VLSI System Design (Graduate Level)

Fall 2020

HOMEWORK II

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: **\_\_\_周昱佑\_\_\_** \_\_\_\_杜冠勳\_\_\_\_

Student ID: **N26090180** \_ **N26094883**\_\_\_

* **Summary :**

|  |  |  |
| --- | --- | --- |
| 1. | AXI.sv verification(jaspergold) | **DONE** |
| 2. | CPU\_wrapper.sv verification(jaspergold) | **DONE** |
| 3. | SRAM\_wrapper.sv verification(jaspergold) | **DONE** |
| 3.1 | IM=slave1(ID=0) | **DONE** |
| 3.2 | DM=slave2(ID=1) | **DONE** |
| 4. | Outstanding | 1 (In order) |
| 5. | Burst operation | Single transfer only |
| 6. | Transfer type: | No burst transfer |
|  |  |  |
|  |  |  |
|  |  |  |

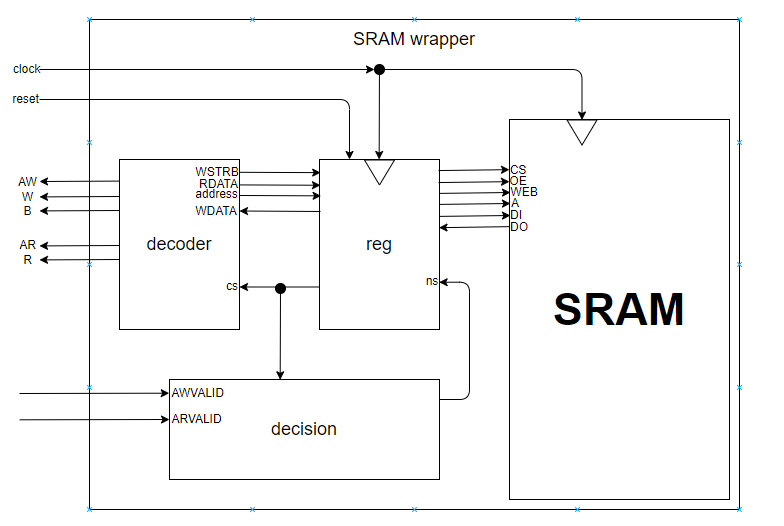
* **分工:**

|  |  |  |
| --- | --- | --- |
| **姓名** | **周昱佑** | **杜冠勳** |
| **負責工作** | **AXI**  **SRAM Wrapper** | **AXI**  **CPU Wrapper** |

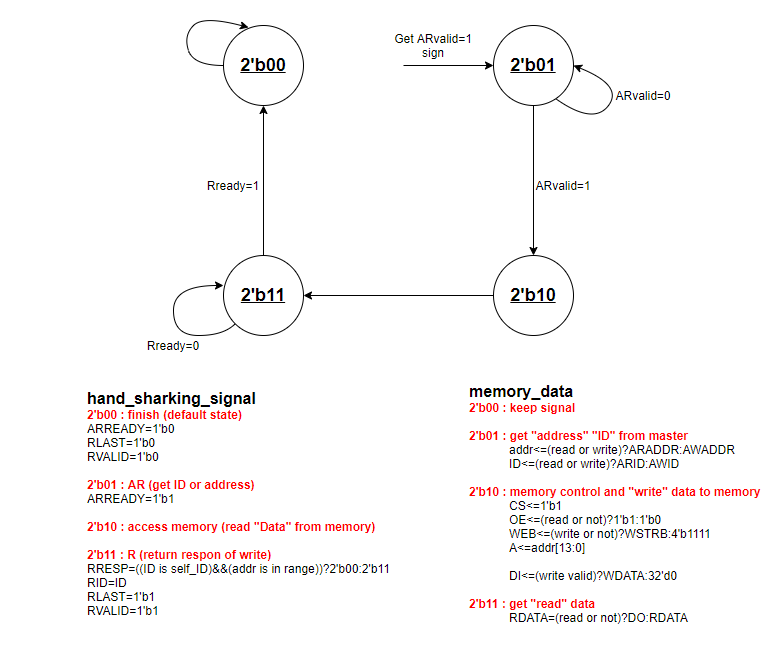
* **Architecture :**

**此次AXI的設計架構以狀態機來做訊號線的控制與讀存資料，以這樣設計的原因為較易於做判定是否要維持訊號與cpu的stall，同時在debug能較清楚的觀察到是哪個狀態的判定上出了問題，狀態的切換依照spec上描述，以valid與ready為主，在slave端會驗證ID與ADDRESS的正確性給予RESP為OK或DECERR，中間AXI的部分以arbiter來決定優先執行的部分，設計上以master\_dm優先於master\_im，write優先於read。**

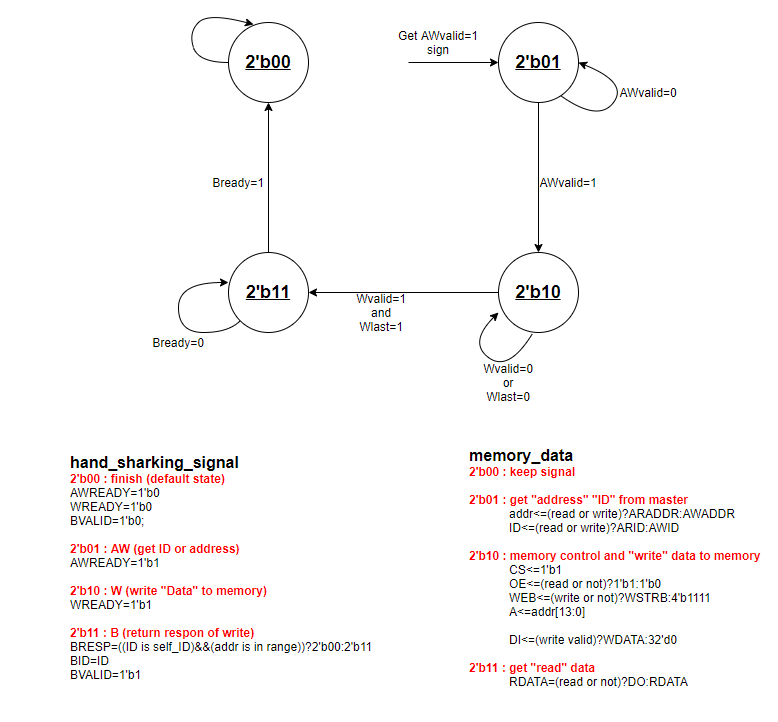
* Slave :



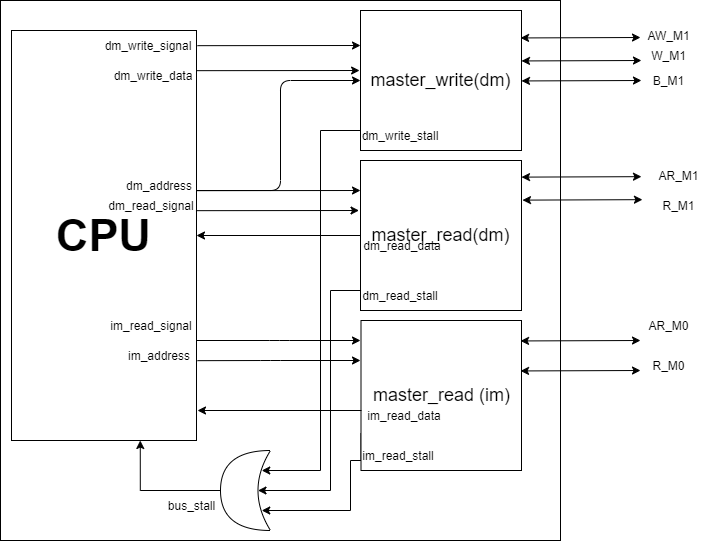
1. Read FSM



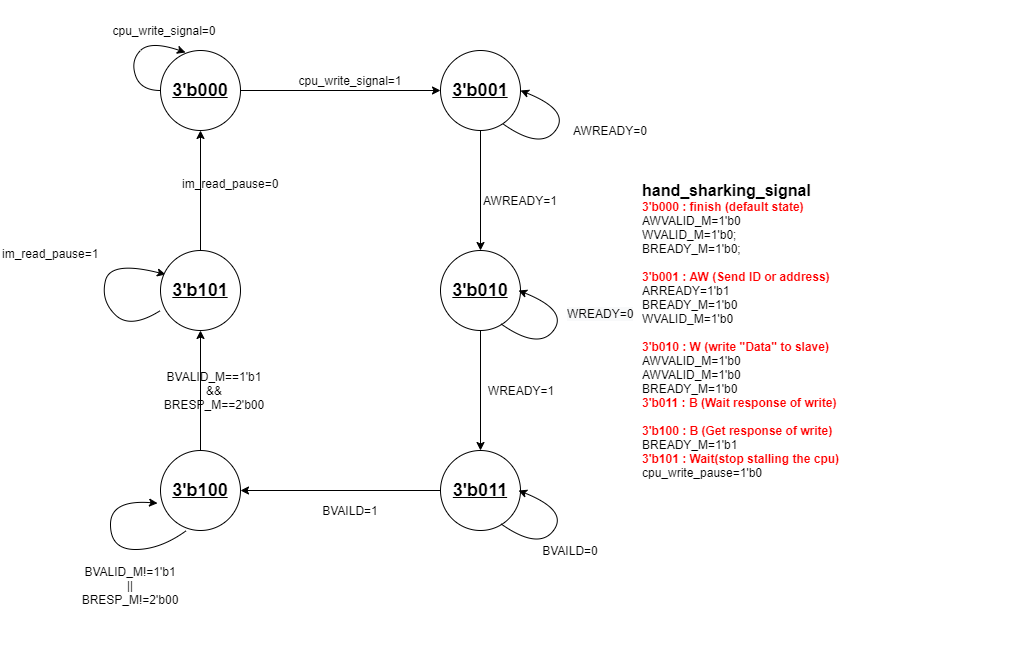
1. Write FSM



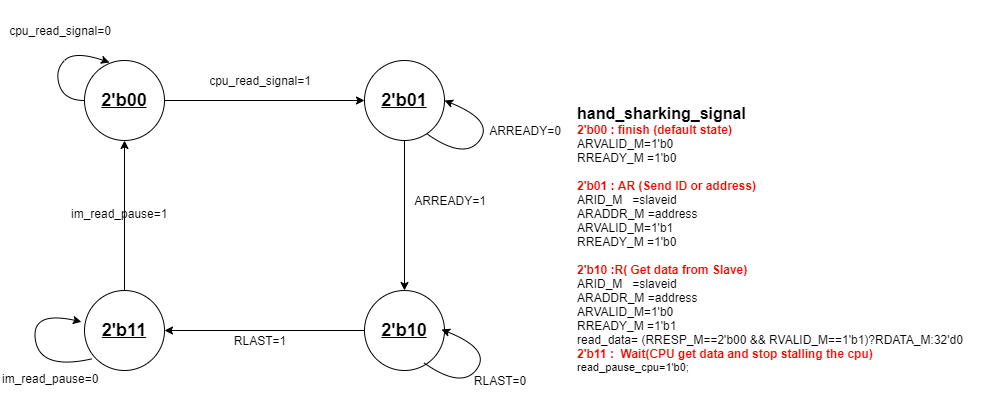
* Master:



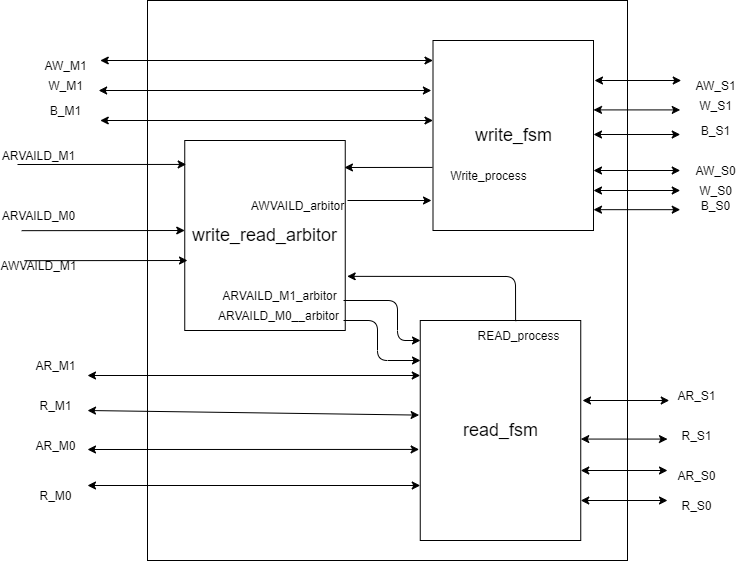
1. Master\_write



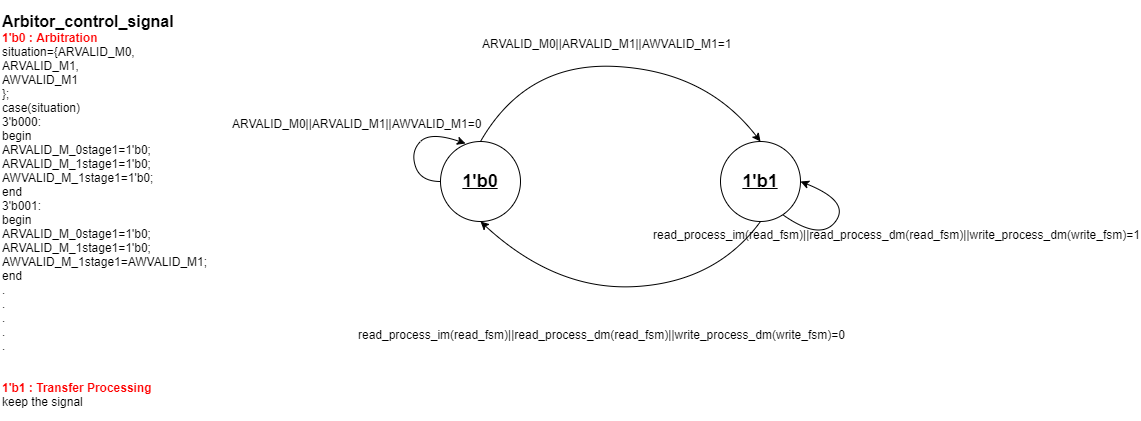
1. Master\_read



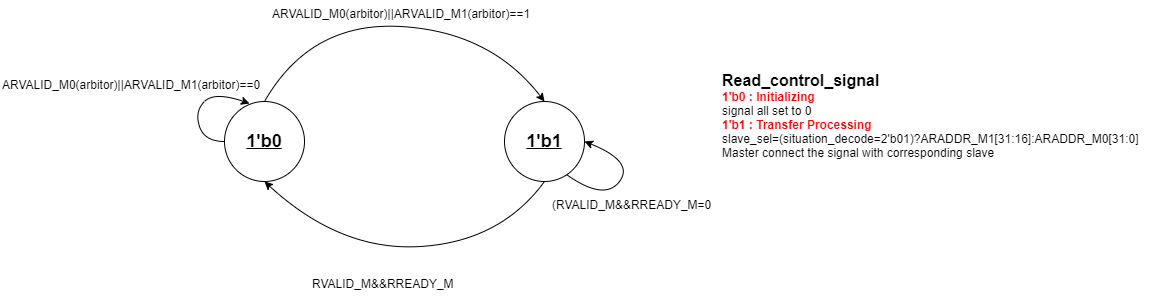
* AXI:

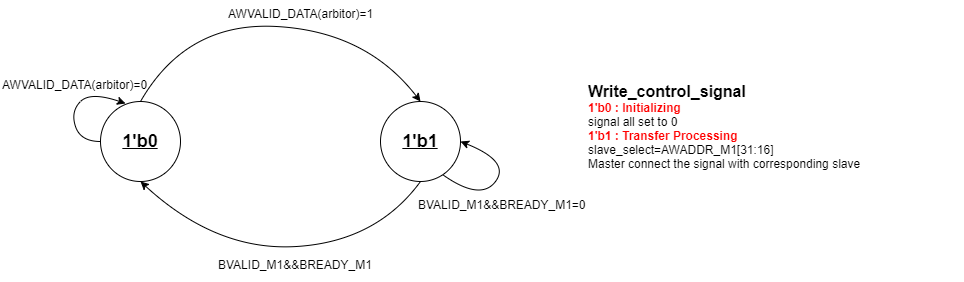


1. write\_read\_aribitor



2. read\_fsm



1. write\_fsm

* **Major problem & resolutions :**
* Slave :

1. RVALID、ARREADY、AWREADY等訊號線為unstable?

<sol.>等待master端送出相對應的hand sharking訊號才進入下一個狀態。

1. RID、RDATA、BID等資料為unstable?

<sol.>在轉換與接收data的state時才做改變並將資料栓鎖進reg中，其他時候則輸出栓鎖的數值。

1. 輸入訊號線出現亂跳的現象，port與線的數值不相同?

<sol.>檢查top.sv的接線，發現有誤，下次接完後要再細心的檢查。

1. 當訊號線讀跟寫同時送入slave端?

<sol.>多設定一個flag訊號來做優先度的判定，以write為優先再來才是read，達到防止狀態機亂跳狀態而送錯訊號。

1. Read data時，資料沒有在cpu stall時回傳，反而多延後一個stall區間才收到?

<sol.>當memory在讀出資料時，address送入為1個clock，送出資料(RDATA)為1個clock，所以在狀態機state10時，多停留1個clock讓資料被讀出並在state11時將資料送出。

* **Results of verification :**
* Max pending number : **1**

1. **Master wrapper :**

一張含有 文字 的圖片

自動產生的描述

1. **AXI :**

**一張含有 文字 的圖片

自動產生的描述**

1. **SRAM wrapper :**

一張含有 文字 的圖片

自動產生的描述

* **Lesson learned :**
* 周昱佑 :

在這次的作業中對於slave與master之間的溝通介面有了更多的了解，這次我們所學的為AXI4.0的協定，在這之中從hand sharking的了解是最主要的設計要點，當valid與ready同時為1時才能將資料當作正確的讀入或送出，所以我以這兩種訊號作為狀態機進下一個state的條件，這個條件設計原本我並沒有考慮到，是因為在測試jaspergold時，發現到這兩個訊號如果其中一者沒有拉為1，則另一者不能放下原本拉起的訊號，由於原本由狀態機的設計是依照valid來在給一個clock time的ready，所以在這裡修正進下一個state的狀態，並依照valid來判定資料抓取的確認訊號，並用address與AW、ARID來判定訪問正確的slave，由於我負責設計slave端的部分，同時考慮到要留一個clock time給memory作讀寫的動作，所以考慮到之後合成不能有multi-edge clock所以給一個stage作memory訪問用，另外一個修正點在AW、ARvalid同時送入時要先執行哪一個，所以多用了一個變數flag來作判斷，總結來說最後我將送至slave的訊號分為hand sharking(控制、確認)與data(要以register栓鎖)，由於一開始對這個協定不是很了解，所以花了不少時間跟實驗室的同學一起研究了一下spec並討論中間AXI的設計，並了解到其中並不能用純組合電路的方式來做設計，需要考慮到master訪問的優先度、valid與ready訊號先後問題、未被選擇的interface訊號保持的問題等，基本上都要加上register來作資料得儲存，在AXI最後的設計上才知道AW與W是有可能同時送出資料的，所以在arbiter上判斷條件又作了點修正，也感謝助教的協助解答了我不少問題，讓我對這次陌生的作業學習到不少。

* 杜冠勳:

此次作業學習到到了 AMBA AXI 的溝通架構 是以handshake 的基礎上完成的 而這次作業中我設計了AXI 架構 以及CPUWrapper架構

再設計AXI架構時因為過去不好的習慣導致在使用狀態機時產生了latch 經過這次的練習修正了許多觀念 也成功設計出來

設計CPU wrapper 時較為簡單 但是當與CPU整合時又遇到不少問題 常常思考不夠周全 忘記要改的地方很多 導致花很多時間在debug

但經過這一個月以來的作業也讓我對整個系統設計架構有更進一步的認識