信号部分文档

基于信号的进程间/进程与用户间通信

使用场景：

有时用户想要杀死一个进程（比如在windows中的任务管理器），或者某个进程想要杀死另外的进程（实际上用户杀死进程也是通过系统调用来实现的），或者进程间通信（如Linux中的IPC）。原始的xv6没有这项功能，即如果我要强行停止某一个进程，我是没法操作的，或者说某一个进程想要杀死另外的进程，这也是没法实现的。

实现方式：

新建一个函数指针类型的结构体sighandler\_t。在proc结构体加成员变量uint signal和sighandler\_t sighandlers，表示目前收到的信号和信号处理函数的指针数组。

为了实现系统调用，实现了两个函数：signal与sigsend。系统调用signal是自定义进程的信号处理函数（也就是说它在系统启动时即开始运行，而且不可被其他进程杀死，“地位”高于其他进程）。系统调用sigsend功能是向指定进程发信号（也就是说，它与其他进程“地位”相同）。

为了让每一个进程开始的时候就拥有相应的信号处理函数，新建一个函数register\_handler，当收到信号时注册相应的信号处理函数。当调度算法选出一个执行的进程后，首先检查它是否收到了信号，若收到，则注册信号处理函数，并将CPU分配给该进程。

目前的信号id有0（杀死进程SIGINT）、1（杀死子进程SIGKILLCHILD）、2（通知父进程，它的子进程已被杀死SIGCHILDEXIT）。这里删去了往届较为混乱的宏定义，转而在代码注释中标明信号id的功能。

测试方式：

见sigtest.c文件，文件中执行一系列进程通信的系统调用，并有相应的输出信息。