山东大学 软件 学院

操作系统 课程实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 学号：201400301291 | 姓名： 雷超 | | 班级： 四班 |
| 实验题目：进程调度算法实验 | | | |
| 实验学时：2 | | 实验日期： 2016-04-19 | |
| 实验目的：  加深对于进程调度概念的理解，体验进程调度机制的功能，了解Linux系统中进程调度策略的使用方法。练习进程调度算法的编程和调试技术。 | | | |
| 硬件环境：  00:00.0 Host bridge: Intel Corporation Xeon E3-1200 v3/4th Gen Core Processor DRAM Controller (rev 06)  00:01.0 PCI bridge: Intel Corporation Xeon E3-1200 v3/4th Gen Core Processor PCI Express x16 Controller (rev 06)  00:02.0 VGA compatible controller: Intel Corporation 4th Gen Core Processor Integrated Graphics Controller (rev 06)  00:03.0 Audio device: Intel Corporation Xeon E3-1200 v3/4th Gen Core Processor HD Audio Controller (rev 06)  00:14.0 USB controller: Intel Corporation 8 Series/C220 Series Chipset Family USB xHCI (rev 05)  00:16.0 Communication controller: Intel Corporation 8 Series/C220 Series Chipset Family MEI Controller #1 (rev 04)  00:1a.0 USB controller: Intel Corporation 8 Series/C220 Series Chipset Family USB EHCI #2 (rev 05)  00:1b.0 Audio device: Intel Corporation 8 Series/C220 Series Chipset High Definition Audio Controller (rev 05)  00:1c.0 PCI bridge: Intel Corporation 8 Series/C220 Series Chipset Family PCI Express Root Port #1 (rev d5)  00:1c.1 PCI bridge: Intel Corporation 8 Series/C220 Series Chipset Family PCI Express Root Port #2 (rev d5)  00:1c.2 PCI bridge: Intel Corporation 8 Series/C220 Series Chipset Family PCI Express Root Port #3 (rev d5)  00:1c.4 PCI bridge: Intel Corporation 8 Series/C220 Series Chipset Family PCI Express Root Port #5 (rev d5)  00:1d.0 USB controller: Intel Corporation 8 Series/C220 Series Chipset Family USB EHCI #1 (rev 05)  00:1f.0 ISA bridge: Intel Corporation HM86 Express LPC Controller (rev 05)  00:1f.2 SATA controller: Intel Corporation 8 Series/C220 Series Chipset Family 6-port SATA Controller 1 [AHCI mode] (rev 05)  00:1f.3 SMBus: Intel Corporation 8 Series/C220 Series Chipset Family SMBus Controller (rev 05)  01:00.0 3D controller: NVIDIA Corporation GM107M [GeForce GTX 860M] (rev a2)  08:00.0 Network controller: Intel Corporation Wireless 3160 (rev 93)  09:00.0 Ethernet controller: Realtek Semiconductor Co., Ltd. RTL8111/8168/8411 PCI Express Gigabit Ethernet Controller (rev 10)  0a:00.0 Unassigned class [ff00]: Realtek Semiconductor Co., Ltd. RTS5249 PCI Express Card Reader (rev 01) | | | |
| 软件环境：  Linux EVA 4.6.2-gentoo-EVOLUTION x86\_64 Intel(R) Core(TM) i7-4710HQ CPU @ 2.50GHz GenuineIntel GNU/Linux  dev-libs/gmp-6.0.0a  dev-libs/mpfr-3.1.3\_p4  dev-libs/mpc-1.0.2-r1  sys-devel/gcc-4.9.3 , 5.3.0 , 5.4.0  sys-libs/glibc-2.22-r4  sys-devel/clang-3.5.0-r100  app-editors/emacs-24.5-r1  sys-kernel/gentoo-sources-4.6.2  sys-kernel/linux-headers-4.6  sys-apps/systemd-226-r2  ABI\_X86=”32 64” | | | |
| 实验步骤与内容：  分别建立两个并发的执行的父子进程，不断循环输出各自进程号，优先数和调度策略。进程初始调度策略均为系统默认和默认优先级。当父进程收到Ctrl+C发出的SIGINT信号时优先数加一，子进程收到Ctrl+Z发出的SIGINT信号时自动减一。  此次实验首先需要在建立了两个进程之后对于各自的进程分别在内核注册不用的信号响应函数，然后根据相关的系统调用的函数在ROOT权限下更改优先级，总体来说这次实验是非常简单的。 | | | |
| 结论分析与体会：  此次实验在多核的处理器上体现的效果不是非常明显，但是通过更改内核参数，即对于Processor type and features 中的 Symmetric multi-processing support 的选项进行更改，并且对内核进行重新编译后得到了相对可以的结果。  或者进行密集型的科学计算，中间I/O不进行输出，只有最后的时候进行I/O输出，此时也可以得到相应的结果，这让我加深理解了Linux操作系统中关于进程调度的理解和不同内核参数对于操作系统的影响。这是我收获程度颇丰的一个实验。 | | | |

附件：