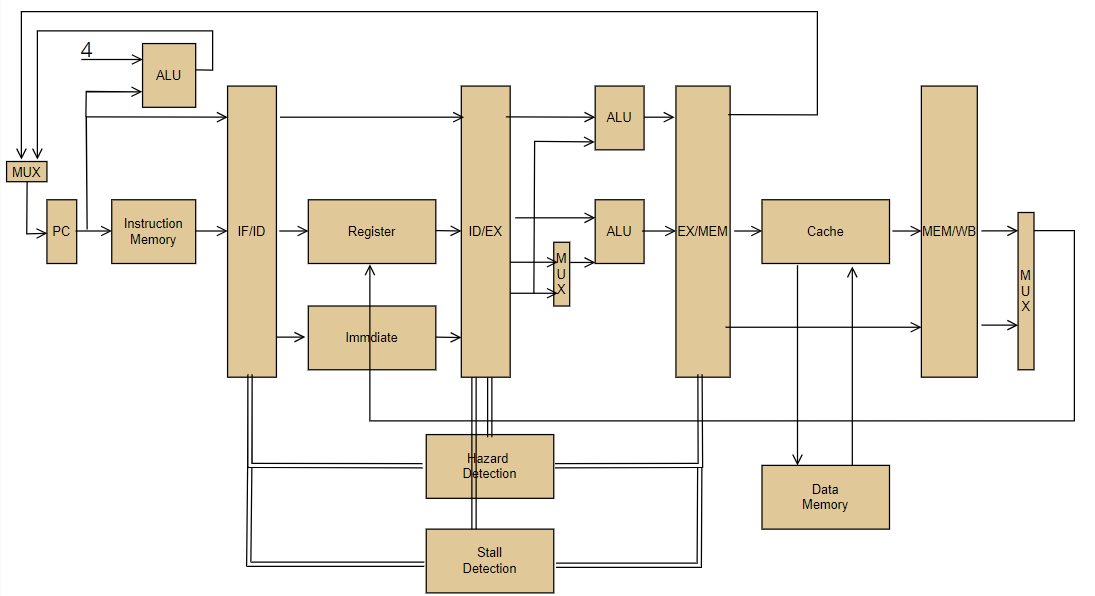
計算機結構 Final Project Report

B10901058 傅渝翔

1. Block Diagram of CPU architecture



1. Data path of instructions not referred
2. JAL

和BNE類似，BNE會檢查alu的輸出，但是jal不會。差別還有寫入register，因為jal不用檢查要不要jump，但jal需要寫入register，所以ALU的輸出就直接是PC+4。

1. JALR

和JAL相同，只是jump的標的是某個register的內容，因此只差在一個MUX而已。

1. AUIPC

只要把PC拉進ALU裡面進行加法就好，以及immediate的處理。不過我的immediate是先處理好再送進ALU，所以在ALU裡面不需要處理immediate。

1. ECALL

送一個結束的信號，結束之後任何信號就保持不動。

1. Handling of multi-cycle instructions

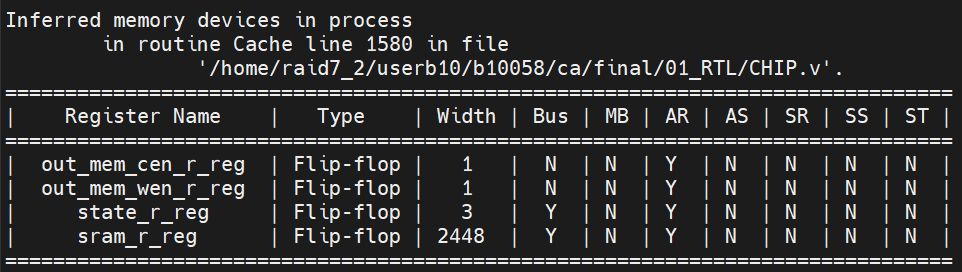
使用finite state machine，有分成single-cycle的state和multi-cycle的state。當程式在multi-cycle的狀態時，就不要繼續讀新的instruction，也就是不要更新PC以及IF/ID等等，如上圖的stall detection。

1. Observation

在使用比較多運算資源的程式，例如兩個遞迴算法裡面，因為access的記憶體數量較少，所以實作cache提升的速度並不明顯，但是在使用很多記憶體的程式，例如排序演算法，因為程式要一直access記憶體，所以實作cache會對程式執行速度有很顯著的提升。

1. Cache Architecture
2. 使用2KB Cache
3. 一個block用16bytes，總共128bits，剛好和從memory來的資料相同。
4. 總共有16個blocks。
5. 因為test bench是檢查最後memory的資料，因此如果使用write back，等於在程式執行完之後又要把cache資料遍歷一遍寫進memory，而且還要處理read miss的寫入，造成麻煩，因此我選擇使用write through的方式實作cache。
6. Width

由圖片可以看到最後的total width是2453，小於3000的限制。

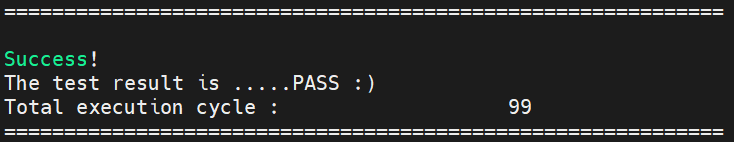


1. Cache Improves time performance

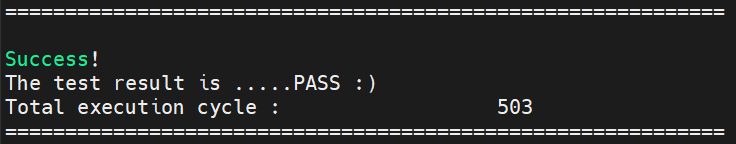
|  |  |  |  |
| --- | --- | --- | --- |
|  | Without cache | With cache | Speed-Up |
| I0 | 114 | 99 | 1.15 |
| I1 | 540 | 503 | 1.07 |
| I2 | 489 | 476 | 1.03 |
| I3 | 1726 | 1121 | 1.54 |

1. Cycle number of each instruction set

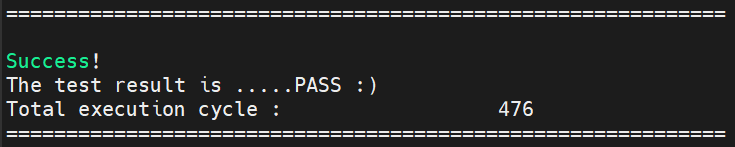
I0



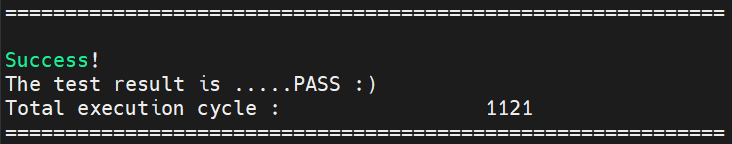
I1



I2



I3



1. Synthesizability

