Computer Architecture

Lab Sheet 9 – Pipeline

Semester 1 – 2023 – 24

Ripes Simulator

You can access the ripes simulator from the following link:

https://ripes.me

Example code to understand the pipeline without any hazards

```
li x1, 1
li x2, 2
li x3, 3
li x4, 4
li x5, 3
li x6, 4
nop
nop
nop
nop
nop
add x1,x1,x2
add x5,x5,x3
add x6,x6,x4
```

Run the above code in ripes with and without operance forwarding with hazard detection.

Q1) **Double Data Hazard**

Paste the following code into the `Editor` window of Ripes:

```
li x1, 1
li x2, 2
li x3, 3
li x4, 4
nop
nop
nop
nop
```

```
nop
add x1,x1,x2
add x1,x1,x3
add x1,x1,x4
```

Go to the `Processor` window and step until there are `nop`'s in all pipeline stages. Now step whilst paying attention to the state in each stage. Notice how the destination register after the EX stage is forwarded.

A 5-stage in-order processor with hazard detection/elimination but no forwarding unit.

- Check how many cycles it takes?
- The number nop instructions or the delay in the pipeline

A 5-stage in-order processor with hazard detection/elimination and forwarding.

- Check how many cycles it takes?
- The number nop instructions or the delay in the pipeline

Q2) **Load Use Hazard**

Paste the following code into the `Editor` window of Ripes:

```
.data
nr:
        .word 16
.text
        la x1, nr
        lw x2, 20(x1)
        and x4, x2, x5
        or x8, x2, x6
        add x9, x4, x2
```

While stepping through the program, notice how the `and` right after the `lw` needs the loaded value, so the processor inserts a `nop` in between. When the `lw` reaches the WB stage, the value is written to the register file, but is also forwarded to the `and` in the EX stage.

A 5-stage in-order processor with hazard detection/elimination but no forwarding unit.

- Check how many cycles it takes?
- The number nop instructions or the delay in the pipeline

A 5-stage in-order processor with hazard detection/elimination and forwarding.

- Check how many cycles it takes?
- The number nop instructions or the delay in the pipeline

Q3) **Branch Hazard**

Paste the following code into the `Editor` window of Ripes:

```
.data
nr:
        .word 16
.text
  la x7, nr
        li x1, 1
        li x3, 1
        nop
        nop
        nop
        nop
        nop
        sub  x10, x4, x8
        beq  x1,  x3, jmp
        and  x12, x2, x5
        or   x13, x2, x6
        add  x14, x4, x2
        sub  x15, x6, x7
        nop
        nop
        nop
jmp:
        lw   x4, 0(x7)
```

Step until the pipeline is filled with `nops`. Then step while noticing the behaviour of `beq` as the branch is taken. During the ID stage the new PC is correctly calculated. However, since the PC register takes a cycle to be updated, the `and` instruction after the `beq` is fetched. In the next cycle,

the PC contains the right address so the the `lw` instruction is fetched. A `nop` is inserted into the ID stage to prevent the `and` from polluting the pipeline.

A 5-stage in-order processor with hazard detection/elimination but no forwarding unit.

- Check how many cycles it takes?
- The number nop instructions or the delay in the pipeline

A 5-stage in-order processor with hazard detection/elimination and forwarding.

- Check how many cycles it takes?
- The number nop instructions or the delay in the pipeline