实验报告

课程：高性能计算应用实践

姓名：王峻阳

学号：220110317

学院：计算机科学与技术学院

学期：2023 年秋季学期

实验日期：2023 年 9 月 25 日

一、实验内容

内容一：用 time 工具观察 `sleep()`和 `spin()`的异同。

令父进程等待 2 秒，子进程等待 1 秒。用 `spin()`，消耗的 user time 比较多，接近 2 秒；用 `sleep()`，user time 和 system time 都接近 0。这说明 `spin()`（反复检查时间）会一直执行程序，消耗 CPU 资源，而 `sleep()`等待时不执行程序，不消耗 CPU 资源。查阅相关资料得知，`sleep()`是由系统实现的，由系统决定何时继续执行程序的后续部分，消耗的性能远少于 `spin()`。

```
> time ./fork_spin
hello world (pid:1244)
hello, I am parent of 1245 (pid:1244)
hello, I am child (pid:1245)
end of process (pid:1245)
end of process (pid:1244)

real    0m2.001s
user    0m1.847s
sys     0m0.000s
```

```
> time ./fork_sleep
hello world (pid:1246)
hello, I am parent of 1247 (pid:1246)
hello, I am child (pid:1247)
end of process (pid:1247)
end of process (pid:1246)

real    0m2.166s
user    0m0.000s
sys     0m0.002s
```

内容二：练习使用 ps 等工具。

用 ps 工具可以直接看出上述两个程序在执行时的的区别：`fork_spin` 程序是 running（R）状态，说明它在执行中，而 `fork_sleep` 程序是 interruptible sleep（S）状态，说明它不在执行。

```
 3569 pts/1    R+     0:00      0     1  2770  1044  0.0 ./fork_spin
 3570 pts/1    R+     0:00      0     1  2770   100  0.0 ./fork_spin
```

```
 3553 pts/1    S+     0:00      0     0  2771   980  0.0 ./fork_sleep
 3554 pts/1    S+     0:00      0     0  2771   100  0.0 ./fork_sleep
```

还观察到在子进程结束后，父进程结束前，子进程变为 zombie（Z）状态。

```
 8998 pts/1    S+     0:00      0     0  2771   928  0.0 ./fork_sleep
 8999 pts/1    Z+     0:00      0     0     0     0  0.0 [fork_sleep] <defunct>
```

用 strace 工具发现 `sleep()`函数产生了系统调用 clock_nanosleep。

```
clock_nanosleep(CLOCK_REALTIME, 0, {tv_sec=2, tv_nsec=0}
```

top 跟 ps 功能类似，但提供的是实时的系统状态。pstree 提供进程之间关系的树状图。

二、遇到的问题和解决办法

问题一：在 manual 中查 time 命令，查到的参数不起作用。

“直接”执行的 time 命令实际上是 shell builtin，而 Linux 的 time 命令是在 /usr/bin/time。

问题二：Linux Programmer’s Manual 的 sleep 条目中写道，sleep()是由 nanosleep 实现的，但实际上是 clock_nanosleep。

应该是 bug。出于某种原因，实现与手册不一致，或者手册不应指定 sleep() 如何实现。