Lab1 report

By 許庭瑋

1. module介紹

基本的module就不再介紹了，之前都講過了，也不是這次作業的重點。

Pipeline的部分就根據助教給的圖去增加因為miss產生的stall。沒啥問題。

CPU要做點小更新，把cache加進來，作為主體的主要功能都沒變。

Data memory就是memory，跟上次相比，需要根據當下狀況作出判斷，需要有 penalty，基本上code助教都寫好了，也沒改甚麼東西。

dcache_controller是處理cache的控制單元，負責處理當下的情況。把addr讀進來，根據上課所教的把addr分成tag、index、offset。這次的valid 和dirty bit是合在tag裡的，需要注意。一些signal的處理部分助教已經幫忙打好了。直接用就行。接著是read data的部分，首先需要根據是否hit來決定read出來的data。然後需要根據offset讀來自sram的data，慣例是offset是4的倍數，所以就把32/4=8種情況列出來，根據哪個情況讀哪段data就好了。

Write data的部分，一樣根據offset分情況，但先需要把cache的data讀出來，再根據哪個情況去把哪段data替換掉，再寫回去。Controller的部分，看起來是用來分辨當下的情況。所以就根據情況來決定訊號要設成甚麼。因為要設的訊號就是其名字，所以這部分很直覺。

dcache_sram就是cache，是存data的部分。Hit的處裡就根據給的pdf上面的圖去做設定。為了處理LRU，我為每個set多設了一個register叫replace，是紀錄如果memory的data來了，要replace哪個。如果一個set有一個部分被read或write，就把replace設成另一個。Read的部分就根據讀進來的tag跟哪個一樣、是不是enable，是不是valid來確定。Write的部分需要看是不是更新值，如果是那就是有hit，代表tag有一樣。如果不是的話，代表是從memory讀的值。

2.所遇困難與解決

這次助教給的預先設好的code有點多，需要花點時間去看清楚怎麼實作。其實也沒有很懂，但要做的部分能做出來就好了。那個sram有點複雜，一開始想說LRU就設register叫replace就可以紀錄了，但後來debug時發現 2 way associative 的cache在write上有點複雜，還需要用hit做判斷。Controller的部分就很直覺。這次的

debug比上次難，然後ALU的input需要改signed。甚至說因為需要不止1個cycle去讓state變成readmiss去跟memory拿data，所以memory的 count也要改，細節很多，比上次多。

3.系統

我是用 Windows 系統和 iverilog 去編譯