## 实验目的

学习 linux 内核、进程、存储和其他资源的一些重要特性。通过使用/proc 文件系统接口,编写一个程序检查反映机器平衡负载、进程资源利用率方面的各种内核值, 学会使用/proc文件系统这种内核状态检查机制。

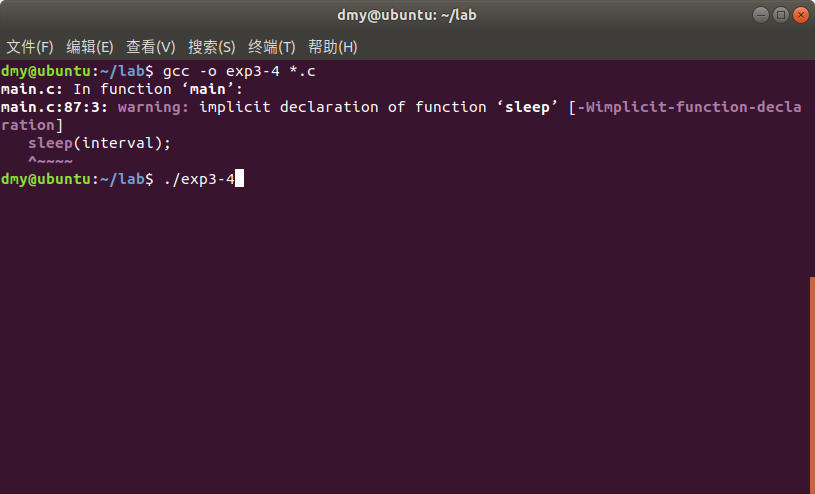
## 实验内容

编写一个默认版本的程序通过检查内核状态报告 Linux 内核行为。程序应该在 stdout上打印以下值：

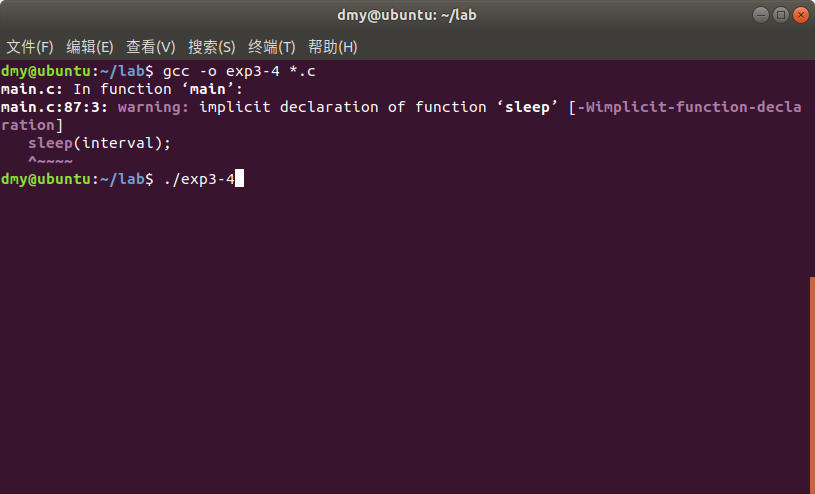
1. CPU 类型和型号；
2. 所使用的 Linux 内核版本；
3. 从系统最后一次启动以来已经经历了多长时间（天，小时和分钟）；
4. 总共有多少 CPU 时间执行在用户态，系统态，空闲态；
5. 配置内存数量；当前可用内存数，磁盘读写请求数；
6. 内核上下文转换数；

## 实验步骤

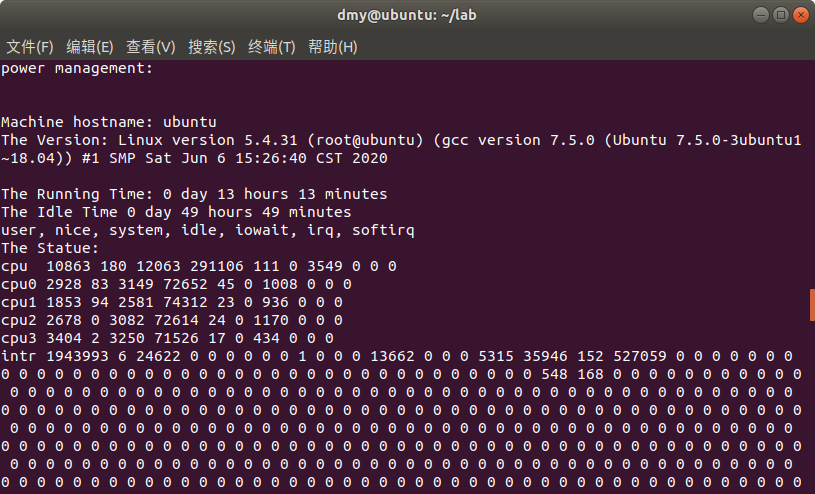
编译运行



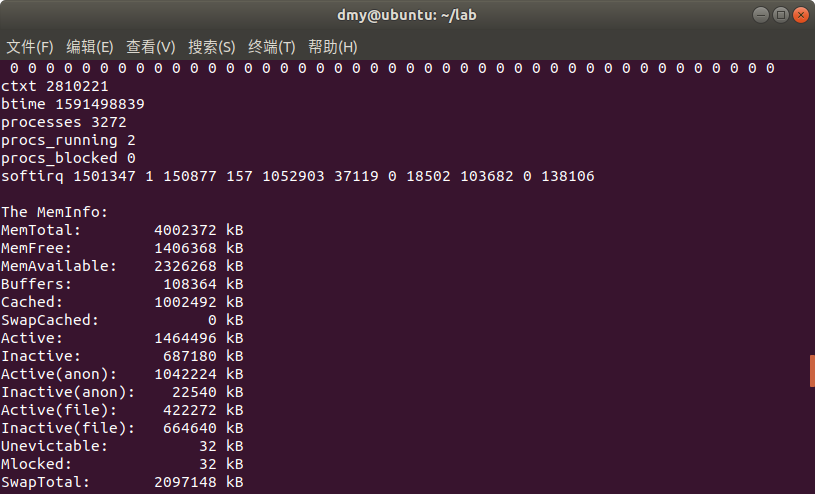
CPU信息



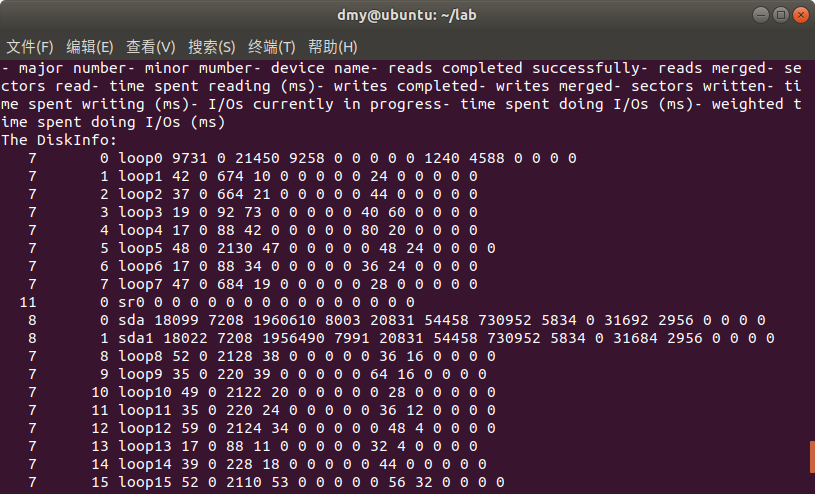
内核版本号，系统运行时间，CPU在用户态、系统态、空闲态时间



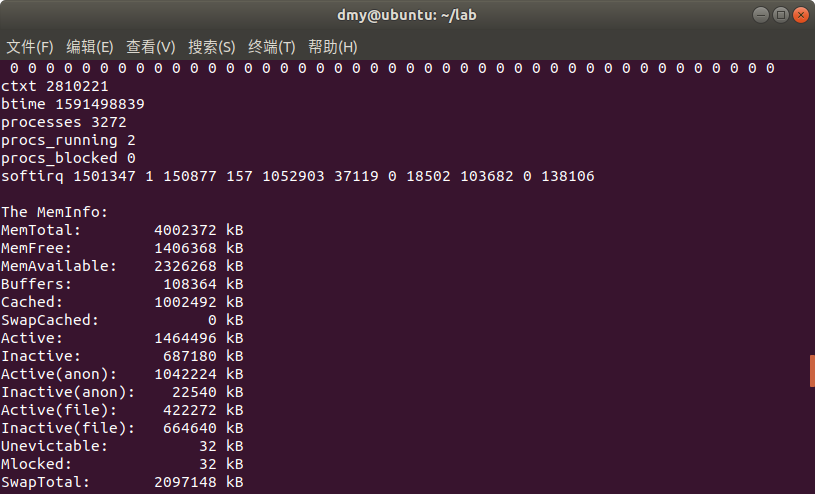
内存数量、可用内存数量



磁盘读写请求数



上下文转换数 ctxt



代码，具体见main.c

|  |
| --- |
| #include<stdio.h>  #include<time.h>  #include<sys/procfs.h>  #include<string.h>  #include<stdlib.h>  int getproc(char\* filename,char \*buf,int max\_size)  {  char path[256];  sprintf(path, "/proc/%s", filename);  FILE\* thisProcFile;  thisProcFile = fopen(path, "r");  int i = 0;  fscanf(thisProcFile, "%[^\377]", buf);  fclose(thisProcFile);  return 0;  }  int main(int argc, char\* argv[])  {  char repTypeName[16];  char c1, c2, ch;  int interval, duration;  char lineBuf[65536];  int LB\_SIZE = 65535;  char now[65536];  int iteration;  //determinate report type  strcpy(repTypeName, "Standard");  if (argc < 1)  {  sscanf(argv[1], "%c%c", &c1, &c2);  if (c1 != '-')  {  fprintf(stderr, "usage:ksamp [-s][-1 inte dur]\n");  exit(1);  }  if (c2 == 's')  {  strcpy(repTypeName, "Short");  }  if (c2 == '1')  {  strcpy(repTypeName, "Long");  interval = atoi(argv[2]);  duration = atoi(argv[3]);  }  }  //current time  time\_t timep;  timep = time(NULL);  printf("Status report type %s at %s\n", repTypeName, ctime(&timep));  //CPU type  getproc("cpuinfo", lineBuf, LB\_SIZE + 1);  printf("The CPU: %s\n", lineBuf);  //hostname  getproc("sys/kernel/hostname", lineBuf, LB\_SIZE + 1);  printf("Machine hostname: %s", lineBuf);  //kernel version  getproc("version", lineBuf, LB\_SIZE + 1);  printf("The Version: %s\n", lineBuf);  //runing time  getproc("uptime", lineBuf, LB\_SIZE + 1);  float t1, t2;  sscanf(lineBuf, "%f%f", &t1, &t2);  printf("The Running Time: %d day %d hours %d minutes\n",(int)t1/86400, (int)t1%86400/60, (int)t1/60 );  printf("The Idle Time %d day %d hours %d minutes\n", (int)t2 / 86400, (int)t2 % 86400 / 60, (int)t2 / 60);  //current statue  printf("user, nice, system, idle, iowait, irq, softirq\n");  getproc("stat", lineBuf, LB\_SIZE + 1);  printf("The Statue:\n%s\n", lineBuf);  //memory info  getproc("meminfo", lineBuf, LB\_SIZE + 1);  printf("The MemInfo:\n%s\n", lineBuf);  //disk info info  puts("- major number- minor mumber- device name- reads completed successfully- reads merged- sectors read- time spent reading (ms)- writes completed- writes merged- sectors written- time spent writing (ms)- I/Os currently in progress- time spent doing I/Os (ms)- weighted time spent doing I/Os (ms)");  getproc("diskstats", lineBuf, LB\_SIZE + 1);  printf("The DiskInfo:\n%s\n", lineBuf);  //check average load  for(iteration = 0,interval = 2,duration = 10; iteration < duration; iteration += interval)  {  sleep(interval);  getproc("loadavg", lineBuf, LB\_SIZE + 1);  printf("%s\n", lineBuf);  }  return 0;  } |