2020 Digital IC Design

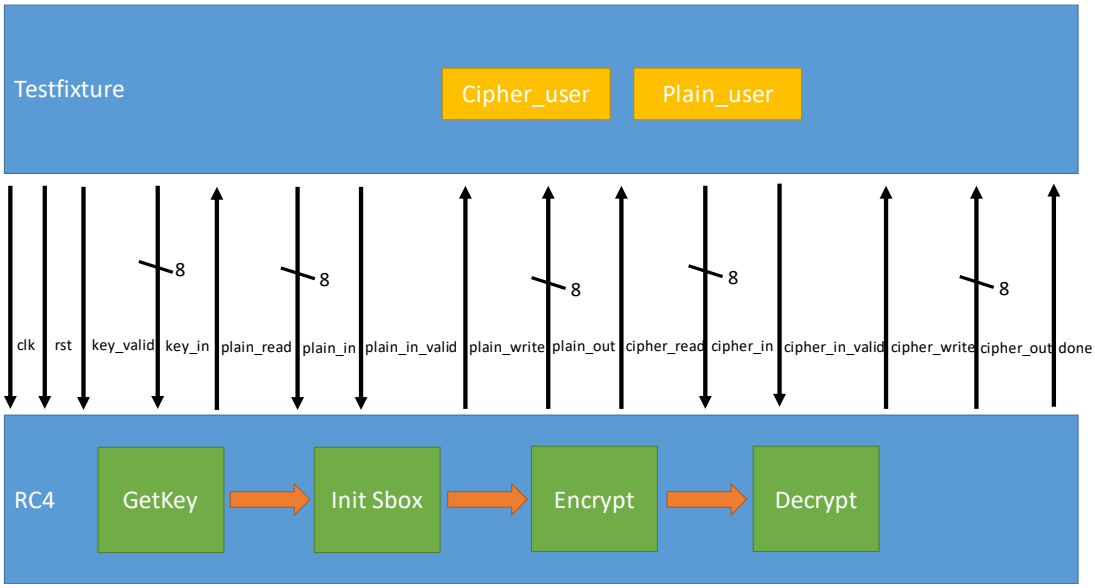
Homework 4: RC4 Encrypt

1. Introduction

在密碼學中，RC4（來自 Rivest Cipher 4 的縮寫）是一種流加密算法，密鑰長度可變。它加解密使用相同的密鑰，因此也屬於對稱加密算法。RC4 是有線等效加密（WEP）中採用的加密算法，也曾經是 TLS 可採用的算法之一。由於 RC4 算法存在弱點，2015 年 2 月所發布的 RFC 7465 規定禁止在 TLS 中使用 RC4 加密算法。RC4 由偽隨機數生成器和異或運算組成。RC4 的密鑰長度可變，範圍是[1,255]。RC4 一個字節一個字節地加解密。給定一個密鑰，偽隨機數生成器接受密鑰並產生一個 S 盒。S 盒用來加密數據，而且在加密過程中 S 盒會變化。由於異或運算的對合性，RC4 加密解密使用同一套算法。此次作業請實作一個 RC4 加解密的電路，其中金鑰的長度固定為 32，明文的長度為不固定，最長長度不超過 2048，Sbox 大小為 64，利用輸入金鑰對明文進行加密，然後將加密完的字元輸出，再將所輸出加密的字元輸入，進行解密，還原出原本的明文。

2. Design Specifications

2.1 Block overview



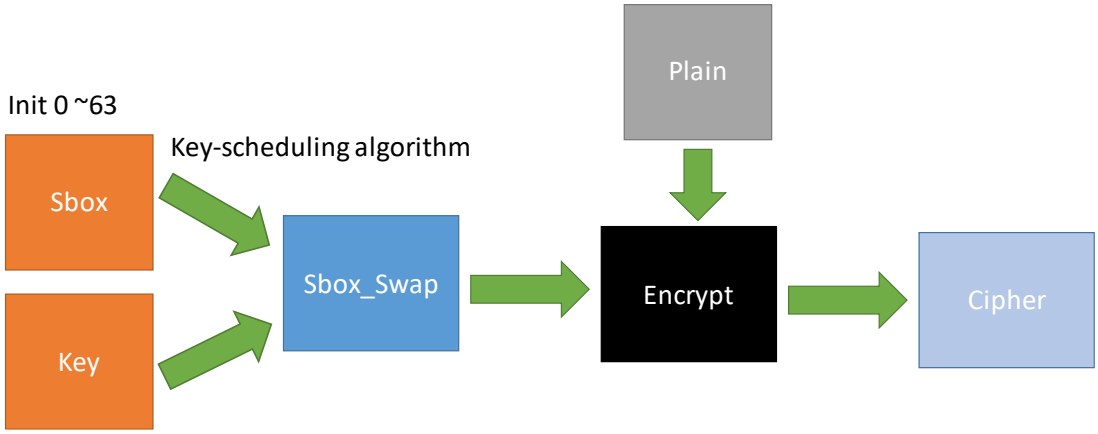
圖一、系統方塊圖

2.2 I/O Interface

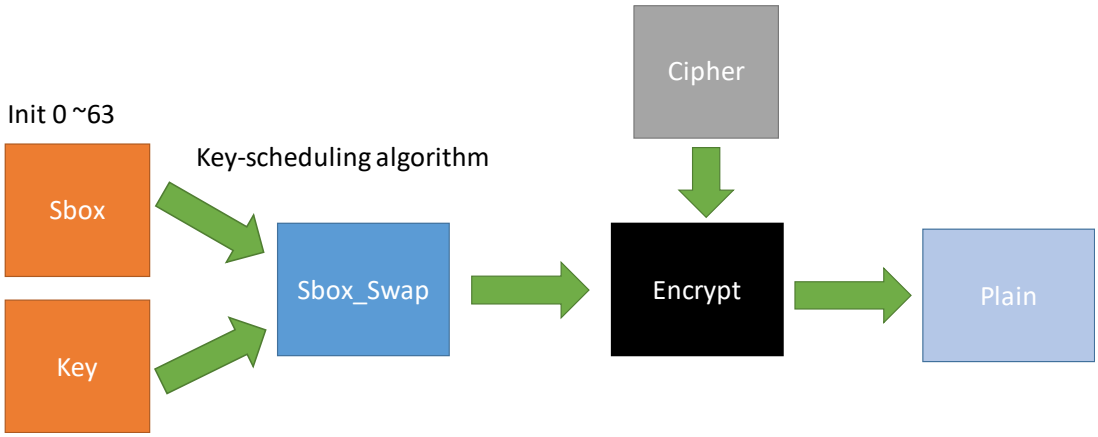
Name	I/O	Width	Description
clk	I	1	系統時脈訊號。本系統為同步於時脈正緣之同步設計
rst	I	1	高位準"非"同步(active high asynchronous)之系統重置信號
key_valid	I	1	當 key 準備好時，會先將 key_valid 設成 high，然後在下一個負緣輸出 key 值。
key_in	I	8	key data 輸入訊號線，輸入的 data size 為 8 bits，在 key_valid 設成 high 的下一個 cycle 的負緣輸出。
plain_read	O	1	當要索取明文時請將 plain_read 設成 high 且 plain_write 設成 low，再下一個 cycle 後會將其值輸入。
plain_in	I	8	輸入明文資料訊號，由 8bits 整數組成，為無號數
plain_in_valid	I	1	因為明文長度不固定，所以當輸入的明文為有效時，plain_in_valid 為 high，若無效時 plain_in_valid 為 low。
plain_write	O	1	若要將**解密後的明文**輸出至 testfixture 記憶體，將 plain_write 設為 high，其他不須寫入時務必設成 low。
plain_out	O	8	解密後運算結果記憶體寫出訊號，由 8bits 整數(MSB)組成，為無號數。
cipher_read	O	1	當要索取密文時請將 cipher_read 設成 high 且 cipher_write 設成 low，再下一個 cycle 後會將其值輸入。
cipher_in	I	8	輸入密文資料訊號，由 8bits 整數組成，為無號數
cipher_in_valid	I	1	因為密文長度不固定，所以當輸入的密文為有效時，cipher_in_valid 為 high，若無效時 cipher_in_valid 為 low。
cipher_write	O	1	若要將**加密後的密文**輸出至 tb 記憶體，將 cipher_write 設為 high，其他不須寫入時務必設成 low。
cipher_out	O	8	加密後運算結果記憶體寫出訊號，由 8bits 整數(MSB)組成，為無號數。
done	O	1	如果系統的運算結束，將 done 訊號輸出

			high，若尚未做完請務必輸出 low

2.3 Function Description



圖二、加密流程圖



圖三、解密流程圖

本系統的 key 長度為 32，而明文的長度為不固定，最長為 2048，key 的資料儲存於 testfixture 中，在系統進行 reset 之後，下一個 cycle 會先輸出 key_valid = high 然後再過一個 cycle 後輸出 key 的值，當 key_valid 為 high(除了第一個 cycle)時代表 key 值有效，當 key 值輸入完畢後，同學需先將 key 跟 Sbox 進行打亂，Sbox 一開始為 0~63，利用下圖四的 Pseudo code 進行打亂後，再利用打亂後的 Sbox 進行加密，加密演算法如下圖五的 Pseudo code，當加密完成後的密文請利用 cipher_write 和 cipher_out 將其結果輸出至 testfixture 的記憶體中，當 plain_in_valid 等於 high 時代表明文為有效輸入，當 plain_in_valid 為 low 時代表明文輸入完畢，當明文輸入完畢後，方可藉由 cipher_read 來控制密文的輸入（注意：若明文加密後有錯，輸入的密文也是錯誤的），當 cipher_in_valid 等於 high 時代表密文為有效輸入，當 cipher_in_valid 為 low 時代表為密文輸入完畢，解密的演算法流程與加密相同，若系統已經將加解密動作完成時，請將 done 設為 high，即可驗證加密

跟解密是正確。

```
for i from 0 to 63
    S[i] := i
endfor
j := 0
for i from 0 to 63
    j := (j + S[i] + key[i mod 32]) % 64
    swap values of S[i] and S[j]
endfor
```

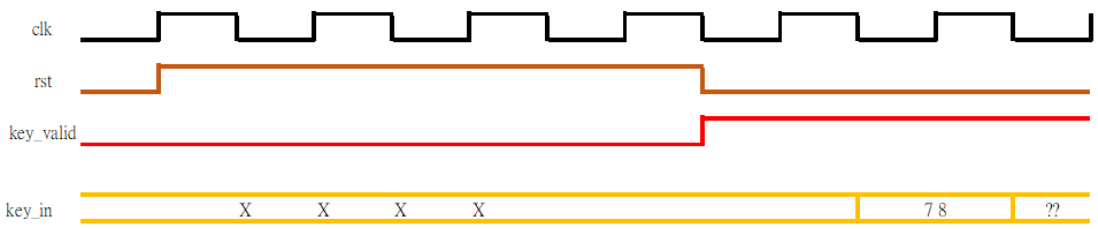
圖四、Key-scheduling algorithm (KSA)圖

```
i := 0
j := 0
while GeneratingOutput:
    i := (i + 1) mod 64   //a
    j := (j + S[i]) mod 64 //b
    swap values of S[i] and S[j]  //c
    k := inputByte ^ S[(S[i] + S[j]) % 64]
    output K
endwhile
```

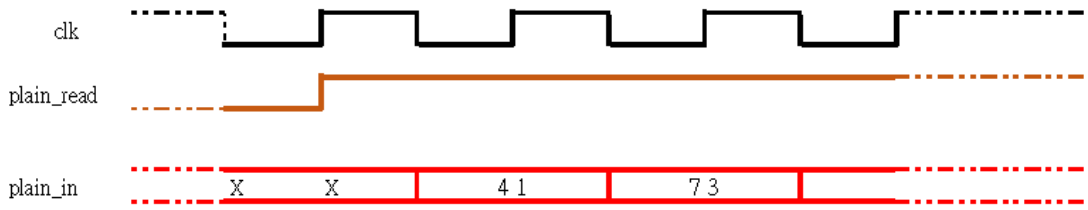
圖五、加解密演算法流程圖

圖六為 key_in 輸入的時序圖，key_valid 為 high 後下一個 clk cycle 便將 key 值輸入。

圖七為 cipher data 及 plain data 的讀取時序圖，當 read 設為 high 的下一個 clk 將會把資料輸入。

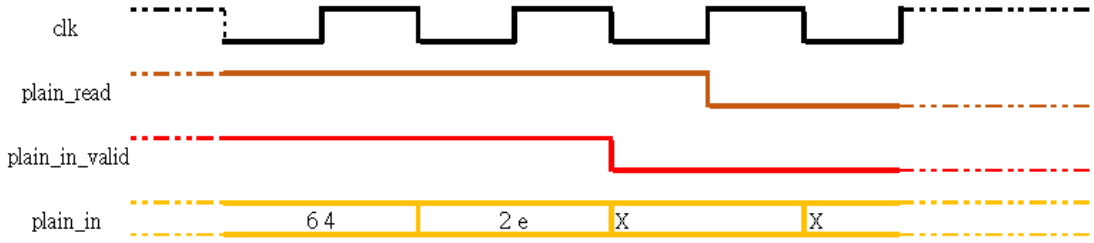


圖六、key_in 時序圖



圖七、plain data 及 cipher data 時序圖

圖八為 cipher data 及 plain data 的資料結束時序圖，當 plain_in_valid 或 cipher_in_valid 由 high 轉 low 時代表資料輸入結束。



圖八、plain data 及 cipher data 結束時序圖

※合成以及合成後模擬皆需要比較久的時間。

3. Scoring

3.1 Functional Part **[60%, 兩份 TB 各 30%]**

```
--------------------- T B 1 - S U M M A R Y -----------------

Congratulations! Cipher data have been generated successfully! The result is PASS!!

Congratulations! Plain data have been generated successfully! The result is PASS!!

** Note: $finish    : C:/Users/dic/Desktop/IC/0420/testfixture.v(244)
   Time: 44445 ns  Iteration: 2  Instance: /testfixture

--------------------- T B 2 - S U M M A R Y -----------------

Congratulations! Cipher data have been generated successfully! The result is PASS!!

Congratulations! Plain data have been generated successfully! The result is PASS!!

** Note: $finish    : C:/Users/dic/Desktop/IC/0420/testfixture2.v(244)
   Time: 39505 ns  Iteration: 2  Instance: /testfixture2
```

3.2 Gate Level Part **[20%, 兩份 TB 各 10%]**

3.2.1 Synthesis

Your code should be synthesizable. After synthesizing in Quartus, the file named *RC4.vo* and RC4.sdo will be obtained.

Device : Cyclone II EP2C70F896C8

3.2.2 Simulation

All of the result should be generated correctly using **RC4.vo** and **RC4.sdo**, and you will get the following message in ModleSim simulation. (There should be no setup or hold time violations.)

```
--------------------- T B 1 - S U M M A R Y -----------------

Congratulations! Cipher data have been generated successfully! The result is PASS!!

Congratulations! Plain data have been generated successfully! The result is PASS!!

** Note: $finish    : C:/Users/dic/Desktop/IC/0421/testfixture.v(244)
   Time: 157348706 ps  Iteration: 0  Instance: /testfixture

 --------------------- T B 2 - S U M M A R Y -----------------

 Congratulations! Cipher data have been generated successfully! The result is PASS!!

 Congratulations! Plain data have been generated successfully! The result is PASS!!

 ** Note: $finish    : C:/Users/dic/Desktop/IC/0421/testfixture2.v(244)
    Time: 171983706 ps  Iteration: 0  Instance: /testfixture2
```

3.3 Performance Part **[20%]**

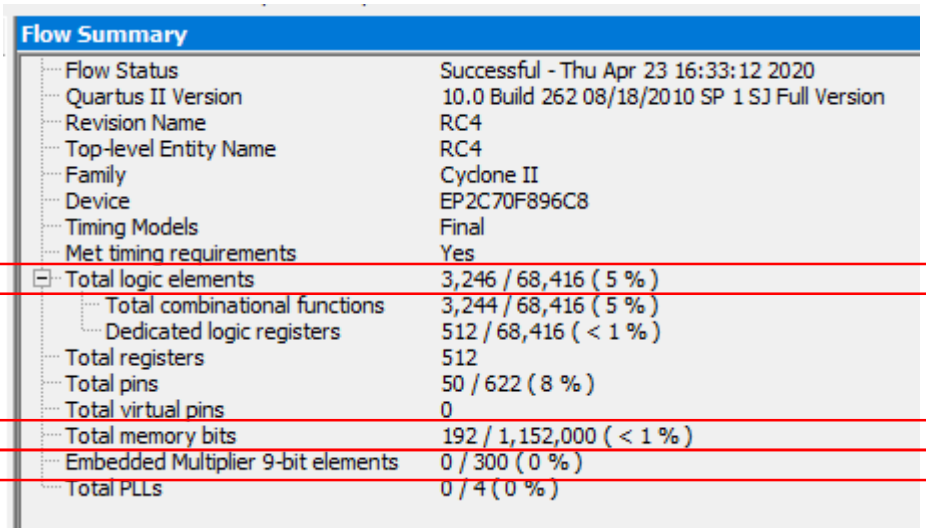
[20%] 完成 3.2 之要求並記錄合成後的 logic element 數量，以及模擬時能通過的最小 cycle 與模擬時間。評分標準為

(Total logic elements + total memory bit + 9*embedded multiplier 9-bit element)×(longest gate-level simulation time in ns)

越低越好

```
`timescale 1ns/10ps
`define CYCLE      30.0
`define End_CYCLE  100000
```

可根據自己的設計調整 CYCLE



Flow Summary	
Flow Status	Successful - Thu Apr 23 16:33:12 2020
Quartus II Version	10.0 Build 262 08/18/2010 SP 1 SJ Full Version
Revision Name	RC4
Top-level Entity Name	RC4
Family	Cyclone II
Device	EP2C70F896C8
Timing Models	Final
Met timing requirements	Yes
Total logic elements	3,246 / 68,416 (5 %)
Total combinational functions	3,244 / 68,416 (5 %)
Dedicated logic registers	512 / 68,416 (< 1 %)
Total registers	512
Total pins	50 / 622 (8 %)
Total virtual pins	0
Total memory bits	192 / 1,152,000 (< 1 %)
Embedded Multiplier 9-bit elements	0 / 300 (0 %)
Total PLLs	0 / 4 (0 %)

```
--------------------- T B 1 - S U M M A R Y -----------------

Congratulations! Cipher data have been generated successfully! The result is PASS!!

Congratulations! Plain data have been generated successfully! The result is PASS!!

** Note: $finish    : C:/Users/dic/Desktop/IC/0423/testfixture.v(244)
   Time: 91675293 ps  Iteration: 0  Instance: /testfixture

--------------------- T B 2 - S U M M A R Y -----------------

Congratulations! Cipher data have been generated successfully! The result is PASS!!

Congratulations! Plain data have been generated successfully! The result is PASS!!

** Note: $finish    : C:/Users/dic/Desktop/IC/0423/testfixture2.v(244)
   Time: 49435293 ps  Iteration: 0  Instance: /testfixture2
```

4. Submission

You should classify your files into three directories and compressed to .zip format. The naming rule is HW4_studentID_name.zip.

	RTL category
*.v	All of your Verilog RTL code
	Gate-Level category
*.vo	Gate-Level netlist generated by Quartus
*.sdo	SDF timing information generated by Quartus
	Documentary category
*.pdf	The report file of your design (in pdf).

4.1 Report file

Please follow the spec of report. You are asked to describe how the circuit is designed as detailed as possible, and the summary results and minimum CYCLE in post-sim are necessary.

4.2 Please submit your .zip file to folder HW4 in the moodle.

Deadline: 2020-05-26 23:55

5. If you have any problem, please contact by the TA by email
   f74044088@gs.ncku.edu.tw 倪祺婷
   weiting84610@gmail.com 陳威廷

Appendix

The Sbox value in ***testbench1***:

[56, 31, 12, 3, 20, 0, 28, 11, 30, 40, 52, 46, 37, 57, 50, 43, 16, 48, 26, 14, 62, 29, 41, 17, 47, 9, 23, 4, 45, 36, 33, 19, 55, 10, 35, 24, 53, 34, 7, 44, 1, 27, 13, 60, 21, 8, 18, 6, 58, 49, 54, 2, 15, 42, 22, 63, 38, 51, 61, 59, 39, 25, 5, 32]

The Sbox value in ***testbench2***:

[7, 13, 8, 12, 30, 44, 18, 37, 24, 29, 26, 51, 60, 20, 4, 34, 14, 32, 46, 2, 17, 53, 63, 3, 62, 6, 25, 5, 56, 21, 52, 1, 43, 16, 47, 35, 11, 61, 58, 57, 33, 27, 10, 42, 28, 19, 15, 36, 23, 59, 54, 40, 41, 45, 39, 22, 38, 48, 9, 50, 55, 0, 31, 49]