ELE 475 | PS#1 | David Wentzlaff

Problem 1)

$$CPI = P_{ALU}\,CPI_{ALU} + P_{BR/JMP}\,CPI_{BR/JMP} + P_{LD/ST}\left(P_{HIT}\,CPI_{HIT} + P_{MISS}\,CPI_{MISS}\right)$$

$$= \underbrace{(1-0.22-0.12-0.2)(1.1)}_{ALU} + \underbrace{(0.2)(3.0)}_{BR/JMP} + \underbrace{(0.22+0.12)\left[\underbrace{(0.6)(1)}_{HIT} + \underbrace{(1-0.6)(120)}_{MISS}\right]}_{LD/ST}$$

$$= \boxed{17.63\ CPI}$$

ECE 475 | PS #1 | David Wentzlaff

Problem 2)

Case A) 1 GHz, CPI 1.2

$$\text{Performance} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Cycles}}{\text{Instruction}} \times \frac{\text{Time}}{\text{Cycle}}$$

$$\text{Performance}_A = y \times 1.2 \times 1\,ns$$

$$= 1.2\,y\,ns$$

Case B) 2 GHz, CPI 2

$$\text{Performance}_B = y \times 2 \times 0.5\,ns$$

$$= 1\,y\,ns$$

Assuming the same number of instructions in the program, y, we see that Case B has lower running time $1y\,ns < 1.2\,ns$, therefore, the 2 GHz, CPI 2 machine is better

ECE 475 | PS#1 | David Wentzloff

Problem 3, H&P5 C.1)

a)

Reg.	Source	Consumer
R1	LD	DADDI
R1	DADDI	SD
R2	DADDI(2)	DSUB
R4	DSUB	BNEZ
R2	DADDI(2)	LD
R2	DADDI(2)	SD

} Around loop

b)

```
                   1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18

LD   R1,0(R2)      F D X M W
DADDI R1,R1,#1       F D D D X M W
SD   0(R2),R1          F F F D D D X  M  W
DADDI R2,R2,#1                   F F  F  D  X  M  W
DSUB R4,R3,R2                             F  D  D  D  X  M  W
BNEZ R4,Loop                                 F  F  F  D  D  D  X  M  W

LD   R1,0(R2)                                                    F  D ...
```

c)

```
                   1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

LD                 F D X M W
DADDI                F D D X M W
SD                     F F D X M W
DADDI                      F D X M W
DSUB                         F D X M W
BNEZ                           F D X M W
(fallthrough)                    F - - - -
LD                                 F D  X  M  W
```

✱ Assumes branch resolved in Decode stage and no delay slots
- Resolving branch in decode requires zero detect after bypass.

C.1d)

```
LD        1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
DADDI
SD             SAME as C
DADDI
DSUB
BNEZ                FDXMW
LD                   FDXMW
```

~~A~~ Assumes branch resolved in Decode stage, no delay slots and early predecode to determine and fetch target of branch in Fetch stage

e)

```
        1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
LD      F1 F2 D1 D2 X1 X2 M1 M2 W1 W2
DADDI      F1 F2 D1 D2 D2 D2 D2 X1 X2 M1 M2 W1 W2
SD            F1 F2 D1 D1 D1 D1 D2 D2 X1 X2 M1 M2 W1 W2
DADDI            F1 F2 F2 F2 F2 D1 D1 D2 X1 X2 M1 M2 W1 W2
DSUB                F1 F1 F1 F1 F2 F2 D1 D2 D2 X1 X2 M1 M2 W1 W2
BNEZ                            F1 F1 F2 D1 D1 D2 D2 X1 X2 M1 M2 W1 W2
LD                                    F1 F2 F2 D1 D1 D2 X1 X2 M1 M2 W1 W2
```

f) 5-stage

$0.8ns + 0.1ns = 0.9ns$

10-stage

$0.4ns + 0.1ns = 0.5ns$

g) For d) $\frac{7 \text{ clocks}}{6 \text{ instructions}} = 1.1\overline{6}$ CPI

For e) $\frac{10 \text{ clocks}}{6 \text{ instructions}} = 1.\overline{6}$ CPI

C.1g (cont)

Average Instruction execution time 5-stage

$= 1.166666 \times 0.9\text{ns} = 1.05\text{ns}$

For 10-stage

$= 1.666666 \times 0.5\text{ns} = 0.83333\text{ns}$

10-stage Faster ✓

Problem 4 H&P5 B.2)

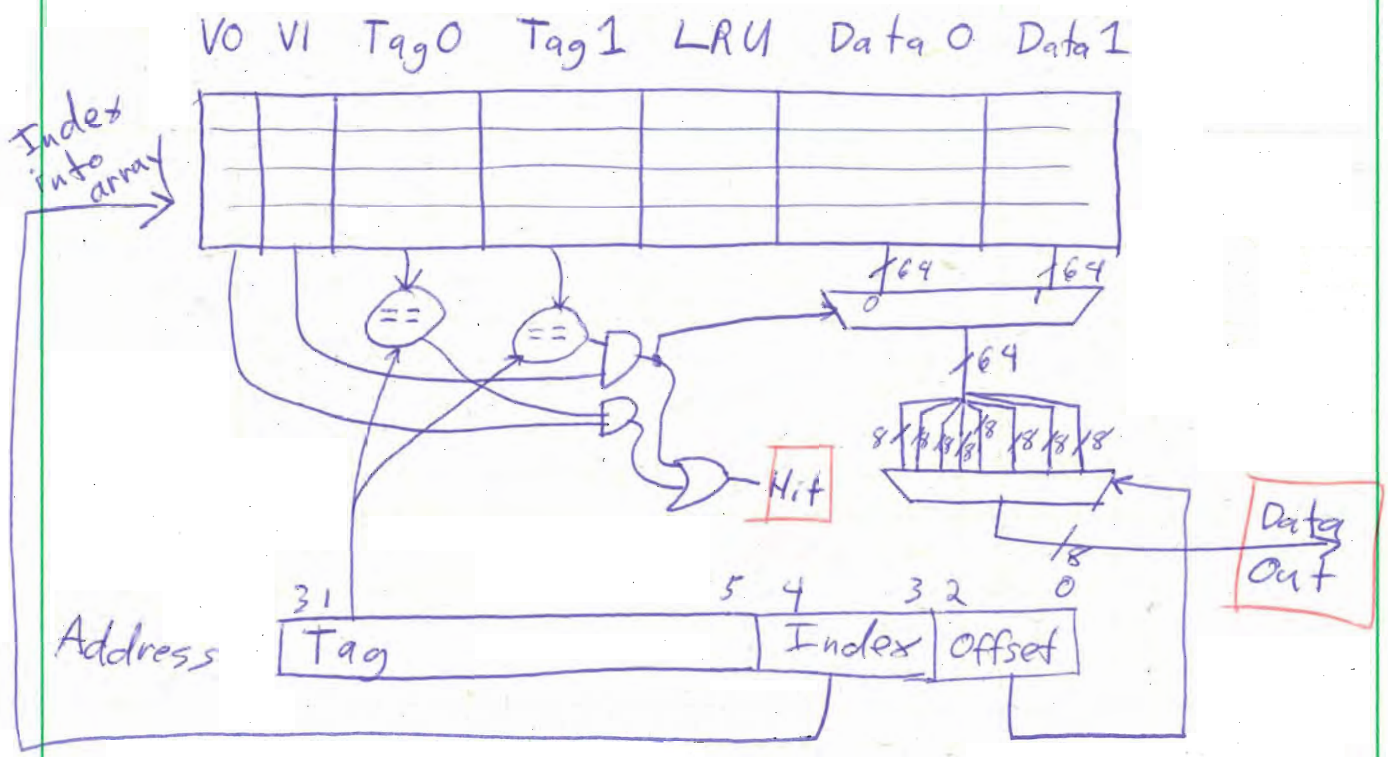
a) Direct-mapped cache

Cache Block	Set	Way	Possible Memory Blocks
0	0	0	M0, M8, M16, M24
1	1	0	M1, M9, M17, M25
2	2	0	M2, M10, M18, M26
3	3	0	M3, M11, M19, M27
4	4	0	M4, M12, M20, M28
5	5	0	M5, M13, M21, M29
6	6	0	M6, M14, M22, M30
7	7	0	M7, M15, M23, M31

b) Four-way set associative cache

Cache Block	Set	Way	Possible Memory Blocks
0	0	0	M0, M2, M4, M6, M8, M10, M12, M14, M16, M18, M20, M22, M24, M26, M28, M30
1	0	1	
2	0	2	
3	0	3	
4	1	0	M1, M3, M5, M7, M9, M11, M13, M15, M17, M19, M21, M23, M25, M27, M29, M31
5	1	1	
6	1	2	
7	1	3	

Problem 5)


V0
V1
Tag0
Tag1
LRU
Data 0
Data 1
Index into array
==
==
64
64
0
1
64
8
Hit
Data Out
31
5
4
3
2
0
Address
Tag
Index
Offset

Problem 6)

RAW

Reg.	Src.	Dest.
R1	ADD	ANDI
R3	SUB 1	SUB 2
R5	MUL	ADDIU

WAW

Reg	First Write	Second Write
R5	MUL	ADDIU

WAR

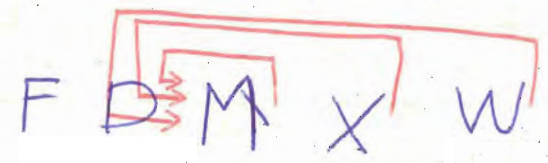
Reg	Read	Write
R3	ADD	SUB
R6	SUB1	SUB2
R2	ADD	ANDI

Problem 7 H&P5 C.6)

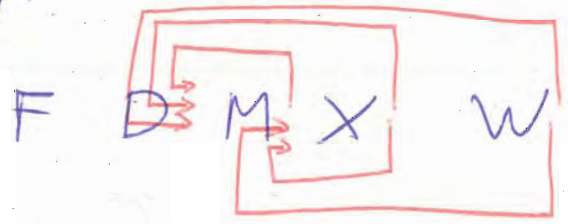
a) F D M X W

Assumes that only sources can use new addressing mode.

b) Case A: Forward everything back to Decode stage and stall on back-to-back ALU ops



Case B: For High performance, [illegible]
~~[illegible]~~ ~~[illegible]~~ ~~[illegible]~~
Bypass in two locations



c) Case A: ALU ops which feed dependant ALU ops will stall

ADD R6, R7, R8
SUB R9, R6, R12

ALU ops which feed LD/ST will stall

ADD R6, R7, R8
LD R9, R6

ALU ops which feed Memory-register ops will stall

ADD R6, R7, R8
SUB R9, (R6), R12

P7 cont)

Case B: ALU ops which feed LD/ST will stall

See above

ALU ops which feed Memory-register ops will stall

See above

d) LDs and STs that had register indirect addressing now require two instructions, an~~e~~ address computation and the LD/ST

LW R5, 10(R2) => ADDIU R5, R2, 10
LW R5, (R5)

Instruction sequences which had LDs followed by ALU operations can merge into ~~a~~ one instruction for simple operations

LW R5, 0(R2)
ADD R7, R5, R8 => ADD R7, (R2), R8

e) LDs followed by dependant instructions will not stall in new pipeline therefore lower CPI

Case A: All of the new stall reasons will introduce extra cycles therefore increasing CPI from 1 to 2 for those combinations.

ALU -> ALU

ALU -> LD/ST

ALU -> Mem-register

P2 cont)

Case B: All of the new stall reasons will introduce extra cycles therefore increasing CPI

ALU → LD/ST

ALU → Memory-register

Problem #8)

256 KB direct Mapped Miss Rate 0.013

64 KB ~~[illegible]~~ 8-way Miss Rate 0.029

256 KB direct - Mapped has lower miss rate, therefore is better

The bulk of the misses are caused by capacity which is why the direct mapped larger cache wins. If conflict dominated, larger cache could lose.