CS230 DLDCA End-Sem LITE, Tue 12 Nov 2024, 13.30-16.30, Max. Marks: 40

General instructions

- Write only in the space provided. Answer briefly but crisply (not lengthily or loosely).
- You are allowed to refer to your own hand-written notes only.
- Write neatly and clearly. Up to **+2 HP** for neat handwriting, neat/crisp answers.
- Answers generally have to be (briefly) explained. State any necessary assumptions.

[Q1] Short answer questions: [1x13+2=15 marks]

1. What real instruction(s) does the pseudo-instruction 'la' (load address) translate to?
   la load the address of an assembly variable onto a register
   It translates to:
   lui <reg>, <upper-16-bits> # not needed if constant value is <= 16 bits
   ori <same reg>, <same reg>, <lower-16-bits> # can also use addi here

2. MIPS has a convention as to which registers are caller saved versus which are callee saved. What is the need for such a convention?

   Such a convention is needed to link independently written/complied code, e.g. 3rd party library

3. Are RAW hazards in memory locations possible in the 5-stage MIPS pipeline? Explain.

   All memory operations in 4th stage (MEM), which always happens in order of instruction execution.
   So no RAW hazards in mem locations in 5-stage MIPS pipeline

4. Give an example of spatial locality in instruction memory access.

   Sequential execution, loops

5. When we say that a machine is a 32-bit architecture, what does this 32-bit refer to?

   The virtual address space of each process

6. State one use of virtual memory as a level of indirection.
   Any one of: illusion of memory being larger than physical memory, program relocation is easy without program rewrite, protection of one process's memory from another, controlled sharing of memory locations across processes, copy-on-write for shared dynamic linked library

7. What is the typical hit-time of TLB, in terms of number of cycles?

   1 cycle or less than 1 cycle (can be one pipeline stage, or part of a pipeline stage)

8. State one advantage of having a long pipeline, i.e. a large number of stages

   Pipeline speedup is proportional to pipeline length – more parallelism

9. State one disadvantage of having a long pipeline, i.e. a large number of stages

   Any one of: control is complex, exception handling is complex, data forwarding logic is complex, more chance of dependence across instructions in pipeline

10. What are the three kinds of bus lines?
    Control, address, data lines

11. In MIPS, **`not R1, R2`** is a pseudo-instruction where R1 gets assigned the value corresponding to the bit-wise toggling of R2. Implement this pseudo-instruction using real MIPS instruction(s).

    nor R1, R2, $0

12. In the instructions **`srl`** & **`sll`**, which of the registers **`{Rs, Rt, Rd}`**, if any, are ignored (i.e. neither read and used, nor written)?
    Rd is the destination, Rs is read and used, Rt is ignored

13. What does the instruction **`jalr`** do?
    jalr <reg> does $ra = PC+4 and PC = <value of given reg>

14. **[2 marks]** Complete the following MIPS assembly code such that when executing the instruction at L3, $s0 has the value of PC.

```
L1:   _ jal L2 _____________________________

L2:   _ addi $s0, $ra, 4 ___________________________

L3:   # here, $s0 must be PC (i.e address corresponding to L3)
```

[Q2] MIPS ISA [5 marks]

Consider the conditional move instruction MOVZ which copies a source register to a destination register only if a third register is zero. For example, the instruction

MOVZ $8, $11, $4

copies the contents of register 11 into register 8, only-if register 4 is zero (otherwise it does nothing).

1. **[1 mark]** Machine M1 is a regular MIPS machine whose assembler implements MOVZ as a pseudo-instruction. Indicate how MOVZ $a, $b, $c can be implemented using real MIPS instruction(s).

```
        bne $c, $0, SKIP
        add $a, $b, $0 # Could also be addi $a, $b, 0
SKIP:
```

2. **[1 mark]** Your answer above would have used a branch instruction. Give the 16-bit immediate value in the machine code of this branch instruction.

   16-bit immediate value = word offset from PC+4
   SKIP is at PC+8 with respect to the branch
   So answer = [(PC+8)-(PC+4)]/4 = 1

3. **[2 marks]** Machine M2 extends the MIPS ISA by implementing MOVZ as a real instruction. A program P while executing on M2, has 5% of executed instructions as MOVZ. For this program, M1 and M2 have the same clock cycle length and the same CPI. Which machine executes P faster and by what factor?

   Time/Program = Instructions/Program x Cycle/Instruction x Time/Cycle
   CPI and Time/Cycle are the same
   And M1 has 1.05 times more instructions
   So M2 is faster by a factor of 1.05

4. **[1 mark]** M2's clock speed is 2.1GHz. What is M1's clock speed?

   Clock speed = 1/clock_cycle_time
   Clock cycle length is same for M1 and M2
   So both have the same clock speed = 2.1 GHz

[Q3] Pipelining [8 marks]

Consider the same MOVZ instruction as above (except for the definition of MOVZ, this question is independent of the previous question). Machine M2 implements MOVZ by extending the original 5-stage MIPS pipeline implementation.

1. **[2 marks]** Suggest an appropriate instruction format for MOVZ $a, $b. $c. Provide a drawing with justification.

   3-regs, so R-format is suitable; $a should be Rd; $c is compared with 0, so it should be Rs for minimal changes to datapath (show the drawing)
   Giving $b as Rs and $c as Rt is also acceptable

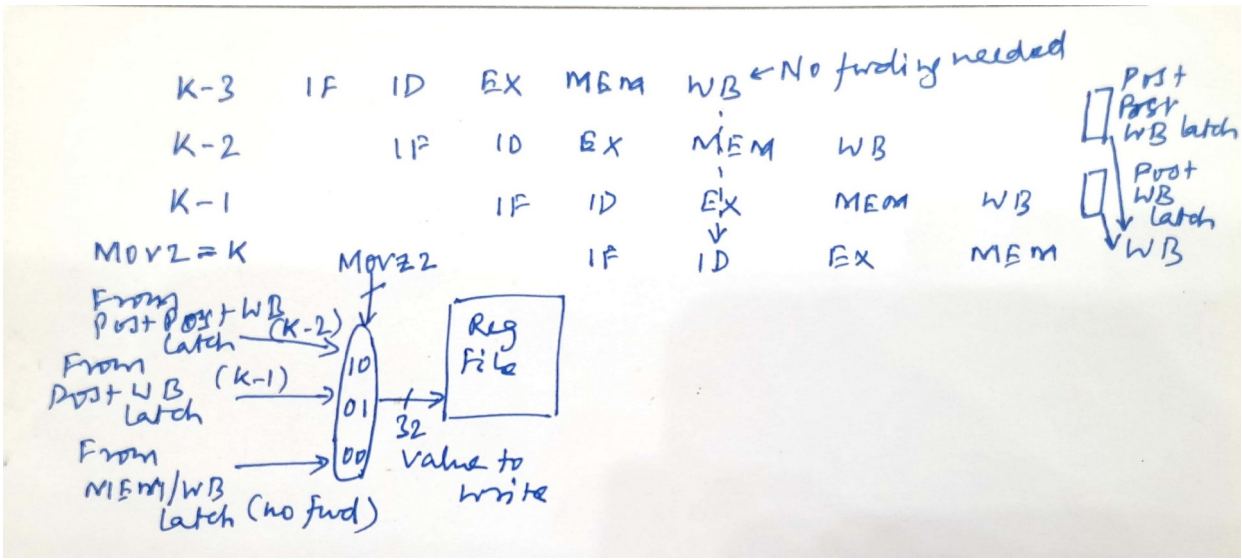
2. **[1 mark]** In which stage of execution does MOVZ need the RegWr (Register Write) control line? Explain briefly.

   It needs it in the WB stage, i.e. 5th stage

3. **[1 marks]** At the end of which stage of MOVZ's execution can the RegWr line be determined?

   Assuming same ALU is used for comparing $c to 0, the result of comparison is known at the end of the 3rd cycle, i.e. EX cycle; thus RegWr line is known at the end of the EX cycle

4. **[4 marks]** Consider data forwarding for the MOVZ instruction when the source register of the 'move' has an RAW dependence on a previous instruction (consider *only* these instances of data forwarding). Draw an appropriate pipeline timing diagram to illustrate the possible scenarios of data forwarding.



[Q4] MIPS Assembly Language [7 marks]

Fill-in-the-blanks to complete the MIPS32 assembly translation of the given C code.

```
struct Complex {
  int re, im;
}
struct Complex C1[], C2[];
int i, N, k;

// Some C code here
// which need not
// be translated

for(i = k; i < N; i++) {
  C1[i] = C2[i];
}

# Assume:
# i in $s0, k in $s1
# N in $s2
# C1 in $s3
# C1's length in $t3
# C2 in $s4
# C2's length in $t4
# Use other temporary
# regs as needed
# Assume int and
# pointers size = 1 word
```

```
# Code begins here; see assumptions given

  addi  $s0, $s1, 0

  __j COMPARE______

LOOP:
  sltu $t0,$s0,$t4  # if i is out of C2's range

  bne $t0,$0,OOB___ # jump to OutOfBounds

  sltu $t0,$s0,$t3  # if i is out of C1's range

  bne $t0,$0,OOB___ # jump to OutOfBounds

  sll  $t0, $s0, 3
  add  $t2, $t0, $s4 # t2 is now &C2[i]
  add  $t1, $t0, $s3 # t1 is now &C1[i]
  # Next four lines achieve C1[i]=C2[i]
  # 1 mark each for next 4 lines

  lw $t0,0($t2)____

  sw $t0,0($t1)____

  lw $t0,4($t2)____

  sw $t0,4($t1)____

  addi $s0, $s0, 1  # i++

COMPARE:

  slt $t0, $s0, $s2

  bne $t0,$0,LOOP___ # loop back as necessary
```

(next question is in next page)

[Q5] Majority Gate [1+1+1+2=5 marks]

Consider the following function: M (x, y, z) = xy + yz + xz. This is called a majority function/gate. This is because the function evaluates to 1(0) only if at least two of the 3 variables are 1(0). In this question, we shall be exploring the majority gates. Turns out that such majority functions have very interesting applications in nanotechnology-based circuits. Implement the following functions using only majority gates. You may use 0 or 1 as input where appropriate. You may also assume that where needed, the complemented input x' of an input variable x is also available, in addition to x itself.

1. F(x) = x ; use exactly one majority gate

   F(x) = M(x, 1, 0)

2. F(x,y) = x+y ; use exactly one majority gate

   F(x,y) = M(x, y, 1)

3. F(x,y) = xy ; use exactly one majority gate

   F(x, y) = M(x, y, 0)

4. F(x,y,z) = xy' + y'z ; use exactly two majority gates

   F(x,y,z) = (x+z)y' = M(M(x,z,1), y', 0)

[Q6] Dharavi and the "once-in-a-century" "pandemic" [optional, 10HP]

The official claim regarding Covid is that it was a highly transmissible, deadly, "once-in-a-century" "pandemic", overwhelming hospitals everywhere. The Dharavi slum is one of the densest and poorest places on earth, with poor access to healthcare. What percentage of people in Dharavi died of Covid? (The next time you meet someone from the working class, ask them whether they found the "pandemic" deadly as claimed).
About 350 Covid deaths among about 800,000 estimated population over 2 years = 0.04% = 0.02% per year
It strains credulity to call this a pandemic overwhelming hospitals
Hospitals were overwhelmed (and people died) in Delhi, New York, London, etc due to exaggerate panic and quack protocols of distancing and killing people with fear itself