

EE 2310 Test Review #3 – Assembly Language and Computer Architecture

1. The loop is doing a computation.
 - a. What is it doing?
 - b. How many times through the loop will it go?
 - c. What will be the decimal value of the final number in \$t2?

.text
main: la \$a0,str
comp: lb \$t0,0(\$a0)
beqz \$t0,done
ble \$t0,0x60,ret
bge \$t0,0x7b,ret
addi \$t2,\$t2,1
ret: addi \$a0,\$a0,1
j comp
done: li \$v0,10
syscall
.data
str: .asciiz "hello world\"

2. Write a program in the space at the right that calculates the function x^y . Locations “x” and “y” hold operands x, y. The result of the calculation is stored in location z when completed. You need only write the text section, as the data declaration is already done. You must use a loop to do the calculation.

- a. How many cycles will the loop run?
- b. What is the decimal answer?
- c. What does the “.space” directive do?

.text
main: li \$t0,1
lw \$t4,x
lw \$t5,y
loop: mul \$t0,\$t0,\$t4
sub \$t5,\$t5,1
beqz \$t5,store
j loop
store: sw \$t0,z
li \$v0,10
syscall
.data
x: .word 100
y: .word 4
z: .space 4

3. The program steps to the right represent a portion of a program. The data declaration is not shown.

- a. What is it doing?
- b. How many times will the loop repeat?
- c. What is the value in \$t2 called?
- d. What is the possible range of final values in \$s5 (state in decimal)?

li \$t2,15
li \$t3,8
lw \$t1,data2
rotate: rol \$t1,\$t1,4
and \$t4,\$t1,\$t2
bnez \$t4,incr
addi \$s5,\$s5,1
incr: addi \$t3,\$t3,-1
bnez \$t3,rotate
over: li \$v0,10
syscall

4. The code sequence on the right is a procedure that performs an analysis. Two instructions are wrong. First correct them, then answer the following questions:

- The important result is when the contents of \$t5 are printed. That being the case, what is the program doing?

- What is the resulting number in \$t5?

- What was wrong with the program?

Note ASCII Codes:

0x 61 = "a"

0x 65 = "e"

0x 69 = "i"

0x 6f = "o"

0x 75 = "u"

0x 7a = "z"

```
.text
main: lw $a0,str

      jal look
      move $a0,$t5
      li $v0,1
      syscall

      li $v0,10
      syscall

look: lb $t0,($a0)
      bnez $t0,done
      blt $t0,0x61,adup
      bgt $t0,0x7a,adup
      addi $t4,$t4,1
      beq $t0,0x61,count
      beq $t0,0x65,count
      beq $t0,0x69,count
      beq $t0,0x6f,count
      beq $t0,0x75,count

adup: addi $a0,$a0,1
      j look
count: addi $t2,$t2,1
      j adup
done: sub $t5,$t4,$t2
      jr $ra

.data
str: .asciiz "hello world\n"
```

5. Program Analysis:

- (20 points) In the early 13th century, the mathematician Fibonacci developed a mathematical formula for a number series that predicted the reproduction rate of rabbits. This series turned out to be an important mathematical development. The program at the right embodies the series formula to calculate $F(n)$ using a recursive routine, where $F(n)$ is the Fibonacci number for any integer n . Note that $F(0) = 0$ by definition, and likewise, $F(1) = 1$. The Fibonacci formula calculates the so-called “Fibonacci number” for any integer n such that $n \geq 2$. Answer the following questions about the program:
- Explain in words exactly how the program uses recursion to facilitate the $F(n)$ calculation.
- After studying the $F(n)$ calculation loop, what is a general formula for $F(n)$? Hint: The loops labeled “fib” and “fret” are the key loops.
- Based on the formula you derived above, what are $F(5)$ and $F(10)$?

```
.text
main: li $v0,4
      la $a0,inpt
      syscall
      li $v0,5
      syscall
      sw $v0,n
      move $t1,$v0
      jal fib

      la $a0, ans1
      li $v0,4
      syscall
      lw $a0,n
      li $v0,1
      syscall
      la $a0, ans2
      li $v0,4
      syscall
      move $a0, $s1
      li $v0,1
      syscall

      la $a0, endl
      li $v0,4
      syscall
      li $v0,10
      syscall

fib:   sub $sp,$sp,8
      sw $t1,0($sp)
      sw $ra,4($sp)

      bgt $t1,2,not2
      li $s3,0 # Fib(0) = 0
      li $s2,1 # Fib(1) = 1
      j fret

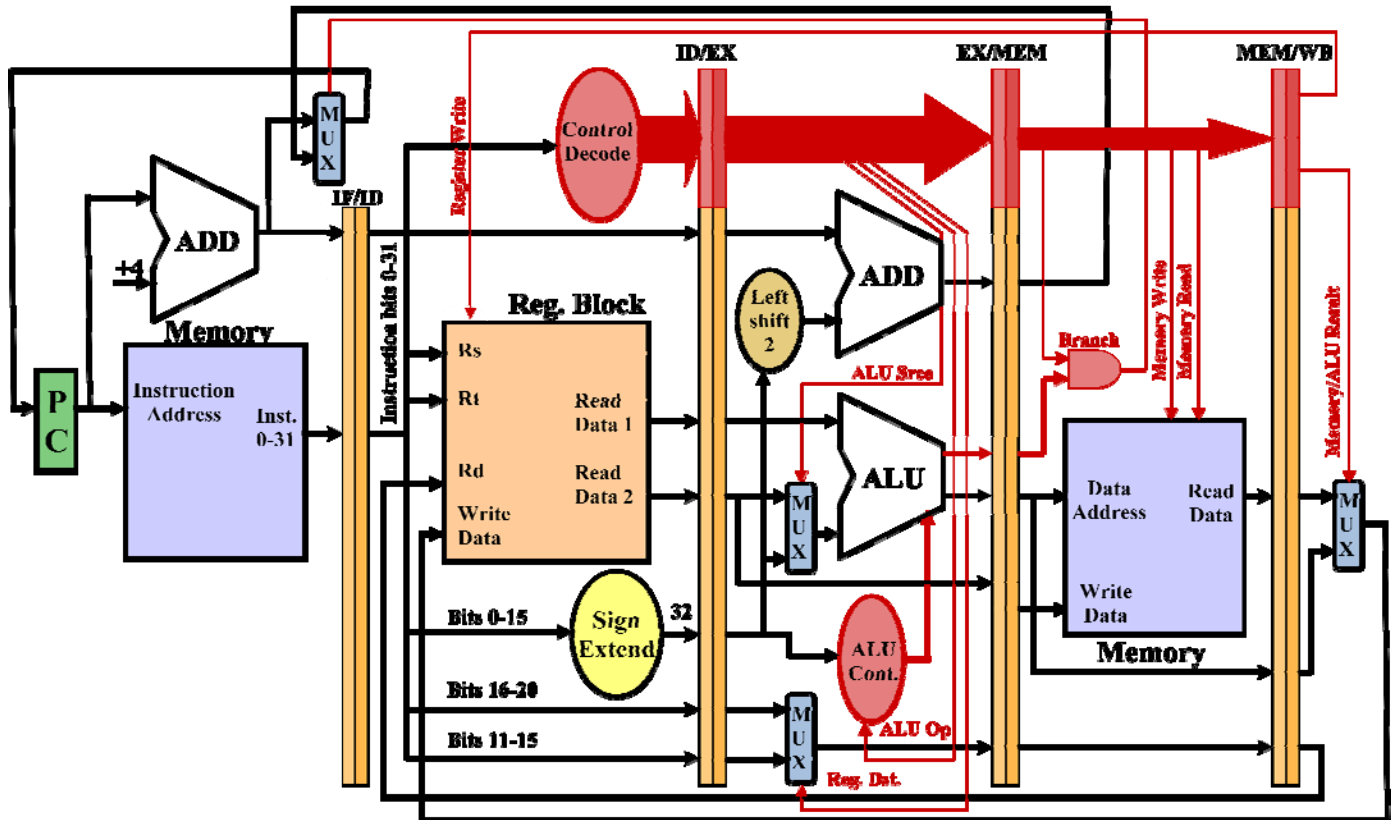
not2: sub $t1,$t1,1
      jal fib

fret: add $s1,$s2,$s3
      move $s3,$s2
      move $s2,$s1
      lw $t1,0($sp)
      lw $ra,4($sp)
      add $sp,$sp,8
      jr $ra

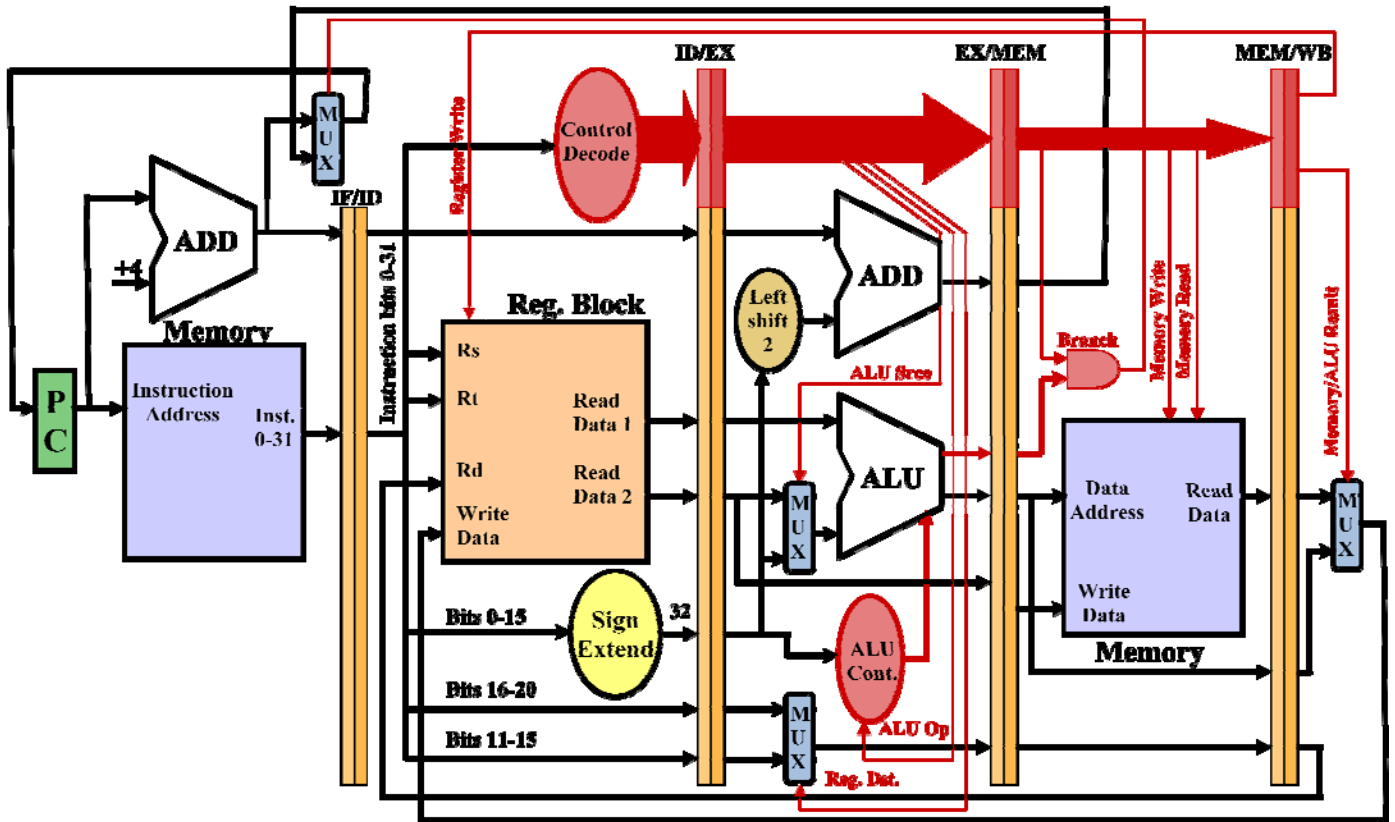
.data
inpt: .ascii "Input 2-digit number,
      40 or less, for Fibonacci
      calculation: "

n:    .word 0
ans1: .ascii "F ("
ans2: .ascii ") is "
endl: .ascii ".\n"
```

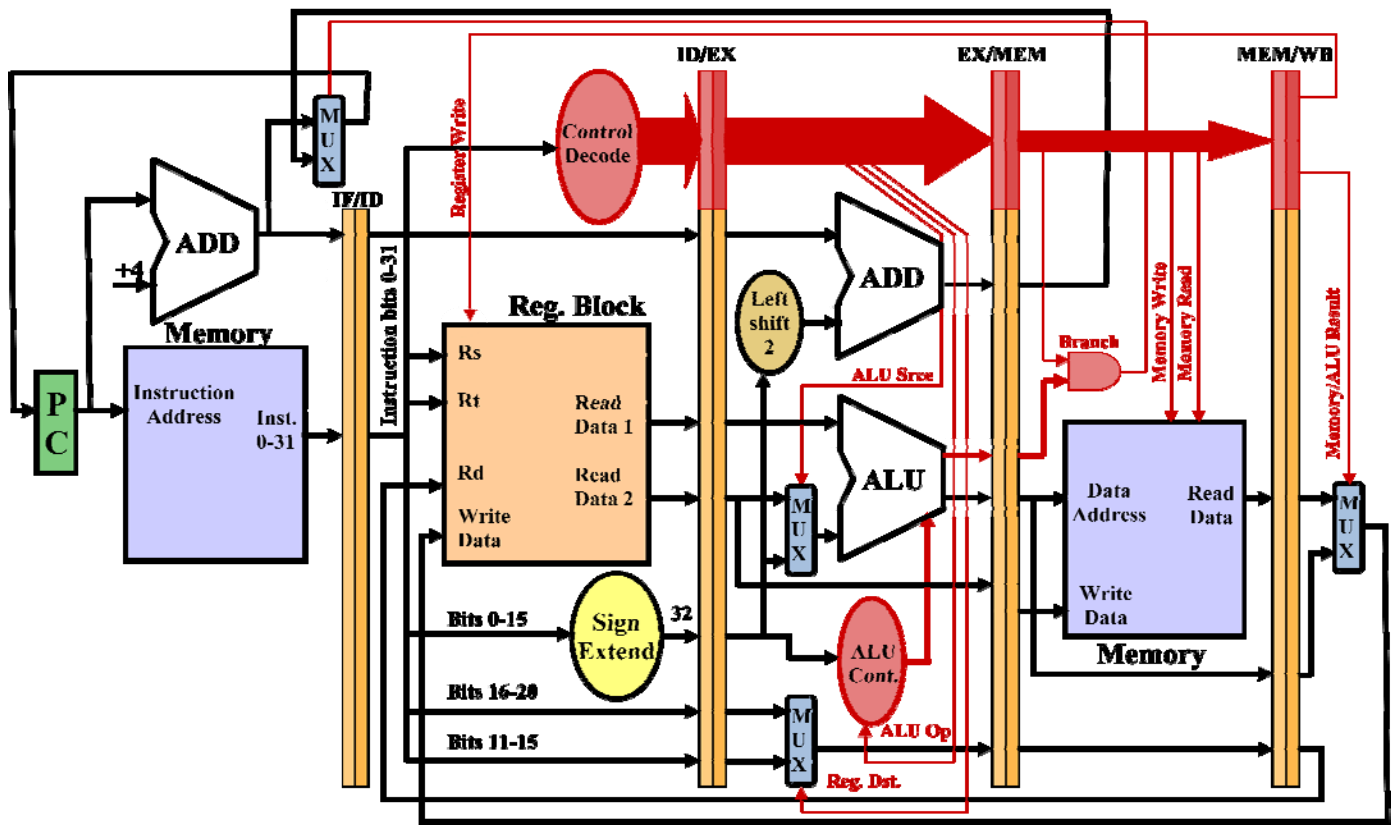
Computer Architecture Review



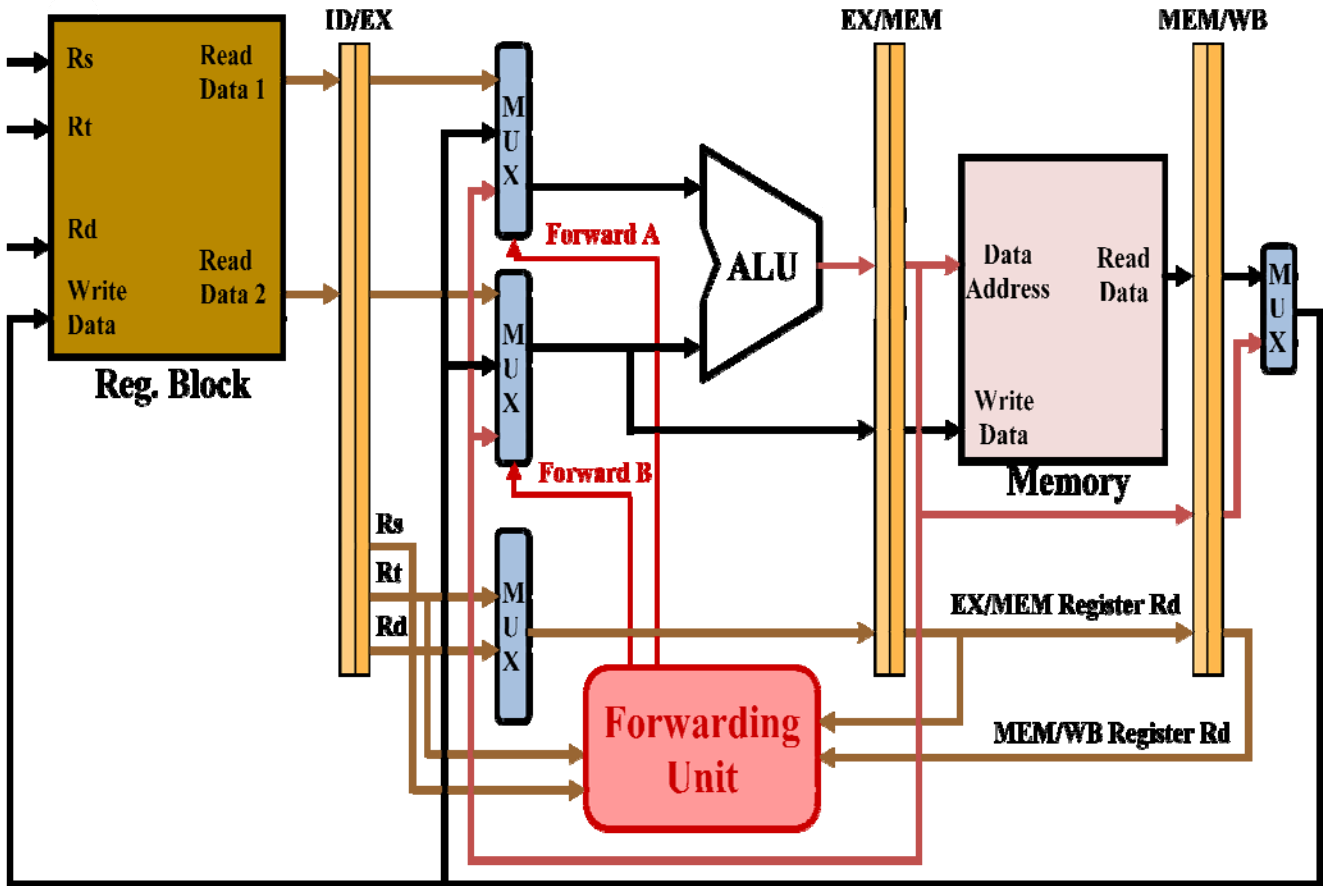
6. Show the data flow of `lw $t7,8($t3)` through the pipeline. Ignore control lines; just show the data flow. Contents of a register should be shown with square brackets (e.g., contents of `$t2 = [$t2]`).



7. Show how `blt $s0,$s3,loop2`, flows through the pipeline, if `loop2` is 10 instructions farther up in memory from the next instruction, and the branch is taken. Ignore controls except for the branch bit flow.



8. Show the pipeline progress of the instruction sub \$t0,\$t1,\$t2. If [$t1$]= 0x 2b and [$t2$] = 0x a, what hex number is on the ALU bypass bus.



9. In the forwarding unit above, if Read Data 1 is from \$s4 and the data on the ALU bypass bus is from the instruction or \$s0,\$s1,\$s4, show the data flow that results in the correct register data arriving at the upper ALU data input.