



# BBM431 ADVANCED COMPUTER ARCHITECTURE

Yunus Emre Akgün

21726875

09/12/2020

# 1 Aim

In this project, you will write a part of a compiler for 5-stage pipelined MIPS processor's instruction set. Specifically, the program you will write prevent some type of data hazards for MIPS processors with and without forwarding logic by inserting NOPs between dependent instructions.

## 2 Parts

---

### 2.1 Part 1

#### 2.1.1 Problem Definition

In a typical 5-stage MIPS processor, if there is no forwarding, we need at least two instructions between two dependent instructions to prevent data hazards

#### 2.1.2 Problem Solution

To find all data hazard i create two class and i put all information of instructions into IType class and RType class in order to solve this problem and i divide them part by part such as type,rs,rt,rd and imm

#### 2.1.3 Problem Definition

we have to place two NOPs between any dependet instruction to remove data hazards as shown on the output instruction sequence.

#### 2.1.4 Problem Solution

To solve this problem i was create a function that takes arguments as cnt and instrucitons queue and first it looks consecutively two instruction then if data hazard finds program will put two NOP's when the function called second time it will look two instructions 2 units away fromeach other then if hazard detected put required NOP's

The screenshot shows an IDE with a project named 'pythonProject'. The editor displays assembly code in 'instructions' mode. The code is as follows:

```
1 lw $r1, 0($t5)
2 lw $r2, 0($t5)
3 add $r3, $r1, $r2
4 addi $r4, $r1, 8
5 sw $r3, 0($t5)
6 lw $r1, 0($t5)
7 lw $r2, 0($t5)
8 sub $r3, $r1, $r2
9 sw $r3, 0($t5)
11 or $r4, $r3, $r8
10 end
```

The 'Run' window shows the execution results for 'main'. It indicates 'No Forwarding' and lists all dependencies and instructions:

No Forwarding  
All dependency :  
2 - 3 on \$r2  
7 - 8 on \$r2  
8 - 9 on \$r3  
3 - 5 on \$r3

All instructions  
1 --> lw \$r1, 0(\$t5)  
2 --> lw \$r2, 0(\$t5)  
3 --> Nop  
4 --> Nop  
5 --> add \$r3, \$r2, \$r1  
6 --> Nop  
7 --> addi \$r4, \$r1, 8  
8 --> sw \$r3, 0(\$t5)  
9 --> lw \$r1, 0(\$t5)  
10 --> lw \$r2, 0(\$t5)  
11 --> Nop  
12 --> Nop  
13 --> sub \$r3, \$r2, \$r1  
14 --> Nop  
15 --> Nop  
16 --> sw \$r3, 0(\$t5)  
17 --> or \$r4, \$r8, \$r3  
18 --> end

Figure 1: Without Forwarding

The screenshot shows an IDE window titled 'pythonProject' with a file named 'instructions'. The code in the editor is as follows:

```
1 lw $s0, 0($t5)
2 add $s1, $s0, $s2
3 addi $t0, $t1, 4
4 or $t3, $t4, $t5
5 sw $t0, 0($t6)
6 end
```

The right-hand pane displays the execution results for the 'main' program. It shows the following output:

```
No Forwarding
All dependency :
1 - 2 on $s0
3 - 5 on $t0
All instructions
1 --> lw $s0, 0($t5)
2 --> Nop
3 --> Nop
4 --> add $s1, $s2, $s0
5 --> addi $t0, $t1, 4
6 --> Nop
7 --> or $t3, $t5, $t4
8 --> sw $t0, 0($t6)
9 --> end
```

The IDE interface includes a sidebar on the left with 'Project', 'main.py', and 'instructions' tabs. The bottom status bar shows '6:6 LF UTF-8 4 spaces Python 3.9 (pythonProject)'.

Figure 2: With Forwarding

## 2.2 Part 2

### 2.2.1 Problem Definition

In a typical 5-stage MIPS processor, if there is forwarding, we need only one NOP after lw instruction if the following instruction is dependent to it

### 2.2.2 Problem Solution

To find all data hazard i create two class and i put all information of instructions into classes and lw instruction is a IType instruction and has a attribute immediate rs,and rt every time program will check if lw.RT is one of the source of other instruction then it will put one NOP's

The screenshot shows an IDE with a project named 'pythonProject'. The editor displays a file named 'instructions' with the following MIPS assembly code:

```
1 lw $r1, 0($t5)
2 lw $r2, 0($t5)
3 add $r3, $r1, $r2
4 addi $r4, $r1, 8
5 sw $r3, 0($t5)
6 lw $r1, 0($t5)
7 lw $r2, 0($t5)
8 sub $r3, $r1, $r2
9 sw $r3, 0($t5)
10 or $r4, $r3, $r8
11 end
```

The 'Run' output window shows the following execution details:

With Forwarding  
All dependency :  
2 - 3 on \$r2  
7 - 8 on \$r2  
All instructions  
1 --> lw \$r1, 0(\$t5)  
2 --> lw \$r2, 0(\$t5)  
3 --> Nop  
4 --> add \$r3, \$r2, \$r1  
5 --> addi \$r4, \$r1, 8  
6 --> sw \$r3, 0(\$t5)  
7 --> lw \$r1, 0(\$t5)  
8 --> lw \$r2, 0(\$t5)  
9 --> Nop  
10 --> sub \$r3, \$r2, \$r1  
11 --> sw \$r3, 0(\$t5)  
12 --> or \$r4, \$r8, \$r3  
13 --> end

Figure 3: Without Forwarding

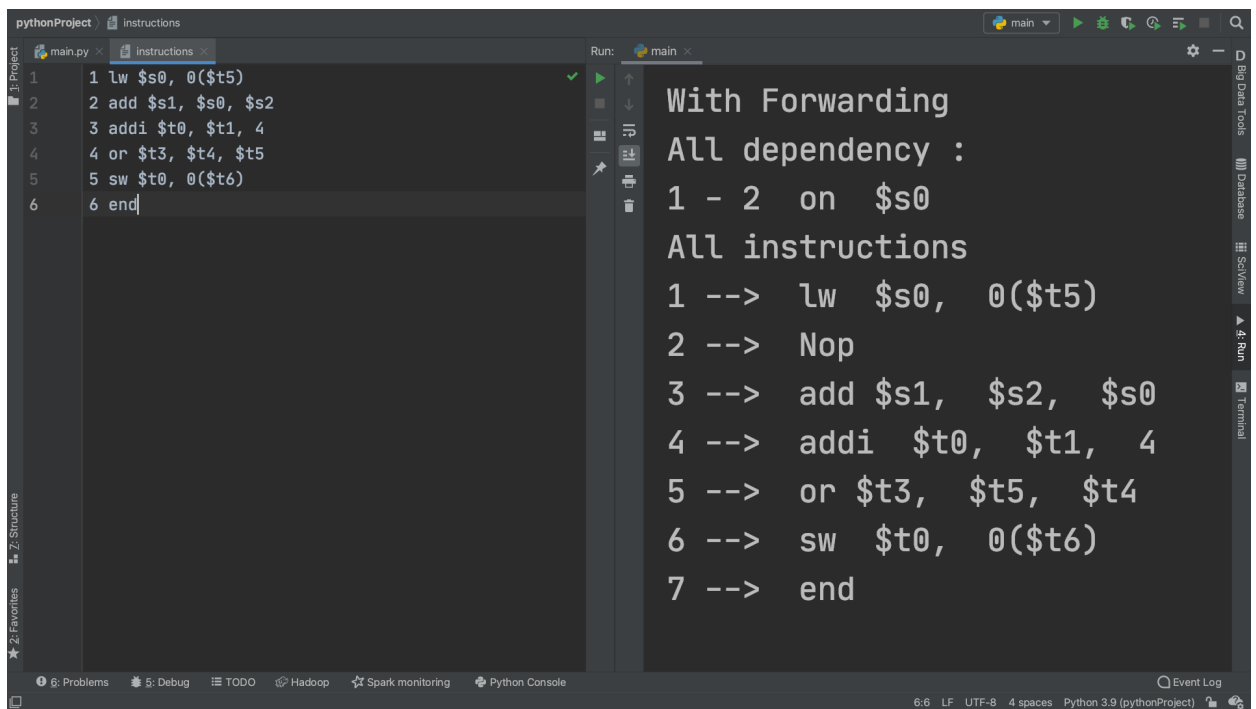


Figure 4: With Forwarding