2010 University/College IC Design Contest

Cell-Based IC Design Category for Graduate Level

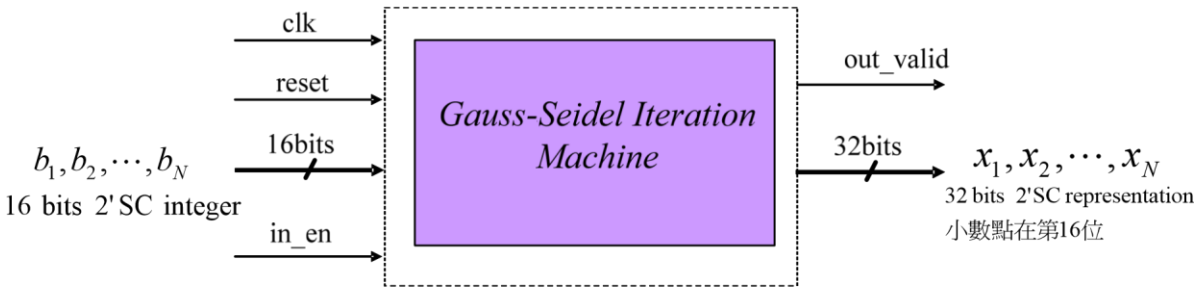
Gauss-Seidel Iteration Machine

1.問題描述

請完成一 Gauss-Seidel Iteration Machine(**後文以 GSIM 表示**)的電路設計來求出多元線性聯立方程式(Linear Equation)之解。線性方程式如(1)所示，矩陣 **A**、**B** 為已知之整數值，待求矩陣 **X** 之解。圖一為 GSIM 電路之方塊圖，其功能為：使用本電路將不同矩陣 **B** 的數值按照 $b_1, b_2, \ldots, b_{16}$ (本題 N 值固定為 16)的順序輸入，經過 GSIM 計算後求得矩陣 **X** 的值，再按照 $x_1, x_2, \ldots, x_{16}$ 的順序輸出計算後的結果，其中，矩陣 **B** 的數值為 16bits 整數值，取二的補數(2'S Complement，後文以 2'SC 表示)來表示，矩陣 **X** 的數值為 32bis，其整數與小數各佔 16bits，採 2'SC 表示。有關 GSIM 詳細規格將描述於後。

本電路各輸入輸出信號的功能說明，請參考表一。每個隊伍必須根據下一節所給的設計規格及附錄 A 中的測試樣本完成設計驗證。

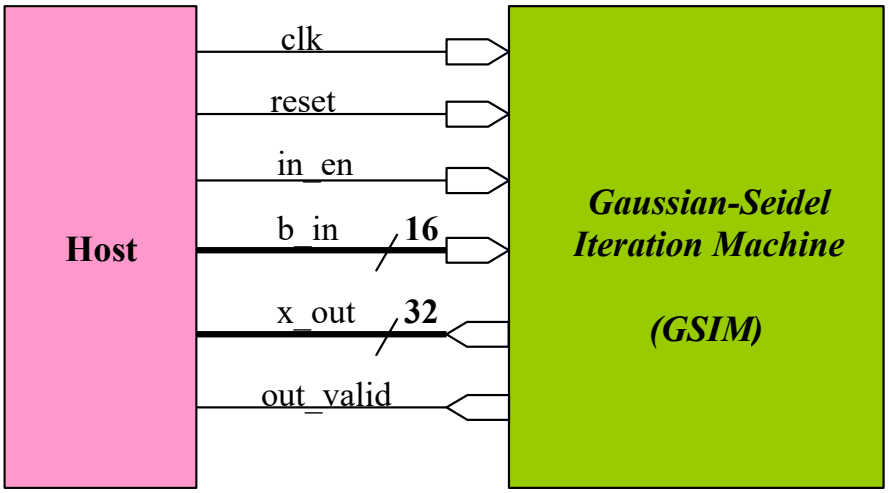
$$AX=B \longrightarrow \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \cdots & \cdots & \cdots & \cdots \\ a_{N1} & a_{N2} & \cdots & a_{NN} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_N \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \cdots \\ b_N \end{bmatrix} \qquad (1)$$



圖一、Gauss-Seidel Iteration Machine 之方塊圖

2.設計規格

2.1 系統方塊圖



圖二、系統方塊圖

2.2 輸入/輸出介面

表一 -輸入/輸出訊號

Signal Name	I/O	Width	Simple Description
clk	I	1	本系統為同步於時脈正緣之同步設計。 (註: Host 端採 clk 正緣時送資料。)
reset	I	1	高位準”非”同步(active high asynchronous)之系統重置信號。
in_en	I	1	資料輸入致能控制訊號。當 Host 端有資料要輸入時，該訊號就會一直維持在 High，直到 Host 端 16 筆資料送完後，該訊號才為 Low。
b_in	I	16	線性方程式(1)式中矩陣 **B** 資料輸入的匯流排。矩陣 **B** 的資料會按照 $b_1, b_2, \ldots, b_{16}$ 的順序輸入，輸入的資料為 16bits 整數，採 2’SC 表示。
x_out	O	32	線性方程式(1)式中矩陣 **X** 資料輸出的匯流排。矩陣 **X** 的資料會按照 $x_1, x_2, \ldots, x_{16}$ 的順序輸出，輸出的資料為 32bits，其整數、小數各佔 16bits，採 2’SC 表示。
out_valid	O	1	輸出資料有效之控制訊號。當為 High 時，表示目前輸出的資料為有效的；反之，當為 Low 時，表示目前輸出資料為無效的，即不被採用。

2.3 系統描述

2.3.1 解多元線性聯立方程式之基本概念

欲求出(1)式之多元線性聯立方程式之解，方法眾多，例如: 利用反矩陣公式、高斯(Gaussian)消去法、Jacobi Iteration、Gauss-Seidel Iteration 等方法求解，前兩種解法只適合小矩陣之計算，速度慢又不適合作硬體，後兩者則是採用疊代的方法，用疊代法不僅可以計算大型矩陣且非常適合作硬體，由於考量到 Gauss-Seidel Iteration 的收斂速度比 Jacobi Iteration 更快些，因此主辦單位今年題目限定，請參賽者只能使用 "Gauss-Seidel Iteration" 方法作成硬體，透過 GSIM 電路來計算多元線性聯立方程式之解。

2.3.2 Gauss-Seidel Iteration Machine 功能描述

將多元線性聯立方程式(1)式展開後，如(2)式所示。

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1N}x_N &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2N}x_N &= b_2 \\ &\vdots \\ a_{N1}x_1 + a_{N2}x_2 + \cdots + a_{NN}x_N &= b_N \end{aligned} \qquad (2)$$

欲求$x_1, x_2, \ldots, x_N$的值，可以將(2)式整理成(3)式，在(3)式中所謂x_2^1是指疊代第一次的 x_2 值，其值可以用等號右邊的式子求得。注意：等號右邊式子，一部分 x 是由該次疊代出之新的 x 值，另外一部分則是由前一次疊代出的 x 值，此乃 Gauss-Seidel Iteration 的特性。由於(3)式只是第一次疊代展開的式子，因此在此所謂的"前一次疊代的 x 值"就是初始值。例如，式中所謂的x_3^0是指x_3在疊代之前所設定的初始值，其餘類似的符號皆依此類推。

$$\begin{aligned} x_1^1 &= \frac{1}{a_{11}}(b_1 - a_{12}x_2^0 - \cdots - a_{1N}x_N^0) \\ x_2^1 &= \frac{1}{a_{22}}(b_2 - a_{21}x_1^1 - a_{23}x_3^0 - \cdots - a_{2N}x_N^0) \\ &\vdots \\ x_N^1 &= \frac{1}{a_{NN}}(b_N - a_{N1}x_1^1 - a_{N2}x_2^1 - \cdots - a_{NN-1}x_{N-1}^1) \end{aligned} \qquad (3)$$

事實上，Gauss-Seidel Iteration 就是將(3)式作相同的動作數次的疊代，其行為如(4)式所示，反覆地疊代數次後，即可將所有待求的 x 值收斂在某一個值，該 x 值即為所求，本題就是要參賽者將(4)式實現出來，製作出 GSIM 電路。

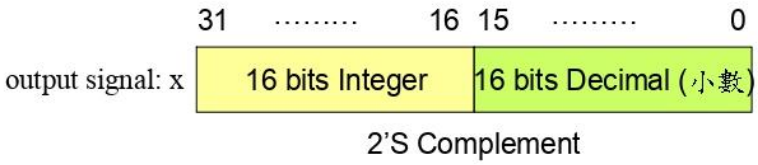
$$x_i^{k+1} = \frac{1}{a_{ii}}\left[b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{k+1} - \sum_{j=i+1}^{N} a_{ij} x_j^{k} \right] \qquad (4)$$

註：

1. 矩陣 X 之初值，由參賽者自行決定，例如：$X^0 = \begin{bmatrix} x_1^0 \\ x_2^0 \\ \vdots \\ x_N^0 \end{bmatrix} = \begin{bmatrix} b_1/a_{11} \\ b_2/a_{22} \\ \cdots \\ b_N/a_{NN} \end{bmatrix}$。

2. 輸入訊號 b 格式：

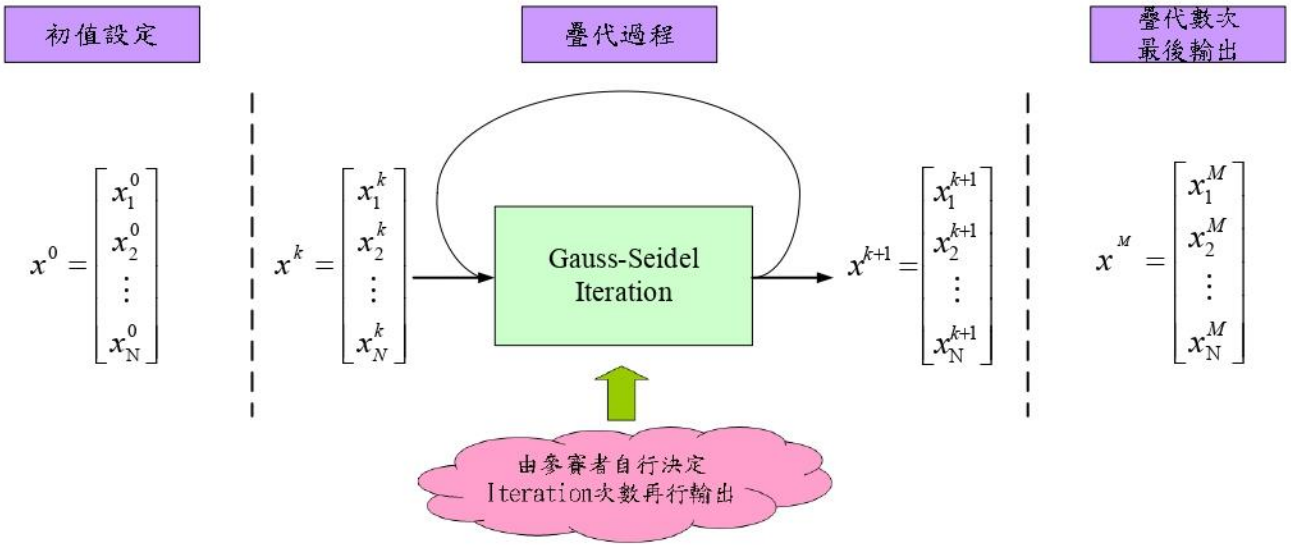
3. 輸出訊號 x 格式：



4. (4)式中的 a_{ij} 的值即為(1)式矩陣 A 的值，該矩陣 A 為固定的整數值，本題主辦單位已規定 N=16，因此矩陣 A 為 16x16 之矩陣，其值如下(5)式所示。

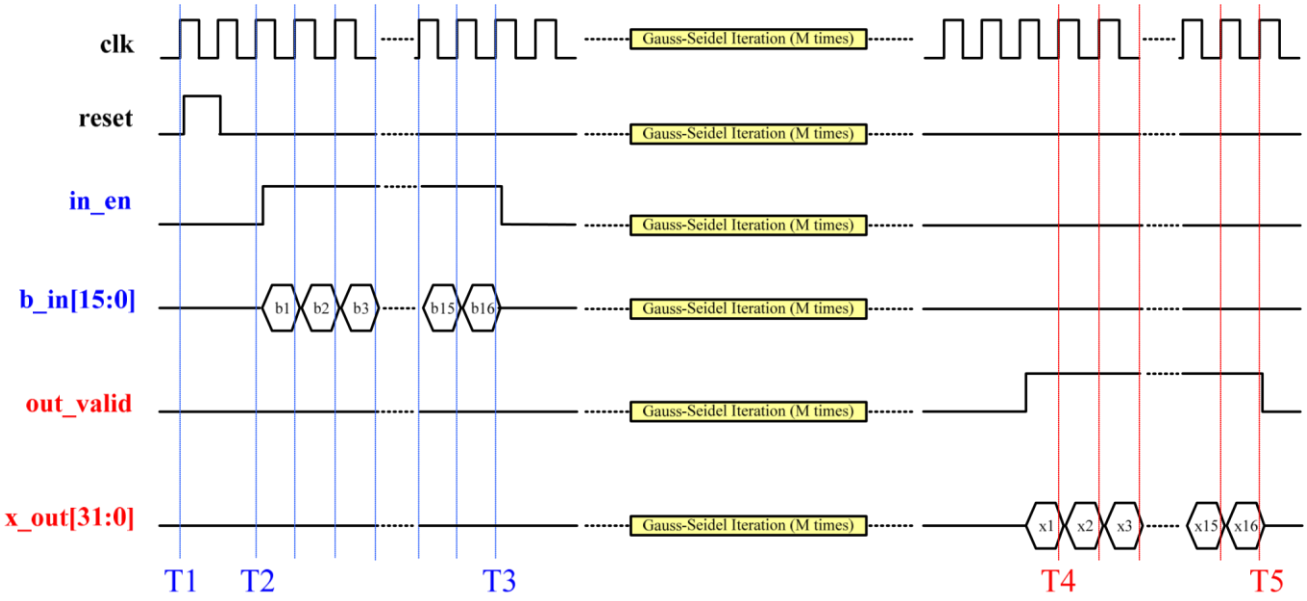
$$A = \begin{bmatrix}
20 & -13 & 6 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-13 & 20 & -13 & 6 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
6 & -13 & 20 & -13 & 6 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 6 & -13 & 20 & -13 & 6 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 6 & -13 & 20 & -13 & 6 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 6 & -13 & 20 & -13 & 6 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 6 & -13 & 20 & -13 & 6 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 6 & -13 & 20 & -13 & 6 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 6 & -13 & 20 & -13 & 6 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 6 & -13 & 20 & -13 & 6 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 6 & -13 & 20 & -13 & 6 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 6 & -13 & 20 & -13 & 6 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 6 & -13 & 20 & -13 & 6 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 6 & -13 & 20 & -13 & 6 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 6 & -13 & 20 & -13 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 6 & -13 & 20
\end{bmatrix} \qquad (5)$$

當參賽同學設定完初值後，便可開始用(4)式進行 Gauss-Seidel 疊代，整個疊代流程如圖三所示，最後疊代到該 x 值收斂或參賽者滿意的疊代次數(例如: 30 次)後再輸出。



圖三、GSIM 電路疊代流程

2.4 GSIM 電路時序規格



圖四、GSIM 電路時序圖

1. T1 時間點，GSIM 電路初始化，reset 一個 Cycle 的時間。

2. T2 時間點，Host 端開始送出第一筆訊號"b_1"，in_en 在送訊號期間都會維持在 High，直到 Host 端送完 16 筆訊號後(i.e. T3 時間點)，in_en 立即拉為 Low。Host 端沒有送訊號期間，b_in 會一直維持在高阻抗 (High Impedance)，in_en 維持在 Low。注意，Host 端送訊號至 GSIM 電路，其延遲時間固定為 1ns。

3. Host 端花了連續 16 個 Cycle 時間傳送訊號 $b_1, b_2, \ldots, b_{16}$ 後，隨即開始進行 Gauss-Seidel Iteration 運算，Iteration 次數由參賽者自行判斷與決定。

4. T4 時間點，Host 端開始接收 Gauss-Seidel Iteration 運算完之第一筆訊號"x_1"，此時將 out_valid 拉為 High 表示 x_out 輸出之值是有效的(Valid)，直到 T5 時間點，共接收 16 筆資料後，即完成整個模擬。請注意，當 x_out 沒有要作輸出時(i.e. out_valid 為 Low 時)，其值主辦單位並無限定，唯有當 out_valid 為 High 時，Host 端才會抓目前 x_out 的輸出值。

3.評分標準

RTL(40%) : RTL 模擬通過所有測資，且誤差值 $E^2 < 0.000001$ (Level A)。其中誤差值算法如下:

$$E^2 = \sum_{i=1}^{16}[\sum_{j=1}^{16} a_{ij}x_j^M - b_i]^2$$

Synthesis (40%): gate level 模擬通過所有測資，且 AT score $< 1.0 \times 10^{10}$。

AT score 算法為 : Total cell area ∗ Total execution time (tb1 + tb2 + … + tb5)

Combinational area:	3875.164193
Buf/Inv area:	434.534396
Noncombinational area:	1147.442383
Macro/Black Box area:	0.000000
Net Interconnect area:	48580.242432
Total cell area:	5022.606576
Total area:	53602.849008

```
----------------------------------------------------

Your Score Level: A

Congratulations! GSIM's Function Successfully!

------------------------PASS------------------------

Simulation complete via $finish(1) at time 3734500 PS + 0
./testfixture5.v:213        #(`CYCLE/2); $finish;
ncsim> exit
```

Ranking (15%): AT ranking with other competitions.

Note that there will be hidden testbenches. Failing to pass the hidden testbenches will only get 30% for both RTL and Synthesis. Furthermore, you will get 0% for Ranking if you fail passing the hidden testbenches.

Report (5%):

1. Gate-level simulation cycle time (ns)
2. Area (um^2)
3. AT score
4. Screenshot of inferred memory devices in process(※No latch should be inferred!)

4.繳交方式

Compress all the files into a single ZIP file and upload it to NTU COOL

File name: DSD_ Exercise _學號.zip

- ❖ EX: DSD_ Exercise _b09901001.zip

The submitted ZIP file should include the following files:

DSD_Exercise_學號/

GSIM.v

GSIM _syn.v

GSIM _syn.sdf

GSIM _syn.ddc

report.pdf

Deadline: 2024/05/01 23:59 ※Late submission is not allowed

附錄

附錄 A 為主辦單位所提供各參賽者的設計檔說明；附錄 B 為主辦單位提供的測試樣本說明。

附錄 A 設計檔(For Verilog or VHDL)

1. 下表為主辦單位所提供各參賽者的設計檔

表二、設計檔案說明

檔名	說明
GSIM.v	參賽者所使用的設計檔，已包含系統輸/出入埠之宣告
pattern1.dat pattern2.dat pattern3.dat pattern4.dat pattern5.dat	五組測試樣本，每組提供 16 個 Signal b 的值，分別為 $b_1, b_2, \ldots, b_{16}$，其格式為 16 進制採 2'SC 表示法。欲模擬 pattern1 的結果，請務必搭配 testfixture1.v 檔案；模擬 pattern2 的結果，請務必搭配 testfixture2.v 檔案，其餘的依此類堆。
testfixture1.v testfixture2.v testfixture3.v testfixture4.v testfixture5.v	五個 Test Bench 檔案。每個 Test Bench 已自動加入對應的 pattern 檔，主辦單位提供五個 Test Bench 檔案，主要是因為每個 Test Bench 都已將線性聯立方程式待求的 Signal $x_1, x_2, \ldots, x_{16}$ (i.e. Golden Pattern) 秀出，好讓參賽者可以比對自己最後的輸出結果與 Golden Pattern Signal x 之間的誤差值作比較與參考。Golden Pattern Signal x 是用 Floating-Point(i.e.無限精確度)計算出來的結果，請注意。
.synopsys_dc.setup	使用 Design Compiler 作合成之初始化設定檔。參賽者請依 Library 實際擺放位置，自行填上 Search Path 的設定。
GSIM_DC.sdc	使用 Design Compiler 作合成之 sdc 檔。參賽者可自行調整 cycle 值。

請使用 ***GSIM.v***，進行 GSIM 電路之設計。其模組名稱、輸出/入埠宣告如下所示：

```
module GSIM ( clk, reset, in_en, b_in, out_valid, x_out);
    input   clk ;
    input   reset ;
    input   in_en;
    output  out_valid;
    input   [15:0]  b_in;
    output  [31:0]  x_out;

endmodule
```

2. 主辦單位提供五個 Test Bench 檔案 testfixture1.v,testfixture2.v,…,testfixture5.v 分別對應到 pattern1, pattern2, …, pattern5，這五組 Pattern 都已加入了，參賽者只要注意 pattern 檔案的路徑即可。

例如：

第一個 Test Bench 模擬，使用 testfixture1.v：

`define PAT " ./pattern1.dat "

註：參賽者無須作修改，只需注意 pattern1.dat 的檔案位置即可，預設為目前目錄。

在每個 Test Bench 裡已提供用 Floating-Point(i.e.無限精確度)計算出的 Signal X 當作 Golden Pattern 給參賽者參考。例如，在 testfixture1.v 檔案裡可以看到如下之內容。其他的 Test Bench 檔案也有提供相對的 Golden Pattern 值，好讓參賽者作 Debug 時可以使用。

```
$display("------------------------------------------------------\n");
$display("        Your Output          Golden X\n");
$display("  X1:     %.10f          402.1120217501  \n", x_f[0 ]);
$display("  X2:     %.10f         1689.5336804421  \n", x_f[1 ]);
$display("  X3:     %.10f         2455.4774264763  \n", x_f[2 ]);
$display("  X4:     %.10f          563.1671481130  \n", x_f[3 ]);
$display("  X5:     %.10f          703.0136705772  \n", x_f[4 ]);
$display("  X6:     %.10f         1745.1919122734  \n", x_f[5 ]);
$display("  X7:     %.10f           33.2002351074  \n", x_f[6 ]);
$display("  X8:     %.10f          607.1379155812  \n", x_f[7 ]);
$display("  X9:     %.10f         -477.5895727426  \n", x_f[8 ]);
$display(" X10:     %.10f          869.0943789319  \n", x_f[9 ]);
$display(" X11:     %.10f         1907.5237775384  \n", x_f[10]);
$display(" X12:     %.10f         1524.3408800767  \n", x_f[11]);
$display(" X13:     %.10f          596.4154551345  \n", x_f[12]);
$display(" X14:     %.10f         1476.6345624265  \n", x_f[13]);
$display(" X15:     %.10f         1011.5707976786  \n", x_f[14]);
$display(" X16:     %.10f        -1330.8985774801  \n", x_f[15]);
$display("------------------------------------------------------\n");
$display("So Your Error Ratio=  %.15f\n", SquareError);
$display("------------------------------------------------------\n");
```

Golden Pattern Signal X

3. 主辦單位所提供的五個 Test Bench 檔案，都有多加上如下的敘述：

```
`define  SDFFILE  "./GSIM_syn.sdf"          // Modify your sdf file name
`ifdef SDF    initial
$sdf_annotate(`SDFFILE, GSIM); `endif
```

其目的是給合成或 APR 後可以順利進行模擬，請參賽者在模擬時，要記得多加一個參數"+define+SDF"，方可順利模擬。此外，SDF 檔名也要搭配合成與 APR 後實際名稱作修改。

例如：模擬合成後使用 VCS 模擬第一個 Test Bench，在 UNIX 下執行下面指令

```
> vcs  testfixture1.v  GSIM_syn.v -full64 -R -debug_access+all +v2k  –v  tsmc13_neg.v
+define+SDF
```

Note: To copy tsmc13_neg.v to your current directory:

```
> cp /home/raid7_2/course/cvsd/CBDK_IC_Contest/CIC/Verilog/tsmc13_neg.v .
```

附錄 B 測試樣本

為了讓參賽者看完題目後，更能確定題意，在此主辦單位以第一個測試樣本為例(注意：下面所列之範例是用 Floating-Point 來計算的)，秀出第一次疊代結果，以下純作參考使用。注意：疊代次數請自行決定，主辦單位沒有限定。

表 3、使用測試樣本 1 執行 Gauss-Seidel 疊代之範例

Gauss-Seidel Iteration Machine (example for pattern1)											
Input Signal B			X初值		疊代第一次		疊代第二次		……	疊代到第M次後輸出	
	10進制	16進制		10進制		10進制		10進制			10進制
b1	248	00F8	x1	12.4	x1	-404.475	x1	……		x1	……
b2	-682	FD56	x2	-34.1	x2	642.14375	x2	……		x2	……
b3	24710	6086	x3	1235.5	x3	1587.0934375	x3	……		x3	……
b4	-9624	DA68	x4	-481.2	x4	-300.7736406	x4	……		x4	……
b5	-3313	F30F	x5	-165.65	x5	709.5062898	x5	……		x5	……
b6	30377	76A9	x6	1518.85	x6	658.9008525	x6	……		x6	……
b7	-29996	8AD4	x7	-1499.8	x7	40.83248512	x7	……		x7	……
b8	30995	7913	x8	1549.75	x8	602.0186741	x8	……	……	x8	……
b9	-20368	B070	x9	-1018.4	x9	-306.7375648	x9	……		x9	……
b10	10952	2AC8	x10	547.6	x10	158.8591049	x10	……		x10	……
b11	5665	1621	x11	283.25	x11	1037.925621	x11	……		x11	……
b12	11476	2CD4	x12	573.8	x12	822.1945442	x12	……		x12	……
b13	-9108	DC6C	x13	-455.4	x13	-352.8707775	x13	……		x13	……
b14	7882	1ECA	x14	394.1	x14	1105.254412	x14	……		x14	……
b15	20391	4FA7	x15	1019.55	x15	861.0238286	x15	……		x15	……
b16	-31505	84EF	x16	-1575.25	x16	-1364.804374	x16	……		x16	……