16.317: Microprocessor Systems Design I

Summer 2013

Homework 4

Due **Tuesday, 7/30/13—NO LATE ASSIGNMENTS**

Notes:

- While typed solutions are preferred, handwritten solutions to these problems are acceptable.
- Any handwritten solutions that are scanned and submitted electronically must be clearly legible and combined into a single file—simply sending a picture of each scanned page is not an acceptable form of submission.
- Because your second exam is on Thursday, 8/1, I plan to post the solution to this assignment early on Wednesday, 7/31. Therefore, **no late submissions will be accepted for this assignment.**
- This assignment is worth a total of 50 points.

1. *(15 points)* The program that follows implements what is known as a *delay loop.*

```
     MOV  CX, 1000H
DLY: DEC  CX
     NOP             ; NOP instruction does nothing
     JNZ  DLY
NXT: ---  ---
```

a. How many times does the JNZ DLY instruction get executed?
b. Change the program so that JNZ DLY is executed just 17 times.
c. Change the program so that JNZ DLY is executed 2^{32} times.

2. *(15 points)* Write a program that compares the elements of two arrays, A(I) and B(I). Each array contains 100 16-bit signed numbers. The comparison is to be done by comparing the corresponding elements of the two arrays until either two elements are found to be unequal or all elements of the arrays have been compared and found to be equal. Assume that the arrays start in the current data segment at offset addresses $A000_{16}$ and $B000_{16}$, respectively. If the two arrays are found to be unequal, save the offset of the first unequal element of A(I) in the memory location with offset $C000_{16}$; otherwise, write all 0s into this location.

3. *(20 points)* Write the following subroutine in x86 assembly:

```
int fib(int n)
```

Given a single integer argument, `n`, return the `n`th value of the Fibonacci sequence—a sequence in which each value is the sum of the previous two values. The first 15 values are shown below—note that the first value is returned if n is 0, not 1.

n	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
fib(n)	0	1	1	2	3	5	8	13	21	34	55	89	144	233	377

Recall that:

- Subroutine arguments are passed on the stack, and can be accessed within the body of the subroutine starting at address EBP+8.
- At the start of each subroutine:
  i. Save EBP on the stack
  ii. Copy the current value of the stack pointer (ESP) to EBP
  iii. Create space within the stack for each local variable by subtracting the appropriate value from ESP. For example, if your function uses four integer local variables, each of which contains four bytes, subtract 16 from ESP.
  iv. Local variables can then be accessed starting at the address EBP-4.
- A subroutine's return value is typically stored in EAX.

See Lecture 6 for more details on subroutines, the x86 architecture, and the conversion from high-level concepts to low-level assembly.