Instructions

Format: This exam consists of 6 multiple choice questions for each of the 12 lectures in the course. You may have to do some calculations in order to determine the correct answer. The motivation for having so many questions is to get a more accurate assessment of your knowledge of the material.

Be sure to circle the answers on the page titled “answers page” or your answers will not be graded. If you choose not to answer a question do not circle a letter. Feel free to detach the answers page from the remainder of the exam and only turn it in.

Grading: To discourage guessing, each incorrect answer will be worth -1/3 point, while each correct answer is worth +1 point. I.e., if you guess randomly, the expected score is 0.

Material: You are allowed a calculator and one double-sided, hand-written A4 sheet of notes during the exam.

Good luck!
Answers page.
Only answers circled here will be graded.

0. Optional anonymous background questions
   0.1. A B C
   0.2. A B C
   0.3. A B C

1. ISA 1
   1.1. A B C D
   1.2. A B C D
   1.3. A B C D
   1.4. A B C D
   1.5. A B C D
   1.6. A B C D

2. ISA 2
   2.1. A B C D
   2.2. A B C D
   2.3. A B C D
   2.4. A B C D
   2.5. A B C D
   2.6. A B C D

3. Computer Arithmetic
   3.1. A B C D
   3.2. A B C D
   3.3. A B C D
   3.4. A B C D
   3.5. A B C D
   3.6. A B C D

4. Logic
   4.1. A B C D
   4.2. A B C D
   4.3. A B C D
   4.4. A B C D
   4.5. A B C D
   4.6. A B C D

5. Processor Control and Datapath
   5.1. A B C D
   5.2. A B C D
   5.3. A B C D
   5.4. A B C D
   5.5. A B C D
   5.6. A B C D

6. Pipelining
   6.1. A B C D
   6.2. A B C D
   6.3. A B C D
   6.4. A B C D
   6.5. A B C D
   6.6. A B C D

7. Hazards
   7.1. A B C D
   7.2. A B C D
   7.3. A B C D
   7.4. A B C D
   7.5. A B C D
   7.6. A B C D

8. Branch Prediction and Exceptions and Interrupts
   8.1. A B C D
   8.2. A B C D
   8.3. A B C D
   8.4. A B C D
   8.5. A B C D
   8.6. A B C D

9. Input/Output
   9.1. A B C D
   9.2. A B C D
   9.3. A B C D
   9.4. A B C D
   9.5. A B C D
   9.6. A B C D

10. Caches
   10.1. A B C D
   10.2. A B C D
   10.3. A B C D
   10.4. A B C D
   10.5. A B C D
   10.6. A B C D

11. Virtual Memory
   11.1. A B C D
   11.2. A B C D
   11.3. A B C D
   11.4. A B C D
   11.5. A B C D
   11.6. A B C D

12. Parallelism
   12.1. A B C D
   12.2. A B C D
   12.3. A B C D
   12.4. A B C D
   12.5. A B C D
   12.6. A B C D

Feel free to detach the remaining pages of the exam and turn in only your answers.
0. OPTIONAL anonymous background questions
(These are not graded and will not affect your grade on the exam. They are for me to be able to correlate what helps students prepare for the exam. Feel free to not answer these questions if you do not want to.)

1. How much of the book did you read?
   a. All
   b. Some
   c. None

2. How many of the practice sessions did you miss/turn in late?
   a. Several
   b. One
   c. None

3. How many of the previous years’ exams did you look at?
   a. All
   b. Some
   c. None

1. ISA 1

1. Which address would be an aligned address on a machine that only loads 64-bit words with a byte-addressable memory?
   a. 2
   b. 4
   c. 8
   d. 12

2. How many control flow instructions are needed to implement a simple for-loop?
   a. 1
   b. 2
   c. 3
   d. Depends on whether you end the loop at 0

3. Why does the PC need information from the ALU?
   a. It uses the ALU to do PC+4
   b. It uses the ALU to load the instruction
   c. It uses the ALU to decide the next instruction
   d. All of the above

4. How many fully usable general-purpose registers does MIPS have?
   a. 31
   b. 32
   c. 33
   d. More than 33

5. How many temporary registers are required to compute A-(B+C) if you need A, B, and C later?
   a. 0
   b. 1
   c. 2
   d. 3

6. From where do immediate instructions get their immediate values?
   a. The register file
   b. The instruction
   c. The ALU
   d. Depends on the instruction type
2. ISA 2

1. Why do we only lose one register field for an I-format instruction vs. an R-format instruction?
   a. Immediate values only need as many bits as a single register field
   b. The immediate value replaces other fields and not the register fields
   c. The immediate value replaces some of the register fields and some other fields
   d. The immediate instructions have more bits

2. What do we do to the immediate field for branch instructions to calculate the branch address and why?
   a. Shift to the left by 2 because instructions are word-aligned.
   b. Shift to the left by 4 because instructions are word-aligned.
   c. Add 4 because instructions are word-aligned.
   d. Sign-extend because the immediate is unsigned.

3. How would you load the value 13 into the PC?
   a. addi $ra, $r0, 13
      jr $ra
   b. addi $pc, $r0, 13
   c. lui $pc, 0
      ori $pc, 13
   d. lui $pc, 13

4. What immediate value should the conditional branch at address 12 have to jump to address 8?
   a. -1
   b. -2
   c. -4
   d. -8

5. Why do we have a convention for callee and caller saving of registers?
   a. So we can call any function without overwriting our own registers or the function’s registers.
   b. To avoid corrupting the stack.
   c. To make it easier for the compiler to generate code.
   d. All of the above.

6. A function uses registers $s0, $s1 and $t3. How much will it adjust the stack when it is called?
   a. -1
   b. -2
   c. -4
   d. -8

3. Computer Arithmetic

1. Why do we use two’s compliment notation?
   a. There is only one zero
   b. Subtraction is just invert addition with the carry in set to 1
   c. We can easily determine if the number is negative by looking at the MSB
   d. All of the above

2. How many bits are required to compute the following in two’s compliment without an overflow? -8+8=0
   a. 4
   b. 5
   c. 6
   d. 7

3. What number format gives you the largest range for a 3-bit fixed-point number?
4. Why did processors switch from serial multipliers to parallel multipliers?
   a. Parallel multipliers became easier to design
   b. We had enough transistors that we didn’t care about size
   c. Serial multipliers require less logic
   d. All of the above

5. Which of these floating point operations is most likely to cause a divide by zero exception if there aren’t enough bits in the number representation?
   a. \[ \frac{1}{(2.1 \times 10^{32} + 5.6 \times 10^{2}) - 2.1 \times 10^{32}} \]
   b. \[ \frac{1}{(5.6 \times 10^{2} + 2.1 \times 10^{2}) - 5.6 \times 10^{2}} \]
   c. \[ \frac{1}{(2.1 \times 10^{32} - 2.1 \times 10^{32}) - 5.6 \times 10^{2}} \]
   d. \[ \frac{1}{(5.6 \times 10^{2} - 2.1 \times 10^{2}) - 5.6 \times 10^{2}} \]

6. Floating point normalization solves what problem?
   a. Exponential spacing of the values on the number line
   b. Linear spacing of the values on the number line
   c. Multiple representations for the same value
   d. Needing to shift the exponent for addition

4. Logic

1. If you choose two groups of 2 in a Karnaugh map instead of 1 group of 4, what do you get as the result?
   a. The wrong logic equation
   b. A logic equation with twice as many terms
   c. A logic equation with half as many terms
   d. A logic equation with XOR gates

2. How many inputs will a 4-bit mux with a 3-bit select signal have?
   a. 3 inputs each 4 bits wide
   b. 4 inputs each 3 bits wide
   c. 8 inputs each 4 bits wide
   d. 16 inputs each 3 bits wide

3. Why does a decoder not have input 0 connected to the logic that generates the output?
   a. It does.
   b. Input 0 corresponds to binary output 0, which is all false, so we don’t use the input to generate any output.
   c. You can never have just input 0 active because the inputs are one hot.
   d. Zero cannot be represented with a one hot input.

4. Why are edge-triggered flipflops better than transparent latches?
   a. They are faster because they only look at edge of the clock signal instead of the whole time it is high or low
   b. They allow the input to pass directly through to the output, so they speed up the circuit
   c. They don’t allow the input to pass directly through to the output, so they avoid feedback loops
   d. They are smaller than transparent latches because they only need two inverters

5. Why is DRAM so much cheaper than SRAM?
   a. DRAM cells are made of two inverters so they are smaller
   b. DRAM cells are made of a single capacitor so they are smaller
   c. DRAM cells are faster so you need fewer of them
   d. DRAM cells are larger so they are easier to make
6. Why is SRAM so much faster than DRAM?
   a. It isn’t.
   b. SRAM cells are made of powered transistors so they can output a stronger signal, which means they can send out their data faster
   c. SRAM cells are made of capacitors so they are smaller and their output doesn’t have to travel as far
   d. SRAM uses caches to make it faster

5. Processor Control and Datapath

1. Why does our basic processor design have two memories?
   a. It doesn’t
   b. We don’t have a cache, so accessing the memory is slow
   c. Different parts of the instruction access the instruction memory and the data memory
   d. Some instructions need to access the instruction memory and the data memory at the same time

2. How many bits are needed for “Write register” in the figure?
   a. 1
   b. 5
   c. 16
   d. 32

3. What is the output of the sign extender for non I-format instructions?
   a. Zero: non-I-format instructions do not have an immediate to sign-extend
   b. Nothing: non-I-format instructions do not have an immediate so the sign-extension logic is turned off
   c. Garbage: non-I-format instructions do not have an immediate so the value is whatever bits were in that part of the instruction
   d. Undefined: non-I-format instructions do not have an immediate so the sign-extension logic has unknown values coming into it and produces unknown values

4. What determines the value of the ALUSrc MUX?
   a. The instruction type: I=Read data 2, R=Sign-extend
   b. The instruction type: I=Sign-extend, R=Read data 2
   c. The ALU Zero output: True=Read data 2, False= Sign-extend
1. What are the values of ALUSrc, MemWrite, MemtoReg, and RegWrite for a lw instruction?
   a. ALUSrc=Read-data-2; MemWrite=0; MemtoReg=Read-data; RegWrite=1
   b. ALUSrc=Sign-extend; MemWrite=0; MemtoReg=Read-data; RegWrite=1
   c. ALUSrc=Sign-extend; MemWrite=0; MemtoReg=ALU-result; RegWrite=1
   d. None of these

2. What is the ALU operation for a sw instruction?
   a. Add
   b. Subtract
   c. Pass through
   d. Undefined (ALU results not used)

6. Pipelining

1. Why did pipelining the processor help if all instructions take the same 5 pipeline stages?
   a. We can execute the instructions in parallel
   b. We can execute each instruction faster
   c. We can move data through the processor faster
   d. It didn’t: pipelining only helps when the instructions take different amounts of time

2. What is the speedup for a processor when we add 10 ideal pipeline stages, but can only keep them half full with instructions?
   a. 10x
   b. 5x
   c. 2x
   d. Need more information

3. What is the change in throughput for a processor that takes 100ns per instruction unpiplined, when it is pipelined with 100 stages, where each stage needs a 2ns pipeline register? (Assume we can keep the pipeline completely full.)
   a. 1/3 (lower throughput)
   b. 3x
   c. 33x
   d. 100x

4. What is the change in latency to complete each individual instruction for a processor that takes 100ns per instruction unpiplined, when it is pipelined with 100 stages, where each stage needs a 2ns pipeline register? (Assume we can keep the pipeline completely full.)
   a. 1/3 (lower latency)
   b. 3x
   c. 33x
   d. 100x

5. What happens in the MEM stage for an addi instruction?
   a. Nothing
   b. Load the value
   c. Calculate the ALU result
   d. Write back the ALU result

6. What is the most important thing for getting performance from a pipelined design?
   a. Have as few pipeline registers as possible
   b. Make each stage fast
   c. Keep the pipeline full
   d. All of the above

7. Hazards

1. What is needed to resolve the following hazard?
   sub R4, R13, R14
   add R14, R4, R13
a. Forwarding from WB to EX
b. Forwarding from MEM to EX
c. Double-pumping the RF
d. Can’t resolve: need to stall

2. What is needed to resolve the following hazard?
   
   lw R4, 12(R14)
   add R14, R4, R13
   a. Forwarding from WB to EX
   b. Forwarding from MEM to EX
   c. Double-pumping the RF
   d. Can’t resolve: need to stall

3. Is stalling the pipeline a good way to solve hazards?
   a. No, it hurts performance too much
   b. No, it requires special compilers to insert stalls
   c. A and B
   d. None of the above

4. What does the data hazard detection logic do?
   a. Look at the instructions in the different pipeline stages to see if they have conflicting register file accesses
   b. Look at the instructions in the different pipeline stages to see if they write to the register file
   c. A and B
   d. None of the above

5. How many branch delay slots do we have if the branch decision is made in the MEM stage?
   a. 0
   b. 1
   c. 2
   d. 3

6. What does the ALU do on a beq instruction if we move the branch computation to the ID stage?
   a. subtract to compute the jump condition
   b. add to compute the jump condition
   c. nothing
   d. doesn’t matter

7. If 20% of the instructions are branches and we can only fill the branch delay slot 50% of the time, how much of a slowdown do we experience due to the branch delay slot?
   a. 5%
   b. 10%
   c. 20%
   d. 25%

8. Branch Prediction and Exceptions and Interrupts
   1. Why do we need branch prediction on modern processors?
      a. The processor runs too fast to stall the pipeline
      b. The pipelines are so long that we could never fill all the branch delay slots
      c. The pipeline is too long to kill instructions from a wrong branch
      d. All of the above
   2. What is the slowdown for processor that does one instruction per cycle and has a branch predictor that is 95% accurate with a 50 cycle misprediction cost? (Assume 20% of instructions are branches.)
      a. 2x faster
      b. 1.5x slower
3. What is the reasoning behind the BTFN branch predictor?
   a. Loops are the most common taken branches and they are usually backwards
   b. Error code usually has forward jumps and are not frequently taken
   c. We guess 50% each way
   d. All of the above

4. What happens if you have a branch predictor with 8 entries, but your program has 9 branches?
   a. The predictor miss-predicts one of them
   b. The predictor confuses two of them
   c. The predictor puts one on the stack to hold the 9th branch
   d. The program crashes

5. Why n-bit branch predictors better than 1-bit branch predictors?
   a. You can make more miss-predictions before forgetting your previous prediction
   b. You can predict for more branches
   c. You can predict more complicated patterns
   d. All of the above

6. When an exception occurs, what happens to the other instructions in the pipeline?
   a. They are all killed
   b. They are killed if they come after the instruction with the exception
   c. They are killed if they have a data dependency on the instruction with the exception
   d. The operating system decides which instructions to kill

9. Input/Output

1. What is the worst-case for accessing data on a hard disk?
   a. The time it takes to move the head from the inner-most track to the outer-most track
   b. The time it takes to rotate the disk all the way around
   c. The sum of A and B
   d. All data takes the same amount of time to access

2. Rank the following technologies in terms of cost per MB: flash, DRAM, hard disk, tape
   a. (cheapest) flash < DRAM < hard disk < tape
   b. (cheapest) tape < hard disk < DRAM < flash
   c. (cheapest) tape < DRAM < flash < hard disk
   d. (cheapest) tape < hard disk < flash < DRAM

3. Why is it hard to make busses fast?
   a. Wires interfere with their neighbors
   b. Wires of different lengths take different amounts of time
   c. Wires get slower as they get smaller
   d. All of the above

4. What is the benefit of differential serial signaling over busses?
   a. Differential signals resist noise because it gets subtracted out at the end
   b. Differential signals resist noise because it gets added in at the end
   c. Serial signals are easier to build than parallel busses
   d. Serial signals use a slower clock because they send data in parallel

5. What is the benefit of memory-mapped IO?
   a. Faster
   b. Automatically moves data
   c. Uses regular instructions to access IO
   d. Does not pollute the address space

6. If you want to write a program with the absolute lowest possible latency to access an IO device, what approach would you use and why?
Introduction to Computer Architecture, August 2013 Makeup Exam

10. Caches
1. Why do we want caches?
   a. DRAM is slow
   b. Processors are fast
   c. Lots of instructions access memory
   d. All of the above
2. Approximately how many processor cycles does it take to access data in DRAM? (E.g., on a cache miss.)
   a. 1
   b. 10
   c. 100
   d. 1000
3. How many lines will each way have in 32kB 8-way set associative cache with a 64 byte line size?
   a. 4096
   b. 512
   c. 64
   d. 8
4. Why are write-back caches better than write-through?
   a. Simpler: use a dirty bit to keep track of which data has been changed
   b. Faster: don’t have to write back data to DRAM when it is changed
   c. Smaller: don’t need a valid bit because they keep the changed data
   d. All of the above
5. How many hits will the following access pattern generate in a 2-way set associative cache with 8 entries. (Assume the line size is 1 entry per line.) 4, 5, 6, 12, 13, 4, 5, 1, 12, 13, 5
   a. 2
   b. 3
   c. 4
   d. 5
6. What happens to the average memory access time on a machine that takes 100 cycles to access DRAM, when you move from a 32kB cache (20% miss ratio, 1 cycle cache hit or miss) to a 2MB cache (5% miss ratio, 4 cycle cache hit or miss)?
   a. Goes up
   b. Stays the same
   c. Goes down
   d. Need more information

11. Virtual Memory
1. Virtual memory does not do which of the following?
   a. Provide security by preventing programs from changing each other’s data
   b. Provide flexibility by letting programs use more memory than is physically available
   c. Provide flexibility by letting the OS put programs anywhere in physical memory
   d. Provide performance by making DRAM accesses faster
2. How much data can a program access without a TLB miss if the TLB has 8 entries and the cache is 16kB with a 64 byte line size.
   a. 512 bytes
b. 16kB
  c. 32kB
  d. Don’t have enough information

3. What is the average CPI of a program that is 33% load/store instructions, run on a 1 CPI processor with a perfect cache, and a perfect TLB that takes 2 cycles per translation?
   a. 0.66
   b. 1.66
   c. 2.66
   d. 3.66

4. Why is a virtually indexed, physically tagged cache a good design?
   a. You can do the translation and cache lookup in parallel
   b. The size of the cache is limited by the associativity and page size
   c. You don’t need the translation to determine if you have a cache hit
   d. All of the above

5. Why is a virtually indexed, virtually tagged cache a good design?
   a. You can do the translation and cache lookup in parallel
   b. The size of the cache is limited by the associativity and page size
   c. You don’t need the translation to determine if you have a cache hit
   d. All of the above

6. Two programs want to share data. Can this be done with virtual memory?
   a. No, each program has its own virtual address space that maps to its own physical memory.
   b. Yes, they can both have pages that map to the same physical memory.
   c. Only if they use a physically-indexed and physically-tagged cache.
   d. Only if they use a virtually-indexed and virtually-tagged cache.

12. Parallelism

1. Why did clock speeds stop increasing in 2004?
   a. We can’t make faster transistors
   b. The memory was too slow so there was no point
   c. We couldn’t cool the chips
   d. All of the above

2. If a program is 80% parallelizable and I have 5000 processors, what is the best speedup I could achieve?
   a. 5x
   b. 4000x
   c. 5000x
   d. Need more information

3. How do locks avoid data synchronization problems?
   a. They prevent any other processor from changing the data while it is locked.
   b. They prevent any other processor from accessing the data while it is locked.
   c. They let the program keep track of whether any other processors are currently accessing the data.
   d. They make copies of the data for each processor so they can update them at the same time.

4. What problem does cache snooping solve?
   a. Prevents multiple processors from accessing the same data at the same time
   b. Keeps DRAM up-to-date with changes that processors make in their own caches
   c. Keeps each processor’s cache up-to-date with changes that other processors make in their own caches
   d. All of the above
5. What has happened as a result of incorrectly synchronized data accesses from parallel processors?
   a. Programs crashed
   b. Banks lost money
   c. People died
   d. All of the above

6. What speedup would we expect from a triple-issue pipeline that can do 1 load/store instruction and 2 other instructions at the same time for a program that is 1/3 load/store instructions?
   a. Less than 3x
   b. 3x
   c. More than 3x, but less than 6x
   d. More than 6x

Make sure you circled the answers on the answer page. Answers on other pages will NOT be graded.