# Review Session 2
# Fri. Jan 26, 2:15 pm – 3:05 pm
# Thornton 102

## 1. Agenda

- Announcements
- Homework hints
- Using xspim
- Example problem (time permitting)





## 2. Announcements

- Homework 2, due Thursday, February 1.
- Daxia Ge is our 3rd TA, he will have office hours on Mondays.
- Contact us if you still don't have a lab partner.
- Lab 1 due Tuesday, January 30, you have 4 days.
- Lab students, let us know if you don't have access to P128 (computer cluster) and P129. Email the TA list for login info.

# 3. Homework hints

Problem 1:

- Assume that the length of the arrays is stored at base address + 4 and array[0] is at base address + 8.
- Error checking, index range, array bounds etc.
- `bne $t0, $t1, exceptionDecriptionHere`

Problem 2:

- Aggregate the instruction frequencies i.e.
  Aggr. instr. freq. = 0.5×instr. freq. Int + 0.5×instr. freq. FP
- $CPI_{effective} = \sum_{Intr.types}$ CPI of instr. type×instr. freq. at execution



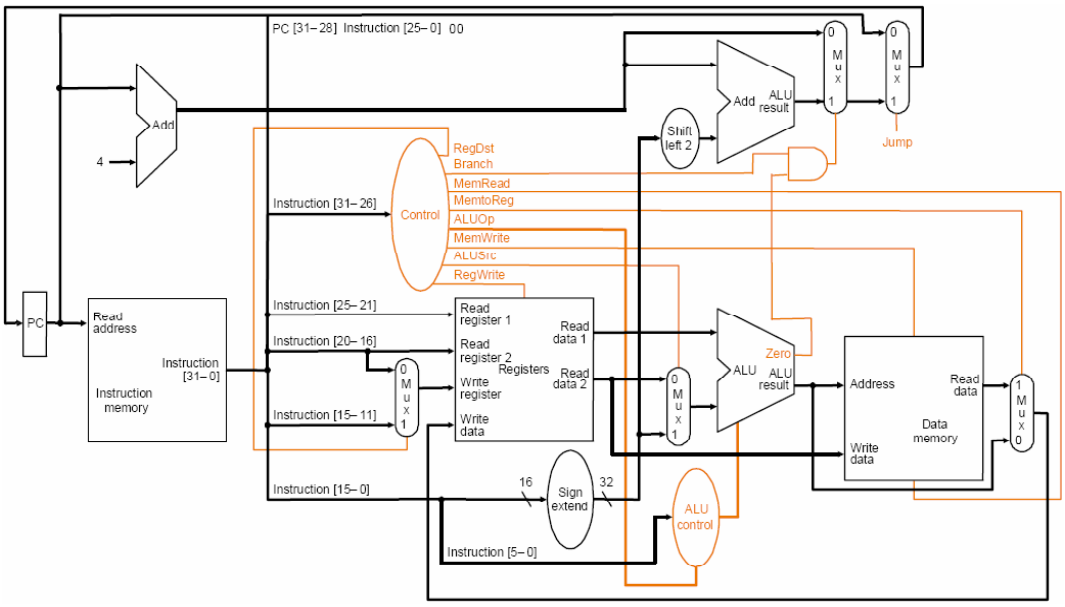


Problem 3:

- Time = Instr. Count × CPI × Clock Cycle Time

Problem 4:

- MIPS hardware architecture will be Tuesday's lecture.
- Think about what steps/operations you can eliminate and how they map to the block diagram.
- Important: extend the modifications to the case if we are no longer implementing **`addi`** or **`addiu`** instructions either.
- Write down the modifications you deem appropriate, it'd be helpful to also make a printout of the block diagram and work off of the given architecture.






EE108B
Winter 2006 – 2007 Session TA: Yi Gu

# Problem 5:

- Register not available → use **`lw`** and **`sw`**.
- Implementation of **`swap`** in hardware means **`swap`** becomes part of MIPS ISA and takes 1 cycle to execute.
- Assume CPI of the original ISA is 1.
- Derive expression for the execution time as a function of the percentage of the
- "What percentage of swap operations in the instruction mix would recommend implementing it in hardware?" Phrased differently: Assume we have code where **`swap`** is native to the MIPS ISA, what fraction R of instructions in such code must be **`swap`** so that the effect of an increased clock period is offset by the effect in a decrease in instruction count in terms of execution time?

## Problem 6:

- Similar to homework 1 problem 1.
- There isn't a one-to-one mapping from the unoptimized to the optimized code.
- If it matches an optimization pattern e.g. strength reduction, then mention it, otherwise just write down what has been done to make code more efficient.
- Dynamic instruction refers to instructions that are actually encountered at run time.
- Pages 87, 108 – 111 helpful for part c.

## Problem 7:

- **`jalr rd, rs`** means jump and link to address in rs, store return address in rd. Similar to problem 4.