**國立清華大學**

**超大型積體電路測試**

**VLSI Testing**



**Homework 1**

**系所級:電子所一年級**

**學號:111063548**

**姓名:蕭方凱**

**指導老師:黃錫瑜教授**

目錄

[1. Consider the testing of a gate-level circuit as shown below. The primary input signals are {x1, x2, x3} and the primary output signal is {z}. The output signals of logic gates {A, B} are denoted as {a, b}, and the branches of primary input signal x2 is clled y1 and y2, respectively. 3](#_Toc129018301)

[(a) Write a software program (using C, C++, or any other programing language) that can exhaustively simulate the logic behavior of the given circuit under each of the 2 3 = 8 input vectors). Note that this can be done by executing the following Boolean equations in sequence in your program: a = ~(x1 or x2); b = ~(x2 and x3); z = (a or b); List the results as a truth table for output signal z. (Note this truth table contains 8 entries, one for each input combination). 3](#_Toc129018302)

[(b) Enhance your program so that it can perform exhaustive fault simulation for bridging faults, {AND-bridging between a and b} and {OR-bridging between y1 and x3}. Note that you need to report the total number of possible test patterns for each of the above two bridging faults. (Hint: perform fault injection, run exhaustive simulation on the faulty circuit, and then compare the results with those of the fault-free circuit. The fault injection can be done manually by changing the compiled code.) 4](#_Toc129018303)

圖表目錄

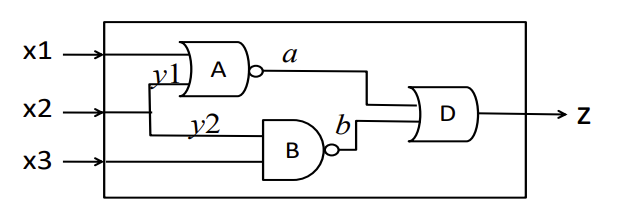
[Fig 1 Python codes 3](#_Toc129018408)

[Fig 2 Truth Table 4](#_Toc129018409)

[Fig 3 Enhanced part of codes 4](#_Toc129018410)

[Fig 4 Truth Table of bridging fault 5](#_Toc129018411)

# Consider the testing of a gate-level circuit as shown below. The primary input signals are {x1, x2, x3} and the primary output signal is {z}. The output signals of logic gates {A, B} are denoted as {a, b}, and the branches of primary input signal x2 is called y1 and y2, respectively.



## Write a software program (using C, C++, or any other programing language) that can exhaustively simulate the logic behavior of the given circuit under each of the 2 3 = 8 input vectors). Note that this can be done by executing the following Boolean equations in sequence in your program: a = ~(x1 or x2); b = ~(x2 and x3); z = (a or b); List the results as a truth table for output signal z. (Note this truth table contains 8 entries, one for each input combination).

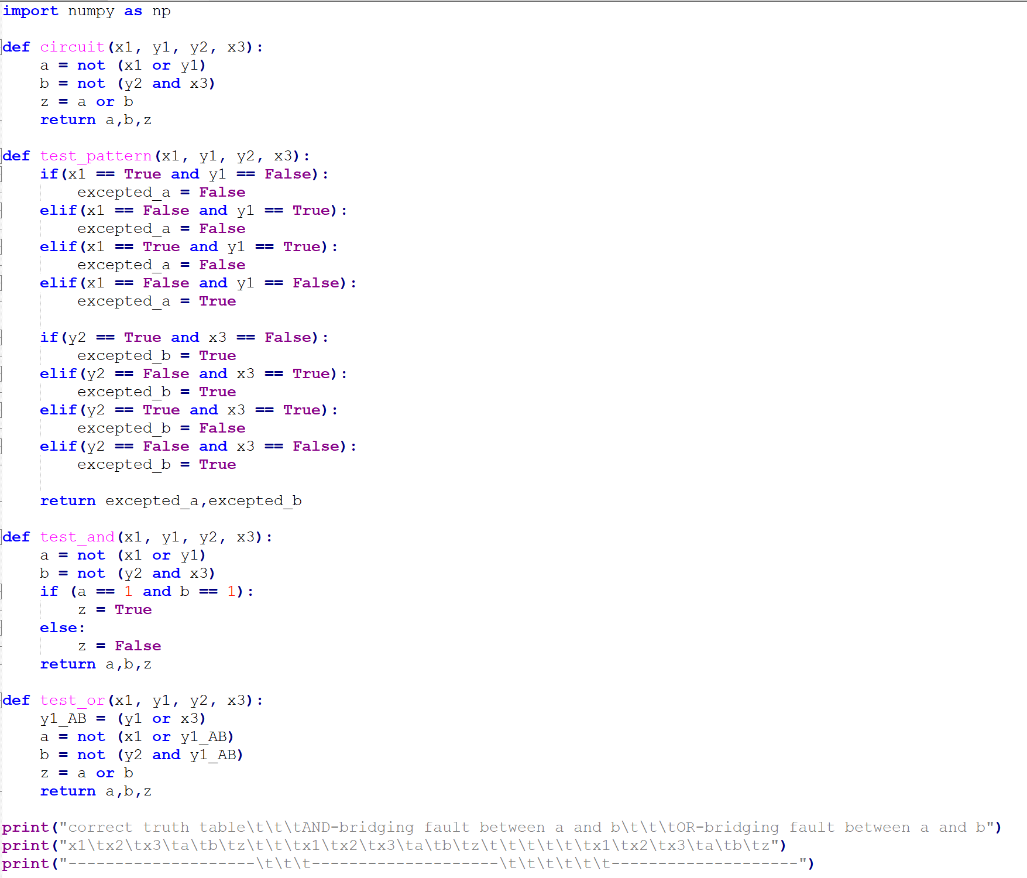


Fig Python codes

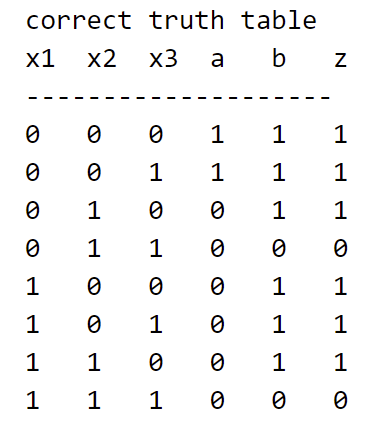


Fig Truth Table

## Enhance your program so that it can perform exhaustive fault simulation for bridging faults, {AND-bridging between a and b} and {OR-bridging between y1 and x3}. Note that you need to report the total number of possible test patterns for each of the above two bridging faults. (Hint: perform fault injection, run exhaustive simulation on the faulty circuit, and then compare the results with those of the fault-free circuit. The fault injection can be done manually by changing the compiled code.)

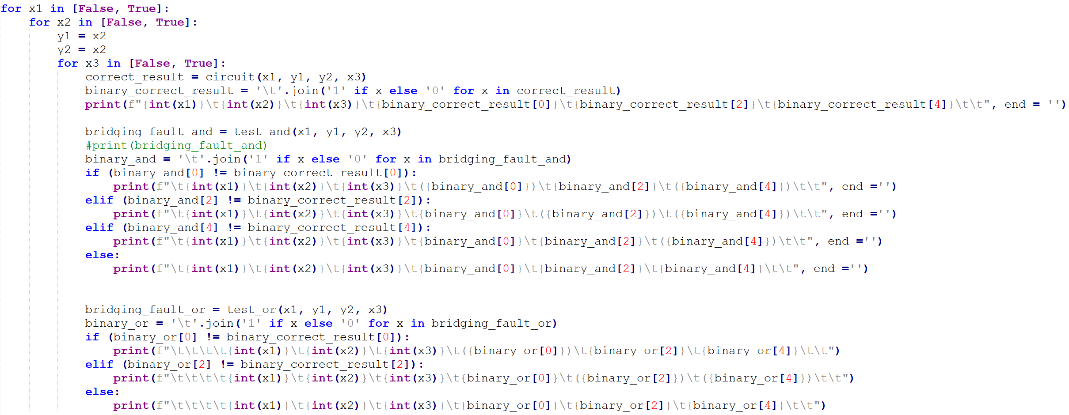


Fig Enhanced part of codes

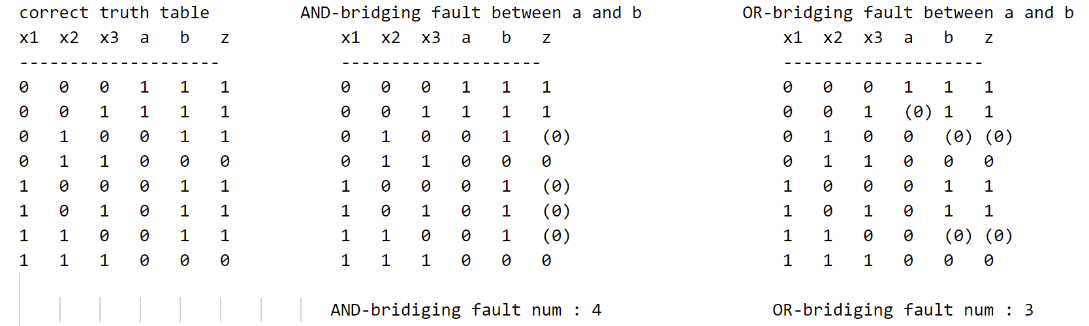
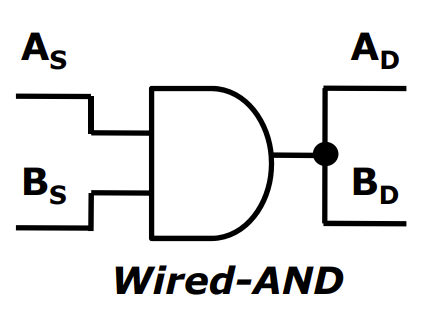


Fig Truth Table of bridging fault(括號數字代表該值異常)

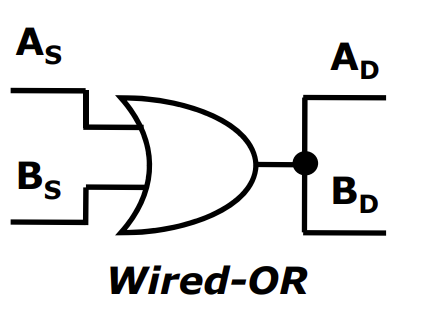
Comment:

Truth Table從左到右分別為:correct truth table、and bridging fault的truth table及or bridging fault的truth table。數字括號起來代表該值是錯誤的，以下分解釋不同程式碼區塊的功能目的:

1. Def circuit : 將電路用python寫出，使output z 根據輸入的x1, x2, x3變化 而產生對應的輸出值。
2. Def Test\_pattern : 寫一個擁有正確電路行為的pattern，供電路偵錯時用。
3. Def test\_and : 與def circuit不同，此副程式的目的是製造a 和 b之間的and bridge。如下圖:



1. Def test\_or : 與def circuit不同，此副程式的目的是製造y1 和 x3之間的or bridge。如下圖:



1. For迴圈部分為主要功能，首先套用def circuit的正確電路，print出正確的correct truth table。接著是偵錯and bridging fault的程式碼，套用test\_and副程式，模擬a和b後方出現and gate，進而影響output z的正確性。最後是偵錯or bridging fault，在def test\_or中畫上一個y1 or x3的or gate，使a或b 不正常輸出，與correct truth table比較後，找出錯誤值。

總結來說，從Truth Table觀察output z是否正確，若不正確，可檢測a和b的值，a和b與correct truth table相同，代表是and bridging fault，也就是a 和b 後方多出的一個and gate; a和b與correct truth table不同，則代表在電路Input訊號處多了一個y1 or x3的or gate。