# 1、进程相关概念

## 程序与进程

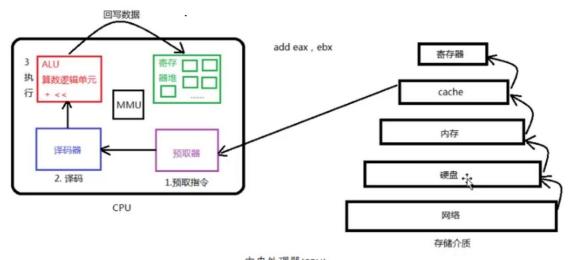
程序在磁盘上,不占用系统资源

进程是活跃的程序,占用系统资源

### 多道程序设计模式

并行运行 时间片轮转 时钟中断 (硬件手段)

### CPU和MMU

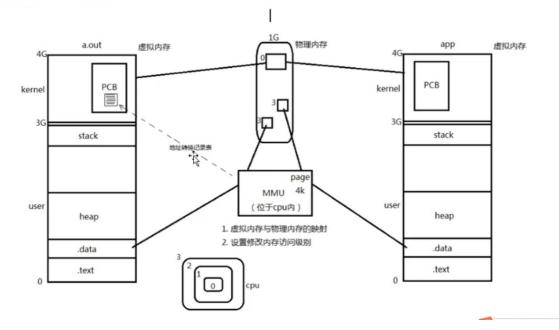


中央处理器(CPU)

进到CPU的是程序的其中的一小部分,也就是一条指令所对应的二进制流

预取器主要负责从cache缓冲区取指令。交给译码器进行译码,分析指令是干什么的,看需要哪些寄存器进行处理。再交给ALU进行运算,ALU只会加法和左移运算,算好后再将数据写回寄存器,寄存器再返回cache缓冲区进行保存。

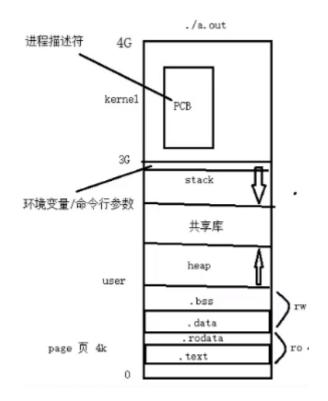
MMU(内存管理单元)主要是用来完成虚拟内存与物理内存的对应,修改CPU内存访问级别语言。



text代码、data数据、heap堆区(低地址向高地址生长)、stack栈区(高地址向低地址生长)、kernel内核区。这0-4G是虚拟内存空间。0-3G为用户空间,不可访问内核区,内核区可以访问用户区。

虚拟地址:可用地址空间有4G(逻辑地址)

用户内存映射到不同的物理地址空间,但内核区PCB位于同一块物理内存中。



## 进程控制块

每个进程在内核中都有一个进程控制块(PCB)来维护进程相关信息,Linux内核的进程控制块是一个task\_struct结构体。包含:

- 进程id,系统中每个进程有唯一的id,在c语言中用pid\_t类型表示,其实就是一个非负整数。
- 进程的状态,有就绪、运行、挂起、停止等状态。
- 进程切换时需要保存和恢复的一些CPU寄存器的值。
- 面熟虚拟地址空间的信息。
- 描述控制终端的信息。
- 当前工作目录

- umask掩码(保护文件创建或者修改权限)
- 文件描述符表,包含很多指向file结构体的指针。
- 和信号量相关的信息
- 用户id和组id
- 会话 (Session)和进程组
- 进程可以使用的资源上限

### 进程的状态

进程的基本状态有五种。分别为初始态、就绪态、运行态、挂起态和停止态。

挂起:等待除CPU意外的其他资源,主动放弃CPU。

## 2、环境变量

环境变量是指在操作系统中用来指定操作系统运行环境的一些参数。

存储形式:与命令行参数类似。char \*[]数组,数组名environ,内部存储字符串,NULL作为哨兵结尾

使用形式:与命令行参数类似

加载位置:与命令行参数类似。位于用户区,高于stack的起始位置。

引入环境变量表:需声明环境变量。extern char \*\*environ; (extern对一个变量进行声明或导出)

```
user@localhost:~ _ □ ×

File Edit View Search Terminal Help
#include<stdio.h>

extern char **environ; 引入环境变量表,通常我们将环境变量设为全局变量
int main(void)
{
    int i;
    for(i=0;emviron[i];i++){
        printf("%s\n",environ[i]); 循环输出环境变量
    }
    return 0;
}
```

```
user@localhost:~
                                                                            File Edit View Search Terminal Help
[user@localhost ~]$ make test_environ 构造程序cc test_environ.c -o test_environ
[user@localhost ~]$ ./test_environ
                                    运行程序
XDG VTNR=1
SSH_AGENT_PID=1562
XDG SESSION ID=1
HOSTNAME=localhost.localdomain
IMSETTINGS_INTEGRATE_DESKTOP=yes
TERM=xterm-256color
SHELL=/bin/bash
XDG MENU PREFIX=gnome-
VTE_VERSION=4602
HISTSIZE=1000
GJS DEBUG OUTPUT=stderr
                                   下面这些都是环境变量
WINDOWID=46139392
GJS DEBUG TOPICS=JS ERROR; JS LOG
IMSETTINGS MODULE=none
USER=user
LS COLORS=rs=0:di=38;5;27:ln=38;5;51:mh=44;38;5;15:pi=40;38;5;11:so=38;5;13:do=3
8;5;5:bd=48;5;232;38;5;11:cd=48;5;232;38;5;3:or=48;5;232;38;5;9:mi=05;48;5;232;3
8;5;15:su=48;5;196;38;5;15:sg=48;5;11;38;5;16:ca=48;5;196;38;5;226:tw=48;5;10;38
;5;16:ow=48;5;10;38;5;21:st=48;5;21;38;5;15:ex=38;5;34:*.tar=38;5;9:*.tgz=38;5;9
:*.arc=38;5;9:*.arj=38;5;9:*.taz=38;5;9:*.lha=38;5;9:*.lz4=38;5;9:*.lzh=38;5;9:*
.lzma=38;5;9:*.tlz=38;5;9:*.txz=38;5;9:*.tzo=38;5;9:*.t7z=38;5;9:*.zip=38;5;9:*.
```

### 常见环境变量

#### **PATH**

可执行文件的搜索路径。例如我们输入命令date, shell解析器会自动遍历环境变量目录, 在/bin/date找到可执行命令执行。

#### **SHELL**

当前shell,它的值通常是/bin/bash.

#### **TERM**

当前终端类型,在图形界面终端下它的值通常是xterm,终端类型决定了一些程序的输出显示方式,比如图形界面终端可以显示汉字,而字符终端一般不可以。

#### **HOME**

当前家目录

#### **LANG**

当前使用语言(中、英、.....)

## 常见环境变量函数

#### getenv函数

传入环境变量名获取环境变量值

#### setenv函数

改变或添加环境变量

## 3、进程控制

## fork函数

返回值2个:

1.返回子进程id (pid\_t)

2.返回0(代表返回成功)

因为创建了子进程,所以有两个进程执行fork。父进程返回的是子进程id,子进程返回是0,可用此来识别进程为父进程还是子进程。

getpid()获取子进程id

getppid()获取父进程id

新建进程由进程创建处开始执行。所以在程序中, fork()函数后的代码命令会执行两遍。

getuid()获取当前进程实际用户id

geteuid()获取当前进程有效用户id

## 4、进程共享

父子进程fork后

父子相同处:全局变量、.data、.text、栈、堆、环境变量、用户ID、宿主目录、进程工作目录、新号

处理方式.....

父子不同处:进程id、fork返回值、父进程id、进程运行时间、定时器、未决信号集

父子进程间遵循读时共享写时复制原则。