

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

6.004 Computation Structures
Fall 2020

Quiz #1

2	/13
3	/13
4	/15
5	/16
6	/13
7	/15
8	/15

<i>Name</i>	<i>Athena login name</i>	<i>Score</i>

Please write your name and Kerberos on your solutions.

Each subproblem is labelled with (Label: X), (e.g., Label: 1A_1). **Please make sure you write down the label of each subproblem together with its solution and circle each answer.** So, for example, if your computed value for problem 1A_1 is 5, you would write:

1A_1: 5

Problem 1. Honor Code Agreement (0 points)

We are using the honor system during this exam, and ask that you accept the following terms of this honor system:

1. You will not download a copy of the exam
2. You will not share the exam with anyone
3. You will not discuss the material on the exam with anyone until after solutions are released
4. You understand that the exam is closed book and that you may only use reference material provided with the exam

(label: 1A) I will abide by the above terms (circle one): Agree / Do Not Agree

Problem 2. Binary Arithmetic (13 points)

- (A) (3 points) What is 0x9A in decimal assuming 8-bit unsigned encoding? What is 0x9A in decimal assuming 8-bit 2's complement encoding?

(label: 2A_1) 0x9A (8-bit unsigned) in decimal: _____

(label: 2A_2) 0x9A (8-bit 2's complement) in decimal: _____

- (B) (4 points) What is -25 in 7-bit 2's complement encoding? What is -40 in 7-bit 2's complement encoding? Show how to compute -25-40 using 2's complement addition. Is it possible to represent the result in 7-bit 2's complement encoding? If so, **show your binary addition work and write the result in binary**. If not, write "Not Possible" and **explain why it's not possible**.

(label: 2B_1) -25 in 7-bit 2's complement notation (0b): _____

(label: 2B_2) -40 in 7-bit 2's complement notation (0b): _____

(label: 2B_3) -25 -40 in 7-bit 2's complement notation or "Not Possible" (show your work)

(0b): _____

- (C) (6 points) What is the value of $(0xC4 \wedge 0xFF) \ll \sim(0xFD)$ where \wedge is bitwise XOR, \ll is bitwise LEFT SHIFT and \sim is bitwise NOT. Provide your intermediate results in binary and your final result in both binary and hexadecimal. **Show your work for partial credit.**

(label: 2C_1) $(0xC4 \wedge 0xFF)$ (0b): _____

(label: 2C_2) $(\sim(0xFD))$ (0b): _____

(label: 2C_3) Final result in binary (0b): _____

(label: 2C_4) Final result in hexadecimal (0x): _____

Problem 3. RISC-V Assembly (13 points)

- (A) (3 points) What is the **hexadecimal** encoding of the instruction **sw t1, -4(t1)**? You can use the template below to help you with the encoding. Please show your work for partial credit.

[31:25]	[24:20]	[19:15]	[14:12]	[11:7]	[6:0]
imm[11:5]	rs2	rs1	funct3	imm[4:0]	opcode

(label: 3A) **sw t1, -4(t1)** instruction encoding (0x): _____

For each of the following code snippets, provide the value left in each register **after executing the entire code snippet** (i.e., when the processor reaches the instruction at the end label), or specify **CAN'T TELL** if it is impossible to tell the value of a particular register. The code snippets are independent of each other.

- (B) (3 points)

```
code_start:
    li x1, 0x26
    lui x2, 0x24
    blt x2, x1, L1
    addi x1, x1, 1
L1:
    add x1, x1, zero
end:
```

(label: 3B_1) **x1: (0x)** _____

(label: 3B_2) **x2: (0x)** _____

(label: 3B_3) **pc: (0x)** _____

(C) (4 points)

```
. = 0x100
li x4, 0x6
addi x5, zero, 0xC00
slli x4, x4, 8
or x6, x4, x5
end:
```

(label: 3C_1) x4: (0x) _____

(label: 3C_2) x5: (0x) _____

(label: 3C_3) x6: (0x) _____

(label: 3C_4) pc: (0x) _____

(D) (3 points)

```
. = 0x100
addi x7, zero, 0x204
li x8, 3
lw x9, -4(x7)
sw x8, 4(x7)
end:
```

```
. = 0x200
.word 0x01010101
.word 0xAAAAAAAA
.word 0x77777777
```

(label: 3D_1) x9: (0x) _____

(label: 3D_2) Which address in memory is written to: (0x) _____

(label: 3D_3) What value is written to memory: (0x) _____

Problem 4. RISC-V Calling Conventions (15 points)

- (A) (12 points) The box below shows the C code for a function `func` and an incorrect implementation in RISC-V assembly. While this implementation follows the logic of the corresponding C code correctly, it fails to follow the RISC-V calling convention.

Please add appropriate instructions (**either Increment/Decrement stack pointer, Load word from stack, or Save word to stack only**) into the blank spaces on the right to make `func` follow the calling convention. You may not need to use all the spaces provided. You should not modify any of the instructions already provided.

Note all values (`x` and `count`) are **signed 32-bit integers**.

`func` uses two other functions, `check` and `change`, shown to right, which follow the RISC-V calling convention. Your answer should still follow calling convention **even if the `check` and `change` functions are modified** to perform something else (that follows the calling convention).

check:

```
not a0, a0
andi a0, a0, 0x1
ret
```

change:

```
srli a0, a0, 1
ret
```

For full credit, you should **only save registers that must be saved onto the stack and avoid unnecessary loads and stores and unnecessary modifications to the stack pointer** while following the calling convention.

```
// C code
// int func(int x) {
//   int count = 0;
//   while (check(x)) {
//     count += 1;
//     x = change(x); }
//   return count; }
```

```
func:
    li t1, 0
while:
    mv s1, a0
    call check
    beqz a0, end
    addi t1, t1, 1
    mv a0, s1
    call change
    j while
end:
    mv a0, t1
    ret
```

```
func:
    li t1, 0
    (label: 4A_1)
```

	<pre> while: mv s1, a0 (label: 4A_2) _____ _____ call check (label: 4A_3) _____ _____ beqz a0, end addi t1, t1, 1 mv a0, s1 (label: 4A_4) _____ _____ call change (label: 4A_5) _____ _____ j while end: mv a0, t1 (label: 4A_6) _____ _____ _____ _____ ret </pre>
--	---

(B) (3 points) Can you make this code more efficient by changing one of the registers s1, t1, or a0 to use a different register? If so, explain which register should be changed and why. If not, explain why not. **(label: 4B)**

Problem 5. Stack Detective (16 points)

Consider the following program that computes the Fibonacci sequence recursively. The C code is shown on the left, and its translation to RISC-V assembly is provided on the right. You are told that the execution has been halted **just prior** to executing the `ret` instruction. The SP label on the stack frame (part C) shows where the stack pointer is pointing to when execution halted.

<pre>int fib(int n) { if (n <= 1) return n; return fib(n-1) + fib(n-2); }</pre>	<pre>fib: addi sp, sp, -12 sw ra, 0(sp) sw a0, 4(sp) sw s0, 8(sp) li s0, 0 li a7, 1 if: ble _____ sum: addi a0, a0, -1 call fib add s0, s0, a0 lw a0, 4(sp) addi a0, a0, -2 call fib mv t0, a0 add a0, s0, t0 done: lw ra, 0(sp) lw s0, 8(sp) L1: addi sp, sp, 12 ret</pre>
--	--

(A) (2 points) Complete the missing portion of the `ble` instruction to make the assembly implementation match the C code.

(label: 5A) : `ble` _____

(B) (2 point) How many distinct words will be allocated and pushed into the stack each time the function **fib** is called?

(label: 5B) Number of words pushed onto stack per call to **fib**: _____

(C) (3 points) Please fill in the values for the blank locations in the stack trace below. Please express the values in hex.

Smaller addresses	0x280	
	0x1	
		(label: 5C_1)
SP→		(label: 5C_2)
		(label: 5C_3)
	0x0	
	0x280	
	0x3	
	0x0	
	0x2108	
	0x4	
	0x6	
Larger addresses	0x1	

(D) (3 points) What is the hex address of the done label?

(label: 5D) Address of done label: (0x) _____

(E) (3 points) What was the argument of the original function call to `fib`?

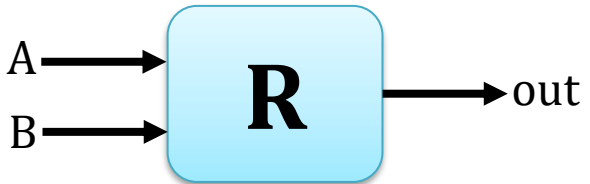
(label: 5E) Argument of the original function call to `fib`: _____

(F) (3 points) What was the address of the original function call to `fib`?

(label: 5F) Address of original function call to `fib`: (0x) _____

Problem 6. Static Discipline (13 points)

The R module below outputs 0.5V when $\min(V_A, 0.5V_B) > 2V$ for 25ns and outputs 6V when $\min(V_A, 0.5V_B) < 1.5V$ for 25ns. This is summarized in the equation below (assume all voltages are positive). Also, assume this circuit obeys a digital signaling specification where low voltages correspond to digital 0 and high voltages correspond to digital 1.

$$V_{out} = \begin{cases} 0.5V, & \min(V_A, 0.5 * V_B) > 2V \\ 6V, & \min(V_A, 0.5 * V_B) < 1.5V \\ 0 \leq ??? \leq 6V, & \text{otherwise} \end{cases}$$


(A) (2 points) If we apply constant V_A, V_B for 25ns and then measure $V_{out} = 0.5V$, what can we conclude about V_B ?

C1: $V_B < 3V$

C2: $V_B \leq 4V$

C3: $V_B > 4V$

C4: $V_B \geq 3V$

C5: None of the above

(label: 6A) Best conclusion about V_B (Select one): C1 ... C2 ... C3 ... C4 ... C5

(B) (3 points). What Boolean expression does R implement? Specify an equation using A and B.

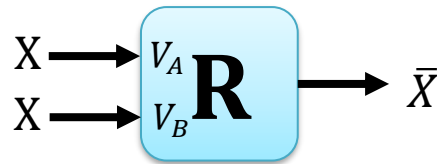
(label: 6B) Boolean Expression: $\text{out}(A, B) = \underline{\hspace{2cm}}$

(C) (5 points) Find a signaling specification that maximizes noise immunity for the R module.

(label: 6C_1) $V_{OL} = \underline{\hspace{1cm}}, V_{IL} = \underline{\hspace{1cm}}, V_{IH} = \underline{\hspace{1cm}}, V_{OH} = \underline{\hspace{1cm}}$

(label: 6C_2) Noise Immunity = $\underline{\hspace{2cm}}$

(D) (3 points) Suppose one wishes to using the R module as an inverter, as shown below. What is the noise immunity of this device?



(label: 6D) Noise Immunity: $\underline{\hspace{2cm}}$

Problem 7. Boolean Algebra (15 points)

Given the following truth table of function h , which is a function of two inputs, x and y :

x	y	h
0	0	1
0	1	1
1	0	0
1	1	1

(A) (2 points) Find the minimal sum of products expression for h .

(label: 7A) Minimal sum of products expression for $h(x, y)$: _____

(B) (4 points) Identify whether $h(x, y)$ is universal, i.e., can be used to implement any Boolean function. If so, **show how to implement AND, OR, and NOT from it; if not, explain why not.**

(label: 7B)

(C) (9 points) Simplify the following Boolean expressions by finding a **minimal sum-of-products expression** for each one. (Note: These expressions can be reduced into a minimal SOP by repeatedly applying the Boolean algebra properties we saw in lecture.)

1. $(a + c)(ad + a\bar{d}) + ac + c =$

(label: 7C_1) _____

2. $a + ab + \overline{ab} =$

(label: 7C_2) _____

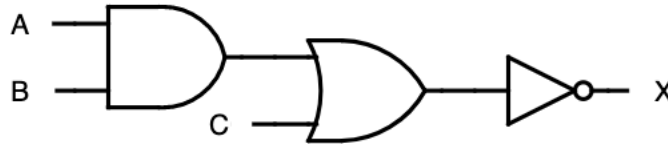
3. $\bar{c}(\overline{abcdef})$

(label: 7C_3) _____

Problem 8. CMOS (15 points)

For each of the following combinational logic functions, write a Boolean expression for the function. Then, determine if the function can be implemented as a single CMOS gate. If so, draw the corresponding CMOS gate. If not, explain why not.

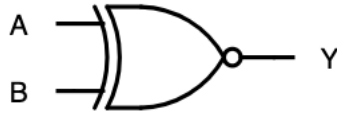
- (A) (5 points) The AOI (AND-OR-Invert) logic function, so called because it consists of an AND gate, followed by an OR gate, followed by a NOT gate.



(label: 8A_1) Boolean expression for $X(A, B, C)$: _____

(label: 8A_2) Draw single CMOS gate implementation of $X(A, B, C)$ or explain why one does not exist.

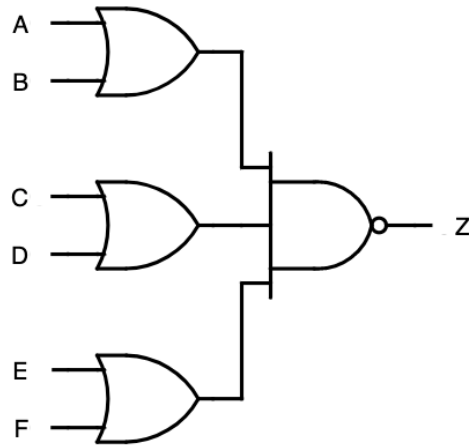
(B) (5 points) The XNOR logic gate.



(label: 8B_1) Boolean expression for $Y(A, B)$: _____

(label: 8B_2) Draw single CMOS gate implementation of $Y(A, B)$ or explain why one does not exist.

(C) (5 points) The following 6-input gate (called an OAI222 gate, as it consists of 3 2-input OR gates, followed by a 3-input AND gate, and an Inverter).



(label: 8C_1) Boolean expression for $Z(A, B, C, D, E, F)$: _____

(label: 8C_2) Draw single CMOS gate implementation of $Z(A, B, C, D, E, F)$ or explain why one does not exist.

END OF QUIZ 1!