Project 2: Pipeline CPU + L1 Data Cache

TA: 蔡承佑

r07922182@ntu.edu.tw

Project 1 Demo

- 4 set of test data (basic, forwarding, stall, flush)
- We will download your files and test them in our laptop
- timeslot sheet: https://reurl.cc/gv10ER

Announcement

- 1~3 persons in a group. Please check your group on NTU COOL
- Deadline: 1/1 (Wed.) 14:20

- Demo:
  - Time slot: TBD
  - Execute your program before TA and answer a few questions
  - All members in the group should attend

Requirement

Use Verilog to model pipeline CPU with

- Off-chip Data Memory
  - Size: 16KB
  - Data width: 32 Bytes
  - Memory access latency: 10 cycles (send an ack when finish access)
- L1 Data Cache
  - Size: 1KB
  - Associative: direct mapped (one-way)
  - Cache line size: 32 Bytes
  - Write hit policy: write back
  - Write miss policy: write allocate
  - offset: 5 bits, index: 5 bits, tag: 22 bits

testbench.v

- Initialize registers in all modules
- Load `instruction.txt` into instruction memory
- Create clock signal
- Dump Register files & Data memories in each cycle
- Print result to `output.txt` and `cache.txt`

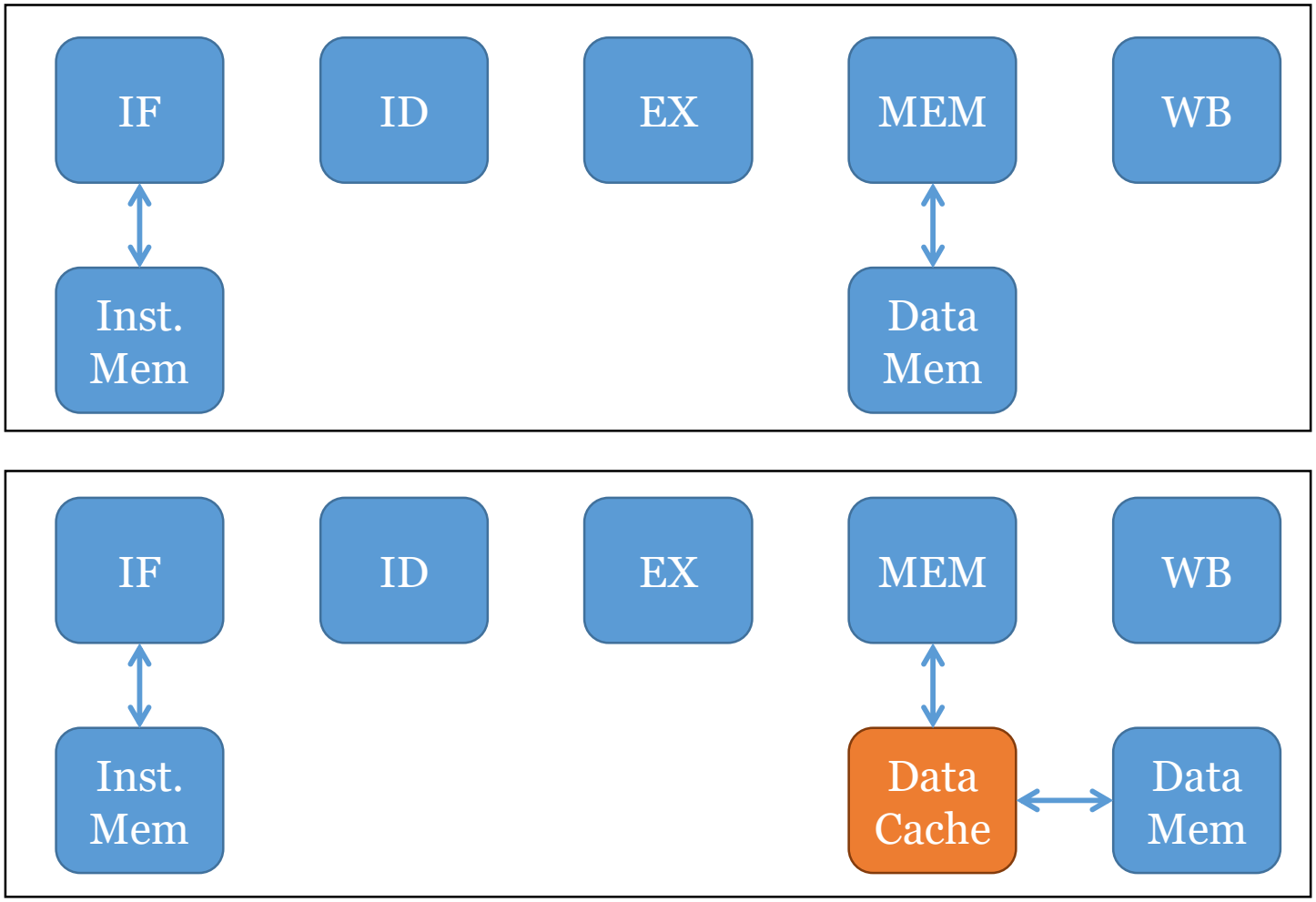
Print results

- 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)

Grading Policy

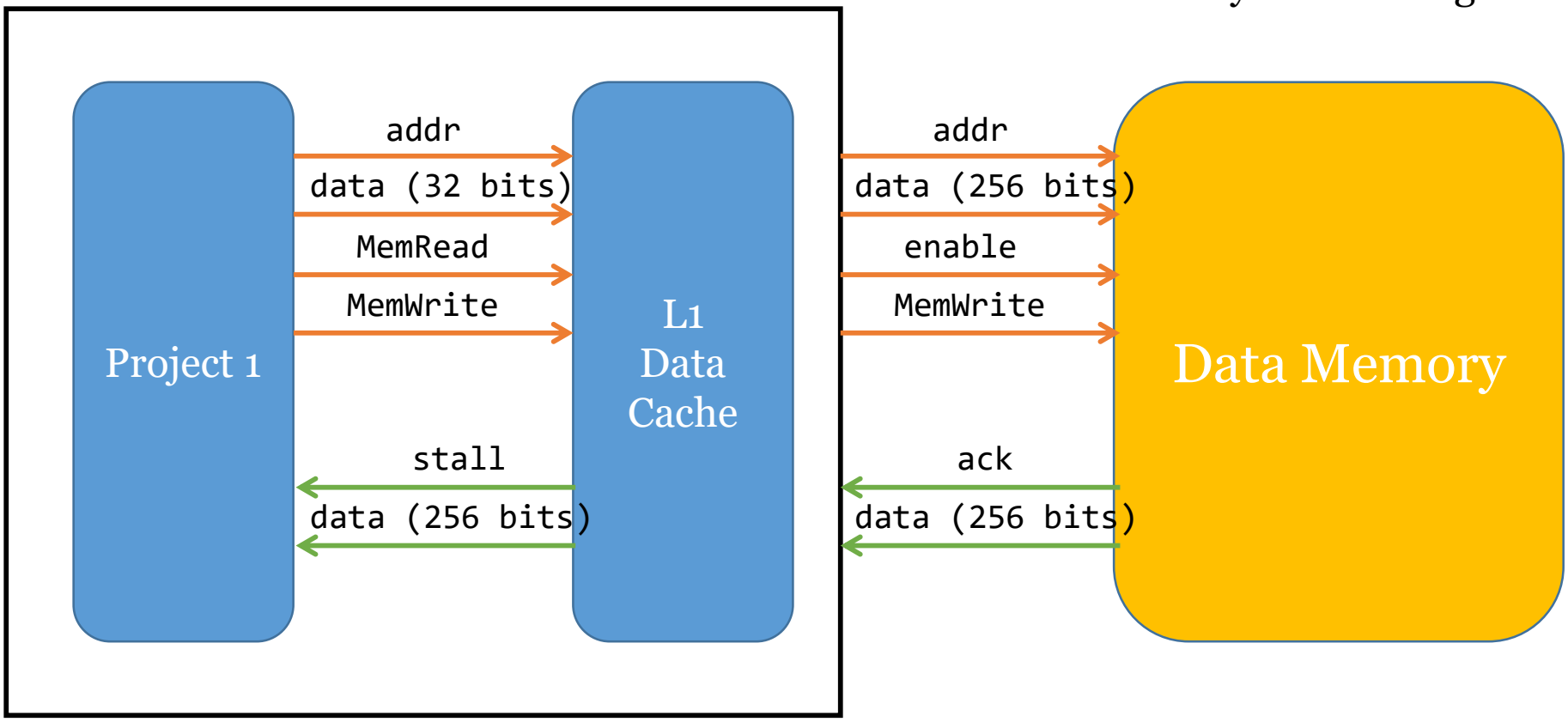
- (80%) Implementation correctness
  - You will get 0 point if your code cannot be compiled
  - Grading at demo. You have to answer several questions about how you implement at demo. You may get 0 point on this part if you cannot clearly answer the questions (regarded as plagiarism)
- (20%) Report
  - Members & Team work (work division)
    - 務必寫組員分工比例
  - How you implement this project
    - Explain in words, not just pasting your code!
  - Cache controller in detail
  - Difficulties encountered and solutions of this projects
- Late punishment: 10 points deduction per day

Project 1 to Project 2

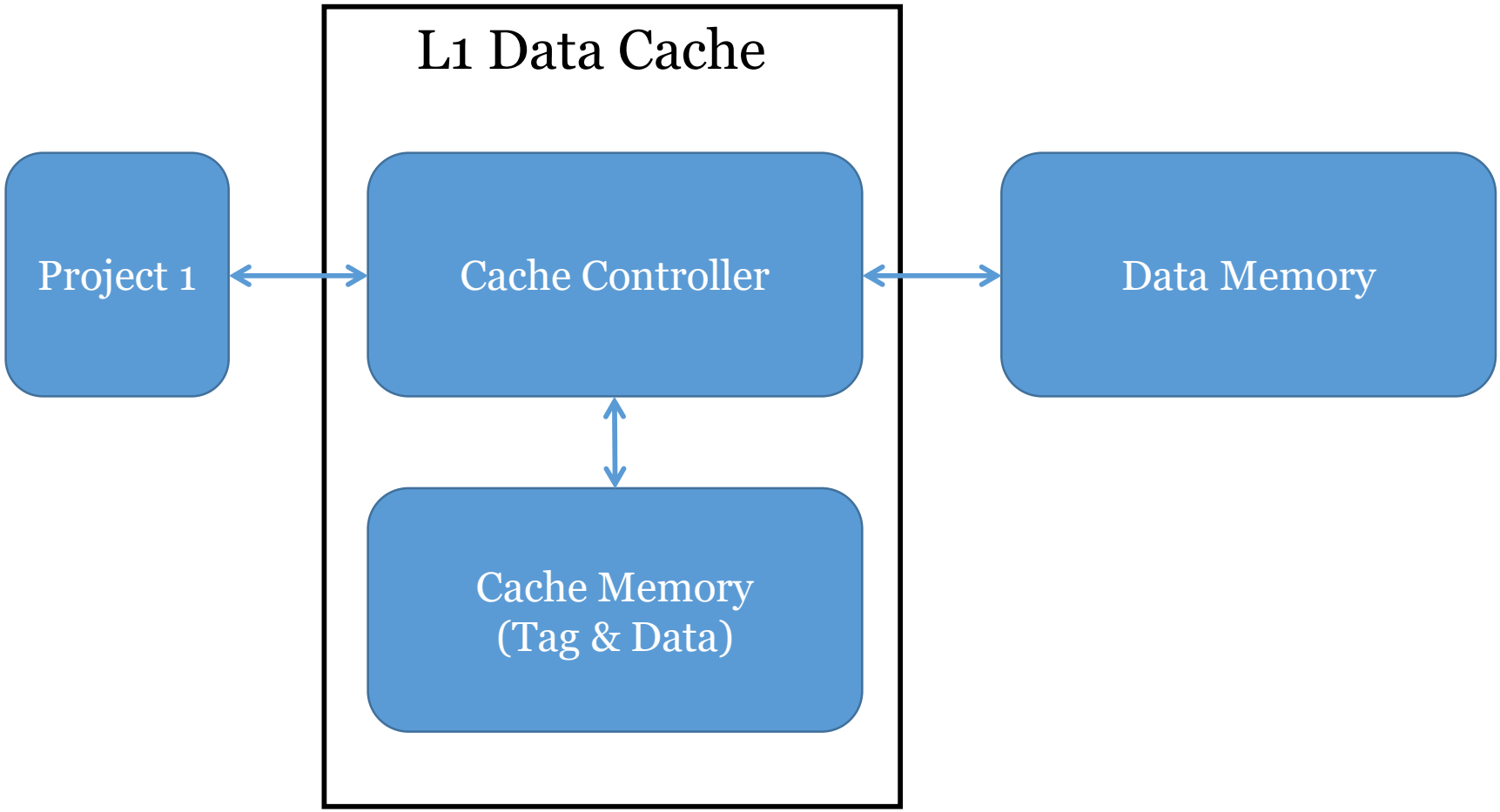


System Block Diagram

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



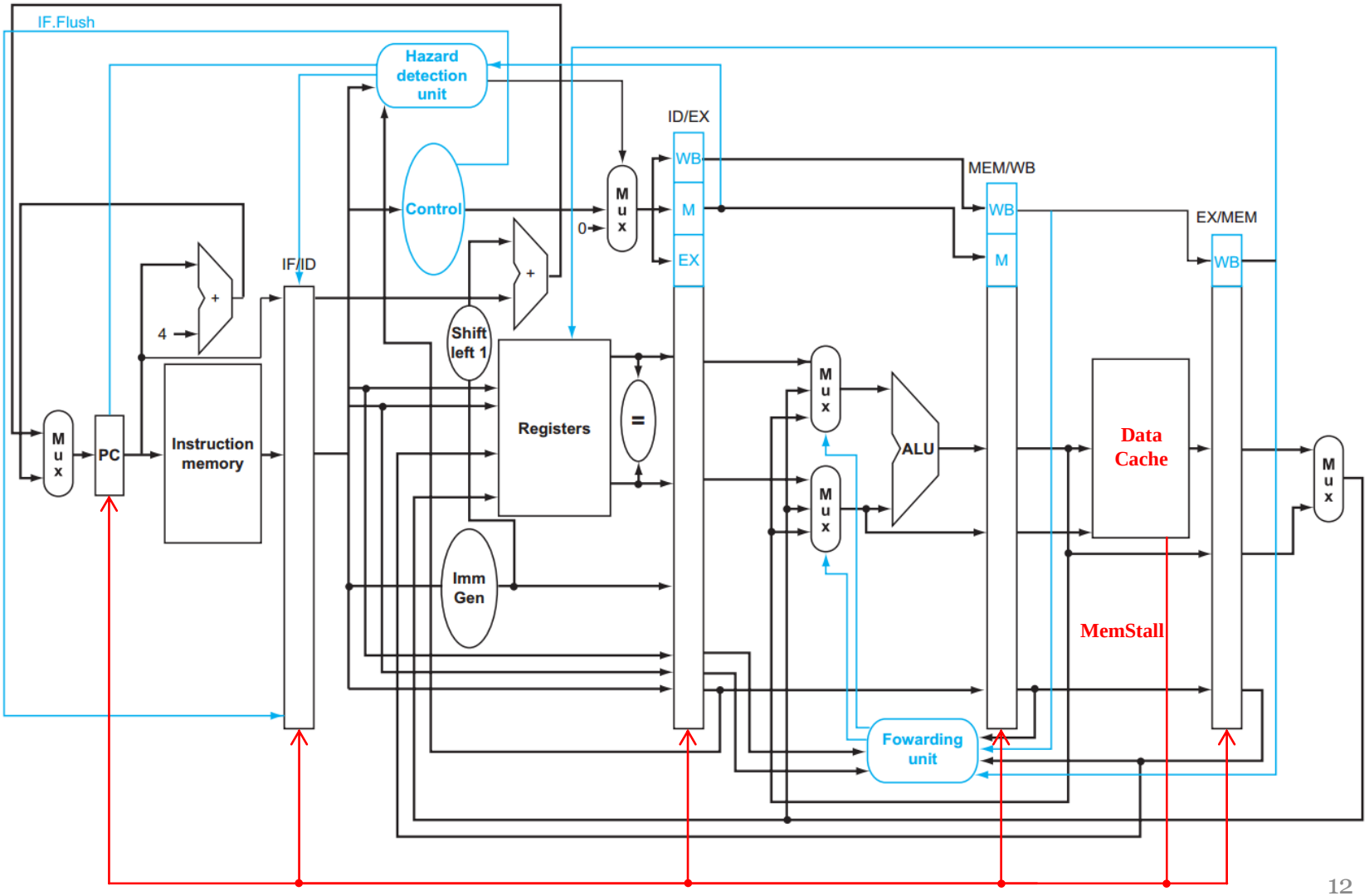
Example files



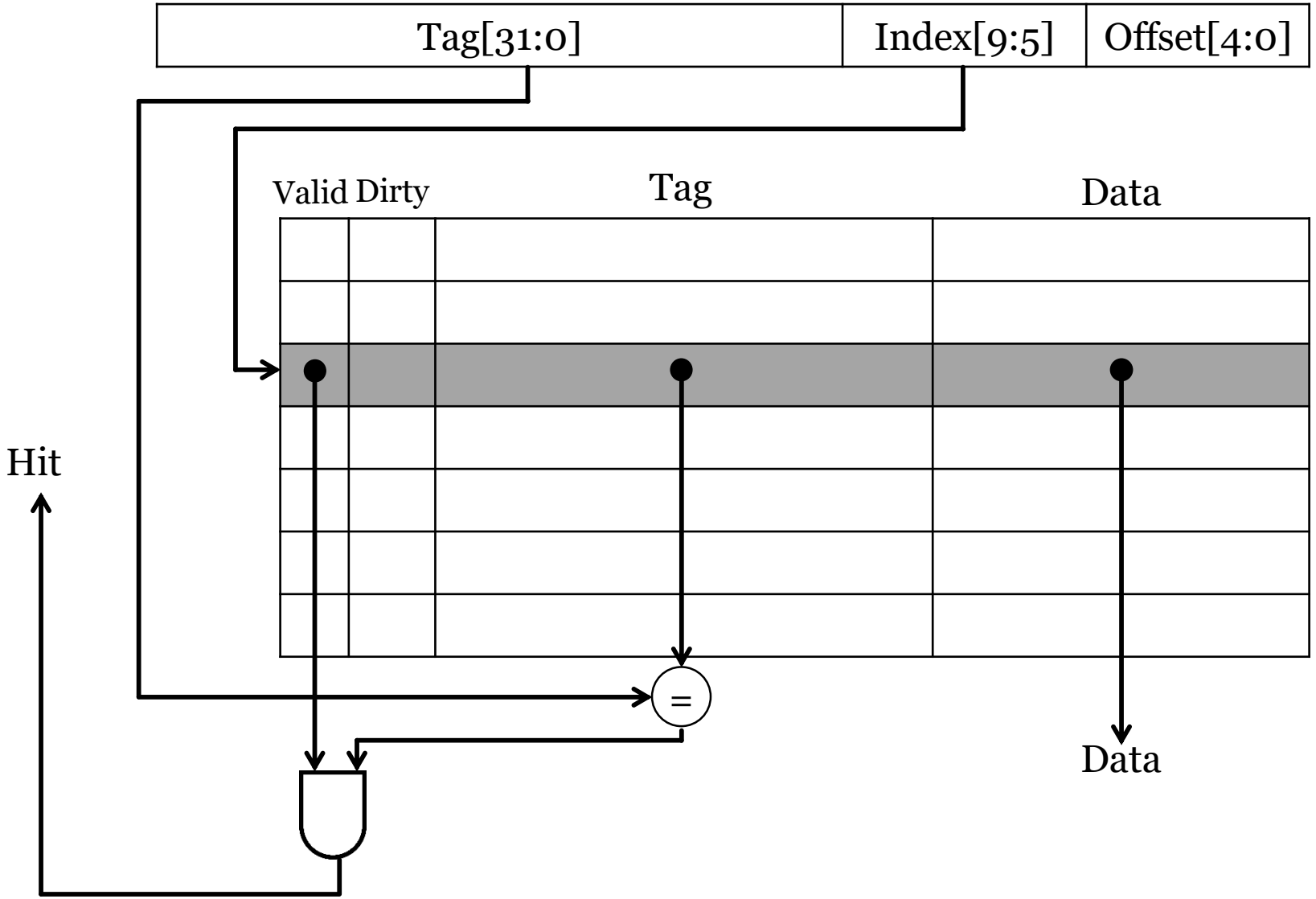
Example files

- CPU.v: connection between modules
- dcache_top.v: implement the cache controller
- dcache_data_sram.v
- dcache_tag_sram.v
- Data_Memory.v
- Instruction_Memory.v
- PC.v
- Register.v
- testbench.v

Datapath & Modules



Direct Mapped L1 dcache



Submission Rules

- `project2_teamXX (dir)`
  - `code/*.v`
  - `project2_teamXX_report.pdf`
- Pack the above **directory** into a zip file
- You can fill in whatever you want in teamXX, but please use only alphabets and numbers.