Gavin McRoy

Final Exam

1. 8 b instruction 4GHz 2 CPI. I need execution time

$8b \cdot CPI \cdot 4 \times 10^{-9}$

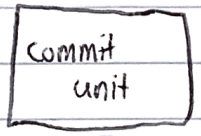
$8 \times 10^{9} \cdot 2 \cdot (4 \times 10^{-9}) = \boxed{64}$

2. 80% can be run in parallel. If 8 processors are working that is 30%

$\frac{100}{30} = 3.\bar{3}$ x speed up

3. I'm pretty sure this is when a large heat of run can be ~~ordered~~ by several CPU's
   - Example of such systems is super computers and warehouse scale computers

4. ↘ Reorder buffer (ROB)



5. False, programs require redesigning to utilize multiple CPU's

6. TU TTU TTU

7.

0 →	0
1 →	1
01 →	1
10 →	1
11 →	1
00 →	0

- I'm unsure how the table looks but if there is any input with a 1 all preceding input including the 1 will be output 1

8. the operation is not thread safe so either code piece may execute first

- Counts could be 1 if increments run first and the second thread interrupted or swapped
- Counts could be -1 if decrements run first and the first thread interrupted or swapped
- Counts could be 0 if both pieces of code run without interruption

9.

RegDest:	1
Mem Write	0
Branch	0
ALU SRC	0
Reg Write	1
Mem toReg	0
ALU Control	0010
Mem Read	0

10 Dependency if the instruction J has any of the conditions on instruction i

- instruction i produces result that may be used by J
- if J is dependent on k and k is dependent on i

Example

Dependency

```
add $S0, $T0, $T1
sub $T2, $S0, $T3
```

Independent if instruction has no result or value used by a previous instruction

imagine loading some value then adding another random value. The instructions are independent of each other

11.
lw R2 4(R1)
STALL → add R4 R2, R3
STALL → addi R5 R4 100
STALL → sw R5, 4(R1)

12. Caches store recently used information that can be accessed again in the future very quickly
of possibly useful information

Loop unrolling utilizes cache features and different optimization flags make better use of caching

13. SRAM
- Store each bit in a bi stable memory cell
- retains value indefinitely as long as kept powered

DRAM
- Stores each bit as a charge on a capacitor
- Very sensitive to disturbances

Caches are made of SRAM and main memory is made of DRAM

14. AMAT = hit time + miss rate x miss penalty

= 2 NS + (.82 x 100) = 22 NS

15 (.4 · 1) + (.2 · 2) + (.4 · 3) = 2
.4 + .4 + 1.2

→

16. $(.6 \cdot 1) + (.2 \cdot 2) + (.2 \cdot 3)$
$.6 + .4 + .6 = 1.6$

$\frac{2}{1.6} = 1.25 \times$ faster, so yes the optimizations are faster

17 Three reasons for cache miss

1 • Compulsory miss or item just wasn't in the cache
2 • Capacity miss or item was in cache but removed from space
3 • Conflict miss or was in cache but not associative enough so removed

Possible Solutions

1. Not exactly what you can do if it first occurs
2. ~~increase cache size or associativity~~ increase cache size
3. increase size or associativity

A fully associative cache probably has conflict misses and capacity misses

18.

A L1 8 bytes
L2 32 Bytes
Physical Memory 256 Bytes
Virtual Memory 256 Bytes (I thought virtual means unlimited)

Log2 (4) ← cache lines Log(Linsize) Remaining is Tag Bits

B. Index Bit = 2 offset = 1 Tag bits $8-(2+1) = 5$

L1 Index bits

[Tag | Index | offset

TAG | Index | offset
10101 | 01 | 1

L2 4 Line 2 way Associative

index $\text{Log}_2(4/2) = 1$
offset $\text{Log}_2(2) = 1$
Tag bits $= 8-2 = 6$

Tag | index | offset
101010 | 1 | 1

C. 10101010

16 Byte pages

8 bits virtual Address
4 pages of physical memory

$2^4 = 16$
4 Byte offset
4 page number

offset
1010

page number
1010

D.
1. bit
2. bit
3. bit
4. bit
5. bit
6. bit
7. bit
8. bit

E
1. Hit
2. Hit
3. Hit
4. Hit
5. Hit
6. Hit
7. Hit
8 Hit

EC. four classifications are
- Single instruction stream, single data stream
~~Single Instruction stream~~
- Single instruction stream, multiple data stream
- Multiple Instruction stream, single data stream
- Multiple instruction stream, multiple data stream