SIL 618
Computer Architecture
Minor Exam, Sem I, 2021-22

Ritik Jain
2021JCS2260

20 September 2021

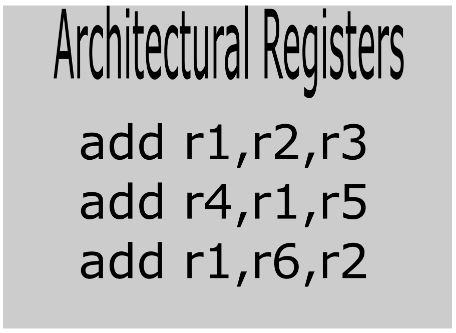
Question 1

Marks: 10

- Renaming is removed with the help of physical registers that are completely internal to the processor and invisible to the compiler or the programmer.
  We generally take more physical registers than architectural registers. We map architectural registers to a set of physical registers. Thus we can remove WAR and WAW hazards. Having some sufficient register eliminates the hazard.

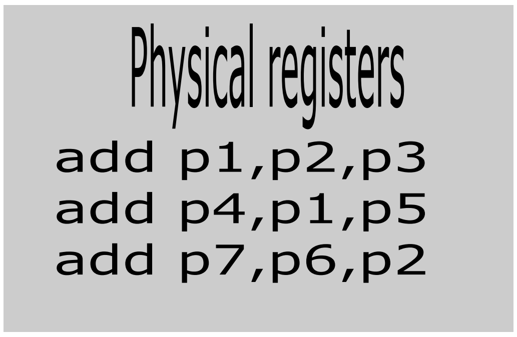
- WAR and WAW occur due to a limited number of architectural registers.
  If we have a very large number of architectural registers then renaming will not be required because we can use different sets of architectural registers for different instructions. Hence, no renaming is required.
  In the example,we have one WAW and WAR dependency between Instruction 1 and instruction 3 AND Instruction 2 and Instruction 3 respectively. We replace r1 by p7 and eliminates the dependency.

- RAW dependency cannot be removed with the help of renaming because it is a true dependency. In the example below, we have removed the WAR and WAW using physical registers mapping but we cannot remove RAW Instruction 2 needs the updated value of r1 from Instruction 1. If we changed to another physical register let's say p8 then p8 still needs to wait till p1 generates the right value in the WB stage.



Architectural Registers

```
add r1,r2,r3
add r4,r1,r5
add r1,r6,r2
```



Physical registers

```
add p1,p2,p3
add p4,p1,p5
add p7,p6,p2
```

Question 2

Marks: 5

- Register Windows: We know that registers are fixed when the hardware is designed. So we cannot add additional registers. Different programs require different sets of registers for different purposes. A register window is a set of registers that are required by a particular instruction can use. It is used to avoid unnecessary costly spills by the compiler.

- Example- Interrupt Handlers generally require some special registers such as OldPC,oldSP,oldFlags, flags, and some general registers r0 and sp.

- Here the registers window of the Interrupt handler consists of 6 registers.Registers window speeds up the implementation of functions because of fewer register spills and fills operations because the register window is rarely overflowed.

- Disadvantages: It increases the complexity of the program because of additional registers.CPU needs to maintain the active state for a function and the dead state of these registers after the function is completed. In recent years, more time is allocated to compiler register allocation and have used the existing registers more judiciously.

Question 3

Marks: 6

- A) Pipeline generally requires storing the intermediate values. In hardware, the output of each stage is stored using latches. Latches are used to store the instruction packet, control signals, intermediate results between pipeline stages. They are connected to a common clock.

- B)1) We can use an additional bit to tell whether the output is reached after the negative edge or not. If the output reaches instantaneously the additional bit will be set to zero and in the normal cases, it will be set to 1.

  2) We can add additional buffers-Add an additional buffer where the hold time constraint is violated. It will increase the additional path. This reduces the hold time.

  3) Designers make sure that hold time and setup time are small fractions of the total cycle time.

Question 4

- Structural Dependency- It refers to the possibility of instructions not being able to execute because of resource conflicts and constraints. Cases when it arises:

  Marks: 1

  - The use of the functional unit requires more than 1 clock cycle. It occurs when the functional unit is not pipelined properly.

  - Resource shared between stages
  Instruction 1:St r1, 10, [r2]
  Instruction 2:Add r3, r4, r5
  Instruction 3:Sub r6, r7, r8
  When Instuction 1 is in MA Stage during the 4th cycle and Instruction3 is in OF stage then the resource conflict condition arises due to memory constraint.

  Instruction 1 and 3 are not sharing any memroy.

- Solution: Use additional hardware up to a certain limit or stall the cycle.

- Data Dependency: Occurs when consumer instruction depends on data from a producer's instruction.

  - Example:

  In which instruction, memory is being accessed in OF stage??

  ld r1,2[r2]
  add r3,r4,r5
  Sub r6,r7,r8
  Add r7,r1,2

  Here RW-OF forwarding path will be required because producer instruction is in the RW stage and consumer Instruction is in the OF stage.

- We cannot remove all dependencies using forwarding in case of Load-Use hazard. Hence we need to stall the cycle for once.

  - Example:
  ld r1,4[r2]
  Add r2,r1,r3

Question 5 Marks: 8

We know that exception is precise if the following conditions are met:

1)Let I will be the last instruction in the program that completes before the first instruction in the exception handler completes its execution.

2) no instruction after the I completes before the exception completes.

3) After the exception completes, we can start from I or I+1 seamlessly.

When the exception arrives, we know that atmost 5 instructions will be present in the pipeline.
We can mark one of the instructions so that the above conditions are satisfied. Here the exception occurs during the WB stage so we can mark the WB stage but there is a catch that the store finishes in the MA stage.
If the next instruction is a store just after the exception it will finish. Hence, condition 2 is violated. So we need to take care of one exception that is a store instruction. So we mark the instruction in the MA stage.
If the next instruction(PC+4) is the store we need to insert a bubble so that avoids its completion. Otherwise do the execution properly.
We also add an exception unit whose role is to process interrupts and exceptions.
Instructions fetched after the marked instruction need to be converted to bubbles. After the exception completes load the old PC value and start execution.