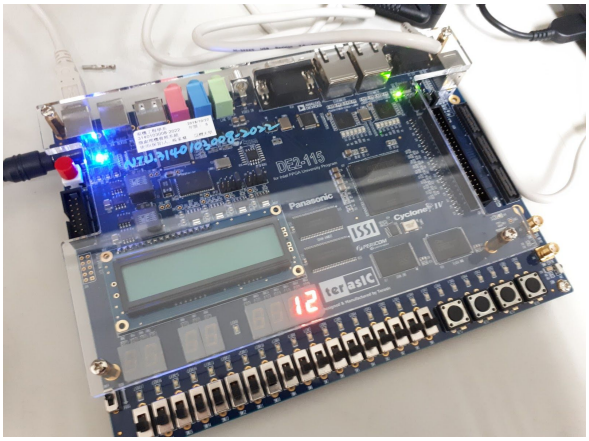
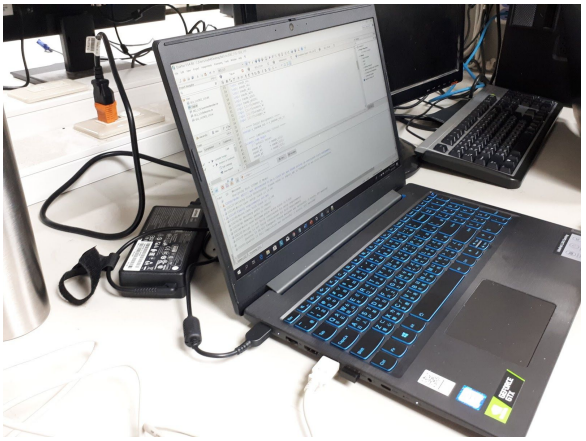
數位電路實驗Lab2 report
RSA256解碼機

組別：team09
組員：鐘民憲(B06901017)
吳睿哲(B06901018)
謝兆和(B06901026)

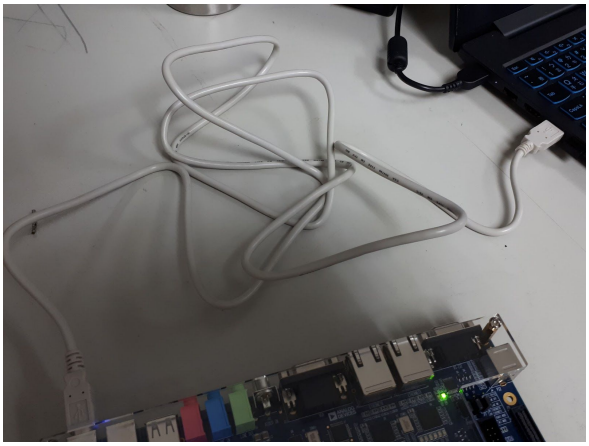
一、使用器材與架設方式



FPGA板



筆電



傳輸線

電源線

二、使用方式與詳細步驟

1.在電腦和FPAG之間連接rs232傳輸線
2.在電腦上用cmd執行rs232.py，便可將key及encrypt data傳輸給FPGA進行解密
3.按下FPGA上的reset鍵(key0)
4.rs232.py執行完畢後在電腦上會產生decrypt file
5.打開decrypt file即可看到decrypt data

三、實作設計技術細節及巧思

Hierarchy：
-Qsys //處理電腦與FPGA之間的訊號傳輸
-DE2_115 //設定FPGA的接腳
-Rsa256Wrapper //將key跟encrypt data傳給Core，再將decrypt data傳出
-Rsa256Core //實際執行解密的部分，包含ModuloProduct及Montgomery演算法
  -ModuloProduct //ab mod(N)
  -Montgomery_Algorithm //ab 2^{-256} mod(N)

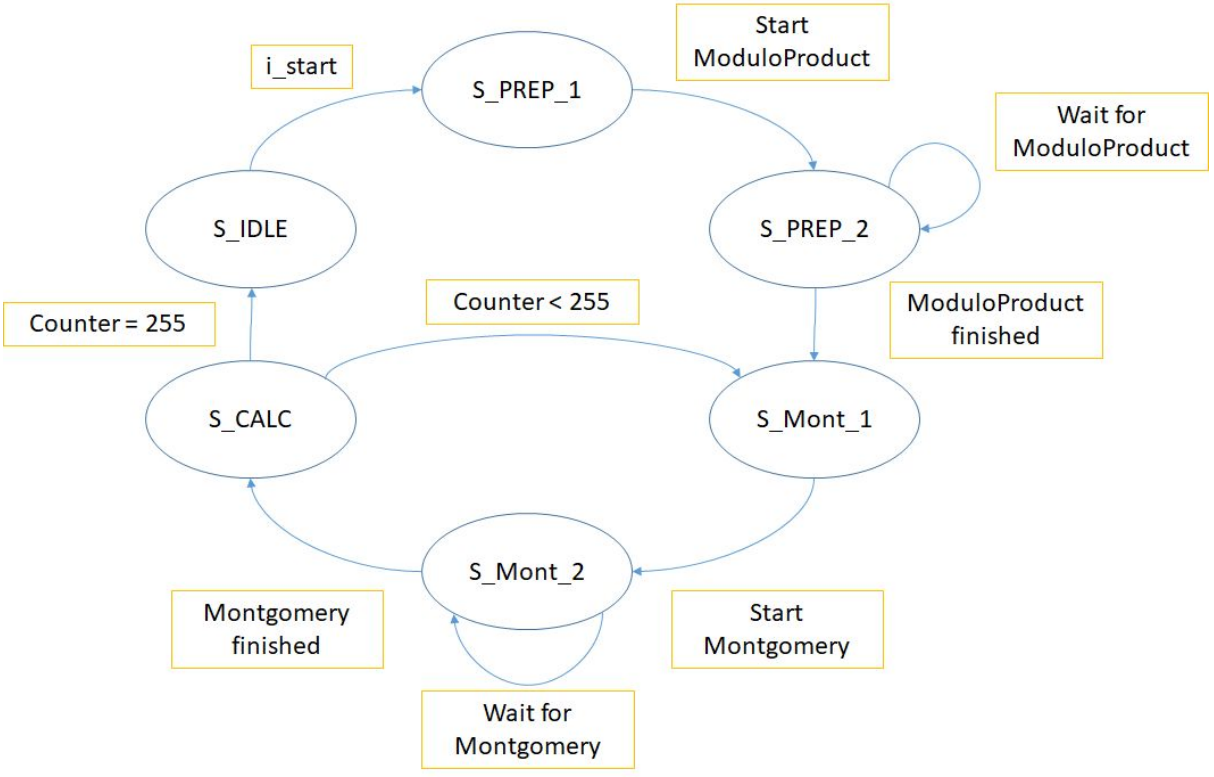
1.Rsa256Core

(1)Algorithm：

Algorithm 4 RSA256 with exponentiation by squaring and Montgomery algorithm

```
1: function RSA256MONT(N, y, d)
2:     t ← ModuloProduct(N, 2^256, y, 256)
3:     m ← 1
4:     for i ← 0 to 255 do
5:         if i-th bit of d is 1 then
6:             m ← MontgomeryAlgorithm(N, m, t)
7:         end if
8:         t ← MontgomeryAlgorithm(N, t, t)
9:     end for
10:    return m
11: end function
```

(2)Finite State Machine：



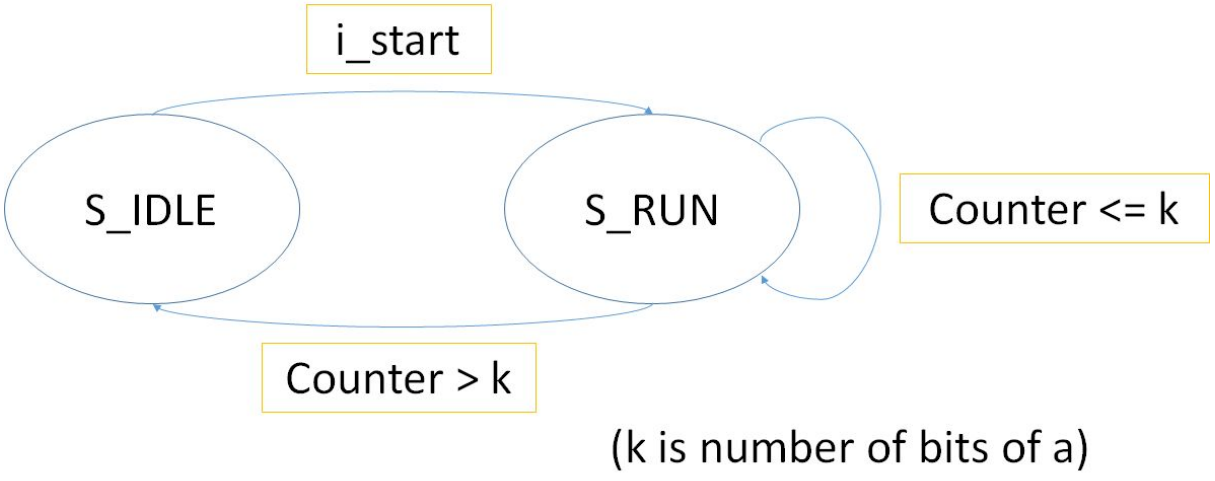
2.ModuloProduct

(1)Algorithm：

Algorithm 2 Modulo of products

```
1: function ModuloProduct(N, a, b, k)                ▷ k is number of bits of a
2:     t ← b
3:     m ← 0
4:     for i ← 0 to k do
5:         if i-th bit of a is 1 then
6:             if m + t ≥ N then
7:                 m ← m + t − N                     ▷ perform modulo operation in each iteration
8:             else
9:                 m ← m + t
10:            end if
11:        end if
12:        if t + t > N then
13:            t ← t + t − N                         ▷ perform modulo operation in each iteration
14:        else
15:            t ← t + t
16:        end if
17:    end for
18:    return m
19: end function
```

(2)FSM：



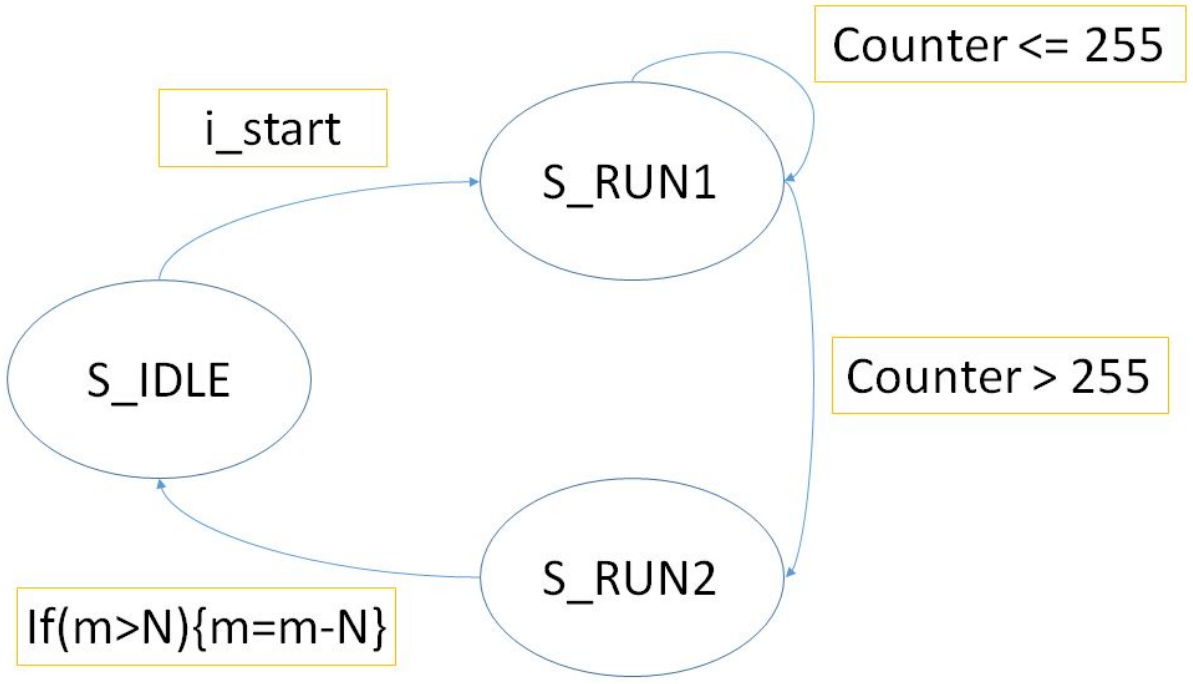
3.Montgomery_Algorithm

(1)Algorithm：

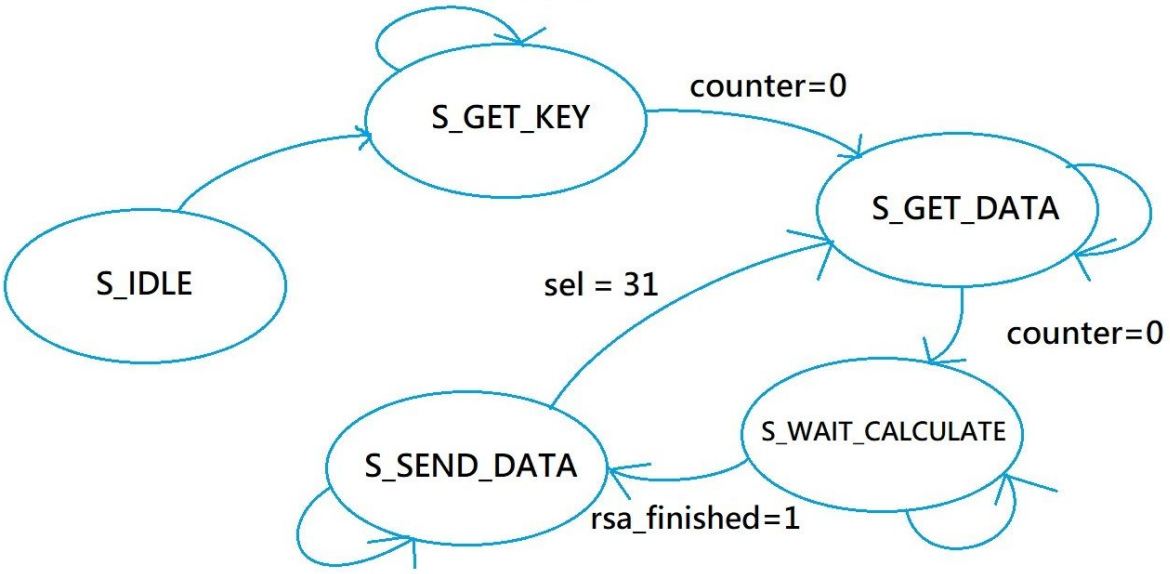
Algorithm 3 Montgomery algorithm for calculating $ab2^{-256} \bmod N$

```
1:  function MontgomeryAlgorithm(N, a, b)
2:      m ← 0
3:      for i ← 0 to 255 do
4:          if i-th bit of a is 1 then
5:              m ← m + b
6:          end if                          ▷ 4~6: replace multiplication with
                                               successive addition
7:          if m is odd then
8:              m ← m + N
9:          end if
10:         m ← m/2                         ▷ 7~10: calculate the modulo of a·2^-1
                                               → Montgomery reduction
11:     end for
12:     if m ≥ N then
13:         m ← m − N
14:     end if
15:     return m
16: end function
```

(2)FSM：



4.Rsa256Wrapper
FSM：

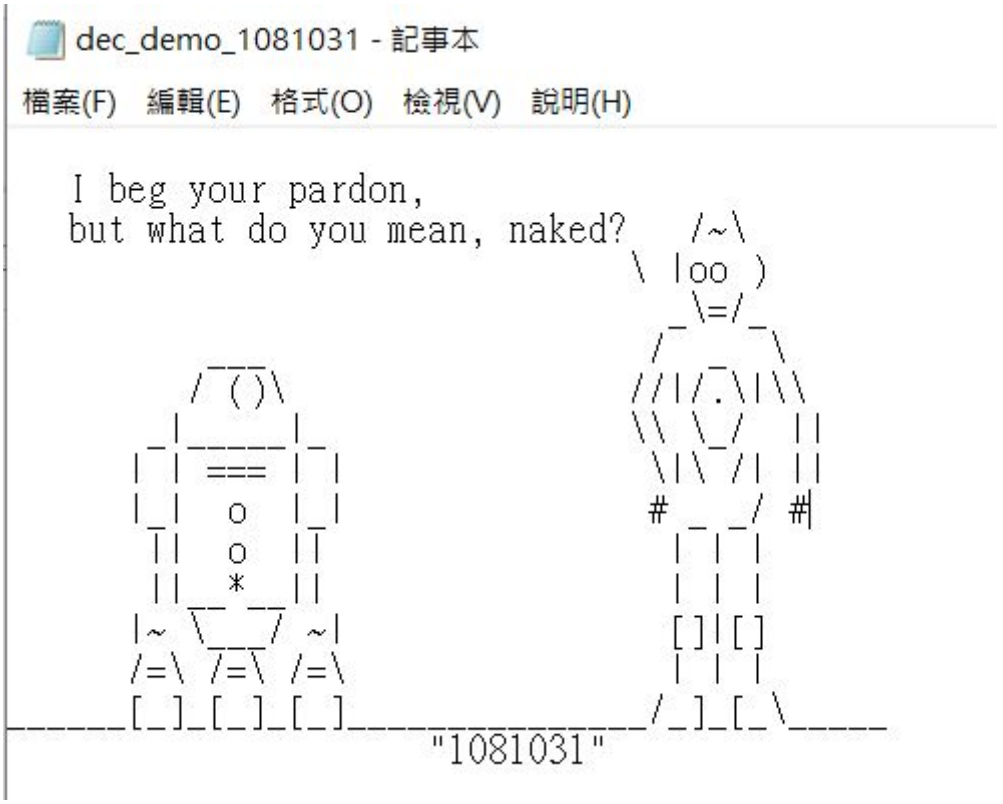


四、碰到的問題或挑戰與解決方式

(1)搞不清楚tb中一部分錯誤訊息的意思，像是simulation abort, simulation fuck, simulation suck……，而且每次出現相同error message，並不代表是相同的錯誤，往往必須藉由nWave追蹤訊號才能搞清楚原因。
(2)因為對測資輸入的格式理解有錯誤，沒有去掉00，也不知道key只有一開始輸入一次，導致wrapper的tb一直沒過，直到合作的另一位同學指出才意識到問題所在。
(3)在寫Rsa256Core時，最大的挑戰便是要如何對兩個submodules進行I/O，除了需要有個i_start去啟動submodule之外，同時也需要有個o_finished讓core知道submodule已經運算完畢，能執行下個動作。經過多次測試與nWave觀察後發現，submodule的o_finished signal最好只有在一個clk的時間內為1，其餘時間皆為0，也就是說只要state回到S_IDLE，便立刻將o_finished重置回0，而不是下次i_start後才設為0。雖然這樣core便只能夠在那個clk的時間內取得output data，但如果沒有這樣做，那麼第一次submodule執行完後，o_finished signal便會一直維持在1，下一次再次啟動submodule時，儘管i_start signal會將o_finished重置，但是會比寫在core中的判斷式：if(o_finished == 1)慢一個clk，因此core會以為submodule已經算完了，跳到下一個state，然而實際上submodule根本就沒有算，如此一來結果便會出錯。
(4)跑tb.sv最初出現的error是simulation abort，原因便是第三點提到的，我沒有將o_finished重設回0，而core裡面的寫法又是：if(!o_finished){i_start=1}，所以就永遠不會進入S_CALC，導致counter不會跳。Debug後成功在時限內跑完，但是跑出來的dec跟gold不一樣，代表解碼錯誤，可是我有自己寫簡單的testbenches分別去測試兩個submodules都沒有問題，後來才發現原因有二：bit overflow以及montgemery判斷式寫錯。自己寫的testbench test data並不會造成overflow，導致我忽略了這個可能性，後來將submodules裡面的register m跟t bit數增加以解決這個問題。bit overflow解決

後發現答案仍然不對，於是再次檢查程式碼發現某個if粗心寫錯，一個小錯誤浪費了超多時間，不過最後能看到testbench出現congratulation便覺得努力沒有白費。
(5)儘管tb.sv跟testwrapper.sv都通過了，然而將Qsys及DE2-115完成，實際燒錄進板子後，執行python卻沒有任何反應，產生的dec.bin是空的，這使得我們百思不得其解，不知道問題出在哪個環節，我們試著重建Qsys，也發現了reset clock在每個module中並不同步，但是FPGA顯示器上仍然是亂碼。最後，感謝同組中某個組員在demo前一晚睡實驗室的付出，發現在wrapper.sv中，沒有read跟write的時候需要將signal reset成0，修正之後才終於成功!!!

五、Demo



https://www.youtube.com/watch?v=5vnAW1XDdyw