**Members & Team work**

| | |
|---|---|
| Name: | 山下夏輝（Yamashita Natsuki） |
| Student ID: | R08922160 |
| Work: | testbench.v, ALU.v, pipeline regesters: modify the faults in project1 |
| | dcache_top.v: implement the cache controller |
| Name: | 徐紹軒(Ryan Hsu) |
| Student ID: | b06902137 |
| Work: | CPU.v: connection between modules |
| | testbench.v: initialize variables in modules |
| | pipeline registers add cache stall |
| | minor change(syntax) on dcache_top.v |

**How we implement this project**

[Yamashita]

- Modified the codes of project1.
    - Initialized pipeline register
    - Modified ALU: deleting the conditions dealing with unknown data
    - Modified regs in some components to wires for avoiding registers having unknown
- Implement dcache_top.

[Ryan]

- 這次山下把主要的部份(dcache_top.v)寫掉了，所以我大概就是看懂架構之後，就山下改好的 project 1 為基礎，把要改的部分（pipeline register 加上 data cache stall 的 input 和判斷）改好，CPU 串起來。

**Implementation of dcache_top.v**

[Yamashita]

- p1_data

    The value of p1_data is for returning CPU The offset*c indicates the place in 'r_hit_data' and extract the continuous 32bits data. r_hit_data is 256 bits, p1_data is 32 bits and p1_offset is 5 bits. Then 256/8 = 32. So c of offset*c is 8. It let p1_data have the continuous 32bits data from r_hit_data.

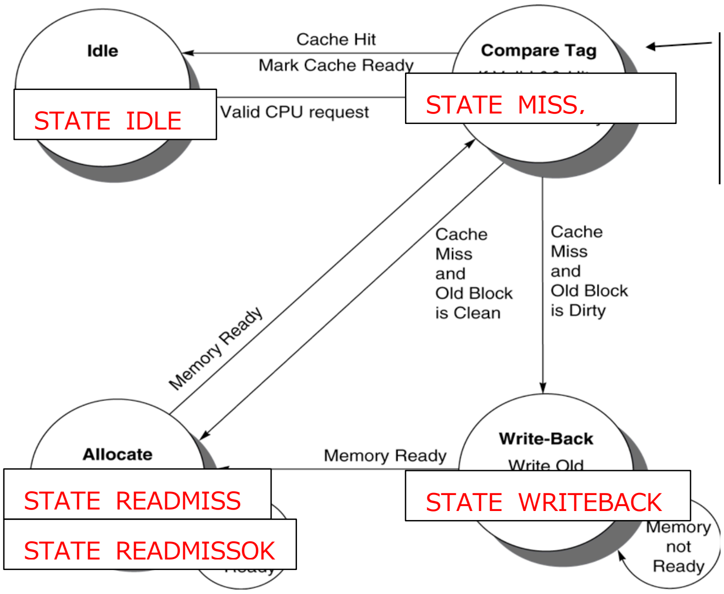
- w_hit_data

    The value of w_hit_data is used in case of write data in cache through the wire below.

    ```
    assign    cache_sram_data = (hit) ? w_hit_data : mem_data_i;
    ```

    There are 2 case: 1) when write data in cache from memory. 2) when write data in cache from CPU. In case 1, w_hit_data gets the 256 bits data. In cese 2, w_hit_data gets the data converted to 256 bits from the 32 bits data from CPU.

- Controller
    - Each state is corresponding to the graph like below.



- The parameter of mem_enable is used for accessing the memory whichever to read data from the memory or to write data into the memory.
- The parameter of mem_write is used for writing data into the memory.
- The parameter of cache_we is used for writing data into the cache.
- The parameter of write_back is used for writing data into the memory when the data is dirty.
- In STATE_READMISS, STATE_WRITEBACK, the value which will not be used in next state needs to be set as 0 in 10th cycle.
- In STATE_READMISSOK, the parameter of cache_we needs to be set as 0 because it was set as 1 and the data was written into the cache in STATE_READMISS.

**Difficulties and solutions**

[Yamashita]

- p1_data

  Difficulty:

  Do not know how to convert 256bits data to 32bits data

  Solution:

  offset*8

  The value of p1_data is for returning CPU The offset*c indicates the place in 'r_hit_data' and extract the continuous 32bits data. r_hit_data is 256 bits, p1_data is 32 bits and p1_offset is 5 bits. Then 256/8 = 32. So c of offset*c is 8. It let p1_data have the continuous 32bits data from r_hit_data.

- w_hit_data

  Difficulty:

  Not easy to understand how w_hit_data works

  Not easy to distinguish between how w_hit_data and r_hit_data

  Solution:

  r_hit_data and p1_data_i is needed as input of w_hit_data because there are 2 case which w_hit_data is used in.

The value of w_hit_data is used in case of write data in cache through the wire below.

assign    cache_sram_data = (hit) ? w_hit_data : mem_data_i;

There are 2 case: 1) when write data in cache from memory. 2) when write data in cache from CPU. In case 1, w_hit_data gets the 256 bits data. In cese 2, w_hit_data gets the data converted to 256 bits from the 32 bits data from CPU.

[Ryan]

- 最開始串完 CPU 之後發現 register 和 data memory 的值都不會變，後來把那些值印出來後發現原來是沒有初始化，就在 testbench.v 裡面加了初始化
- output.txt 的 final state 跟助教的 reference 不相同，好像在 cycle 100 左右開始 cache 內的值會有出入，後面就都不一樣了，不知道是不是 project 1 的部分沒做好造成的(unsolved)