EE2003: Computer Organization
Indian Institute of Technology, Madras (Jul - Nov '23)

Assignment 4: RISCV Simulator

Arnav Mahajan [EP21B004]

Contents

1 Part A: Hazards

1.1 Problem Statement

(P12) Sum of Integer Array: Find the sum of the integer array of length 10. The integers in the array can range from -8 to 7. Store them wisely using the least number of registers or memory locations possible and find their sum.

1.2 RISC-V Assembly Code

```
addi t0, x0, 11 # Loop Limit
addi t1, x0, 1 # Loop Counter

addi t2, x0, 0 # Sum Register

add s0, gp, x0 # Memory Address Register

# Store Integers into System Data Memory: 1024 -> 1060
addi a0, x0, 2
sw a0, 1024(x0) # 1024 -> 2
addi a0, x0, 7
sw a0, 1028(x0) # 1028 -> 7
addi a0, x0, -5
sw a0, 1032(x0) # 1032 -> -5
addi a0, x0, 7
sw a0, 1036(x0) # 1036 -> 7
addi a0, x0, -8
```

```
sw a0, 1040(x0) # 1040 -> -8
addi a0, x0, 0
sw a0, 1044(x0) # 1044 -> 0
addi a0, x0, 4
sw a0, 1048(x0) # 1048 -> 4
addi a0, x0, -2
sw a0, 1052(x0) # 1052 -> -2
addi a0, x0, 5
sw a0, 1056(x0) # 1056 -> 5
addi a0, x0, 2
sw a0, 1060(x0) # 1060 -> 2

ArrayAdder:
beq t0, t1, Terminate # Branch to Terminate Loop Limit = Loop Counter
lw a1, 0(s0) # Load Word from Memory
addi s0, s0, 4 # Increment Memory Address
addi t1, t1, 1 # Increment Loop Counter
add t2, t2, a1 # Add Word to Sum
j ArrayAdder # Iterate

Terminate:
```

1.3 Data and Control Hazards

1. **Data Hazards:** The Data Hazards present (or possibility after reordering) in the program are:

   - The use of the following code for ArrayAdder raises a RAW Hazard:

     ```
     ArrayAdder:
     beq t0, t1, Terminate # Branch to Terminate Loop Limit =
                             Loop Counter
     lw a1, 0(s0) # Load Word from Memory
     add t2, t2, a1 # Add Word to Sum
     addi s0, s0, 4 # Increment Memory Address
     addi t1, t1, 1 # Increment Loop Counter
     j ArrayAdder # Iterate
     ```

     The difference is the reordering of the two increment instructions. Here, a1 is immediately required to be read after being loaded from memory. Thus the pipeline is stalled for 2 cycles. The original case runs the increment instructions until the loading instruction is completed so that stalling is prevented.

lw a1, 0(s0)			F	D	X	M	W			
add t2, t2, a1				F	-	-	D	X	M	W
addi s0, s0, 4							F	D	X	M

   - Another case of RAW Hazard is seen when words are stored in the memory. Since the register-store of the word is not completed until the memory-store is called, the pipeline is again stalled for 2 cycles.

addi a0, x0, 2					F	D	X	M	W			
sw a0, 1024(x0)						F	-	-	D	X	M	W
addi a0, x0, 7									F	D	X	M

2. **Control Hazards:** Control Hazard will occur for always-flush after the last iteration of the loop, where the branch is taken to Terminate. It will also occur when using Branch Prediction, as it will predict to be taken.

1.4 Throughput for various Configurations

Total number of instruction executions = **85**
Throughput for:

1. **w/o Forwarding - Flush Instruction:** $0.714f$
2. **w Forwarding - Flush Instruction:** $0.859f$
3. **w/o Forwarding - Branch Delay Slot:** $0.708f$
4. **w Forwarding - Branch Delay Slot:** $0.850f$

(f - Clock Frequency)

1.5 Comparison of Results

Activating the forwarding makes use of the feed-forward path. Hence the RAW Hazard during storing words in memory can be resolved, therefore saving 2 cycles per store. This accounts for the higher throughput.

The decrease in throughput using Delay Slot is due to the processor executing the instruction

```
lw a1, 0(s0)
```

completely, which is not the case in Flush, where it is flushed immediately after branch instruction is decoded.

Note: Pipeline is stalled even with forwarding, in case of load instruction as the load is carried out in the 'M' stage of pipeline and there is no feed-forward from 'X' stage, unlike an ALU operation.

2 Part B: Branch Prediction

2.1 Problem Statement

(P6) Assigned Program: test_branch.riscv

2.2 Execution Report

Parameter	AT	NT	BTFNT	BPB
Number of Instructions	752	740	752	752
Number of Cycles	1064	1037	1051	1064
Avg CPI	1.4149	1.4014	1.3976	1.4149
Branch Prediction Accuracy(%)	63.89	35.48	75.00	63.89
Number of Control Hazards	81	89	77	81
Number of Data Hazards	497	494	498	497
Number of Memory Hazards	70	70	70	70

Acryonyms:
AT: Always Taken
NT: Always Not Taken
BTFNT: Back Taken Forwarded Not Taken
BPB: Branch Prediction Buffer
CPI: Cycles per Instruction