數位積體電路設計期中報告 – 1d DCT

第三組

學號：411086006 姓名：張愷和

學號：411086022 姓名：張君愷

一、 DCT 介紹

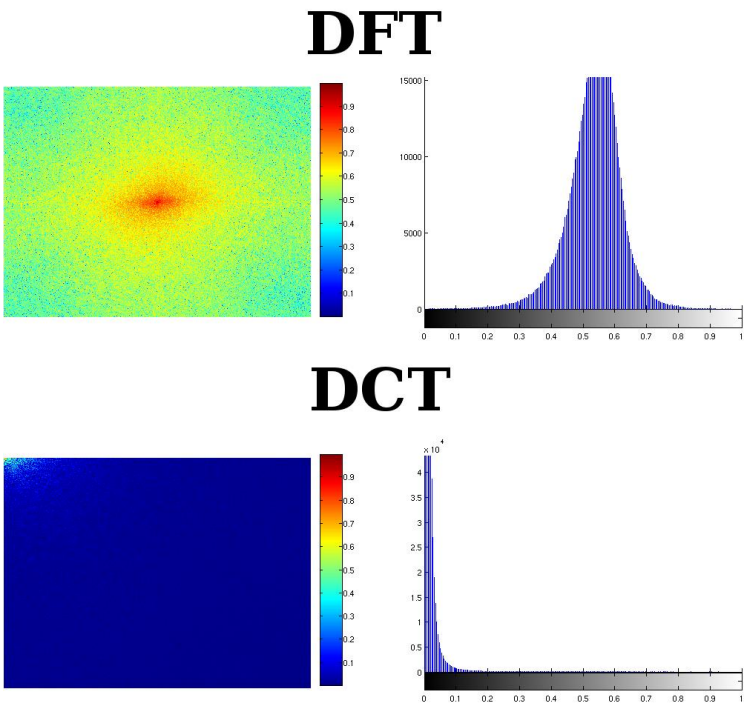
DCT，discrete cosine transform，一種類似傅立葉轉換的方法，主要目的應用在影像壓縮(JPEG 壓縮)、音訊編碼、圖片編碼等等。由於之前有修過多媒體訊號處理的課程，也有實作過模擬 JPEG 壓縮，因此來說明一下 DCT 在 JPEG 中的應用。

在圖片和聲波一樣，有分高頻和低頻的區域，如果像素數值與附近像素數值相差較大，可以稱區域為高頻區域，如果像素數值與附近像素數值相差較小，可以稱區域為低頻區域(圖一)。由於人眼對於高頻訊號的改變

圖一：圖片高頻區域(紅圈)、圖片低頻區域(藍圈)

沒有那麼敏感，因此可以將圖片由像素數值，轉換為頻域數值，之後對高頻區域做量化，刪減高頻的數值，這樣可以達到壓縮的效果，並且不太會影響到圖片視覺上的體驗。

不過可以使用 DFT(discrete fourier transform)或 FFT(fast fourier transform)來轉換頻域，為什麼要用 DCT 來轉換，從圖二可以看到 DCT 轉換可以對低頻保留較多的訊息，轉換範圍較小，儲存訊息量較多，但是 DFT 轉換後分布的範圍較廣，低頻儲存量較小，做量化會刪掉太多的細節，因此 DCT 來轉換是比較好的選擇。



圖二：DFT 和 DCT 轉換結果差異

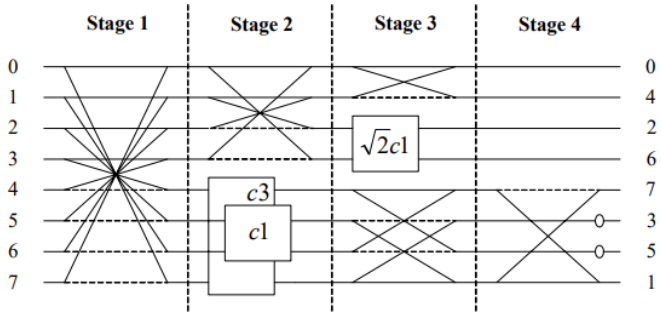
以下是 DCT(式一)和 inverse DCT(式二)的數學公式：

$$F(k)=\begin{cases}\frac{1}{\sqrt{N}}\sum_{i=0}^{N-1}f(i), k=0\\ \sqrt{\frac{2}{N}}\sum_{i=0}^{N-1}f(i)\cos\frac{(2i+1)k\pi}{2N}, k=1,2,3,\dots,N-1\end{cases} \quad \text{(式 1)}$$

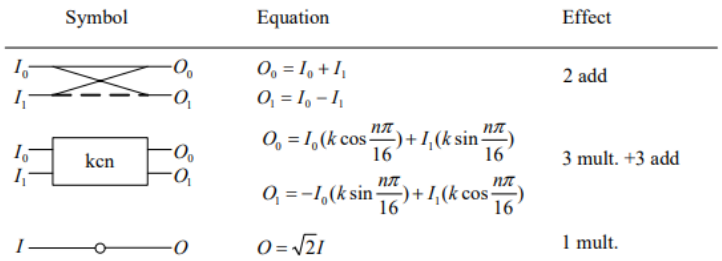
$$f(i)=\frac{1}{\sqrt{N}}F(0)+\sqrt{\frac{2}{N}}\sum_{k=0}^{N-1}F(k)\cos\frac{(2i+1)k\pi}{2N} \quad \text{(式 2)}$$

上面兩個公式如果要直接實作在 verilog 中的話，會用到比較多的加法器以及乘法器，使用到較多的加法器和乘法器，對應設計的功耗以及效率會較差，因此有人簡化了 DCT 演算法，得到了另一個 loeffler DCT algorithm，這個演算法總共會使用 29 個加法器以及 11 個乘法器。以下圖片是 loeffler DCT 演算法計算的階段，以及各個 block 計算的方式。在圖四中可以看到 kcn symbol equation 上會使用到四個乘法和兩個加法，但是 effect 上顯示的乘法器和加法器數量都是 3，會減少一個乘法器是因為 equation 可以改成另一種公式寫法(式三)，所以結果來說可以在減少一個乘法器。

$$\begin{cases}O_0=aI_0+bI_1=(b-a)I_1+a(I_0+I_1)\\ O_1=-bI_0+aI_1=-(a+b)I_0+a(I_0+I_1)\end{cases} \quad \text{(式三)}$$



圖三：loeffler's DCT algorithm 計算 stage



圖四：各個 symbol 代表的計算方法

二、 1d-DCT 實作 verilog 程式碼說明

程式碼有實作兩個版本，第一個版本是按照上方 loeffler's DCT algorithm 一步一步按照 stage 的方法計算的，總共使用了 29 個 adder 以及 11 個 multiplier，但是可能是我 verilog 學藝不精，以結果來說用 design compiler 產出的面積和時間都不太理想，使用 100MHz clk 來模擬這個程式，slack time 非常接近 0，使用 primetime 測出來的時間報告會變成負數。

第二個版本是直接使用 always blocking 寫法，這裡就可以用像是 c 語言的寫法，直接將教授給的 c 語言範例翻譯成 verilog 語言。結果來說 design compiler 合成出來的面積和計算的 power 比第一個版本還要小，但是 slack time 差不多，使用 primetime 測出來的時間報告也是會變成負數。

1. Version 1 code

首先需要設定一些常數項，這裡的常數項設定唯一些已經計算好的 cos 和 sin 值，不過 verilog 中不能直接計算小數，所以需要將計算好的常數乘上 2 的次方項，次方越大之後計算出來的結果精度會越好，但是計算過程中需要開較大空間的 wire 來避免計算出現 overflow，最後結果需要將之前乘上的 256 除回來，實作上使用右移 8 位元實現除法，圖五是設定各項 sin、cos 計算常數。

```
//b sin //a cos   //中間的a和m代表加和減
//b-a sin - cos   //sq表示有根號2
//b+a sin + cos   //數字表示對應的kcn n數值

wire signed [10:0] c1as1 = 9'd301;
wire signed [10:0] s1mc1 = -201;
wire signed [10:0] c1 = 9'd251;

wire signed [10:0] c3as3 = 9'd355;
wire signed [10:0] s3mc3 = -71;
wire signed [10:0] c3 = 9'd213;

wire signed [10:0] sqc1as1 = 9'd470;
wire signed [10:0] sqs1mc1 = 9'd196;
wire signed [10:0] sqc1 = 9'd137;

wire signed [10:0] sq2 = 9'd362;
```

圖五：各項計算常數設定，都有在乘上 256(2^8)

接著就可以將各項數值通過各個 stage 來計算結果，再將結果丟到下個 stage 計算。

Stage 1：

```
//stage = 1/////////////////

assign s10 = win0 + win7;
assign s17 = win0 - win7;
assign s11 = win1 + win6;
assign s16 = win1 - win6;
assign s12 = win2 + win5;
assign s15 = win2 - win5;
assign s13 = win3 + win4;
assign s14 = win3 - win4;
```

圖六：stage 1 計算程式碼

Stage 2：

```
//stage = 2/////////////////////////////////////////////////////////////////

assign s20 = s10 + s13;
assign s23 = s10 - s13;
assign s21 = s11 + s12;
assign s22 = s11 - s12;

//c3
assign tmp01 = s14 + s17;
mult_kcn m1(s17, s3mc3, s14, c3as3, tmp01, c3, s24, s27);  // multi * 3 and adder * 2

//c1
assign tmp02 = s15 + s16;
mult_kcn m2(s16, s1mc1, s15, c1as1, tmp02, c1, s25, s26);
```

圖七：stage 2 計算程式碼

```
assign tmp1 = (in1 * in2) >> 8;
assign tmp2 = (in5 * in6) >> 8;
assign out1 = tmp1 + tmp2;
assign tmp3 = (in3 * in4) >> 8;
assign out2 = tmp2 - tmp3;
```

圖八：對應的 mult_kcn module 計算方法(式 3 計算方法)

Stage 3：

```
assign s30 = s20 + s21;
assign s31 = s20 - s21;

assign tmp03 = s22 + s23;
mult_kcn m3(s23, sqs1mc1, s22, sqc1as1, tmp03, sqc1, s32, s33);

assign s34 = s24 + s26;
assign s36 = s24 - s26;
assign s37 = s27 + s25;
assign s35 = s27 - s25;
```

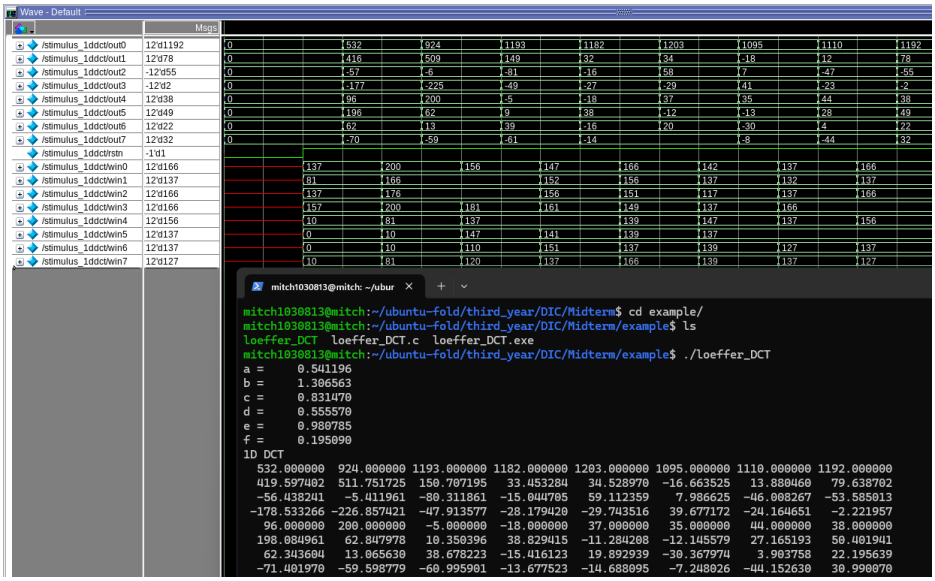
圖九：stage 3 計算過程，mult_kcn 可參照圖八

Stage 4：

```
assign o0 = s30;
assign o4 = s31;
assign o2 = s32;
assign o6 = s33;

assign o1 = s37 + s34;
assign o7 = s37 - s34;

assign o3 = (s35 * sq2) >> 8;
assign o5 = (s36 * sq2) >> 8;
```

圖十：最後 stage 4 計算過程

首先使用 vsim 先模擬計算是否正確，以下比較 verilog 和 c 語言計算出來的結果。



圖十一：c 語言計算和 verilog 計算數據差異

由圖十一可以看到計算公式上基本上應該是沒有出錯，沒有看到太離譜的數值錯誤，至於計算精度上面，最差有到正負 3，看起來只乘上 2 的 8 次方計算精度還是沒有那麼好。接下來使用 design compiler 來看 area、time 和 power 的結果。

```
****************************************
Report : area
Design : loeffler_1d
Version: R-2020.09
Date   : Mon Apr 15 14:17:54 2024
****************************************

Library(s) Used:

    fsa0m_a_generic_core_tt1p8v25c (File: /home/cad/tech/CBDK018_UMC_Faraday_v1.1/CIC/SynopsysDC/db/fsa0m_a_generic_core_tt1p8v25c.db)

Number of ports:                         2077
Number of nets:                          5918
Number of cells:                         3320
Number of combinational cells:           3181
Number of sequential cells:                96
Number of macros/black boxes:               0
Number of buf/inv:                        464
Number of references:                      37

Combinational area:              99081.158740
Buf/Inv area:                     3962.246368
Noncombinational area:            6599.577393
Macro/Black Box area:                0.000000
Net Interconnect area:      undefined  (Wire load has zero net area)

Total cell area:                105680.736132
Total area:                 undefined
```

圖十二：合成 area 結果，total cell area 為 105690.736132

```
mult_107/U173/O (NR2)                                   0.065     14.291 f
mult_107/U168/O (NR2)                                   0.148     14.439 r
mult_107/U207/O (INV1S)                                 0.076     14.515 f
mult_107/U164/O (OAI12HS)                               0.191     14.706 r
mult_107/U231/O (AOI12HS)                               0.091     14.796 f
mult_107/U229/O (XOR2HS)                                0.104     14.900 r
mult_107/product[18] (loeffler_1d_DW_mult_tc_5)         0.000     14.900 r
out3_reg[10]/D (QDFFRBT)                                0.000     14.900 r
data arrival time                                                 14.900

clock clk (rise edge)                                  15.000     15.000
clock network delay (ideal)                             0.500     15.500
clock uncertainty                                      -0.500     15.000
out3_reg[10]/CK (QDFFRBT)                               0.000     15.000 r
library setup time                                     -0.099     14.901
data required time                                                14.901
---------------------------------------------------------------------------
data required time                                                14.901
data arrival time                                                -14.900
---------------------------------------------------------------------------
slack (MET)                                                       0.001
```

圖十三：time 結果，slack time 為 0.001

從圖十三的結果來看，clock 設定為 100MHz 的話會差一點點就 violate slack time。

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	1.1014	1.2252	2.8724e+04	2.3266	(18.26%)	
sequential	0.0000	0.0000	0.0000	0.0000	(0.00%)	
combinational	5.2794	5.1374	3.7010e+05	10.4171	(81.74%)	
Total	6.3807 mW	6.3626 mW	3.9883e+05 pW	12.7437 mW		

圖十四：預測的 total power 為 12.7437mW

版本一的程式碼測試到這邊後，就開始改良程式碼架構，一開始式認為說中間宣告的 wire 太多占太多空間，於是就把中間的線一個一個拔掉和合併計算公式，但是出來的結果 slack time 只多了 0.01，跑 primetime slack time 還是會爆掉，面積跟 power 也沒有變得多好，之後使用了 always blocking 的寫法，以面積跟 power 來說有比較好，但是 slack time 還是一樣，不過最後還是決定用第二個版本的程式碼來跑 primetime，這樣 power 的數據比較好看。所以來看一下第二個版本的程式碼吧。

2. Version 2 code

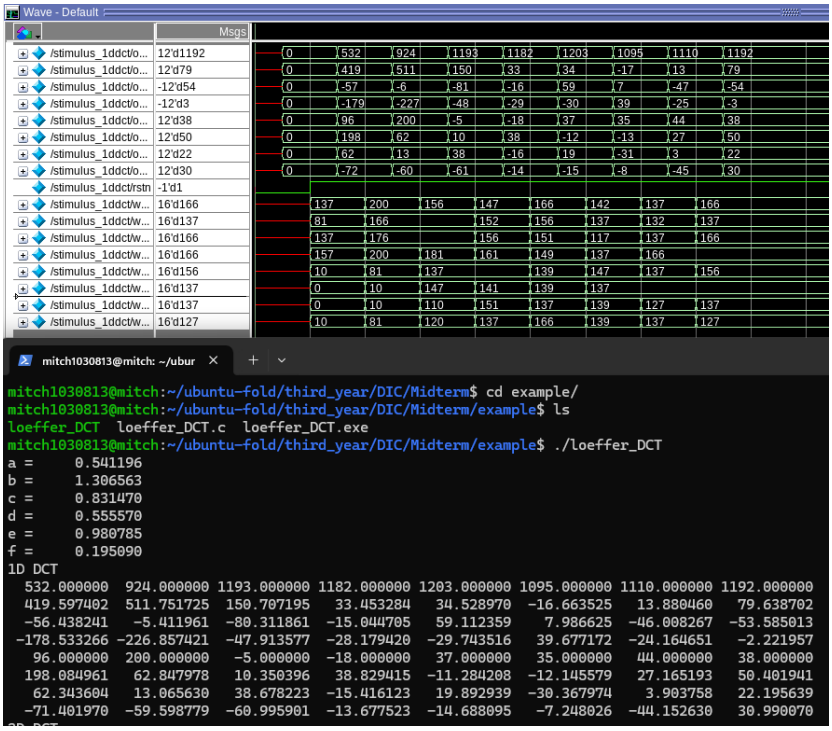
這個版本使用 always blocking 方法實作，以下是程式碼架構，計算過程是放在 always block 裡面。

```
tmp = (vp0 + vp1 + vp2 + vp3);
out0 = tmp;
tmp = (vp0 - vp1 - vp2 + vp3);
out4 = tmp;
tmp = ((vp1-vp2)*a + (vp0-vp3)*b) >> 10;
out2 = tmp;
tmp = ((vp2-vp1)*b + (vp0-vp3)*a) >> 10;
out6 = tmp;
tmp = ((((-vn3)*d+vn0*c)+(vn2*e+vn1*f))-((vn3*c+vn0*d) + ((-vn2)*f+vn1*e))) >> 10;
out7 = tmp;
tmp = (PT*(((-vn3)*d+vn0*c)-(vn2*e+vn1*f))) >> 20;
out3 = tmp;
tmp = (PT*((vn3*c+vn0*d)-((-vn2)*f+vn1*e))) >> 20;
out5 = tmp;
tmp = ((((-vn3)*d+vn0*c)+(vn2*e+vn1*f))+((vn3*c+vn0*d) + ((-vn2)*f+vn1*e))) >> 10;
out1 = tmp;
```

圖十五：loeffler DCT 合併計算方式

圖十五的程式碼中可以想成是 c 語言的順序執行方法，上面算完換下面計算，計算精度從 8 次方改成 10 次方，上面計算的公式可以認為說是將版本 1 的程式碼，stage 計算全部拉在一起寫成一個給結果的等式，直接來看 adder 和 multiplier 應該是比版本 1 還要多，但是結果來看這個寫法面積和 power 卻比版本 1 好了不少，目前還不太清楚是為甚麼。

接下來使用 vsim 來看計算結果是否有誤，首先用範例資料來檢查。



圖十六：比較版本二程式碼計算 DCT 和 c 語言計算 DCT 精度

從圖十六模擬結果來看，誤差幾本上在正負 1，看起來乘上 10 次方會比乘上 8 次方的精度比起來好太多了，結果看來程式碼也沒有計算上的錯誤。

接著使用 design compiler 來看合成的 time、area 和 power。

```
****************************************
Report : area
Design : loeffler_1d
Version: R-2020.09
Date   : Mon Apr 15 14:51:47 2024
****************************************

Library(s) Used:

    fsa0m_a_generic_core_tt1p8v25c (File: /home/cad/tech/CBDK018_UMC_Faraday_v1.1/CIC/SynopsysDC/db/fsa0m_a_generic_core_tt1p8v25c.db)

Number of ports:                         2695
Number of nets:                          5376
Number of cells:                         2219
Number of combinational cells:           2076
Number of sequential cells:                96
Number of macros/black boxes:               0
Number of buf/inv:                        542
Number of references:                      66

Combinational area:              78048.130208
Buf/Inv area:                     5140.295927
Noncombinational area:            6299.596436
Macro/Black Box area:                0.000000
Net Interconnect area:      undefined  (Wire load has zero net area)

Total cell area:                 84347.726643
Total area:                 undefined
```

圖十七：合成 area 結果，total cell area 為 84347.726643

從圖十七結果來看，版本二程式碼 area 比版本一程式碼 area(圖十二)少了將近 20000 cell area。

```
clock clk (rise edge)                                  15.000     15.000
clock network delay (ideal)                             0.500     15.500
clock uncertainty                                      -0.500     15.000
out5_reg[11]/CK (DFCRBN)                                0.000     15.000 r
library setup time                                     -0.116     14.884
data required time                                                14.884
-------------------------------------------------------------------------
data required time                                                14.884
data arrival time                                                -14.882
-------------------------------------------------------------------------
slack (MET)                                                       0.002
```

圖十八：time 結果，slack time 為 0.002

slack time 結果比版本一程式碼多了 0.001(圖十三)。

```
                 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         0.9460           0.1075         2.6525e+04            1.0535  (   9.77%)
sequential       0.0000           0.0000             0.0000            0.0000  (   0.00%)
combinational    3.8194           5.9088         2.8898e+05            9.7285  (  90.23%)
---------------------------------------------------------------------------------------------
Total            4.7653 mW        6.0163 mW      3.1550e+05 pW         10.7820 mW
```

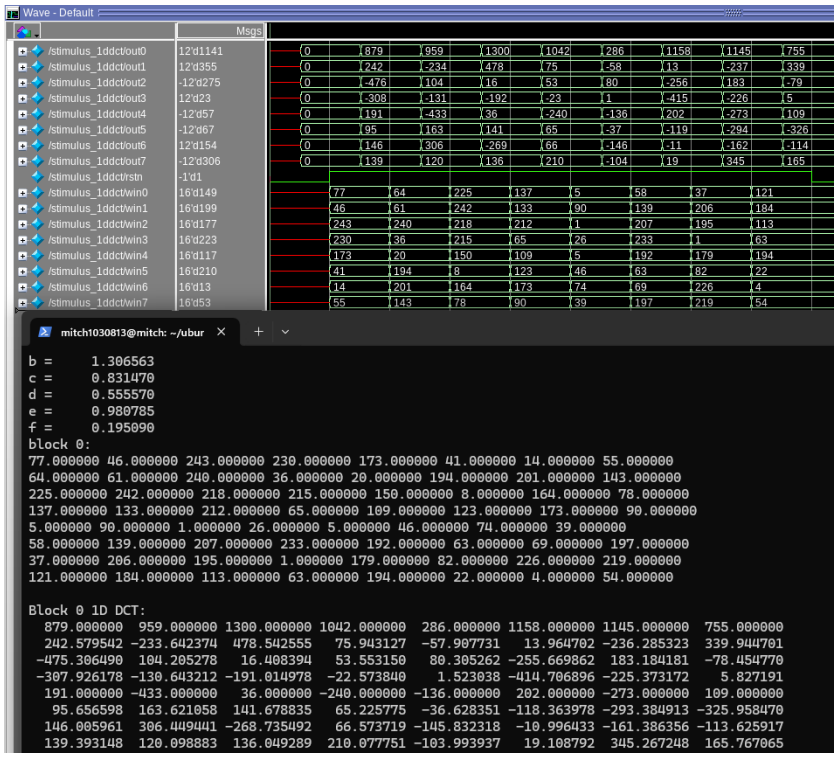
圖十九：power 結果，total power 預測為 10.7820mW

Power 結果比版本一程式碼少了約 2mW(圖十四)。
結論來說版本二合成後的效果較好，接下來更改 testbench 測試數據，然後生成 vcd 檔案給 primetime 計算 slack time 和 total power。
以下是使用 python 生成數據的程式碼：

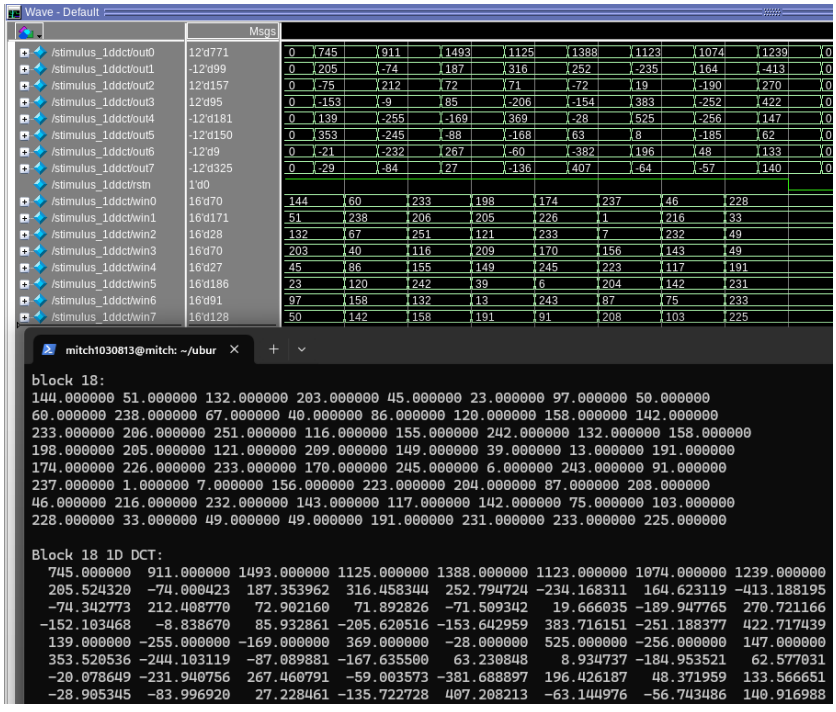
```python
import numpy as np

fp1 = open("hex1.txt", 'w')
fp2 = open("dec1.txt", 'w')
l = np.random.randint(0, 255, 6400)
for i in l:
    fp1.write(hex(i)[2:])
    fp1.write('\n')
    fp2.write('{}'.format(i))
    fp2.write('\n')
fp1.close()
fp2.close()
```

圖二十：生成 100 組 8*8 block 數據程式碼

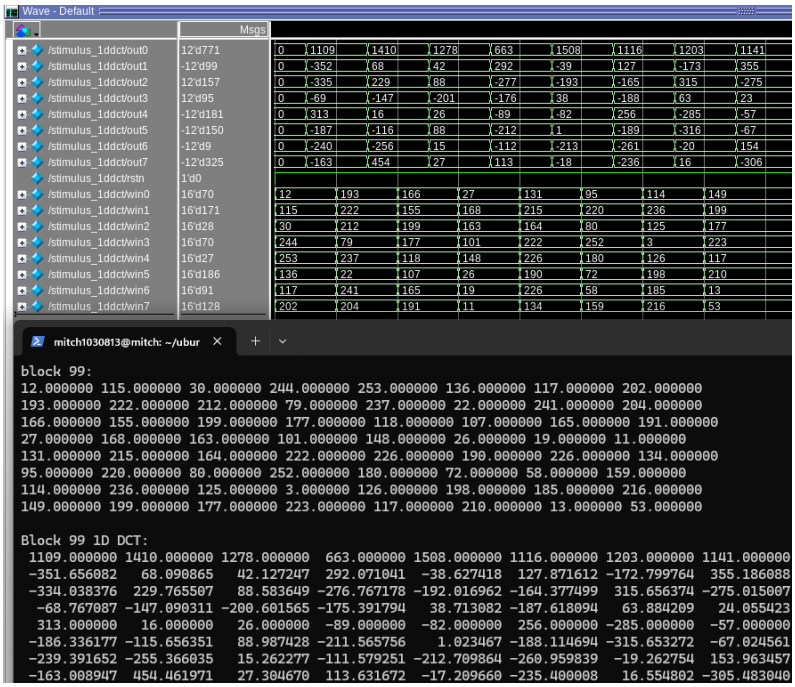
以上程式碼給定一個 0~255(unsigned char，影像像素資料型態)範圍並隨機生成 6400 筆數據，之後會以類似 8*8 block 的方式寫入 mem 中，會有 hex 和 dec 是因為 c 語言那邊也要檢測一下計算是否正確。
接著使用 vsim 加入 sdf 檔案進行時序仿真，這裡會生成 vcd 檔案並且驗證一下剛剛生成的數據計算 DCT 是否有誤。



圖二十一：block 0 DCT 計算結果比較。



圖二十二：block 18 DCT 計算結果比較



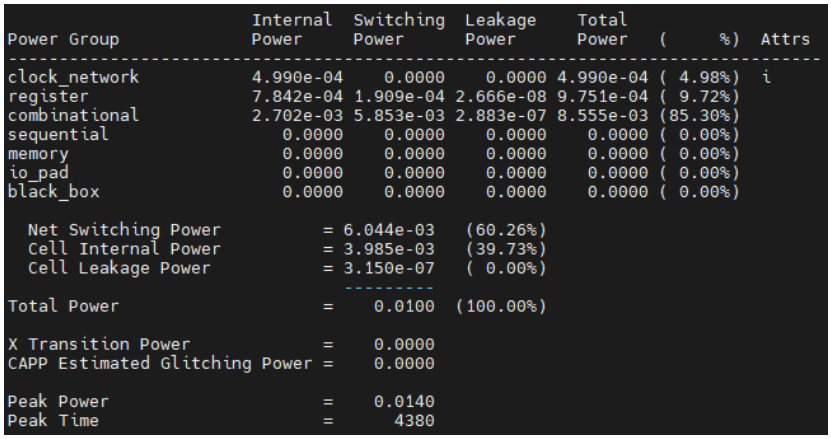
圖二十三：block 99 DCT 計算結果比較

從上面三個 block 看起來應該計算是沒有太大的問題，接下來可以用 primetime 來計算 power。

```
clock clk (rise edge)                          15.00       15.00
clock network delay (propagated)                0.00       15.00
clock reconvergence pessimism                   0.00       15.00
clock uncertainty                              -0.50       14.50
out5_reg[11]/CK (DFCRBN)                                   14.50 r
library setup time                             -0.12       14.39
data required time                                         14.39
-----------------------------------------------------------------
data required time                                         14.39
data arrival time                                         -14.40
-----------------------------------------------------------------
slack (VIOLATED)                                           -0.01
```

圖二十四：primetime 計算 slack time 為-0.01

slack time 來說以 100MHz clock 來跑會出問題。



Power Group	Internal Power	Switching Power	Leakage Power	Total Power	(%)	Attrs
clock_network	4.990e-04	0.0000	0.0000	4.990e-04	(4.98%)	i
register	7.842e-04	1.909e-04	2.666e-08	9.751e-04	(9.72%)	
combinational	2.702e-03	5.853e-03	2.883e-07	8.555e-03	(85.30%)	
sequential	0.0000	0.0000	0.0000	0.0000	(0.00%)	
memory	0.0000	0.0000	0.0000	0.0000	(0.00%)	
io_pad	0.0000	0.0000	0.0000	0.0000	(0.00%)	
black_box	0.0000	0.0000	0.0000	0.0000	(0.00%)	

Net Switching Power	= 6.044e-03	(60.26%)
Cell Internal Power	= 3.985e-03	(39.73%)
Cell Leakage Power	= 3.150e-07	(0.00%)
Total Power	= 0.0100	(100.00%)
X Transition Power	= 0.0000	
CAPP Estimated Glitching Power	= 0.0000	
Peak Power	= 0.0140	
Peak Time	= 4380	

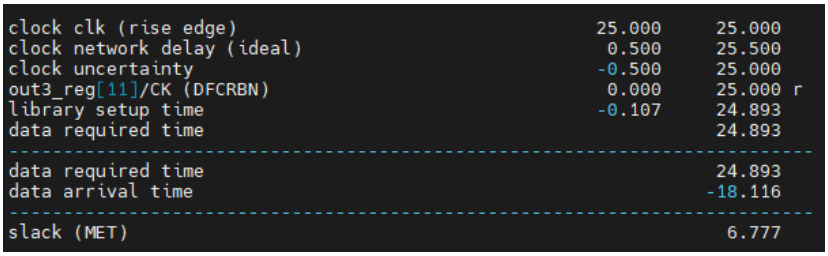
圖二十五：primetime 計算 total power 為 10mW

Primetime 計算的 power 和 design compiler 預測的 power(圖十九)幾乎一樣。

結論來說，版本二程式碼 area 和 power 都比版本一程式碼還要好，但是 slack time 還是一樣，兩個都很差。

三、 問題與解決方法

Slack time 目前是最大的問題，第一個問題可能是我不太會寫程式，改了兩天還改不好，但是目前時間緊迫，因此用第二個方法處理，就是將 clock 的切換頻率降低降到 50MHz 或是其他筆 100MHz 還要小的頻率。以下會將合成 clock 和 testbench clock 頻率降低至 50MHz，然後再看 design compiler time report 和 primetime time report。



clock clk (rise edge)	25.000	25.000
clock network delay (ideal)	0.500	25.500
clock uncertainty	-0.500	25.000
out3_reg[11]/CK (DFCRBN)	0.000	25.000 r
library setup time	-0.107	24.893
data required time		24.893
data required time		24.893
data arrival time		-18.116
slack (MET)		6.777

圖二十六：降低 clock 頻率，slack time 為 6.777。

可以看到降低 clock 頻率後，slcak time 變得比較正常了，不過由於改變頻率，power 和 area 也有跟著改變。

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	0.4901	5.3719e-02	2.5703e+04	0.5439	(10.63%)	
sequential	0.0000	0.0000	0.0000	0.0000	(0.00%)	
combinational	1.7392	2.8329	2.7766e+05	4.5724	(89.37%)	
Total	2.2293 mW	2.8866 mW	3.0336e+05 pW	5.1162 mW		

圖二十七：降低 clock 頻率，預估 total power 為 5.1162mW

```
Number of ports:                         2695
Number of nets:                          5145
Number of cells:                         1911
Number of combinational cells:           1768
Number of sequential cells:                96
Number of macros/black boxes:               0
Number of buf/inv:                        453
Number of references:                      59

Combinational area:              76035.758997
Buf/Inv area:                     4018.492735
Noncombinational area:            6299.596436
Macro/Black Box area:                0.000000
Net Interconnect area:      undefined  (Wire load has zero net area)

Total cell area:                 82335.355433
Total area:                 undefined
```

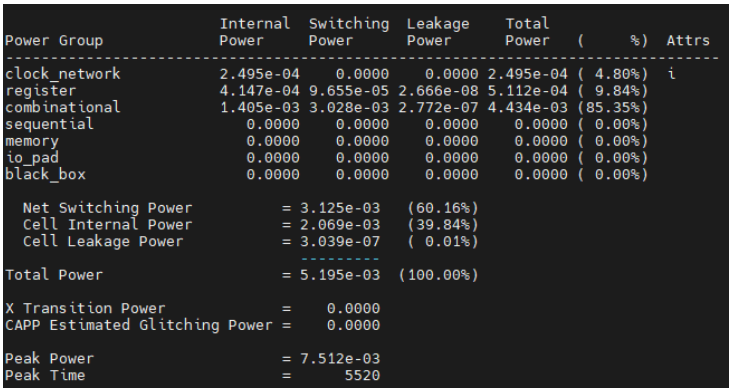
圖二十八：降低 clock 頻率，total cell area 為 82335.35433

跟之前 clock 100MHz 相比，power 和 cell area 都有再下降，power 預估少了一半(圖十九)，cell area 少了 2000 多(圖十七)，可知降低 clock 頻率可以把面積時間跟 power 都用的比較好，但是做的速度會比較慢。

最後再丟去 primetime 做分析。

```
clock clk (rise edge)                    25.00      25.00
clock network delay (propagated)          0.00      25.00
clock reconvergence pessimism             0.00      25.00
clock uncertainty                        -0.50      24.50
out3_reg[11]/CK (DFCRBN)                            24.50 r
library setup time                       -0.11      24.39
data required time                                  24.39
-----------------------------------------------------------
data required time                                  24.39
data arrival time                                  -17.63
-----------------------------------------------------------
slack (MET)                                         6.77
```

圖二十九：primetime 計算 slack time 為 6.77



```
                    Internal  Switching  Leakage    Total
Power Group         Power     Power      Power      Power   (     %)  Attrs
--------------------------------------------------------------------------
clock_network       2.495e-04    0.0000    0.0000 2.495e-04 ( 4.80%)  i
register            4.147e-04 9.655e-05 2.666e-08 5.112e-04 ( 9.84%)
combinational       1.405e-03 3.028e-03 2.772e-07 4.434e-03 (85.35%)
sequential             0.0000    0.0000    0.0000    0.0000 ( 0.00%)
memory                 0.0000    0.0000    0.0000    0.0000 ( 0.00%)
io_pad                 0.0000    0.0000    0.0000    0.0000 ( 0.00%)
black_box              0.0000    0.0000    0.0000    0.0000 ( 0.00%)

  Net Switching Power          = 3.125e-03   (60.16%)
  Cell Internal Power          = 2.069e-03   (39.84%)
  Cell Leakage Power           = 3.039e-07   ( 0.01%)
                                 ---------
Total Power                    = 5.195e-03  (100.00%)

X Transition Power             =    0.0000
CAPP Estimated Glitching Power =    0.0000

Peak Power                     = 7.512e-03
Peak Time                      =      5520
```

圖三十：primetime 計算 power 為 5.195mW

從圖二九和圖三十來看 power 和 time 都有變好，而且跟 design compiler 估計的數值差不多。

最後總結一下，降低 clock 頻率可以讓面積、power 和 slack time 變好，缺點我看了一下 vcd 檔案，總執行時間變長了。還有寫程式碼好像計算能寫再一起就寫一起，電腦好像比較會排上面那種寫法，雖然版本一可讀性比較好但是 area、power 還有 time 就比較爛(這裡有點小 bug，因為沒有重新跑過 50MHz 的 clock，但是後來跑了一下發現真的比較爛)。

四、 FINAL CODE

這段程式碼是在截止日期前一個小時半完成的，所以報告可能不完全，不過解決了版本一和版本二程式碼的問題，area、time 和 power 都有再進步，程式碼主要融合了版本一的 stage 計算方法和版本二的 blocking and non-blocking 寫法，還有將版本一的計算精度更改為乘上 2 的 10 次方，計算出來的差距為正負一。之前兩個版本的寫法是一個 clock 就要想辦法全部計算完成，現在改成需要幾個 clock 才可以把結果計算完成。總的來說可以在 clock 100MHz 下完成目標，只是會多出來大約 100ns 時間完成 100 筆 block 的計算目標。

程式碼：

```
always @(posedge clk) begin
    s10 <= win0 + win7;
    s17 <= win0 - win7;
    s11 <= win1 + win6;
    s16 <= win1 - win6;
    s12 <= win2 + win5;
    s15 <= win2 - win5;
    s13 <= win3 + win4;
    s14 <= win3 - win4;
end
```

圖三十一：stage 1 計算，non-blocking

```
assign tmp4 = (s17 * s3mc3);
assign tmp5 = (s14 * c3as3);
assign tmp7 = (s16 * s1mc1);
assign tmp8 = (s15 * c1as1);

always @(*) begin
    tmp02 = s15 + s16;
    tmp9 = (tmp02 * c1);
end

always @(*) begin
    tmp01 = s17 + s14;
    tmp6 = (tmp01 * c3);
end

always @(posedge clk) begin
    s20 <= s10 + s13;
    s23 <= s10 - s13;
    s21 <= s11 + s12;
    s22 <= s11 - s12;
    s24 <= (tmp4 + tmp6) >> 10;
    s27 <= (tmp6 - tmp5) >> 10;
    s25 <= (tmp7 + tmp9) >> 10;
    s26 <= (tmp9 - tmp8) >> 10;
end
```

圖三十二：stage 2 計算，non-blocking

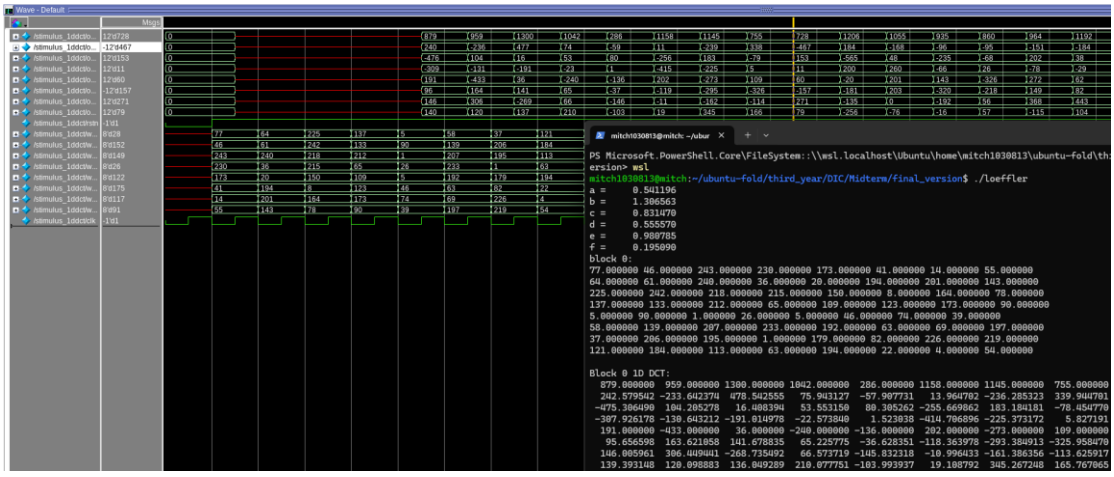
```
assign tmp1 = (s23 * sqs1mc1);
assign tmp2 = (s22 * sqc1as1);

always @(*) begin
    tmp03 = s22 + s23;
    tmp3 = (tmp03 * sqc1);
end

always @(posedge clk) begin
    s30 <= s20 + s21;
    s31 <= s20 - s21;
    s32 <= (tmp1 + tmp3) >> 10;
    s33 <= (tmp3 - tmp2) >> 10;
    s34 <= s24 + s26;
    s36 <= s24 - s26;
    s37 <= s27 + s25;
    s35 <= s27 - s25;
end
```

圖三十三：stage 3 計算，non-blocking

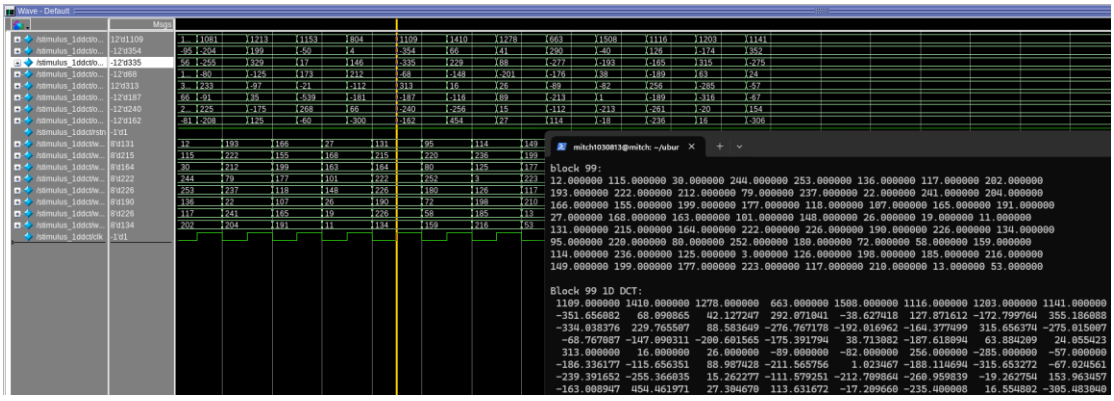
```
always @(posedge clk) begin
    o0 <= s30;
    o4 <= s31;
    o2 <= s32;
    o6 <= s33;
    o1 <= s37 + s34;
    o7 <= s37 - s34;
    o3 <= (s35 * sq2) >> 10;
    o5 <= (s36 * sq2) >> 10;
end
```

圖三十四：stage 4 計算，non-blocking

上方程式碼 always 中會同步好每條 reg 接收到資料，並且下個 stage 就會開始計算，直到下個 clock 到更新資料，上面會有看到 blocking 的寫法是因為在實作 kcn block 的時候會需要使用到前一個計算的 wire，所以需要有順序下去做，因此用 blocking，但是想了一下好像把 tmp 的計算等式直接合併到下面 tmp 的等式就好了。最後可以知道從一開始資料輸入到最後計算完成輸出中間會間隔 5 個 posedge clk，上面有四個 clock 再包含最後輸入到 output 的一個 clock，所以最後執行時間會是 8 * 100 * 10 + 50 = block 列數量 * block 數量 * posedge clk 週期 + 5 個 posedge clk 週期 = 8050ns。以下是展示一些計算結果，由於輸入和輸出沒有太同步，因此我只 verilog 計算出來的輸出，和 c 語言計算出來的輸出。



圖三十五：block 0 DCT 計算，黃線為 block 0 計算後範圍，vsim 介面上面是輸出 DCT，下面是輸入數據，終端機介面會顯示 C 語言計算對應的 DCT



圖三十六：block 99 DCT 計算，黃線為 block 99 計算前範圍，vsim 介面上面是輸出 DCT，下面是輸入數據，終端機介面會顯示 C 語言計算對應的 DCT

接著就來看一下 design compiler 合成 area、time 和 power。

```
clock clk (rise edge)                                  15.000     15.000
clock network delay (ideal)                             0.500     15.500
clock uncertainty                                      -0.500     15.000
s27_reg[11]/CK (QDFFN)                                  0.000     15.000 r
library setup time                                     -0.078     14.922
data required time                                                14.922
--------------------------------------------------------------------------
data required time                                                14.922
data arrival time                                                -13.598
--------------------------------------------------------------------------
slack (MET)                                                       1.324
```

圖三十五：clock 100MHz 下 slack time 為 1.324

```
****************************************
Report : area
Design : loeffler_1d
Version: R-2020.09
Date   : Mon Apr 15 22:17:38 2024
****************************************

Library(s) Used:

    fsa0m_a_generic_core_tt1p8v25c (File: /home/cad/tech/CBDK018_UMC_Faraday_v1.1/CIC/SynopsysDC/db/fsa0m_a_generic_core_tt1p8v25c.db)

Number of ports:                         1761
Number of nets:                          4369
Number of cells:                         2430
Number of combinational cells:           1933
Number of sequential cells:               461
Number of macros/black boxes:               0
Number of buf/inv:                       1094
Number of references:                      49

Combinational area:             51827.933029
Buf/Inv area:                    8690.068705
Noncombinational area:          25992.086594
Macro/Black Box area:               0.000000
Net Interconnect area:      undefined  (Wire load has zero net area)

Total cell area:                77820.019622
Total area:                 undefined
```

圖三十六：clock 100MHz 下 total cell area 為 77820.019622

```
Global Operating Voltage = 1.8
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  =   5.5137 mW   (62%)
  Net Switching Power  =   3.3100 mW   (38%)
                         ---------
Total Dynamic Power    =   8.8237 mW  (100%)

Cell Leakage Power     = 299.2046 nW


                 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           4.0311           1.5895          1.1726e+05             5.6208  (  63.70%)
sequential         0.0000           0.0000              0.0000             0.0000  (   0.00%)
combinational      1.4826           1.7205          1.8195e+05             3.2032  (  36.30%)
--------------------------------------------------------------------------------------------------
Total              5.5137 mW        3.3100 mW       2.9920e+05 pW          8.8240 mW
```

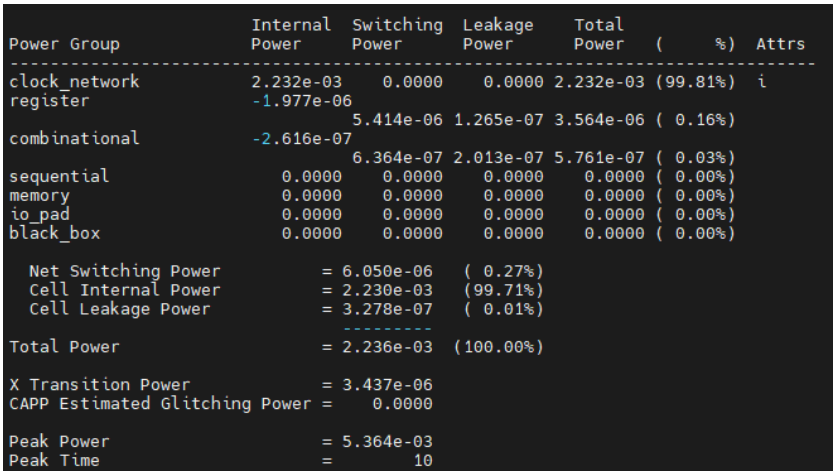
圖三十七：clock 100MHz 下 total power 預估為 8.8240mW

從結果來看，可知沒有一次做完 slack time、area，會比一個 clock 直接做完還要好，上面的結果跟版本二程式碼 clock 設定在 50MHz 相比較，area 少了約 5000 cell area，雖然 slack time 沒有很好看，但是至少不用花兩倍的時間去完成計算。

最後生成 vcd 檔案然後使用 primetime 來計算 time 和 power。

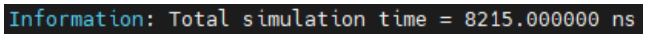
```
clock clk (rise edge)                          15.00      15.00
clock network delay (propagated)                0.00      15.00
clock reconvergence pessimism                   0.00      15.00
clock uncertainty                              -0.50      14.50
s27_reg[11]/CK (QDFFN)                                    14.50 r
library setup time                             -0.08      14.43
data required time                                        14.43
-----------------------------------------------------------------
data required time                                        14.43
data arrival time                                        -13.12
-----------------------------------------------------------------
slack (MET)                                               1.31
```

圖三十七：clock 100MHz 下 primetime 計算 slack time 為 1.31



```
                          Internal  Switching  Leakage    Total
Power Group               Power     Power      Power      Power   (     %)  Attrs
--------------------------------------------------------------------------------
clock_network             2.232e-03    0.0000    0.0000 2.232e-03 (99.81%)  i
register                  -1.977e-06
                                     5.414e-06 1.265e-07 3.564e-06 ( 0.16%)
combinational             -2.616e-07
                                     6.364e-07 2.013e-07 5.761e-07 ( 0.03%)
sequential                   0.0000    0.0000    0.0000    0.0000 ( 0.00%)
memory                       0.0000    0.0000    0.0000    0.0000 ( 0.00%)
io_pad                       0.0000    0.0000    0.0000    0.0000 ( 0.00%)
black_box                    0.0000    0.0000    0.0000    0.0000 ( 0.00%)

  Net Switching Power          = 6.050e-06   ( 0.27%)
  Cell Internal Power          = 2.230e-03   (99.71%)
  Cell Leakage Power           = 3.278e-07   ( 0.01%)
                                 ---------
Total Power                    = 2.236e-03  (100.00%)

X Transition Power             = 3.437e-06
CAPP Estimated Glitching Power =    0.0000

Peak Power                     = 5.364e-03
Peak Time                      =         10
```

圖三十八：clock 100MHz 下 primetime 計算 total power 為 2.236mW



```
Information: Total simulation time = 8215.000000 ns
```

圖三十九：VCD 檔案模擬時長

以結果來看，primetime 計算的 power 比 clock 設定為 50MHz 的還要少約 3mW(圖三十)，然後圖三十九的 vcd 檔案模擬時長也跟剛剛計算出來的模擬時長(8050ns)差不多，因為 testbench 檔案裡有設定一些#延時。總結上述的程式碼在計算精度上差距為正負一，area、time 和 power report 都比版本一和版本二的程式碼還要更好。