Solutions Exam E2018

Problem 1

Question 1.1

As the byte gets loaded into a 32(64) bit register it needs to be interpreted unsigned or signed to determine the upper bits (0 or sign extended). A store byte writes that byte into a single byte in memory and needs no upper bits handling.

Question 1.2

When a page is not in main memory and it has to be retrieved from the disk.

Question 1.3

A cache for fast translation from virtual to physical address.

Question 1.4

At AI=2.0, X2 is computation bound, while X4 is memory bound.

We can increase the memory bandwidth of X4.

Problem 2

Question 2.1

P1:

	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
loop: ld x31, 0(x20)	F	D	E	M	W																									
addi x20, x20, -8		F	D	E	M	W		[x20]=8																					
add x31, x31, x21			F	-	D	E	M	W																					
sd x31, 8(x20)					F	-	-	D	E	M	W																		
bne x20, x0, loop								F	D	E	M	W		*(branch taken)*															
sub x23, x23, x24									~~F~~ *flushed*																				
loop: ld x31, 0(x20)										F	D	E	M	W															
addi x20, x20, -8											F	D	E	M	W		[x20=0]												
add x31, x31, x21												F	-	D	E	M	W												
sd x31, 8(x20)														F	-	-	D	E	M	W									
bne x20, x0, loop																	F	D	E	M	W								
sub x23, x23, x24																		F	D	E	M	W							
xor																			**F**	D	E	M	W						

It takes 19 cycles to fetch instruction `xor ...`, corresponding to an execution time of 19 ns.

Question 2.2

One possible implementation is with a small cache (fully associative, in this example)

Tag **Actual PC**	Data **Target PC**	Branch **Taken**
...	...	...
...	...	...
0x01020	0x01010	1
...	...	...
...	...	...

When we fetch any instruction if we have a match (tag) we read the taken bit in the cache. If Taken=1 we load the "branch target" in the PC. If we have a miss, we continue.

If in the decode stage, we detect a branch, we store the "branch target" in the cache and update the taken bit.

 ... continued on the next page

Question 2.3

P2:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
loop: ld x31, 0(x20)	F	D	E	M	W																
addi x20, x20, -8		F	D	E	M	W															[x20]=8
add x31, x31, x21			F	D	E	M	W														*(forward [x31] M→E)*
sd x31, 8(x20)				F	D	E	M	W													*(forward [x31] E→E)*
bne x20, x0, loop					F	D	E	M	W												*(branch taken = prediction)*
loop: ld x31, 0(x20)						F	D	E	M	W											
addi x20, x20, -8							F	D	E	M	W										[x20=0]
add x31, x31, x21								F	D	E	M	W									
sd x31, 8(x20)									F	D	E	M	W								
bne x20, x0, loop										F	**D**	E	M	W							*(branch not taken → WRONG pred.)*
loop: ld x31, 0(x20)											~~F~~										*flushed*
sub x23, x23, x24													F	D	E	M	W				
xor													**F**	D	E	M	W				

At cycle 11 in stage D (**D**) decision on the branch is made: "branch not taken". Need to flush the pipeline.

It takes 13 cycles to fetch instruction `xor ...`, corresponding to an execution time of 13 ns.

Problem 3

Question 3.1

$t_{seek} = 11\ ms$

$t_{rot} = (1/2)/(7200/60)\ s = 4.17\ ms$

$t_{dtr1} = 1024 \times (1/34\ MB/s) = 0.03\ ms$

$t_{ctr1} = 1024 \times (1/(480/8)\ MB/s) = 0.017\ ms$

$t_1 = t_{seek} + t_{rot} + t_{dtr1} + t_{ctr1} = 15.22\ ms$

Question 3.2

$t_{dtr2} = t_{dtr1} \times 2 = 0.06\ ms$

$t_{ctr2} = t_{ctr2} \times 2 = 0.034\ ms$

$t_2 = t_{seek} + t_{rot} + t_{dtr2} + t_{ctr2} = 15.26\ ms$

Question 3.3

The dominant factor is the seek time and, to some extent, the wait till the sector is under the head (rotation time).

Improve seek time. Probably by smarter allocation of sectors by the operating system. Larger sectors can also help.

Problem 4

Question 4.1

C1:

Byte offset = 2 bits (4 bytes)

Block offset = 1 bits (2 words)

Block count = $2^{11}/2^2/2^1 = 2^8$ blocks

Index = 8 bits

Tag = $32 - 8 - 1 - 2 = 21$ bits

C2:

Byte offset = 2 bits (4 bytes)

Block offset = 2 bits (4 words)

Block count = $2^{11}/2^2/2^2 = 2^7$ blocks

Index = 7 bits

Tag = $32 - 7 - 2 - 2 = 21$ bits

C3:

Byte offset = 2 bits (4 bytes)

Block offset = 1 bits (2 words)

Blocks in set = 1 bits (2 blocks)

Set count = $2^{11}/2^1/2^1/2^2 = 2^7$ sets

Index = 7 bits

Tag = $32 - 7 - 1 - 2 = 22$ bits

Question 4.2

C1:

Overhead = 2^8 blocks $*$(21 tag +1 valid) = 5632 bits

Total = $2KB * 8$+Overhead= $2^{11} * 2^3 + 2^8 * 22 = 2^8 * (64 + 22) = 22016$ bits

C2:

Overhead = 2^7 blocks $*$(21 tag +1 valid) = 2816 bits

Total = $2KB * 8$+Overhead= $2^{11} * 2^3 + 2^7 * 22 = 2^7 * (128 + 22) = 19200$ bits

C3:

Overhead = 2^7 sets $*$ 1 LRU $+2^7$ sets $*2^1$ blocks $*$(22 tag +1 valid) = 6016 bits

Total = $2KB*8+$ Overhead = $2^{11}*2^3+2^7+2^7*2^1*23 = 2^7*(128+1+46) = 22400$ bits

Question 4.3

	C1				C2				C3			
Address	Tag	Idx	Off	H/M	Tag	Idx	Off	H/M	Tag	Idx	Off	H/M
000000000000	0	0	0	M	0	0	0	M	0	0	0	M
000000000100	0	0	4	H	0	0	4	H	0	0	4	H
000000001000	0	1	0	M	0	0	8	H	0	1	0	M
000000001100	0	1	4	H	0	0	C	H	0	1	4	H
100000000000	1	0	0	M	1	0	0	M	2	0	0	M
000000001000	0	1	0	H	0	0	8	M	0	1	0	H
000000000000	0	0	0	M	0	0	0	H	0	0	0	H
100000000000	1	0	0	M	1	0	0	M	2	0	0	H

Question 4.4

$R_{\mathbf{C1}} = 3/8$

$R_{\mathbf{C2}} = 4/8$

$R_{\mathbf{C3}} = 5/8$

Problem 5

Question 5.1

CPI $= 0.2 \times 12 + 0.3 \times 5 + 0.1 \times 3 + 0.4 \times 4 = 5.8$

Question 5.2

$t_{exe} = (\text{IC} \times \text{CPI})/f_{clock} = (5.8 \times 10^7)/(3 \times 10^9) = 5.8/3 \times 10^{-2} = 1.9\overline{3} \times 10^{-2} = 19.3\, ms$

Question 5.3

Miss CPI = (instructions percent) × (miss rate) × (miss penalty)

Instruction miss CPI $= 1.0 \times 0.05 \times 8 = 0.4$

lw, sw miss CPI $= 0.3 \times 0.12 \times 8 = 0.288 \simeq 0.3$

Excess CPI (due to misses) $= 0.4 + 0.3 = 0.7$

CPI (including cache misses) $= 5.8 + 0.7 = 6.5$

$t_{exe} = (6.5 \times 10^7)/(3 \times 10^9) = 2.1\overline{6} \times 10^{-2} = 21.7\ ms$

End of the Exam