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. Earlier problems tend to be easier.

1. (5) Using 1s and 0s, show the representation of -3 as a 5-bit 2s complement number.

2. (5) Fill in the blank: A one-word memory unit which is part of the CPU and which programmers use for faster access than external memory (RAM) is called a ________________.

3. (5) In class, we discussed the fact that when you ask the operating system to shut down a computer, the OS delays for a few seconds before the shutdown. What is the purpose of this which we discussed in class? Be very specific in your answer.

4. (5) The author of the %d portion of printf() does not have to worry about endian-ness because the _________________ takes this into account.

5. (10) Consider a file abc.s, with the following contents:

```
.text
movl $8,%eax
addl (%eax),%edx
```

Suppose we run this file through the command "as -a". Using hex notation, state what machine language the assembler will produce, and state at what offsets in the text segment the assembler would place this machine language.

6. Answer the following questions about caches:

(a) (5) True or false: In an associative cache which is currently not full, a cache miss will definitely not result in an eviction of some block currently in the cache.

(b) (5) True or false: In a direct-mapped cache which is currently not full, a cache miss will definitely not result in an eviction of some block currently in the cache.

(c) (10) Fill in the blanks: Suppose a cache hit occurs. Then the MAR and MDR will not be used for this memory access, except in the case in which the access is a(n) ______________ and the cache uses a(n) _______________ policy.

7. Suppose we write the following C code, to be run on an Intel machine with 16-bit word size:

```
int u,v,w,i,*p;  // locals
...
i  = &w;
i += 2;
p = (int *) i;
```

(a) (5) After the code is executed, to which variable will p be pointing?

(b) (5) If this code were to be changed to run on a 32-bit Intel machine, what specific number should the 2 be changed to?

(c) (10) If the 16-bit version of the code were to be run without change on a 32-bit Intel machine, would the program run correctly, or for that matter, run at all? Answer in detail.

(d) (5) In order to ensure that the same C source code works on both word sizes, what should the 2 in the 16-bit version be changed to?

8. (5) Suppose that the EAX register currently contains the value 0x204, and that word 0x204 contains 0x206. Ignoring instruction fetches and cache effects, state the sequence of values, if any, which go through MAR and MDR during the execution of the instruction **decl (%ecx)**.

9. (10) Write a C function with the following declaration:

```
int ExtractByte(int X, int K)
```

The function returns the K-th least-significant byte from X. So for example ExtractByte(259,1) would return 3, ExtractByte(259,2) would return 1, and so on, (Note: $259_{10} = 103_{16}$, so that the least-significant byte is 0x03, i.e. 3 and the second least-significant byte is 1.) In other words, the larger the value of K, the more significant the byte returned. Make your function fully general, runnable on any machine. You are allowed to call the function Endian() from the statement of Problem I, Homework 2 (you need not write the function on your paper here; just call it). Recall that that function returns 1 on a little-endian machine, 0 otherwise.

10. (10) A **disassembler** takes an object file (.o) and changes it back to a corresponding assembly language file (.s). Say we have Total.s, from the Linux assembly handout in PLN, and do this:

```
as -o tot.o Total.s
dis tot.o TotalNew.s
```

where **dis** is the disassembler, and it produces TotalNew.s from tot.o. What, if any, differences will there be between Total.s and TotalNew.s?

Solutions:

1. 11101

2. register

3. Need to write back to disk the file copies in the disk cache.

4. hardware (or ALU)

6a. true

6b. false

6c. write, write-through

7a. v

7b. 4

7c. The program would attempt to access the middle of a word, i.e. it would generate an address which is not a multiple of 4. A bus error would result.

7d. sizeof(int)

8.

```
MAR    MDR

204    206
204    205
```

9.

```
int ExtractByte(int X, int K)

{  char *P;

   P = (char *) &X;
   if (Endian()) return *(P+K-1);
   else return *(P+sizeof(int)-K);
}
```

10. The labels (variable names, etc.) would be missing. Also comments and directives.