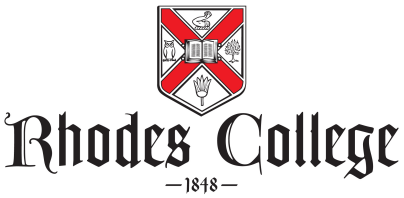
COMP 231-01
Introduction to Computer Organization
Exam II Review



- Answer the following questions. For the assembly language questions, assume that all variables are signed integers. All fragments are to be written as a NIOS II assembly language fragment.

1. Write a NIOS II assembly language fragment that reads a 32 bit value from memory location 0x0040, doubles it, and stores the result in r13.

2. Translate the following into assembly, assuming x is stored in register r11 and y is stored at memory location 0x0084:

```
y = 5;
y++;
x = y + 12;
```

3. Translate the following into assembly, assuming the register-like variables are stored in a register.

   r8 = r12 + 20;

4. Translate the following into assembly, assuming x is stored in register r11 and y is stored at memory location 0x0084:

```
if(x != y)
  x=50;
else
  y=25;
```

5. Translate the following into assembly, assuming x is stored in register r11, y is stored at memory location 0x0084, and z is stored at 0x0200:

```
while(y <= 10 && x > 10)
  y++;
  x--;
```

6. Translate the following into assembly :

```
r8 = 10;
r9 = 0;

while(r8 > 10)
  r9 = r9 + r8;
```

7. Translate the following into assembly:

```
for(int i = 0; i < 6; i++)
  r11 = r11 + i;
```

8. Translate the following into assembly:

```
if(r8 == 0)
  r8 = r12 + 9;
else if(r8 < 100 || r12 > 100)
  r12 = r8;
else
  r8 = 10;
  r12 = 10;
```

9. Assume we have an array, a[], of 20 integers starting at memory address x1000. Write a program that stores each power of 2 (starting at 1) in the first 8 indices of this array. Your program should calculate the power of 2 and then store that value in the index. You are free to use your choice of registers, memory space, and labels as needed for your program.

   a[0] will become 2
   a[1] will become 4
   a[2] will become 8
   etc.

10. Assume we have an array, a[], of 20 integers starting at memory address x1000. Write a program that compares each index in that array to a variable initialized to 100. If the value in the index is greater than or equal to the variable, add 10 to the variable. If the value at the index is less than the variable, subtract 10 from it. You are free to use your choice of registers, memory space, and labels as needed for your program.

    If a[0] is 12, the value changes to 90
    If a[1] is 120, the value changes to 100
    If a[2] is 200, the value changes to 110
    etc.

11. How many bits does a single memory address store?

12. What is the size of a word in NIOS II assembly language?

13. What does the 0x symbolize in the memory address 0x5433?

14. What is the difference between the stw and stwio commands?

15. What is sign extension? Why would you use sign extension?

16. What is a pseudoinstruction? In what situation would a pseudoinstruction be used?

17. How many regular registers does NIOS II have?

18. If you were to include a value directly in an assembly command (that is, not from a register or memory location), what addressing mode would you be using?

19. Describe what the following absolute address describes: $20(r12)$

20. Describe what the following absolute address describes: $0(r10)$

21. Describe what the following absolute address describes: $11(r9)$

22. What is the content of Register 0?

23. What happens if you change the value of the program counter?

24. Why can the program counter not be set to an odd number without an error?

25. If you were to look inside the contents of a command in binary, what are three components that you would see?

NIOS II Instruction Reference

This is a list of the syntax for the instructions that you may use from the NIOS instruction set:

instruction	usage
ldw	ldw rB, off(rA)
ldb	ldb rB, off(rA)
ldbu	ldbu rB, off(rA)
ldbu	ldbu rB, off(rA)
ldh	ldh rB, off(rA)
ldhu	ldhu rB, off(rA)
stw	stw rB, off(rA)
stb	stw rB, off(rA)
sth	stw rB, off(rA)
add	add rC, rA, rB
sub	sub rC, rA, rB
mul	mul rC, rA, rB
div	div rC, rA, rB
addi	addi rC, rA, IMMED16
subi	subi rC, rA, IMMED16
muli	muli rC, rA, IMMED16
divu	divu rC, rA, rB
and	and rC, rA, rB
or	or rC, rA, rB
xor	xor rC, rA, rB
andi	andi rC, rA, IMMED16
ori	ori rC, rA, IMMED16
xori	xori rC, rA, IMMED16
andhi	andhi rC, rA, IMMED16
orhi	orhi rC, rA, IMMED16
xorhi	xorhi rC, rA, IMMED16
mov	mov rC, rA
movi	movi rB, IMMED16
movui	movui rB, IMMED16
srl	srl rC, rA, rB
srli	srli rC, rA, IMMED5
sra	sra rC, rA, rB
srai	srai rC, rA, IMMED5
sll	sll rC, rA, rB
slli	slli rC, rA, IMMED5
jmp	jmp rA
br	br LABEL
beq	beq rA, rB, LABEL
bne	bne rA, rB, LABEL
blt	blt rA, rB, LABEL
ble	ble rA, rB, LABEL
bgt	bgt rA, rB, LABEL
bge	bge rA, rB, LABEL
bltu	bltu rA, rB, LABEL
bleu	bleu rA, rB, LABEL
bgtu	bgtu rA, rB, LABEL
bgeu	bgeu rA, rB, LABEL
call	call LABEL
ret	ret