Report

B07901068 黃敬騰

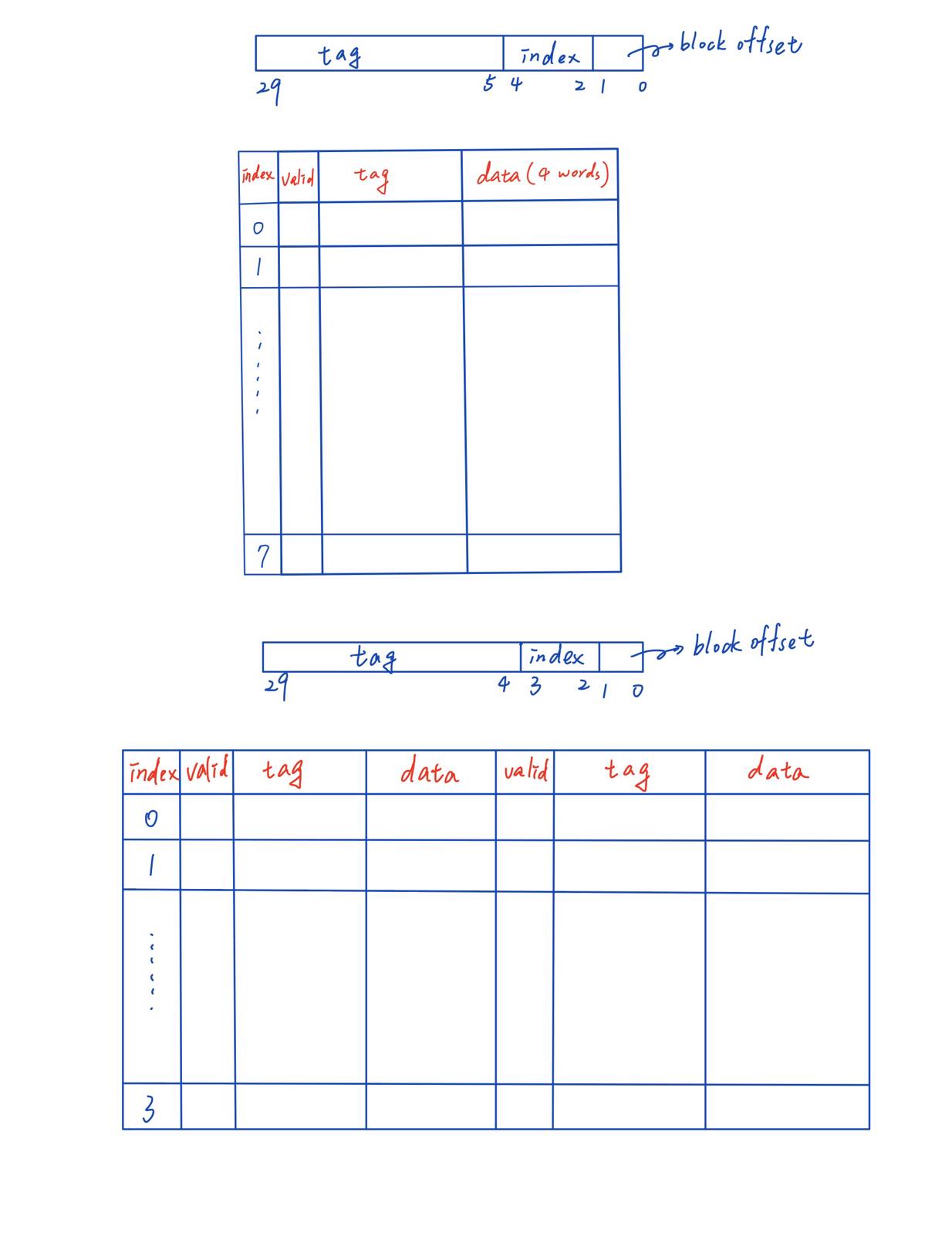
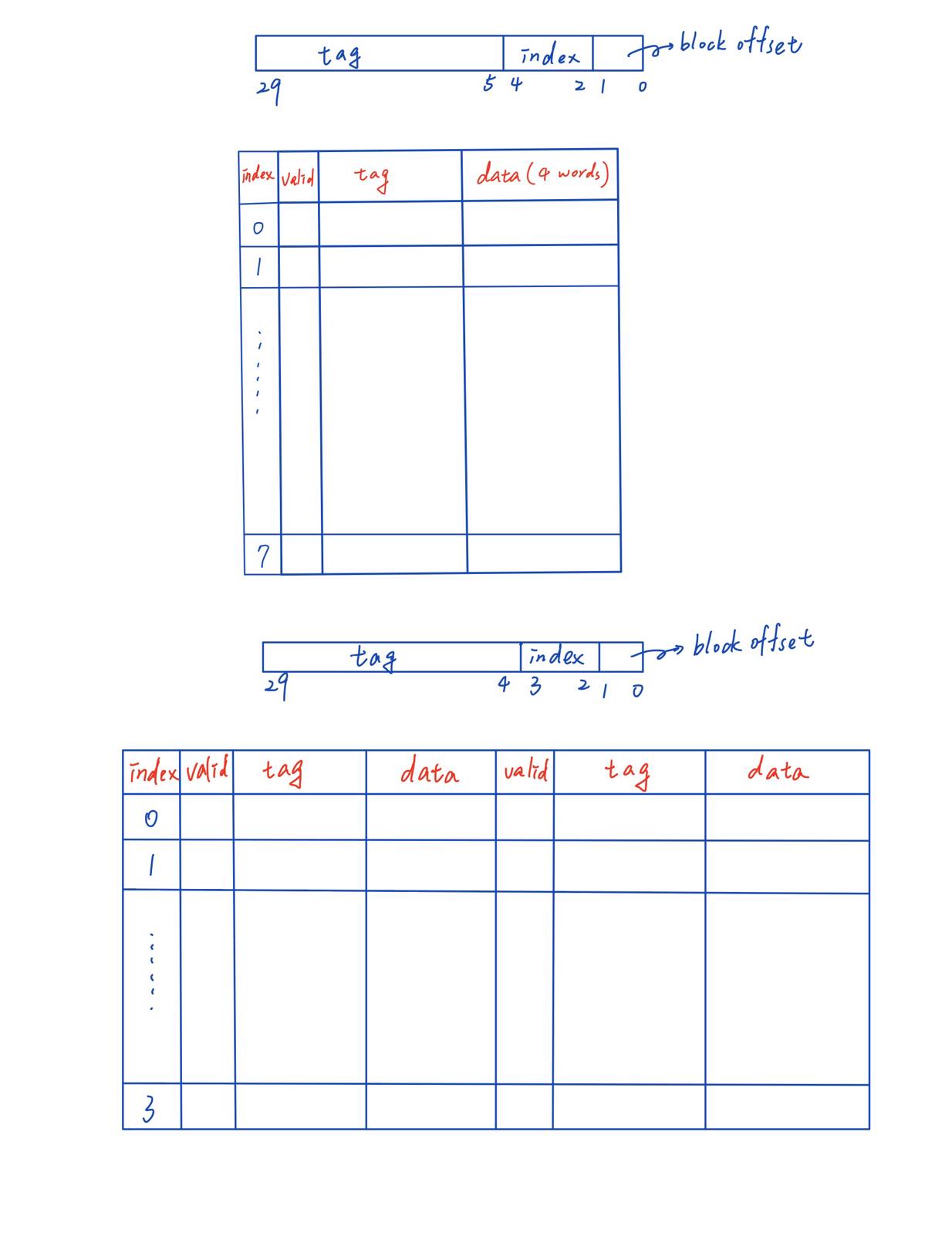
1. The cycle time used in **cache\_syn.sdc** : 10ns

The gate-level pass cycle time for **cache\_dm\_syn.v** : 10ns

The gate-level pass cycle time for **cache\_2way\_syn.v** : 10ns

1. Placement of cache

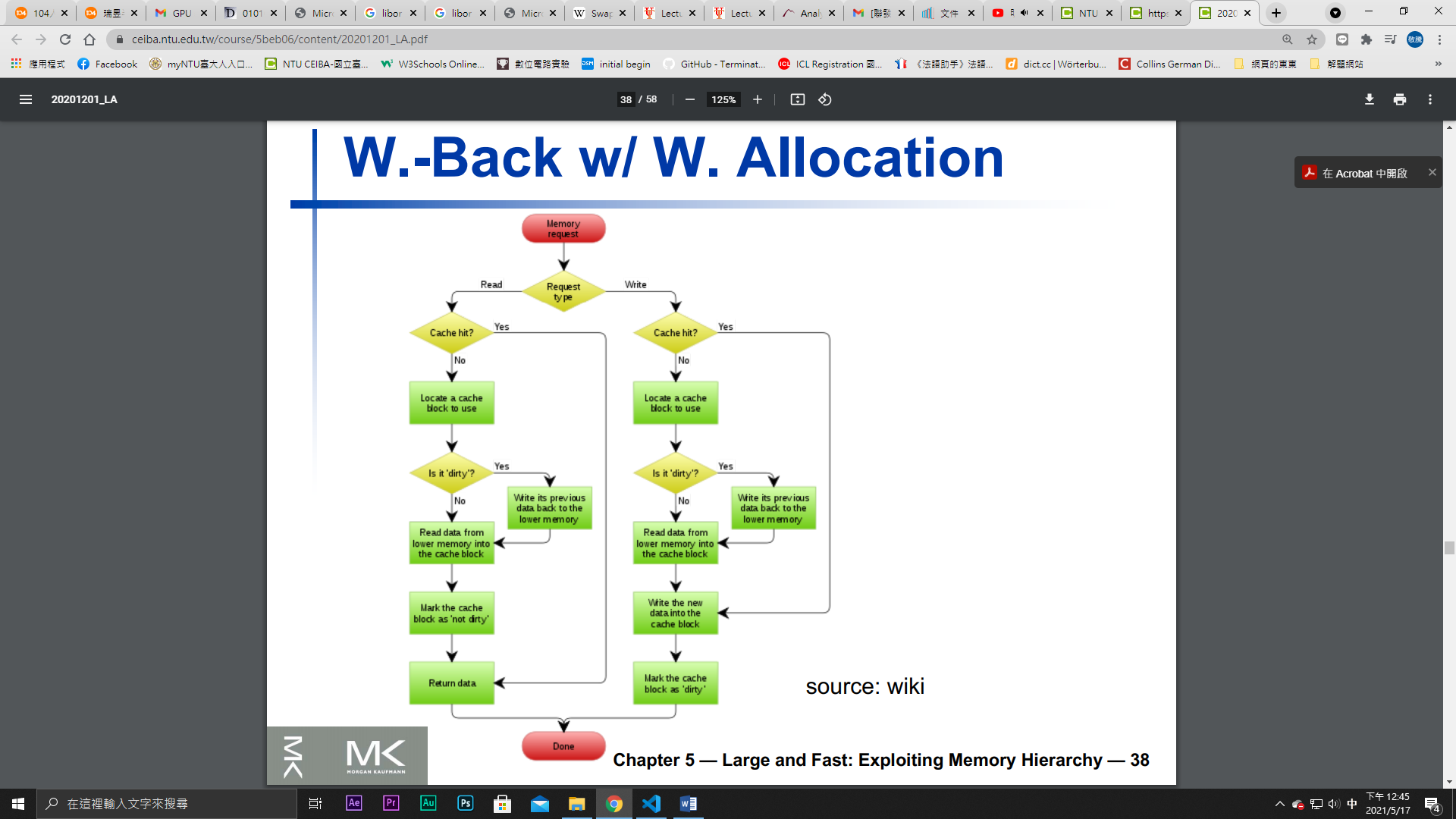
**Direct-mapped 2way-associated**



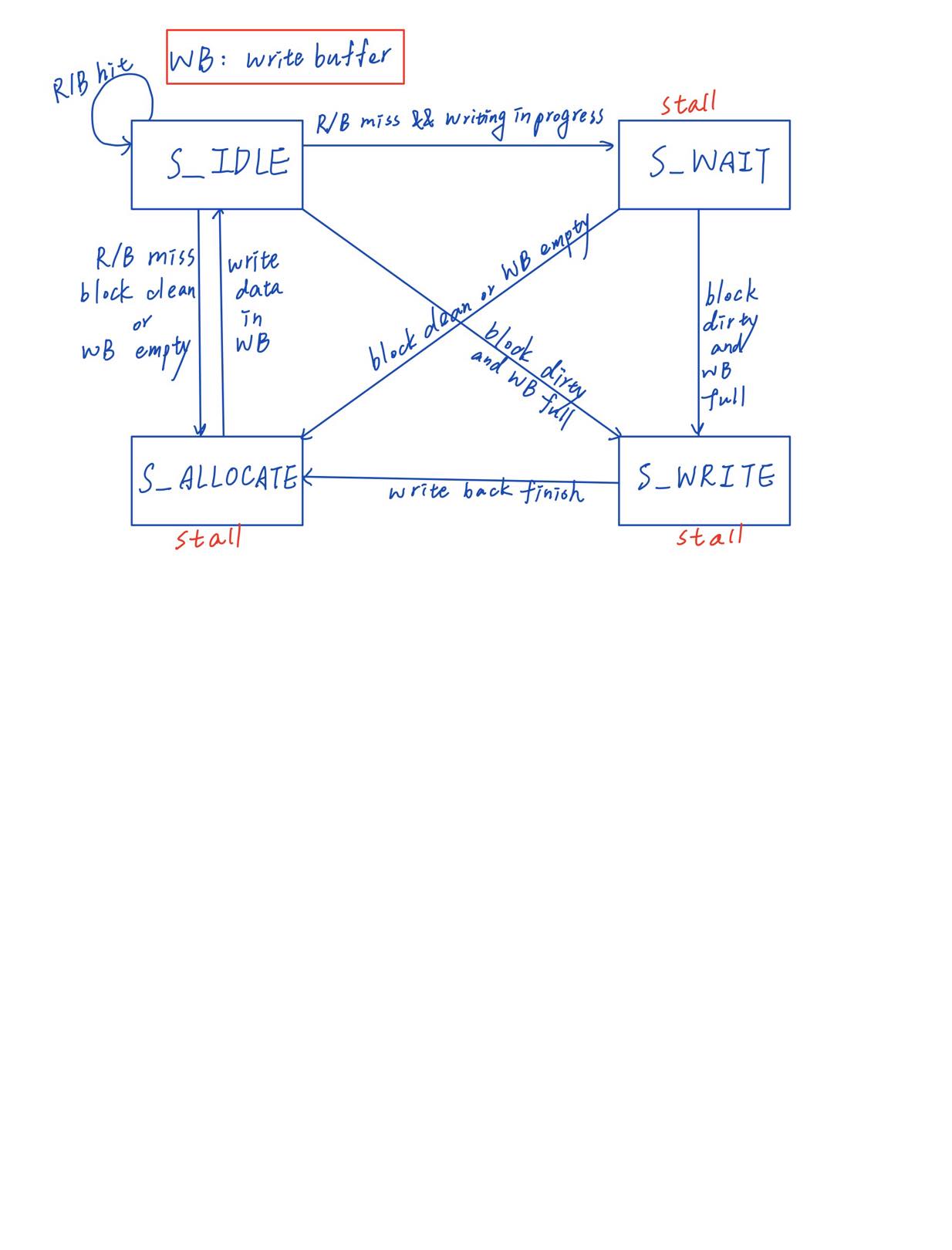
*For 2way-associated cache, if a miss occur(memory allocation is needed), I will choose the replaced block as “less recently accessed”. If a block is just replaced by the block from lower memory, it will be marked as “recently used” and the other block in the same index will be marked as “less recently used”.*

1. Read/Write policy

I choose **write-back** as memory allocation, picture below is the block diagram of write-back method I use.



1. FSM(with write buffer)



1. Performance Evaluation

**Direct-mapped(without write buffer)**

Read miss rate : 1280 / 2048 = 62.5%

Write miss rate : 256 / 1024 = 25%

Write back (dirty) rate : 256 / 3072 = 8.33%

Execution cycles : 12031

Stalled cycles : 8960

**2way-associated(without write buffer)**

Read miss rate : 512 / 2048 = 25%

Write miss rate : 256 / 1024 = 25%

Write back (dirty) rate : 256 / 3072 = 8.33%

Execution cycles : 8191

Stalled cycles : 5120

1. Compare

From (e), we can see that the most significant difference between two designs is “Read miss rate”, which is much higher in direct-mapped cache design. The reason is in the **last part of processor read**, whose input address pattern is 0 -> 32 -> 1 -> 33 -> 2 -> 34 …… 991 -> 1023.

For direct-mapped cache, memory of address 32 will erase memory of address 0 since they possess the same index and 1 must erase 32 for the same reason because each entry **has the capacity of only one block**, so it has to allocate memory on **every** read operation.

On the other hand, every entry of 2way-associated cache has cacpacity of 2 blocks, so memory allocation only occurs at 0 , 32, 4, 36 ……….., which is **four times smaller** than that of direct-mapped.

Additional features(Bonus)

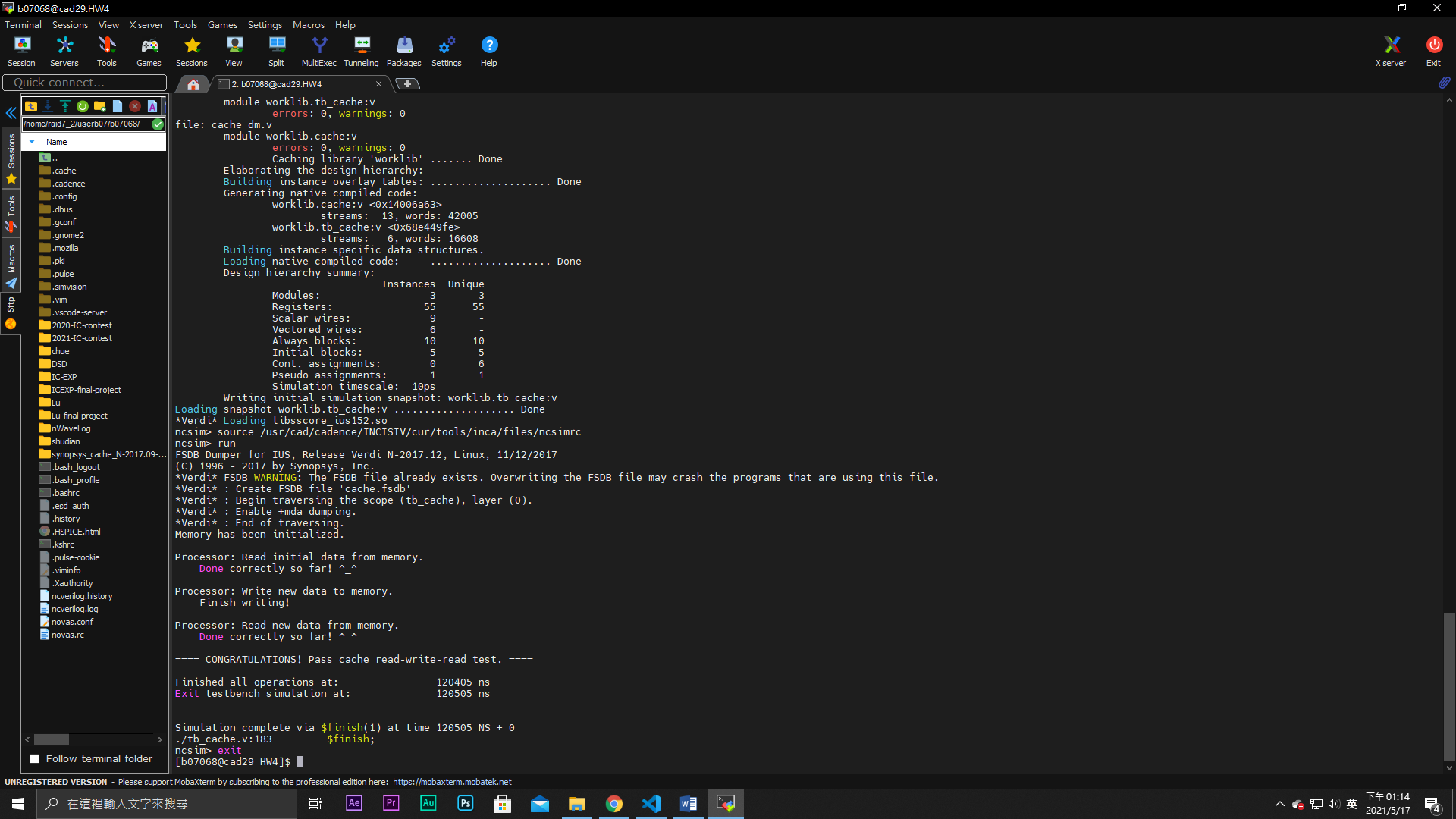
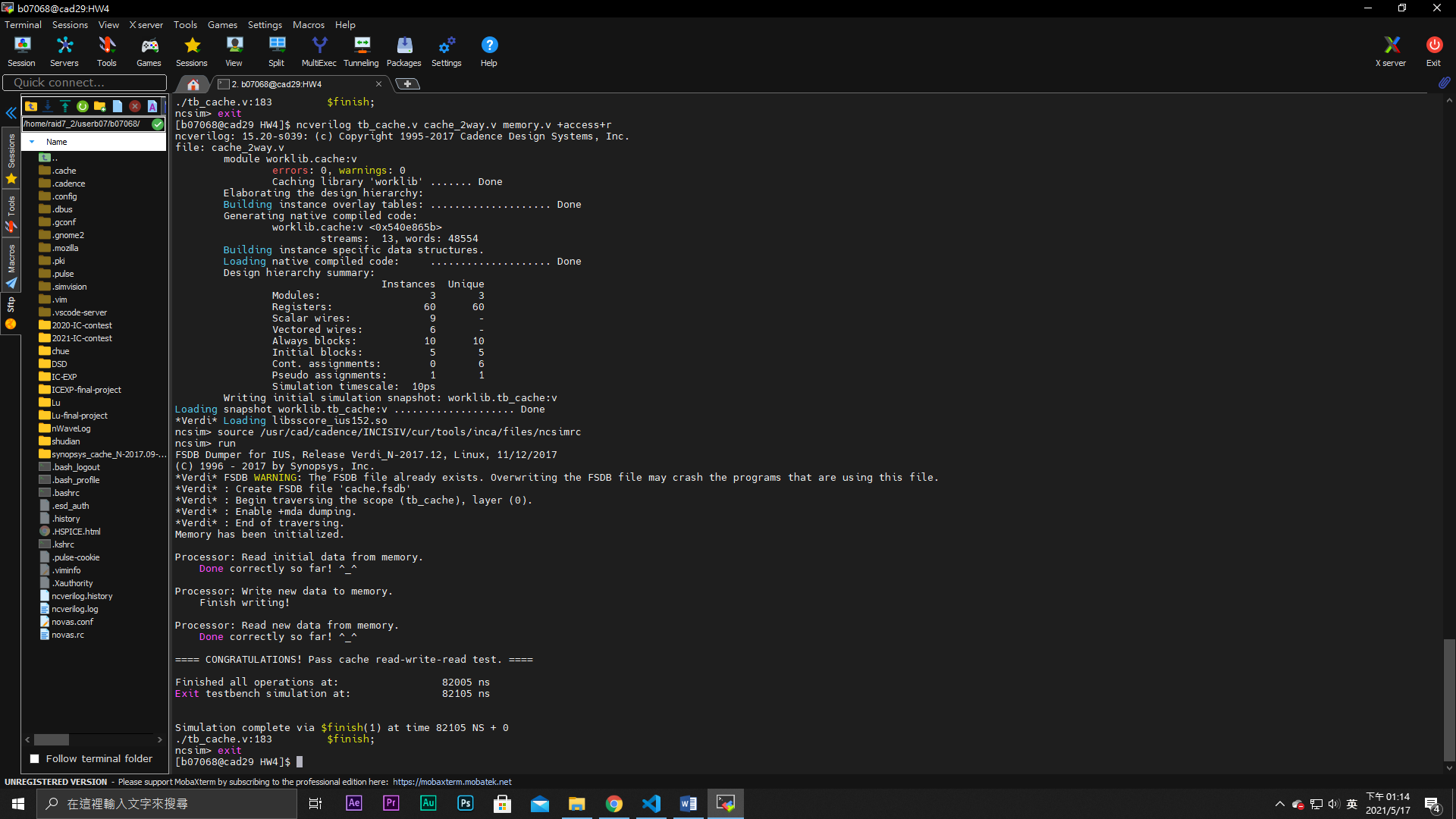
1. **Write buffer**

In order to decrease the processor stall on writing the dirty memory back to lower level memory, I make a 1-entry write buffer in both direct-mapped and 2way-associated cache. When a dirty block must be written back to memory, It will first check whether the write buffer is empty. If so, It will put the dirty block and its address in the buffer and proceed directly to fetch new block from memory. After that, the data in write buffer can write the data to memory without stalling due to memory write back. In short, the design only have to stall on writing if the write buffer is not empty.

Below is the improvement in total time(RTL simulation)

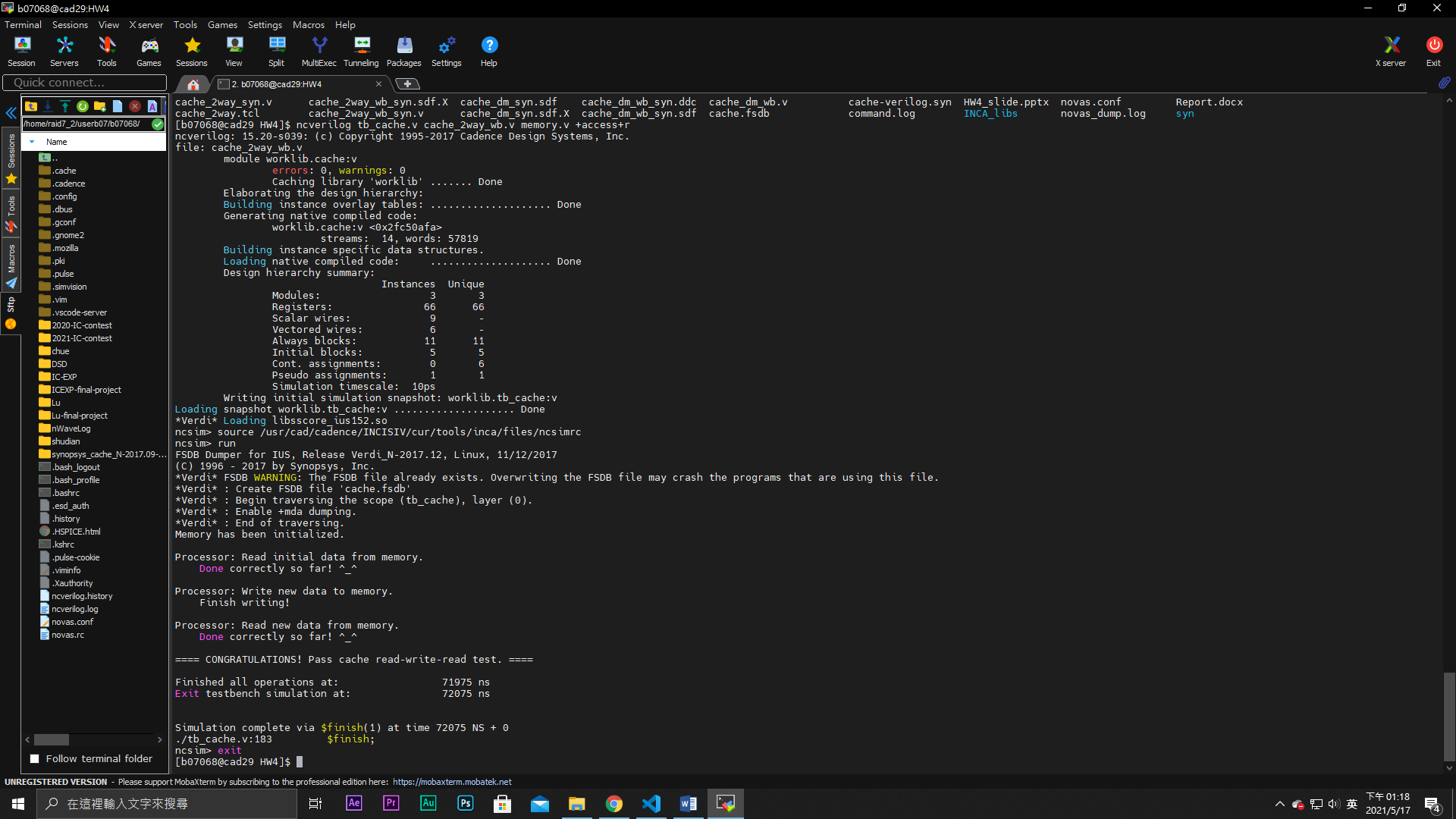
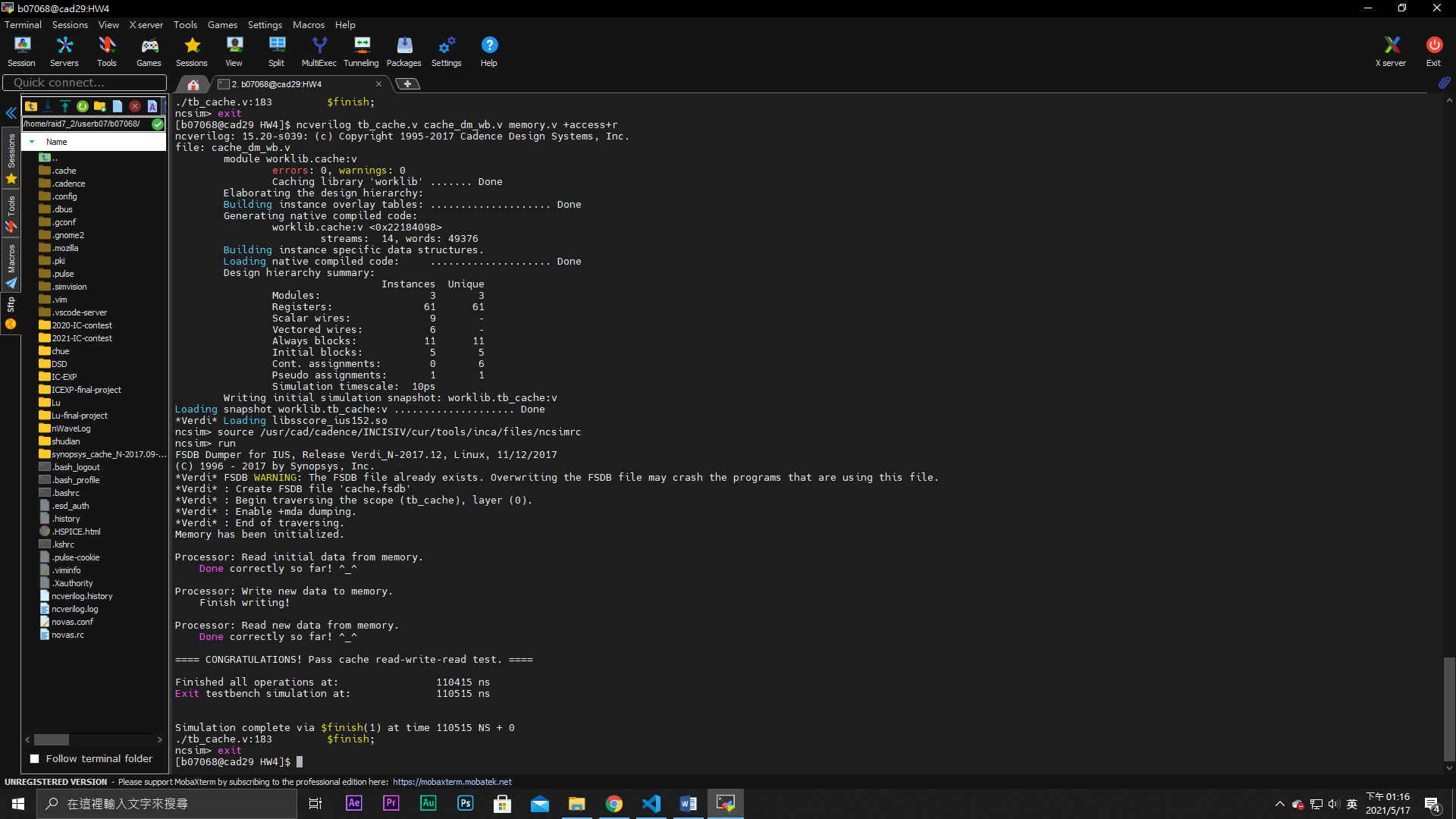
**Without write buffer**

**Direct-mapped 2-way associated**

**With write buffer**

**Direct-mapped 2-way associated**

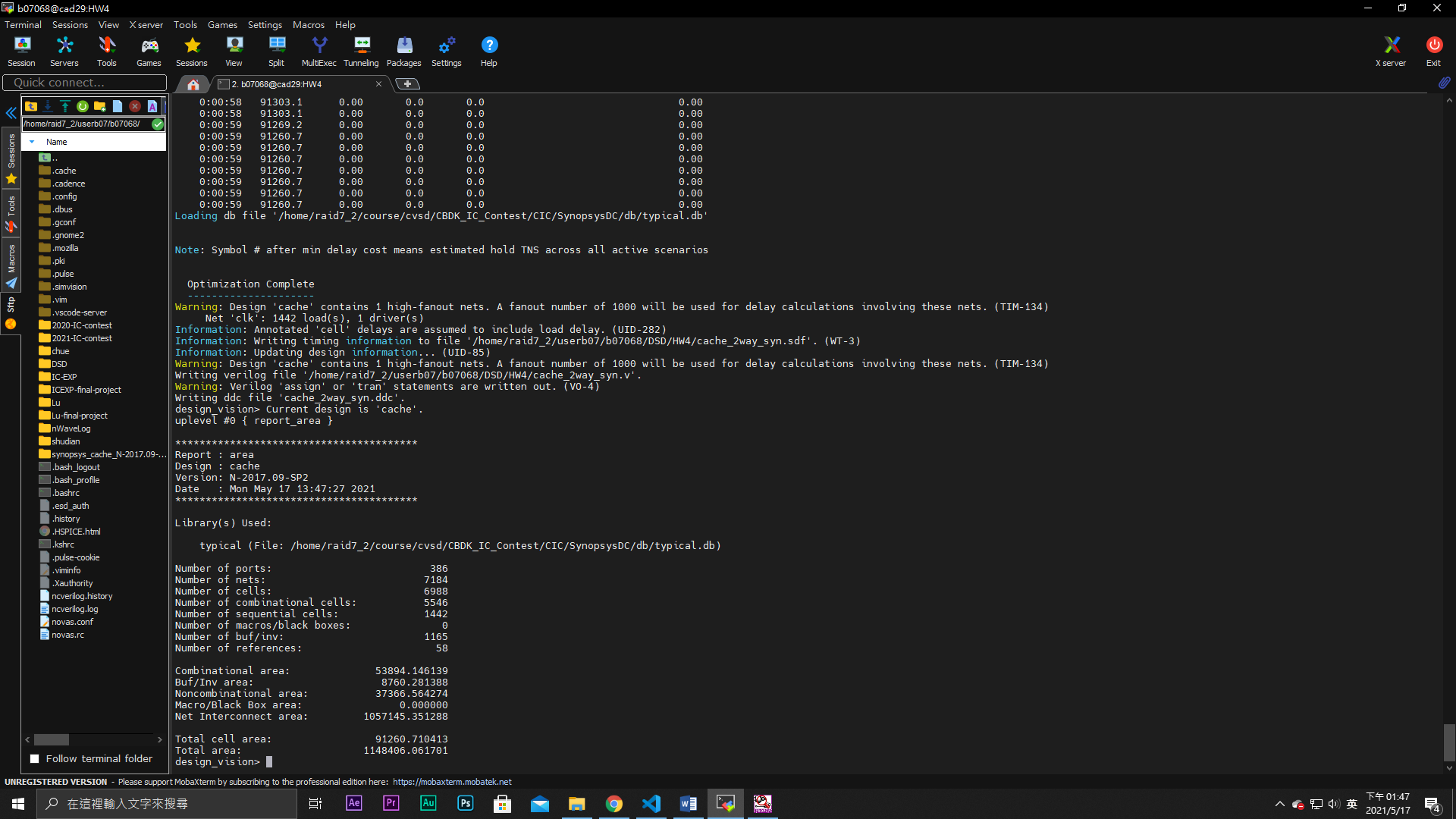
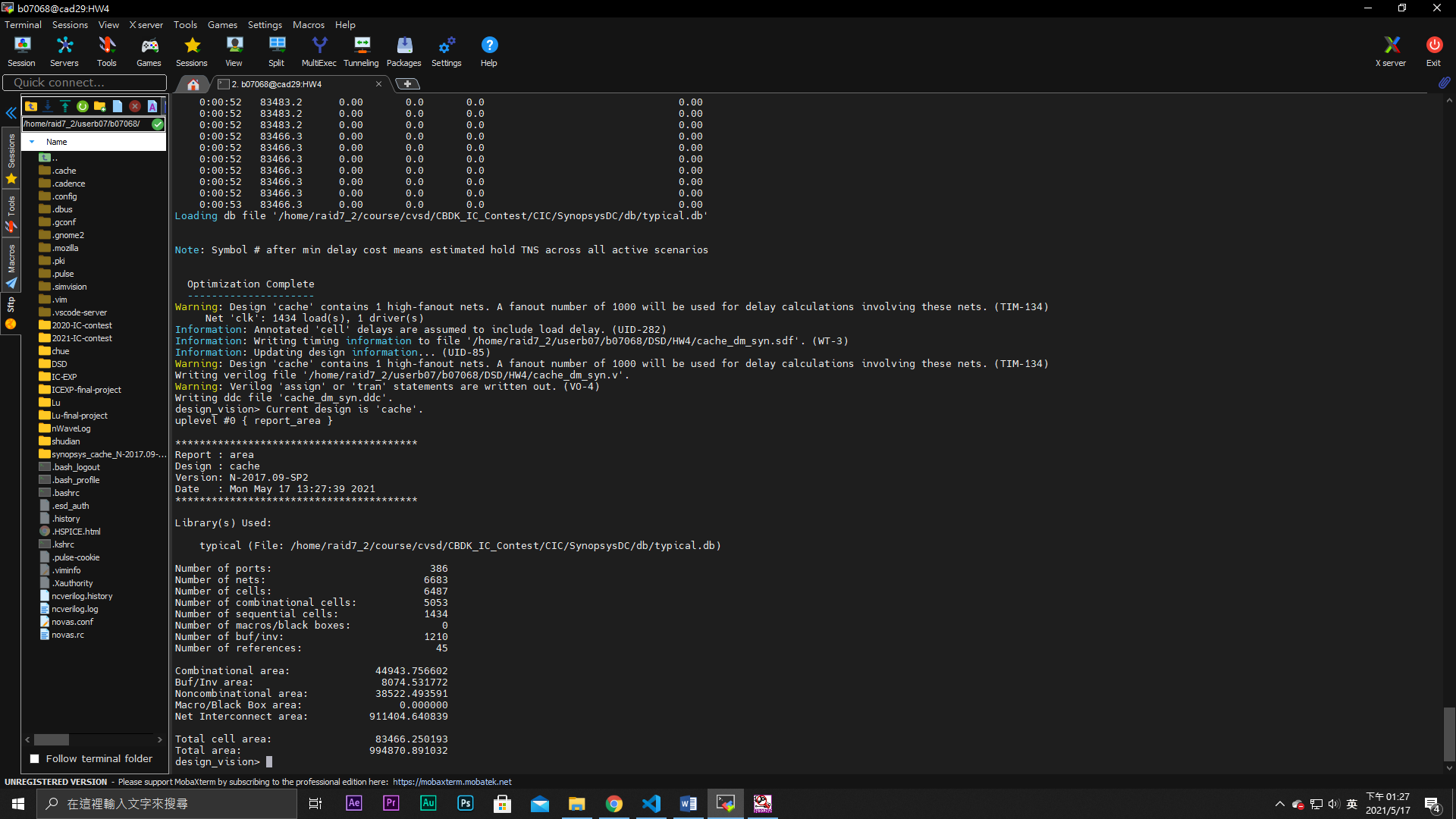


The time save is about **10000ns**, **1000 clock cycle**. Since the testbench in this homework does not produce many “dirty” blocks (only 256 operations induce write-back), there is few needs of writing back. Therefore the timing improvement is not so clear. But I think if the memory access is more random, the improvement will be significantly greater than this amount !!

The additional **area cost** to implement write buffer

**Without write buffer**

**Direct-mapped 2way-associated**



**With write buffer**

**Direct-mapped 2way-associated**

