16.317: Microprocessor Systems Design I

Spring 2015

Homework 3

Due **Wednesday, 3/11/15**

Notes:

- While typed solutions are preferred, handwritten solutions are acceptable
- All solutions must be legible and contained in one file. Archive files are not acceptable.
- Electronic submissions should be e-mailed to Dr. Geiger at Michael_Geiger@uml.edu.
- This assignment is worth 100 points.

Each of the questions on the next two pages asks you to convert a high-level code sequence to x86 assembly. Please note that each of these code sequences is part of a larger function—do not worry about any of the function call and stack frame details discussed during lectures 14-16. (That material will be covered on HW 4.)

1. (25 points) Implement the following conditional statement. You may assume that "X" and "Y" refer to 16-bit variables stored in memory, which can be directly accessed using those names (for example, `MOV AX, X` would move the contents of variable "X" to `AX`).

```
if (AX < 10) {
   CX = X + 10;
}
else if (AX == 20) {
   CX = CX - Y;
}
else {
   CX = X + Y;
}
```

2. (25 points) Implement the following loop. As in question 1, assume "X" is a 16-bit variable in memory that can be accessed by name. (Hint: Any loop that executes the correct number of iterations is acceptable—you do not necessarily have to change your loop counter in exactly the same way as the for loop, since `i` is not used in the body of the loop.)

```
for (i = 0; i < X; i++) {
   AX = AX + X;
   BX = BX - X;
   if (AX == BX)
        break;          // Exit loop early
}
```

3. (25 points) Implement the following conditional statement. As in question 1, assume "X" and "Y" are 16-bit variables in memory that can be accessed by name. (Note: Make sure you carefully count the parentheses to make sure you combine conditions correctly!)

```
if (((AX < X) && (BX < Y)) || ((AX > Y) && (BX > X))) {
   AX = AX - BX;
}
```

4. (25 points) Implement the following loop. As in previous questions, assume "X", "Y", and "Z" are 16-bit variables in memory that can be accessed by name. Recall that a while loop is a more general type of loop than the for loop seen in question 2—a while loop simply repeats the loop body as long as the condition tested at the beginning of the loop is true.

```
while (X >= Y) {
   Y = Y + Z - 1;
   X = X - Z + 1;
}
```