CSSE 232 – Computer Architecture I
Rose-Hulman Institute of Technology
Computer Science and Software Engineering Department

Quiz 3 - 20 minutes

Name: ______________________________ Section: 1 2 3 4

This quiz is **closed book**. You are allowed to use the reference card from the book (attached at the back of this quiz) and one 8.5" × 11" single sided page of hand written notes. You may not use a computer, phone, etc. during the examination.

Write all answers on these pages. Be sure to **show all work** and document your code. Do not use instructions that we have not covered (e.g. no `mul` or `div` but you can use instructions like `slli`, `srl`, `xori`, etc).

RISC-V code is judged both by its correctness and its efficiency. Unless otherwise stated, you may not use RISC-V pseudoinstructions when writing RISC-V code.

Question	Points	Score
Problem 1	10	
Total:	10	

Problem 1. You are tasked with adding some new instructions to the Single Cycle RISC-V datapath we've created in class. For each of the following instructions, write the RTL that would implement the instruction described.
RTL from class is provided on the next page for your reference.

(a) (5 points) `kitcat` (reverse half concatenate):
This R-format instruction reverses the order of the first and second halves of a register (rs1) and puts the result into a destination register. For example, assume register `x10` initially contains `0x1234ABCD`. After `kitkat x11, x10` executes, `x11` would contain `0xABCD1234`. Write the RTL below:

(b) (5 points) `popcrn` (Pop if Check Register Nonzero):
This I-format instruction pops a value off the stack into a destination register rd **only if** the operand register (rs1) is *not equal to zero*.
For example, assume `x10` initially contains the value `0x0232`. After executing "`popcrn x11, x10`", register `x11` will contain whatever was on top of the stack, and the stack pointer will have moved to deallocate that top element.
(HINT: recall the stack pointer is register number 2)
Write the RTL below:

Single-Cycle RISC-V RTL

add
newPC = PC+4
PC = newPC
inst = Mem[PC]
a = Reg[inst[19:15]]
b = Reg[inst[24:20]]
result = a + b
Reg[inst[11:7]] = result

addi
newPC = PC+4
PC = newPC
inst = Mem[PC]
a = Reg[inst[19:15]]
imm = SE(inst[31:20])
result = a + imm
Reg[inst[11:7]] = result

lw
newPC = PC+4
PC = newPC
inst = Mem[PC]
a = Reg[inst[19:15]]
imm = SE(inst[31:20])
result = a + imm
memOut = Mem[result]
Reg[inst[11:7]] = memOut

sw
newPC = PC+4
PC = newPC
inst = Mem[PC]
a = Reg[inst[19:15]]
b = Reg[inst[24:20]]
imm = SE({inst[31:25], inst[11:7]})
result = a + imm
Mem[result] = b

beq
newPC = PC+4
inst = Mem[PC]
a = Reg[inst[19:15]]
b = Reg[inst[24:20]]
imm = SE({inst[31], inst[7], inst[30:25], inst[11:8]})<<1
target = PC + imm
if(a == b): PC = target
else: PC = newPC