

计算机系统 (A) 试 题

主管
领导
审核
签字

题号	一	二	三	四	五	六	七	八	九	十	总分
得分											
阅卷人											

片纸鉴心 诚信不败

授课教师

姓名

学号

系别

密

封

线

一、单项选择题 (每小题 2 分, 共 20 分)

- 下列软件哪一个不用于可执行程序的生成 (C)
A. cc1 B.as C.gdb D.cpp
- long 类型在 64 位的 Linux 系统中占 (D) 个二进制位
A. 8 B. 16 C. 32 D. 64
- 指令 movsbl \$0xfe,%eax 执行后 eax 结果为 (B)
A. 0xfffe B.-2 C.0xfffffe D.-1
- 下列最快的存储器是 (A)。
A.L1 Cache B.L2 Cache C.固态硬盘 D.光盘
- 计算机的存储管理单元访问 TLB 使用的地址是 (A)
A. 虚拟地址 B.物理地址 C. 逻辑地址 D.线性地址
- fork 后父进程与子进程不同的是 (C)
A. 地址空间 B.优先级 C. pid D.占用时间片
- 模块 m.c 中 extern int x; 语句说明程序中的 x 是 (A) 符号
A.全局变量 B. 局部变量 C.强符号 D.弱符号
- 存储器山中的 size 与 stride 反映程序局部性, 正确的是 (A)
A. size 时间局部性, stride 空间局部性 B.size 空间局部性, stride 时间局部性
C. size 时间局部性, stride 时间局部性 D. size 空间局部性, stride 空间局部性
- 关于异常与信号, 描述正确的是 (D)
A. 信号是一种异常类型。 B. 信号类型比异常多。
C. 按键产生硬件的异步异常, 不会产生信号。
D. 异常处理子程序运行在内核态, 信号处理子程序运行在用户态
- 下列那些不属于面向 CPU 的优化 (B)
A. 减少数据相关。 B. 复杂指令简化
C. 使用 sse 或 avr 指令 D.采用分离累加器的循环展开。

二、填空题（10 分，每空 2 分）

11. 程序中 float x 为正无穷大，则 &x 地址开始处的 4 个字节值是（十六进制）
_00_00_80_7F_____。（倒序也算对）0 111 1 111 1 000000000000000000000000
12. 函数调用 c=f(a,b,c); 64 位环境下，c 采用__rdx__寄存器传递参数。
13. 反汇编后 Intel x64 指令如下，400800: e8_ff_ff_fd_fb__callq_400600
14. 对一个无源码的可执行程序进行监控与性能分析，应采用
__加载运行时__打桩方式。（运行时、加载时 都算对）
15. 调用一次可返回两次的函数是__fork__

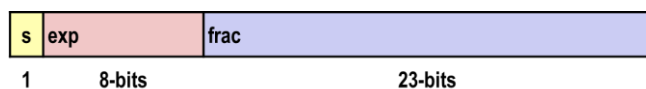
三、判断对错（共 10 分，每题 2 分，正确打√、错误打×）

16. (×) 机器中浮点数的阶码，是有符号整数，采用补码表示。
17. (×) switch 多分支都采用跳转表机制来进行机器级实现
18. (√) 相对于程序的代码与数据区，堆栈在内存的高地址。
19. (×) 对一个固定内存地址引用，编译器在循环体中使用寄存器实现以加快访问
20. (√) CTRL-Z 向前台进程组发送停止信号

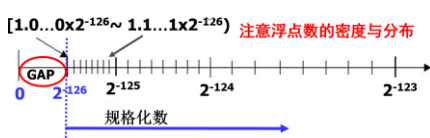
四、简答分析题（共 40 分，每题 10 分）

21. 简述浮点数的机器级表示方法，分析其在数轴上的分布规律，并说明为什么不建议直接进行两个浮点数的比较。

答：浮点数在机器内采用 IEEE754 编码，以 float 类型为例：共 32 bits，1 个符号位，8 位指数（127 移码），23 位尾数（先导为 1 的规格化）



浮点数在数轴上越靠近原点 0，密度越高，精度越高，越远离原点 0 则密度越低，精度也越低。以 float 为例，其分布如下。



由于浮点数存在精度问题，所以即使两个数不等，但由于精度问题，其机器内表示可能相同，所以不能直接进行两个浮点数比较。可以用其差小于某精度范围的数来判断。

22. 下列函数 f 的机器级表示，请阅读分析每一条语句。

```

0x4011d8  mov    (%rsi), %ebx  ____将参数 y 地址内数据 =>  ebx ____
0x4011da  neg     %ebx             ____ebx 内容取反____
0x4011dc  mov    (%rdi), %edx  ____将参数 x 地址内数据 =>  edx ____
0x4011de  neg     %edx
0x4011e0  mov    %edx, (%rsi)  ____将 edx 值 => 参数 y 地址指向的内存____
0x4011e2  mov    %ebx, (%rdi)  ____将 ebx 值 => 参数 x 地址指向的内存____
0x4011e4  retq

```

请分析此函数的各参数及返回值类型, 在空格处说明每条语句的功能。

答: 此函数有两个参数 x 、 y , 无返回值。

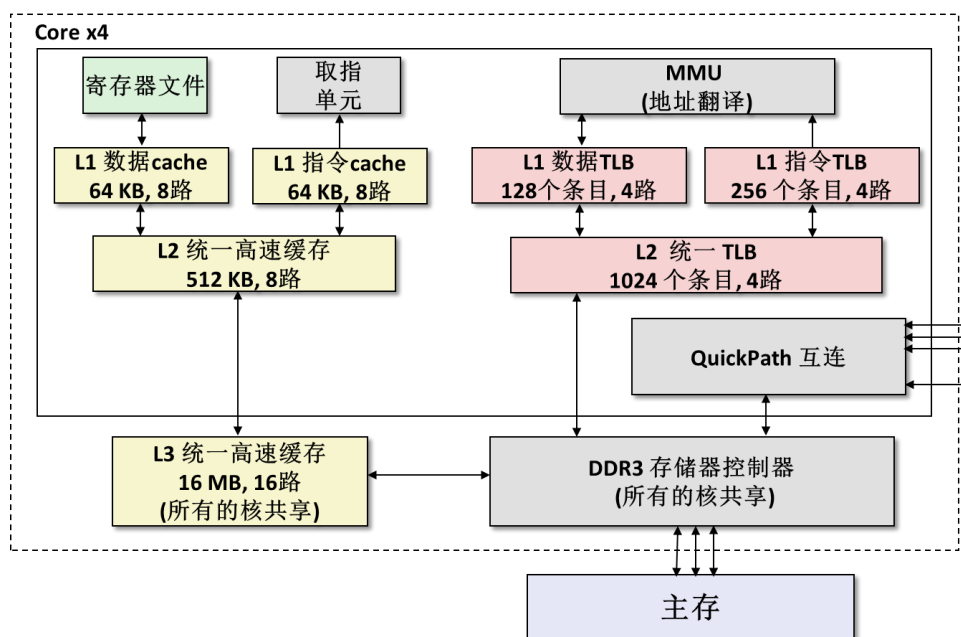
两个参数类型为 $\text{int } *x$; $\text{int } *y$;

两个参数都采用传地址方式。

22. 请给出此子程序的 C 语言实现形式, 包括函数声明。

```
void swapn(int *x, int *y)
{
    int t1, t2;
    t1=*y;
    t1=-t1;
    t2=*x;
    t2=-t2;
    (*y)=t2;
    (*x)=t1;
}
```

24. Intel I7 CPU, 虚拟地址 48 位, 物理地址 52 位, 每页 4KB, 每块 64B。



请分析此 CPU 的体系结构, 填写如下数据: 虚拟地址中的 VPN 占__36__位; 其一级页表为__512__项。 L1 指令 TLB 的组索引位数 TLBI 为__6__位。 L1 指令 Cache 共__128__组。用物理地址访问 L1 数据 Cache 时, Cache 标记 CT 占__39__位。

简述 CPU 对多级 Cache 及 TLB 的访问流程。

答: 指令中的逻辑地址经段式存储管理变换成虚拟地址, 访问 TLB1、TLB2、页表 1、页表 2、页表 3、页表 4 (若命中则获得物理页号, 返回), 获得物理地址。 如果 TLB 或页表中的 PTE 未缓存, 则发生缺页中断, cpu 执行缺页中断处理子程序完成磁盘到物理内存的加载(可能替换), 更新页表与 TLB 等。

CPU 通过物理地址访问 L1、L2、L3、RAM (若命中则获得指令或数据, 返回), 取得指令或数据返回 CPU。

五、综合设计题（20 分）

25. 一个计算两个向量内积的函数 inner 如下：

```
void inner(vec *u,vec *v,double *dest){
    for(long i=0,*dest=0;i<vec_length(u);i++)
        *dest=*dest+get_vec_element(u,i)* get_vec_element(v,i);
}
typedef struct{
    long len;    //向量的元素个数
    double *data;
}vec;
long    vec_length(vec *v) 函数返回向量 v 的元素个数;
double  get_vec_element(vec *u,long i) 函数返回向量 u 的第 i 个元素值;
```

1. 请对 inner 程序面向现代计算机系统进行了优化，以提高程序性能。

写出优化后的程序（至少两种方法）（10 分）

2. 请给出你所采用的优化理由（至少两个理由），以及进一步优化的方法（10 分）

答：（1）采用一般有用的方法，代码移动，共享公用子表达式 get_vec_start 等。

（2）采用面向编译器的优化方法，把函数 vec_length 移到循环外。把函数 get_vec_element 用数组元素或指针访问替代。

（3）采用面向编译器的优化方法，用临时/局部变量累积结果，编译器会将其编译成寄存器，大大提高运行速度。

```
void inner (vec *u, vec *v, double *dest)
{
    long        i;
    long        length = vec_length(u);
    data_t      *ud = get_vec_start(u);
    data_t      *vd = get_vec_start(v);
    data_t      t = 1;
    for (i = 0; i < length; i++)
        t = t + ud[i] * vd[i];
    *dest = t;
}
```

（4）面向超标量 CPU 的优化方法：

采用带分离的累加器的循环展开。通过比较不同展开因子 L 时的最小 CPE，从而确定最优的 L 展开因子。

```
for(i=0;i<length;i+=L)
{
    t0=t0+ud[i]*vd[i];
    t1=t1+ud[i+1]*vd[i+1];
    .....共 L 级.....
}
```

（5）面向向量 CPU 的优化方法：

面向向量 CPU 优化：采用向量乘法指令及 YMM/ZMM 等寄存器编程

（6）面向 Cache 优化：

空间局部性：重新排列（局部变量、循环变量顺序重排）提高空间局部性

时间局部性：分块，考虑到 Cache 大小 B，使得内循环的所有数据都能够放在 Cache 内。。

注意：根据学生回答，针对性给分。任选 2 种即可满分，不扣分。