實用數位系統設計：

Final Project

Geofence

組別：第十九組 好想吃汕浜

組長：伍思愷 B093011013

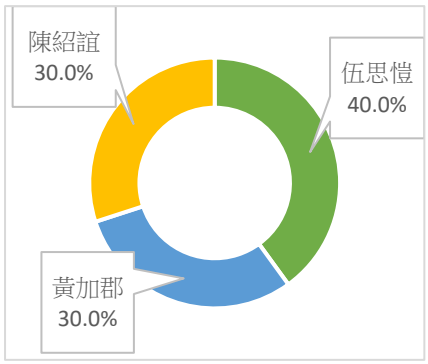
組員：黃加郡 B093011042

組員：陳紹誼 B083012023

設計實作摘要

是否通過 RTL (合成前, pre-synthesis)驗證?	yes
是否通過 gate-level(post-synthesis)驗證?	yes
Clock period (ns)	15
Area (um^2)	30683
Power (mW)	0.6269
Simulation time (ns) (合成後)	16575
是否引用其他組的設計構想或架構?	no
Coding style check and enhancement?	no
Code coverage evaluation and enhancement?	no
其他重要特色	比較不同週期下的最佳結果

工作分配與貢獻



伍思愷 40.0%：

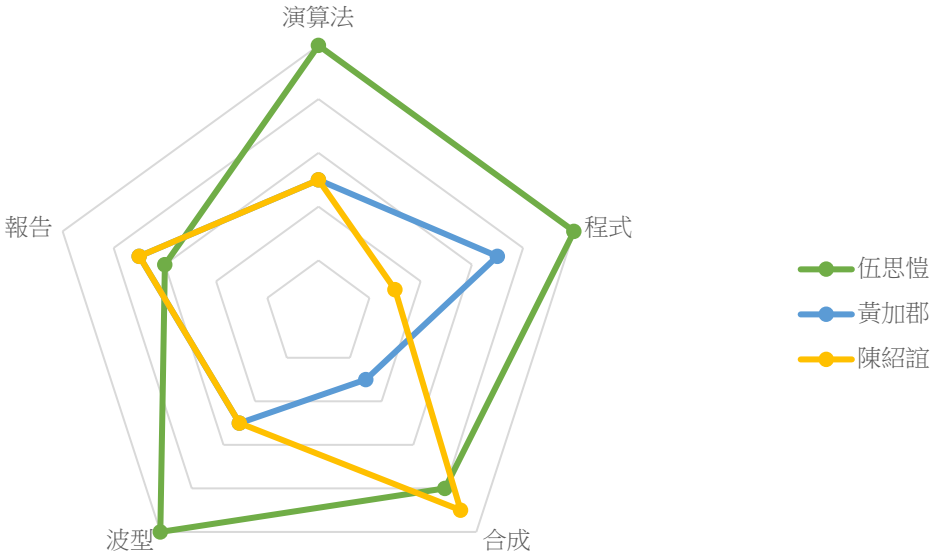
演算法、電路 code、合成電路、優化電路、波型模擬、結報製作

黃加郡 30.0%：

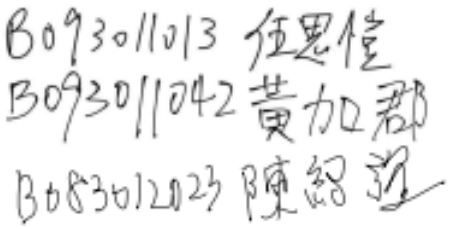
電路 code、簡報製作、結報製作

陳紹誼 30.0%：

合成電路、簡報製作、結報製作

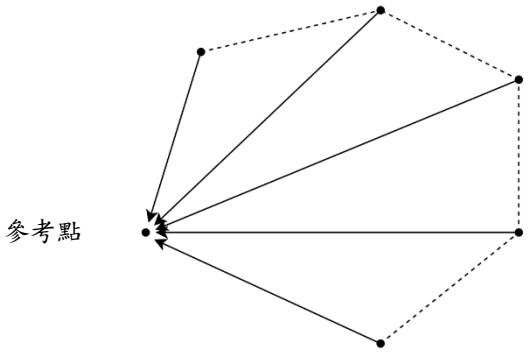


(附上簽名截圖、照片)

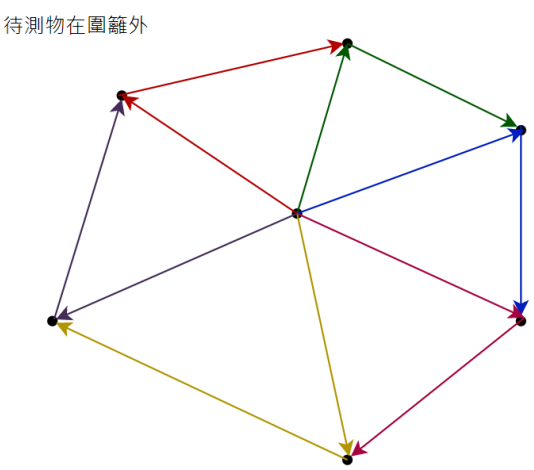


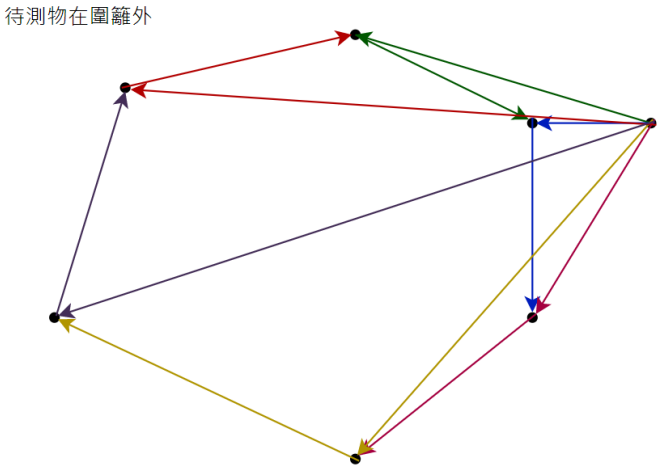
一、演算法介紹與說明

主要是參考助教的做法，首先訊號會一個接著一個輸入暫存器，在訊號輸入暫存器後挑一個接收器(第二組輸入)為參考點，利用外積然後比大小進行排序建立順時鐘的圍籬。



在完成圍籬後接著就是繼續利用外積來判斷待測物是否在圍籬內，外積結果為全正代表待測物在圍籬內，反之則代表待測物在圍籬外。

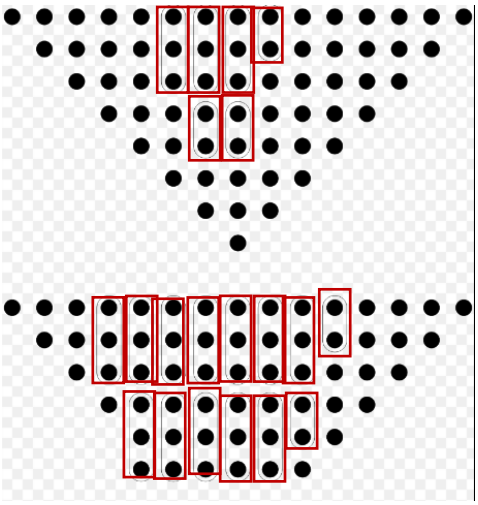




Dadda 乘法器

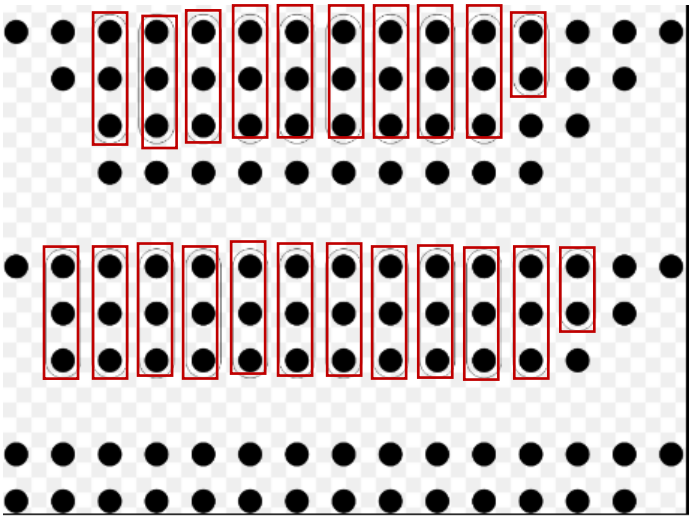
特性：Dadda 乘法器試圖最小化使用的邏輯閘數以及輸入/輸出延遲。

算法：

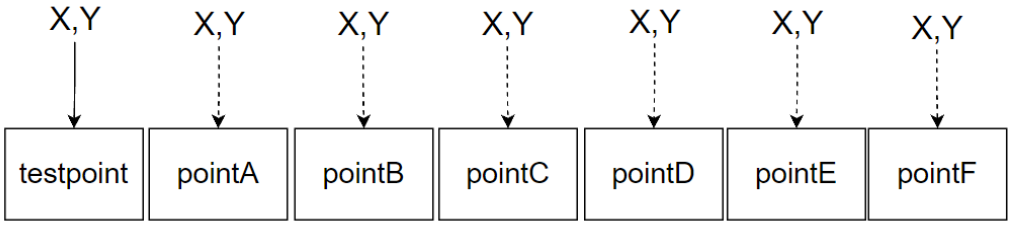


上圖以 4x4 乘法為例，為了減少邏輯閘的使用數，首先定義 d，d 為該層乘法器化簡後，每一行所能擁有最多的元素數量(高度)，下一層的 d 都是上一層 d 的 1.5 倍且無條件捨去小數。依此得知 d=2, 3, 4. 6, 9……，接著檢查每一行所擁有的元素，從最右邊開始檢查，如果元素數量小於或等於，則檢查下一行;如果超出數量，就要用 HA 或 FA 來產生和，減少元素數量至 d，並且要把加法器的進位添加至下一行。

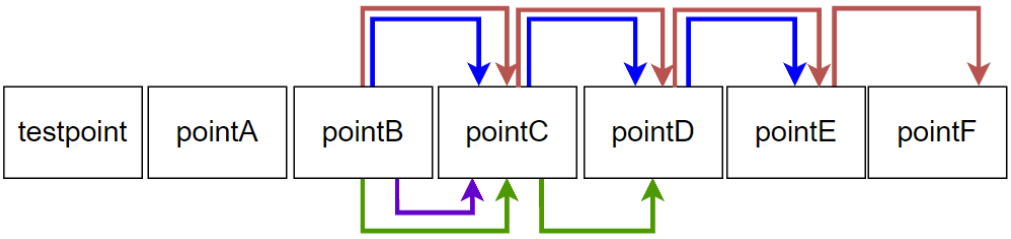
4x4 乘法器的 d 一開始是 6，所以每一行都要把元素數量減少到六個以下，看到上圖紅色方框，每個方框都代表一個加法器，用來減少元素數量。接著 d 的值是 4，重複做法直到每一行元素數量小於等於 4 個。看到下圖，接下來 d 是 3，繼續重複做法。最後是 d=2，化簡完畢後可以發現，其表達式會等於一個多位元的加法器，就完成化簡邏輯閘數量的目的，同時也減少了延遲。



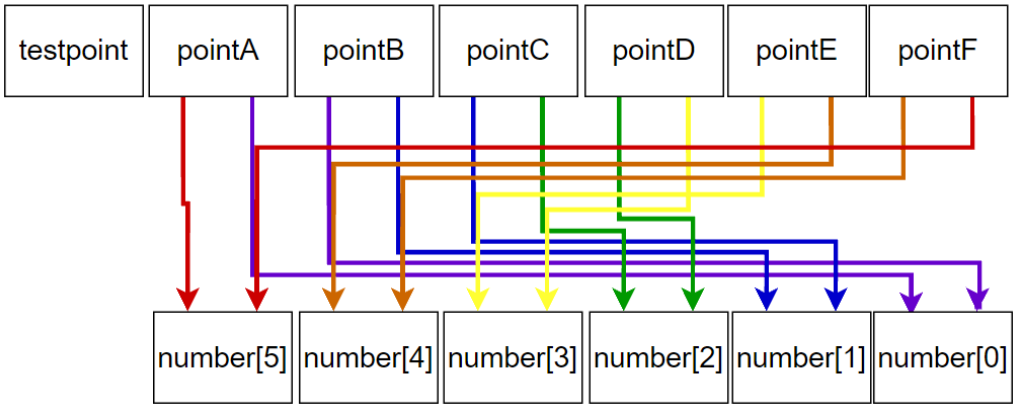
整體演算法流程



先根據順序，在每個週期把 XY 值輸入到對應的點。

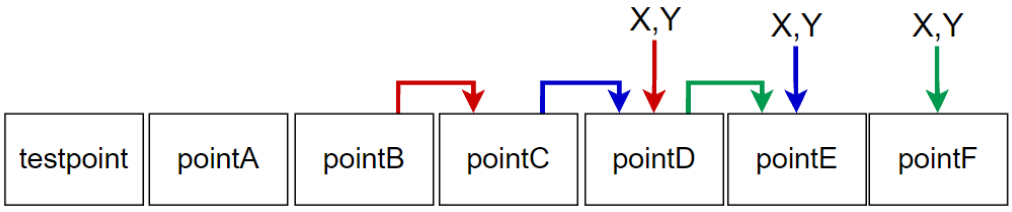


開始排序，兩個點兩個點之間比較、交換，最後得出順時鐘排序的六個點。



排序後，比對測試點和邊之間的關係，並把比對過後的結果存到 number，順時針為 1，反之為 0。Number 如果等於 111111，就代表測試點在六邊形內部。

另一種演算法



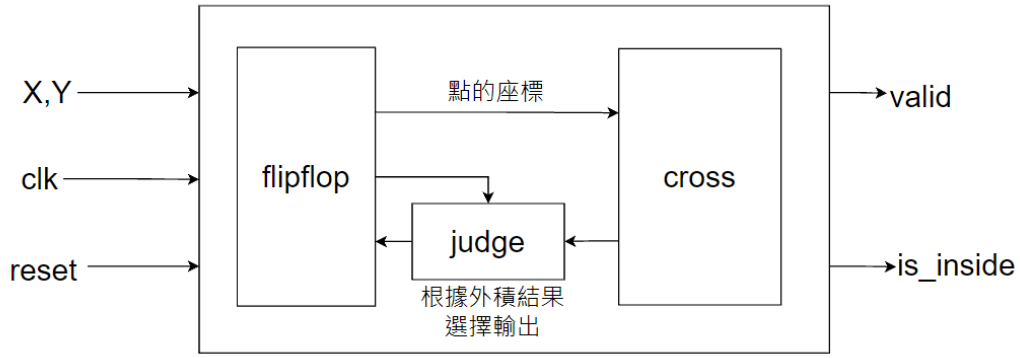
可以輸入資料的同時進行排序，例如紅色，在 D 點輸入的同時，對 BC 兩點進行排序

比較優缺點

先輸入好再進行排序會花費較多的週期，但電路不會那麼複雜，面積較小。
邊輸入邊排序可以減少週期，但電路要處理較多訊號，較複雜，面積較大。

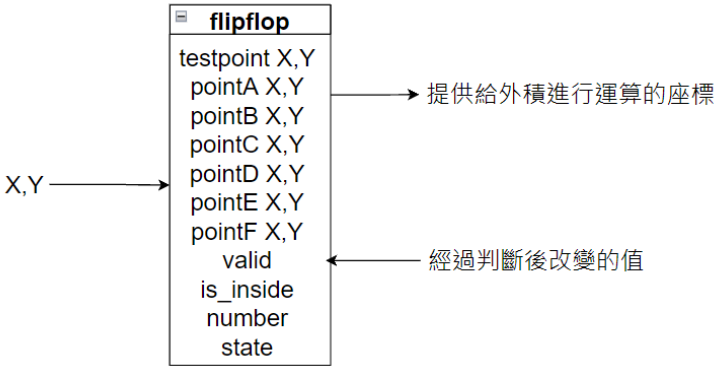
二、架構介紹與說明

Geofence



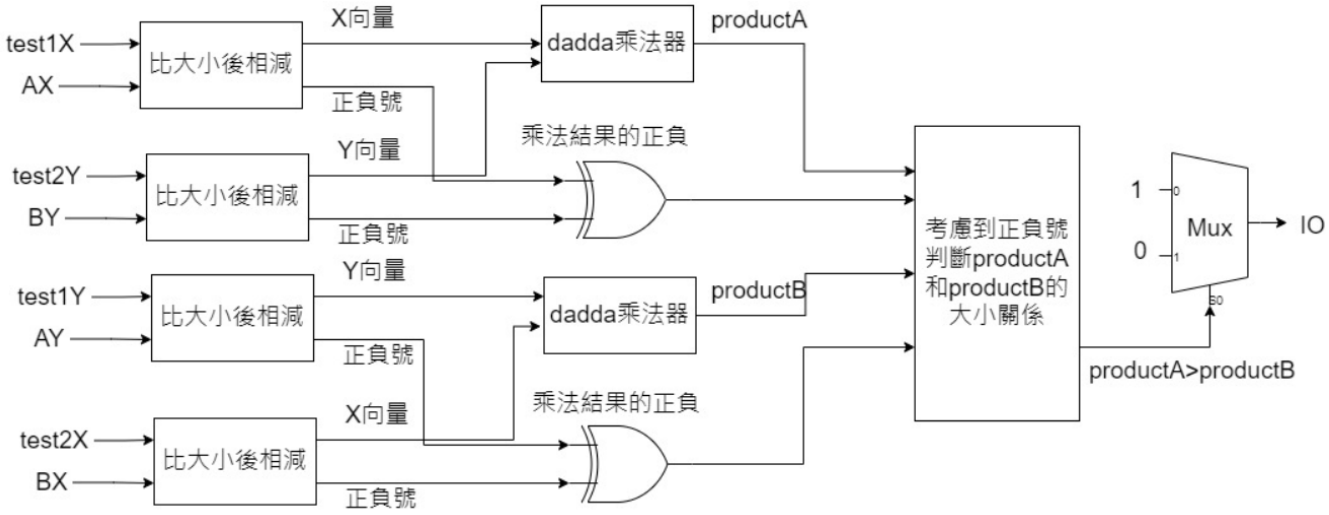
由暫存器、外積和判斷所組成，其中暫存器用來儲存每一點座標的資訊，而外積可以依據狀態機，從暫存器中提取任意四個點的座標來進行運算，並將外積的結果輸出。這時 judge 會根據外積結果，來對暫存器裡面的內容進行更改。

Flipflop



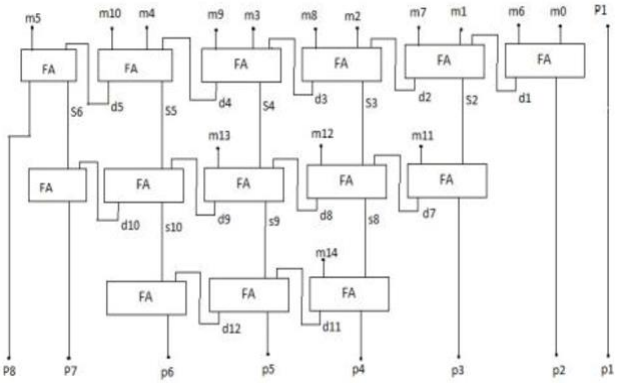
儲存每個點座標及判斷值，會用到 20x7(座標)+5(狀態機)+2(輸出)+6(判斷值)=153 個暫存器

Cross



根據外積的定義，(test1-A)*(test2-B)=(test1X-AX)(test2Y-BY)-(test1Y-AY)(test2X-BX)。先求出各向量後再進行相乘，考慮到正負號的問題，要先對每個向量進行大小判別決定正負號，以 0 為正 1 為負，之後再帶入乘法，乘法為數值相乘的結果加上正負號的相乘，其中正負號相乘的結果可以用一個 XOR 閘來表示。得到兩組乘積後，再根據正負號來判斷兩者的大小關係，最終輸出的 IO 即可用來進行後續排序或判斷內外的依據。

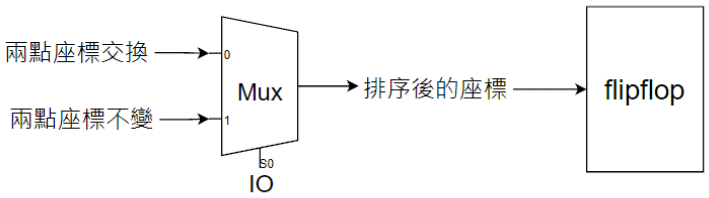
Dadda



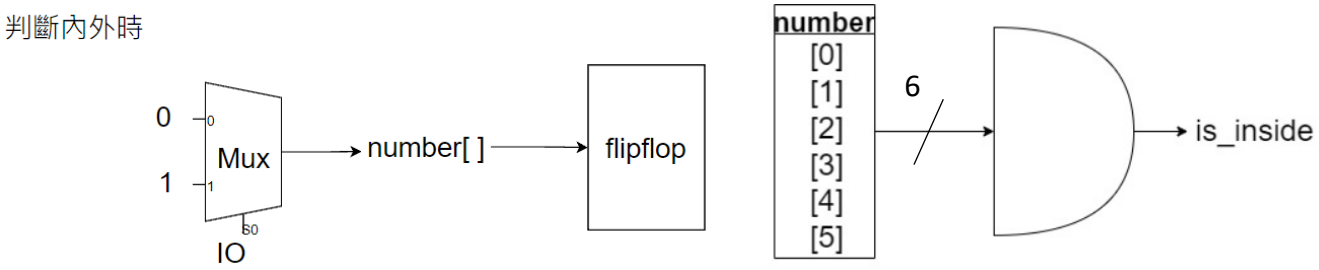
由於 10x10 的架構太龐大，這邊從網路上找 4x4 的範例來做說明，會先對每個要相乘的位元做 OR 閘得到相乘結果，再根據前面演算法介紹的方式，用 FA 和 HA 組合在一起，即可得到相乘後的數字。

Judge

排序時



判斷內外時



會藉由外積所輸出的結果(IO)來選擇要執行的動作，由於我們是做順時針排序，所以當 IO 為 0 時，代表兩個座標位置相反，需要交換;IO 為 1 時，則不需要交換。接著再把交換過後的值存進暫存器。

排序過後要進行內外的判斷，同樣也是用外積輸出的結果，總共要進行六次的判斷，I0 為 1 時代表在內部，此時會把 IO 的值輸入進判斷值 number 對應的位元，做第一次判斷時把結果輸進 number[0]，第二次時結果輸進 number[1]，依此類推。而要判斷一個點在內部，number 的每個位元都必須為 1 才行，所以最後用一個 AND 閘來決定 is_inside 的值。

不同演算法的架構

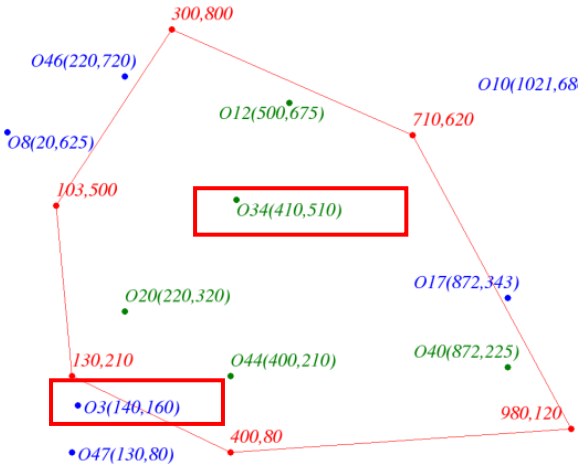
由於我們不同的演算法(先輸入再排序、邊輸入邊排序)改變的只有在不同狀態機做的事，所以電路架構並不會受到影響。

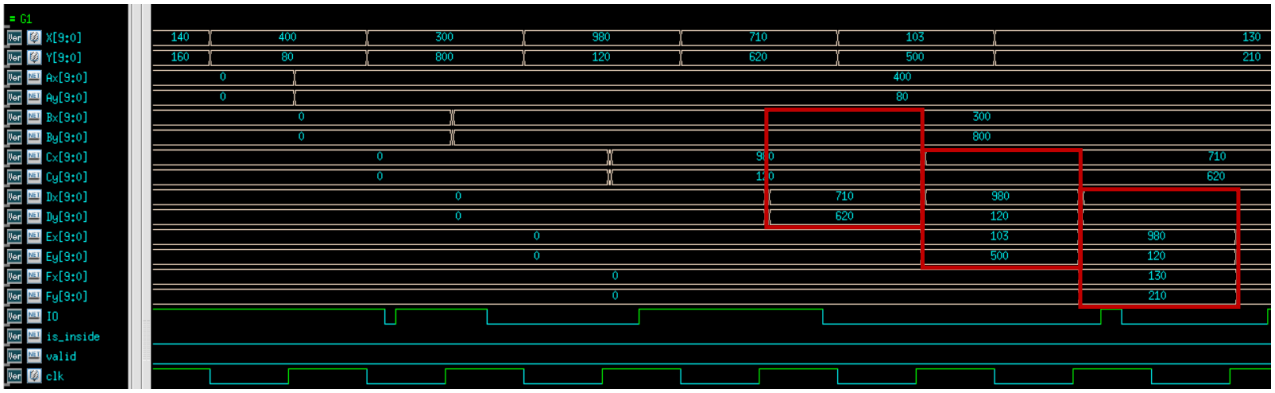
三、驗證流程說明

利用助教給的測試檔以及自己想到的特殊案例，來進行驗證。

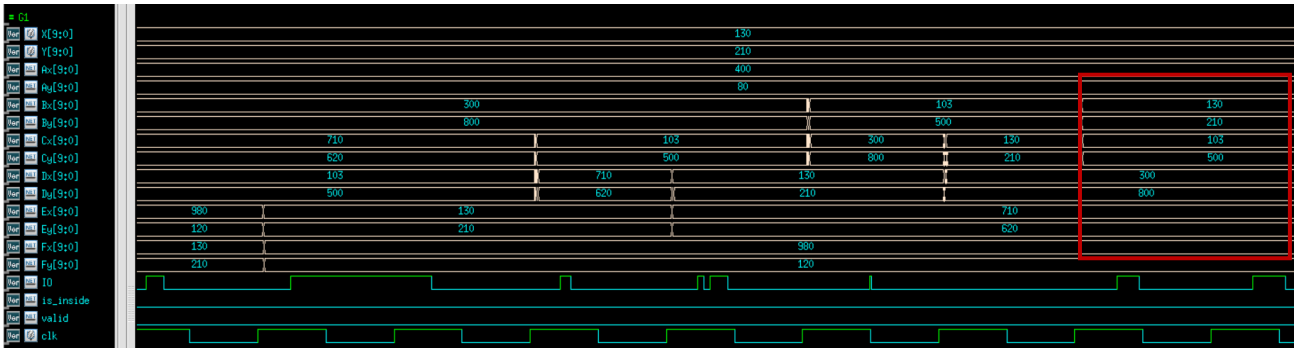
一般六邊形

以助教提供的測試檔為例，討論 O3(外部)和 O34(內部)的情況



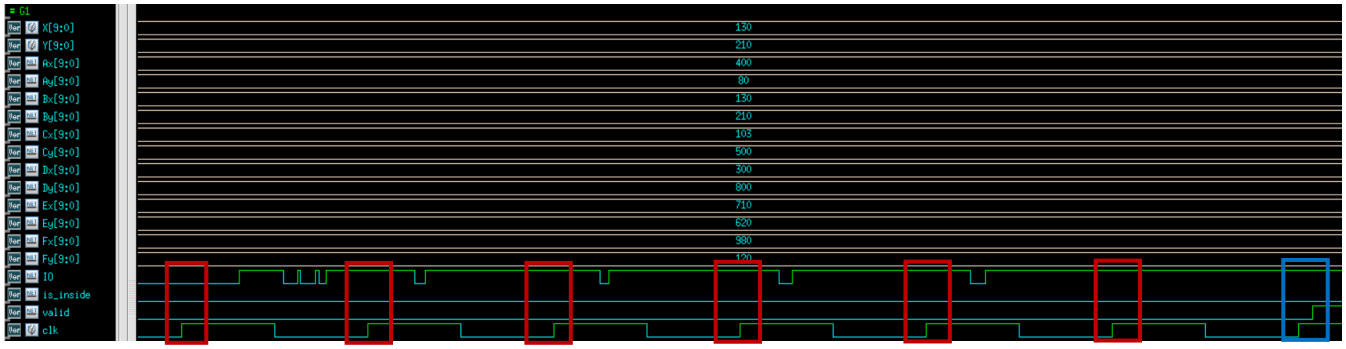


一開始歸零後，座標依序輸入至測試點、A 點、B 點、C 點、D 點、E 點、F 點。可以注意到在排序時，已經開始對前面輸入好的座標開始進行排序交換，如上圖紅色方框所示。



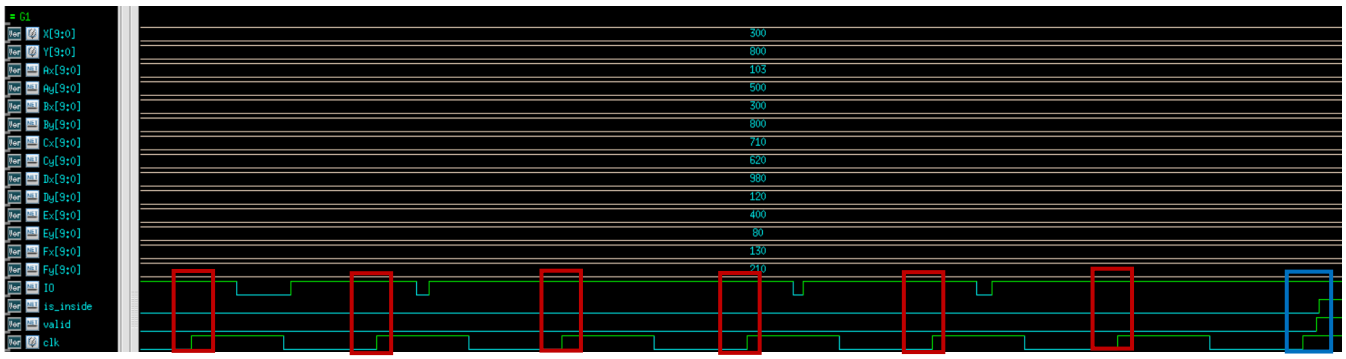
輸入完畢後會繼續進行排序交換，到最後會形成以 A 點開始順時鐘方向的排列。

O3



開始對待測物和每條邊進行判斷可以看到最左邊的紅色方框 IO 值並不是 1，代表待測物在六邊形外面。到最後藍色方框時輸出結果，valid 為 1，is_inside 為 0。

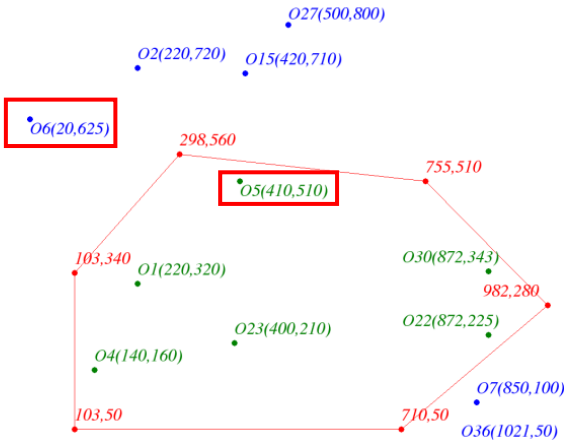
O34

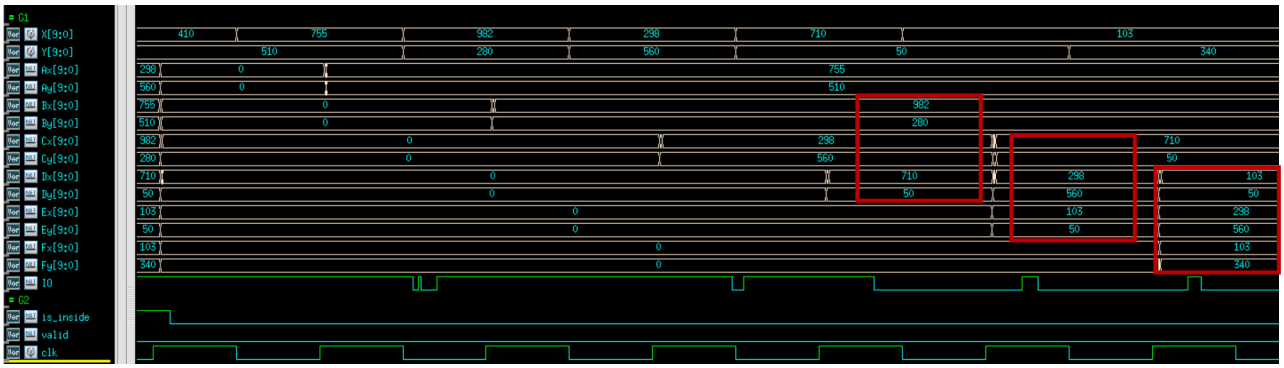


由於兩個待測點所用的六邊形都相同，只是差在點輸入的順序不同，但最後都會排出順時鐘方向的六邊形座標(參考點不同)，這邊就不再重複上面輸入及排序的流程。可以看到每個紅色方框(正緣觸發時)IO 的值都是 1，符合設計的規範。因此得知 O34 這個點在六邊形內部。藍色方框時輸出結果，valid 為 1，is_inside 為 1。

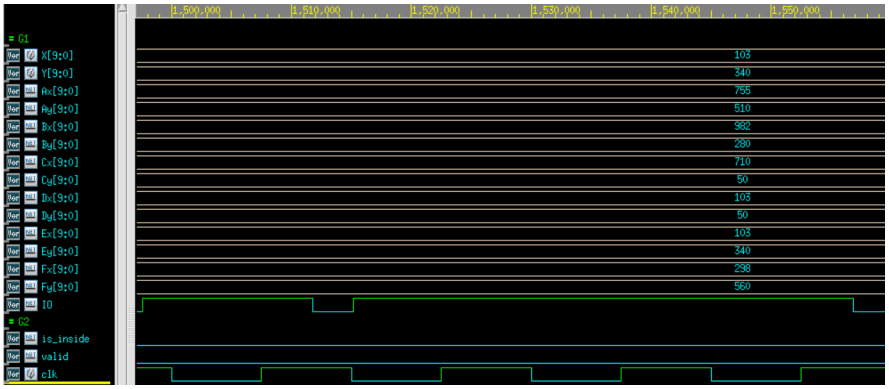
有一個直角的六邊形

以助教提供的測試檔為例，討論 O6(外部)和 O5(內部)的情況



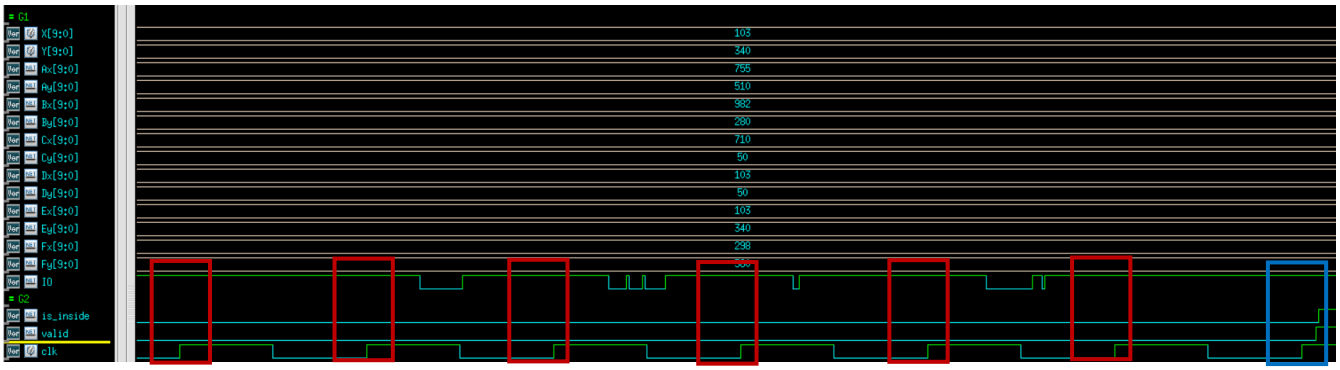


一開始歸零後，座標依序輸入至測試點、A 點、B 點、C 點、D 點、E 點、F 點。可以注意到在排序時，已經開始對前面輸入好的座標開始進行排序交換，如上圖紅色方框所示。



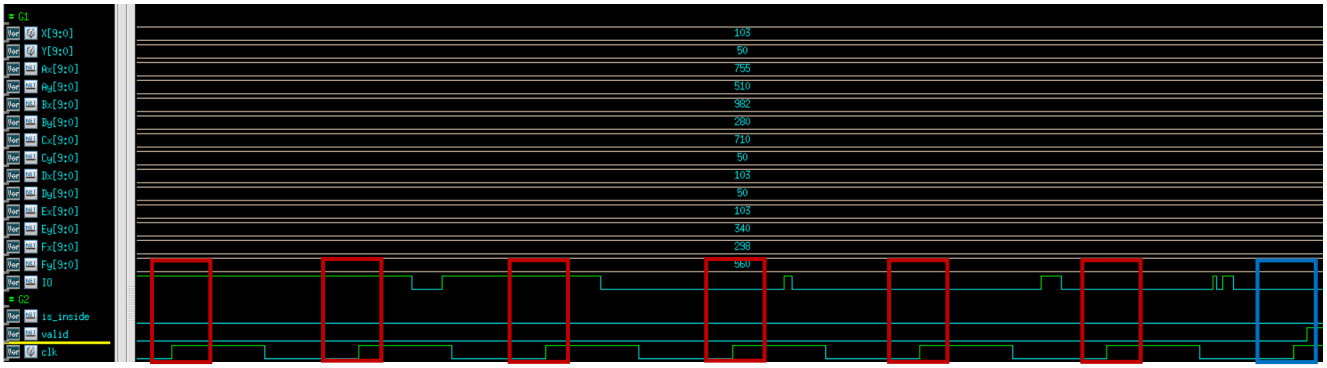
經過前面的邊輸入邊排序，發現已經把座標點排序完畢了，所以後續用來交換的週期，每個座標點的數值都不會改變。

O5



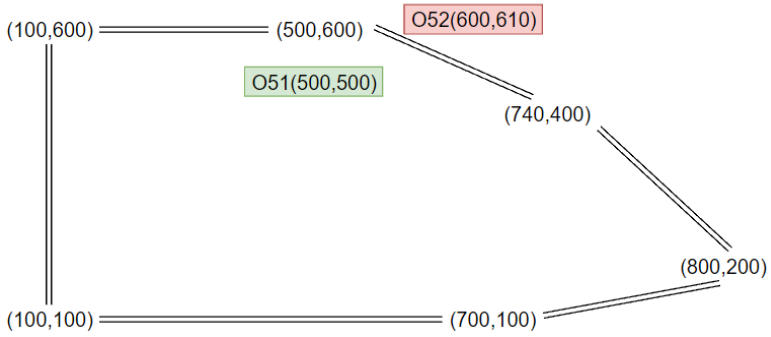
可以看到 IO 會有 glitch 的情況，這是因為在計算外積時運算產生的變化，只要在下次正緣觸發前完成計算即可。可以看到紅色方框正緣觸發時，IO 都為 1，代表 O5 在六邊形的內部。在藍色方框時輸出，valid 為 1，is_inside 為 1。

O6



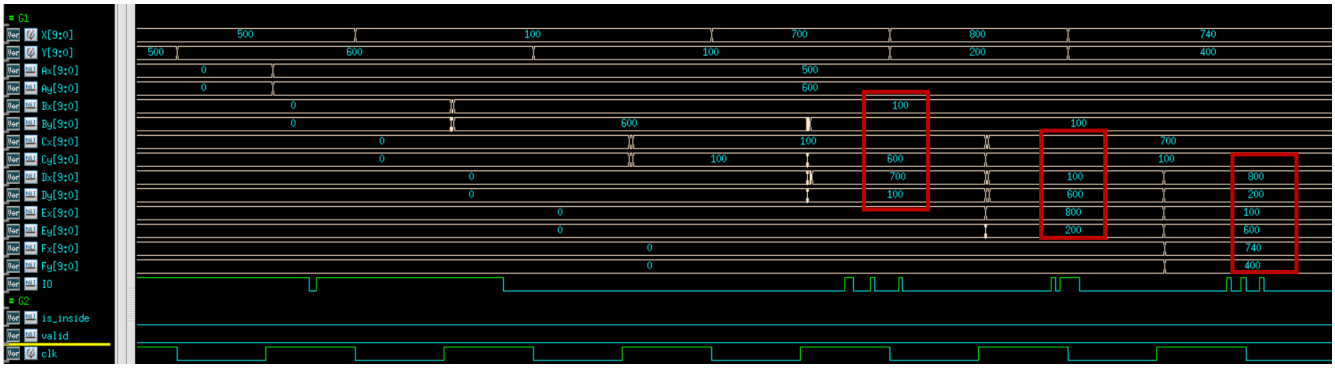
IO 並不是每一次都為 1，也有 0 的情況發生，代表 O6 在六邊形的內部。在藍色方框時輸出，valid 為 1，is_inside 為 0。

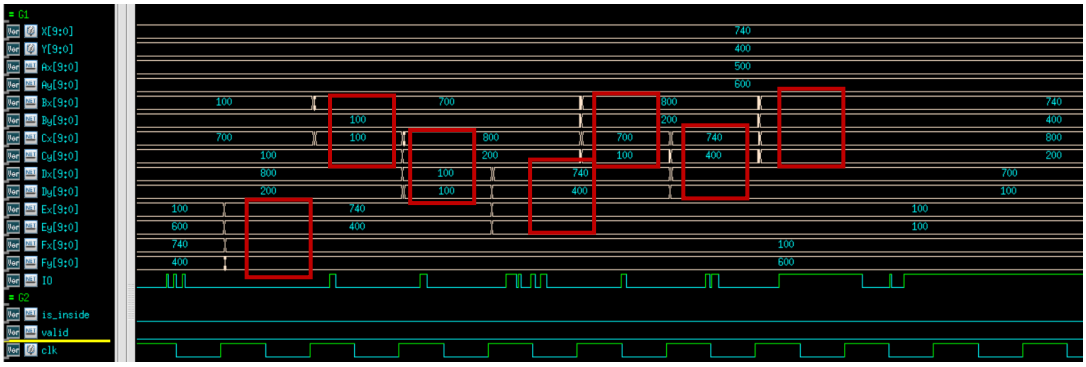
有兩個直角的六邊形



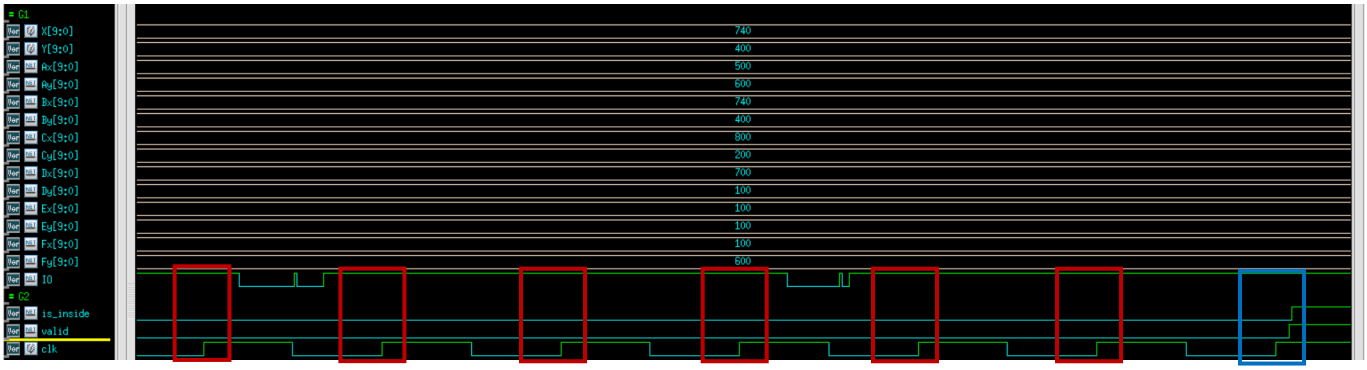

```
Scenario51(in):      X     Y
        Object:    500,  500
           AP1:    500,  600
           AP2:    100,  600
           AP3:    100,  100
           AP4:    700,  100
           AP5:    800,  200
           AP6:    740,  400
Object51: Golde/Return => 1/1, PASS
```

討論有兩個直角的情況，我們修改助教給的測試檔，在最後面加上一項。





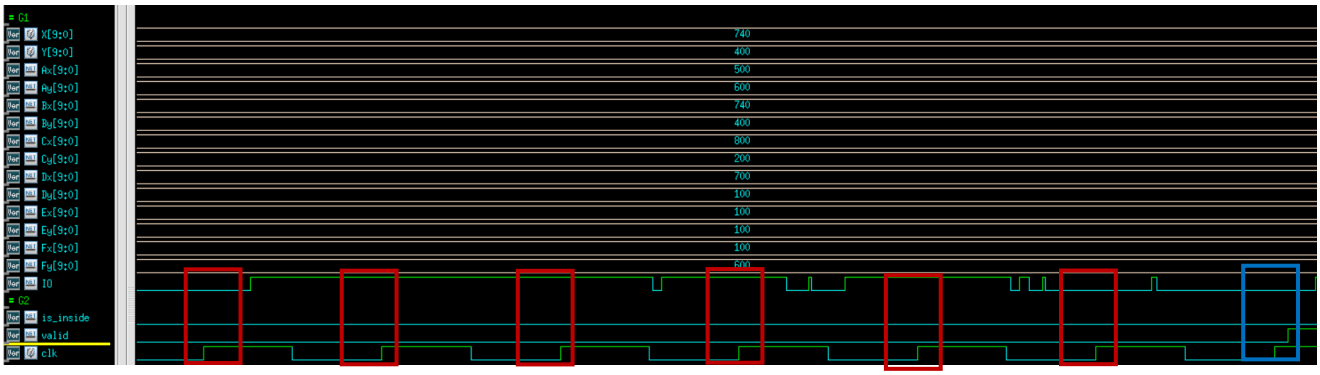
首先會依照順序輸入座標到每個點，可以注意到紅色方框中，由於設定點輸入的方向是逆時針，為了形成順時針方向，每個週期都會交換。



每個紅色方框都是代表在正緣觸發時，測試點與六邊形的邊外積值都為 1，代表 O51 在六邊形內部。在藍色方框時輸出，valid 為 1，is_inside 為 1。

052

```
Scenario52(out):    X     Y
        Object:   600,  610
           AP1:   500,  600
           AP2:   100,  600
           AP3:   100,  100
           AP4:   700,  100
           AP5:   800,  200
           AP6:   740,  400
Object52: Golde/Return => 0/0, PASS
```



由於是用同一組六邊形，輸入和排序的過程就和前面一樣。這邊只討論判斷的過程。測試點與六邊形的邊外積值有 1 有 0，代表 O52 在六邊形外部。在藍色方框時輸出，valid 為 1，is_inside 為 0。

四、邏輯合成流程說明與合成參數(以最佳結果為例)

在 Xshell 輸入 dv &即可打開 Design Compiler，修改 sdc 檔中的週期，改為自己訂的週期 15ns，接著再 Design Compiler 指令欄中輸入 source dc_syn.tcl 即可完成電路合成(如下圖)，出現 Optimization Complete 代表合成完成，須注意是否有 latch，若有則代表 code 要修改為 flipflop。

Register Name	Type	Width	Bus	MB	AR	AS	SR	SS	ST
is_inside_reg	Flip-flop	1	N	N	Y	N	N	N	N
valid_reg	Flip-flop	1	N	N	Y	N	N	N	N
state_reg	Flip-flop	5	Y	N	Y	N	N	N	N
Tx_reg	Flip-flop	10	Y	N	Y	N	N	N	N
Ty_reg	Flip-flop	10	Y	N	Y	N	N	N	N
Ax_reg	Flip-flop	10	Y	N	Y	N	N	N	N
Ay_reg	Flip-flop	10	Y	N	Y	N	N	N	N
Bx_reg	Flip-flop	10	Y	N	Y	N	N	N	N
By_reg	Flip-flop	10	Y	N	Y	N	N	N	N
Cx_reg	Flip-flop	10	Y	N	Y	N	N	N	N
Cy_reg	Flip-flop	10	Y	N	Y	N	N	N	N
Dx_reg	Flip-flop	10	Y	N	Y	N	N	N	N
Dy_reg	Flip-flop	10	Y	N	Y	N	N	N	N
Ex_reg	Flip-flop	10	Y	N	Y	N	N	N	N
Ey_reg	Flip-flop	10	Y	N	Y	N	N	N	N
Fx_reg	Flip-flop	10	Y	N	Y	N	N	N	N
Fy_reg	Flip-flop	10	Y	N	Y	N	N	N	N
number_reg	Flip-flop	6	Y	N	Y	N	N	N	N

```
ce.sdc   geofence.v   tb.sv
# operating conditions and boundary conditions #

set cycle 15.0
create_clock -name clk  -period $cycle   [get_ports  clk]

set_dont_touch_network      [all_clocks]
set_fix_hold                [all_clocks]
set_clock_uncertainty  0.1  [all_clocks]
set_clock_latency      0.5  [all_clocks]
set_ideal_network           [get_ports clk]
```

```
    0:00:16   31872.1      0.15       4.2       0.0                          0.00
    0:00:16   31844.9      0.15       4.2       0.0                          0.00
    0:00:16   31844.9      0.15       4.2       0.0                          0.00
    0:00:16   31882.3      0.00       0.0       0.0                          0.00
    0:00:16   27907.0      2.89     285.3       0.0                          0.00
    0:00:16   27655.7      2.84     280.5       0.0                          0.00
    0:00:16   27570.9      2.79     275.2       0.0                          0.00
    0:00:16   27570.9      2.79     275.2       0.0                          0.00
    0:00:16   27570.9      2.79     275.2       0.0                          0.00
    0:00:16   27570.9      2.79     275.2       0.0                          0.00
    0:00:16   27570.9      2.79     275.2       0.0                          0.00
    0:00:16   27570.9      2.79     275.2       0.0                          0.00
    0:00:17   28360.2      1.55     147.4       0.0 Cy_reg[1]/D              0.00
    0:00:17   29120.6      0.95      83.3       0.0 Cy_reg[1]/D              0.00
    0:00:18   29706.2      0.62      49.7       0.0 Cy_reg[1]/D              0.00
    0:00:19   30356.3      0.24      11.1       0.0 Cy_reg[1]/D              0.00
    0:00:19   30609.2      0.04       0.1       0.0 Cy_reg[1]/D              0.00
    0:00:19   30683.9      0.00       0.0       0.0                          0.00
Loading db file '/mnt3/CBDK_IC_Contest_v2.1/SynopsysDC/db/slow.db'


Note: Symbol # after min delay cost means estimated hold TNS across all active scenarios

  Optimization Complete
Writing ddc file 'geofence_syn.ddc'.
Information: Annotated 'cell' delays are assumed to include load delay(UID-282)
Information: Writing timing information to file '/home/PDSD110/PDSD110a31/RTLtest/geofence
Information: Updating design information...(UID-85)
Writing verilog file '/home/PDSD110/PDSD110a31/RTLtest/geofence_syn.v'.
Current design is 'geofence'.
design_vision>
Log   History
design_vision> source dc_syn.tcl
```

(1) Power

Dynamic power =596.6024uW

Static power = 26.3260uW

```
****************************************
Report : power
        -analysis_effort low
Design : geofence
Version: P-2019.03
Date   : Mon Jun 13 04:15:52 2022
****************************************


Library(s) Used:

    slow (File: /mnt3/CBDK_IC_Contest_v2.1/SynopsysDC/db/slow.db)


Operating Conditions: slow   Library: slow
Wire Load Model Mode: top

Design        Wire Load Model            Library
------------------------------------------------
geofence              tsmc13_wl10       slow


Global Operating Voltage = 1.08
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000pf
    Time Units = 1ns
    Dynamic Power Units = 1mW    (derived from V,C,T units)
    Leakage Power Units = 1pW


  Cell Internal Power  = 349.2153 uW   (58%)
  Net Switching Power  = 249.3871 uW   (42%)
                         ---------
Total Dynamic Power    = 598.6024 uW  (100%)

Cell Leakage Power     =  28.3260 uW


                 Internal         Switching           Leakage            Total
Power Group      Power            Power               Power              Power   (   %    )  Attrs
--------------------------------------------------------------------------------------------------
io_pad             0.0000           0.0000            0.0000            0.0000  (   0.00%)
memory             0.0000           0.0000            0.0000            0.0000  (   0.00%)
black_box          0.0000           0.0000            0.0000            0.0000  (   0.00%)
clock_network      0.0000           0.0000            0.0000            0.0000  (   0.00%)
register           0.2400        2.2484e-02        4.6659e+06            0.2672  (  42.62%)
sequential         0.0000           0.0000            0.0000            0.0000  (   0.00%)
combinational      0.1092           0.2269        2.3660e+07            0.3597  (  57.38%)
--------------------------------------------------------------------------------------------------
Total              0.3492 mW        0.2494 mW     2.8326e+07 pW         0.6269 mW

***** End Of Report *****
```

(2) Area

電路面積為 30683.899462um^2， 約使用了 6136 個邏輯閘。

```
****************************************
Report : area
Design : geofence
Version: P-2019.03
Date   : Mon Jun 13 04:20:29 2022
****************************************

Library(s) Used:

    slow (File: /mnt3/CBDK_IC_Contest_v2.1/SynopsysDC/db/slow.db)

Number of ports:                         1368
Number of nets:                          3778
Number of cells:                         2442
Number of combinational cells:           2088
Number of sequential cells:               153
Number of macros/black boxes:               0
Number of buf/inv:                        436
Number of references:                     108

Combinational area:              25432.143969
Buf/Inv area:                     2973.844780
Noncombinational area:            5251.755493
Macro/Black Box area:                0.000000
Net Interconnect area:          282941.417450

Total cell area:                 30683.899462
Total area:                     313625.316912

***** End Of Report *****
```

(3) Timing

最大延遲為 15.22ns;slack 為 0。

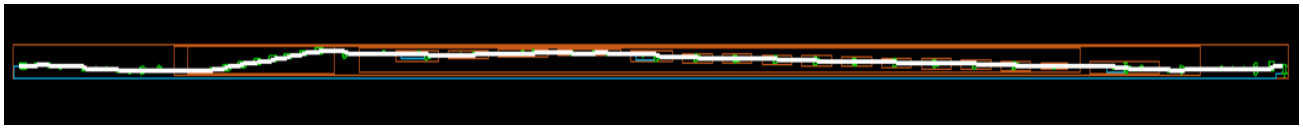
```
  EF/D1/fafinal15/Cout (FA_2)                                0.00      12.09 f
  EF/D1/fafinal16/Cin (FA_1)                                 0.00      12.09 f
  EF/D1/fafinal16/U2/Y (AO22X4)                              0.29      12.37 f
  EF/D1/fafinal16/Cout (FA_1)                                0.00      12.37 f
  EF/D1/fafinal17/Cin (FA_0)                                 0.00      12.37 f
  EF/D1/fafinal17/U1/Y (XOR2X4)                              0.16      12.54 r
  EF/D1/fafinal17/Y (FA_0)                                   0.00      12.54 r
  EF/D1/Y[18] (dadda_0)                                      0.00      12.54 r
  EF/r402/A[18] (Cross_DW01_cmp6_0)                          0.00      12.54 r
  EF/r402/U5/Y (INVX8)                                       0.10      12.64 f
  EF/r402/U72/Y (XOR2X4)                                     0.15      12.79 f
  EF/r402/U16/Y (AOI32X2)                                    0.31      13.10 r
  EF/r402/U14/Y (OAI21X4)                                    0.13      13.23 f
  EF/r402/LT (Cross_DW01_cmp6_0)                             0.00      13.23 f
  EF/U157/Y (NAND2X2)                                        0.10      13.33 r
  EF/U169/Y (OAI2BB2X4)                                      0.22      13.55 r
  EF/IO (Cross)                                              0.00      13.55 r
  U818/Y (BUFX20)                                            0.17      13.72 r
  U713/Y (OR2X4)                                             0.15      13.87 r
  U549/Y (INVX4)                                             0.08      13.95 f
  U548/Y (BUFX16)                                            0.23      14.18 f
  U1064/Y (AOI222XL)                                         0.66      14.84 r
  U1063/Y (OAI221XL)                                         0.38      15.22 f
  Cy_reg[3]/D (DFFRX2)                                       0.00      15.22 f
  data arrival time                                                    15.22

  clock clk (rise edge)                                     15.00      15.00
  clock network delay (ideal)                                0.50      15.50
  clock uncertainty                                         -0.10      15.40
  Cy_reg[3]/CK (DFFRX2)                                      0.00      15.40 r
  library setup time                                        -0.18      15.22
  data required time                                                   15.22
  ---------------------------------------------------------------------------
  data required time                                                   15.22
  data arrival time                                                   -15.22
  ---------------------------------------------------------------------------
  slack (MET)                                                          0.00


***** End Of Report *****
```

(4) Critical Path

最長路徑如下圖所示，共經過 60 層邏輯閘，是從 state 到 C 點的路徑



```
Timing Path Group 'clk'
-----------------------------------
Levels of Logic:               60.00
Critical Path Length:          14.72
Critical Path Slack:            0.00
Critical Path Clk Period:      15.00
Total Negative Slack:           0.00
No. of Violating Paths:         0.00
Worst Hold Violation:           0.00
Total Hold Violation:           0.00
No. of Hold Violations:         0.00
-----------------------------------
```

五、邏輯合成與驗證結果(與比較)

(1) 電路面積

合成後電路面積為 30683.899462um^2， 約使用了 6136 個邏輯閘。

```
Area
-----------------------------------
Combinational Area:    25432.143969
Noncombinational Area:  5251.755493
Buf/Inv Area:           2973.844780
Total Buffer Area:          1206.85
Total Inverter Area:        1766.99
Macro/Black Box Area:      0.000000
Net Area:             282941.417450
-----------------------------------
Cell Area:             30683.899462
Design Area:          313625.316912
```

(2) 電路運算時間

在 cycle=15ns 情況下，所得的運算時間為 16575ns。

```
---------------------------------------------
--    Simulation finish,  ALL PASS           --
---------------------------------------------
Simulation complete via $finish(1) at time 16575 NS + 0
./tb.sv:144             $finish;
ncsim> exit
```

(3) 電路面積*總運算時間

電路面積：30683.899462

總運算時間：16575

電路面積*總運算時間：30683.899462×16575 = **508,585,633.6**

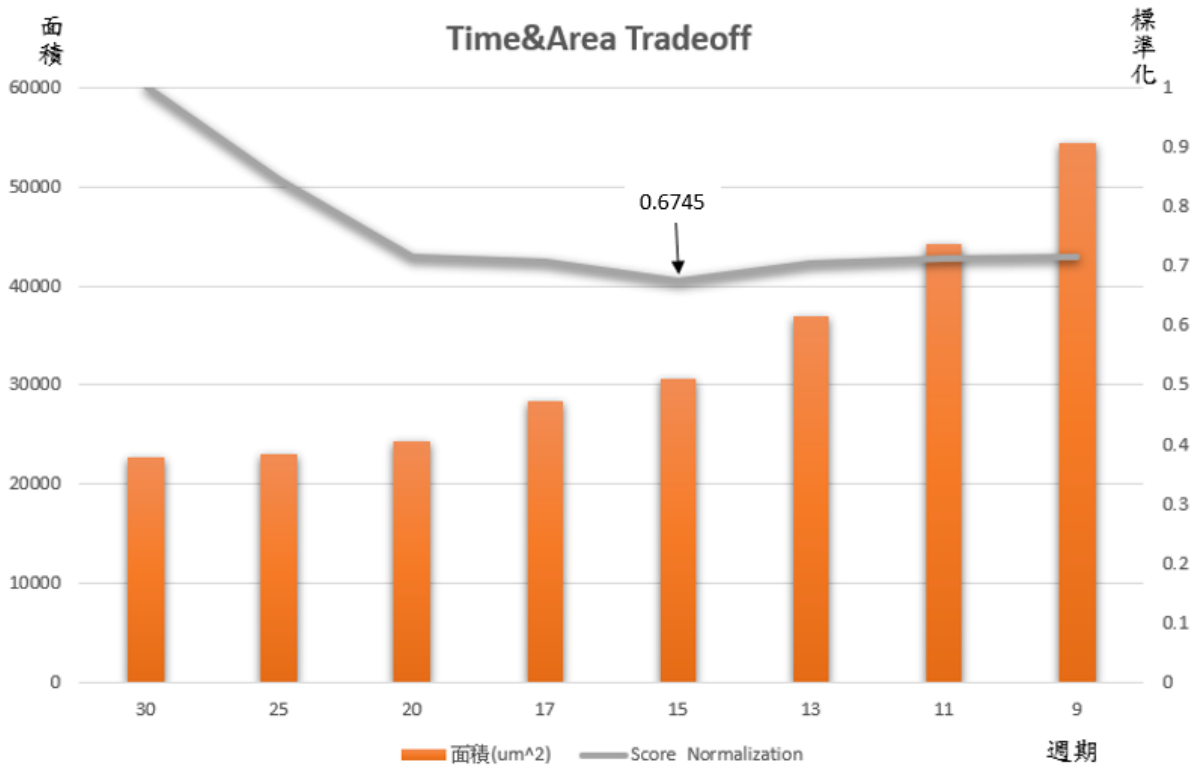
優化前後的合成結果比較

優化內容:縮短週期時間、減少暫存器數量、減少狀態機數量

從原本的輸入完之後排序，改為邊輸入邊排序，減少了狀態機數量。並且在過程中共用了暫存器，把暫存器數量減少至原本的一半以下。在優化後也嘗試了不同週期下的結果，最好的結果(乘積)只有原本的一半!

	優化前	優化後(最佳)
時間週期	30ns	15ns
暫存器數量	366 個	153 個
Area	30865.521443 um^2	30683.899462um^2
邏輯閘數量	6173	6136
Power(Dynamic/Static)	455.1575uW/27.8273uW	596.6024uW/26.3260uW
邏輯閘層數	55	60
運算時間	37650ns	16575ns
電路面積*總運算時間	1,162,086,882	508,585,633.6

優化後不同週期的比較



上圖是針對優化後的電路，透過不同週期來模擬出面積，長條圖即為面積大小，電路最小可以模擬到 9ns，但可以發現隨著時間變小，面積成長的幅度越來越大。於是把所需要的週期乘上面積並且正規化，可以得到折線圖，發現在 15ns 時有最小值，代表這是最好的結果。

六、心得與討論

(1) 伍思愷

經過了這學期的課程，讓我對硬體描述語言的了解，從寫出來功能正確，變成了考慮不同寫法的優缺點。原本只要考慮波型是否正確，到了這堂課要考慮的卻是以合成出來的面積、速度為主，也學到了很多 coding 時要注意的事項，像是正負緣不能混用、clk 訊號要獨立使用等等。

做期末專題的時候，一開始我把小的模組寫好就直接拿去用了，結果發現要除錯時完全找不出來問題出在哪裡，於是又重新幫每個小模組寫測試檔，還真的被我發現了不少漏洞，有了這次教訓，以後寫好模組就應該直接去測試它的功能正確性，才能有效率的完成電路。

另外在合成過程中，電路會合成出 Latch，但我始終找不到問題所在，並不是 if-else 或是沒有 default 的問題，後來我的組員發現是沒有在每一個狀態都規定所有變數的值，才會產生 Latch。解決問題後我有上網找了一下相關問題的資料，發現只要變數沒有在狀態規定值，就會被系統認定為我想要讓這個變數產生記憶性，因此就會合成出 Latch，就算是自己指派給自己，也會產生相同的問題。

最後就是合成出來的結果，很慶幸的是我自己寫的乘法器真的有比系統預設的乘法器面積來的小。有了最初的結果之後，再去把週期的時間減少、優化電路，也想盡辦法來減少周期的數還真的被我想到從原本的輸入完後排序，改成邊輸入邊排序。減少了運算所需的週期數量。到最後優化出來的結果竟然比原本小了一倍，讓我體會到優化的重要性。

(2) 黃加郡

這次期末專題以分組的形式進行，其中有許多地方是我之前獨自打作業時沒有碰到的，像是我認為最明顯的就是打 code 的模式，看到其他人的 code 時會需要適應一下，不像看自己的時候那樣輕鬆，但這次和組員合作的過程中，我並沒有因為這個問題而卡很久，組員們打 code 的習慣很好，因此都蠻容易理解的，而在團體分組作業中最怕遇到的就是團隊的氣氛不良，在這段期間，雖然有遇到一些小困難，但大家並沒有因此撕破臉，而是持續保持優良的團隊氣氛，最終一起完成了這次的期末專題，另外我認為我們這組的組長非常優秀，在我們遇到困難時，他都會有耐心地指導我們，並且在平常的討論中指出一些潛在的小問題，讓我們有明確的改進方向，我認為他是我們之中功勞最大的，也覺得很幸運能找到如此厲害的組長。

而在這次的期末專題中，我負責的是有限狀態機 FSM 的部分，我自己認為這個部分在打 code 方面並不複雜，主要是需要確認我們的設計構想以及組員們的一些輸出名稱，在我們大家都完成自己負責的部分後，進行了合成以及跑模擬，雖然一開始跑出來的數值是在規範之內，但面積過大以及週期壓不下來還是迫切需要優化的部分，其中面積過大主要原因是我們用了太多暫存器，在將許多暫存器共用後面積獲得了很大的改善，週期也順利地往下壓，達到我認為蠻理想的數值。

最後還是要再感謝組長在最後優化的部分做出了巨大的貢獻，也感謝大家彼此的配合讓期末專題順利完成。

(3) 陳紹誼

這次期末專題是我第一次嘗試完成這麼大的一個電路，因為和之前幾次作業的難度相差蠻多的，因此一開始有點害怕沒辦法做出來，幸好後來有助教的方法提供了我們一個大致的方向可以參考。

一開始因為覺得助教的方法要乘太多次，因此便想看有沒有其他更好的方法，後來我們想到了另一個方法，也就是射線法，但後來在設計時才發現因為這個架構會使用到除法，因此會使電路面積大幅上升，且還要考慮精度的問題，因此我們後來在參考同學的方法後還是使用了助教的方法，雖然要乘很多次但是相對穩定而且也不用考慮太多 corner case。

後來在寫 code 時我因為自己 coding 能力不足，導致我負責的部分一直沒辦法完成，後來幸好在組員的幫忙下才有辦法完成這個電路，因此我認為我的 coding 能力還需要很大的加強，後來在電路合成及優化時原本一直跑不過 30ns 還有出現 hold time violation 的問題，後來照老師的方法優化還有在指令加上 maxdelay 這些問題才可以解決，我才發現原來電路優化對電路影響這麼大。

另外，我們還發現我們用自己設計的乘法器還有合成軟體的乘法器所合成出的電路面積和運算時間是差不多的，只比合成軟體好一點而已，我認為可能是因為電路合成軟體可以使用較先進的邏輯閘去合成出電路，例如 AOI 等，而我們只能使用基本的邏輯閘才會造成這個結果，下次在設計電路時，應該讓合成軟體先合成出一個電路，若結果不符合預期，再試著自己設計才對，最後，謝謝教授讓我們有這個機會去嘗試設計這個較大的電路，讓我對設計電路的流程更加了解。

七、參考資料

Dadda 乘法器架構 https://en.wikipedia.org/wiki/Dadda_multiplier

解決合成出 latch 的問題 https://blog.csdn.net/tianyake_1/article/details/79031582

https://www.ptt.cc/bbs/Electronics/M.1344342810.A.991.html