LAB 2 Report

Team 4

B08901006 蔡亞辰 / B08901074 尤韻嘉 / B08901174 郭尚睿

Hierarchy

→ DE2_115.sv

→ rsa_qsys.v

→ Rsa256Wrapper.sv

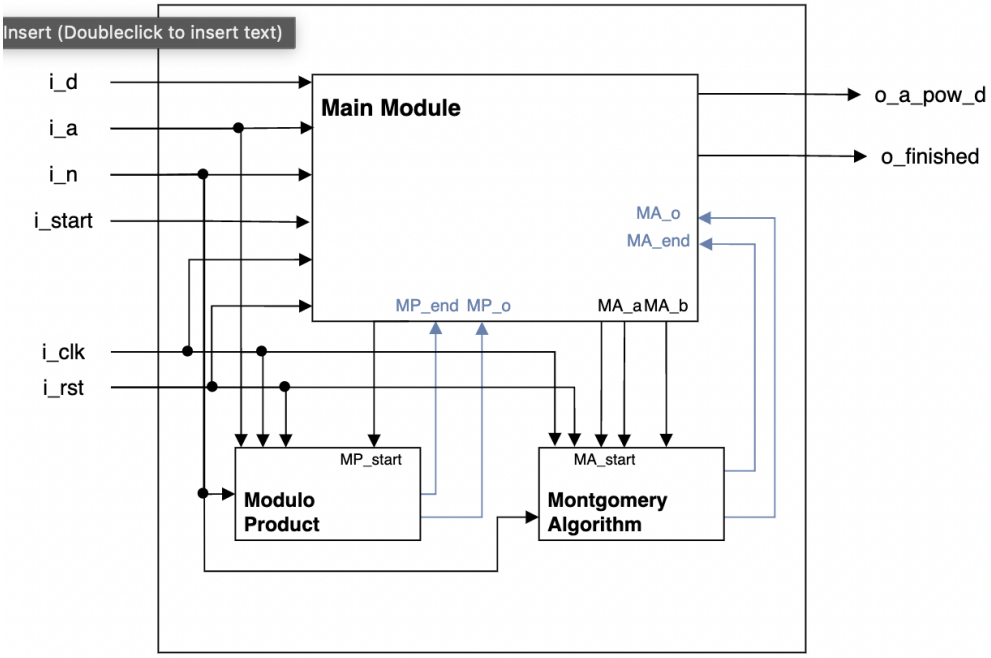
→ Rsa256Core.sv

→ ModProd (modulo product)

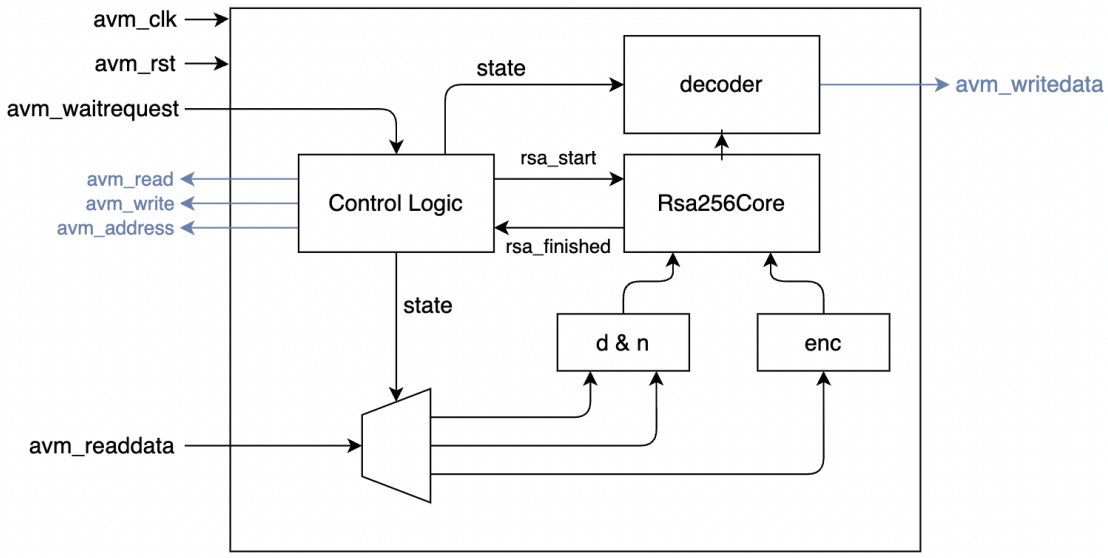
→ MontAlg (montgomery algorithm)

Block Diagram

[Core]

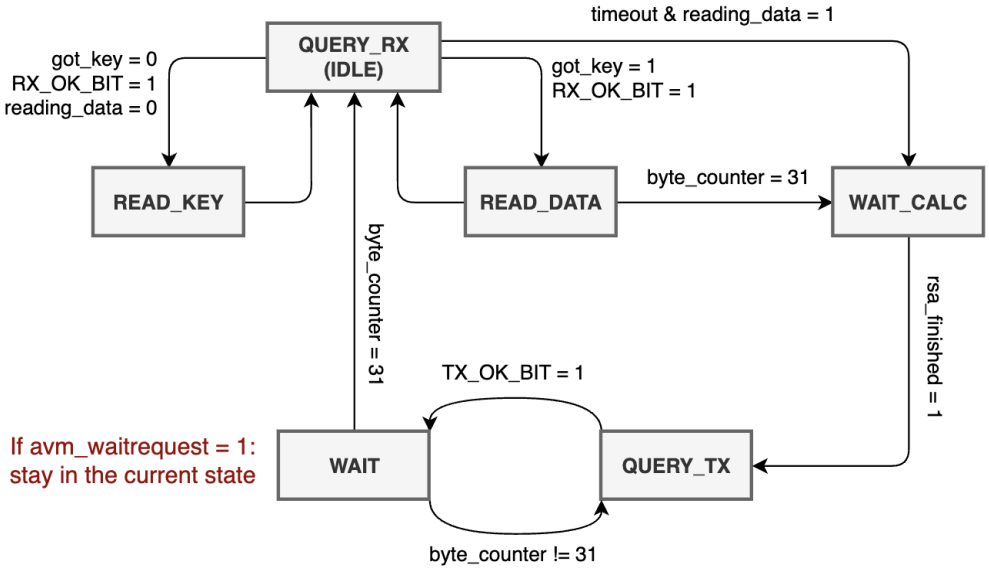


[Wrapper]

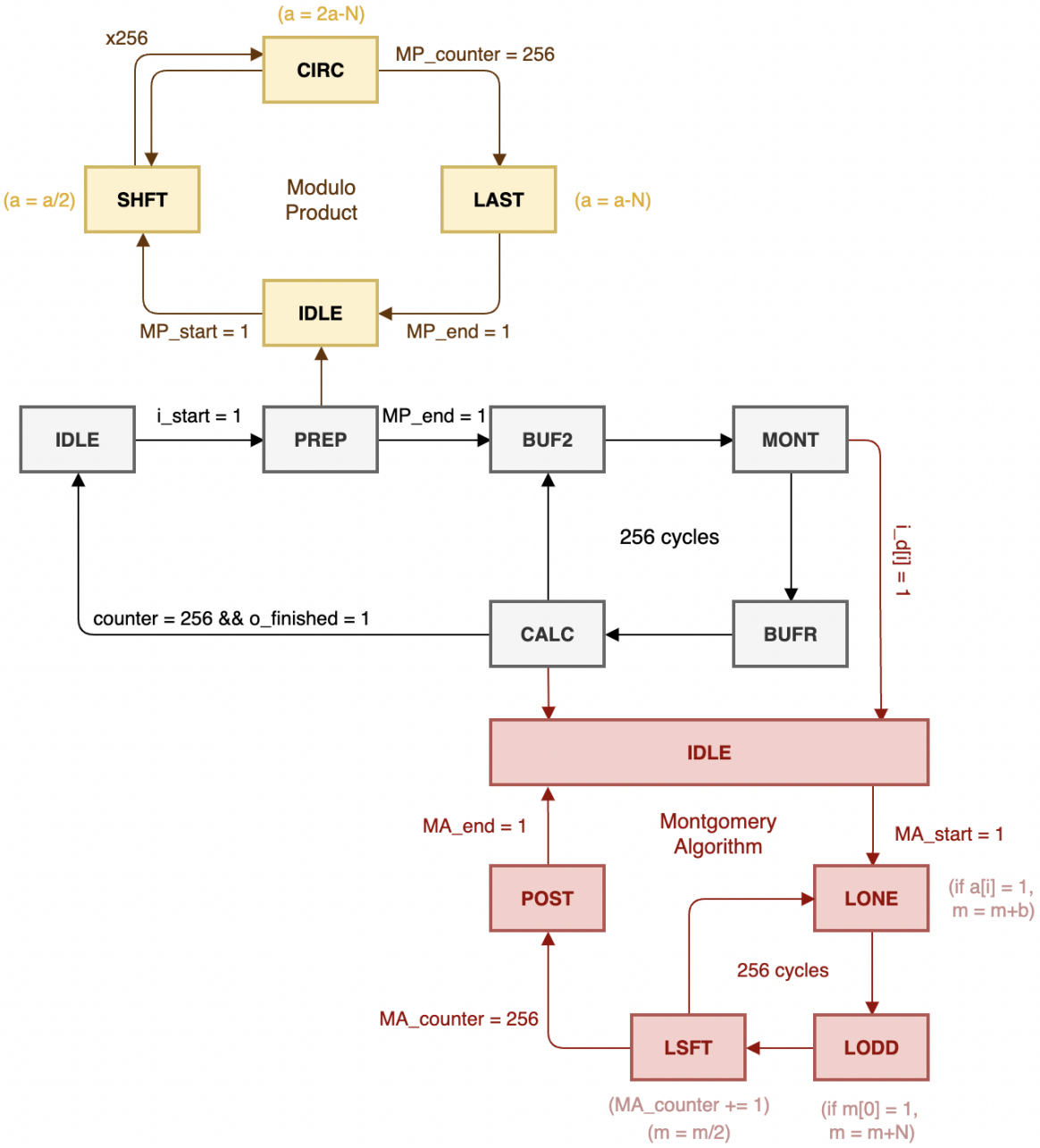


FSM

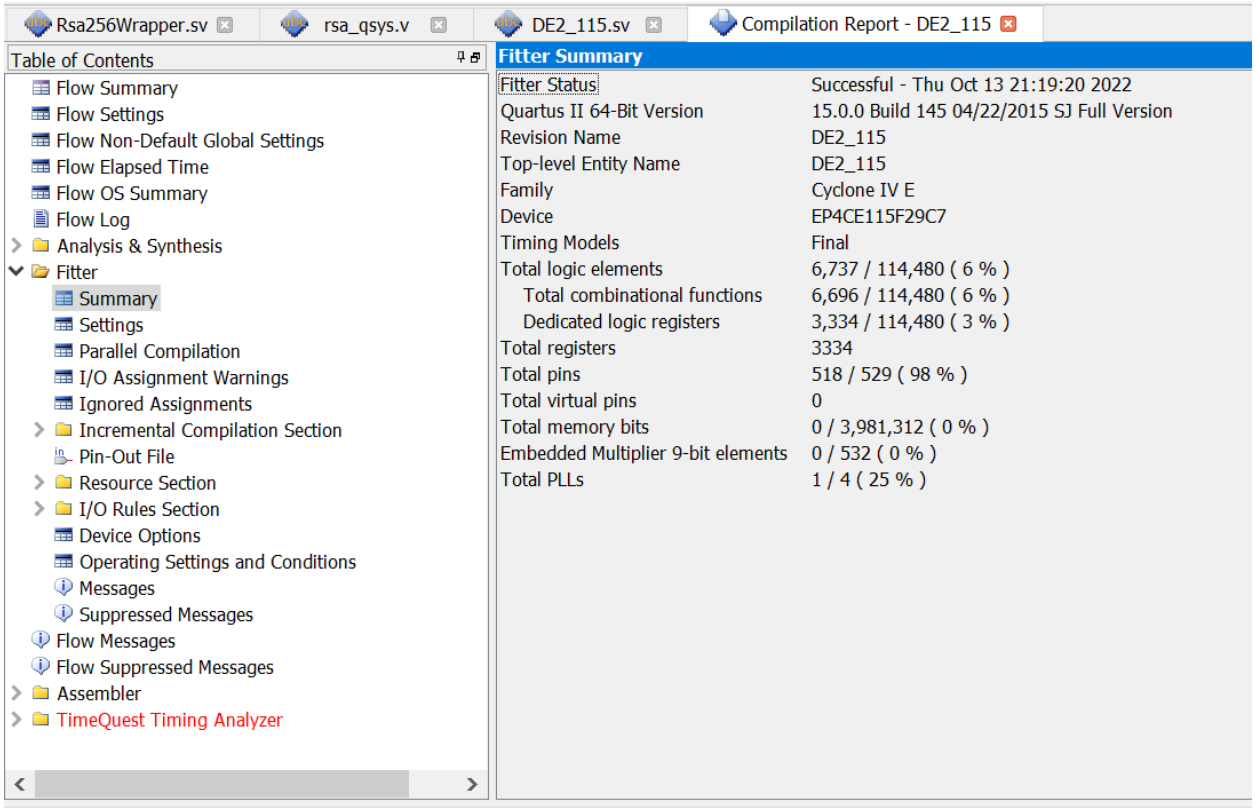
[Wrapper]



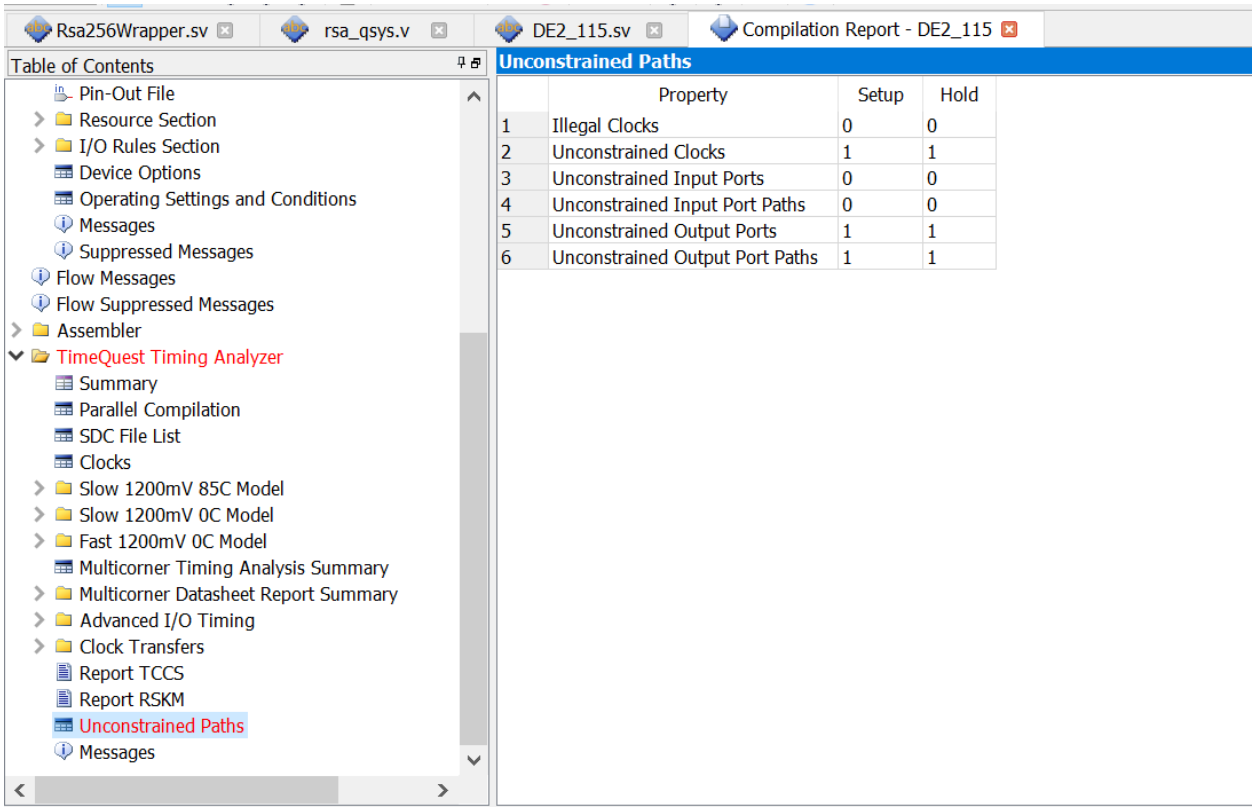
[Core]


(a = 2a-N)
x256
CIRC
MP_counter = 256
(a = a/2)
SHFT
Modulo
Product
LAST
(a = a-N)
IDLE
MP_start = 1
MP_end = 1
IDLE
i_start = 1
PREP
MP_end = 1
BUF2
MONT
i_d[i] = 1
256 cycles
counter = 256 && o_finished = 1
CALC
BUFR
IDLE
MA_end = 1
Montgomery
Algorithm
MA_start = 1
POST
LONE
(if a[i] = 1,
m = m+b)
256 cycles
MA_counter = 256
LSFT
LODD
(MA_counter += 1)
(m = m/2)
(if m[0] = 1,
m = m+N)

Fitter Summary



Timing Analyzer



Difficulties & Solutions

- [Core] 在MONT state時把MA_start拉起來等於1，但整段時間MA_start都會等於1（理想上是想要他等於1一個clk cycle後就掉下來）；這會導致MA_start = 1延續到整個CALC state，Montgomery Algorithm input (a, b) 還沒assign好，MA的小module就會開始跑，而導致錯誤的計算結果：
  - 在MONT跟CALC前各加一個buffer state，準備input以及使MA_start短暫變成0
- [Core] 計算時發現overflow：
  - 把output 以及計算時用到的變數開到270 bits
- [Core] 把output開到270 bits 會不容易跟wrapper對接（應該要是256 bits）：
  - 因為雖然在計算過程中可能overflow，但最終的output應該不會overflow，所以我們另開一個270-bit register (o_a_pow_d_r)，並assign final_output = o_a_pow_d_r [255:0]
- [Core] 一開始想跟Lab1一樣，只靠counter跟FSM實作，但這樣Code變得很複雜也難以Debug：
  - 所以改以把各個function實作成小module，靠i_start跟o_finish跟大module溝通

- [Wrapper] 如果收到的data長度不是256 bytes的整數的話，在跑python時會卡在FSM中，無法回到end state 或IDLE state：
    - 加一個判斷式，若大於一段時間沒有收到下一筆data的話，就自動回到IDLE

Bonus!