

**JOMO KENYATTA UNIVERSITY OF AGRICULTURE AND TECHNOLOGY**

**SCHOOL OF COMPUTING**

**DEPARTMENT:** COLLEGE OF PURE AND APPLIED SCIENCE

**UNIT NAME:** COMPUTER ARCHITECTURE

**UNIT CODE:** BCT 2408

**NAME:** WARWINU PHILLIP KAMAU

**REG NO:** SCT212-0048/2021

**LECTURER NAME:** DR. MULANG ONANDO

**TASK:** LAB3

**Consider the following MIPS code fragments, each containing two instructions. For each code fragment identify the type of hazard that exists between the two instructions and the registers involved.**

**a.**

**LD R1, 0(R2)**

**DADD R3, R1, R2**

**b.**

**MULT R1, R2, R3**

**DADD R1, R2, R3**

**c.**

**MULT R1, R2, R3**

**MULT R4, R5, R6**

**d.**

**DADD R1, R2, R3**

**SD 2000(R0), R1**

**e.**

**DADD R1, R2, R3**

**SD 2000(R1), R4**

**Explain the behavior of a 2-bit saturating counter branch predictor.**

**Show the state of the predictor and the transition for each outcome of the**

**Branch.**

a. Read After Write (RAW): The DADD instruction needs the value stored in R1 by the earlier LD instruction.

b. Write After Write (WAW): DADD writes to R1 after MULT reads it.

c. Structural Hazard: There's a potential hardware conflict because the multiplier unit might already be in use.

d. RAW Dependency: The SD instruction, during its memory access (MEM) stage, depends on the value of R1 calculated by the DADD instruction.

e**.** RAW Dependency: The SD instruction also relies on R1 in its ALU (execute) stage, which is produced by DADD.

**b. Consider the following code:**

**for (i=0; i<N; i++)**

**if (x[i] == 0)**

**y[i] = 0.0;**

**else**

**y[i] = y[i]/x[i];**

**Assume that the assembly code generated is then:**

**loop: L.D**  **F1,**  **0(R2)**

**L.D**  **F2,**  **0(R3)**

**BNEZ**  **F1,**  **else**

**ADD.D**  **F2,**  **F0,**  **F0**

**BEZ**  **R0,**  **fall**

**else: DIV.D**  **F2,**  **F2,**  **F1**

**fall: DADDI**  **R2,**  **R2,**  **8**

**DADDI**  **R3,**  **R3,**  **8**

**DSUBI**  **R1,**  **R1,**  **1**

**S.D**  **-8(R3), F2**

**BNEZ**  **R1,**  **loop**

**where:**

**• the value of N is already stored in R1**

**• the base addresses for x and y are stored in R2 and R3, respectively**

**• register F0 contains the value 0**

**• register R0 (always) contains the value 0**

**Assuming that every other element of x has the value 0, starting with the**

**first one, show the outcomes of predictions when a 2-bit saturating counter**

**is used to predict the inner branch BNEZ F1, else. Assume that the initial**

**value of the counter is 00.**

1. **Branch predictor using a 2-bit saturating counter**

|  |  |  |  |
| --- | --- | --- | --- |
| **Current counter value** | **prediction** | **Actual outcome** | **New counter value** |
| 00 | NT | NT | 00 |
| 00 | NT | T | 01 |
| 01 | NT | NT | 00 |
| 01 | NT | T | 10 |
| 10 | T | NT | 01 |
| 10 | T | T | 11 |
| 11 | T | NT | 10 |
| 11 | T | T | 11 |

**b. 2-bit counter prediction rate**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Iteration** | **Current counter value** | **Prediction** | **Actual outcome** | **New counter value** | **Result** |
| 1 | 00 | NT | NT | 00 | Hit |
| 2 | 00 | NT | T | 01 | Miss |
| 3 | 01 | NT | NT | 00 | Hit |
| 4 | 00 | NT | NT | 00 | Hit |
| 5 | 00 | NT | T | 01 | Miss |
| 6 | 01 | NT | NT | 00 | Hit |
| 7 | 00 | NT | NT | 00 | Hit |
| 8 | 00 | NT | T | 01 | Miss |