Lab 03:
HW 2

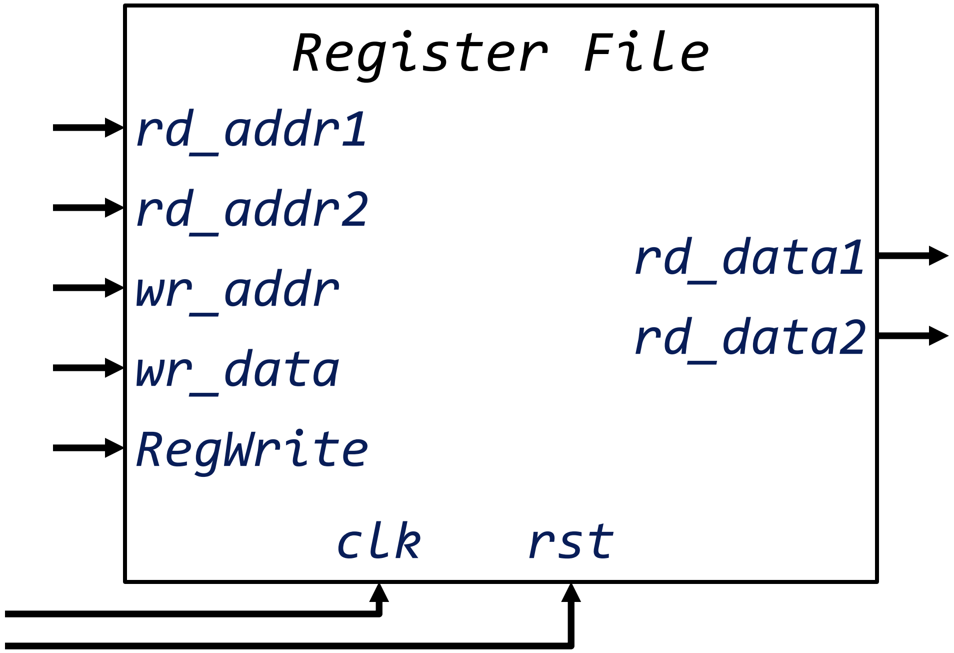
Hunjun Lee
hunjunlee@hanyang.ac.kr

TODOs

- Implement the register file → asynchronous read and synchronous write
- Combine the RF implementation with the ALU implementation in HW1

Specification

- 32-entry 32-bit 2-read / 1-write register file
  - 5-bit two read register ID (`rd_addr1, rd_addr2`) and one write register ID (`wr_addr`)
  - 32-bit two read data (`rd_data1, rd_data2`)
  - 32-bit one write data (`wr_data`)
  - 1-bit one write enable signal (`RegWrite`)
  - 1-bit clock (`clk`) & 1-bit synchronous reset signal (`rst`)



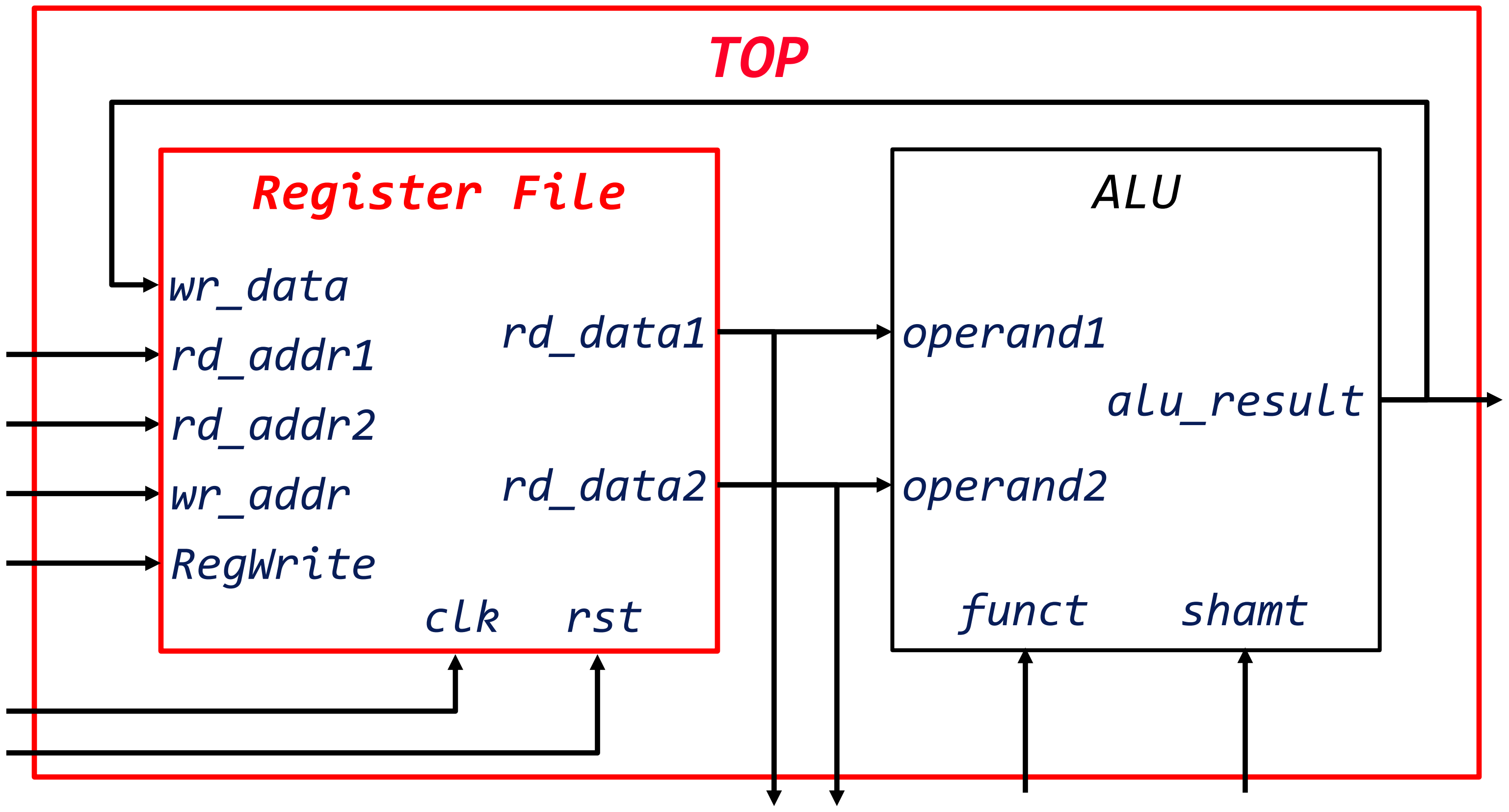
Timing Issues!

- Read happens asynchronously
  - The `rd_data1` and `rd_data2` change as soon as `rd_addr1` and `rd_add2` change

- Write happens synchronously
  - `write_data` is written to the register file at the positive edge of the clock

Combining ALU + RF

- Integrate RF with ALU to provide operands to the ALU and write alu_result to the register file



Assignments

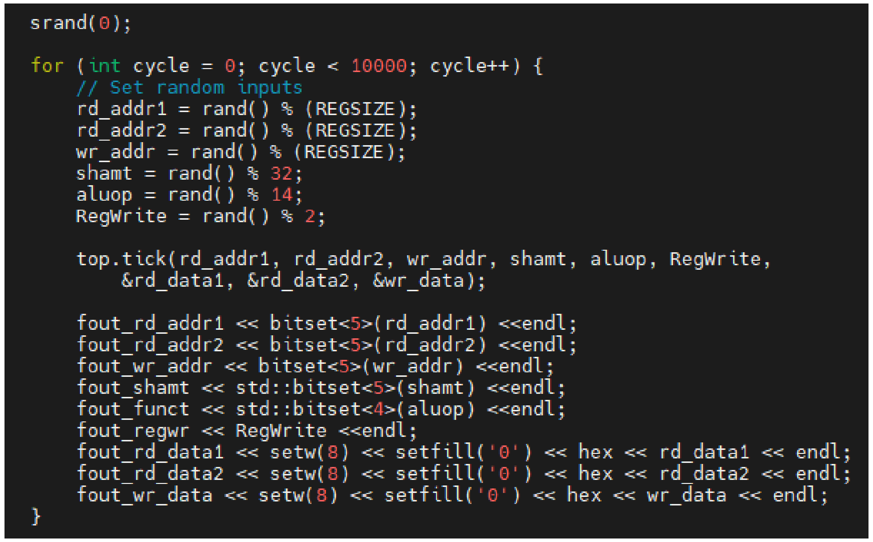
- **CPP / Verilog simulator**
  - **`global.h + GLOBAL.v`**
    - `Include constant values`
  - **`ALU.cpp/h + ALU.v`**
    - `You can copy & paste the code from HW1`
  - **`RF.cpp/h + RF.v            (FIXME)`**
    - `Implement your own RF code`
  - **`TOP.cpp/h + TOP.v          (FIXME)`**
    - `A HW module that includes ALU + RF`
  - **`main.cpp + TOP_tb.v`**
    - `A testbench to test the TOP module`

You are allowed to modify both header / cpp file if you need

Debugging CPP / Verilog

- I do not have a reference test files for this HW. But you have your fully debugged ALU ➔ Use it to make your own test files

- Again, I made a code in the main.cpp to generate a testbench for your Verilog code ➔ Use it to debug your Verilog files



```
srand(0);

for (int cycle = 0; cycle < 10000; cycle++) {
    // Set random inputs
    rd_addr1 = rand() % (REGSIZE);
    rd_addr2 = rand() % (REGSIZE);
    wr_addr = rand() % (REGSIZE);
    shamt = rand() % 32;
    aluop = rand() % 14;
    RegWrite = rand() % 2;

    top.tick(rd_addr1, rd_addr2, wr_addr, shamt, aluop, RegWrite,
        &rd_data1, &rd_data2, &wr_data);

    fout_rd_addr1 << bitset<5>(rd_addr1) <<endl;
    fout_rd_addr2 << bitset<5>(rd_addr2) <<endl;
    fout_wr_addr << bitset<5>(wr_addr) <<endl;
    fout_shamt << std::bitset<5>(shamt) <<endl;
    fout_funct << std::bitset<4>(aluop) <<endl;
    fout_regwr << RegWrite <<endl;
    fout_rd_data1 << setw(8) << setfill('0') << hex << rd_data1 << endl;
    fout_rd_data2 << setw(8) << setfill('0') << hex << rd_data2 << endl;
    fout_wr_data << setw(8) << setfill('0') << hex << wr_data << endl;
}
```

Submission

- RF implementation + TOP module that includes both RF and ALU

- Submission (Zip all the files)
  - For a two-people team: lab2_student_id1_student_id2.zip
  - For a one-person team: lab2_student_id.zip
    - You must follow the format (-10% for wrong file format)
    - Example:
      - lab2_2020102030.zip
      - lab2_2020102030_2022103040.zip
  - The zip file should contain:
    - ALU.cpp/h, ALU.v, RF.cpp/h, RF.v, TOP.cpp/h, and TOP.v
    - lab2_report.pdf

Submission

- You need to write 4-page report for each submission
  - Explain the overall structure and how you implemented each program
  - Write down the difficulties if you had one
  - Draw the hardware modules if needed …

- Due: Fri. April 4th
  - 1 week delay: -20%
  - 2 week delay: -50%
  - Further delay: 0 point