CS211 Computer Architecture II

Pipeline and CPU Cache

Assigned 22/09/2022 Homework #1 *Due 23:59:59 08/10/2022*

https://toast-lab.sist.shanghaitech.edu.cn/courses/CS211@ShanghaiTech/Fall-2022/

The homework is intended to help you learn the material, and we encourage you to collaborate with other students and to ask questions in discussion sections and office hours to understand the problems. However, each student must turn in their own solution to the problems.

The homework also provides essential background material for the mid-term and final exams. It will be graded primarily on an effort basis, but if you do not work through it, you are unlikely to succeed on the mid-term and final exams! We will distribute solutions to homework assignment. Note that each homework assignment is due at its respective due time, and all assignments are to be submitted **in English**. However, late submissions will **NOT** be accepted, except for extreme circumstances and/or with prior arrangement.

Name: 俞政宏 Zhenghong Yu

ID: 2020533156

Problem 1: Pipeline and Hazards

In this problem, your task is to go through following problems to understand pipeline and hazards.

Problem 1.1 Pipeline and Hazards

Let us begin by considering the following assembly code:

```
Loop: addi x12, x12, 4      ; x12=x12+4
      ld x11,0(x12)         ; load x11 from address 0+x12
      sub x14, x13, x12     ; x14=x13-x12
      addi x11, x11, 4      ; x11=x11+4
      bnez x14, Loop        ; branch to Loop if x14 != 0
      sd x11,0(x12)         ; store x11 at address 0+x12
```

X14 = X12

Assume that the initial value of `x13` is `x12+12`.

1.1.a. (0.5 point) Data hazards are caused by data dependencies in the code. Whether a dependency causes a hazard depends on the machine implementation (i.e., number of pipeline stages). List all of the data dependences in the code above. Record the register, source instruction, and destination instruction; for example, "there is a data dependency for register `x11` from the `ld` to the `addi`."

Solution: there is a data dependency:

for register X12 from the ld to the [addi X12, X12, 4]

for register X12 from the sub to the [addi X12, X12, 4]

for register X11 from the [addi X11, X11, 4] to the ld

for register X14 from the bnez to the sub

for register X12 from the sd to the [addi X12, X12, 4]

for register X11 from the sd to the [addi X11, X11, 4]

for register X11 from the sd to the sub

1.1.b. (1 point) Show the timing of this instruction sequence for the 5-stage RISC pipeline without any forwarding or bypassing hardware but assuming that a register read and write in the same clock cycle "forwards" through the register file. Fill the following pipeline chart (F: instruction fetch; D: instruction decode; X: execution; M: memory access; W: writeback) and add more rows and/or columns if necessary. Assume that the branch is handled by *flushing the pipeline*. If all memory references take 1 cycle, how many cycles does this loop take to execute? You can draw a table by yourself if needed.

`addi x12,x12,4`	F	D	X	M	W							
`ld x11,0(x12)`		F										

The loop will excute 3 times. 39 cycles

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
addi x12, x12, 4	F	D	X	M	W																																		
ld x11, 0(x12)		S	S	F	D	X	M	W																															
sub x14, x13, x12					F	D	X	M	W																														
addi x11, x11, 4						S	F	D	X	M	W																												
bnez x14, Loop								F	D	X	M	W																											
sd, x11, 0(x12) (FLASHED)									F	D	X	M	W																										
addi x12, x12, 4										S	S	S	F	D	X	M	W																						
ld x11, 0(x12)														S	S	F	D	X	M	W																		
sub x14, x13, x12																	F	D	X	M	W																	
addi x11, x11, 4																		S	F	D	X	M	W															
bnez x14, Loop																				F	D	X	M	W														
sd, x11, 0(x12) (FLASHED)																					F	D	X	M	W													
addi x12, x12, 4																						S	S	S	F	D	X	M	W									
ld x11, 0(x12)																										S	S	F	D	X	M	W						
sub x14, x13, x12																													F	D	X	M	W					
addi x11, x11, 4																														S	F	D	X	M	W			
bnez x14, Loop																																F	D	X	M	W		
sd x11, 0(x12) (FLASHED)																																	F	D	X	M	W	
sd x11, 0(x12)																																		S	F	D	X	M	W

1.1.c. (1 point) Show the timing of this instruction sequence for the 5-stage RISC pipeline with full forwarding and bypassing hardware. Fill the pipeline timing chart below. Assume that the branch is handled by predicting it *as taken* and is determined after decode. If all memory references take 1 cycle, how many cycles does this loop take to execute?

`addi x12,x12,4`	F	D	X	M	W							
`ld x11,0(x12)`		F										

The loop will excute 3 times, 24 cycles

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
addi x12, x12, 4	F	D	X	M	W																			
ld x11, 0(x12)		F	D	X	M	W																		
sub x14, x13, x12			F	D	X	M	W																	
addi x11, x11, 4				F	D	X	M	W																
bnez x14, Loop					F	D	X	M	W															
addi x12, x12, 4						S	F	D	X	M	W													
ld x11, 0(x12)								F	D	X	M	W												
sub x14, x13, x12									F	D	X	M	W											
addi x11, x11, 4										F	D	X	M	W										
bnez x14, Loop											F	D	X	M	W									
addi x12, x12, 4												S	F	D	X	M	W							
ld x11, 0(x12)														F	D	X	M	W						
sub x14, x13, x12															F	D	X	M	W					
addi x11, x11, 4																F	D	X	M	W				
bnez x14, Loop																	F	D	X	M	W			
addi x12, x12, 4 (flashed)																		S	F	D	X	M	W	
sd x11, 0(x12)																				F	D	X	M	W

Problem 2: Cache (2.5 points)

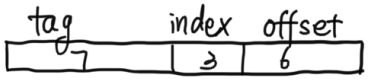
Suppose that your friend Li Hua has got a new processor with an 8-line data cache as shown below. Each cache line has 64 bytes and memory addresses are 16 bits with byte-addressable memory.

Tags	Data

Problem 2.1 (1.25 Points)

If the cache is a direct-mapped cache, divide 16 bits of a memory address for tag, index, and offset.

tag: 7 bits
index: 3 bits
offset: 6 bits



Suppose the processor accesses the following data addresses starting with an empty cache:
0x0F28, 0x0127, 0x0F28, 0x7A0D, 0x0F28, 0x1935, 0x2468, 0x3721, 0x9527, 0x0127
What would the cache tags look like after accessing this sequence of data? Do list all the changes.
How many hits and misses would be?

	1	2	3 Tags	4	5	6	7	8	9	10
0				0x3D	0x3D	0x3D	0x3D	0x3D	0x3D	0x3D
1							0x12	0x12	0x12	0x12
2										
3										
4	0x7	0x0	0x7	0x7	0x7	0xC	0xC	0x1B	0x4A	0x0
5										
6										
7										
	M	M	M	M	H	M	M	M	M	M

hits: 1

misses: 9

0x0F28	0x7	0x4
0x0127	0x0	0x4
0x0F28	0x7	0x4
0x7A0D	0x3D	0x0
0x0F28	0x7	0x4
0x1935	0xC	0x4
0x2468	0x12	0x1
0x3721	0x1B	0x4
0x9527	0x4A	0x4
0x0127	0x0	0x4

8	2	6

Problem 2.2 (1 Point)

@910

Suppose Li Hua reconfigures the data cache into a 2-way set-associative cache and employs the NMRU replacement policy. Given the same sequence of memory addresses in Problem 2.1, what would be the cache tags look like after accessing? Still assume the cache is initially empty and list all the changes. How many cache hits and misses now?

	1	2	3 Tags	4	5	6	7	8	9	10
0	0xF	0xF	0xF	0xF	0xF	0xF	0xF	0x37	0x37	0x1
1							0x24	0x24	0x24	0x24
2										
3										
0		0x1	0x1	0x7A	0x7A	0x19	0x19	0x19	0x95	0x95
1										
2										
3										
	M	M	H	M	H	M	M	M	M	M

hits: 2

misses: 8

0x0F28	0xF	0x0
0x0127	0x1	0x0
0x0F28	0xF	0x0
0x7A0D	0x7A	0x0
0x0F28	0xF	0x0
0x1935	0x19	0x0
0x2468	0x24	0x1
0x3721	0x37	0x0
0x9527	0x95	0x0
0x0127	0x1	0x0

Problem 2.3 (0.25 Points)

Suppose that the data cache has an access time of one CPU cycle, a hit rate of 60% on one of your programs, and a miss penalty to memory of forty cycles. What is the average memory access time?

AMAT = hit cycle + miss rate x miss penalty

= 1 cycle + 40% x 40 cycle

= 17 cycles