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!

Note: Please take this exam seriously and pass it so you do not have to take future makeup exams! While it is essential to give students appropriate chances to demonstrate their knowledge, producing and grading an exam consumes significant teaching resources that could be spent otherwise improving the course. For example, this exam will cost the IT department roughly 10,000SEK in teaching time to prepare, administer, and grade.
0. Optional anonymous background questions
0.1. A B C
0.2. A B C
0.3. A B C D
0.4. Program: ______________

1. ISA 1
1.1. A B C
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.

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

2. How many practice exams did you do?
   a. None
   b. One
   c. Several

3. How much time did you spend studying for this exam?
   a. None
   b. <2 hours
   c. 2-8 hours
   d. >8 hours

1. ISA 1

1. Why is it less efficient to compute directly on data in memory than data in registers?
   a. Registers are faster than main memory
   b. Registers require fewer address bits than main memory
   c. A and B
   d. Computing on data in memory is as efficient as data in registers

2. Why do we want our processor to appear to be atomic and sequential?
   a. Smaller
   b. Easier to build
   c. Faster
   d. Easier to understand

3. What address does the following code load?
   ```
   addi R2, R0, 10
   addi R1, R2, 2
   addi R2, R0, 12
   lw R2, 24(R1)
   ```
   a. 12
   b. 24
   c. 36
   d. 48

4. What value should be put in X to load the word starting at address 26 into R8?
   ```
   lw R2, 24(R0)
   sll R2, R2, X
   lw R3, 28(R0)
   str R3, R3, 16
   or R8, R2, R3
   ```
   a. 8
   b. 16
   c. 24
   d. 32

5. How much space does a branch label take up in the machine code?
   a. 0 words
   b. 1 word
   c. 1 word
   d. 1 byte per character in the label name

6. Which of the following MIPS registers can be written with a regular add instruction?
a. R0
b. RA
c. PC
d. None can be written with a regular add

2. ISA 2

1. What problem does the stack solve?
   a. Storing data before loading it into a register and after removing it from a register
   b. There are not enough registers for each procedure to have its own
   c. Procedures do not know which registers their caller and callees use
   d. How to access more than 32 words of data

2. What problem does the MIPS calling convention solve?
   a. Storing data before loading it into a register and after removing it from a register
   b. There are not enough registers for each procedure to have its own
   c. Procedures do not know which registers their caller and callees use
   d. How to access more than 32 words of data

3. The following code is part of a library and may be called by other programs. What registers does the compute function need to save on the stack for it to work correctly?

```assembly
begin:
    add $t0, $t1, 3
    ld $a0, 16($t0)
    jal compute
    add $s0, $v0, $t0
    ... (program ends eventually)
compute:
    add $s1, $a0, $a0 // double the input
    addi $t1, $s1, -2 // subtract 2
    beq $t1, $r0, skip // go to skip if it is zero
    add $v0, $t1, $r0 // otherwise set the result to the double
    jr $ra // return result
skip:
    addi $v0, $r0, 2 // set it to 2 if it was zero
    jr $ra // return result
```

a. None
b. $s1
c. $s1 and $s0
d. $s1 and $t1

4. The compute function will never be called by other code or from another place. What registers does the compute function need to save on the stack for it to work correctly?

```assembly
begin:
    add $t0, $t1, 3
    ld $a0, 16($t0)
    jal compute
    add $s0, $t20, $t0
    ... (program ends eventually)
compute:
    add $s1, $a0, $a0 // double the input
    addi $t1, $s1, -2 // subtract 2
    beq $t1, $r0, skip // go to skip if it is zero
    add $v0, $t1, $r0 // otherwise set the result to the double
    jr $ra // return result
skip:
    addi $v0, $r0, 2 // set it to 2 if it was zero
    jr $ra // return result
```

a. None
b. $s1
c. $s1 and $s0
d. $s1 and $t1

5. Does a procedure need to save $a and $v registers on the stack when it calls a subprocedure?
   a. No, they are callee save
b. Depends on whether they are needed again after the sub-procedure returns
c. No, they are special registers
d. No, it is guaranteed to either not need them or overwrite them

6. This is the last code in a procedure. What instructions should be placed in _______?

```
lw $t0, 0($sp)
lw $t1, 4($sp)

______
```

a. addi $sp, $sp, 8; jr $ra
b. addi $sp, $sp, -8; jal procedure
c. addi $sp, $sp, 2; jr $ra
d. addi $sp, $sp, -2; jal procedure

3. Computer Arithmetic

1. Why do computers use binary signals (1 and 0 only) instead of analog signals (including fractional values)?
   a. We cannot build math circuits for analog signals
   b. Analog signals change more slowly
   c. Binary signals are more immune to noise
   d. More than one of the above

2. What is the result of the following signed-magnitude addition: 1000+0001?
   a. -1
   b. 0
   c. 1
   d. 8

3. What would the following 2’s complement math compute: (!A+1) + (!B + C)
   a. A-C+B
   b. C-A-B+1
   c. C-A-B
   d. C-A+B

4. Which of the following number formats has only one zero?
   a. Signed magnitude
   b. Two’s Complement
   c. Floating Point
   d. More than one of the above

5. How many bits do you need to store the temperature in Sweden to an accuracy of 0.2 degrees, assuming it ranges from -40 to +20?
   a. 6.3 bits
   b. 7 bits
   c. 9 bits
   d. You can’t perfectly represent 0.2 in binary

6. What is the largest magnitude number you can represent with a two’s complement fixed-point ddd.dd format?
   a. -4
   b. -3.75
   c. 3.75
   d. 4

4. Logic

1. What does the clock do in a counter circuit?
   a. Control when the computed next value is stored as the current value
   b. Control the storage element to prevent a feedback loop
   c. Make sure the next value logic has enough time to compute the next value
   d. All of the above
2. Which is faster: DRAM or SRAM and why?
   a. DRAM: it uses capacitors to drive the signal onto the output wire
   b. DRAM: it uses a powered feedback loop to drive the signal onto the output wire
   c. SRAM: it uses capacitors to drive the signal onto the output wire
   d. SRAM: it uses a powered feedback loop to send the signal onto the output wire

3. What will be the value of R12 after executing the following two instructions?
   and R12, R12, R12
   or R12, R12, R0
   a. 0x0000 (Zero)
   b. 0xFFFF (All ones)
   c. It depends on the value of R12 before the first instruction
   d. The value of R12 will not change

4. What signals do you put into A and B to build a latch that stores data on the rising edge of the clock?
   a. A=CLK, B=!CLK
   b. A=CLK, B=CLK
   c. A=!CLK, B=CLK
   d. A=!CLK, B=!CLK

5. Which is true about circuit above if both A and B are attached to CLK?
   a. The circuit can cause feedback when the CLK is low
   b. The circuit is slower
   c. A and B
   d. Neither

6. A designer connects a long chain of AND gates together. (E.g., the output from the first AND goes into one of the inputs of the second AND and so forth.) The circuit uses 1.0V to represent 1 and 0.0V to represent 0. When the circuit comes near a mobile phone it experiences +0.1V of noise on the wires between each AND gate. How many AND gates can be put in a chain until the noise can cause the wrong result?
   a. 0
   b. 1
   c. 5
   d. No limit
5. Processor Control and Datapath

1. How closely does the single-cycle datapath respect the ISA’s promise of sequential and atomic execution?
   a. Sequential, but NOT atomic
   b. NOT sequential, but atomic
   c. BOTH Sequential AND atomic
   d. NEITHER sequential NOR atomic

2. Why do we not have hazards in the single-cycle processor?
   a. Every instruction finishes before the next one starts so the register file always has the right data
   b. No instructions are executing in parallel so there are no conflicts between instructions
   c. Each instruction has access to the whole processor while it is executing so there is no fighting for resources
   d. All of the above

3. What are the control signals in the above datapath for a lw instruction?
   a. ALUSrc=SignExt, ALUOp=Add, RegWrite=1, MemRead=1, PCSrc=PC+4
   b. ALUSrc=SignExt, ALUOp=Add, RegWrite=1, MemRead=1, PCSrc=ALU
   c. ALUSrc=SignExt, ALUOp=NOP, RegWrite=1, MemRead=1, PCSrc=PC+4
   d. ALUSrc=Reg, ALUOp=Add, RegWrite=1, MemRead=1, PCSrc=PC+4

4. Why do we need a MUX going into Write register in a full design?
   a. To choose the data to write into the register file
   b. To choose whether we decode an R- or I-format instruction
   c. To choose whether the destination is in the rt or rd position
   d. To choose which register we write to for all instructions

5. Which instructions are not supported on the datapath design above?
   a. j
   b. jal
   c. jr
   d. All of the above

6. What do you not need to know to determine PCSrc in the datapath design above?
6. Pipelining

1. Why do single-cycle processors have slower clock speeds than pipelined processors?
   a. They are limited by the slowest instruction
   b. They have instructions that take different lengths of time
   c. They can only execute one instruction at a time
   d. All of the above

2. What is needed from a pipelined processor to get better performance than a single-cycle one?
   a. Keep the pipeline full
   b. Execute multiple instructions in parallel
   c. A higher clock speed
   d. All of the above

3. How much faster does a processor run if we divide each instruction into 8 cycles, run the clock 4 times faster, and only run one instruction at a time?
   a. 32x faster
   b. 2x faster
   c. 2x slower
   d. 32x slower

4. Why do we need to keep the RF2 data in the EX/MEM pipeline register?
   a. For the ALU to handle R-type instructions
   b. For the memory data
   c. For the pipeline to write back to the register file
   d. We don’t

5. A single-cycle processor takes 80ns per instruction. It is perfectly pipelined into 10 stages with pipeline registers that take 2ns each and can be kept 50% full. What change in instruction throughput do you expect compared to the single-cycle processor?
   a. 10x
   b. 5x
   c. 4x
   d. 2x
6. A single-cycle processor takes 80ns per instruction. It is perfectly pipelined into 10 stages with pipeline registers that take 2ns each and can be kept 50% full. What change in instruction latency do you expect compared to the single-cycle processor?
   a. 1.0x
   b. 1.25x
   c. 0.75x
   d. 0.5x

7. Hazards
   1. When can we forward data in a pipeline?
      a. Whenever we need data from an instruction before us
      b. When multiple instructions need the same hardware at the same time
      c. When the data is in another stage
      d. All of the above
   2. Why are hazards a problem?
      a. Because we only have one register file
      b. Because the ISA promises sequential execution
      c. Because writeback happens at the end of the pipeline
      d. Because the ISA promises atomic execution
   3. Why can we not forward between these instructions to execute them back-to-back?
      \( \text{lw} \ R3, 12(R1) \)
      \( \text{add} \ R2, R1, R3 \)
      a. It’s a structural hazard so we need to add more hardware
      b. We can forward if we add another forwarding path
      c. We would break the serial and atomic ISA promise
      d. The data is not available when the add would need it
   4. How many register values need to be forwarded in the following code?
      \( \text{xor} \ R3, R4, R7 \)
      \( \text{add} \ R6, R4, R5 \)
      \( \text{sub} \ R2, R3, R4 \)
      \( \text{beq} \ R2, R3, \text{done} \)
      a. 0
      b. 1
      c. 2
      d. 3
   5. The standard MIPS pipeline with no forwarding has 50% of the instructions use data from the previous instruction. On average, the pipeline finishes one instruction every how many cycles?
      a. 1.0
      b. 1.33
      c. 1.5
      d. 2.0
   6. A processor is ideally pipelined to 20 stages, but due to hazards, 10% of the instructions are NOPs. What is the speedup over the original processor?
      a. 2x
      b. 10x
      c. 18x
      d. 20x

8. Branch Prediction and Exceptions and Interrupts
   1. Why do we need branch predictors on modern processors?
      a. The pipelines are so long that we could never fill all the branch delay slots
      b. The pipeline is too long to kill instructions from a wrong branch
      c. The processor runs too fast to stall the pipeline
2. When do predictors help avoid hazards?
   a. When the answer hasn’t been computed yet
   b. When multiple instructions need the same hardware at the same time
   c. When the data is in another stage
   d. All of the above

3. Which is better: a processor with a 20-cycle branch penalty and a branch predictor that is 90% accurate or a processor with an 8-cycle branch penalty and a branch predictor that is 80% accurate?
   a. 20-cycle/90%
   b. 8-cycle/80%
   c. Same
   d. Need more information

4. A processor has a 2-bit branch predictor that can hold information on 128 different branches. A program has 129 branches. What will happen if we increase the size of the branch predictor table to hold information on 256 branches?
   a. The program will stop crashing
   b. The program will stop producing the wrong results
   c. The program will run a lot faster
   d. The program will probably run a bit faster

5. A divide-by-zero exception in the EX stage causes an interrupt that jumps to code to clean up the math. What should the processor do with the rest of the instructions in the pipeline before jumping to the interrupt?
   a. IF-finish, ID-finish, EX-finish, MEM-finish, WB-finish
   b. IF-kill, ID-kill, EX-finish, MEM-finish, WB-finish
   c. IF-kill, ID-kill, EX-kill, MEM-finish, WB-finish
   d. IF-kill, ID-kill, EX-kill, MEM-kill, WB-kill

6. Which of the following is true for both a procedure call and an interrupt?
   a. Can control where in the code it will happen
   b. Have to save and restore registers that are overwritten
   c. Can determine what code is executed
   d. Know which program will execute the code

9. Input/Output

   1. What are the tradeoffs between interrupts and polling?
      a. Interrupts use up little processor time but polling gives low latency
      b. Interrupts are good for low-latency I/O and polling for high-throughput I/O
      c. Interrupts use up a lot of processor time but have a low latency
      d. Polling uses up little processor time but has a high latency

   2. What problem does DMA solve?
      a. How to access an IO device
      b. How to move data to/from an IO device
      c. How to know when an IO device is ready
      d. All of the above

   3. What problem does memory-mapping solve?
      a. How to access an IO device
      b. How to move data to/from an IO device
      c. How to know when an IO device is ready
      d. All of the above

   4. How does Ethernet ensure reliable communication?
      a. Sending both + and – versions of the data
      b. Using a checksum to verify the data we receive
c. Reading the data as we send it to see if it is corrupted
d. All of the above

5. Why is it hard to make busses fast?
   a. Wires become slower as they get smaller
   b. Wires interfere with other wires
   c. Wires of different lengths take different times to send data
   d. All of the above

6. DRAM is faster than Flash (NVRAM) and Hard disks. So why do we not use it for permanent storage?
   a. Too expensive
   b. Slower than SRAM
   c. Doesn’t keep data without power
   d. We do use it for permanent storage

10. Caches

1. What is the problem caches try to solve?
   a. We want a large and fast memory
   b. The ISA promised a full 32-bit address space
   c. We need to do a translation at the same time as accessing data
   d. All of the above

2. Which of the following assumptions is not required for caches to be effective?
   a. The amount of data being used at any one time is small
   b. Recently used data is likely to be used again
   c. Data is accessed in regular patterns
   d. Data near recently used data is likely to be used soon

3. Which is not a benefit of making the cache line twice as large?
   a. Can store twice as much data in the cache
   b. Half as many address tags needed for the same capacity
   c. Nearby data is more likely to be in the cache
   d. Tags are smaller

4. What is a downside of making the cache line twice as large?
   a. Programs that access data sequentially will waste space
   b. Programs that access data randomly will waste space
   c. Tag comparisons are slower
   d. The percentage of bits spent on tag storage is higher

5. Approximate how long does it take to load data from DRAM on a modern computer?
   a. 1 cycle
   b. 10 cycles
   c. 100 cycles
   d. 1000 cycles

6. How many possible locations are there for storing the address 0100 0101 1110 0100 in a 64kB fully-associative write back cache with 4096 byte cache lines?
   a. 1
   b. 4
   c. 16
   d. 2048

11. Virtual Memory

1. How does virtual memory help with security?
   a. Provide each program a full 32-bit address space
   b. Prevent programs from accessing other programs’ physical memory
   c. Allow the OS to place a program’s data anywhere in physical memory
   d. All of the above
2. What would happen if two programs ran together without virtual memory and accessed the following data?

Program A:
- read-only: 0x000-0x020
- read/write: 0x100-0x200, 0x350-0x400

Program B:
- read-only: 0x000-0x080
- read/write: 0x080-0x90, 0x220-0x320, 0x800-0x950.

a. They would crash
b. They would run fine
c. They would run out of memory
d. They could not access their full 32-bit address space

3. What does it mean if we have fewer bits in the virtual page number than we do in the physical page number?

a. We have small pages
b. We have run out of physical memory
c. We have less virtual memory space than physical memory space
d. We have more virtual memory space than physical memory space

4. A program writes to VA 14 which maps to PPN (physical page number) 10 for the first time. This page is marked as COW (copy-on-write). What can you say about the next time the program accesses the same address?

a. It will be in the TLB
b. It will go to PPN 10
c. It will not go to PPN 10
d. It will cause a page protection fault

5. What is/are the biggest reasons that are page faults so excruciatingly slow?

a. The TLB has to be reloaded with the right page table data
b. The data has to be accessed from the hard disk
c. The OS has to choose which data to evict from DRAM
d. All of the above

6. If you use a much faster flash drive (NVRAM) instead of a spinning hard disk, what changes about virtual memory? Accessing the data on the drive is now so much faster that:

a. We need to optimize the TLB to be faster on hits
b. Page faults are no longer a performance problem
c. We need to optimize how quickly the OS can choose which page to evict.
d. We no longer need to evict data from DRAM if we run out of physical pages.

12. Parallelism

1. Why do we do parallelism?
   a. Faster
   b. Easier
   c. Lower power
   d. All of the above

2. How does a lock help?
   a. Prevents other processors from changing data
   b. Makes sure other processors see changes to data in my cache
   c. Tells other processors that bother to check that they should not change data
   d. Make the code parallel

3. What happened in 2003/2004 that led us to have multicore chips everywhere today?
   a. Processors got small enough that you could put multiple cores on the same chip to get better performance
   b. Processors got so hot that it was too expensive to cool them, so manufacturers switched to putting multiple slower (cooler) processors on a chip
3. c. Manufacturers couldn’t increase the clock speed, so they put multiple cores on the same chip to get better throughput
d. All of the above

4. A program takes 100s to run on a single CPU and 50% of it can be run in parallel. A multicore CPU has 4 processors. How long will it take to run on the multicore CPU?
   a. 25s
   b. 37.5s
   c. 50s
   d. 62.5s

5. A program takes 100s to run on a single CPU and 50% of it can be run in parallel. A graphics processor (GPU) has 240 processors that are each 1/4 as fast as a single CPU. How long will it take to run on the GPU?
   a. 0.83s
   b. 50.83s
   c. 200s
   d. 200.83s

6. A program takes 100s to run on a single CPU and 50% of it can be run in parallel. A multicore CPU has 4 processors. A graphics processor (GPU) has 240 processors that are each 1/4 as fast as a single CPU. How long will the program take to run if you put the serial part on the best processor and the parallel part on the best processor?
   a. 0.83s
   b. 1.66s
   c. 25.83s
   d. 50.83s

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