Name: Kevin Lam

Student ID: 9900 6599

EEL 4713: Computer Architecture - Spring 2012
Midterm 1 - 100 points possible

1. (12 pts) What are three drawbacks for taking a single cycle to execute each instruction?

12

1) a single cycle will take as long as the longest instruction
2) need separate instruction and data memories
3) slower program execution because each instruction occurs sequentially one after another

2. (6 pts) If you were to add a no-op instruction (the instruction performs no operation) to the MIPS ISA and considering the multi-cycle datapath, how many cycles would this instruction require?

3

The no op instruction would need only one 2 cycle for instruction decode.

3. (5 pts) Which of the following statements are potential arguments for MIPS to have 32 general-purpose registers (GPRs) $r0-$r31 instead of 64 GPRs? Mark all that apply.

4

1) The ALU can be made smaller with 32 GPRs
(2) The register file can be faster with 32 GPRs
3) The use of a 32-bit architecture implies the use of 32 GPRs
(4) Encoding of R-type instructions with 64 GPRs would lead to very few opcodes available in the 32-bit instruction word
(5) The control logic is simpler with 32 GPRs

4. (10 pts) The MIPS "jump and link" instruction implicitly stores the PC+4 value to a fixed register (R31). Discuss the implications, if any, if the ISA was changed such that any arbitrary register could be specified.

10

The JAL instruction is J type so its instruction consists of the 6 bit op code and a 26 bit address. If any register could be specified to hold the return address, we'd have to add a 5 bit register field to the instruction and shorten the address field. This means that our jump and link has a smaller range it can jump to.

5. (7 pts) Which method, callee or caller saved registers, may result in fewer assembly instructions and why.

7

Callee saved requires less assembly instructions. The callee only needs to save the $s registers that it knows its using. The caller, not knowing what the callee might use, would have to save all of its registers.

6. (10 pts) In MIPS ISA the source registers are fixed (*$rs* and *$rt*), while the destination register can be *$rd* (R-type) or *$rt* (I-type). Why is this approach preferable to keeping the destination register fixed and the source registers different?

10

Keeping the source registers fixed means the ISA can get the register information while the instruction is being decoded. We don't need to know what type of instruction it is to begin. Once we know what kind of instruction (R or I type), we can look up $rd. If the source was not fixed, we'd have to know instruction type first to find the reguter data.

7. (50 pts total) Consider the addition of a new instruction into the MIPS instruction set given the multi-cycle instruction execution implementation. This instruction, *inc2 $rd, $rt*, is used to automatically increment the values stored in both registers *$rd* and *$rt* in a single instruction (i.e., *$rd* = *$rd* + 1 and *$rt* = *$rt* + 1).

49.5

a) (10 pts) Discuss why this instruction cannot be supported by the current multi-cycle datapath?

9

10

In the current multi cycle data path, this new instruction would take 2 addi instructions. Even with our multi cycle datapath, we would decode both registers and need to do an ALU operation on both of them. We cannot because we'd need 2 ALU's or a latch to hold data while ALU did 2 operations.

Kevin Lam
9900 6599

95.5/100

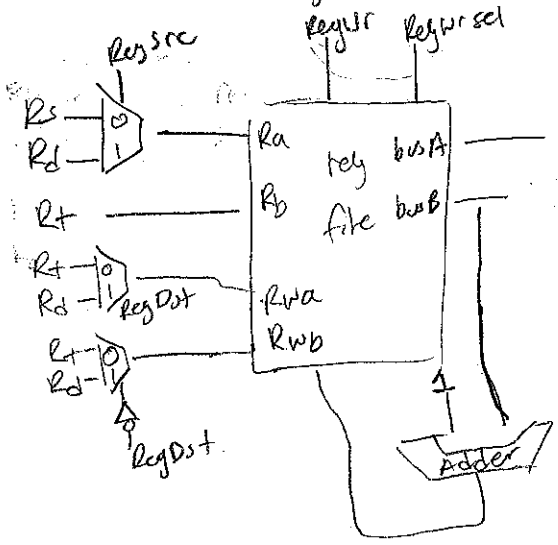
b) (15 pts) If the instruction cannot be supported by the current multi-cycle datapath, discuss the architectural and control logic changes, at a high level, that are necessary to support this instruction.

15

Inc2 $rd $rt

* assume rs is not used

The first change would be to the registerFile. $Rd would have to be able to be read into the Ra port of RegisterFile. We could put a mux in front of Ra to determine Rs or Rd when this instruction is used. It would have a special select control signal called RegSrc. We would also need to add a second write line/bus Rwb so that we can write 2 different registers.



Next we'd add a special adder (for +1) connected to bus B of register file. It will add 1 and feed the result back into our new write bus on RegFile.

From Bus A, we do a normal Add except that we add 1 as one of the options in the ALUSel B mux and select it.

great!

One more new control signal, RegWrSel would select which Rw to select and which bus to write. Normally, this would be 0 and the adder data from Bus B does nothing. In this instruction, it writes both registers from both buses.

d) (15 pts) How many cycles will this new instruction require? What will be done on each cycle?

15

This instruction requires instruction fetch/decode, register fetch/decode, Rtype exec, and then Rtype finish. 4 cycles in all. The R type exec would set the new control signals and perform both adds. The last cycles would have to do 2 writes (for both registers) to register File. So there are 4 cycles in all.

e) (5 pts) Does the clock cycle time need to change to support this new instruction?

4.5

Yes, since we need to do 2 writes in a cycle, the clock cycle time would increase. ~~put~~ in parallel so no increase. But, add another path would increase access time slightly

f) (5 pts) What instruction type (I-, R-, J-type) would be most appropriate for this instruction and why?

5

R-type would be best. The instruction uses Rd so we need the R type format.

[opcode] [Rs is not used] [rt] [rd] [shamt not used] [func]

* if Rd were Rs, then I type is best, we would use Rs and Rt and make the immediate value = 1