卡通画

描述已自动生成

**操作系统实验报告**

**实验题目**  进程的状态与进程调度

**学生姓名**  杨程锦

**学 号**  2021214710

**专业班级**  计科21-1班

**指导教师**  田卫东

**完成日期**  2023.12.21

**合肥工业大学 计算机与信息学院**

1. **实验目的和任务要求**

调试进程在各种状态间的转换过程，熟悉进程的状态和转换。

 通过对进程运行轨迹的跟踪来形象化进程的状态和调度。

 掌握Linux下的多进程编程技术。

1. **实验原理**

该实验内容包括使用Linux 0.11应用程序调用fork函数创建多个子进程，并编写一个可以创建多个子进程的Linux 0.11应用程序；通过数据可视化工具观察父进程与子进程的运行轨迹。

1. **实验内容**

**在Linux 0.11应用程序中调用fork函数创建多个子进程**

**编写一个可以创建多个子进程的Linux 0.11应用程序**

请读者按照下面的步骤使用fork函数编写一个可以创建多个子进程的Linux 0.11应用程序，一方面可以学习Linux的多进程编程技术，另一方面，通过观察多个进程的运行过程，可以对进程的调度过程有一个初步认识。

1. 使用VSCode打开之前克隆到本地的Linux 0.11应用程序项目。

2. 打开linuxapp.c文件main函数，让父进程新建三个子进程，并分别输出子进程id及其父进程id。代码如下：

int main( int argc, char \* argv[] )

{

if( 0 == fork() )

{

printf("child process pid=%d ppid=%d line=%d\n",

getpid(), getppid(), \_\_LINE\_\_);

}

else if( 0 == fork() )

{

printf("child process pid=%d ppid=%d line=%d\n",

getpid(), getppid(), \_\_LINE\_\_);

}

else if( 0 == fork() )

{

printf("child process pid=%d ppid=%d line=%d\n",

getpid(), getppid(), \_\_LINE\_\_);

}

else

{

wait( NULL );

printf("parent process pid=%d ppid=%d\n",getpid(), getppid());

}

return 0;

}

其中，getppid函数可以在子进程中获取父进程的id。“\_\_LINE\_\_”是GCC编译器提供的一个预定义宏，它的值是其在源代码文件中的行号。

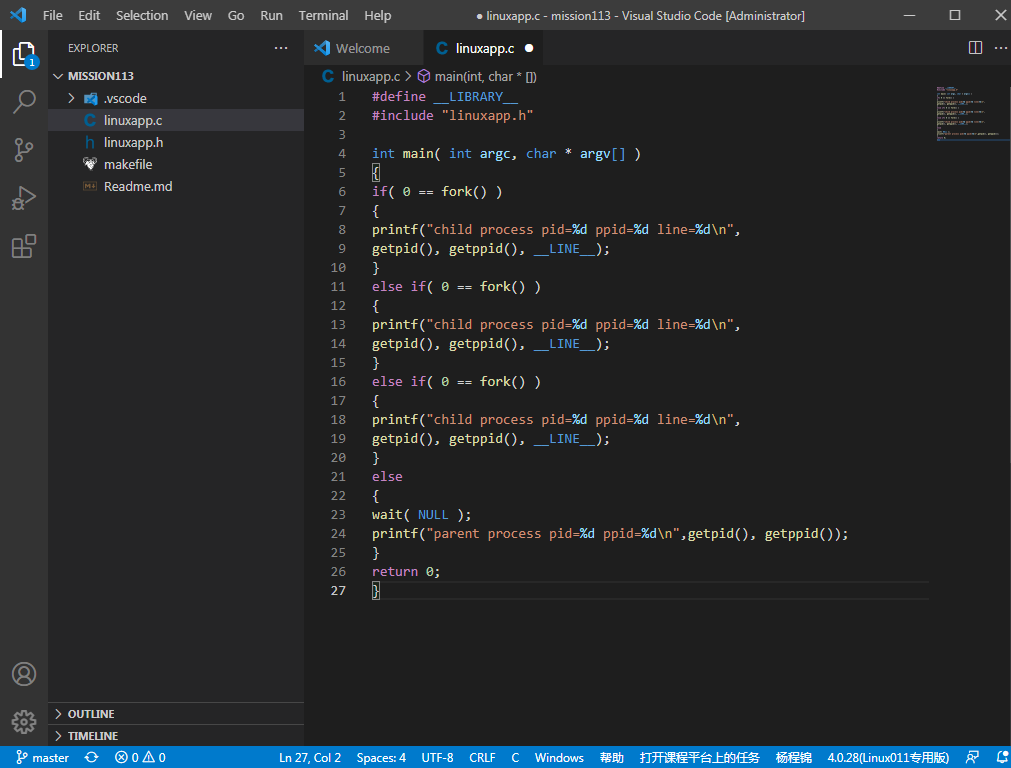


图 1 linuxapp.c文件

3. 不能在VSCode中按F5启动调试Linux 0.11应用程序。只能在VSCode的“Terminal”菜单中选择“Run Build Task...”，会在VSCode的顶部中间位置弹出一个可以执行的Task列表，选择其中的“Bochs 运行(不调试)”。

4. 待Linux 0.11启动完毕后，将生成的可执行文件从软盘B拷贝到硬盘，命令如下：

mcopy b:linuxapp.exe app

5. 为app文件添加可执行权限，命令如下：

chmod +x app

6. 执行“sync”命令确保app文件写入硬盘。

7. 使用命令./app运行可执行文件app，观察各个进程开始执行的顺序和结束执行的顺序，理解进程在其生命周期中状态的转换过程和进程调度过程。

文本

描述已自动生成

图 2 运行app可执行文件

8. 结束调试，关闭Bochs虚拟机。

**查看多个子进程的运行轨迹**

在前面的练习中，读者可以通过父进程和子进程在屏幕上打印输出的信息来间接的判断出父进程和子进程运行的轨迹，但是并不直观，得到的信息也不全面。接下来，读者可以使用数据可视化窗口，更加直观、形象的观察到父进程与子进程的运行轨迹。步骤如下：

1. 使用VSCode的“File”菜单中的“Open Folder”打开之前克隆到本地的Linux 0.11内核项目文件夹。

2. 在VSCode的“Terminal”菜单中选择“Run Build Task...”，会在VSCode的顶部中间位置弹出一个可以执行的Task列表，选择其中的“生成项目”。

3. 待Linux 0.11内核项目生成成功后，使用Windows资源管理器分别打开之前克隆到本地的Linux 0.11应用程序项目和Linux 0.11内核项目所在的文件夹。

4. 用Linux 0.11应用程序项目文件夹中的硬盘镜像文件harddisk.img覆盖Linux 0.11内核项目文件夹中的harddisk.img文件，这样就可以在Linux 0.11内核项目的硬盘中使用之前生成的app文件了。

5. 使用VSCode的“File”菜单中的“Open Folder”打开之前克隆到本地的Linux 0.11内核项目文件夹。

6. 为了方便观察app应用程序中的父进程和子进程的运行轨迹，需要在父进程结束的位置添加一个条件断点。请读者在kernel/exit.c文件的第166行（进程结束后触发进程调度的位置）添加一个条件断点，条件为“current->pid == 6”。

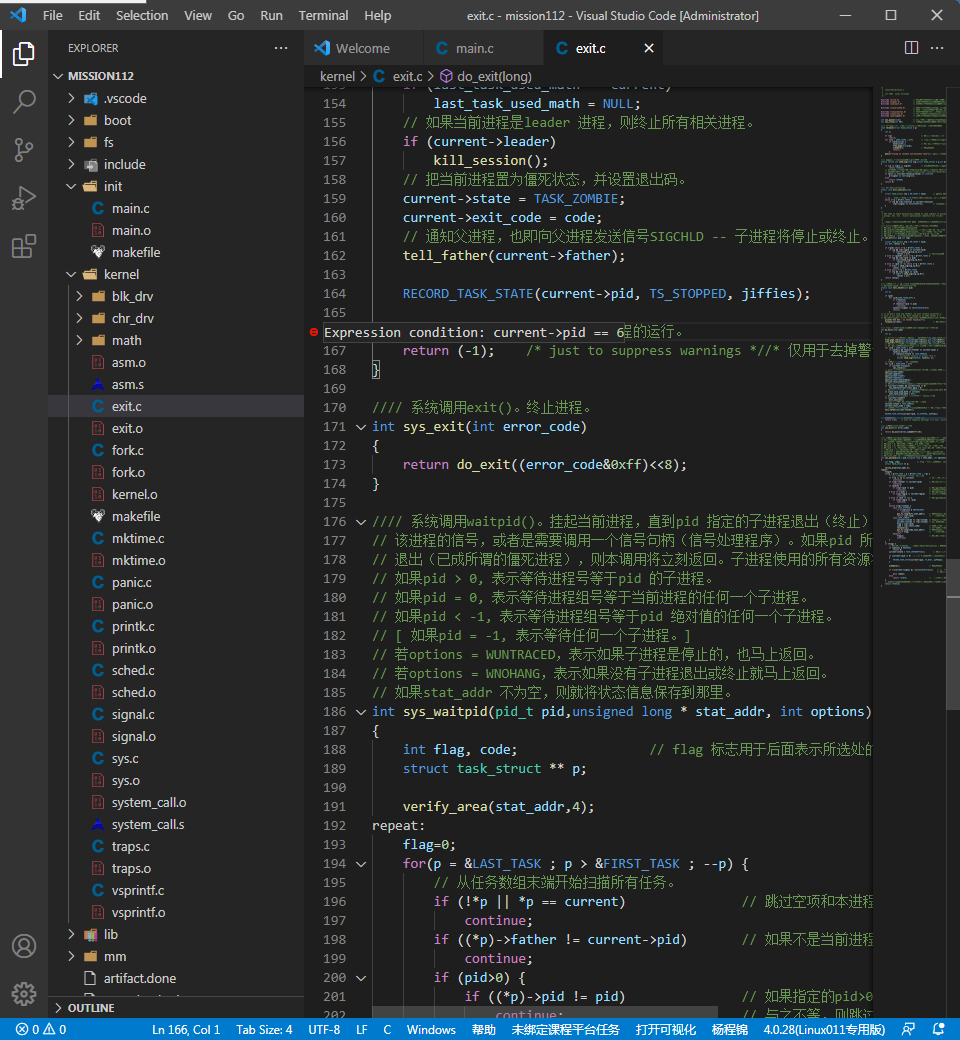


图 3 添加条件断点

7. 按F5启动调试。输入命令./app后，会命中刚刚添加的条件断点。

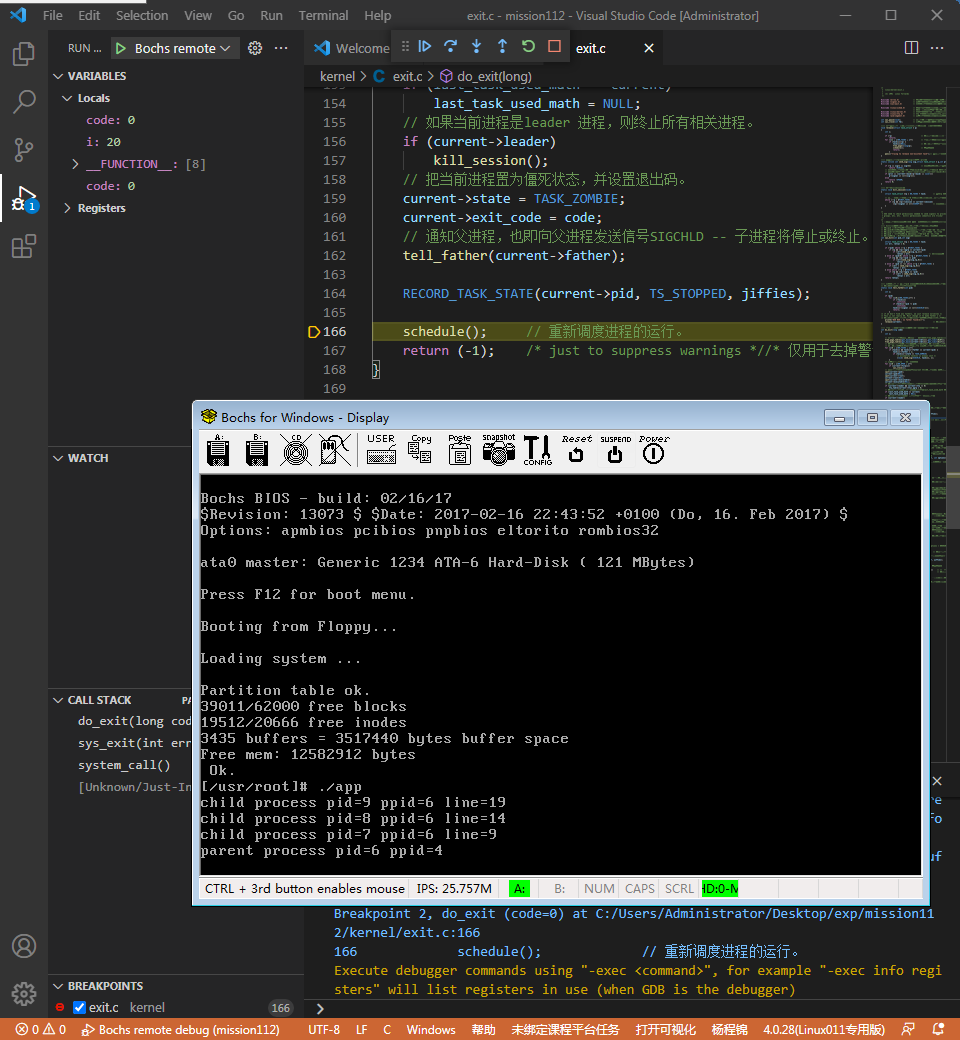


图 4 调试并命中断点

8. 在VSCode的“View”菜单中选择“Command Palette...”，会在VSCode的顶部中间位置显示命令面板，输入“Lab: New Visualizer View”命令后，VSCode会在其右侧弹出一个窗口让读者查看可视化视图。在右侧可视化视图顶部的编辑框中输入命令“#sched”后按回车（需要等待较长时间完成刷新），就可以查看进程运行的轨迹了，如5所示。

图表

描述已自动生成

图 5 进程运行轨迹

在进程运行轨迹图示的最左侧是jiffies，表示每一次进程状态转换发生

1. **总结和感想体会**

这份报告展示了我在进程状态与进程调度方面的实验学习。该实验内容包括使用Linux 0.11应用程序调用fork函数创建多个子进程，并编写一个可以创建多个子进程的Linux 0.11应用程序。该过程包括使用VSCode打开项目，编写代码以创建多个子进程，并通过一系列的步骤来运行和观察这些进程。我记录了每个实验步骤，包括代码编写、程序构建、调试过程，以及如何通过数据可视化工具观察父进程与子进程的运行轨迹。报告中展示了对进程状态和调度的深入理解，以及对Linux多进程编程的实际应用。