VLSI System Design (Graduate Level)

Fall 2021

HOMEWORK IV

REPORT

Must do self-checking before submission:

Compress all files described in the problem into one tar

All SystemVerilog files can be compiled under SoC Lab environment

All port declarations comply with I/O port specifications

Organize files according to File Hierarchy Requirement

No any waveform files in deliverables

Student name: 楊承翰

Student ID: NE6091124

Summary:

1.完成sensnor\_ctrl\_wrapper

2.完成CSR module

3.完成APR

4.完成Prog0、Prog1、Prog2的RTL simulation

5.完成Prog0、Prog1、Prog2的Gate-Level simulation

6.完成Prog0、Prog1、Prog2的Post-layout simulation

Lesson learn:

1.了解CSR指令在RISC-V CPU的運作方式

2.知道當發生Interrupt時CPU應該要如何處理

3.學會用Innovus這套tool來進行APR並完成DRC、LVS & Antenna等驗證

Simulation Results:

RTL: Gate-level: Post-layout:

Program0:

一張含有 文字 的圖片

自動產生的描述 一張含有 文字 的圖片

自動產生的描述 一張含有 文字 的圖片

自動產生的描述

Program1:

一張含有 文字 的圖片

自動產生的描述 一張含有 文字 的圖片

自動產生的描述 一張含有 文字 的圖片

自動產生的描述

Program2:

一張含有 文字 的圖片

自動產生的描述 一張含有 文字 的圖片

自動產生的描述 一張含有 文字 的圖片

自動產生的描述

Area report:

一張含有 文字 的圖片

自動產生的描述

Timing report: cycle = 20.0

一張含有 文字 的圖片

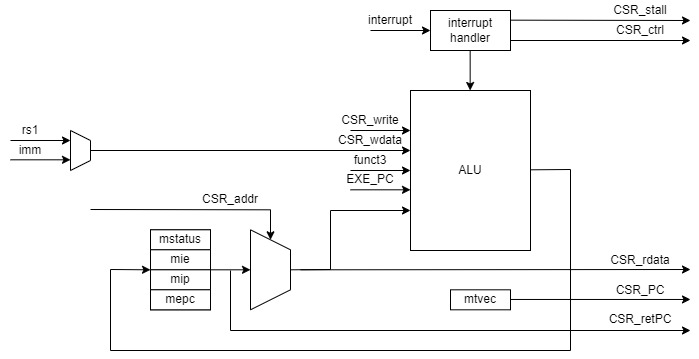
自動產生的描述

一張含有 文字 的圖片

自動產生的描述

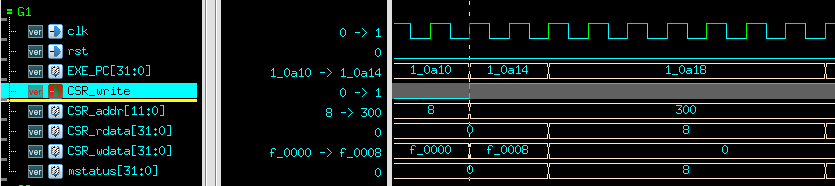
Block Diagram:

CSR:



Wave form:

CSRRW:



當EXE\_PC = 0x10a14時，代表EXE stage正在處理csrw mstatus, t0指令，因此會把CSR\_write的訊號拉高，同時因為CSR\_addr = 0x300，代表是要對mstatus register做操作，把CSR\_wdata[3]、CSR\_wdata[7] & CSR\_wdata[11:12]的值assign給mstatus[3]、mstatus[7] & mstatus[11:12]，同時把mstatus原先的值透過CSR\_rdata傳出來，寫回到rd裡面。

CSRRS:

一張含有 文字, 時鐘 的圖片

自動產生的描述

當EXE\_PC = 0x10b00時，代表EXE stage正在處理csrrs t0, mstatus, t1指令，因此會把CSR\_write的訊號拉高，同時因為CSR\_addr = 0x300，代表是要對mstatus register做操作，先判斷CSR\_wdata是否為0，再把CSR\_wdata[3]、CSR\_wdata[7] & CSR\_wdata[11:12]的值跟mstatus[3]、mstatus[7] & mstatus[11:12]做OR運算，並寫回mstatus[3]、mstatus[7] & mstatus[11:12]，同時把mstatus原先的值透過CSR\_rdata傳出來，寫回到rd裡面。

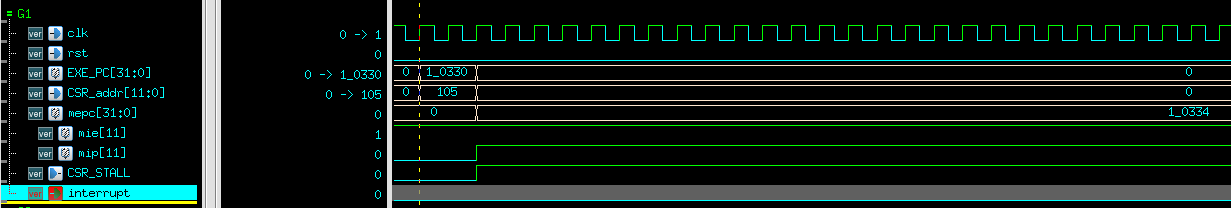
CSRRC:

一張含有 文字, 螢幕, 陳列 的圖片

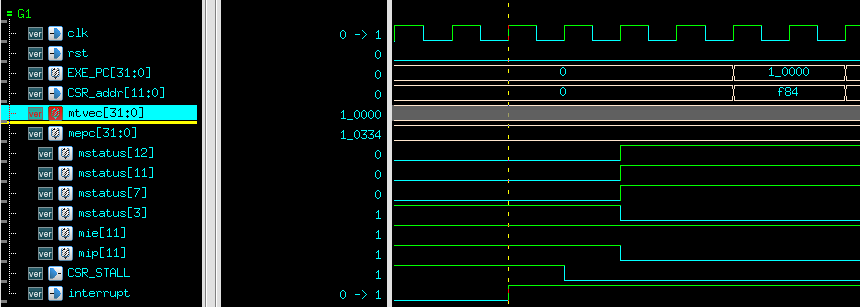
自動產生的描述

當EXE\_PC = 0x10b40時，代表EXE stage正在處理csrrc t0, mstatus, t1指令，因此會把CSR\_write的訊號拉高，同時因為CSR\_addr = 0x300，代表是要對mstatus register做操作，先判斷CSR\_wdata是否為0，再把~CSR\_wdata[3]、~CSR\_wdata[7] & ~CSR\_wdata[11:12]的值跟mstatus[3]、mstatus[7] & mstatus[11:12]做AND運算，並寫回mstatus[3]、mstatus[7] & mstatus[11:12]，同時把mstatus原先的值透過CSR\_rdata傳出來，寫回到rd裡面。

WFI:

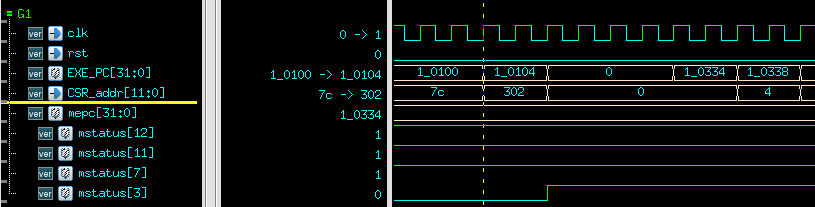


當EXE\_PC = 0x10330時，代表EXE stage正在處理wfi指令，先判斷mie[11]是否為0，若為1 ，會將mip[11]拉成1，並將pc + 4的值存入mepc裡，同時將CSR\_stall拉成1，暫停整個CPU直到interrupt來為止。



interrupt來了之後CSR\_stall就會被拉下來，pc跳到mtvec，並把mstatus[3]改成0，mstatus[7]改成mstatus[3]，mstatus[11:12]改成2’b11。

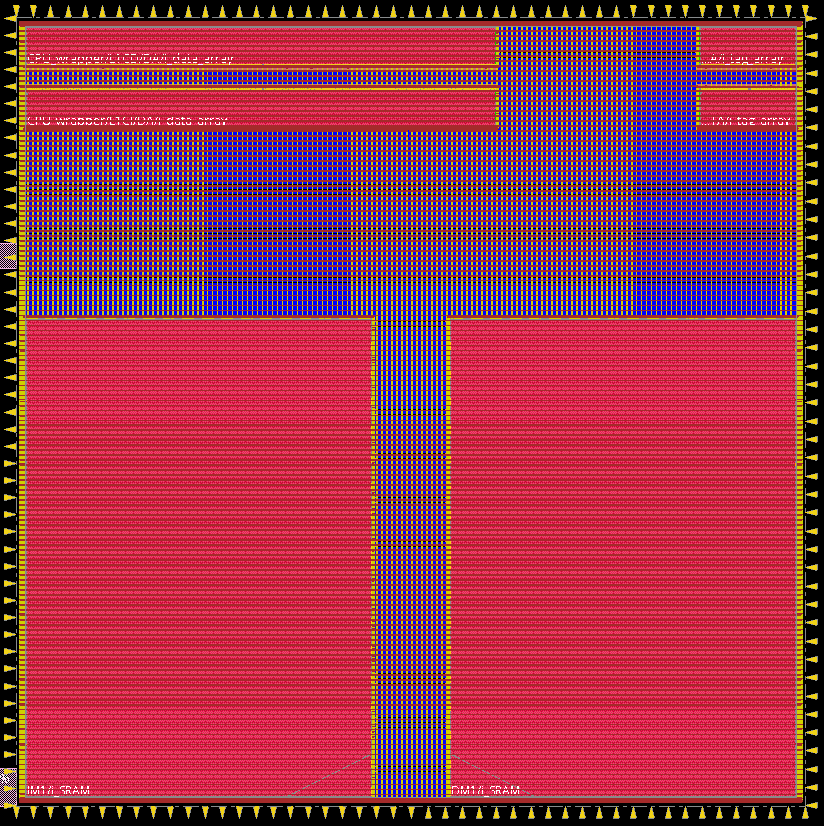
MRET:



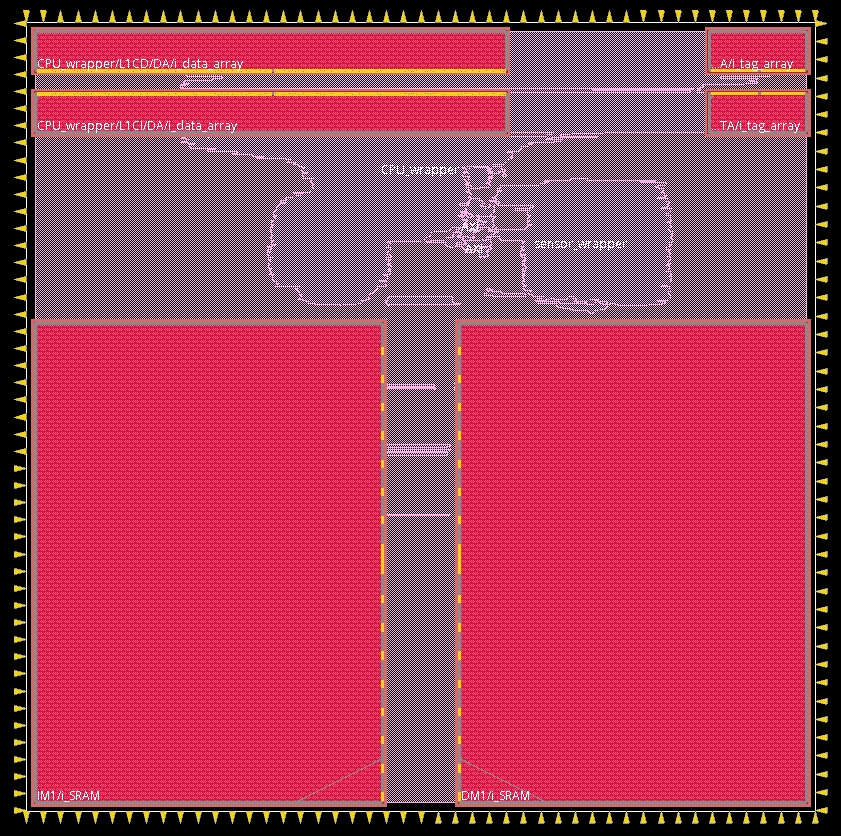
當EXE\_PC = 0x10104時，代表EXE stage正在處理mret指令，會將pc改成mepc，同時將mstatus[3]改成mstatus[7]，mstatus[7]改成1，mstatus[11:12]改成2’b11。

APR:

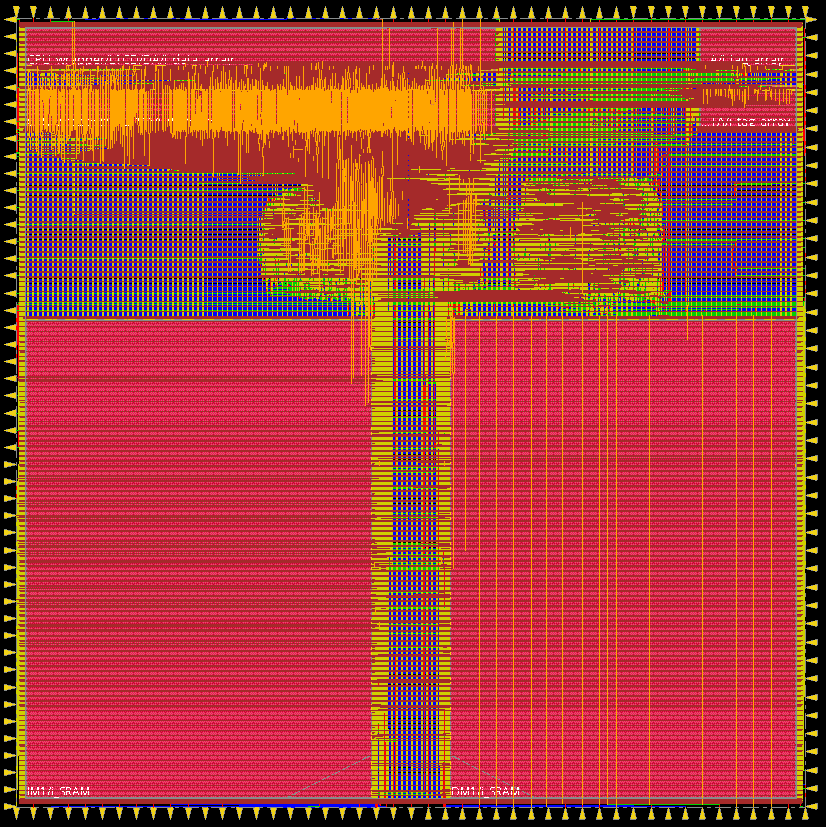
Floorplan view:



Amoeba view:



Physical view:



DRC:

一張含有 文字 的圖片

自動產生的描述

LVS:

一張含有 文字 的圖片

自動產生的描述

ANTENNA:

一張含有 文字 的圖片

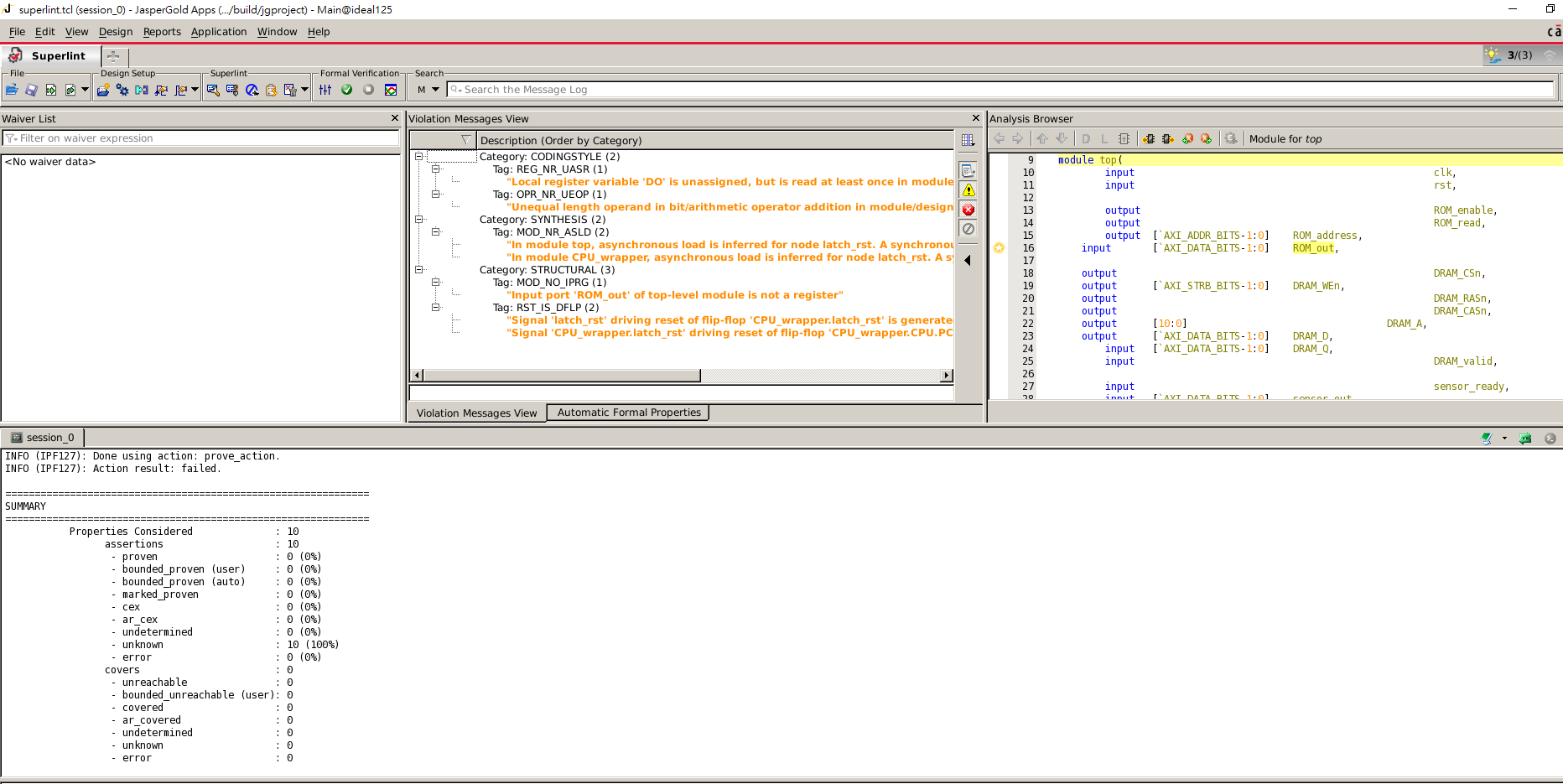
自動產生的描述

Superlint result:

Total warning/errors: 7

Total lines of code: 6890

Coverage:99.9%



一張含有 文字 的圖片

自動產生的描述

C program implement:

boot.c:

先將\_dram\_i\_end減去dram\_i\_start 得知總共會有多少筆資料要從Dram搬到Instruction Memory，再用一個for迴圈按照順序搬過去即可。而Data Memory的資料也是用一樣的方法。

Program2:

矩陣乘法是用三個for迴圈來達成，但是因為不能用乘法，所以先用boot algorithm算出兩數相乘的值，再按照順序擺進去即可。