|  |  |
| --- | --- |
| **Lab 2** | |
| 學號: 110062131 | 姓名: 馬毓昇 |

1. **Lab Implementation**

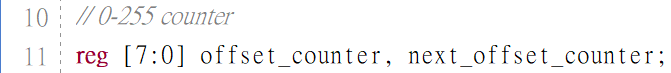
Lab2\_1: 以7-bit counter加密8-bit的in\_data再加上4-bit的error correct code做成12-bit的out\_data。

一張含有 文字 的圖片

自動產生的描述

Lab2\_1 segments:

1. 7-bit counter (挪用offset\_counter)



2. 5 states

一張含有 文字 的圖片

自動產生的描述

3. 因為in\_data是8 bit而out\_data是12 bit，所以用concat的方式在前面加上4個0

一張含有 文字 的圖片

自動產生的描述

4. 用offset\_counter % 128的方式來模擬0~127的7-bit counter

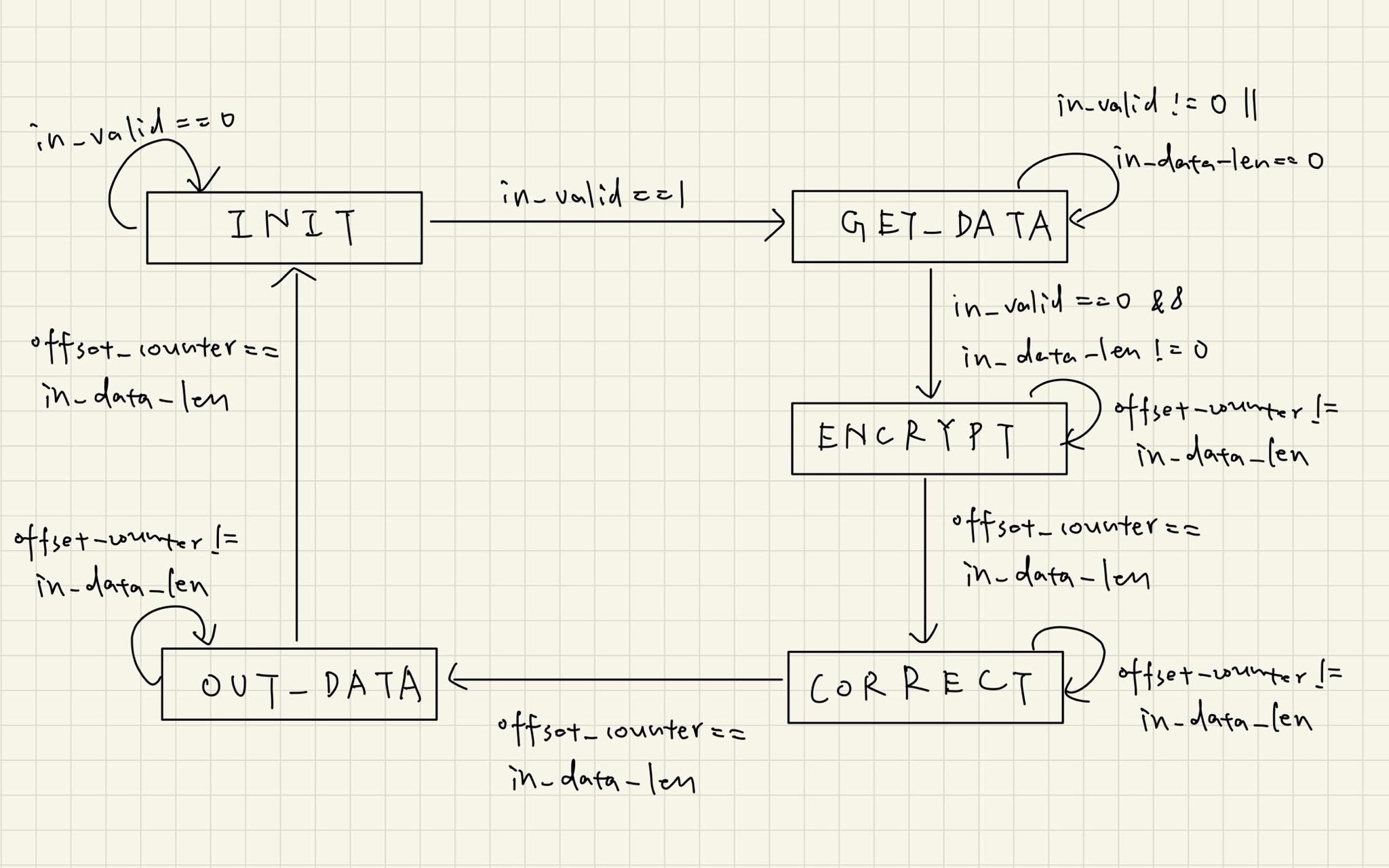
一張含有 文字 的圖片

自動產生的描述

5. 用concat的方式來加上correct code做成12-bit的out\_data

一張含有 文字 的圖片

自動產生的描述Lab2\_1 FSM:



1. INIT -> GET\_DATA:

in\_valid是1，代表正在接受input。

2. GET\_DATA -> ENCRYPT:

in\_valid是0，代表不在接收input，如果此時in\_data\_len不是0，就代表已經接收到data且接收完畢可以處理data了。

3. ENCRYPT -> CORRECT:

offset\_counter數到in\_data\_len代表data已經全數處理好了可以進下一個階段。

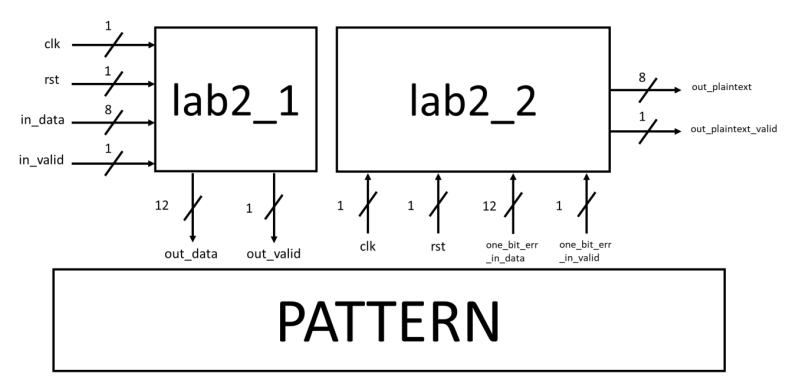
4. CORRECT -> OUT\_DATA:

offset\_counter數到in\_data\_len代表data已經全數處理好了可以進下一個階段。

5. OUT\_DATA -> INIT:

offset\_counter數到in\_data\_len代表data已經全數處理好了可以進下一個階段。

Lab2\_2: 依Error correct algorithm修正12-bit的in\_data，再用同樣的計數器邏輯解密data成8-bit的out\_data。



Lab2\_2 segments:

1. 7-bit counter

2. 5 states

一張含有 文字 的圖片

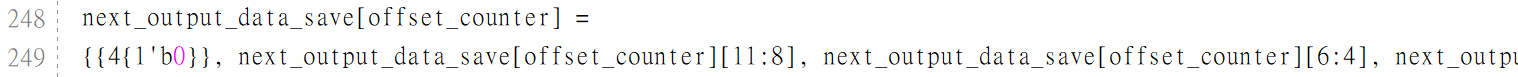
自動產生的描述

3. 依code correct algorithm來修正data

一張含有 桌 的圖片

自動產生的描述

最後再用concat的方式做出8-bit的解密前out\_data (前面補0 )

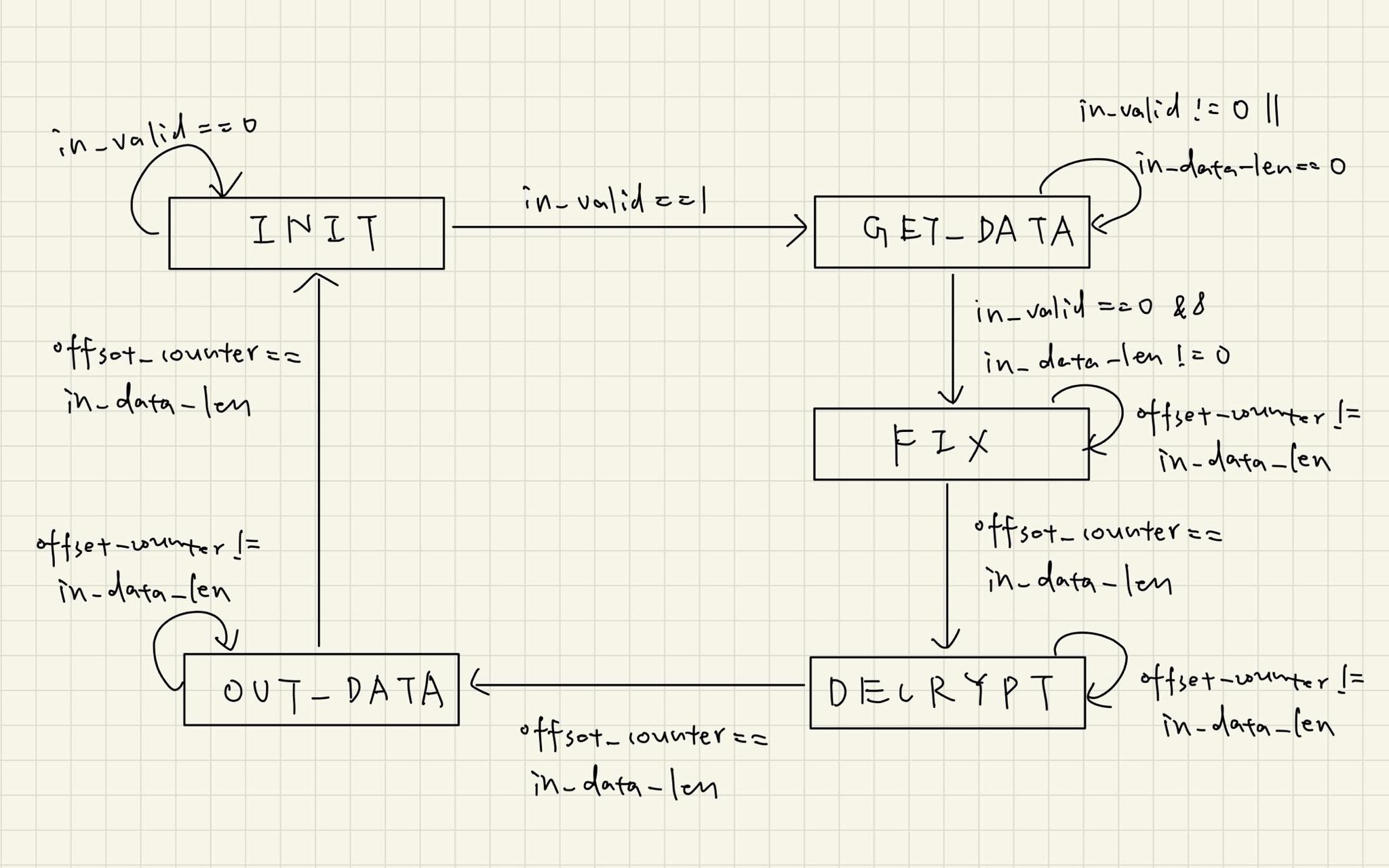


4. 解密

一張含有 文字 的圖片

自動產生的描述

Lab2\_2 FSM:



1. INIT -> GET\_DATA:

in\_valid是1，代表正在接受input。

2. GET\_DATA -> FIX:

in\_valid是0，代表不在接收input，如果此時in\_data\_len不是0，就代表已經接收到data且接收完畢可以處理data了。

3. FIX -> DECRYPT:

offset\_counter數到in\_data\_len代表data已經全數處理好了可以進下一個階段。

4. DECRYPT -> OUT\_DATA:

offset\_counter數到in\_data\_len代表data已經全數處理好了可以進下一個階段。

5. OUT\_DATA -> INIT:

offset\_counter數到in\_data\_len代表data已經全數處理好了可以進下一個階段。

我在本次lab了解到了怎麼設計一個FSM還有用verilog實作，不僅學會了一種error correct的演算法，和重新複習了一遍counter的實作方式。

1. **Questions and Discussions**

A: 如果是非同步reset的話，只要rst是HIGH，不論clk進來與否都會重置。一張含有 文字 的圖片

自動產生的描述

將這種有if(rst) 的always block的sensitivity list加上negedge rst即可達成非同步reset，例如：把35行改成always @(posedge clk or negedge rst) begin。

B: 可以用while、repeat或是forever+break。例如：

一張含有 文字 的圖片

自動產生的描述

1. **Problem Encountered**

1. 一開始思考了很久，counter究竟是每個testcase都會歸零還是會繼續數下去。解決辦法沒想到就是直接去比對message的那兩個檔案，馬上就真相大白了，又快又正確。

2. 在lab2\_2 fix bit error實作的過程中，沒發現一開始給的spec的B11的case判斷是錯的，後來debug了一段時間才發現這邊怪怪的，於是自己整理了一次。雖然多花了一段時間debug，但也因為這次自己整理的機會，能更加理解到為何透過XOR的使用，就能夠達成這個Error correct algorithm，原來XOR的應用是這麼的廣。

1. **Suggestions**

希望之後的spec能開好！辛苦助教了！