Name: _____

Directions: Work only on this sheet (on both sides, if needed); do not turn in any supplementary sheets of paper. There is actually plenty of room for your answers, as long as you organize yourself BEFORE starting writing. In order to get full credit, SHOW YOUR WORK.

1. (5) Show the hex notation for the string 101111010110.

2. (5) Suppose c(EBX) = 0x12aa34bb. What is c(BH)?

3. This problem is about the JS instruction.

(a) (5) The piece of hardware which this instruction checks is the _____.

(b) (5) In part (a), why did it not explicitly add the specification "on our Intel CPUs"?

4.: (10) On a 32-bit machine, a **short int** might be implemented in 16 bits. This is a decision made by the developer of the _____, who in turn might be more likely to do this if the developer of the _____ has provided for 16-bit operations.

5. (5) Suppose in C code there is a **char** array **z**, and a statement

```
strcat(z,"uvw");
```

Which one is correct?

(i) If the machine is little-endian, then the 'u' will have a smaller address than the 'w'.

(ii) If the machine is little-endian, then the 'u' will have a greater address than the 'w'.

(iii) Regardless of endian-ness of the machine, the 'u' will have a smaller address than the 'w'.

(iv) Regardless of endian-ness of the machine, the 'u' will have a greater address than the 'w'.

6. (10) Next to each of the following instructions, write "Yes" if that instruction would cause the assembler to temporarily substitute an offset for an address, and thus need to "leave a note" in the **.o** file. Otherwise write "No." Here, **z**, **y** and **w** are labels.

```
jmp z
call y
movl $w, %eax
addl $8, %eax
```

7. Consider the program in Section 8 of the PLN unit on Linux Intel assembly language.

(a) (15) Look at the three instructions (**decl**, **jz** and **addl**) which start at **nextj**. Show the output of **as -a** for those three instructions (minus line numbers and offsets). You'll need to know that the register code for EBP is 101. Also, assume for the sake of this problem that the format for the **addl** instruction is 11111DDDIMM4.

(b) (10) Suppose the **.data** segment starts at 0x1248. As we enter **swap()** for the first time, what values will be in EAX and ESI?

8. (30) A program contains the code excerpt shown below, which finds the first instance of the bit substring 111 in ECX. (That substring is guaranteed to be in ECX somewhere.) The answer, reported in EDX, will be the position of the first 1 in 111, and will range from 31 (meaning at the most significant bit position) to 0 (meaning the least significant bit position). For example, if c(ECX) = 0x70f02222 (that 7 is the most significant hex digit), then the final value in EDX will be 30. If c(ECX) = 0x11f02222, then EDX will turn out to be 24. Fill in the blanks.

```
    movl $31, %edx
top:
    movl %ecx, %ebx
```

```
    andl ____________________, %ebx
    cmpl $0xe0000000, %ebx
    __________________ found111
    decl ______________________
    shll _________________, %ecx
    jmp top
found111: # (no blank here)
```

Solutions:

1. 0xbd6

2. 0x34

3.a. EFLAGS (or Sign Flag)

3.b. Each CPU has a different instruction set. JS is an Intel instruction, so we don't have to specify Intel.

4. compiler, CPU

5. (iii)

6. No; Yes (No acceptable though actually wrong); Yes; No

7.a.

```
4D
7407
FA04000000
```

7.b. The first swap occurs when "x[i]" is the 5, and the "x[j]" is the 2, at 0x124c and 0x1250, respectively. So, EAX and ESI contain those values.

8.

```
    # overall plan:
    #    while (1)
    #       check for 111 on left end
    #       if found break
    #       shift left 1 bit}
    # EDX will be the original bit position of the 3-bit set we
    #    are currently checking
    movl $31, %edx
top:
    movl %ecx, %ebx
    # check for 111 on left end
    andl $0xe0000000, %ebx
    cmpl $0xe0000000, %ebx
    # if found break
    jz found111
    decl %edx
    shll $1, %ecx
    jmp top
found111: # (no blank here)
```