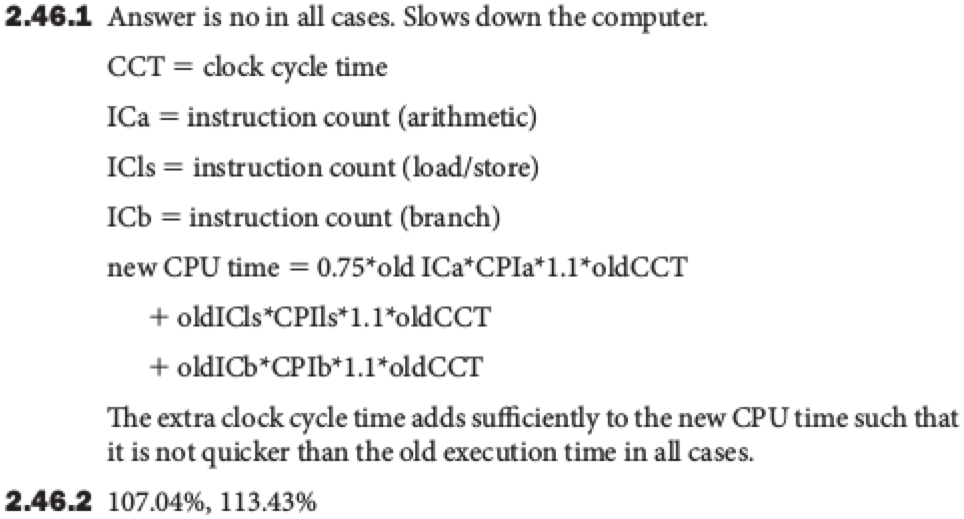
Department of Computer Science **University of San Francisco**

Computer Science 315
Spring 2016
Computer Architecture
Homework #5 Solutions

1)



2.46.1 Answer is no in all cases. Slows down the computer.

CCT = clock cycle time

ICa = instruction count (arithmetic)

ICls = instruction count (load/store)

ICb = instruction count (branch)

new CPU time = 0.75*old ICa*CPIa*1.1*oldCCT

+ oldICls*CPIls*1.1*oldCCT

+ oldICb*CPIb*1.1*oldCCT

The extra clock cycle time adds sufficiently to the new CPU time such that it is not quicker than the old execution time in all cases.

2.46.2 107.04%, 113.43%

2)

```
# Function:     Fibo
# Purpose:      Return the nth Fibonacci number, where
#
#               Fibo(0) = 0
#               Fibo(1) = 1
#               Fibo(n) = Fibo(n-1) + Fibo(n-2), n >= 2
#
# C Prototype:  long Fibo (long n)
# Args:         n = rdi
# Return val:   Fibo(n) = rax
#
```

```
        .section .text
        .global Fibo

Fibo:
        push %rbp
        mov  %rsp, %rbp
        sub  $16, %rsp          # We may need to store n and a return
                        #    val from a recursive call

        # Is n = 0?
        cmp  $0, %rdi           # Is n = rdi == 0?  Note that the immediate
                        #    must come first here
        jne  n_gt_0            # Look at the flags register to see whether
                        #    the previous comparison result is != 0
        mov  $0, %rax           # Return 0
        jmp  done              # Go to done

n_gt_0:
        # Is n = 1?
        cmp  $1, %rdi           # Is n = rdi == 1?
        jne  n_gt_1            # Look at the flags register to see whether
                        #    the previous comparison result is != 1
        mov  $1, %rax           # Return 1
        jmp  done              # Go to done

n_gt_1:
        # n >= 2
        mov  %rdi, 8(%rsp)      # Save n = rdi on the stack
        sub  $1, %rdi           # n = n-1
        call Fibo
        mov  %rax, 0(%rsp)      # Save Fibo(n-1) on the stack
        mov  8(%rsp), %rdi      # Retrieve n
        sub  $2, %rdi           # n = n-2
        call Fibo
        add  0(%rsp), %rax      # return Fibo(n-1) + Fibo(n-2)

done:
        leave                  # Assigns rbp to rsp:  no need to
                        #    add 16 to rsp
        ret
```