Suggestions for making your OS

1. Use a linker script.

I had lots of trouble with my kernel until I used a linker script. Basically, the linker script assures that your kernel gets linked into the right order(text,data,bss). Here is an example linker script for a kernel loaded at 0xFF80000(John Fine's bootf2.zip bootsector does this). Note that this linkere script is for LD:

kernel.lnk

```
OUTPUT_FORMAT("binary")
ENTRY(start)
SECTIONS
{
  .text  0xFF800000 : {
    *(.text)
  }
  .data  : {
    *(.data)
  }
  .bss  :
  {
    *(.bss)
  }
}
```

Then link like this:

```
ld -T kernel.lnk kernel.o
```

2. Create a "boiler plate" file in ASM that runs the kernel.

I suggest that if your kernel is written in C, that you run your "kernel main" function from a ASM file like this(assuming PMode):

kernel_asm.asm

```
[bits 32] ; hey, we're in PMode

[global start]
[global _kernel_main] ; always add a "_" in front of a C function to call it

start:
  call _kernel_main
  jmp $ ; halt
```

kernel_c.c

```
kernel_main()
{
     k_init();
     k_sayhello();
     ...
};
```

Then link the two files together.

3. Use and emulator so you don't have to reboot a computer constantly.

Rebooting a computer over and over again is slow, decreases the computer's lifespan, and a computer can't tell you easily what happened wrong in your code that caused a 3rd exception or general protection fault or something. An emulator such as Bochs solves these problems.

4. Back up your code.

Back up your code whenever you make a major(or slightly major) change to it. This serves several purposes. First, it's easy to 'roll back' your source code if you've messed it up a lot. Second, you can use your backups to prove that you coded your OS if for some reason someone claims you just stole their code and modified it. And of course, you should also keep a backup of your code on a different computer or on a ZIP disk or CD somewhere in case your harddrive crashes and ruines your code.

5. When frustrated with a nasty bug or if you don't understand something, ask for help.

This is why OS development forums exist. The Mega-tokyo.com OS dev forum is a terrific place to ask OS dev questions are ask for help fixing a bug in some code. You can also email me a question, though due to the huge amount of email I get per day and my busy schedual, I might not be able to answer it quickly. And remember, no question is too stupid.

6. Focus on your kernel, not on fancy graphics.

So many people want to have fancy graphics. Wait to work on graphics after your kernel is mature or completed. Why? Because graphics programming(especially SVGA) is as hard or harder than coding a kernel.

Written by K.J. 2002. Updated 2002.11.20 by K.J.