程序计数器假设一个值的序列：
$a_0, a_1, \ldots, a_{n-1}$
其中a_k是某个相邻的指令I_k的地址，每次从a_k到a_{k+1}的过渡叫控制转移。这样的控制转移序列叫做处理器的控制流。
有平滑控制流和突变控制流

系统对系统状态的变化做出反应：
现代操作系统通过使控制流发生突变来对这些情况做出反应，将这些突变称之为异常控制流(ECF)。

- ECF是操作系统用来实现I/O，进程和虚拟存储器的基础
- 应用程序通过使用一个叫做陷阱或者系统调用的ECF形式，向操作系统请求服务
- 操作系统为应用程序提供了强大的ECF机制，用来创建新进程、等待进程终止、通知其他进程系统中的异常事件，以及检测和响应这些事件
- ECF是计算机系统中实现并发 的基本机制。

8.1 异常

异常就是控制流中的突变，用来响应处理器状态中的某些变化。
状态变化被称为事件。事件可能和当前指令的执行直接相关，比如发生虚拟存储器缺页等。
任何情况下，当处理器检测到有事件发生时，它就会通过一张叫做异常表的跳转表，进行一个间接过程调用（异常），到一个专门设计用来处理这类事件的操作系统子系统（异常处理程序）。

8.1.1 异常处理

异常与过程调用的区别：

- 过程调用时，在跳转到处理程序之前，处理器将返回地址压入栈中。然而根据异常的类型，返回地址要么是当前指令，要么是下一条指令
- 处理器也把一些额外的处理器状态压到栈里，在处理程序返回时，重新开始被中断的程序也会需要这些状态。
- 如果控制从一个用户程序转移到内核，那么所有这些项目都被压到内核栈中。
- 异常处理程序运行在内核模式下，它们对所有的系统资源都有完全的访问权限。
  一旦异常处理完成后，通过执行一条特殊的“从中断返回”指令，可选地返回到被中断的程序，该指令将适当的状态弹回到处理器的控制和数据寄存器中，
  如果异常中断的是一个用户程序，就将状态恢复为用yogtu户模式，然后将控制返回给被中断的程序。

8.1.2 异常的类别

- 中断：异步发生，是来自处理器外部的I/O设备的信号的结果, 总是返回到下一条指令
  同步和异步的最大区别在于：同步中传输方和接收方使用同步时钟，而异步通讯允许双方使用各自不同的时钟。

- 陷阱：同步，总是返回到下一条指令。
  陷阱最重要的用途是在用户程序和内核之间提供一个接口，叫系统调用。
- 故障：同步，潜在的可恢复的错误，可能返回到当前指令。
  若处理程序能够修正这个错误，就将控制返回到引起故障的指令而重新执行它，否则就将程序返回到内核中的abort例程，abort例程会终止引起故障的应用程序。
  如缺页异常
- 终止：同步，不可恢复的错误，不会返回

8.2 进程

异常是允许操作系统提供进程的概念所需要的基本构造块。

进程是一个执行中的程序的实例。系统中的每个程序都是运行在某个进程的上下文中。
上下文是由程序正确运行所需的状态组成：存放在存储器中的程序的代码和数据，栈，程序计数器，通用目标寄存器，环境变量以及打开文件描述符的集合。

进程提供给应用程序的关键的抽象：

- 一个独立的逻辑控制流，它提供一个假象，好像我们的程序独占地使用处理器
- 一个私有地地址控制，它提供过一个假象，好像我们地程序独占地使用存储器系统

8.2.1 逻辑控制流

程序计数器（PC）的值为一地对应于包含在程序地可执行目标文件中的指令，这个PC的值的序列叫做逻辑控制流。

8.2.2 并发流

并发流：一个逻辑流的执行在时间上与另一个流重叠。
并发；多个流并发地执行地一般现象
多任务：一个进程和其他进程轮流运行
时间片：一个进程执行它地控制流的一部分的每一时间段。
并发的思想和流运行的处理器核数或者计算机数无关。如果两个流在时间上重叠，那么它们就是并发的，即使它们是运行在同一个处理器上的。
并行流是并发流的一个真子集，如果两个流并发的运行在不同的处理器核或者计算机上，那么称之为并行流，它们并行的运行，且并行的执行。

私有地址空间

一个进程为每个程序提供它自己的私有地址空间，不和其他进程共享。
每个私有地址空间内容一般不相同，但是每个这样的空间都有相同的通用结构。
进程的地址空间？？？

8.2.4 用户模式和内核模式

内核模式：一个运行在内核模式的进程可以执行指令集中的任何指令，并且可以访问系统中任何存储器的位置
用户模式：只能执行特定的指定和访问特定的空间。

进程从用户模式变为内核模式唯一的方法是通过诸如中断、故障或者陷入系统调用这样的异常。

8.2.5 上下文切换

调度：在进程执行的某些时刻，内核可以决定抢占当前进程，并重新开始一个先前被抢占的进程
由内核中的调度器的代码处理。

上下文切换：

1. 保存当前进程的上下文
2. 恢复某个先前被抢占的进程被保护的上下文
3. 将控制权传递给这个新恢复的进程。

8.4 进程的控制

fork()

- fork()调用一次，返回两次，一次是返回到父进程，一次是返回到新创建的子进程。0为子进程，大于0为父进程，-1表示出错。
- 并发执行。父进程和子进程是并发运行的独立进程
- 相同的但是独立的地址空间。子进程拥有父进程堆栈段和数据段的拷贝。
- 共享文件：子进程继承了父进程所有打开的文件。
  vfork()
- 创建的子进程与父进程共享数据段和堆栈段
- 创建子进程后要保证子进程先运行，直到子进程调用exec和exit之后，或者子进程结束才可以运行。

8.4.3 回收子进程

僵死进程：一个终止了但是还没有被回收的进程称为僵死进程。
当一个进程由于某种原因终止时，内核并不是立即把它从系统中清楚。相反进程被保持在一个已终止状态中。直到它被它的父进程回收。当父进程回收已终止的
子进程的时候，内核将子进程的退出状态传递给父进程，然后抛弃已终止的进程。
函数：wait和waitpid

8.4.4 让进程休眠

8.4.5 加载并运行进程

execve()函数和fork()函数的区别
fork()函数在新的子进程中运行相同的程序，新的子进程是父进程的一个复制品。
execve()函数在当前进程的上下文中加载并运行一个新程序，它会覆盖当前进程的地址空间，但并没有创建一个新进程。新的程序仍然有相同的PID，并且
继承了调用execve()函数时已打开的所有文件描述符。

信号（后续再看）