**Project 2 Design Documentation**

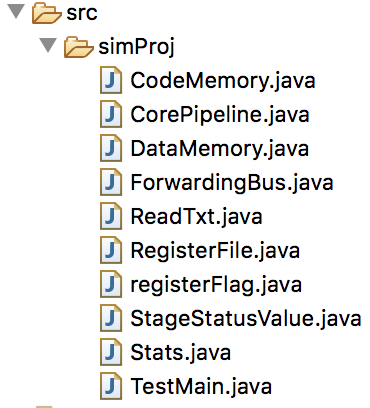
**Computer architecture and organization (CS 520)**

**Summary:**

Base on the first project, project 2 contains several new parts/functions:

1. Output dependence
2. Forwarding Bus
3. DIV FU (4 cycle pipeline)
4. JAL instruction
5. Improve Register File to register class. Each register is an object of this class with ready flag. So, the stalled mechanism in DRF is more reasonable.
6. Bit operation instructions (AND, OR, EX-OR) do not affect PSW zero bit any more.
7. Out of order completion
8. HALT instruction goes through DIV (4 stages) path.

The new structure in my design:



**Note:** How to use project 2 program?

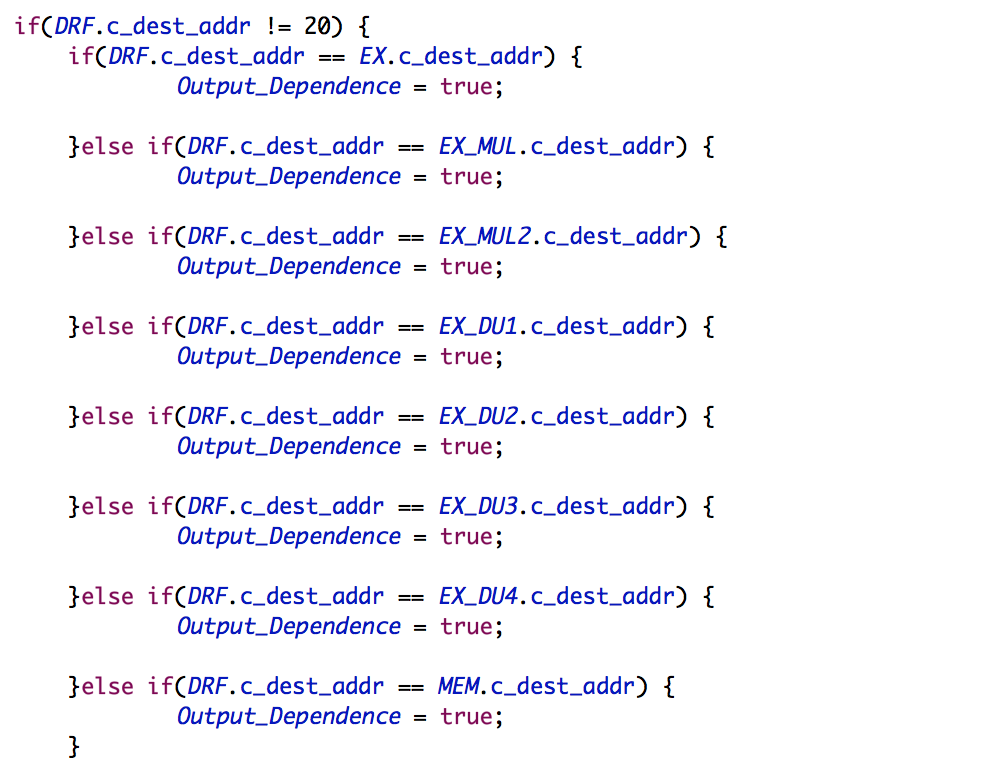
For example, in my case, put the txt test file in this path first.

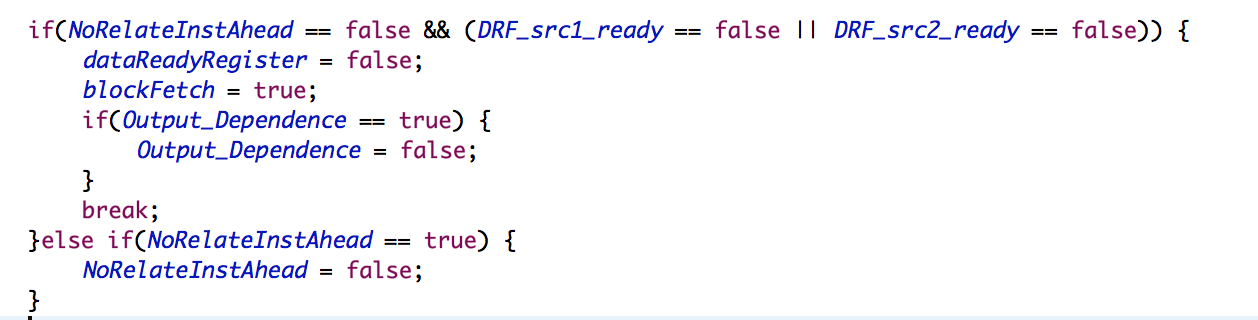
Then RUN the program. The rest operation should be clear, just follow UI guide.



1. **Output Dependence**

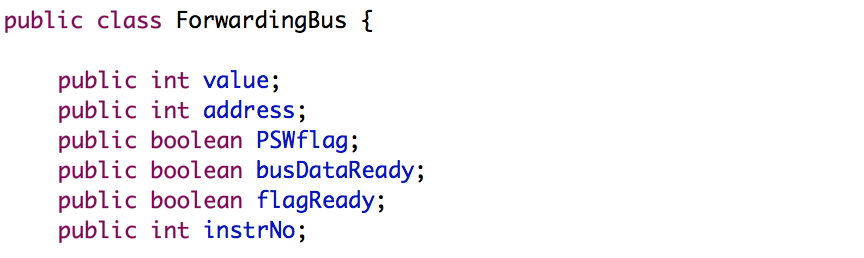
Compared the destination register which belonged to instruction in DRF stage to each stage’s destination register, if it has. Then using the comparing result to figure out if the instruction in DRF need to be stalled for Output dependence. (There is no need to handle anti-dependencies – a later instruction cannot overwrite the sources of a later instruction as **dispatches/issues take place in program order** from D/RF in the pipeline for Project 2)



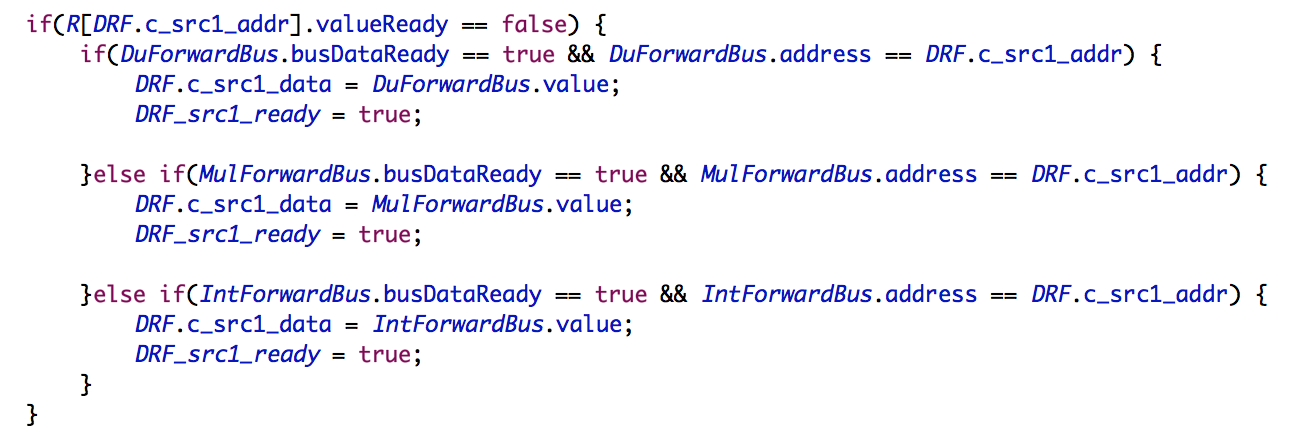


1. **Forwarding Bus**

Create a class named ForwardingBus. This class can be used to realize data forwarding and zero flag forwarding.

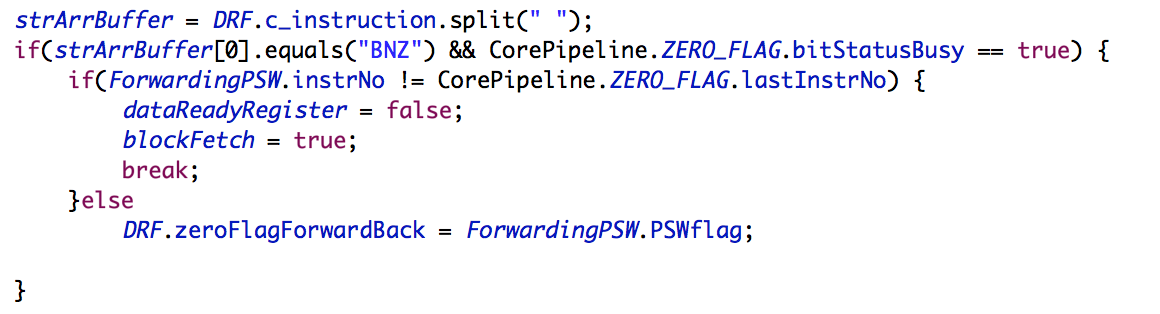
****

When source register is not ready, instruction in DRF stage can use related data in forwarding bus.

****

Same logic, branch instruction can use the related zero flag in forwarding bus too if the instruction in

DRF stage.

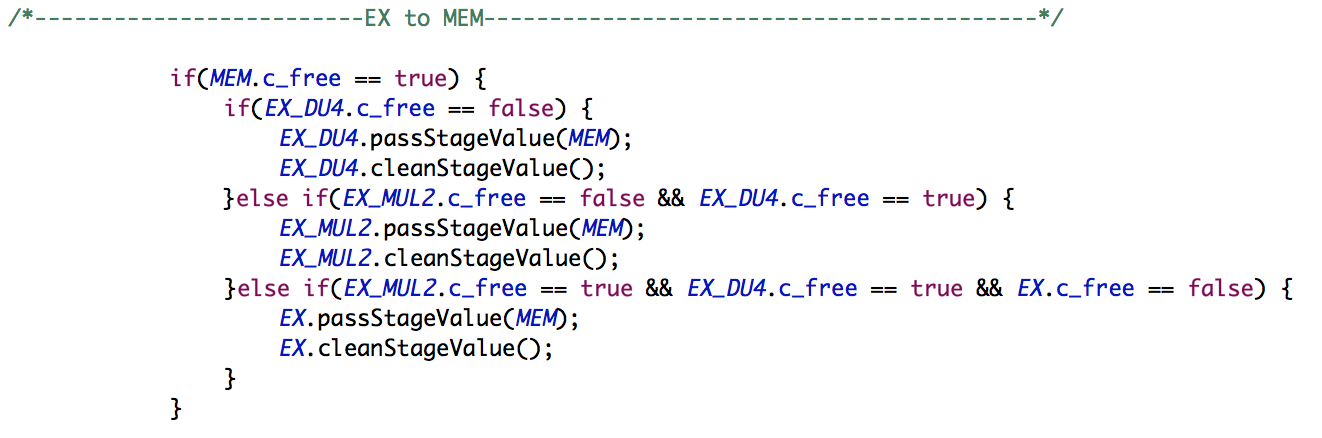


1. **DIV FU**

No other special in DIV stages, just follow design in MUL stages.

When multiple function units contend to use the MEM stage, the priorities for using the MEM stage are as follows: DU (highest), MULtiply FU, IntegerFU (lowest).

Which part should be passed to MEM stage need to be considerd.



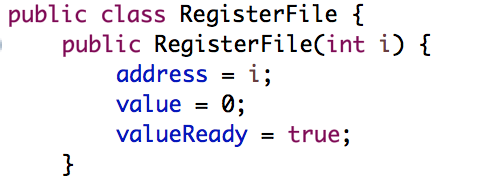
1. **JAL instruction**

JAL instruction can be considered as parts of “Load instruction” plus “Jump instruction”. So, most of the new code are copy from these two parts. Each part which has Jump related code should be add JAL code. No other special.

1. **New register file structure**

In my first design, I did not use the register ready flag mechanism. It can work because that the instruction flow is relatively easy in project 1. But in project 2, if I still use the old register file mechanism, the logic about the instruction flow would be complex.

In second project, I use register which contains ready flag instead of simple integer array in project1.



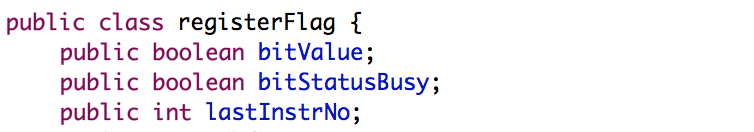
And I use array to store these register object, it is easy to use.

****

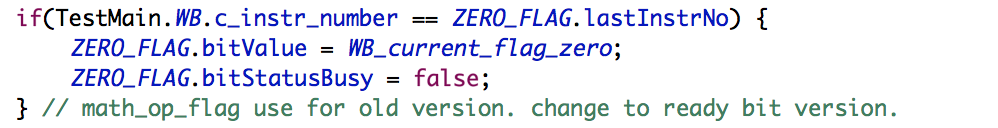
So, in project 2, stages/instruction can read related register when the register is ready to read.

1. In project2, only ADD, SUB, MUL, DIV can update the PSW bit. So, the update mechanism about bit operation instruction such AND, EXOR, OR should be deleted.
2. There are two parts of out of order completion should be handle. First part is output dependence, just follow A part.

The second part is PSW update mechanism, which related to forwarding bus mechanism. In my design, in order to make that only the last math instruction which is most close to branch instruction (BZ, BNZ) has right to update the PSW flag in WB stage, it is necessary to set a variable named lastInstrNo in PSW register, which can hold the value of the last math instruction number before the branch instruction.

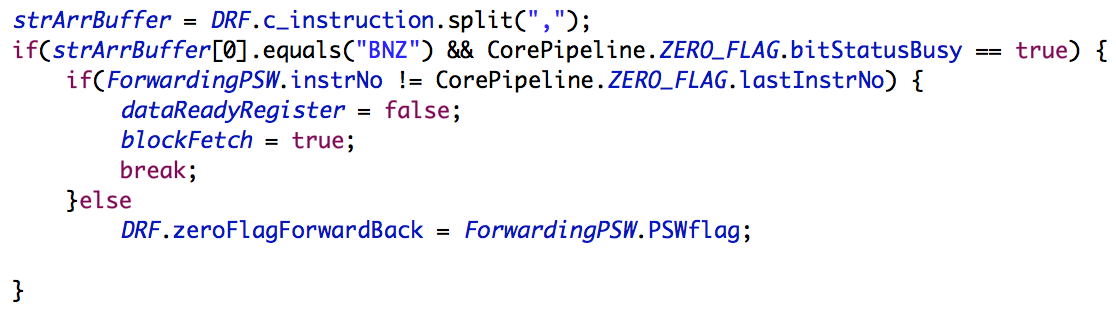
****

When the math instruction is in WB stage, it can update the PSW register only if the instruction number in WB equals to the last instruction number saved in PSW register.

****

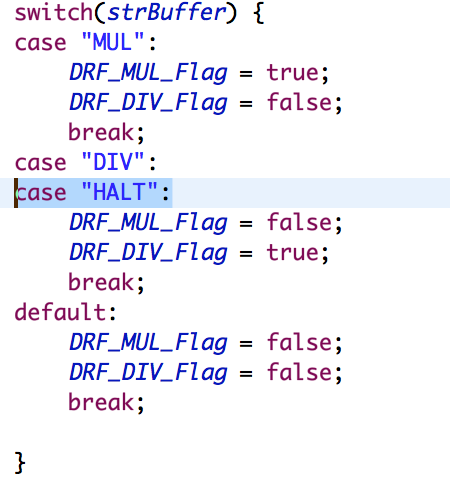
Same logic in forwarding bus, when a new branch instruction in DRF, it need to know which is the math instruction has right to update PSW, in other word, this math instruction is the one which most close to the new branch instruction.

When the branch instruction is in DRF stage, it can use the zero flag from forwarding bus only if the instruction number on forwarding bus was equal to the instruction number in PSW register.

****

1. HALT instruction should be go through the path which needs the most quantity of cycles. In this way, other instructions which before HALT instruction will have enough time to finish the all operation.

In project 2, just pass the HALT instruction from DRF stage to DIV1 stage.

****