Bilkent University

Computer Science

CS224

Design Report

Lab 5

Section 4

Mehmet Hasat Serinkan

21901649

02.12.2022

B)

+ RTL expression for sracc:

IM[PC]

RF[rd] ← RF[rd] + [RF[rs] >>> RF [rt]]

PC ← PC + 4

+ Table of Hazards

Hazard Type	Hazard Name	Pipeline Stages Affected
Data Hazard	Compute-Use	Decode Stage
Data Hazard	Load-Use	Execute and Memory Stages
Data Hazard	Load-Store	Memory Stage
Control Hazard	Branch	Memory Stage

C) **Solutions**

Data Hazards:

1- Solution for Compute-Use: This type of hazard occurs when the result of a former instruction has not been written to register file and a subsequent instruction is reading the data of that register. Forwarding is the best solution for Compute-Use Hazard. After the execution stage, the value of the register can be forwarded to the next instruction's execution stage.

2- Solution for Load-Use: This type of hazard occurs when the next instruction cannot read data because the previous instruction have to read data from data memory, to do that, it must reach the end of memory. The best solution for Load-Use is stalling. Stalling for one clock cycle will solve the hazard. Execute stage will be delayed until correct data is fetched from memory.

3- Solution for Load-Store: This type of hazard occurs when instructions loading to register have not completed and next instructions are trying to use that register. This hazard can be solved successfully by stalling the next instruction until the previous instruction is complete.

4- Solution for Branch: This type of hazard occurs when a branch instruction has to make a branch decision but it is not made by time. It can be solved by stalling the next 3 instructions.

D) **Equations**

-Data forwarding

If((rsE !=0) AND (rsE==WriteRegM) AND RegWriteM)

then ForwardAE = 10

else if((rsE !=0) AND (rsE==WriteRegW) AND RegWriteW)

then ForwardAE = 01

else

ForwardAE = 00

If((rtE !=0) AND (rtE==WriteRegM) AND RegWriteM)

then ForwardBE = 10

else if((rtE !=0) AND (rtE==WriteRegW) AND RegWriteW)

then ForwardBE = 01

else

ForwardBE = 00

-Stalling/Flushing

lwstall = ((rsD==rtE) OR (rtD==rtE)) AND MemtoRegE

StallF = StallD = FlushE = lwstall

-Branch

ForwardAD = (rsD!=0) AND (rsD==WriteRegM) AND RegWriteM

ForwardBD = (rtD!=0) AND (rtD==WriteRegM) AND RegWriteM

branchstall = BranchD AND RegWriteE AND (WriteRegE==rsD OR WriteRegE==rtD)

OR BranchD AND MemtoRegM AND (WriteRegM==rsD OR WriteRegM==rtD)

StallF = StallD= FlushE=lwstall OR branchstall

E) sraac may cause Compute-use hazards which can be solved by Forwarding.

Compute-use hazard test code

addi $s0,$s0, -10

add $s2, $s0, $zero

subi $s1, $s2, 4

Load-use hazard test code

addi $t0, $zero, 7

addi $t1, $zero, 4

addi $t2, $zero, 100

lw $s0, 0($s2)

add $s2,$s0,$s2

Load-store hazard test code

```
addi $s0, $zero, 5
addi $t0, $zero, 2
sw $s0, 0($t0)
lw $t0, 0($t0)
```

Branch hazard test code

```
beq $t0, $t0, done
add $t0, $t0, $zero
add $t0, $t0, $zero
add $t0, $t0, $zero
done:
```

No hazard:

```
add $s0, $s1, $zero
add $s2, $s3, $zero
and $s4, $s5, $zero
add $t0, $t1, $zero
sub $t2, $t3, $zero
```