Project 2
Pipeline + L1 Data Cache

2018/12/12

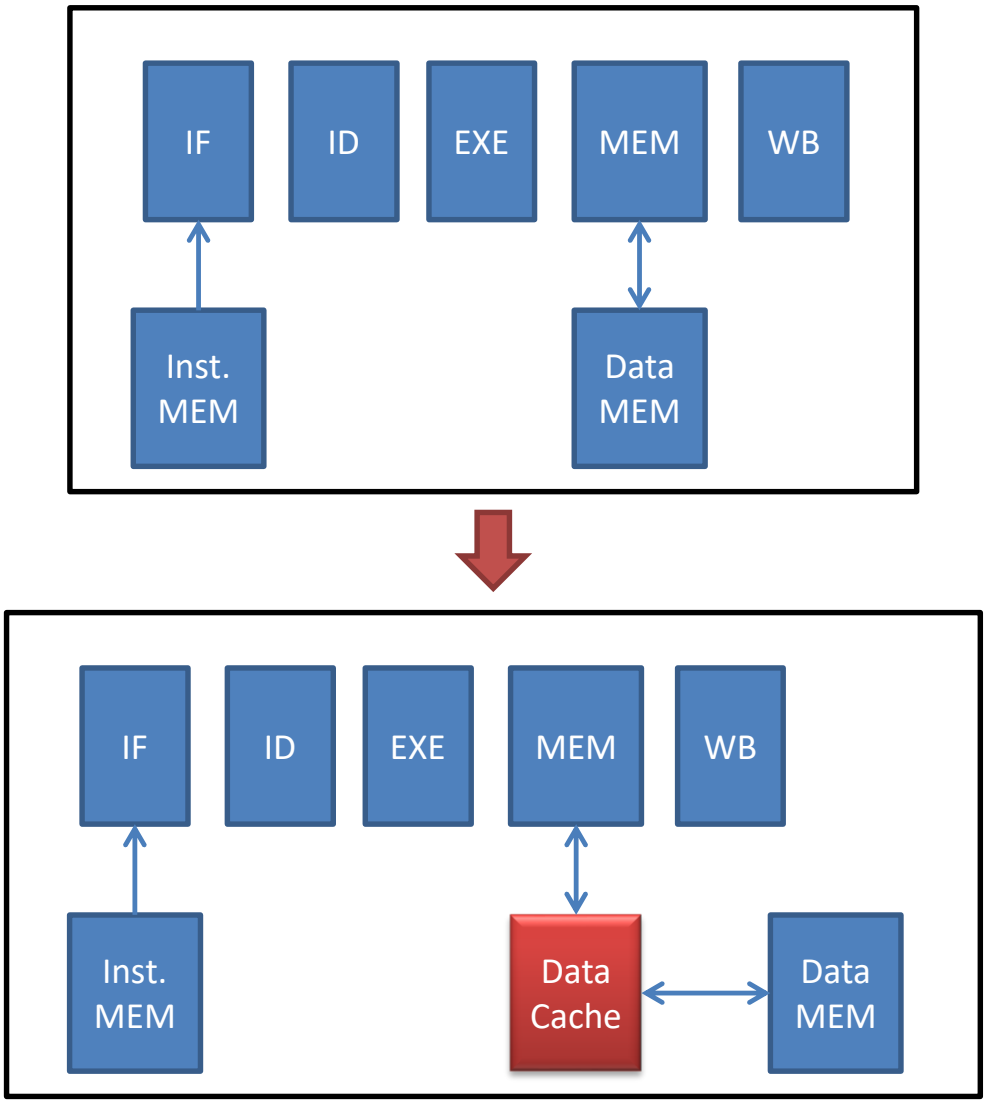
Pipeline with L1 Data Cache

- Write a Verilog behavior Pipeline CPU with L1 Data Cache and off-chip data memory.
  - Size: 16K Bytes
  - Data Width : 32Bytes
  - Memory access Latency: 10 cycle (send an acknowledge when finish access.)
- L1 Data Cache
  - Size : 1KBytes
  - Associative : direct mapped (one-way)
  - Cache line size : 32 Byte
  - Byte-addressable, word granularity (0x00, 0x04, 0x08, 0x0C, …)
  - Write Hit Policy: Write back
  - Write Miss Policy: Write allocate
  - (offset : 5bits, index: 5bits, tag:22bits)

Requirement

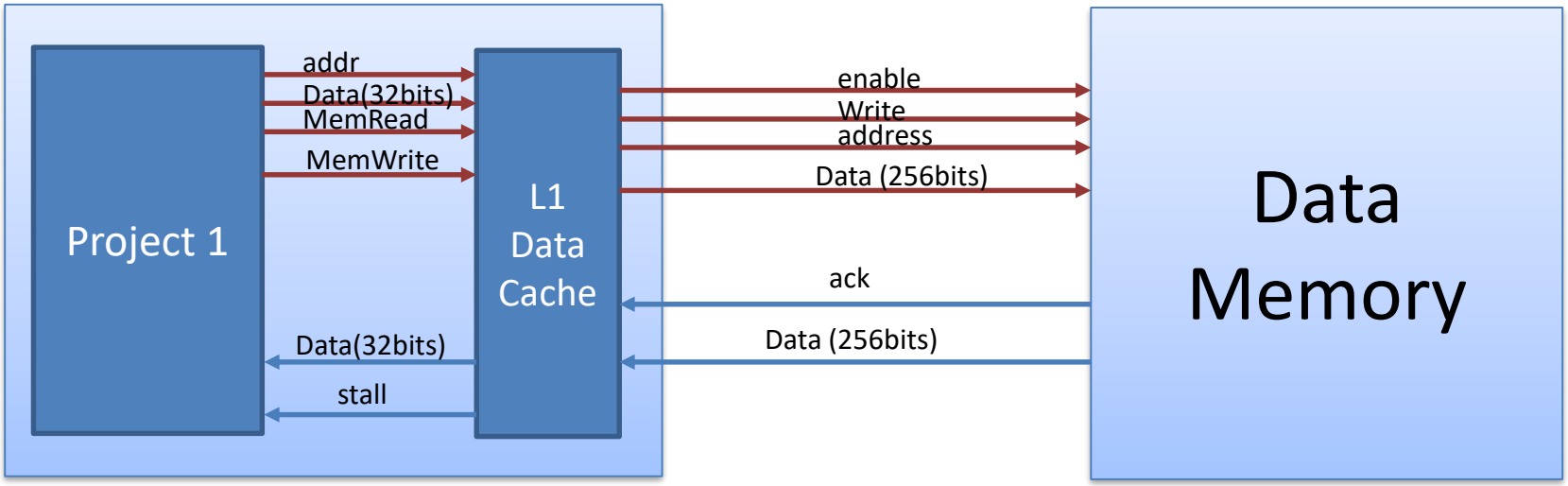
- (80%) Source code (put all .v files into “code” directory)
  - TestBench.v
    - Initialize storage units
    - Load instruction.txt into instruction memory
    - Create clock signal
    - Output cycle count in each cycle
    - Output Register File & Data Memory in each cycle
  - Print result to output.txt
    - Output cache status when memory access occurs.
    - Criteria: We will check the final state is correct or not. (The cycle count does not matter.)
  - Print result to cache.txt
    - Record cache hit or cache miss for each cache access
    - Criteria: We will check the order of hit and miss accesses is identical to the correct answer. (The cycle count does not matter.)
  - Demo(40%)
- (20%) Report (project2_teamXX.pdf)
  - Members & Team Work
  - How do you implement this project
  - Cache Controller in detail (可畫圖說明)
  - Problems and solution of this project
- Put all file and directory into Project2_teamXX, zip all into Project2_teamXX.zip, and upload to NTU COOL
- Due Date: 1/6 Midnight

Project1 to Project2

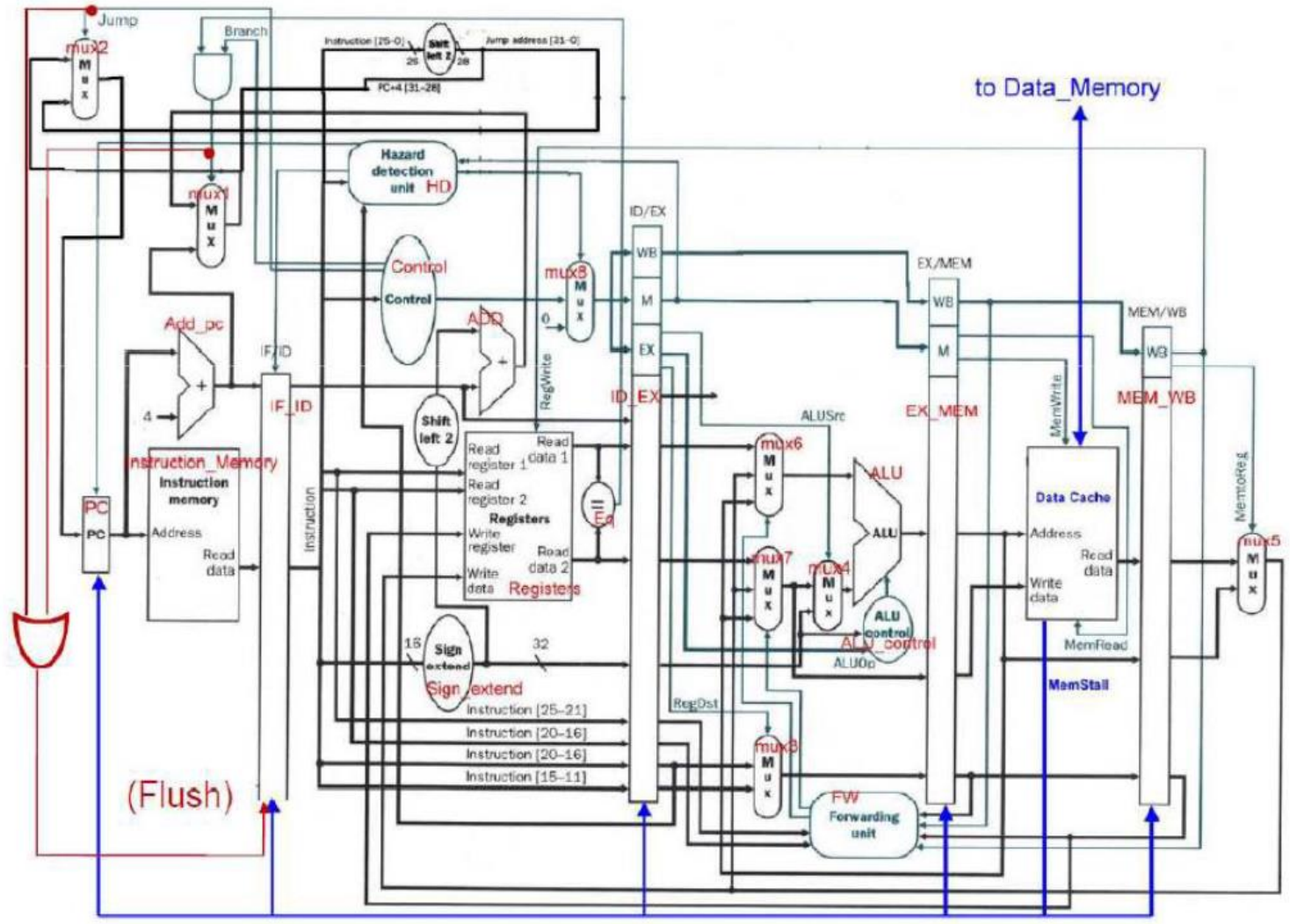


System Block Diagram

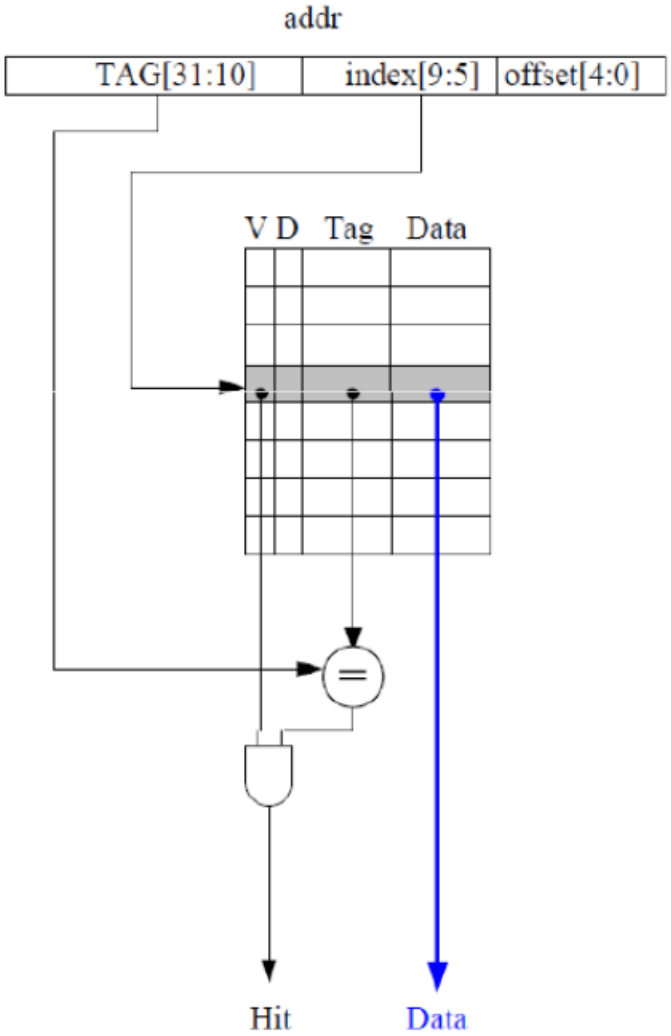
enable : memory access enable
write : write data to memory
ack : memory acknowledge



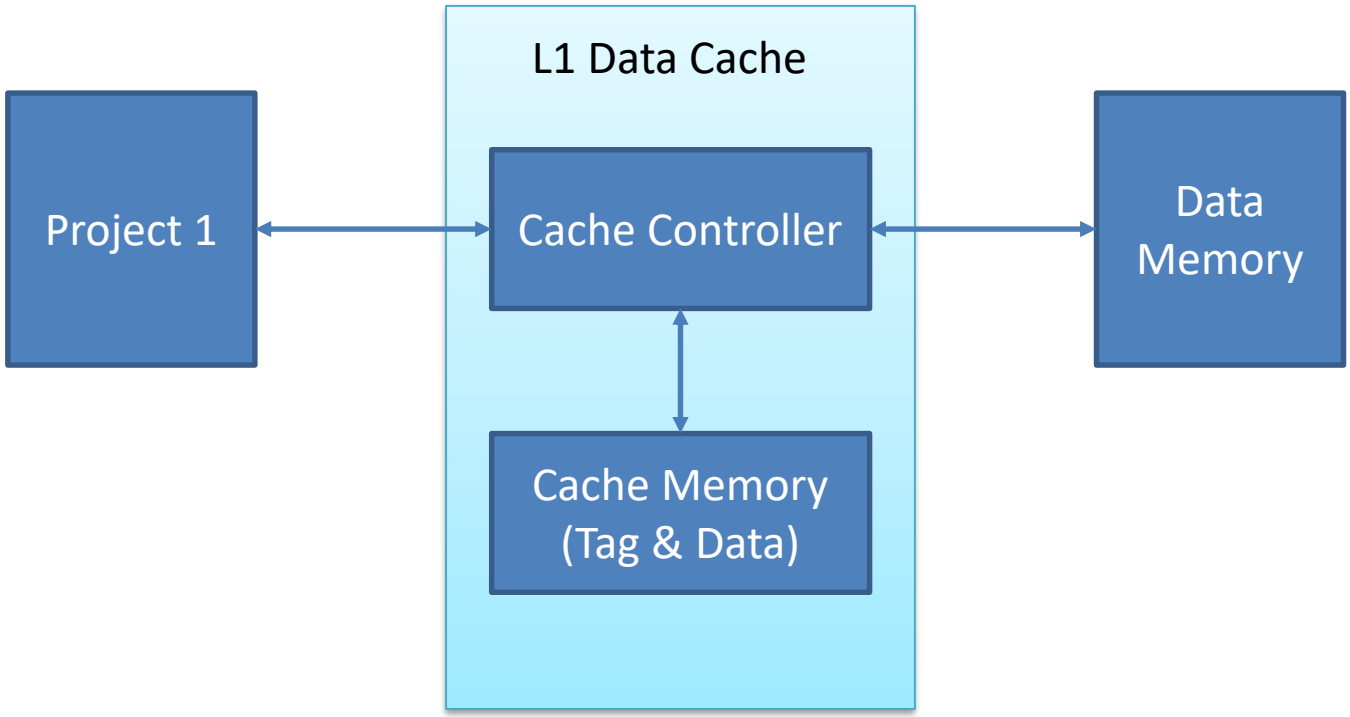
CPU Data Path



Direct Mapped L1 Data Cache



Verilog Example Code



Example file

- CPU.v : need to handle the connect between module
- dcache_top.v : need to implement cache controller
- dcache_data_sram.v
- dcache_tag_sram.v
- Data_Memory.v
- Instruction_Memory.v
- PC.v
- Registers.v
- TestBench.v

You can modify these files if you want.