Susmitha Shailesh

I pledge my honor that I have abided by the Stevens Honor System.

**4.3:**

**4.3.1**

35% - LDUR and STUR use data memory.

**4.3.2**

100% - All instructions use instruction memory.

**4.3.3**

76% - LDUR, STUR, I-type, CBZ, and B use sign-extend.

**4.3.4**

Sign extend is only used 76% of the time. For the other 24% of cycles, sign extend is computed, but not used; it is neglected at these times.

**4.5:**

Convert from hexadecimal to binary: 1111 1000 000 0001 0100 0000 0110 0010.

**4.5.1**

Sign-extend output: 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0001 0100

“Shift left 2” output: 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0101 0000

**4.5.2**

The value of the ALU control unit’s inputs for this instruction is 0010.

**4.5.3**

****

After following this path, we add 4 to the old PC address to get the new PC address PC + 4.

**4.8 Assume for this problem an RType/IType takes 725ps, LDUR takes 925ps, STUR takes**

**880ps, CBZ takes 735ps, and B takes 525ps.**

New: (725)(.52) + (925)(.25) + (880)(.10) + (735)(.11) + (525)(.02) = 787.6

Old : 250 + 150 + 25 + 200 + 150 + 5 + 30 + 20 + 50 + 50 = 930

930/787.6 = 1.18

18% speedup

**4.9: For this problem, assume the clock cycle is 925ps.**

**4.9.1**

With: 925 + 300 = 1225ps

Without: 925ps

**4.9.2**

This is actually a slowdown, not a speedup. The clock cycle time is actually increased after adding in the multiplier.

**4.9.3**

The new ALU unit cannot be any slower and still improve performance.

**4.16:**

**4.16.1**

Pipelined processor: Determined by slowest stage Instruction Decode = 350 ps

Non-Pipelined: Sum of all stages = 1250 ps

**4.16.2**

Pipelined processor takes 5 cycles at 350ps per cycle

Pipelined processor: 5 ∗ 350 = 1750 ps

Non-pipelined processor: 1250 ps

**4.16.3**

The best choice for which stage to split in order to reduce cycle time is the longest stage. The longest stage in this case is ID, which is 350ps. The longest stage after splitting up ID is MEM at 300 ps.

**4.16.4**

LDUR and STUR use data memory. 35% of instructions use data memory.

**4.18**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ADDI X1, X2, #5 | IF | ID | EX | ME | WB |  |  |  |  |  |
| ADD X3, X1, X2 |  | IF | S | S | S | ID | EX | ME | WB |  |
| ADDI X4, X1, #15 |  |  |  |  |  | IF | ID | EX | ME | WB |

X3 = 33, X4 = 26

**4.20**

ADDI X1, X2, #5

NOP

NOP

ADD X3, X1, X2

ADDI X4, X1, #15

NOP

ADD X5, X3, X2

**4.22:**

**4.22.1**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| CC | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 |
| STUR X16, [X6, #12] | IF | ID | EX | MEM | WB |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| LDUR X16, [X6, #8] |  | IF | NOP | NOP | NOP | ID | EX | MEM | WB |  |  |  |  |  |  |  |  |  |  |  |  |
| SUB X7, X5, X4 |  |  | IF | NOP | NOP | NOP | NOP | ID | EX | MEM | DB |  |  |  |  |  |  |  |  |  |  |
| CBZ X7, Label |  |  |  | IF | NOP | NOP | NOP | NOP | NOP | NOP | NOP | ID | EX | MEM | WB |  |  |  |  |  |  |
| ADD X5, X1, X4 |  |  |  |  | IF | NOP | NOP | NOP | NOP | NOP | NOP | NOP | NOP | ID | EX | MEM | WB |  |  |  |  |
| SUB X5, X15, X4 |  |  |  |  |  | IF | NOP | NOP | NOP | NOP | NOP | NOP | NOP | NOP | NOP | NOP | NOP | ID | EX | MEM | WB |

**4.22.2**

No, is it not possible to reduce the number of stalls/NOPs resulting from this structural hazard by reordering code.

**4.25:**

**4.25.1**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| CC | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| LDUR | IF | ID | EX | ME | WB |  |  |  |  |  |  |  |  |  |  |  |
| LDUR |  | IF | ID | EX | ME | WB |  |  |  |  |  |  |  |  |  |  |
| ADD |  |  | IF | S | ID | EX | ME | WB |  |  |  |  |  |  |  |  |
| SUBI |  |  |  |  | IF | ID | EX | ME | WB |  |  |  |  |  |  |  |
| CBNZ |  |  |  |  |  | IF | ID | EX | ME | WB |  |  |  |  |  |  |
| LDUR |  |  |  |  |  |  | IF | ID | EX | ME | WB |  |  |  |  |  |
| LDUR |  |  |  |  |  |  |  | IF | ID | EX | ME | WB |  |  |  |  |
| ADD |  |  |  |  |  |  |  |  | IF | S | ID | EX | ME | WB |  |  |
| SUBI |  |  |  |  |  |  |  |  |  |  | IF | ID | EX | ME | WB |  |
| CBNZ |  |  |  |  |  |  |  |  |  |  |  | IF | ID | EX | ME | WB |

S = Stall

**4.27:**

**4.27.1**

ADD X5, X2, X1

NOP

NOP

LDUR X3, [X5, #4]

LDUR X2, [X2, #0]

NOP

ORR X3, X5, X3

NOP

NOP

STUR X3, [X5, #0]

**4.27.2**

There is no way to reduce the number of NOPS by rearranging the code. This is because the instructions rely on instructions that come before it, so rearranging the code is not practical.

**4.27.3**

You need the hazard system to detect errors whether you have forwarding or not. Without the hazard system, the code will not execute properly, and the last two instructions will return false values.

**4.29:**

**4.29.1**

Always taken: The predictors are T, T, T, T, T. This is 60% accuracy.

Never taken: The predictors are NT, NT, NT, NT, NT. This is 40% accuracy.

**4.29.2**The first branch will result in a T. Move from strong predict not taken to weak predict not taken. Predicted outcome is not taken. The predictor value is 0.The second branch will result in a NT. Move from the weak predict not taken back to the strong predict not taken. Predicted outcome is not taken. The predictor value is 1.

The third branch will result in a T. Move from strong predict not taken to weak predict not taken. The predicted outcome is not taken. The predictor value is 0.

The fourth branch will result in a T. Move from weak predict not taken to weak predict taken. The predicted outcome is not taken because it started out in the weak predict not taken. The predictor value is 0.

Only the second branch is right, so the accuracy is 25%.