**國立清華大學**

**超大型積體電路測試**

**VLSI Testing**

一張含有 文字, 筆跡, 字型, 圓形 的圖片

自動產生的描述

**Homework 3**

**系所級:電子所一年級、電機系碩士班一年級**

**學號:111063548、111061624**

**姓名:蕭方凱、尤弘瑋**

**指導老師:黃錫瑜教授**

目錄

[分工 4](#_Toc136623399)

[(a) Write the Verilog model for this SRAM and verify it with a simple testbench to show the functional correctness. 4](#_Toc136623400)

[RTL 5](#_Toc136623401)

[Testbench 7](#_Toc136623402)

[nWave 10](#_Toc136623403)

[(b) Write a synthesizable Verilog code to realize the Built-In Self-Test (BIST) function for this SRAM block. Include the checkerboard test and the march test as discussed in class in your test algorithm. Replace the testbench in (a) with this BIST circuit and conduct the Verilog simulation again. 11](#_Toc136623404)

[March Test 11](#_Toc136623405)

[i. MARCH.v 11](#_Toc136623406)

[ii. FSM of March.v 13](#_Toc136623407)

[iii. MUX.v 14](#_Toc136623408)

[iv. Delay.v 14](#_Toc136623409)

[v. Comparator.v 14](#_Toc136623410)

[vi. BIST\_MARCH test 15](#_Toc136623411)

[vii. nWave 16](#_Toc136623412)

[Checkerboard Test 22](#_Toc136623413)

[i. Checkerboard.v 22](#_Toc136623414)

[ii. FSM of Checkerboard 24](#_Toc136623415)

[iii. MUX.v 25](#_Toc136623416)

[iv. Delay.v 25](#_Toc136623417)

[v. Comparator.v 25](#_Toc136623418)

[vi. BIST\_Checkerboard Module 26](#_Toc136623419)

[vii. nWave 27](#_Toc136623420)

[(c) Synthesize your BIST circuit using Design Compiler. Report the critical path delay of your BIST circuit. 30](#_Toc136623421)

[Critical path 30](#_Toc136623422)

[BIST\_MARCH TEST Synthesize report 30](#_Toc136623423)

[BIST\_CHECKERBOARD TEST Synthesize report 31](#_Toc136623424)

[(d) Perform gate-level simulation including your BIST circuit and the Verilog model for the SRAM block under test. Assume a clock rate of 100MHz. Does you BIST circuit report a test result of “pass”? If not, discuss why. 32](#_Toc136623425)

[BIST\_MARCH\_TEST Timing report (Clock rate = 100MHz ) 32](#_Toc136623426)

[BIST\_MARCH\_TEST nWave result 32](#_Toc136623427)

[BIST\_CHECKERBOARD\_TEST Timing report (Clock rate = 100MHz ) 33](#_Toc136623428)

[BIST\_CHECKERBOARD\_TEST nWave result 33](#_Toc136623429)

[(e) Increase the clock rate gradually (i.e., from 100MHz to some higher rates) and check if at some point the test result becomes erroneous (i.e., turning from “pass” to “fail”). Discuss if you can derive the maximum operating speed of your BIST circuit in this experiment. 34](#_Toc136623430)

[Test Result 34](#_Toc136623431)

[BIST\_march 34](#_Toc136623432)

[(f) Try to inject a stuck-at-1 fault to bit #2 of the SRAM word #5, and report the test result of your BIST at 100MHz 36](#_Toc136623433)

[SRAM stuck-at-1 fault 36](#_Toc136623434)

[BIST\_March Test 36](#_Toc136623435)

[BIST\_Checkerboard Test 36](#_Toc136623436)

## 分工

**蕭方凱:**

**Algorithm、RTL(SRAM and BIST March Test)、Testbench、Report**

**尤弘瑋:**

**Algorithm、RTL(BIST Checkerboard Test、s-a-1 SRAM)、Synthesis、Report**

# Write the Verilog model for this SRAM and verify it with a simple testbench to show the functional correctness.

一張含有 文字, 螢幕擷取畫面, 數字, 圖表 的圖片

自動產生的描述

依照題目的功能，我們設計的SRAM總共有16個address，以下為其運作模式:

1. we\_n=0且cs\_n=0時寫入data\_in。
2. we\_n=1且cs\_n=0時讀出data\_out。
3. cs\_n=1時，SRAM不動作。

## RTL

一張含有 文字, 螢幕擷取畫面 的圖片

自動產生的描述

Fig 1 RTL code (SRAM.v)

一張含有 文字, 螢幕擷取畫面, 數字 的圖片

自動產生的描述

Fig 2 RTL code (SRAM.v)

一張含有 文字, 螢幕擷取畫面, 數字, 軟體 的圖片

自動產生的描述

一張含有 文字, 字型, 行, 螢幕擷取畫面 的圖片

自動產生的描述

Fig 3 RTL code (SRAM.v)

RTL架構說明:

1. 將data\_in分為兩個bits，高位元為data1，低位元為data0
2. 宣告一個二維的register mem再依據address的值來存data0跟data1的值
3. 最後再根據不同的address把存在mem裡的值寫入到data\_out。

## Testbench

**(Write Mode) → we\_n = 0**

一張含有 鍵盤, 輸入裝置, 螢幕擷取畫面, 周邊設備 的圖片

自動產生的描述一張含有 螢幕擷取畫面, 文字, 設計 的圖片

自動產生的描述

此為第一筆pattern的寫入模式，由於是第一筆，一定會寫入address 0-15的data in，填滿SRAM。

右下方的格子即為目前SRAM填入的狀態，**從左上依序填入address 0-15到右下**，以PATTERN 1為例，address = 0的data in為0; address = 1的data in為3，address = 2的data in為3，以此類推。

**(Write Mode) → we\_n = 0**

一張含有 文字, 螢幕擷取畫面, 軟體, 設計 的圖片

自動產生的描述

此為第二筆pattern的寫入模式，在PATTERN 2中，SRAM只被更新address 6、9、11、13、14、15。

觀察SRAM Word Space，以address 0-3為例，因沒有被更新數值，故其值仍為PATTERN 1的數值。

**(Read Mode) → we\_n = 1**

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

此為PATTERN 2的讀出模式，address 2的data out為3; address 15的data out為1;address 4的data out為1，**與上方SRAM Word Space填入的狀態相符。**

一張含有 文字, 螢幕擷取畫面 的圖片

自動產生的描述

PATTERN 3在Write Mode時**遇到cs\_n=1的情況**，即便有指定address，也不會進行寫入動作，**可以從Word Space觀察，目前的SRAM狀態與PATTERN 2時相同。**

而在Read Mode下，分別讀出address 4及11的值，**與上方SRAM Word Space填入的狀態相符。**

## 一張含有 螢幕擷取畫面, 電子產品, 行, 鮮豔 的圖片 自動產生的描述nWave

Pattern 1 Write Mode

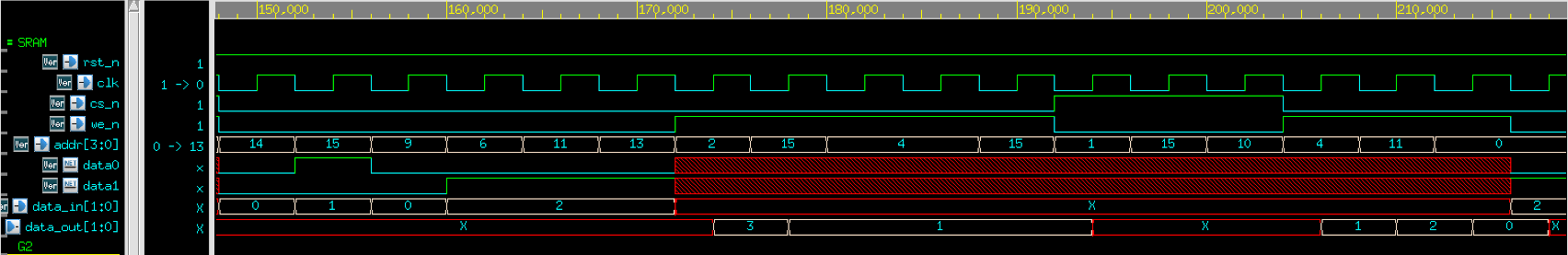
Pattern 1 Read Mode

1. Pattern 1 Write Mode:

寫入分別對應address 0-15的data\_in。

1. Pattern 1 Read Mode:

因cs\_n為1，故不做讀取動作。



Pattern 2 Read Mode

Pattern 2 Write Mode

1. Pattern 2 Write Mode:

指定address 14、15、9、6、11、13並寫入data\_in，其值分別為0、1、0、2、2、2。

1. Pattern 2 Read Mode:

指定address 2、15、4、15讀取SRAM，因為RTL是用時序邏輯描述讀取動作，故波形都會晚一個clock。

# Write a synthesizable Verilog code to realize the Built-In Self-Test (BIST) function for this SRAM block. Include the checkerboard test and the march test as discussed in class in your test algorithm. Replace the testbench in (a) with this BIST circuit and conduct the Verilog simulation again.

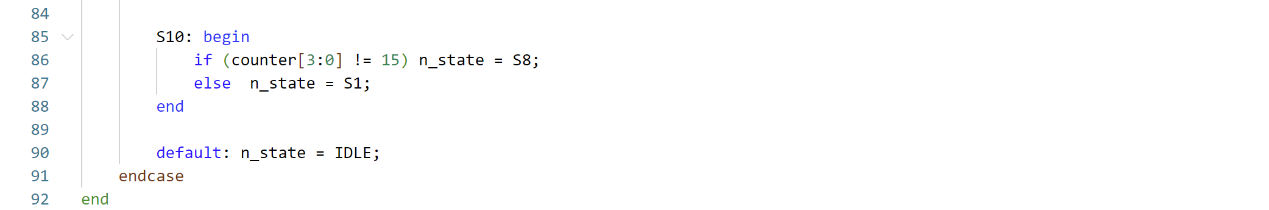
## March Test

### MARCH.v

一張含有 文字, 螢幕擷取畫面, 數字 的圖片

自動產生的描述

Fig 4 RTL code (March.v)

一張含有 文字, 螢幕擷取畫面 的圖片

自動產生的描述

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

Fig 5 RTL code (March.v)

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

Fig 6 RTL code (March.v)

### FSM of March.v

MARCH的FSM架構參考了講義的內容，但由於我們設計的SRAM是4\*8，所以總共設了12個state，IDLE跟FINISH分別代表開始跟結束，S1代表一開始要把SRAM的值都設為0，接下來的S2~S4就是要將SRAB bit0的部分分別做讀0、寫1、讀1，此時bit0部分就全為1；S5~S7就是要將bit1的部分再做讀0、寫1、讀1，此時bit0、bit1就全為1；S8~S10就是再將bit0的部分再做讀1、寫0、讀0，此時bit0就會變回0；最後回到S1將bit0、bit1變回0，這樣就完成了一輪的BIST MARCH TEST。

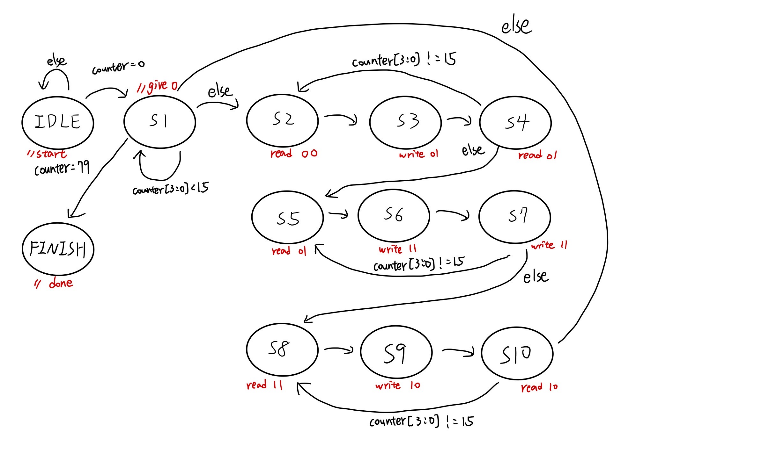


Fig 7 FSM of March

一張含有 圖表, 文字, 正方形, Rectangle 的圖片

自動產生的描述

Fig 8 March Pattern變化順序

### MUX.v

一張含有 文字, 字型, 螢幕擷取畫面 的圖片

自動產生的描述

Fig 9 RTL code (MUX.v)

MUX有4個，分別對data、address、cs\_n、we\_n進行比較，可參考Fig8 March架構。

### Delay.v

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

Fig 10 RTL code (Delay.v)

Delay是以buffer形式存在於BIST module內，delay訊號addr、data\_in、we\_n。

### Comparator.v

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

Fig 11 RTL code (Comparator.v)

Comparator對Pattern Generator的data\_out及SRAM的data\_out進行比較，若兩者相等，則使output Equal\_or\_not為1;否則為0。

### BIST\_MARCH test

在MARCH test的設計中我們參考了講義上的架構，利用已設計好的MARCH.v，把cs\_n、we\_n、data、addr的值都存到output buffer裡，再利用4個MUX去做選取。當判斷tester=1時就利用(a)小題testbench的normal\_input跟normal\_addr給入SRAM module；當tester=0時就利用pattern generator產生的資料給入SRAM。

若tester=0時，因為pattern generator產生的data\_out要經過中間的邏輯閘跟SRAM module才會得到SRAM\_out，所以data\_out要先經過一個delay buffer才能輸入到comparator跟SRAM\_out做比較，否則就會有時序問題。

當comparator的兩個input值為相同，結果就為pass，若不同就為fail。當把tester=1或0的條件都測完之後，test\_done則會拉起為1。

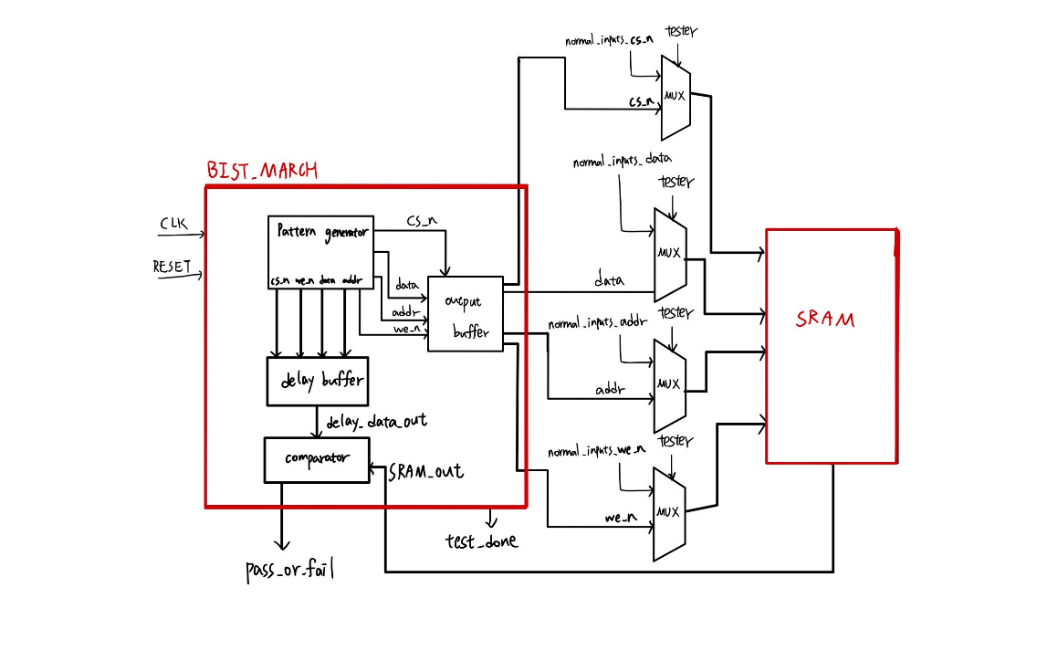


Fig 12 BIST\_March架構

### nWave

一張含有 文字, 螢幕擷取畫面, 軟體, 陳列 的圖片

自動產生的描述

1. **BIST:**

上圖波形tester為1，表示現在模式是讓SRAM接收Normal Inputs，即為(a)小題的testbench。

1. **March Pattern Generator:**

march\_addr在每個位址(0,1,2…)皆與we\_n = 1→0→1搭配，並將data\_out = 依照不同狀態輸出，因現在是tester = 1，SRAM不會接收March Pattern的訊號，這邊先不詳細探討。

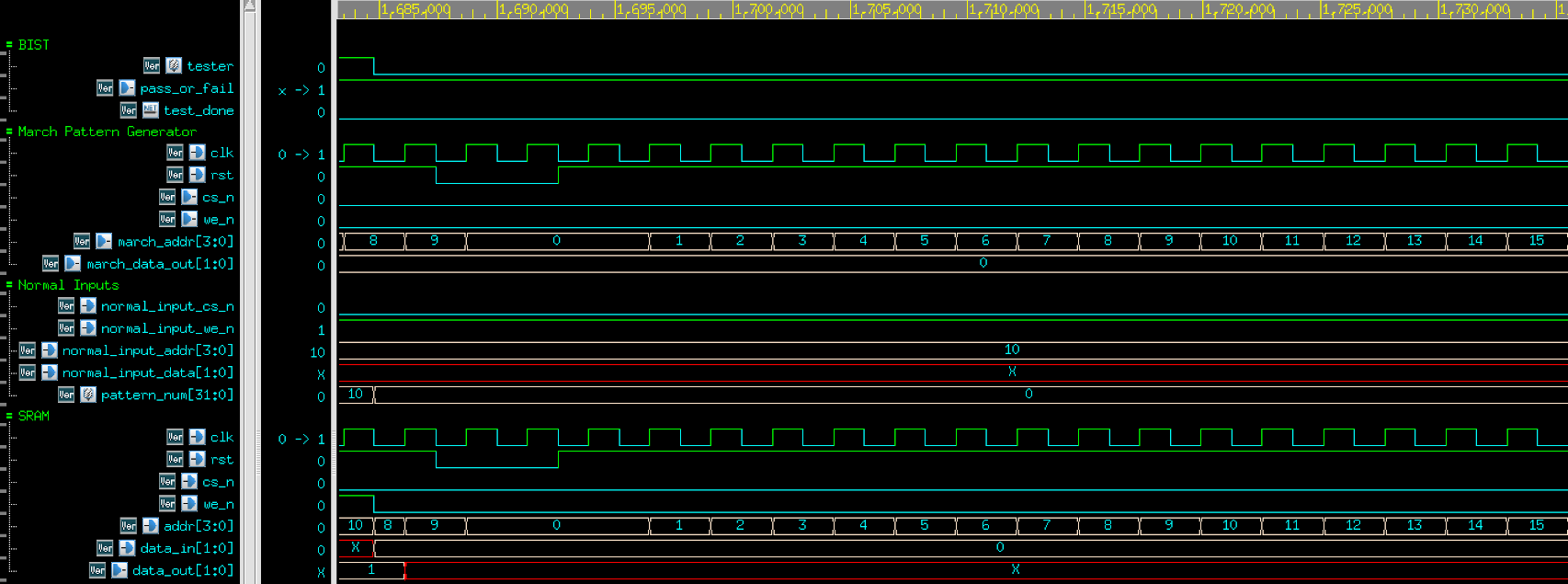
**因現在是tester = 1的模式，rst是由Normal Inputs決定，故不會等待march pattern generator完成所有動作。**

1. **Normal Inputs:**

此為由(a)小題的testbench所產生之we\_n, cs\_n, data\_in, address。

1. **SRAM**

因tester為1，故現在為Normal Inputs模式，SRAM接收的訊號來自Normal Inputs。



1. **BIST:**

上圖波形tester為0，表示現在模式是讓SRAM接收BIST電路輸出的訊號，即為March Pattern Generator。

Pass\_or\_fail即為Comparator在we\_n=1時進行比較的結果，pass\_or\_fail = 1代表SRAM data\_out與March Pattern Generator值相等，反之則為0。

1. **March Pattern Generator:**

輸出we\_n = 0 (write mode)，並指定march\_addr從0-15將data\_out = 0輸出，如下圖示意圖:

一張含有 文字, 螢幕擷取畫面, 數字, 正方形 的圖片

自動產生的描述

1. **Normal Inputs:**

此為由(a)小題的testbench所產生之we\_n, cs\_n, data\_in, address，因現在是tester = 0，故不會有Normal Inputs輸入進SRAM。

1. **SRAM**

因tester為0，故現在為BIST test模式，SRAM接收的訊號來自March Pattern Generator。

一張含有 螢幕擷取畫面, 文字, 電子產品, 電路 的圖片

自動產生的描述

1. **BIST:**

上圖波形tester為0，表示現在模式是讓SRAM接收BIST電路輸出的訊號，即為March Pattern Generator。

Pass\_or\_fail即為Comparator在we\_n=1時進行比較的結果，pass\_or\_fail = 1代表SRAM data\_out與March Pattern Generator值相等，反之則為0。

1. **March Pattern Generator:**

march\_addr在每個位址(0,1,2…)皆與we\_n = 1→0→1搭配，並將data\_out = 0→1→1依序輸出，直到march\_addr為15，如下圖示意圖:

一張含有 文字, 螢幕擷取畫面, 數字, 字型 的圖片

自動產生的描述

1. **Normal Inputs:**

此為由(a)小題的testbench所產生之we\_n, cs\_n, data\_in, address，因現在是tester = 0，故不會有Normal Inputs輸入進SRAM。

1. **SRAM**

因tester為0，故現在為BIST test模式，SRAM接收的訊號來自March Pattern Generator。

因March Pattern Generator輸出的we\_n為1→0→1，故SRAM的data\_out會有unknown狀態。

一張含有 螢幕擷取畫面, 電子產品, 電路, 電腦 的圖片

自動產生的描述

1. **BIST:**

上圖波形tester為0，表示現在模式是讓SRAM接收BIST電路輸出的訊號，即為March Pattern Generator。

Pass\_or\_fail即為Comparator在we\_n=1時進行比較的結果，pass\_or\_fail = 1代表SRAM data\_out與March Pattern Generator值相等，反之則為0。

1. **March Pattern Generator:**
2. march\_addr在每個位址(0,1,2…)皆與we\_n = 1→0→1搭配，並將data\_out = 1→3→3依序輸出，直到march\_addr為15，如下圖示意圖:

一張含有 文字, 數字, Rectangle, 螢幕擷取畫面 的圖片

自動產生的描述

1. **Normal Inputs:**

此為由(a)小題的testbench所產生之we\_n, cs\_n, data\_in, address，因現在是tester = 0，故不會有Normal Inputs輸入進SRAM。

1. **SRAM**

因tester為0，故現在為BIST test模式，SRAM接收的訊號來自March Pattern Generator。

因March Pattern Generator輸出的we\_n為1→0→1，故SRAM的data\_out會有unknown狀態。

一張含有 螢幕擷取畫面, 文字, 行, 電子產品 的圖片

自動產生的描述

1. **BIST:**

上圖波形tester為0，表示現在模式是讓SRAM接收BIST電路輸出的訊號，即為March Pattern Generator。

Pass\_or\_fail即為Comparator在we\_n=1時進行比較的結果，pass\_or\_fail = 1代表SRAM data\_out與March Pattern Generator值相等，反之則為0。

1. **March Pattern Generator:**

march\_addr在每個位址(0,1,2…)皆與we\_n = 1→0→1搭配，並將data\_out = 3→2→2依序輸出，**但這時的march\_addr是從15-0**，也就是說現在是由左下往右上依序填入數值，如下圖示意圖:

一張含有 文字, 螢幕擷取畫面, 數字, 字型 的圖片

自動產生的描述

1. **Normal Inputs:**

此為由(a)小題的testbench所產生之we\_n, cs\_n, data\_in, address，因現在是tester = 0，故不會有Normal Inputs輸入進SRAM。

1. **SRAM**

因tester為0，故現在為BIST test模式，SRAM接收的訊號來自March Pattern Generator。

因March Pattern Generator輸出的we\_n為1→0→1，故SRAM的data\_out會有unknown狀態。

一張含有 螢幕擷取畫面, 文字, 軟體, 多媒體軟體 的圖片

自動產生的描述

1. **BIST:**

上圖波形tester為0，表示現在模式是讓SRAM接收BIST電路輸出的訊號，即為March Pattern Generator。

Pass\_or\_fail即為Comparator在we\_n=1時進行比較的結果，pass\_or\_fail = 1代表SRAM data\_out與March Pattern Generator值相等，反之則為0。

1. **March Pattern Generator:**

march\_addr在每個位址(0,1,2…)皆與we\_n = 1→0→1搭配，並將data\_out = 2→0→0依序輸出，**但這時的march\_addr是從15-0**，也就是說現在是由左下往右上依序填入數值，如下圖示意圖:

一張含有 文字, 螢幕擷取畫面, 數字, 正方形 的圖片

自動產生的描述

1. **Normal Inputs:**

此為由(a)小題的testbench所產生之we\_n, cs\_n, data\_in, address，因現在是tester = 0，故不會有Normal Inputs輸入進SRAM。

1. **SRAM**

因tester為0，故現在為BIST test模式，SRAM接收的訊號來自March Pattern Generator。

因March Pattern Generator輸出的we\_n為1→0→1，故SRAM的data\_out會有unknown狀態。

## Checkerboard Test

### Checkerboard.v

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

一張含有 文字, 字型, 螢幕擷取畫面 的圖片

自動產生的描述

Fig 13 RTL Code (Checkerboard.v)

一張含有 文字, 螢幕擷取畫面, 字型, 文件 的圖片

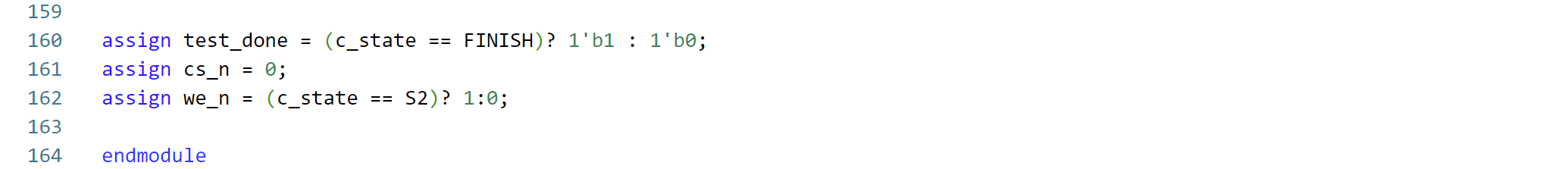
自動產生的描述

一張含有 文字, 螢幕擷取畫面, 數字 的圖片

自動產生的描述

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述



### FSM of Checkerboard

Checkerboard 的FSM架構也是參考講義內容，將4\*8的SRAM bit0跟bit1的相對address的位置同時寫值再讀入，所以在S1狀態時就會把bit0 跟bit1相對應addr[0]填入00，addr[1]填入11，addr[2]填入00….以此類推；由於要將每個address都填入0跟1，就必須要有跟S1相反的狀態；在S2狀態時就會跟S1相反，會把bit0 跟bit1相對應addr[0]填入11，addr[1]填入00，addr[2]填入11….以此類推，這樣一輪過後就完成了BIST checkerboard test。



Fig 14 FSM of Checkerboard

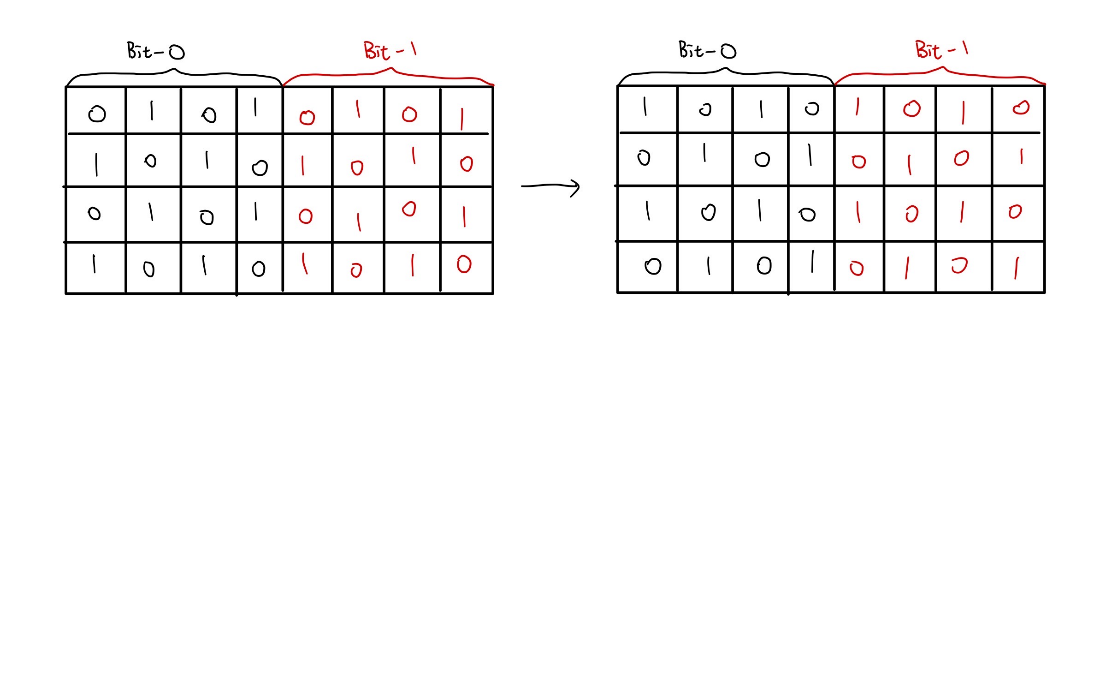


Fig 15 Checkerboard變化示意圖

### MUX.v

一張含有 文字, 字型, 螢幕擷取畫面 的圖片

自動產生的描述

Fig 16 RTL code (MUX.v)

MUX有4個，分別對data、address、cs\_n、we\_n進行比較，可參考Fig8 March架構。

### Delay.v

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

Fig 17 RTL code (Delay.v)

Delay是以buffer形式存在於BIST module內，delay訊號addr、data\_in、we\_n。

### Comparator.v

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

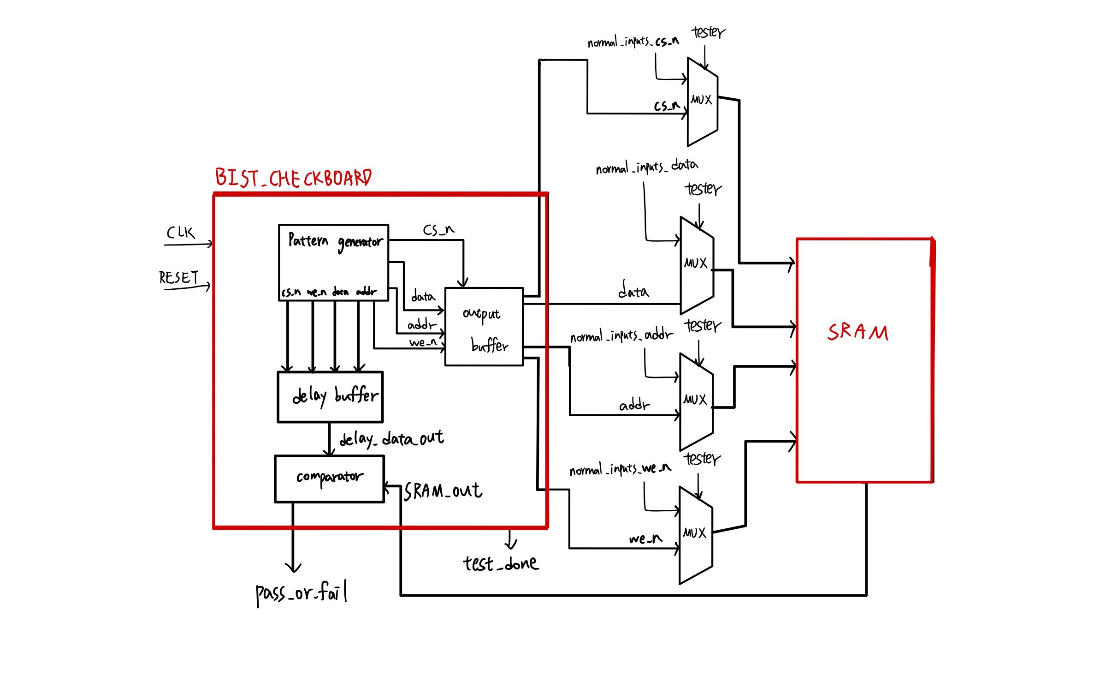
自動產生的描述

Fig 18 RTL code (Comparator.v)

Comparator對Pattern Generator的data\_out及SRAM的data\_out進行比較，若兩者相等，則使output Equal\_or\_not為1;否則為0。

### BIST\_Checkerboard Module

BIST\_Checkerboard的電路架構也跟BIST\_March相同，因為兩者只是差在寫值或讀值的值不同，且由於BIST\_checkerboard的狀態較少所以比起BIST\_March所需的cycle數也少很多，很快就可以將所有SRAM的memory address都測過一遍。



### nWave

一張含有 螢幕擷取畫面, 行, 鮮豔, 電路 的圖片

自動產生的描述

1. **BIST:**

上圖波形tester為1，表示現在模式是讓SRAM接收Normal Inputs，即為(a)小題的testbench。

1. **Checkerboard Pattern Generator:**

因現在是tester = 1，SRAM不會接收Checkerboard Pattern的訊號，這邊先不詳細探討。

1. **Normal Inputs:**

此為由(a)小題的testbench所產生之we\_n, cs\_n, data\_in, address。

1. **SRAM**

因tester為1，故現在為Normal Inputs模式，SRAM接收的訊號來自Normal Inputs。

一張含有 螢幕擷取畫面, 文字, 電子產品, 陳列 的圖片

自動產生的描述

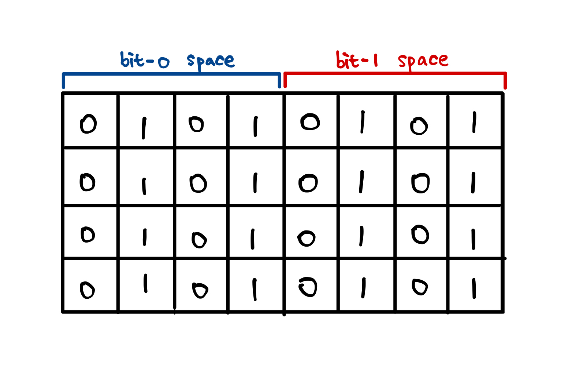
1. **BIST:**

上圖波形tester為0，表示現在模式是讓SRAM接收BIST電路輸出的訊號，即為Checkboard Pattern Generator。

Pass\_or\_fail即為Comparator在we\_n=1時進行比較的結果，pass\_or\_fail = 1代表SRAM data\_out與Checkerboard Pattern Generator值相等，反之則為0。

1. **Checkerboard Pattern Generator:**

we\_n一開始為0，即寫入模式，從addr = 0-15依序寫入0, 3, 0, 3…(當addr為偶數時 data\_out為0;奇數時 data\_out為3)，如下圖示意圖:



當完成addr 0-15的寫入後，we\_n轉為1，data\_out仍與剛剛一樣，當addr為偶數時 data\_out為0;奇數時 data\_out為3，**目的是為了在得到SRAM data\_out後，可以與其compare。**

1. **Normal Inputs:**

此為由(a)小題的testbench所產生之we\_n, cs\_n, data\_in, address，因現在是tester = 0，故不會有Normal Inputs輸入進SRAM。

1. **SRAM**

因tester為0，故現在為BIST test模式，SRAM接收的訊號來自Checkboard Pattern Generator。

一張含有 文字, 螢幕擷取畫面, 陳列, 軟體 的圖片

自動產生的描述

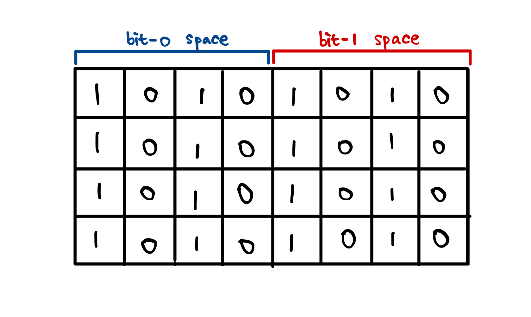
1. **BIST:**

上圖波形tester為0，表示現在模式是讓SRAM接收BIST電路輸出的訊號，即為Checkboard Pattern Generator。

Pass\_or\_fail即為Comparator在we\_n=1時進行比較的結果，pass\_or\_fail = 1代表SRAM data\_out與Checkerboard Pattern Generator值相等，反之則為0。

1. **Checkerboard Pattern Generator:**

we\_n一開始為0，即寫入模式，但與上頁不同，這裡從addr = 0-15依序寫入3, 0, 3, 0…(當addr為偶數時 data\_out為3;奇數時 data\_out為0)，如下圖示意圖:



當完成addr 0-15的寫入後，we\_n轉為1，data\_out仍與剛剛一樣，當addr為偶數時 data\_out為3;奇數時 data\_out為0，**目的是為了在得到SRAM data\_out後，可以與其compare。**

1. **Normal Inputs:**

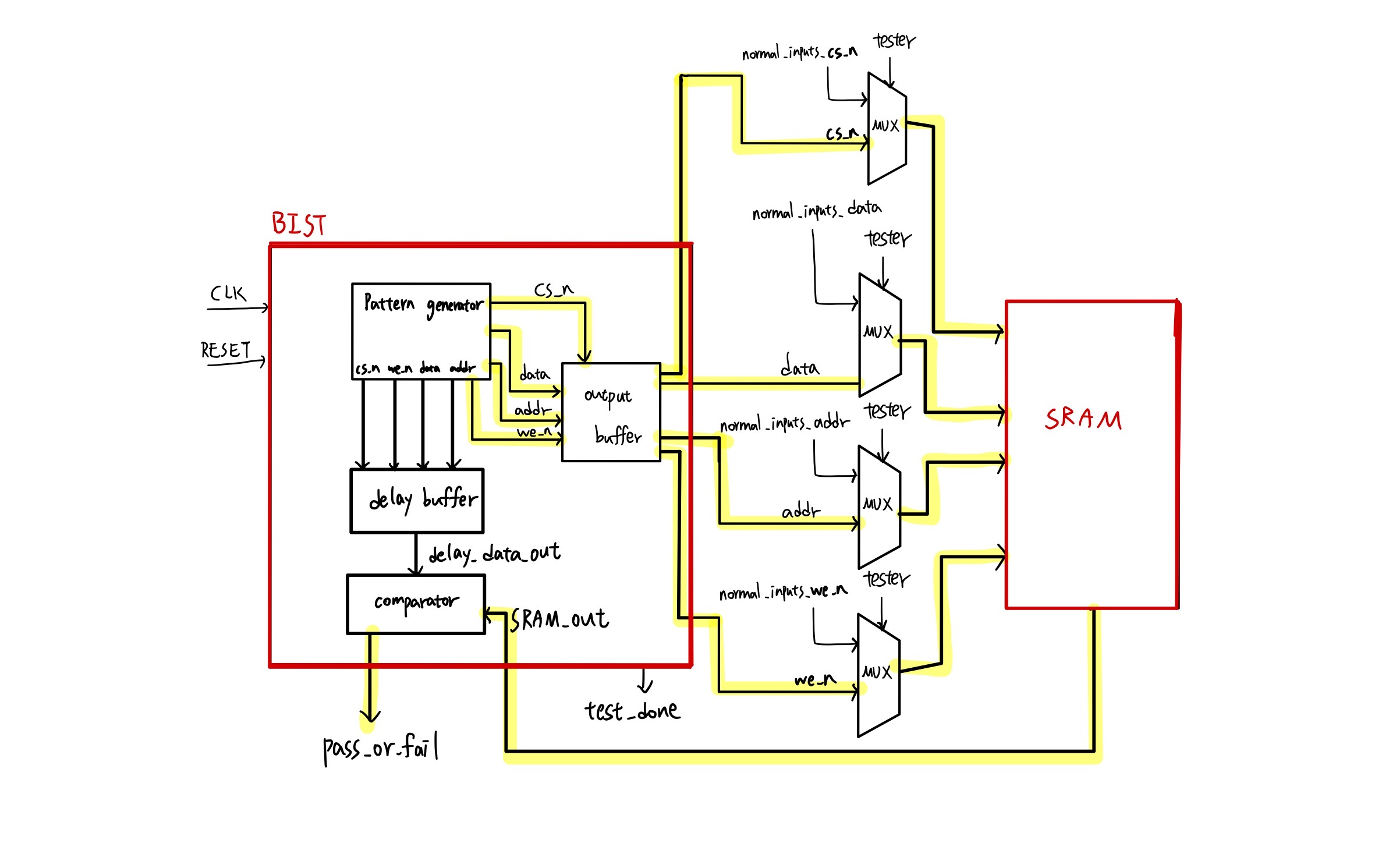
此為由(a)小題的testbench所產生之we\_n, cs\_n, data\_in, address，因現在是tester = 0，故不會有Normal Inputs輸入進SRAM。

1. **SRAM**

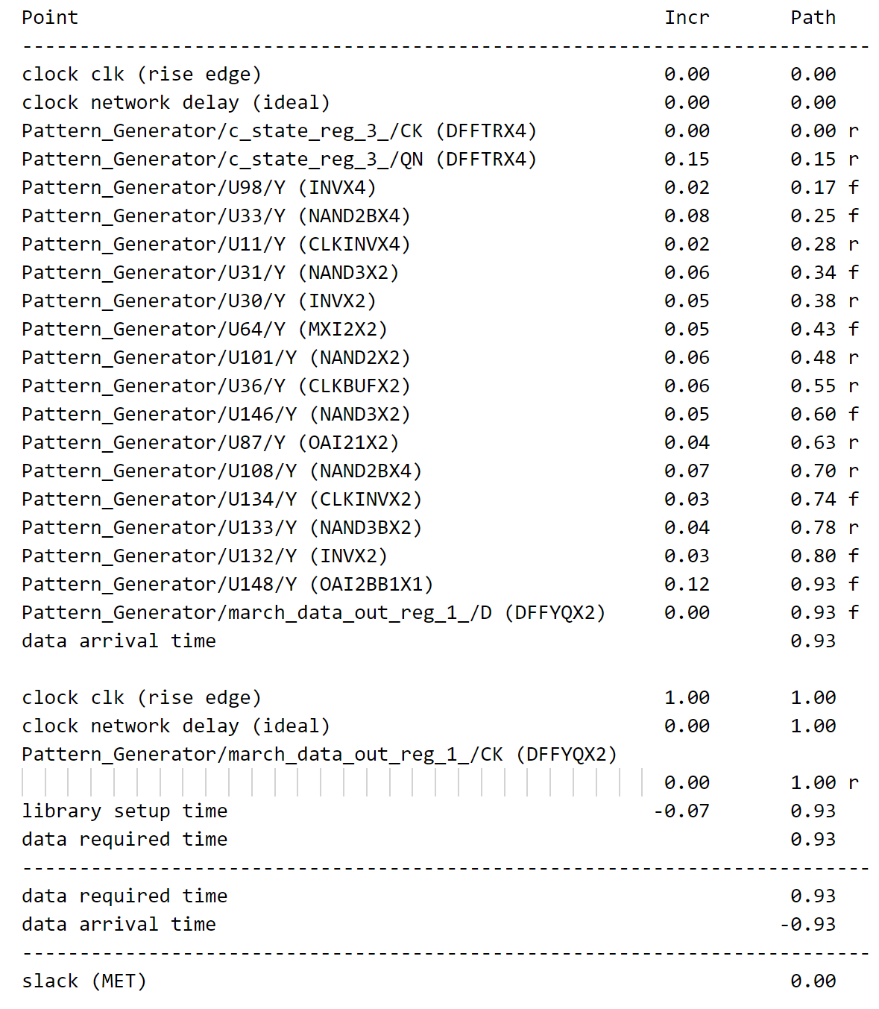
因tester為0，故現在為BIST test模式，SRAM接收的訊號來自Checkboard Pattern Generator。

# Synthesize your BIST circuit using Design Compiler. Report the critical path delay of your BIST circuit.

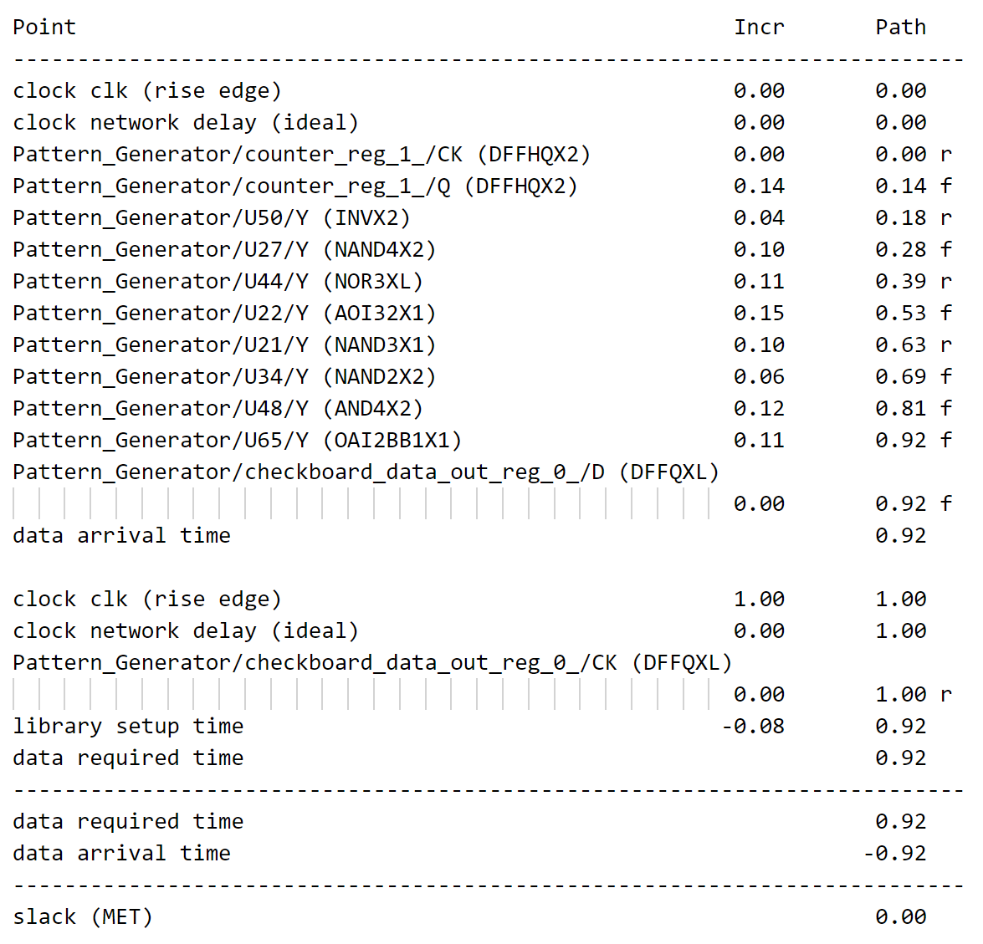
Critical path



BIST\_MARCH TEST Synthesize report



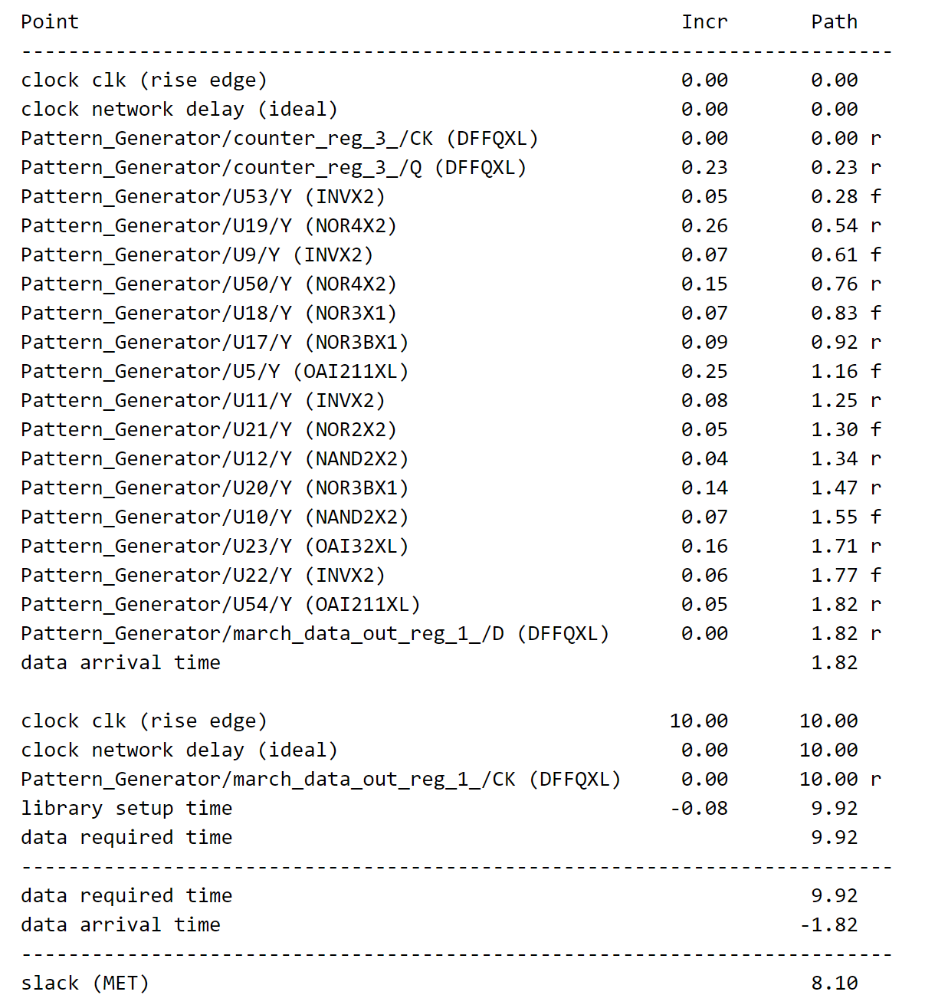
BIST\_CHECKERBOARD TEST Synthesize report



# Perform gate-level simulation including your BIST circuit and the Verilog model for the SRAM block under test. Assume a clock rate of 100MHz. Does you BIST circuit report a test result of “pass”? If not, discuss why.

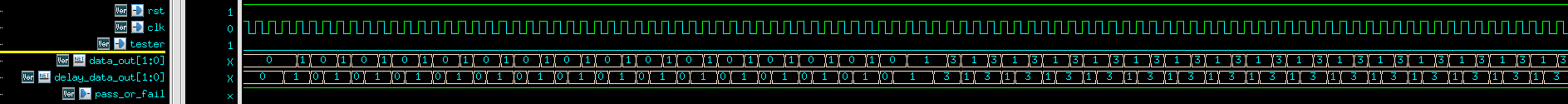
BIST\_MARCH\_TEST Timing report (Clock rate = 100MHz )

用100MHz下去合成可以看到timing slack = 8.1不會有timing violation的問題。



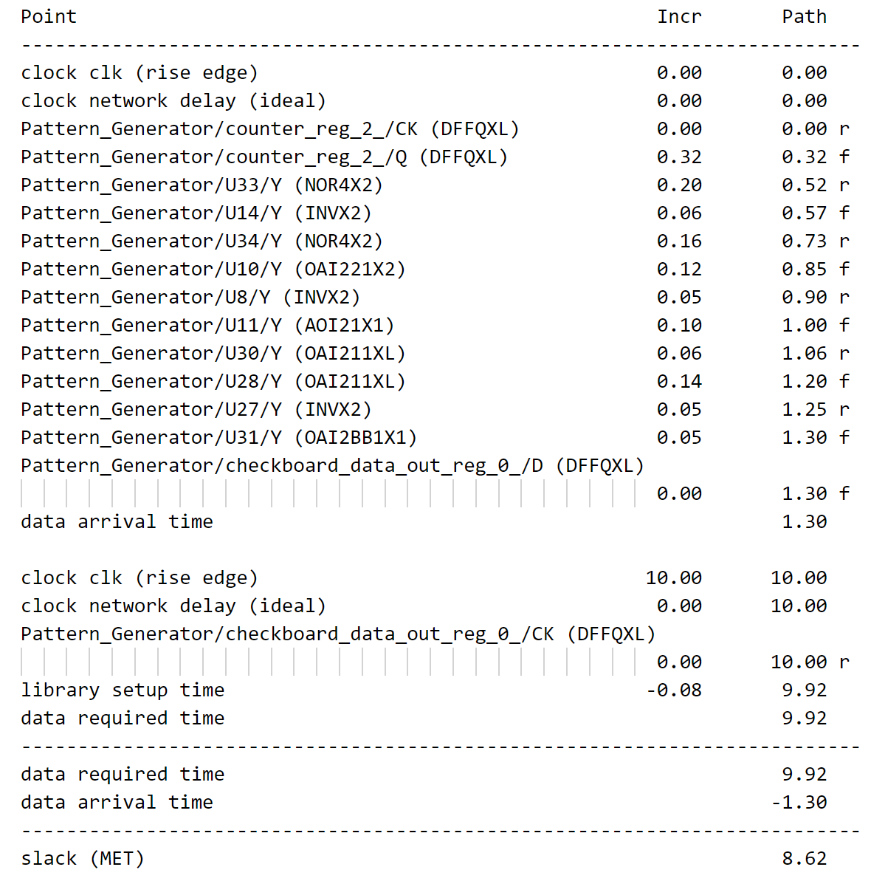
BIST\_MARCH\_TEST nWave result

當tester=0時代表是選擇測試MARCH的測資，可以看到合成完的結果pass\_or\_fail都是為1代表結果正確。



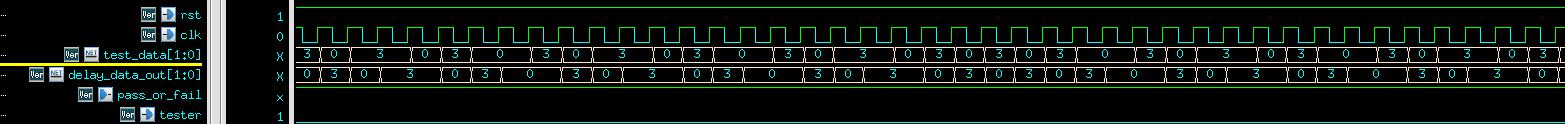
BIST\_CHECKERBOARD\_TEST Timing report (Clock rate = 100MHz )

用100MHz下去合成可以看到timing slack = 8.62不會有timing violation的問題。



BIST\_CHECKERBOARD\_TEST nWave result

當tester=0時代表是選擇測試CHECKERBOARD的測資，可以看到合成完的結果pass\_or\_fail都是為1代表結果正確。



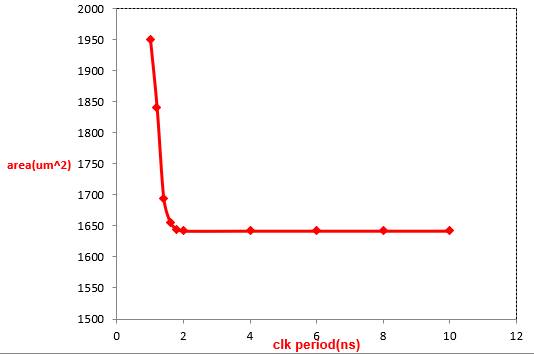
# Increase the clock rate gradually (i.e., from 100MHz to some higher rates) and check if at some point the test result becomes erroneous (i.e., turning from “pass” to “fail”). Discuss if you can derive the maximum operating speed of your BIST circuit in this experiment.

Test Result

下面討論了我們把march\_test跟checkerboard\_tset的clock rate從100MHz提升到1000MHz所得到的結果，但再把CLK的cycle time降到0.9後發現會有timing violation，得到的timing slack為負值，所以我們在兩個test的電路中所能達到的最快運行速度皆是1GHz。

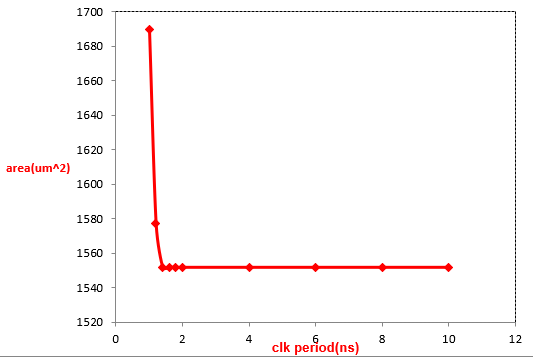
BIST\_march

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CLK period(ns)** | **freq(MHz)** | **total area(um^2)** | **timing slack(ns)** | **power(mW)** |
| 1 | 1000 | 1950.98 | 0 | 0.7995 |
| 1.2 | 833.3333333 | 1840.91 | 0 | 0.6166 |
| 1.4 | 714.2857143 | 1694.85 | 0 | 0.4838 |
| 1.6 | 625 | 1656.04 | 0.01 | 0.4205 |
| 1.8 | 555.5555556 | 1644.75 | 0.08 | 0.3749 |
| 2 | 500 | 1641.93 | 0.13 | 0.3376 |
| 4 | 250 | 1641.93 | 2.13 | 0.1714 |
| 6 | 166.6666667 | 1641.93 | 4.13 | 0.116 |
| 8 | 125 | 1641.93 | 6.13 | 0.0883 |
| 10 | 100 | 1641.93 | 8.13 | 0.0717 |



**BIST\_Checkerboard**

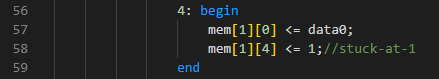
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CLK period(ns)** | **freq(MHz)** | **total area(um^2)** | **timing slack(ns)** | **power(mW)** |
| 1 | 1000 | 1689.91 | 0 | 0.8013 |
| 1.2 | 833.3333333 | 1577.02 | 0 | 0.597 |
| 1.4 | 714.2857143 | 1551.61 | 0.02 | 0.5122 |
| 1.6 | 625 | 1551.61 | 0.22 | 0.4488 |
| 1.8 | 555.5555556 | 1551.61 | 0.42 | 0.3995 |
| 2 | 500 | 1551.61 | 0.62 | 0.36 |
| 4 | 250 | 1551.61 | 2.62 | 0.1824 |
| 6 | 166.6666667 | 1551.61 | 4.62 | 0.1232 |
| 8 | 125 | 1551.61 | 6.62 | 0.0936 |
| 10 | 100 | 1551.61 | 8.62 | 0.0759 |



# Try to inject a stuck-at-1 fault to bit #2 of the SRAM word #5, and report the test result of your BIST at 100MHz

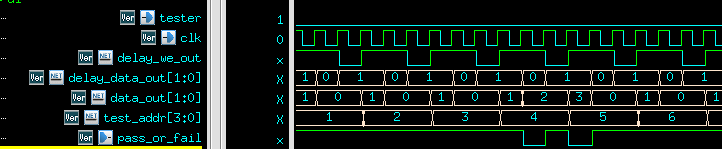
SRAM stuck-at-1 fault

依照題目敘述將word#5也就是address=4的bit1位置插入一個SA-1 fault，之後再利用先前的電路設計架構測試BIST\_March test跟BIST\_Checkerboard test，看看這個fault會不會被偵測出來。



BIST\_March Test

用CLK cycle time=10ns下去合成，而且因為我們寫的是時序電路，pass\_or\_fail會判斷若addr為4時，下一個cycle才會掉下去為0，(但若delay\_we\_out為0時，pass\_or\_fail強制為1)，如圖所示。所以我們設計的電路能偵測出這個stuck-at-1 fault。



BIST\_Checkerboard Test

BIST\_Checkerboard Test也是利用相同方法，用CLK cycle time=10ns下去合成，而且因為我們寫的是時序電路，pass\_or\_fail會判斷若addr為4時，下一個cycle才會掉下去為0，如圖所示。所以我們設計的電路能偵測出這個stuck-at-1 fault。

