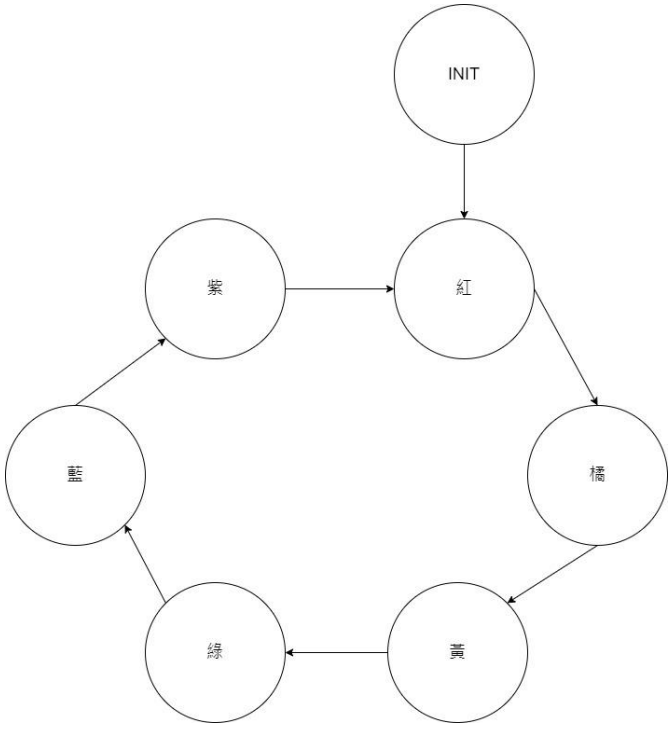
FPGA DESIGN

Homework 3
Report

	姓名	學號
組員 1	郭家佑	F64096114
組員 2	蔡奇佑	E24096213
組員 3	趙泓瑞	E14071025

1. Problem1

- Design introduction
  - Algorithm

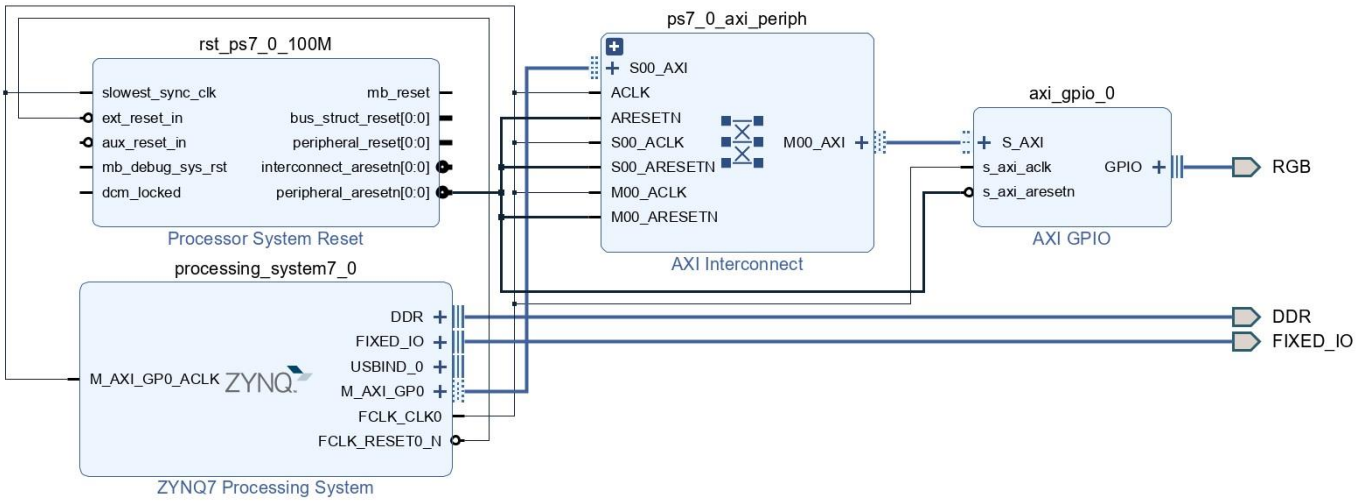


初始完 rgb_data 和 RGB_Status，顏色與顏色之間的 Hex Triplet 差，切 2560000 等分以 rgb_data 用 for loop 累進達到 rgb 混色的效果，顏色與顏色之間用 while(1)連續下去。其中較特別的橙色與紫色是利用 mod 來控制顏色比例。

  - Interface between software and hardware

使用 ZYNQ7 Processing System IP 建立 block design，再透過 GPIO 模組 Vitis IDE 中的 led.c 互動。

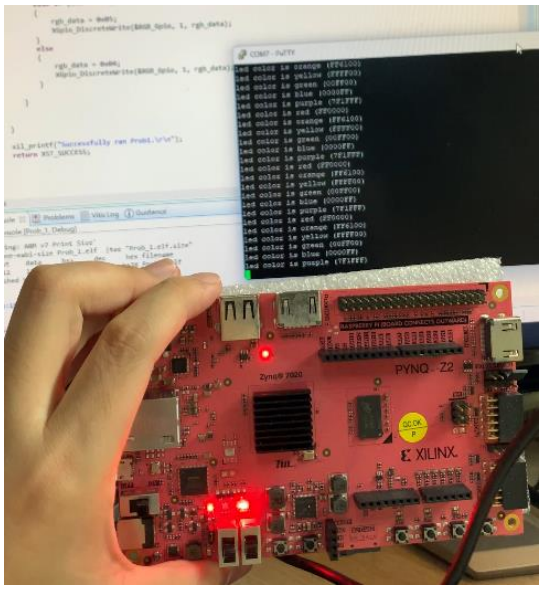
- Block Design

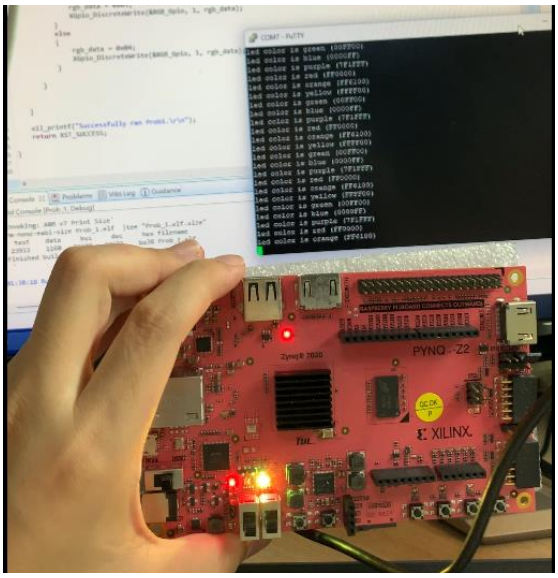


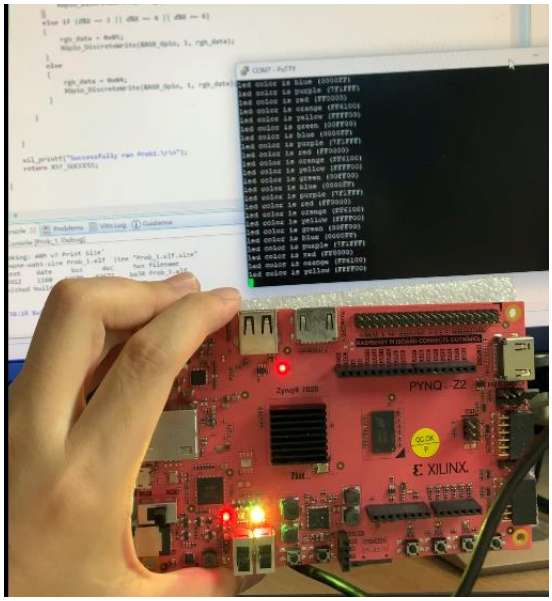
- Simulation result

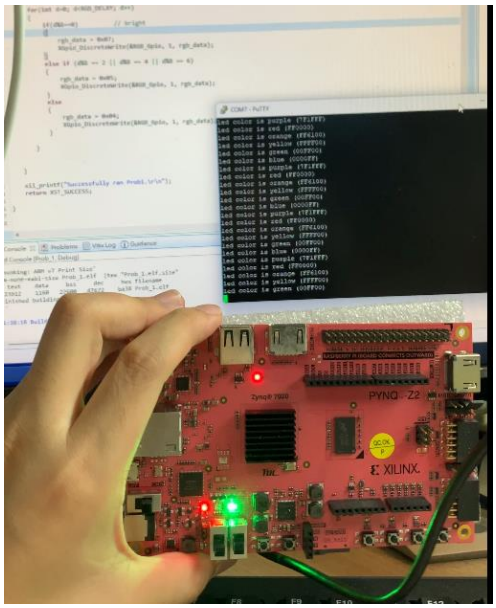
如下影片中所示，依序顯示演算法中各色於 PUTTY，每 RGB_DELAY(2560000ns)換顏色。XGpio_DiscreteWrite()函數與板子的 GPIO 溝通，使過程隨 d 混色逐步變成下一顏色，並修正明暗的比例。

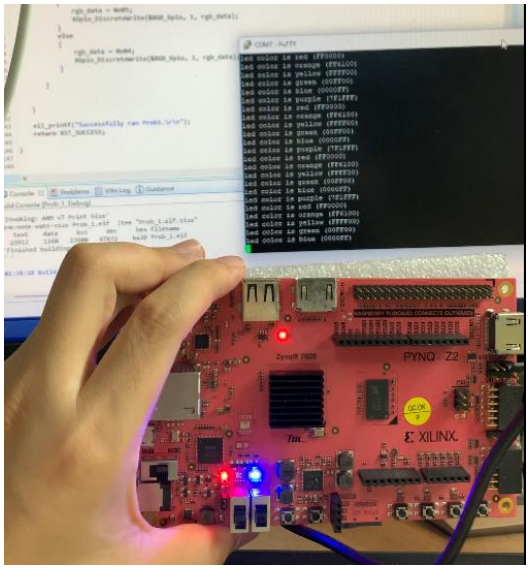
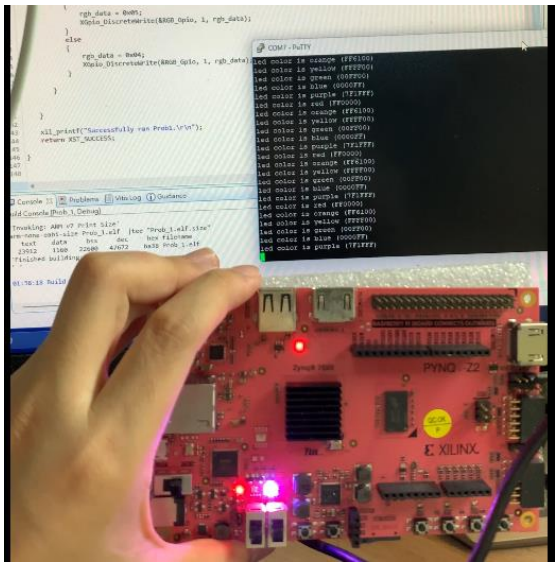
```
led color is red (FF0000)
led color is orange (FF6100)
led color is yellow (FFFF00)
led color is green (00FF00)
led color is blue (0000FF)
led color is purple (7F1FFF)
led color is red (FF0000)
led color is orange (FF6100)
led color is yellow (FFFF00)
led color is green (00FF00)
led color is blue (0000FF)
led color is purple (7F1FFF)
led color is red (FF0000)
led color is orange (FF6100)
led color is yellow (FFFF00)
```









詳細操作影片可參考:

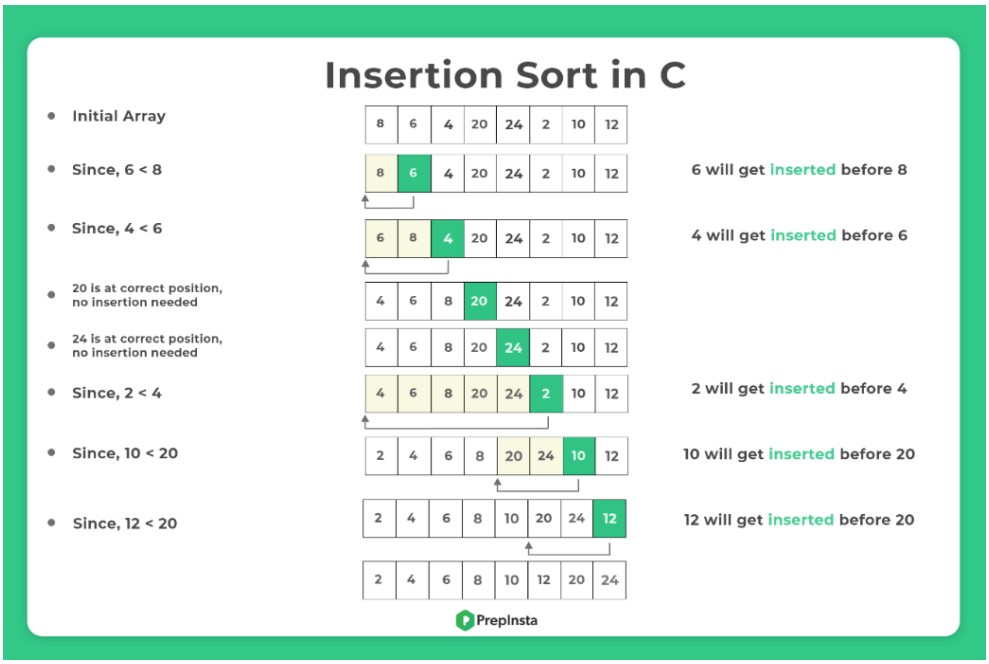
https://drive.google.com/file/d/1QJvmil-omPVpnWusTpNbNTsGttnaSVGj/view?usp=share_link

https://drive.google.com/file/d/1ExhjwlBhNiODbeCPBBXBCUwLSVgpjhlE/view?usp=share_link

2. Problem2

- Design introduction
  - Algorithm

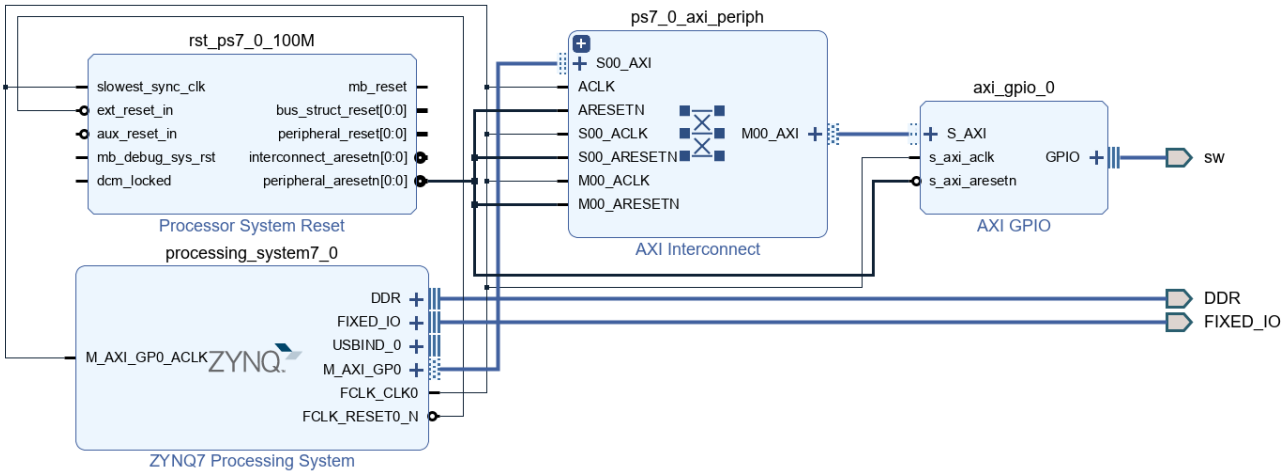
    採用插入排序法(Insertion Sort)，逐一將原始資料加入已排序好資料中，並逐一與已排序好的資料作比較，找到對的位置插入。



- Interface between software and hardware

  使用 ZYNQ7 Processing System IP 建立 block design，隨後由 programmable logic 執行的所以需產生 bitstream 將 AXI GPIO 燒錄到 FPGA 上。利用 Vitis IDE 和軟體 sort.c 互動。依據 sort.c 的 sw_data 與 GPIO 互動達到選擇由大到小還是由小到大。

- Block Design

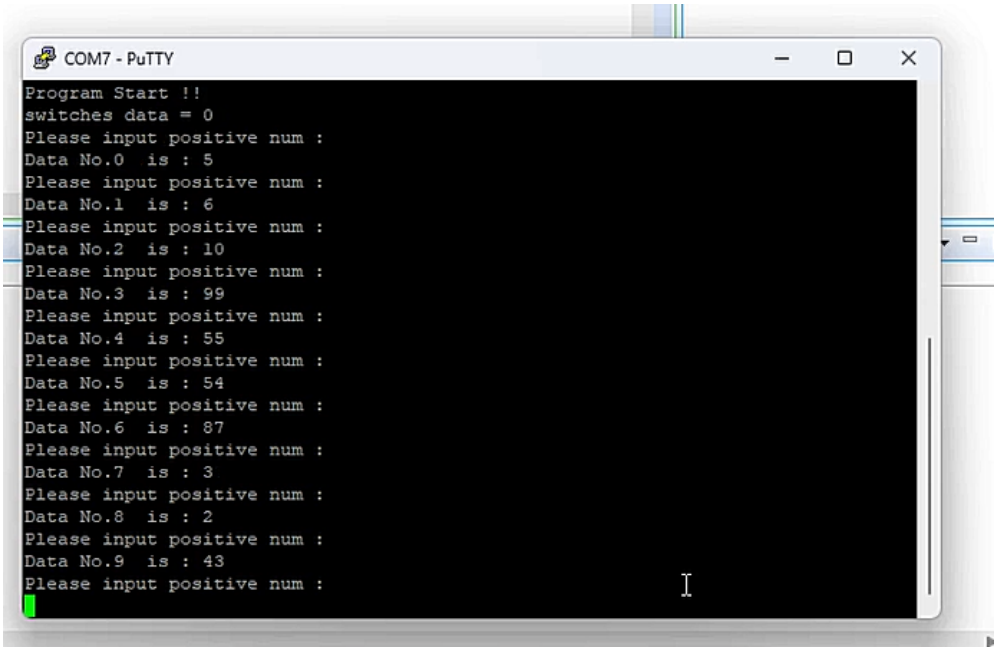


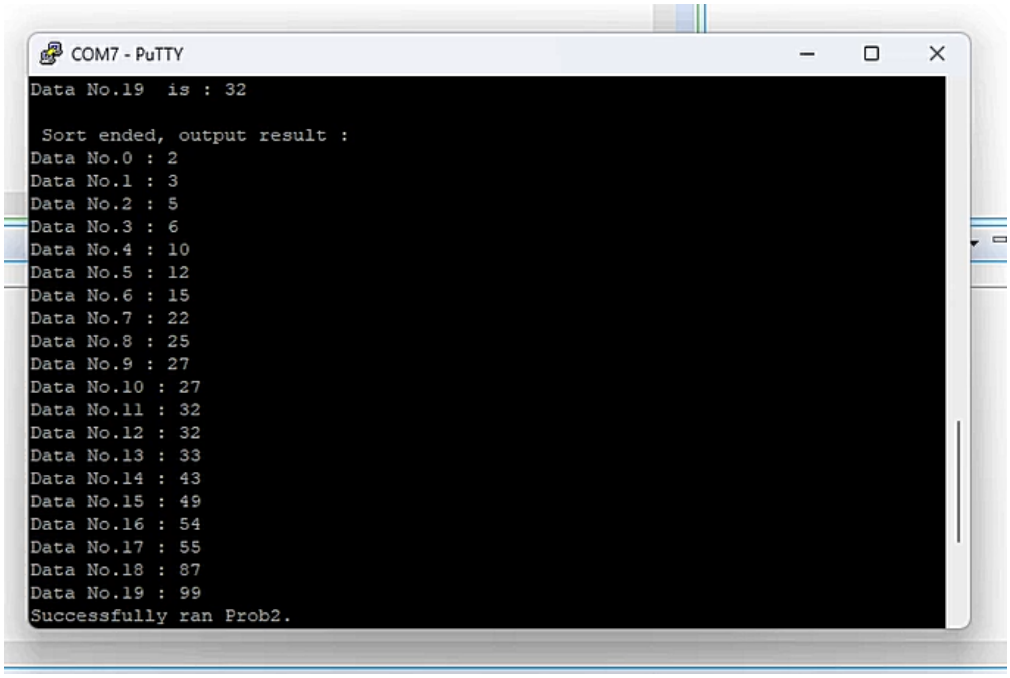
- Simulation result

詳細操作影片可參考:

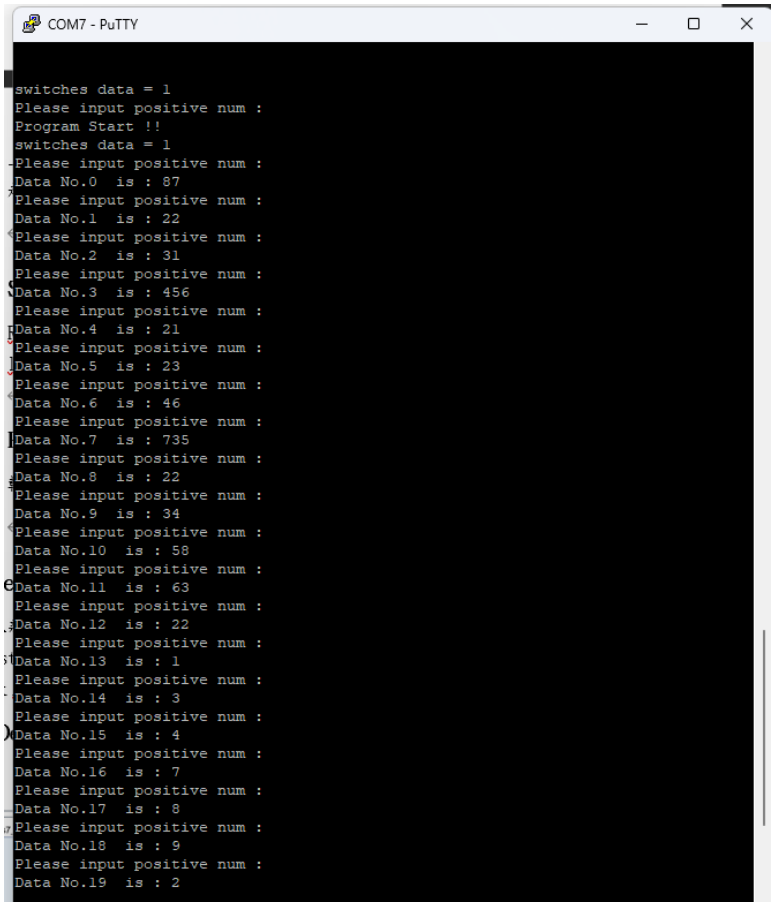
首先輸入 1/0 選擇由大到小孩是由小到大，隨後輸入 20 個正整數，隨後進行插入排序。

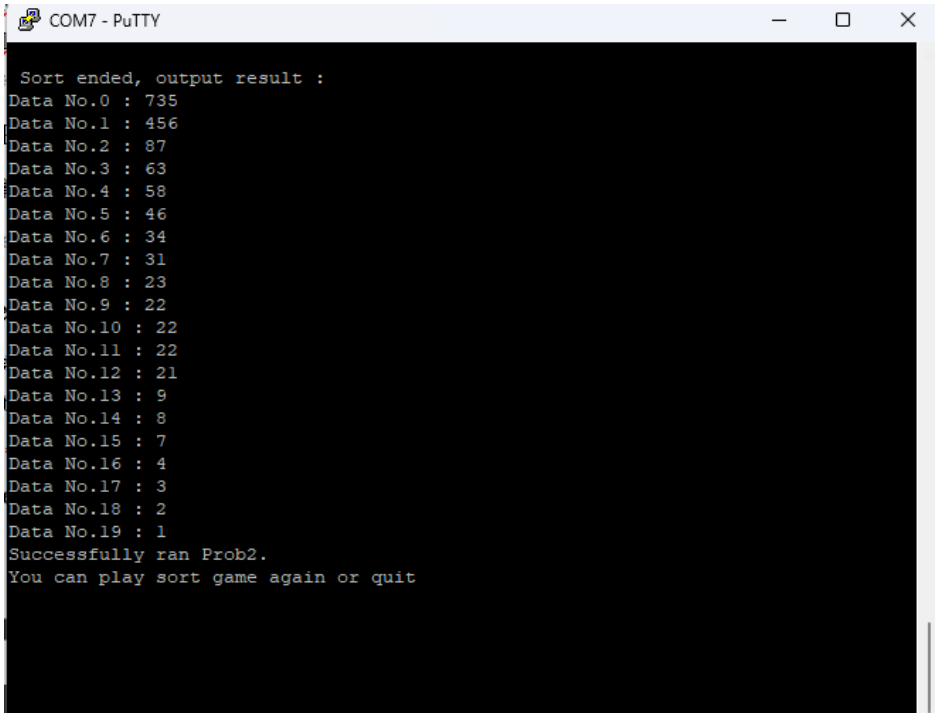
- Sw=0(小到大)




COM7 - PuTTY
Data No.19 is : 32
Sort ended, output result :
Data No.0 : 2
Data No.1 : 3
Data No.2 : 5
Data No.3 : 6
Data No.4 : 10
Data No.5 : 12
Data No.6 : 15
Data No.7 : 22
Data No.8 : 25
Data No.9 : 27
Data No.10 : 27
Data No.11 : 32
Data No.12 : 32
Data No.13 : 33
Data No.14 : 43
Data No.15 : 49
Data No.16 : 54
Data No.17 : 55
Data No.18 : 87
Data No.19 : 99
Successfully ran Prob2.

■ Sw=1(大到小)

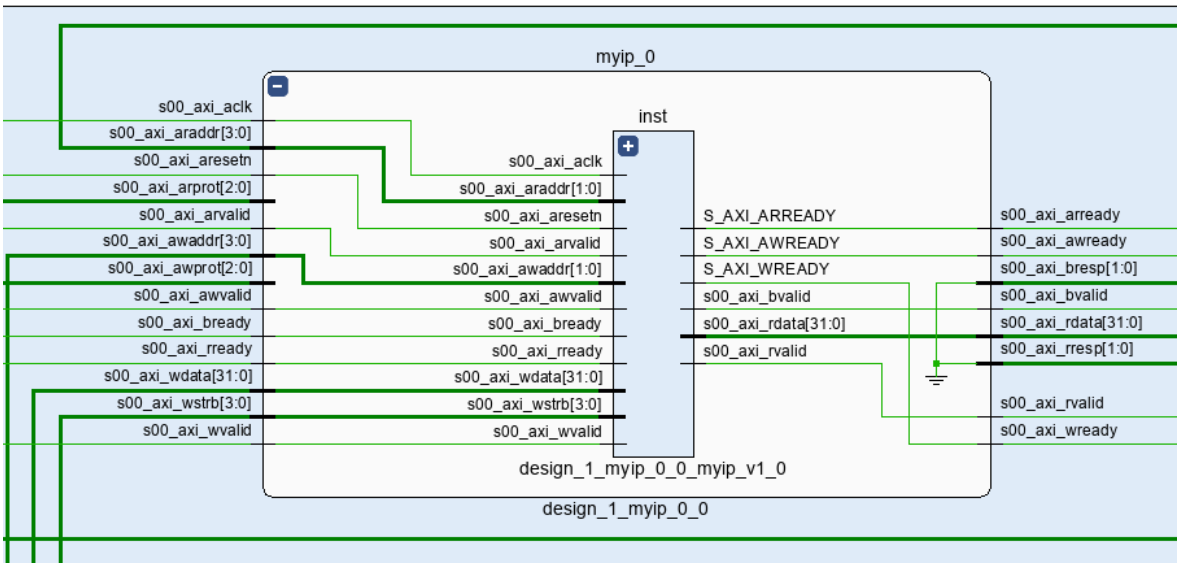

COM7 - PuTTY
switches data = 1
Please input positive num :
Program Start !!
switches data = 1
Please input positive num :
Data No.0 is : 87
Please input positive num :
Data No.1 is : 22
Please input positive num :
Data No.2 is : 31
Please input positive num :
Data No.3 is : 456
Please input positive num :
Data No.4 is : 21
Please input positive num :
Data No.5 is : 23
Please input positive num :
Data No.6 is : 46
Please input positive num :
Data No.7 is : 735
Please input positive num :
Data No.8 is : 22
Please input positive num :
Data No.9 is : 34
Please input positive num :
Data No.10 is : 58
Please input positive num :
Data No.11 is : 63
Please input positive num :
Data No.12 is : 22
Please input positive num :
Data No.13 is : 1
Please input positive num :
Data No.14 is : 3
Please input positive num :
Data No.15 is : 4
Please input positive num :
Data No.16 is : 7
Please input positive num :
Data No.17 is : 8
Please input positive num :
Data No.18 is : 9
Please input positive num :
Data No.19 is : 2



https://drive.google.com/file/d/1N13vBP4fy_E13s94VghvcDJT71-35nX4/view?usp=share_link

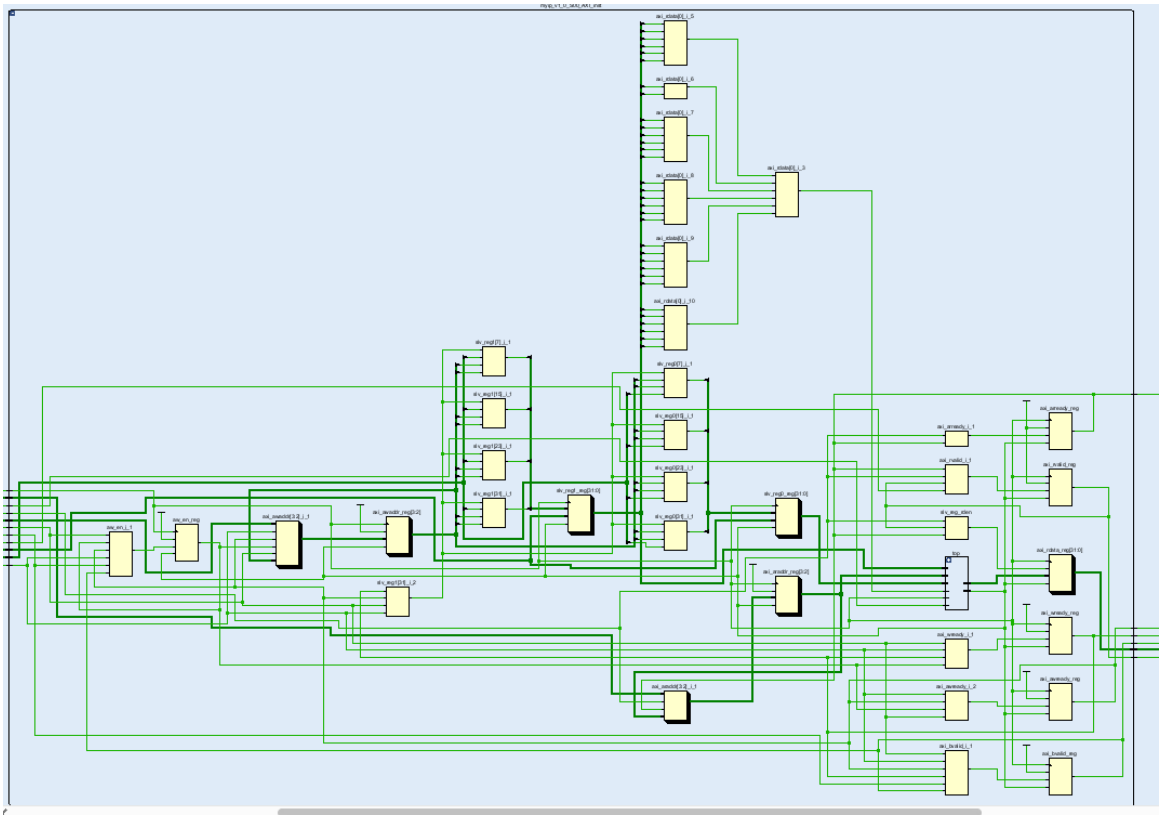
3. Problem3

- Design introduction
  - Architecture (HDL)
    - My_IP



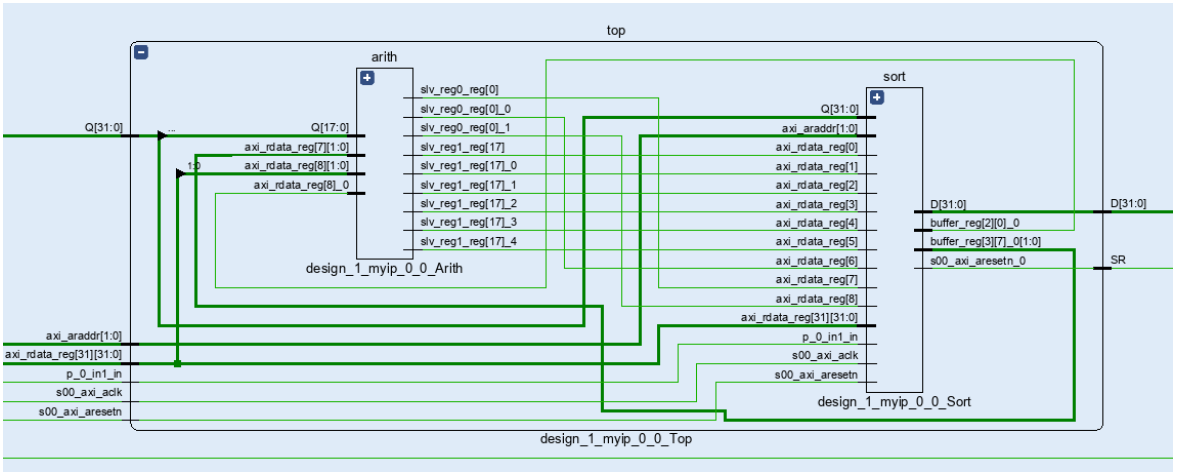
裡面建一個 my_ip 的 instance

- AXI



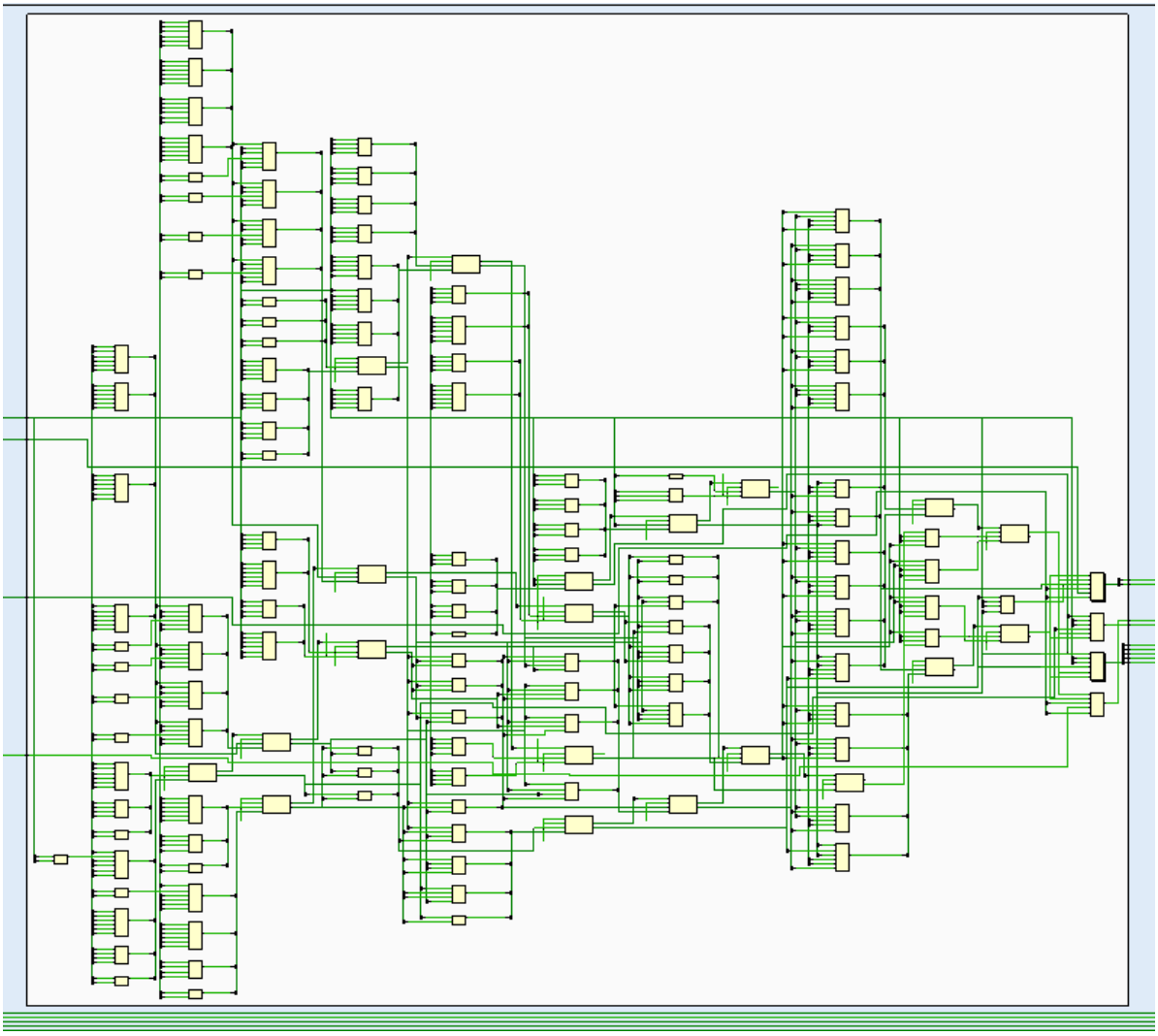
裡面包含了 AXI 的基本架構之外還有自己設計的 Top module

- Top



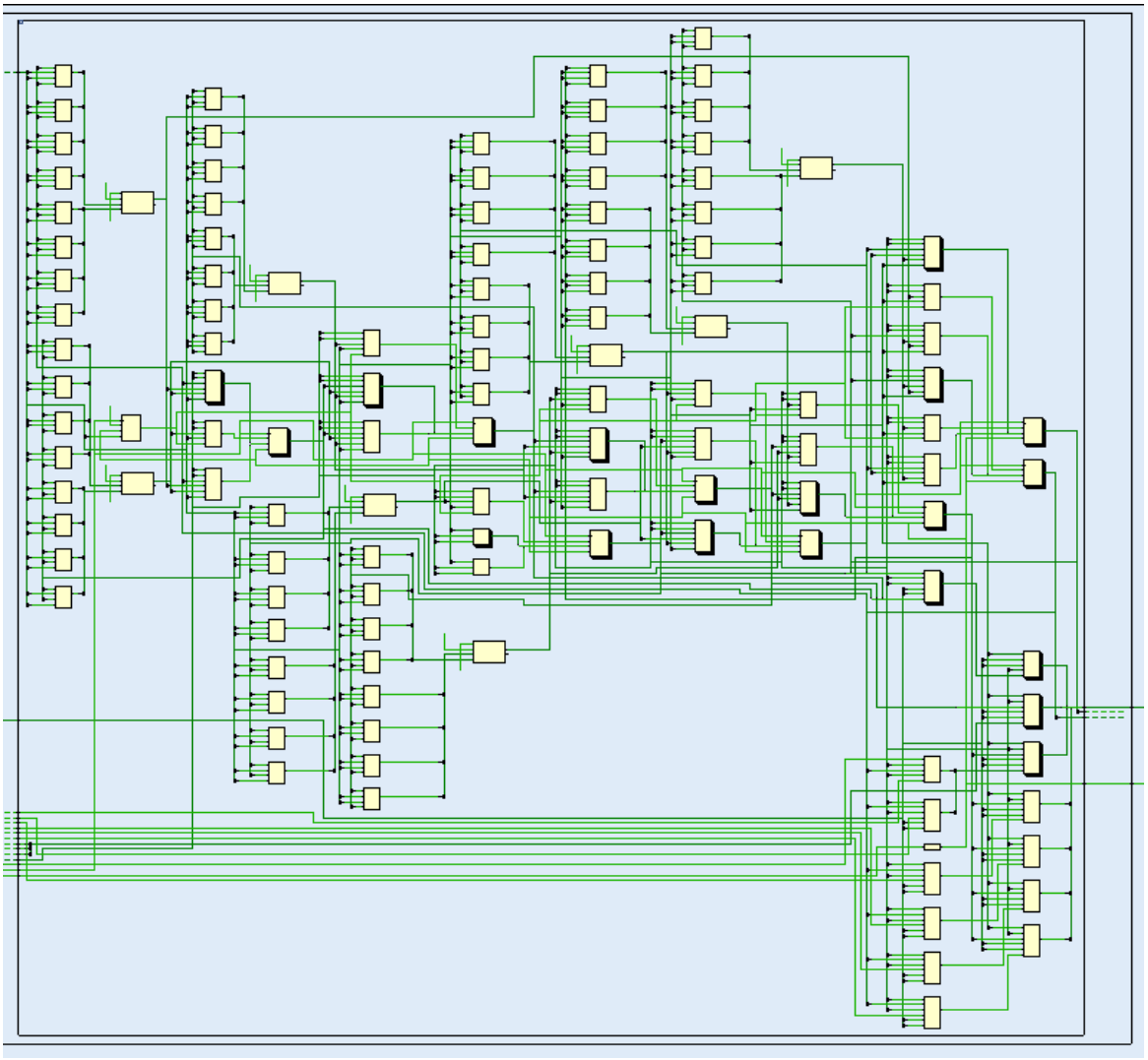
Top.v 將輸入 in_data 接上 Arith.v Sort.v 和 Parity.v，output reg 有 slv_reg2 和 slv_reg3，其中 slv_reg2 是一般的三種 mode 的 output data(mode0 的 sum+ov; mode1 的 small1~small4; mode2 的 paratibit)，而 slv_reg3 則是 mode2 的 small5~8。此外，mode0 可以做加減乘運算並且上下限分別為 $signed(16'd127)和$signed(-16'd128)，超過會 overflow。Mode1 則執行 insertionsort 運算，Mode2 則執行 parity 運算。slv_reg3 則穩定做執行 sort 運算後 4 個 data。

◆ Arith



輸入 signed op1 和 op2，隨 op 不同做加減乘運算存入 signed temp_result 中，上下限分別為$signed(16'd127)和$signed(-16'd128)，超過會 overflow(ov)提起。提起則輸出歸零。

◆ Sort



Rst 或 clear 後 buffer 為 8'd255，buffer 的輸入 in_data 過程中同時用 larger_equal_than 做 sort。隨後組合電路輸出給 small_0~small_7。

其中最困難的就是當切換到 model 時要 reset 內部的 counter，所以我們會先在 myip.c 傳入一個訊號給 top，top 再傳入一個 clear 訊號給 sort ckt 來清空 buffer 和 reset counter。

但又會遇到外部(main.c)執行的時間比內部的 clk 慢很多，所以變成說可能執行一行 c code，電路內部已經跑了 100 個 cycle(insert 入 100 個同一個值)，因此再用一個 en + en_reg 訊號來擋掉。

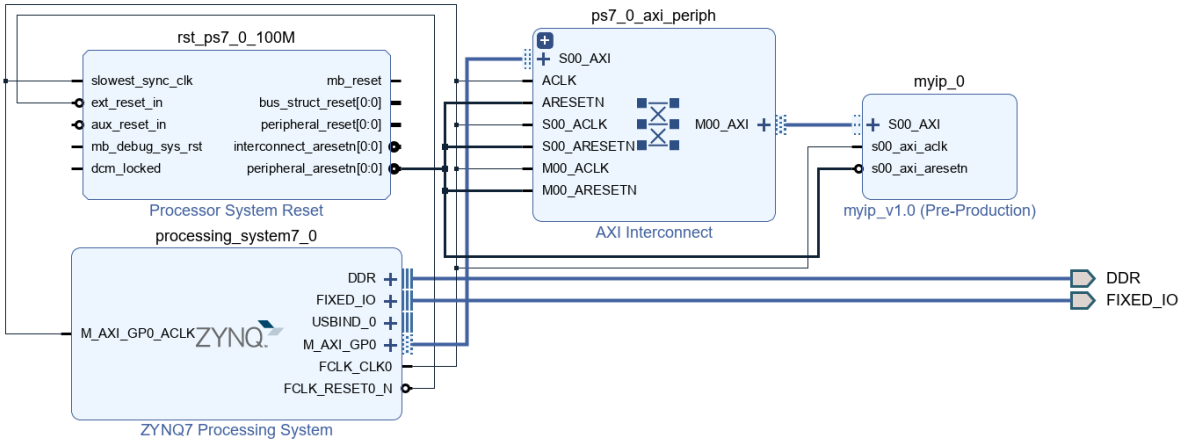
◆ Parity

利用 XOR 輸出 Parity 為確認 32bit 輸入 pattern 是偶數還是奇數個 1。

■ Interface between software and hardware

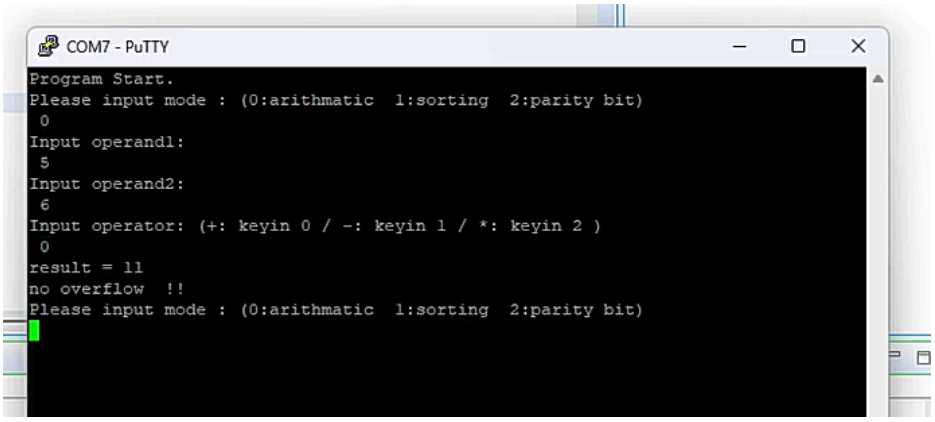
建立新的 IP 連接 AXI Interface Register 然後設計為 block design。產生 Bitstream 和 HDL Wrapper 後，利用 Vitis IDE 和軟體 main.c 互動，可以輸入 mode 選擇做 Arith 還是做 Sort 或 Parity。

- Block Design

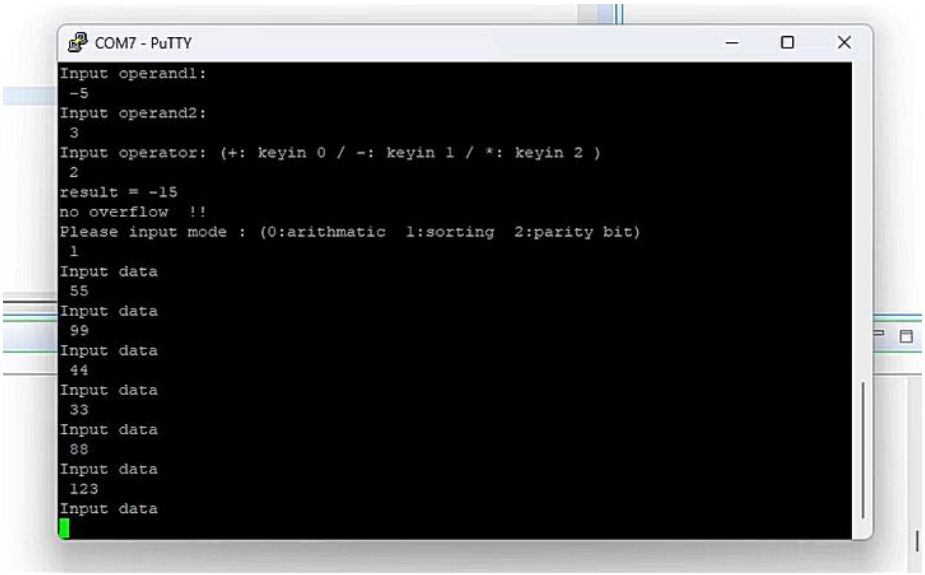


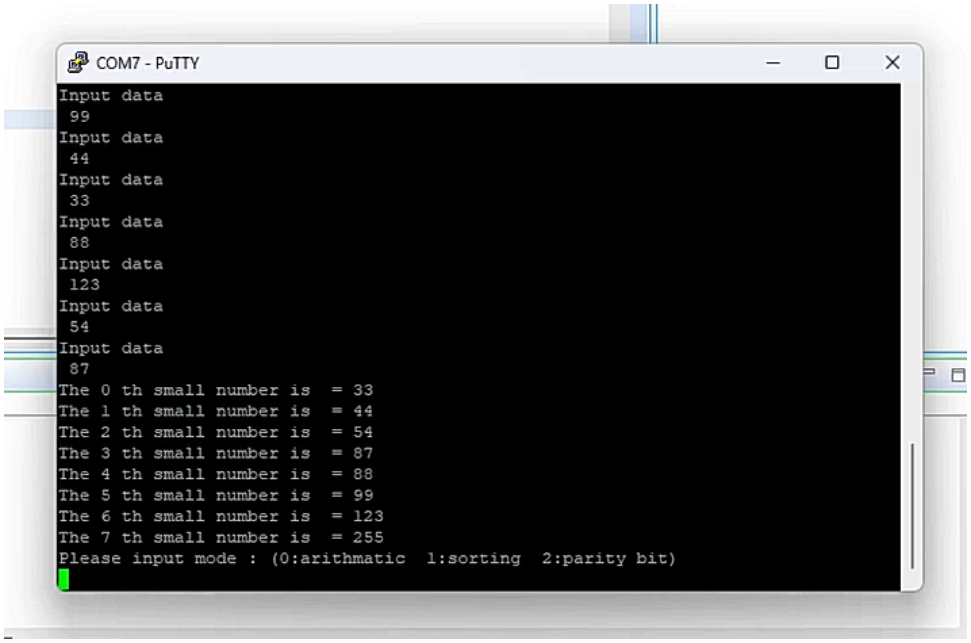
- Simulation result

可以從 putty 上選擇 mode，mode0:

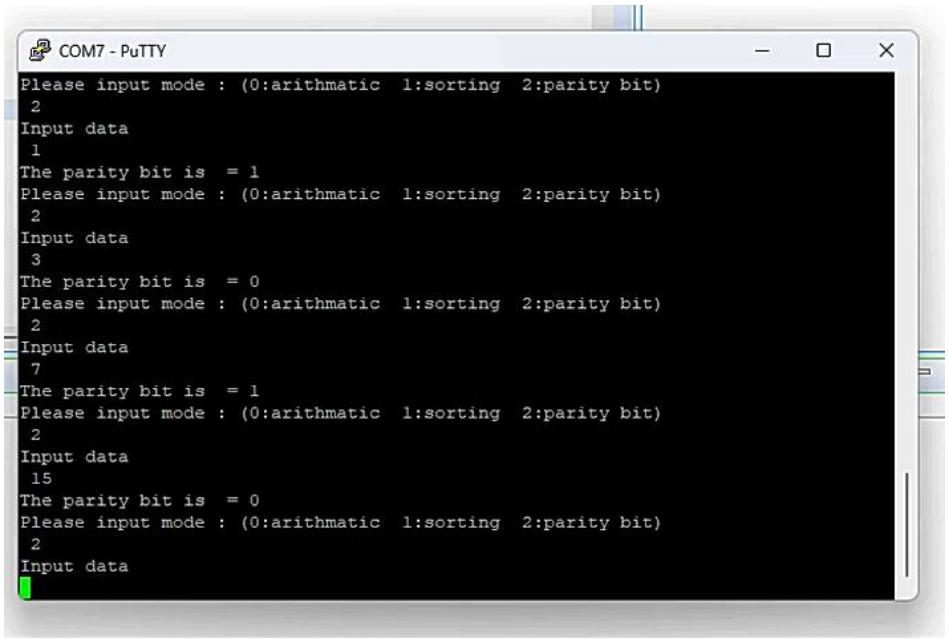


Mode1:





Mode2:



詳細操作影片可參考:

https://drive.google.com/file/d/1U9ECgpxSt6_WBtiXoG95SuvfccFV6Gf_/view?usp=share_link