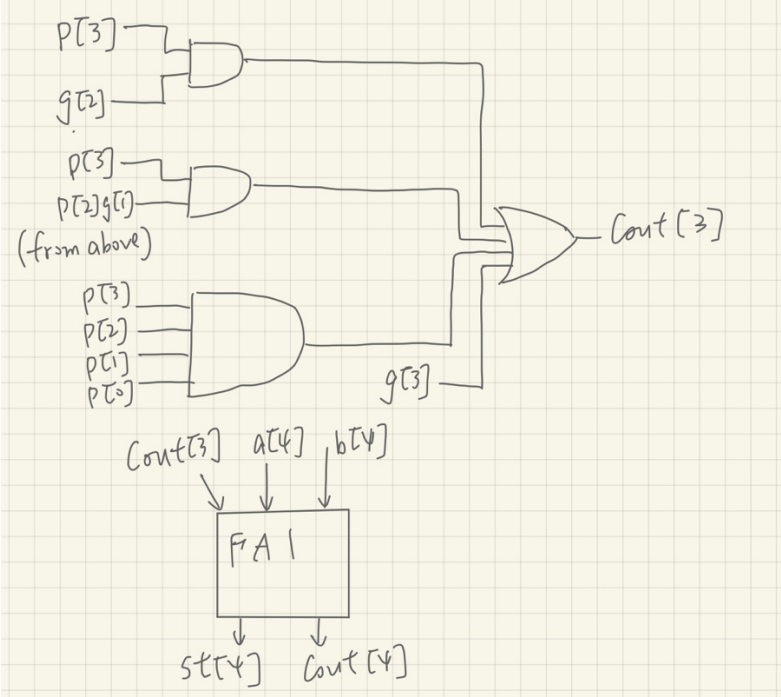

i_a
i_b
processing unit
stage 0
dff
processing unit
stage 1
dff
processing unit
stage 2
dff
processing unit
stage 3
dff
processing unit
stage 4
dff
processing unit
stage 5
dff
processing unit
stage 6
dff
processing unit
stage 7
dff
o_q
o_r

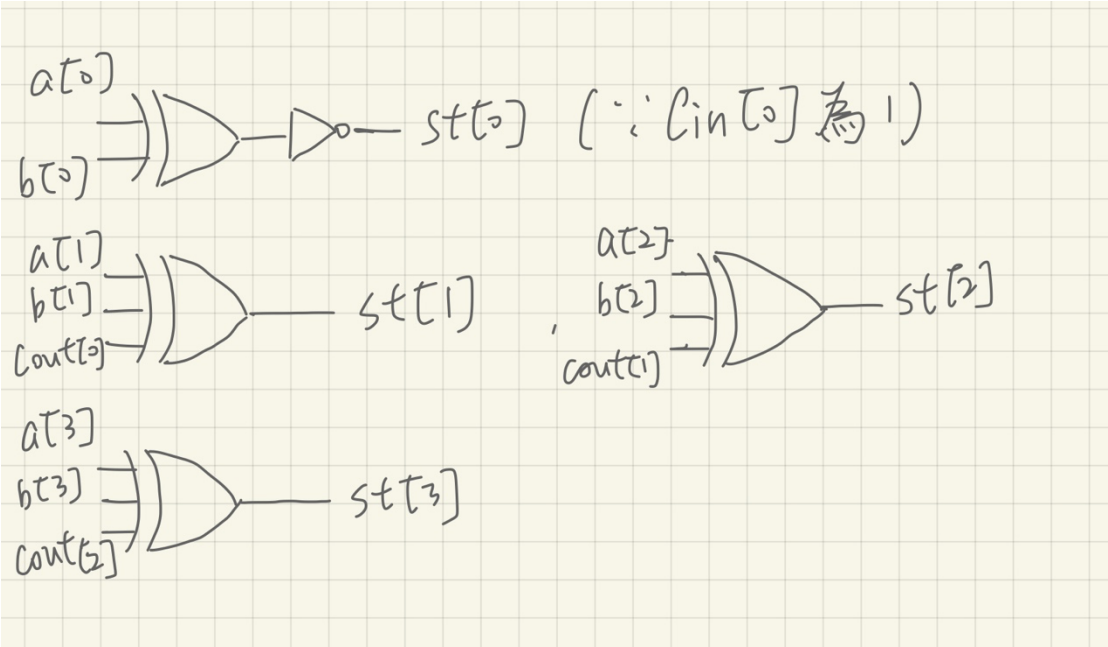
而在每一個 processing unit 中，架構如下：
a 是該 stage 要運算的 input(包含上一 stage 的 output[3:0]和 i_a 的其中一位)
b 為 divisor，先讓 b 通過 inverter 是為了要使 adder 產生 subtractor 的效果
a+(-b)=a-b
而 ctrl 則是這個 5bit adder 中，MSB 的 full adder 中的 cout，用意是判斷 a 或 b 誰大，如果是 a 大的話就採用 adder 的結果，如果是 b 大的話就採用 a 作為 output（就不相減）



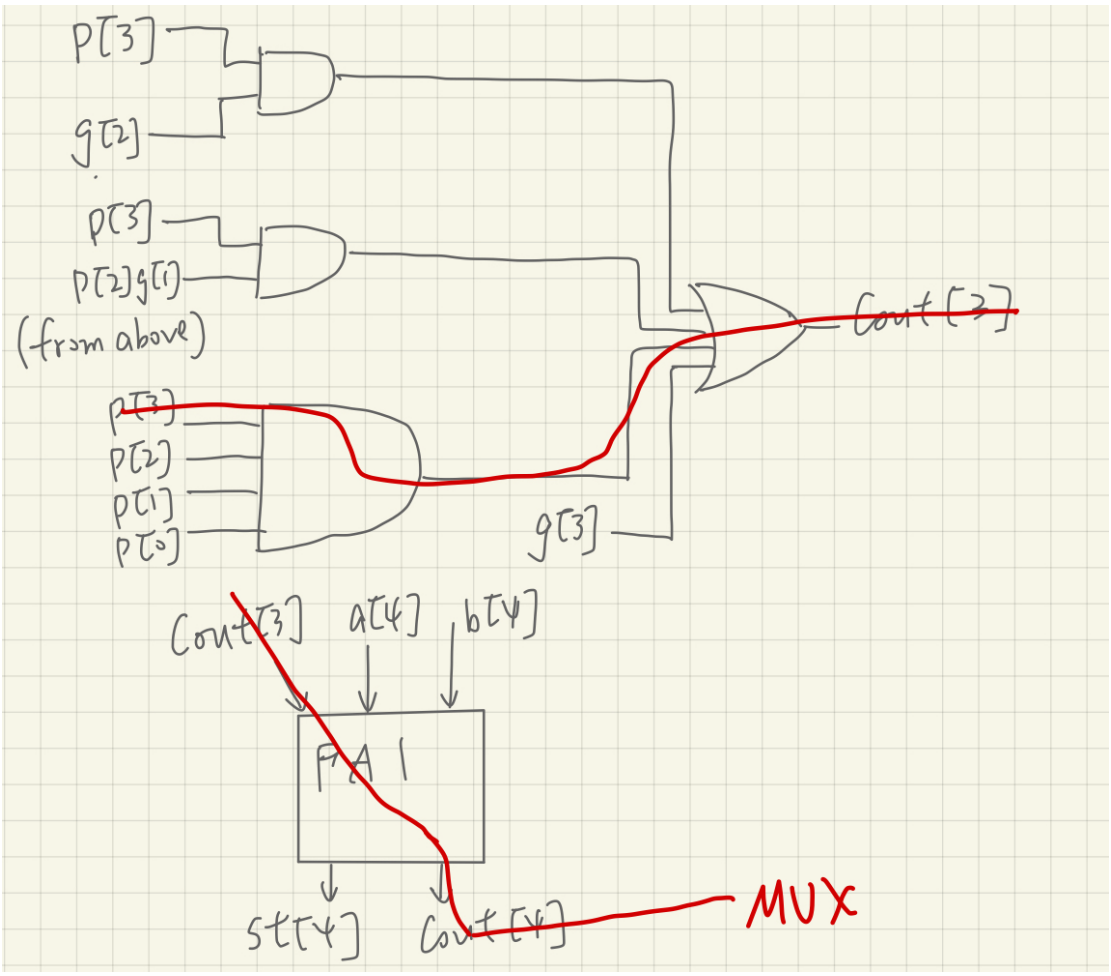
而 adder 使用的是 4bit 的 carry lookahead adder + 1bit lib.v 中提供的 FA1，
carry lookahead adder 的架構如下:
cout[0]=cin[1], cout[1]=cin[2],cout[2]=cin[3],cout[3]=cin[4],cout[4]=ctrl,



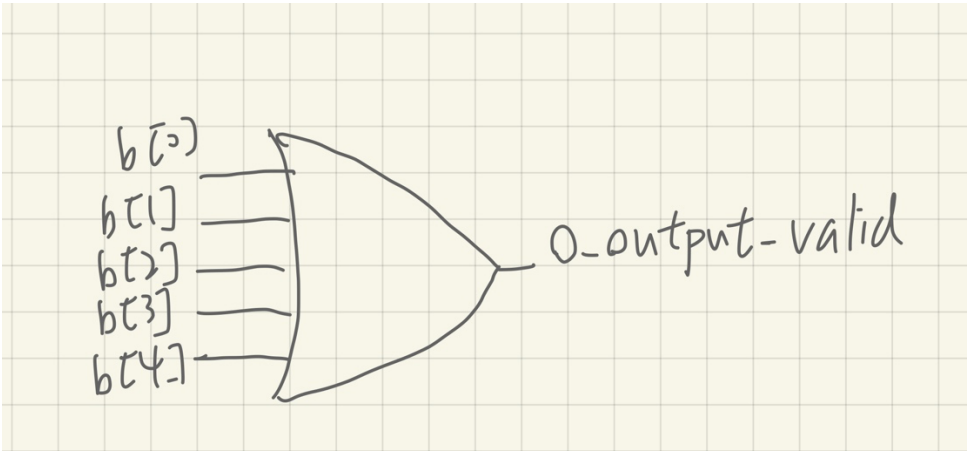




然後把 st[4:0]放入 MUX, 把 cout[4]接到 Ctrl，就可以得到最後的 output
而 dflip flop 的部分則是由數量不定的 FD2 所組成，用來儲存計算結果到下一個 cycle。
而 critical path 是紅色線的部分，從 A[3]和 B[3]做 OR2 得到 p[3]，p[3]經過 AN4, OR4,FA1 最後到 MUX21H



最後是 output valid 的部分，這裡的 b[4:0]指的是已經經過 8 個 stage 的 dff 的 i_b[4:0]



(c)discussion

1.這個電路主要的架構在上題有介紹過了，由 processing unit 和 d flip flop 所組成，而 processing unit 則是由一個 adder 和 MUX 組成，而這個 5bit 的 adder 在 MSB 的 adder 使用的是 lib.v 提供的 FA1，後 4 個 bit 的 adder 則是使用 carry lookahead adder，計算出的 st[4:0]再放入 MUX 中，由 cout[4]作為 ctrl 來決

定 output，

2.pipeline 的切法則是切成 8 個 stage，每層做一次 processing unit 的操作，每層都會對一個 i_a 的 bit 做運算，8 個 stage 後會對所有 i_a 做過運算，而在每一個 processing unit 中，如果 a 比 b 大，output 就會是 a-b 的結果，若 a 比 b 小，output 就會是 a（不使用減法器的 output）。

3.原本使用的是 ripple adder，但 ripple adder 在 ripple 的過程中，會花到相當多的時間，當時使用 5bit ripple adder 最好的 clock cycle 大約是 4.8ns，而 critical path 則是算出 MSB 的 cout，為了降低 critical path，我把 5bit ripple adder 改成 1bit FA1+4bit carry lookahead adder，因為 carry lookahead adder 的特性是，可以不用透過 ripple 的方式，平行計算每一位數的 cout，如此一來就會加快速度，而會使用 1bit FA1+4bit carry lookahead adder 的原因則是因為 carry lookahead adder 會使用相當大量的 transistors，故在 area 還有 speed 的 trade off 之下，我選擇犧牲了一個 bit 的 adder 作為 FA1，減少 transistors 的使用。

4.在 trade off 的部分，我使用的策略是，先想辦法降低 critical path，再來找找看有哪些 input 固定是 0 或 1，再做簡化邏輯的步驟，想辦法把使用多個 transistor 的邏輯閘以使用較少 transistor 的邏輯閘取代

5.在使用 carry lookahead adder 之前，我還曾經用過 carry skip adder，雖然有成功降低 clock cycle，但也相對消耗較多的 transistors，經過比較 performance 之後，還是選擇暫時採用原本的 ripple adder，但最後採用的是 carry lookahead adder 和 FA1 的組合。