Instructions:

- Print your name in the space provided below.
- This examination is closed book and closed notes, aside from the permitted one-page formula sheet. No calculators or other computing devices may be used. The use of any such device will be interpreted as an indication that you are finished with the test and your test form will be collected immediately.
- Answer each question in the space provided. If you need to continue an answer onto the back of a page, clearly indicate that and label the continuation with the question number.
- If you want partial credit, justify your answers, even when justification is not explicitly required.
- There are 7 questions, some with multiple parts, priced as marked. The maximum score is 100.
- When you have completed the test, sign the pledge at the bottom of this page and turn in the test.
- If you brought a fact sheet to the test, write your name on it and turn it in with the test.
- Note that either failing to return this test, or discussing its content with a student who has not taken it is a violation of the Honor Code.

Do not start the test until instructed to do so!

Name Solution

printed

Pledge: On my honor, I have neither given nor received unauthorized aid on this examination.

signed

CHECK IT OUT—OLD COMPUTER SHOPPERS! WOW—IN 1996, $3,000 WOULD GET YOU A 100 MHz PENTIUM SYSTEM WITH A PARALLEL PORT, TWO SERIAL PORTS, A 2MB VIDEO CARD, AND "MS-WINDOWS"
MISC
NICE!
AND $299 WOULD GET YOU A PALM PILOT 1000—16 MHz, 128 KB STORAGE, AND A MEMO PAD, CALENDAR, AND STATE-OF-THE-ART ADDRESS BOOK THAT CAN STORE OVER 100 NAMES!
OOOH!
AND $110 WOULD GET YOU A BULKY TI GRAPHING CALCULATOR WITH AROUND A 10MHz CPU, 24 KB RAM, AND A 96×64-PIXEL B/W DISPLAY!
TIMES SURE HAVE... ...HAVE... UH.
OKAY, WHAT THE HELL, T.I.?
MAYBE THEY COST SO MUCH NOW BECAUSE THERE'S ONLY ONE ENGINEER LEFT WHO REMEMBERS HOW TO MAKE DISPLAYS THAT CRAPPY.

xkcd.com

1. [10 points] Two machines implement support for the same machine language, but have different hardware designs. Aside from the processor, the two machines are identical (RAM, hard drive performance, etc.) For a particular machine language program, Machine A has a substantially lower average CPI than machine B. Nevertheless, execution of that machine language program, with the same input, takes substantially less time on machine B than on machine B. Explain how this is possible. Be precise and cite relevant formulas from our discussion of performance.

Execution time is the product of the number of instructions executed, the average CPI for the program, and the reciprocal of the clock rate. Even though machine A has a higher clock rate, it is entirely possible that machine B has a lower average CPI for this program, in which case the program could execute in less time on machine B.

2. Suppose a program consists of 30% floating point multiply instructions, 20% floating point divide instructions, and the remaining 50% are other instructions. Floating point division, floating point multiplication, and each of the other instructions have the same CPI.

a) [12 points] What speedup can be achieved, for that program, if the execution of a floating point multiplication instruction is made 5/2 times faster and the execution of a floating point division is made 5/3 times faster? Justify your conclusion.

Apply Amdahl's Law: T_after = .50 * T_before + .30 * T_before/(5/2) + .20 * T_before/(5/3)

=.50 * T_before + .12 * T_before + .12 * T_before

= .74 * T_before

So, the speedup is: T_before / (.74 * T_before) = 1 / .74 = 50/37 (about 1.351)

b) [8 points] What is the upper bound on the speedup can be achieved, for the same program, by making the execution of floating point multiplication faster? Justify your conclusion.

Since floating point multiplication makes up only 20% of the instructions, we could not reduce the overall running time to less than 70% of the original time (in fact, we can't even achieve that unless we figure out how to perform a floating point multiplication in time 0).

So, the maximum speedup would be bounded by 1 / .70 ≈ 1.43.

3. [10 points] Computer A has an overall CPI_A of 1.5 and can be run at a clock rate Clk_A of 2.5 GHz. Computer B has a CPI_B of 2.0 and can be run at a clock rate Clk_B of 3.0 GHz. We have a particular program we wish to run. When compiled for computer A, this program will execute exactly $I_A = 40{,}000$ instructions.

Set up an algebraic equation (i.e., symbolic) to determine how many instructions would the program need to execute when compiled for Computer B, in order for the two computers to have exactly the same execution time for this program?

Since execution time equals the product of instruction count, average CPI and clock rate, we are asking when would:

$$I_A * CPI_A / Clk_A == I_B * CPI_B / Clk_B$$

which would be true iff

$$I_B = (I_A * CPI_A * Clk_B) / (CPI_B * Clk_A)$$

4. These questions refer to the simplified single-cycle MIPS32 datapath (full diagram supplied with the test). Recall that this datapath supports the following instructions: `add`, `sub`, `and`, `or`, `slt`, `lw`, `sw`, `beq` and `j`.

a) [12 points] Consider the multiplexor, labeled **4a** on the datapath diagram, that is controlled by the RegDst signal. Which of the supported instructions could not be executed correctly if RegDst was stuck at 0? Why?

If RegDst was stuck at 0, the MUX it controls would always select bits 20:16 to specify the register to be written to.

That would not matter if the instruction does not write to a register (sw, beq, j).

It also would not matter if those were the correct bits to specify the destination register (lw).

It would prevent any R-type (add, sub, and, or, slt) instruction from executing correctly, since for those instructions, the destination register is specified by bits 15:11.

b) [12 points] Consider the Shift left 2 unit, labeled **4b** on the datapath diagram. Which of the supported instructions depend on this unit for their correct operation? Why is it needed?

This Shift left 2 unit is used in the computation of an address that may (or may not) be used when a beq instruction is executed.

That address is never used for any other instruction.

c) [10 points] Suppose the RegWrite signal was stuck at 0. Which of the supported instructions could not be executed correctly? Why?

If RegWrite was stuck at 0, no value would ever be written to a register. That would prevent any of the instructions add, sub, and, or, slt and lw from executing correctly, since all of them must write to a register.

5. [4 points] Why is the single-cycle datapath called the single-cycle datapath?

Because the entire datapath fetches and executes an instruction in one (somewhat long) clock cycle.

6. For some instruction(s), it does not matter whether the MemtoReg signal is set to 0 or 1.

a) [4 points] For which instruction(s) is that true?

sw, beq and j

b) [6 points] The fact that MemtoReg does not matter for some instruction(s) depends upon the presence and correct use of some other feature of the datapath. What is that feature, and how is it relevant to this question?

The output from the MUX controlled by MemtoReg goes to the Write data input to the Register file. If an instruction writes a value into the Register file, the correct value must be delivered to the Register file.

If an instruction does not write a value into the Register file, we do not care what value is delivered to the Register file, as long as the value is not written to a register.

To prevent writes to the Register file, we must have the RegWrite signal and set it correctly.

7. [12 points] Consider the following proposed instruction for the simplified single-cycle MIPS32 datapath discussed in class:

```
exchng   $rt, Imm($rs)        # GPR[rt] = Mem[GPR[rs] + Imm]
                              # Mem[GP[rs] + Imm] = GPR[rt]
```

opcode	rt	rs	Imm

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

The instruction exchanges the values in register `$rt` and at memory address `$rs + Imm`. This could be accomplished by using a sequence of `lw` and `sw` (and other) instructions, but that would require three clock cycles and an extra register for temporary storage.

Assume that the Data memory unit is capable of handling a read operation and a write operation, targeting the same memory address, in a single clock cycle, and that the read will be completed before the write occurs. Then, the given single-cycle datapath can be extended to this new instruction, and all that is required is that the Control unit be modified to recognize the opcode for `exchng` and set the existing control signals correctly.

Assume we've modified the Control unit to handle `exchng`. How should the Control unit set each of the following control signals if a `exchng` instruction is being executed? No justification is required.

Control Signal	**value for `exchng`**
RegWrite	1
ALUSrc	1
MemWrite	1
MemRead	1
MemtoReg	1
RegDst	0

Note: the registers rs and rt were reversed in the description of the machine instruction; if you mentioned that and said it was impossible to execute the instruction correctly, I gave full credit.

If you did not mention that, I graded your answer as if the registers had been shown correctly.