Lab3 Traps 实习说明

本 lab 的任务是添加系统调用，理解 traps 的实现。

详细要求及提示见链接：
(https://pdos.csail.mit.edu/6.828/2020/labs/traps.html)

实习内容

Exercise 0 源代码阅读

阅读下列源代码，理解 xv6 traps。

- `kernel/trampoline.S`
- `kernel/trap.c`

Exercise 1 RISC-V assembly

执行 `make fs.img`，编译 `user/call.c` 生成 `user/call.asm`。

阅读 `call.asm`，回答以下问题：

1. Which registers contain arguments to functions? For example, which register holds 13 in main's call to `printf`?
2. Where is the call to function `f` in the assembly code for main? Where is the call to `g`? (Hint: the compiler may inline functions.)
3. At what address is the function `printf` located?
4. What value is in the register `ra` just after the `jalr` to `printf` in `main`?
5. Run the following code.

```
unsigned int i = 0x00646c72;
printf("H%x Wo%s", 57616, &i);
```

What is the output? Here's an ASCII table that maps bytes to characters.

The output depends on that fact that the RISC-V is little-endian. If the RISC-V were instead big-endian what would you set `i` to in order to yield the same output? Would you need to change `57616` to a different value?

Here's a description of little- and big-endian and a more whimsical description.

6. In the following code, what is going to be printed after `'y='`? (note: the answer is not a specific value.) Why does this happen?

```
printf("x=%d y=%d", 3);
```

Exercise 2 Backtrace

在 `kernel/printf.c` 中实现函数 `backtrace()`。

系统调用接收一个参数：一个指向 `struct sysinfo` 的指针（请参阅 `kernel/sysinfo.h`）。内核应填写此结构体的字段：`freemem` 字段应设置为可用内存的字节数，`nproc` 字段应设置为 `state` 为 `UNUSED` 的进程数。

Exercise 3 Alarm

向 xv6 添加一个警报功能，该功能会定期向使用 CPU 时间的进程发出警报。