

CAD/VLSI Circuit Design

期末報告

在 FPGA 板上使用 CORDIC 演算法
搭配脈衝陣列的定點數 QR 分解

Fixed point QR decomposition using CORDIC
Algorithms on FPGA with systolic array

學號：7111064109

學生：林軒宇

指導教授：范志鵬

日期：2022/11/28

目錄

一、	簡介.....	- 1 -
二、	理論.....	- 1 -
	given rotation 說明	- 1 -
	使用 CORDIC 達成 given rotation	- 2 -
三、	架構改良.....	- 2 -
	改良一：PE Element 改良	- 2 -
	改良二：input size 改良	- 2 -
	改良三：systolic array 改良.....	- 4 -
	改良四：重新定義 K	- 4 -
四、	設計規格.....	- 5 -
	系統方塊圖.....	- 5 -
	QR_CORDIC 輸入輸出介面	- 5 -
	GG 輸入輸出介面	- 5 -
	GR 輸入輸出介面.....	- 6 -
五、	模擬結果.....	- 6 -
	Matlab 模擬結果	- 6 -
	Vivado 模擬結果	- 8 -
	Cell based simulation	- 10 -
六、	FPGA 驗證.....	- 14 -
	系統架構圖.....	- 14 -
	Top module 輸入輸出介面(AXI-Stream).....	- 14 -
	Block Design	- 15 -
	Result.....	- 15 -
七、	結果與討論.....	- 16 -
	對照組.....	- 16 -
	實驗組(一顆 GG+一顆 GR)	- 17 -
八、	參考文獻.....	- 18 -

一、簡介

QR 分解是數值線性代數中具備多種用途的計算工具，主要應用於線性方程、最小平方法和特徵值問題。常見的 QR 分解的計算方法包括 Householder 變換、Givens rotation 以及 Gram-Schmidt 正交法。本文使用 given rotation 搭配 CORDIC Algorithms。

本次實作採用 8*4 矩陣，每個數字大小定義在 $\pm 0.25 \sim \pm 1$ ，預期得到一組 8*4 的上三角矩陣 R。實驗流程為先使用 MATLAB 估算預期使用 **定點數(fixed point)** 的長度(浮點數與定點數的誤差需足夠小)以及 iteration 的次數，再將 MATLAB 生成的隨機 8*4 矩陣以定點數格式匯入 verilog，並將 verilog 算出答案與 matlab 算出答案做比較，最後使用 FPGA 做驗證。

二、理論

➤ given rotation 說明

$$x' = x \cos \phi - y \sin \phi$$

$$y' = y \cos \phi + x \sin \phi$$

找到角度 Φ 使得 $y' = 0$ ，以圖一為例， $a_{1,1}$ 為 x' 、 $a_{2,1}$ 為 y' ，找出角度 Φ 後，需將右側同一列的數值皆經過同樣的旋轉矩陣運算(原理同基本矩陣第二定理)，之後依序將 $a_{3,1}$ 、 $a_{4,1}$... 變成 0，直到第一行除了 $a_{1,1}$ 外均變成 0。同理第二行，將 $a_{2,2}$ 為 x' 、 $a_{3,2}$ 為 y' 進行相同動作，直到第二行除了 $a_{2,2}$ 外皆變 0。持續到第 n 行(最後一行)，即可得到上三角矩陣 R。

$$\begin{bmatrix} \text{1} & \text{2} & \text{3} & & \\ \text{1} & a_{1,1} & a_{1,2} & a_{1,3} & \dots \\ \text{2} & a_{2,1} & a_{2,2} & a_{2,3} & \dots \\ \text{3} & a_{3,1} & a_{3,2} & a_{3,3} & \dots \\ & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

(圖一)given rotation 範例圖

➤ 使用 CORDIC 達成 given rotation

$$x' = \cos \phi \cdot [x - y \tan \phi]$$

$$y' = \cos \phi \cdot [y + x \tan \phi]$$

先將 $\cos \phi$ 提出，接著限制 $\tan(\phi) = \pm 2^{-i}$ ，即可將算式簡化如下：

$$x_{i+1} = K_i [x_i - y_i \cdot d_i \cdot 2^{-i}] \quad K_i = \cos(\tan^{-1} 2^{-i}) = 1/\sqrt{1+2^{-2i}}$$

$$y_{i+1} = K_i [y_i + x_i \cdot d_i \cdot 2^{-i}] \quad d_i = \pm 1$$

為了簡化運算量，將每次的運量係數省略，最後再乘上所有系數的乘積和 (A_N)

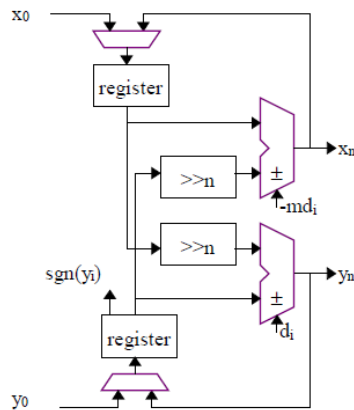
$$A_n = \prod_n \sqrt{1+2^{-2i}}$$

n 為疊代的次數

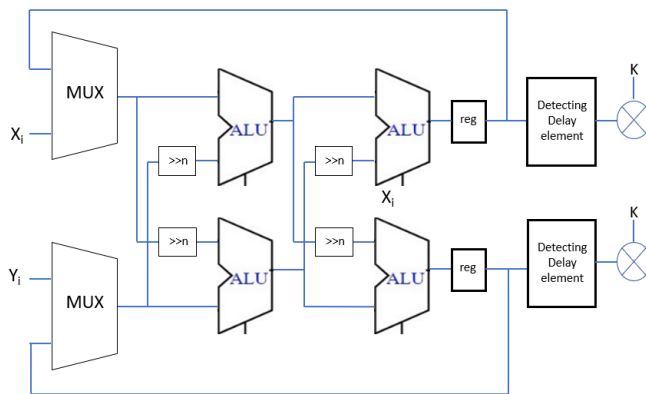
三、架構改良

➤ 改良一：PE Element 改良

(圖二)為論文中一顆 PE 元件的架構，(圖三)為改良後一顆 PE 元件的架構，最大的改進在於完成一次計算所需花的 CLK 數變原先的一半(unfolding factor(J)=2)。缺點為會增加 Critical path，但此架構的 Critical path 為乘法器，因此可以近乎無代價的提高 Performance



(圖二)論文 PE 元件



(圖三)改良後 PE 元件

➤ 改良二：input size 改良

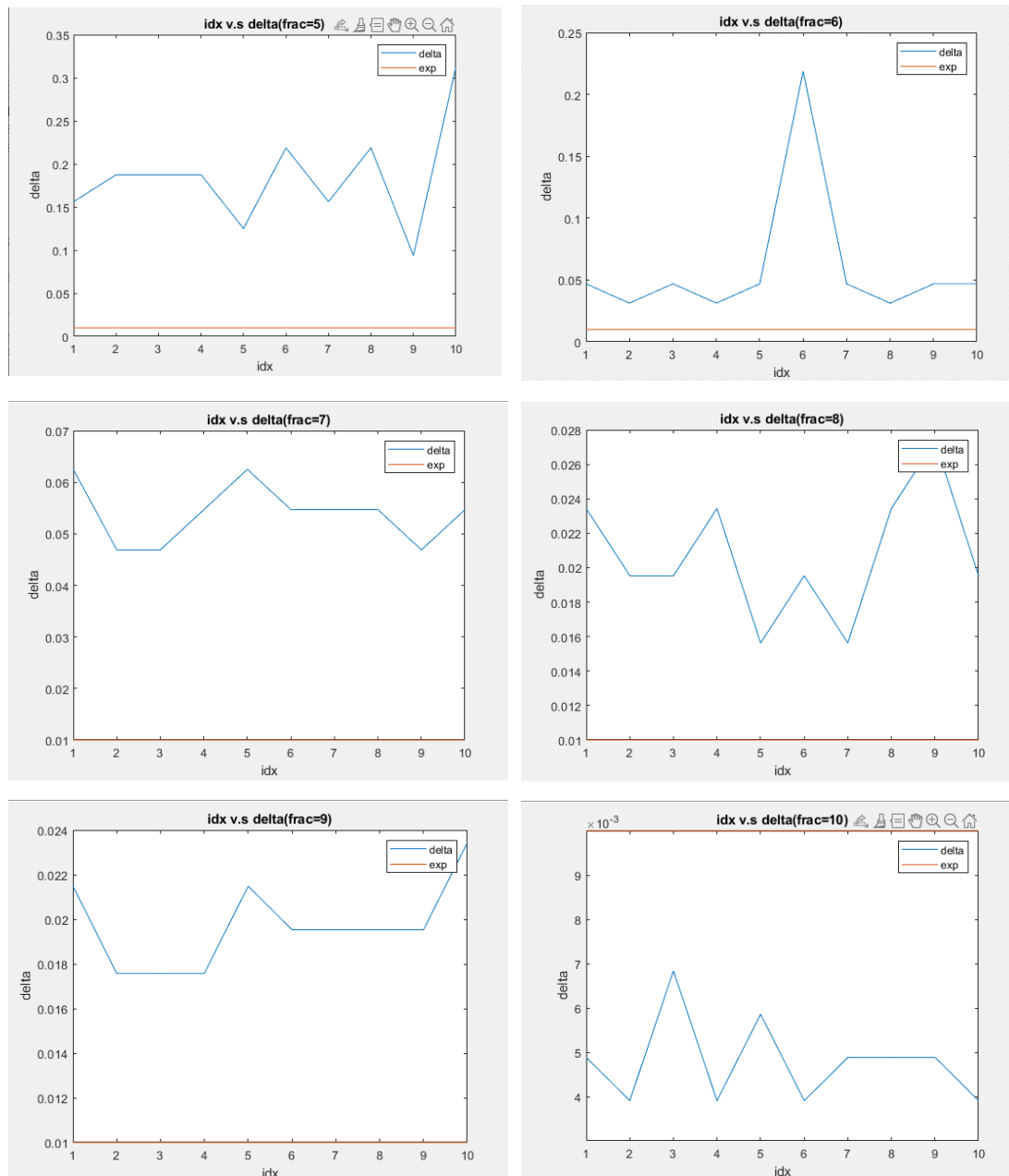
原論文疊代次數 n 為 9，為了使用上文圖三架構，我們須選擇偶數次疊代次數，因此此文選用疊代次數(n)為 8。此外原論文每個 input 為 16bits，此文將每個 input 改為 13bits(1 sign bit, 2 decimal bits, 10 fraction bits)，找法如下文。

首先定義 delta 函式如下：

$$\delta = \frac{\sqrt{\sum (r_{ij} - \hat{r}_{ij})^2}}{\sqrt{\sum r_{ij}^2}}$$

接著使用 matlab 畫圖，做法如下：

先將 Fraction bit 設定為 5 bit，皆著連續測試 10 組 δ 值，如果不滿足 $\delta < 0.01$ ，就將 Fraction bit 加 1，結果如下圖：



橫軸為 δ 值，縱軸為 index(從 1~10)，由上圖可以看出唯有 Fraction bit=10 時， δ 值會小於 0.01，最終測出 Fraction bit 最小需要 10 bit。

➤ 改良三：systolic array 改良

(圖四)為原論文架構，(圖五)為改良版架構，改良版為 pipeline 版本，在高頻下也可以成功運作。由於 R22 與 R12 間有資料相依，中間需要加 delay，delay 數由疊代次數(n)與 J(unfolding factor)有關，算式如下：

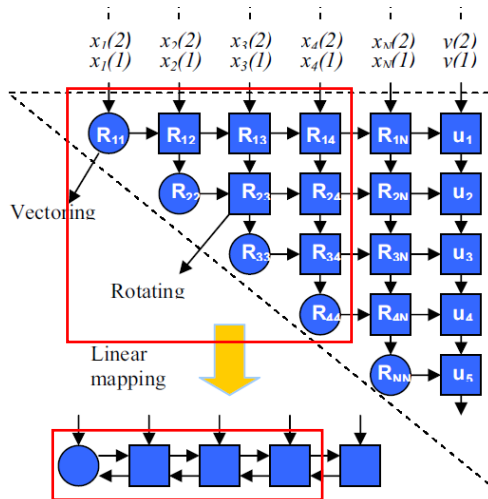
$$\text{Delay} \geq (n/J+1)*2$$

n/J 為 X_{ij} 執行 **Rotation mode** 的次數，加 1 為乘法器，乘 2 為有兩級資料相依。

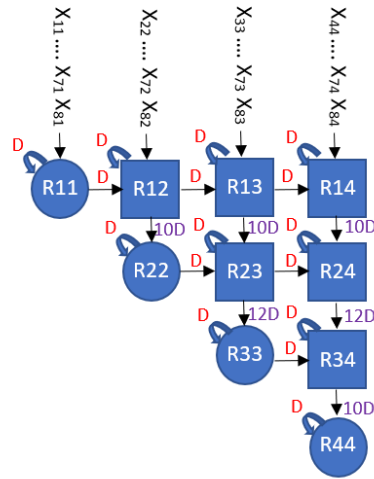
舉例：以 X_{72} 、 X_{82} 為例，R22 執行需等到 R12 執行完 X_{72} 、 X_{82} 才能計算。

同理(R13,R23)、(R14,R24)、(R34,R44)。

(R23,R33)、(R24,R34)多 delay 兩級目的為使用 **pipeline 技巧**，讓原先需要八顆乘法器降成 4 顆乘法器。最後結果如(圖五)。



(圖四)論文 systolic array



(圖五)改良後 systolic array

➤ 改良四：重新定義 K

從前文理論中，我們可以得知 A_n 如下：

$$A_n = \prod_n \sqrt{1+2^{-2i}}$$

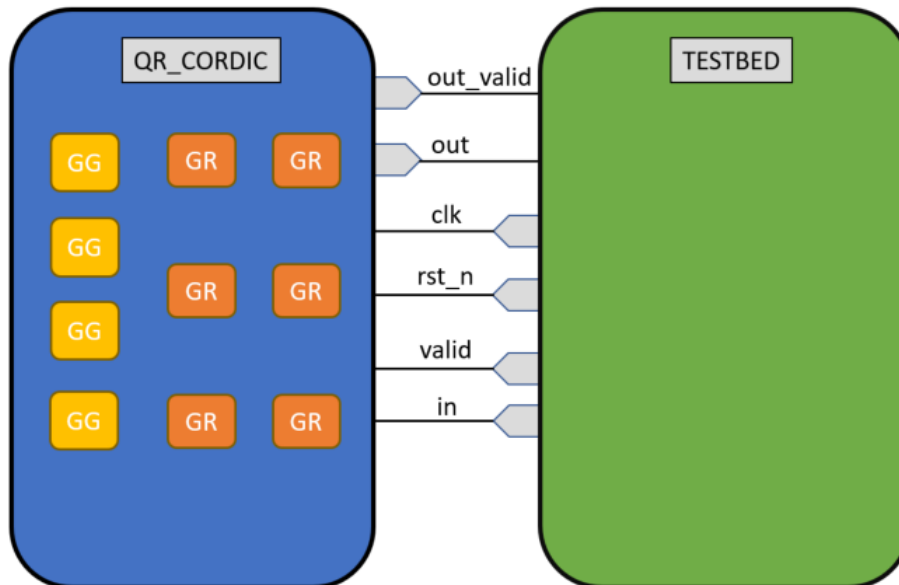
其中 n 為疊代次數，將 n=8 帶入，並將其重新定義成 K，可得結果如下：

$$K=0.6074$$

最後將其轉為 FIXED POINT(共 11 bits, sign bit : 1, fraction bit : 10)

四、設計規格

➤ 系統方塊圖



➤ QR_CORDIC 輸入輸出介面

Signal Name	I/O	Width	Simple Description
Clk	I	1	Clock Signal(posedge trigger)
Rst_n	I	1	Negedge reset
valid	I	1	valid 為 high，in 資料開始輸入
out_valid	O	1	out_valid 為 high，out 資料開始輸出
in	I	52	為第 i 列資料，4 筆 13bits 資料組成輸入
out	O	52	為第 i 列資料，4 筆 13bits 資料組成輸出

➤ GG 輸入輸出介面

Signal Name	I/O	Width	Simple Description
out_X	O	13	X_{ij} 資料輸出
out_Y	O	13	$X_{i(j-1)}$ 資料輸出
Sign_d	O	2	決定第 i 與(i+1)次疊代旋轉方向
Iter_num	I	3	為第 i 次疊代，i 為 0、2、4、6
In_X	I	13	X_{ij} 資料輸入
In_Y	I	13	$X_{i(j-1)}$ 資料輸入

➤ GR 輸入輸出介面

Signal Name	I/O	Width	Simple Description
out_X	0	13	X_{ij} 資料輸出
out_Y	0	13	$X_{i(j-1)}$ 資料輸出
Sign_d	1	2	決定第 i 與 $(i+1)$ 次疊代旋轉方向
Iter_num	1	3	為第 i 次疊代， i 為 0、2、4、6
In_X	1	13	X_{ij} 資料輸入
In_Y	1	13	$X_{i(j-1)}$ 資料輸入

五、模擬結果

➤ Matlab 模擬結果

1. 8*4 input matrix in floating

A1 =

```
-0.4453  -0.7188  -0.4297  -0.9297
 0.3594  -0.5156   0.5625   0.5234
 0.3516  -0.6328  -0.2813   0.3281
 0.9063   0.5469   0.9297  -0.8359
-0.6875   0.3047  -0.9609  -0.5391
-0.6641   0.4297  -0.6172  -0.4297
-0.3594  -0.3359   0.6172  -0.5547
 0.8906   0.3828   0.5000   0.3203
```

```
DataTypeMode: Fixed-point: binary point scaling
Signedness: Signed
WordLength: 13
FractionLength: 10
```

2. 8*4 input matrix in sign decimal

```
-456  -736  -440  -952
 368  -528   576   536
 360  -648  -288   336
 928   560   952  -856
-704   312  -984  -552
-680   440  -632  -440
-368  -344   632  -568
 912   392   512   328
```


3. 8*4 output matrix in floating

R_hat_cordic =

-1.7646	-0.2178	-1.3789	-0.6240
0.0010	-1.4063	0.1191	0.3076
-0.0049	0.0029	-1.2158	0.3896
-0.0039	-0.0020	-0.0068	-1.4883
-0.0029	0.0049	-0.0098	-0.0020
0.0039	0.0049	-0.0020	0.0049
0	0	-0.0049	0.0010
0.0029	-0.0020	0.0010	-0.0010

DataTypeMode: Fixed-point: binary point scaling
Signedness: Signed
WordLength: 13
FractionLength: 10

4. 8*4 output matrix in sign decimal

-1807	-223	-1412	-639
1	-1440	122	315
-5	3	-1245	399
-4	-2	-7	-1524
-3	5	-10	-2
4	5	-2	5
0	0	-5	1
3	-2	1	-1

5. Delta

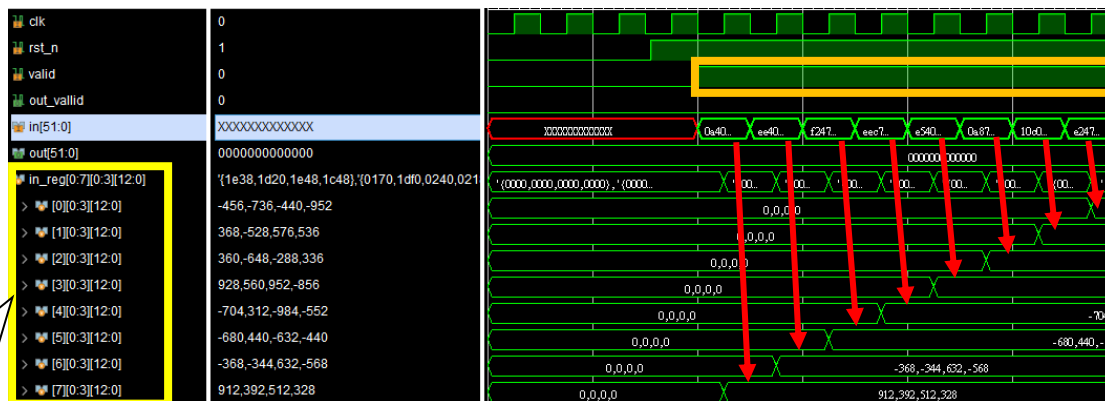
delta_p2 =

0.0039

DataTypeMode: Fixed-point: binary point scaling
Signedness: Signed
WordLength: 19
FractionLength: 10

➤ Vivado 模擬結果

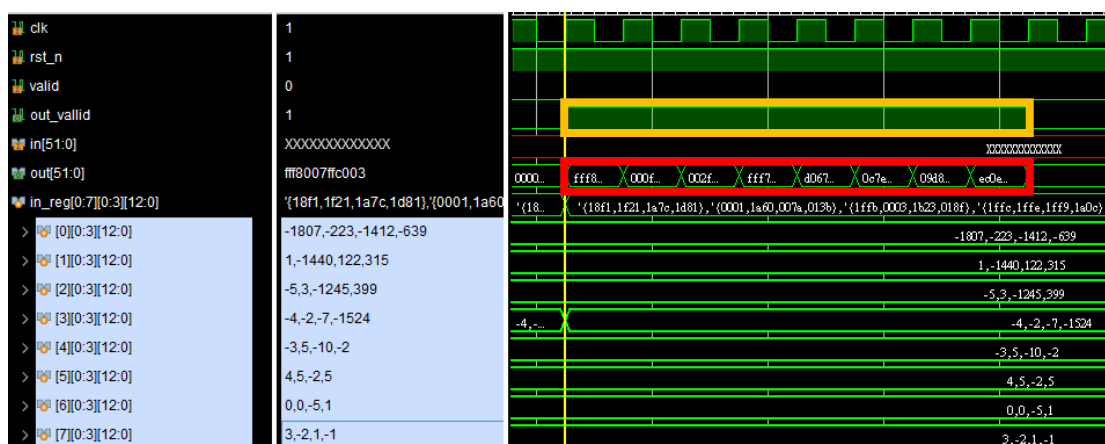
1. valid 拉起時



2. 中間計算的 control 訊號



3. outvalid 拉起時



4. TESTBED 最終測試結果

START!!! Simulation Start

Your input matrix is :

-456.000000	-736.000000	-440.000000	-952.000000
368.000000	-528.000000	576.000000	536.000000
360.000000	-648.000000	-288.000000	336.000000
928.000000	560.000000	952.000000	-856.000000
-704.000000	312.000000	-984.000000	-552.000000
-680.000000	440.000000	-632.000000	-440.000000
-368.000000	-344.000000	632.000000	-568.000000
912.000000	392.000000	512.000000	328.000000

Your matrix[8][0] is	3, expect matrix[8][0] is	3
Your matrix[8][1] is	-2, expect matrix[8][1] is	-2
Your matrix[8][2] is	1, expect matrix[8][2] is	1
Your matrix[8][3] is	-1, expect matrix[8][3] is	-1

Your matrix[7][0] is	0, expect matrix[7][0] is	0
Your matrix[7][1] is	0, expect matrix[7][1] is	0
Your matrix[7][2] is	-5, expect matrix[7][2] is	-5
Your matrix[7][3] is	1, expect matrix[7][3] is	1

Your matrix[6][0] is	4, expect matrix[6][0] is	4
Your matrix[6][1] is	5, expect matrix[6][1] is	5
Your matrix[6][2] is	-2, expect matrix[6][2] is	-2
Your matrix[6][3] is	5, expect matrix[6][3] is	5

Your matrix[5][0] is	-3, expect matrix[5][0] is	-3
Your matrix[5][1] is	5, expect matrix[5][1] is	5
Your matrix[5][2] is	-10, expect matrix[5][2] is	-10
Your matrix[5][3] is	-2, expect matrix[5][3] is	-2

Your matrix[4][0] is	-4, expect matrix[4][0] is	-4
Your matrix[4][1] is	-2, expect matrix[4][1] is	-2
Your matrix[4][2] is	-7, expect matrix[4][2] is	-7
Your matrix[4][3] is	-1524, expect matrix[4][3] is	-1524

Your matrix[3][0] is	-5, expect matrix[3][0] is	-5
Your matrix[3][1] is	3, expect matrix[3][1] is	3
Your matrix[3][2] is	-1245, expect matrix[3][2] is	-1245
Your matrix[3][3] is	399, expect matrix[3][3] is	399

Your matrix[2][0] is	1, expect matrix[2][0] is	1
Your matrix[2][1] is	-1440, expect matrix[2][1] is	-1440
Your matrix[2][2] is	122, expect matrix[2][2] is	122
Your matrix[2][3] is	315, expect matrix[2][3] is	315

Your matrix[1][0] is	-1807, expect matrix[1][0] is	-1807
Your matrix[1][1] is	-223, expect matrix[1][1] is	-223
Your matrix[1][2] is	-1412, expect matrix[1][2] is	-1412
Your matrix[1][3] is	-639, expect matrix[1][3] is	-639

-----Congratulations!-----
----- The test result is PASS -----

Your output matrix is :

```
-1807.000000  -223.000000  -1412.000000  -639.000000
   1.000000  -1440.000000   122.000000   315.000000
  -5.000000    3.000000  -1245.000000   399.000000
  -4.000000  -2.000000   -7.000000  -1524.000000
  -3.000000    5.000000  -10.000000   -2.000000
   4.000000    5.000000   -2.000000    5.000000
   0.000000    0.000000   -5.000000    1.000000
   3.000000  -2.000000    1.000000   -1.000000
```

The delta result is 0.0039, calculation time is 55 clk

➤ Cell based simulation

使用 CIC 0.13um 製程

1. Area report (area.log)

Library(s) Used:

slow (File: /usr/cad/Design_Kit/CBDK_IC_Contest_v2.5 SynopsysDC/db/slow.db)

```
Number of ports:          108
Number of nets:           12830
Number of cells:          12213
Number of combinational cells: 11692
Number of sequential cells:  521
Number of macros/black boxes: 0
Number of buf/inv:        1492
Number of references:      95

Combinational area:       91374.436870
Buf/Inv area:             5277.216517
Noncombinational area:    16863.668461
Macro/Black Box area:     0.000000
Net Interconnect area:    undefined (No wire load specified)
```

```
Total cell area:         108238.105330
Total area:               undefined
```

2. Timing report(timing.log)

Startpoint: row_index_gg1_reg[1]
(rising edge-triggered flip-flop clocked by clk)
Endpoint: in_reg_reg[7][0][12]
(rising edge-triggered flip-flop clocked by clk)
Path Group: clk
Path Type: max

Point	Incr	Path
clock clk (rise edge)	0.00	0.00
clock network delay (ideal)	0.00	0.00
row_index_gg1_reg[1]/CK (DFFSX4)	0.00	0.00 r
row_index_gg1_reg[1]/Q (DFFSX4)	0.39	0.39 f
U12903/Y (NAND2BX1)	0.18	0.57 f
U12904/Y (NOR2BX1)	0.10	0.67 r
U7881/Y (CLKBUF3)	0.20	0.87 r
U7730/Y (INVX1)	0.12	0.99 f
U12949/Y (OAI22XL)	0.24	1.23 r
U12399/Y (NOR3XL)	0.13	1.36 f
U12953/Y (NOR2X1)	0.14	1.50 r
U12954/Y (CLKINVX1)	0.10	1.59 f
U12955/Y (AOI2BB2X1)	0.23	1.83 f
U9998/Y (OAI211XL)	0.15	1.97 r
U12956/Y (XOR2X1)	0.16	2.14 r
U10369/Y (NOR2XL)	0.10	2.24 f
U12995/Y (OAI21X1)	0.14	2.39 r
U9069/Y (AOI21XL)	0.11	2.50 f
U12505/Y (OAI21XL)	0.24	2.74 r
U8528/Y (AOI21XL)	0.14	2.88 f
U8525/Y (OAI21XL)	0.27	3.15 r

U13040/Y (AOI21X1)	0.15	3.31 f
U13047/Y (AOI21X1)	0.16	3.46 r
U9104/Y (AOI21XL)	0.12	3.58 f
U12634/Y (AOI21XL)	0.21	3.79 r
U16823/CO (ADDFHX1)	0.23	4.01 r
U17192/CO (ADDFHX1)	0.27	4.28 r
U8297/Y (XOR2X2)	0.18	4.47 f
U8006/Y (BUF8)	0.15	4.61 f
U12574/Y (INVX3)	0.12	4.74 r
U16792/Y (XOR2X1)	0.16	4.90 r
U16796/Y (AOI21X1)	0.13	5.02 f
U7629/Y (AOI21XL)	0.30	5.33 r
U16822/Y (AOI21X1)	0.15	5.48 f
U7623/Y (AOI21XL)	0.30	5.77 r
U17195/Y (AOI21X1)	0.16	5.93 f
U17197/Y (AOI21X1)	0.19	6.12 r
U17208/Y (AOI21X1)	0.13	6.24 f
U17264/Y (AOI21X1)	0.18	6.42 r
U17275/Y (AOI21X1)	0.13	6.55 f
U17321/Y (AOI21X1)	0.16	6.70 r
U7936/Y (AOI21X1)	0.19	6.89 r
U7853/CO (ADDFXL)	0.26	7.15 r
U18594/Y (INVXL)	0.08	7.23 f
U7932/Y (MX2X2)	0.20	7.43 f
U12788/Y (MXI2X1)	0.13	7.56 r
U18954/Y (AOI222XL)	0.19	7.75 f
in_reg_reg[7][0][12]/D (DFFRX1)	0.00	7.75 f
data arrival time		7.75

Clock period
設定為 8

Slack 不能
為負

clock clk (rise edge)	8.00	8.00
clock network delay (ideal)	0.00	8.00
in_reg_reg[7][0][12]/CK (DFFRX1)	0.00	8.00 r
library setup time	-0.25	7.75
data required time		7.75

data required time		7.75
data arrival time		-7.75

slack (MET)		0.00

3. Power(power.log)

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

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 = 1.5125 mW (92%)
Net Switching Power = 130.5112 uW (8%)

Total Dynamic Power = 1.6430 mW (100%)
Cell Leakage Power = 68.4201 uW

Power Group	Internal Power	Switching Power	Leakage Power	Total 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	1.3738	2.3115e-02	1.5550e+07	1.4124	(82.53%)	
sequential	0.0000	0.0000	0.0000	0.0000	(0.00%)	
combinational	0.1387	0.1074	5.2870e+07	0.2990	(17.47%)	

Total	1.5125 mW	0.1305 mW	6.8420e+07 pW	1.7114 mW		
1						

4. Gatelevel simulation(neverilog.log)

```
SDF statistics: No. of Pathdelays = 37064   Annotated = 100.00% -- No. of Tchecks = 3126   Annotated = 100.00%
              Total      Annotated      Percentage
Path Delays   37064      37064          100.00
$width        1563      1563          100.00
$setuphold    1563      1563          100.00
```

Building instance overlay tables: Done

	Instances	Unique
Modules:	11485	112
UDPs:	877	2
Primitives:	21746	8
Timing outputs:	12202	50
Registers:	533	19
Scalar wires:	14821	-
Expanded wires:	52	1
Always blocks:	2	2
Initial blocks:	3	3
Pseudo assignments:	4	4
Timing checks:	4689	1050
Interconnect:	32119	-
Delayed tcheck signals:	1563	523
Simulation timescale:	lps	

Writing initial simulation snapshot: worklib.tb:v

Loading snapshot worklib.tb:v Done

Verdi Loading libsscore_ius152.so

ncsim> source /usr/cad/cadence/INCISIV/cur/tools/inca/files/ncsimrc

ncsim> run

START!!! Simulation Start

Your input matrix is :

-456.000000	-736.000000	-440.000000	-952.000000
368.000000	-528.000000	576.000000	536.000000
360.000000	-648.000000	-288.000000	336.000000
928.000000	560.000000	952.000000	-856.000000
-704.000000	312.000000	-984.000000	-552.000000
-680.000000	440.000000	-632.000000	-440.000000
-368.000000	-344.000000	632.000000	-568.000000
912.000000	392.000000	512.000000	328.000000

Your matrix[8][0] is 3, expect matrix[8][0] is 3
Your matrix[8][1] is -2, expect matrix[8][1] is -2
Your matrix[8][2] is 1, expect matrix[8][2] is 1
Your matrix[8][3] is -1, expect matrix[8][3] is -1

Your matrix[7][0] is 0, expect matrix[7][0] is 0
Your matrix[7][1] is 0, expect matrix[7][1] is 0
Your matrix[7][2] is -5, expect matrix[7][2] is -5
Your matrix[7][3] is 1, expect matrix[7][3] is 1

Your matrix[6][0] is 4, expect matrix[6][0] is 4
Your matrix[6][1] is 5, expect matrix[6][1] is 5
Your matrix[6][2] is -2, expect matrix[6][2] is -2
Your matrix[6][3] is 5, expect matrix[6][3] is 5

Your matrix[5][0] is -3, expect matrix[5][0] is -3
Your matrix[5][1] is 5, expect matrix[5][1] is 5
Your matrix[5][2] is -10, expect matrix[5][2] is -10
Your matrix[5][3] is -2, expect matrix[5][3] is -2

Your matrix[4][0] is -4, expect matrix[4][0] is -4
Your matrix[4][1] is -2, expect matrix[4][1] is -2
Your matrix[4][2] is -7, expect matrix[4][2] is -7
Your matrix[4][3] is -1524, expect matrix[4][3] is -1524

Your matrix[3][0] is -5, expect matrix[3][0] is -5
Your matrix[3][1] is 3, expect matrix[3][1] is 3
Your matrix[3][2] is -1245, expect matrix[3][2] is -1245
Your matrix[3][3] is 399, expect matrix[3][3] is 399

```

-----
Your matrix[2][0] is      1, expect matrix[2][0] is      1
Your matrix[2][1] is    -1440, expect matrix[2][1] is    -1440
Your matrix[2][2] is     122, expect matrix[2][2] is     122
Your matrix[2][3] is     315, expect matrix[2][3] is     315
-----
Your matrix[1][0] is    -1807, expect matrix[1][0] is    -1807
Your matrix[1][1] is     -223, expect matrix[1][1] is     -223
Your matrix[1][2] is    -1412, expect matrix[1][2] is    -1412
Your matrix[1][3] is     -639, expect matrix[1][3] is     -639
-----

```

```

-----Congratulations!-----
----- The test result is ..... PASS -----

```

Your output matrix is :

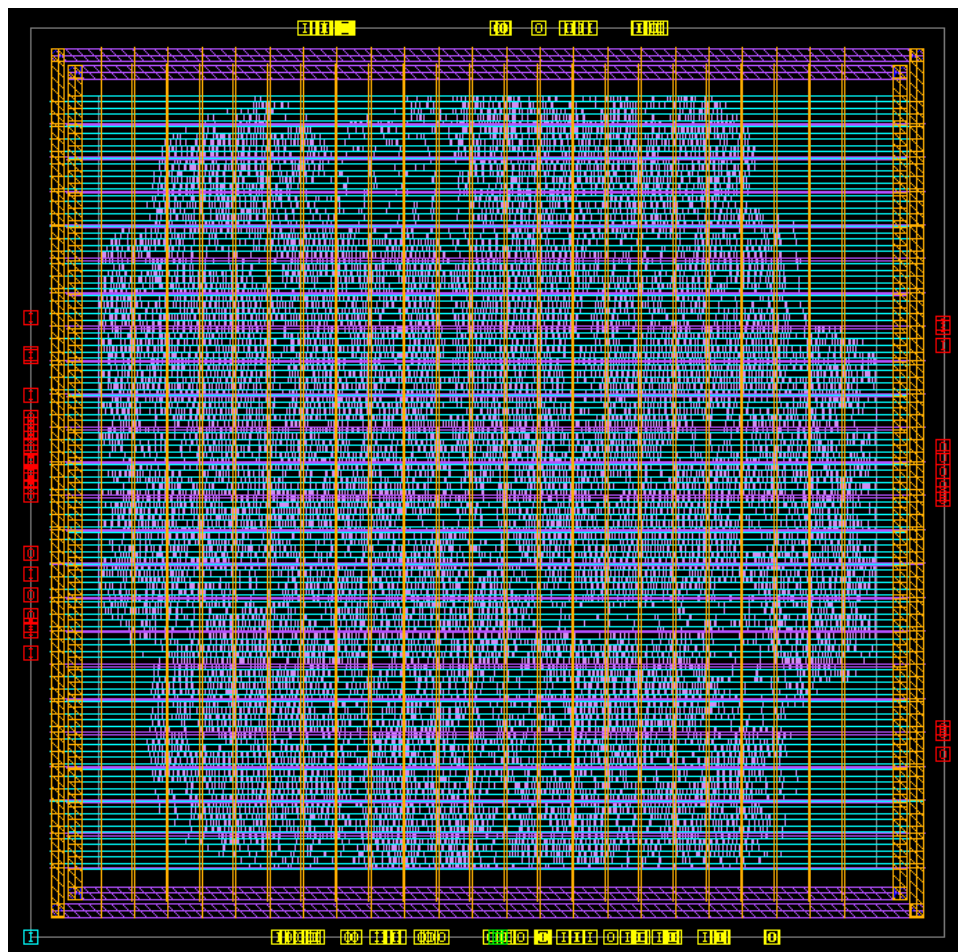
```

-1807.000000  -223.000000  -1412.000000  -639.000000
  1.000000   -1440.000000   122.000000   315.000000
 -5.000000    3.000000  -1245.000000   399.000000
 -4.000000   -2.000000   -7.000000  -1524.000000
 -3.000000    5.000000  -10.000000   -2.000000
  4.000000    5.000000   -2.000000    5.000000
  0.000000    0.000000   -5.000000    1.000000
  3.000000   -2.000000    1.000000   -1.000000

```

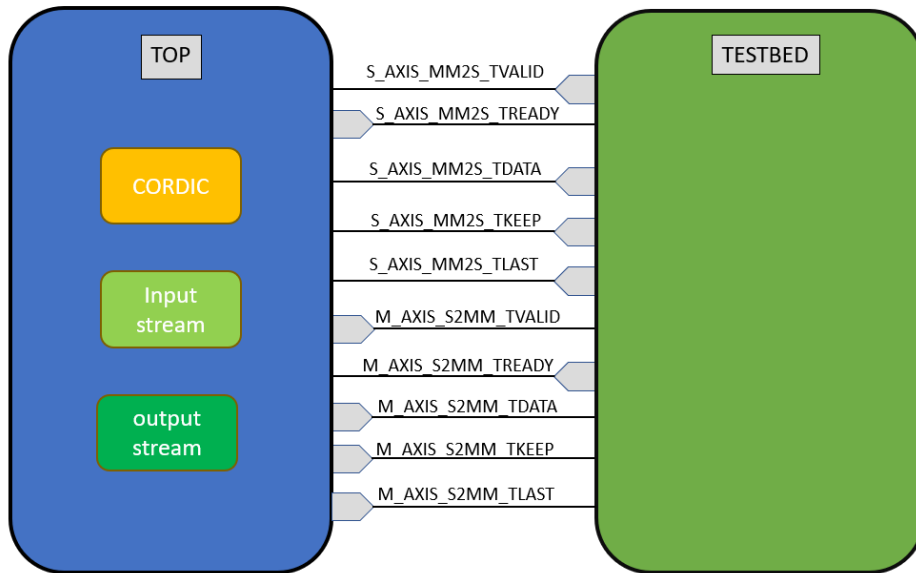
The delta result is 0.0039, calculation time is 54 clk

5. APR



六、FPGA 驗證

➤ 系統架構圖



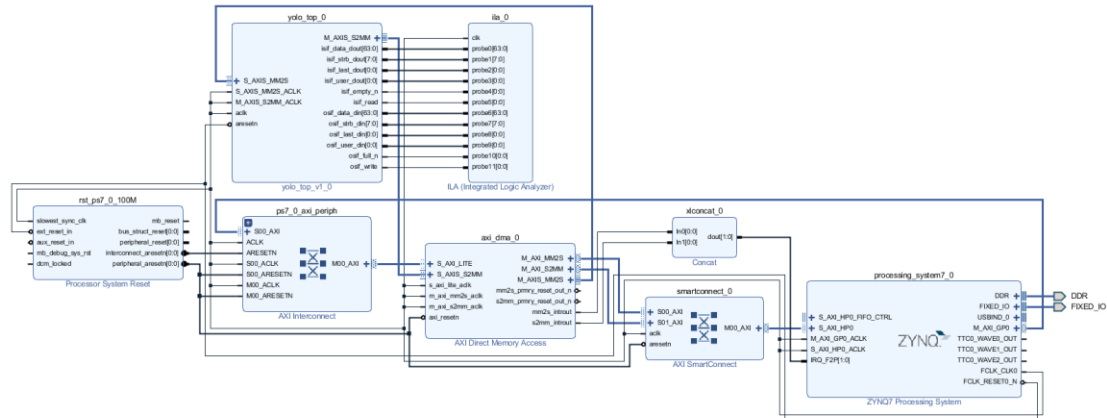
Top module 輸入輸出介面(AXI-Stream)

Signal Name	I/O	Width	Simple Description
S_AXIS_MM2S_TVALID	I	1	TVALID 表示主設備正在驅動一個有效的傳輸。當 TVALID 和 TREADY 都置位時，發生一個傳輸。
S_AXIS_MM2S_TREADY	O	1	TREADY 表示從設備在當前週期能夠接收一次傳輸。
S_AXIS_MM2S_TDATA	I	64	TDATA 是基本的有效載荷，用來提供跨越接口的數據。數據為整數個字節。
S_AXIS_MM2S_TKEEP	I	8	TKEEP 是字節修飾符。用來表明 TDATA 相關字節的內容是否作為數據流的一部分被處理。TKEEP 字節修飾符未被確認的那些相關字節是空字節，可以從數據流中去除。
S_AXIS_MM2S_TLAST	I	1	TLAST 表明了包的邊界。
aclk	I	1	Clock Signal(posedge trigger)
aresetn	I	1	negedge reset
M_AXIS_S2MM_TVALID	O	1	同上
M_AXIS_S2MM_TREADY	I	1	同上
M_AXIS_S2MM_TDATA	O	64	同上

M_AXIS_S2MM_TKEEP	O	8	同上
M_AXIS_S2MM_TLAST	O	1	同上

備註: (M2S:master to slave、S2M:slave to master)

➤ Block Design



➤ Result

1. Timing Report

Setup	Hold	Pulse Width
Worst Negative Slack (WNS): 0.160 ns	Worst Hold Slack (WHS): 0.018 ns	Worst Pulse Width Slack (WPWS): 5.750 ns
Total Negative Slack (TNS): 0.000 ns	Total Hold Slack (THS): 0.000 ns	Total Pulse Width Negative Slack (TPWS): 0.000 ns
Number of Failing Endpoints: 0	Number of Failing Endpoints: 0	Number of Failing Endpoints: 0
Total Number of Endpoints: 28306	Total Number of Endpoints: 28290	Total Number of Endpoints: 11347

All user specified timing constraints are met.

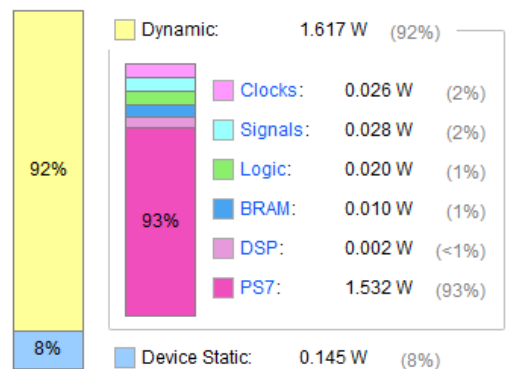
2. Power Report

Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.

Total On-Chip Power:	1.762 W
Design Power Budget:	Not Specified
Power Budget Margin:	N/A
Junction Temperature:	45.3°C
Thermal Margin:	39.7°C (3.3 W)
Effective θ_{JA} :	11.5°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Medium

[Launch Power Constraint Advisor](#) to find and fix invalid switching activity

On-Chip Power



3. Total Report

Name	Constraints	Status	WNS	TNS	WHS	THS	TPWS	Total Power	Failed Routes	LUT	FF	BRAMs	URAM	DSP
✓ synth_1	constrs_1	synth_design Complete!								10437	10130	17.50	0	4
✓ impl_1	constrs_1	route_design Complete!	0.160	0.000	0.018	0.000	0.000	1.762	0	9651	9571	17.50	0	4

七、結果與討論

➤ 對照組

```
-----
Your matrix[8][0] is      3, expect matrix[8][0] is      3
Your matrix[8][1] is     -2, expect matrix[8][1] is     -2
Your matrix[8][2] is      1, expect matrix[8][2] is      1
Your matrix[8][3] is     -1, expect matrix[8][3] is     -1
-----
Your matrix[7][0] is      0, expect matrix[7][0] is      0
Your matrix[7][1] is      0, expect matrix[7][1] is      0
Your matrix[7][2] is     -5, expect matrix[7][2] is     -5
Your matrix[7][3] is      1, expect matrix[7][3] is      1
-----
Your matrix[6][0] is      4, expect matrix[6][0] is      4
Your matrix[6][1] is      5, expect matrix[6][1] is      5
Your matrix[6][2] is     -2, expect matrix[6][2] is     -2
Your matrix[6][3] is      5, expect matrix[6][3] is      5
-----
Your matrix[5][0] is     -3, expect matrix[5][0] is     -3
Your matrix[5][1] is      5, expect matrix[5][1] is      5
Your matrix[5][2] is    -10, expect matrix[5][2] is    -10
Your matrix[5][3] is     -2, expect matrix[5][3] is     -2
-----
Your matrix[4][0] is     -4, expect matrix[4][0] is     -4
Your matrix[4][1] is     -2, expect matrix[4][1] is     -2
Your matrix[4][2] is     -7, expect matrix[4][2] is     -7
Your matrix[4][3] is  -1524, expect matrix[4][3] is  -1524
-----
Your matrix[3][0] is     -5, expect matrix[3][0] is     -5
Your matrix[3][1] is      3, expect matrix[3][1] is      3
Your matrix[3][2] is  -1245, expect matrix[3][2] is  -1245
Your matrix[3][3] is     399, expect matrix[3][3] is     399
-----
Your matrix[2][0] is      1, expect matrix[2][0] is      1
Your matrix[2][1] is  -1440, expect matrix[2][1] is  -1440
Your matrix[2][2] is     122, expect matrix[2][2] is     122
Your matrix[2][3] is     315, expect matrix[2][3] is     315
-----
Your matrix[1][0] is  -1807, expect matrix[1][0] is  -1807
Your matrix[1][1] is   -223, expect matrix[1][1] is   -223
Your matrix[1][2] is  -1412, expect matrix[1][2] is  -1412
Your matrix[1][3] is   -639, expect matrix[1][3] is   -639
```

```
-----Congratulations!-----
----- The test result is ..... PASS -----
```

```
The delta result is 0.0039, calculation time is 55 clk
```

➤ 實驗組(一顆 GG+一顆 GR)

```
Output pixel      0 are wrong!the real output s 1694.000000, but expected result is 1693.000000
Output pixel      1 are wrong!the real output s 1.000000, but expected result is 2.000000
Output pixel      2 are wrong!the real output s 6.000000, but expected result is 8.000000
Output pixel      3 are wrong!the real output s 4.000000, but expected result is 5.000000
Output pixel      6 are wrong!the real output s 3.000000, but expected result is 4.000000
Output pixel      8 are wrong!the real output s 793.000000, but expected result is 792.000000
Output pixel      9 are wrong!the real output s 1846.000000, but expected result is 1847.000000
Output pixel     12 are wrong!the real output s -8.000000, but expected result is -10.000000
Output pixel     13 are wrong!the real output s -3.000000, but expected result is 0.000000
Output pixel     14 are wrong!the real output s -8.000000, but expected result is -7.000000
Output pixel     15 are wrong!the real output s 0.000000, but expected result is 1.000000
Output pixel     16 are wrong!the real output s 130.000000, but expected result is 129.000000
Output pixel     17 are wrong!the real output s 324.000000, but expected result is 325.000000
Output pixel     18 are wrong!the real output s -1173.000000, but expected result is -1170.000000
Output pixel     19 are wrong!the real output s 7.000000, but expected result is 8.000000
Output pixel     20 are wrong!the real output s 5.000000, but expected result is 4.000000
Output pixel     21 are wrong!the real output s -6.000000, but expected result is -5.000000
Output pixel     22 are wrong!the real output s 3.000000, but expected result is 0.000000
Output pixel     23 are wrong!the real output s -1.000000, but expected result is 1.000000
Output pixel     24 are wrong!the real output s 944.000000, but expected result is 943.000000
Output pixel     25 are wrong!the real output s 919.000000, but expected result is 923.000000
Output pixel     26 are wrong!the real output s -123.000000, but expected result is -117.000000
Output pixel     27 are wrong!the real output s -1586.000000, but expected result is -1583.000000
Output pixel     28 are wrong!the real output s 9.000000, but expected result is 10.000000
Output pixel     29 are wrong!the real output s -11.000000, but expected result is -10.000000
Output pixel     30 are wrong!the real output s -9.000000, but expected result is 7.000000
Output Pixel: 0 ~      31 are wrong ! The wrong pixel reached a total of      26 or more !
```

FAIL! There are 26 errors at functional simulation !

```
----- The test result is .....FAIL -----
----- The calculation time is ..... 491 -----
```

實驗組僅將 PE 元件改成一顆 GG+一顆 GR(即表示無 systolic array)，其餘皆固定，可得結果如下：

1. 如上圖黃框所標處，有 systolic array 的總計算時間少了將近 9 倍
2. expect 值為 matlab 計算後的值，造成實驗組紅框處 expect value 與 output value 不同的原因在於 rounding 的差異，需將 matlab 的 fi 函數 rounding 方式改為“Floor”，才可以使 expect value 與 output value 符合預期。

八、 参考文献

- [1] D. Boppana, K. Dhanoa and J. Kempa, "FPGA based embedded processing architecture for the QRD-RLS algorithm," *12th Annual IEEE Symposium on Field-Programmable Custom Computing Machines*, 2004, pp. 330-331, doi: 10.1109/FCCM.2004.34
- [2] ANDRAKA, Ray. A survey of CORDIC algorithms for FPGA based computers. In: *Proceedings of the 1998 ACM/SIGDA sixth international symposium on Field programmable gate arrays*. 1998. p. 191-200