

主管  
领导  
审核  
签字

哈尔滨工业大学 2018 学年 秋 季学期

# 计算机系统（B） 试 题

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

## 片纸鉴心 诚信不败

### 一、 单项选择题（每小题 1 分，共 20 分）

1. C 语言中整数-1 与无符号 0 比较，其结果是（ A ）  
A. 大于 B. 小于 C. 可能大于可能小于 D. 无法比较
2. 在 x86-64 系统中，调用函数 `int gt (long x, long y)` 时，保存参数 y 的寄存器是（ B ）  
A. %rdi B. %rsi C. %rax D. %rdx
3. 在 Y86 的硬件结构中，不需要采用时序控制的部件是（ D ）  
A. 程序计数器 B. 条件码寄存器  
C. 数据内存 D. 指令内存
4. 进程 P1 是进程 P11 的父进程，P1 有全局变量 x=0，下列说法错误的是（ C ）  
A. P1 与 P11 有相同的地址空间  
B. P1 与 P11 是并发执行的独立进程  
C. 若 P1 与 P11 均对 x 执行一次加 1 操作，则 x=2  
D. P1 与 P11 有相同的代码和数据段
5. 下列异常中可能从异常处理返回也可能不返回的是（ C ）  
A. I/O 中断 B. 陷阱 C. 故障 D. 终止
6. 不属于进程上下文的是（ C ）  
A. 页全局目录 pgd B. 内核栈 C. 内核代码 D. 打开的文件表
7. 下列函数中属于系统调用且调用一次，从不返回的是（ B ）  
A. fork B. execve C. setjmp D. longjmp
8. 在 IEEE 浮点数标准中，单精度浮点数采用（ A ）位的小数字段对尾数进行编码。  
A. 23 B. 24 C. 52 D. 63
9. 下列各种存储器中存储速度最慢的是（ C ）  
A. 寄存器 B. 主存 C. 磁盘 D. 高速缓存
10. 连接过程中，赋初值的非静态全局变量名，属于（ A ）  
A. 强符号 B. 弱符号 C. 可能是强符号也可能是弱符号 D. 以上都错
11. 共享库（动态链接库）在程序的（ D ）阶段由动态链接器加载到任意的内存地址。  
A. 编译 B. 链接 C. 运行 D. 运行或链接

授课教师

姓名

学号

院系

12. 条件跳转指令 JZ 是依据 ( A ) 做是否跳转的判断。  
A. ZF      B. OF      C. SF      D. CF
13. 关于异常处理后返回的叙述, 错误的叙述是 ( B )  
A. 中断处理结束后, 会返回到下一条指令执行  
B. 故障处理结束后, 会返回到下一条指令执行  
C. 陷阱处理结束后, 会返回到下一条指令执行  
D. 终止异常, 不会返回
14. 虚拟页面的状态不包含 ( D )  
A. 未分配      B. 已分配未缓存      C. 已分配已缓存      D. 已缓存未分配
15. C 语言中不同类型的数值进行强制类型转换时, 下列说法正确的是 ( C )  
A. 从 int 转换成 float 时, 数值可能会溢出  
B. 从 int 转换成 double 后, 数值虽然不会溢出, 但有可能是不精确的  
C. 从 double 转换成 float 时, 数值可能会溢出  
D. 从 double 转换成 int 时, 数值不可能溢出
16. 操作系统提供的抽象表示中, ( B ) 是对主存和磁盘 I/O 设备的抽象表示。  
A. 进程      B. 虚拟存储器      C. 文件      D. 虚拟机
17. 给定字长的整数 x 和 y 按补码相加, 和为 s, 则发生正溢出的情况是 ( A )  
A.  $x > 0, y > 0, s \leq 0$       B.  $x > 0, y < 0, s \leq 0$   
C.  $x > 0, y < 0, s \geq 0$       D.  $x < 0, y < 0, s \geq 0$
18. 关于 X86-64 的机器代码、汇编代码与 C 代码, 下列说法错误的是 ( C )  
A. 寄存器对机器代码是可见的, 但对 C 语言是不可见的  
B. C 语言中的聚合数据类型对机器代码而言只是一组连续的字节  
C. 一条 C 指令对应着一条汇编指令  
D. 一条汇编指令对应着一条机器指令
19. 属于异步异常的是 ( A )  
A. 中断      B. 陷阱      C. 故障      D. 终止
20. C 语言程序中的常量表达式的计算是由 ( B ) 完成的  
A. 编辑器      B. 编译器      C. 链接器      D. 加载器

## 二、填空题 ( 每空 1 分, 共 10 分 )

21. 在计算机存储层次结构中, 主存 是磁盘的缓存。
22. 程序所具有的 局部性 特点使得高速缓存能够有效。
23. 若主存地址 32 位, 高速缓存总大小为 2K 行, 块大小 16 字节, 采用 2 路组相连, 则标记位的总位数是 36k 位。(每组 2 行, 共 1k 组, 标记占  $32-4-10=18$  位, 总位数占  $2k \times 18=36k$  位)
24. 若高速缓存的块大小为 B ( $B > 8$ ) 字节, 向量 v 的元素为 int, 则对 v 的步长为 1 的应用的不命中率为  $4/B$ 。
25. CPU 在执行异常处理程序时其模式为 内核模式/超级用户模式。
26. Linux 系统中运行 hello world 这样的 C 程序时, 标准 I/O 函数 printf 实际是通过 系统级 Unix I/O 函数 write, 系统级 I/O、Unix I/O、或 write 函数

都算对\_\_\_\_\_实现的。

27. 若  $x$  和  $y$  的字节值分别为  $0x12$  和  $0x34$ , 则 C 表达式  $x \&\& y$  的值为  
\_\_\_\_\_  $0x01$  \_\_\_\_\_。

28. C 语言程序运行时, 局部变量存在放 栈 段。

29. C 语句中的全局变量, 在 链接/重定位 阶段被定位到一个确定的内存地址。

30. 虚拟内存发生缺页时, 缺页中断是由 MMU 触发的。

### 三、判断对错 (每小题 1 分, 共 10 分, 在题前打 $\checkmark$ X 符号)

31. (  $\checkmark$  ) C 语言中一个有符号整数转换成无符号整数时其位模式是不变的。
32. ( X ) 一个整型机器数采用大端还是小端方式存储, 其值是不同的。
33. (  $\checkmark$  ) 浮点数 IEEE 标准中, 规格化数比非规格化数多。
34. (  $\checkmark$  ) 一个 C 程序中的跳转表数据经链接后被映射到代码段。
35. (  $\checkmark$  ) 使用栈随机化的方法不能完全避免针对缓冲区溢出的攻击。
36. ( X ) c 函数调用过程中, 调用函数的栈帧一旦被修改, 被调用函数则无法正确返回。
37. ( X ) 当执行 fork 函数时, 内核为新进程创建虚拟内存并标记内存区域为私有的写时复制, 意味着新进程此时获得了独立的物理页面。
38. (  $\checkmark$  ) 进程在进行上下文切换时一定会运行内核函数。
39. ( X ) 隐式空闲链表的优点是在对堆块进行搜索时, 搜索时间只与堆中的空闲块数量成正比。
40. (  $\checkmark$  ) 相比标准 I/O, **Unix I/O 函数是异步信号安全的**, 可以在信号处理程序中安全地使用。

### 四、简答题 (每小题 5 分, 共 20 分)

41. 某高速缓存大小 256 字节, 直接映射, 块大小为 16 字节。定义 L 为数据装载命令, S 为存储, M 为数据修改。若每一数据装载 (L) 或存储 (S) 操作可引发最多 1 次缓存缺失 (miss); 数据修改操作 (M) 可认为是同一地址上 1 次装载后跟 1 次存储, 因此可引发 2 次缓存命中 (hit) 或 1 次缺失加 1 次命中外加可能的 1 次淘汰/驱逐 (evict)。根据下列的访存命令序列, 分析每一命令下的上述高速缓存 (高速缓存最初是空的) 的命中及淘汰情况。

L 10, 1

M 20, 1

L 22, 1

S 18, 1

L 110, 1

L 210, 1

M 12, 1

说明: L 10, 1 表示从地址  $0x10$  处加载 1 个字节数据, 其它同理。

L 10, 1 miss //10 是  $0x10$ , 块地址  $0x1 \bmod 0x10 = 1$  (组) , 第 1 块映射

## 到第 1 组

M 20, 1, miss hit//块地址  $0x2 \bmod 0x10 = 2$  (组) , 第 2 块映射到第 2 组

L 22, 1 hit//块地址  $0x2 \bmod 0x10 = 2$  (组), 第 2 块映射到第 2 组

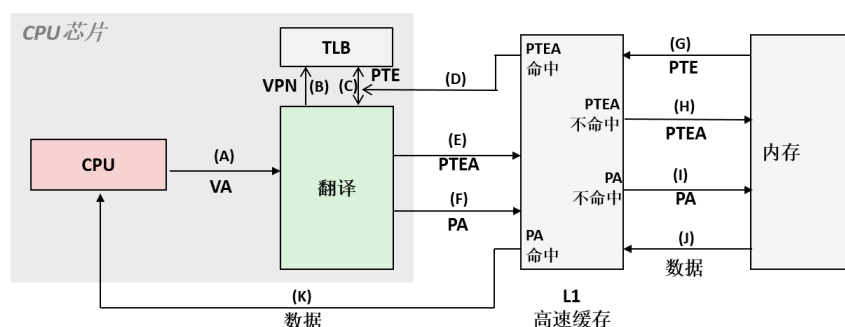
S 18, 1 hit //块地址  $0x1 \bmod 0x10 = 1$  (组), 第 1 块映射到第 1 组

L 110, 1 miss eviction//块地址  $0x11 \bmod 0x10 = 1$  (组), 第 17 块映射到第 1 组

L210, 1 miss eviction//块地址  $0x21 \bmod 0x10 = 1$  (组), 第 33 块映射到第 1 组

M 12, 1 miss eviction//块地址  $0x1 \bmod 0x10 = 1$  (组), 第 1 块映射到第 1 组

42. 下图展示了一个虚拟地址的访存过程，每个步骤采用不同的字母表示。请分别针对下述情况，用字母序列写出每种情况下的执行流程：（1）TLB 命中、缓存物理地址命中；（2）TLB 不命中，缓存页表命中，缓存物理地址命中；（3）TLB 不命中，缓存页表不命中，缓存物理地址不命中。



(1) A → B → C → F → K

(2) A → B → E → D → F → K

(3) A → B → E → H → G → (D) → F → I → J → (K) ; D, K 缺少也正确

43. 列举至少 5 种程序优化的方法。

代码移动，通过将代码从循环中移出减少计算执行的频率

用简单计算替代复杂计算/操作，如移位、加替代乘法/除法

共享共用子表达式、重用表达式的一部分

使用局部变量作为累积量

循环体展开减少循环次数

通过多个累积量、重新结合（组合）的方法，提高指令级并行性

尽量缩短关键路径（利于流水操作）

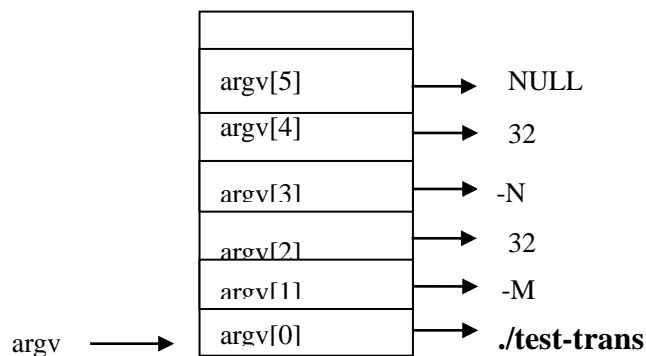
减少过程调用，小函数使用 inline 形式声明

用条件表达式替代 if-else 语句（用功能性的风格重写条件操作，有利于编译器采用条件数据传送实现，而非使用分支结构实现）

消除不必要的内存引用

使用速度更快的 CPU 指令，例如 SSE 等 SIMD 指令。

44. 在 shell 命令行输入命令：Ubuntu>./test-trans -M 32 -N 32[回车]  
shell 命令行解释器将构造参数 argv 和 envp，请写出参数 argv 的内容。



### 五、系统分析题（每小题 5 分，共 20 分）

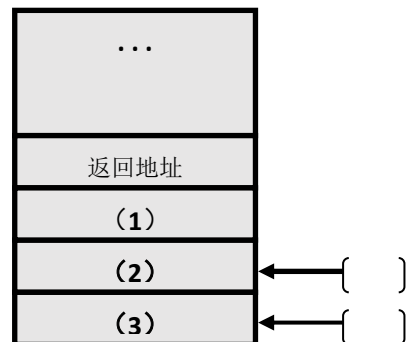
45. 函数 call\_incr2 的汇编代码如下所示，画出函数 call\_incr2 相应的栈帧内结构与内容。

call\_incr2:

```

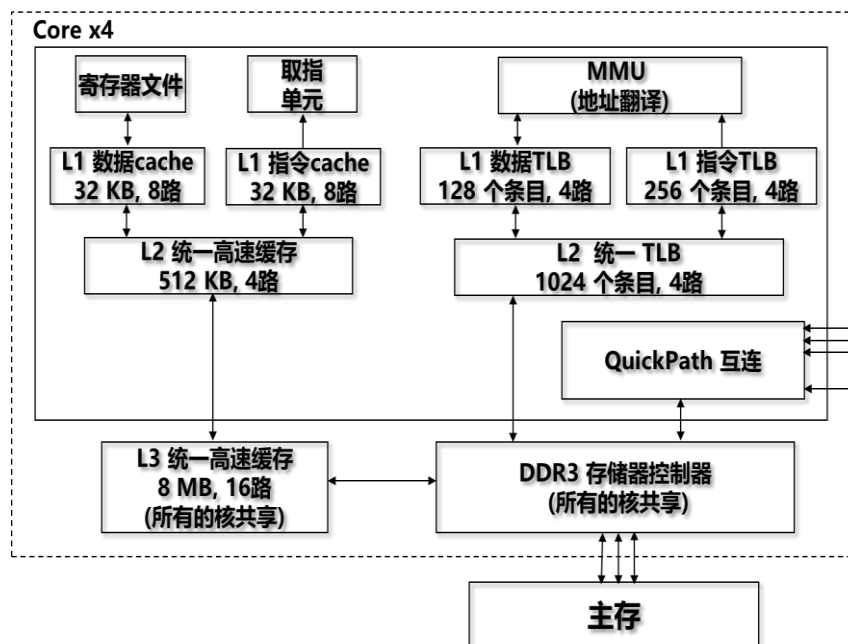
pushq    %rbx
subq     $16, %rsp
movq     %rdi, %rbx
movq     $15213, 8(%rsp)
movl     $3000, %esi
leaq     8(%rsp), %rdi
call     incr
addq     %rbx, %rax
addq     $16, %rsp
popq     %rbx
ret

```



请填写出上述栈帧缺失的内容

- (1) 保存的%rbx
- (2) 15213
- (3) 未用
- (4) %rsp+8
- (5) %rsp



46. Intel I7 CPU 的虚拟地址 48 位，虚拟内存的每一页面 4KB，物理地址 52 位，cache 块大小 64B，物理内存按字节寻址。其内部结构如下图所示，依据此结构，分析如下项目：

某指令 A 的虚拟地址为 0x804849b，则该地址对应的 VP0 为 0x 49b；访问 L1 TLB 的 TLBI 为 0x 08 (6 位)；

若指令 A 的物理地址为 0x86049b，则该地址对应的 PPN 为

0x 860 (40 位)；访问 L1 cache 的 CT 为 0x 860 (40 位)，CO 为 0x1b (6 位)。

47. 设一个 C 语言源程序 p.c 编译链接后生成执行程序 p，反汇编如下：

C 程序

反汇编程序的 main 部分（还有系统代码）如下：

```
#include <stdio.h>
unsigned short b[2500];
unsigned short k;
void main()
{
    b[1000] = 1023;
    b[2000] = 2049*k;
    b[10000] = 20000;
}
```

main 的地址为 0x80482C0 (short 占 2 字节)

```
1  movw    $0x3ff, 0x80497d0
2  movw    0x804a324, %cx      ;k->cx
3  mov     $0x801, %eax
4  xorw    %dx, %dx
5  div     %ecx                ;2049/d
6  movw    %dx, 0x804a324
7  movw    $0x4e20,0x804de20
8  ret
```

现代 Intel 桌面系统，采用虚拟页式存储管理，每页 4KB，p 首次运行时系统中无其他进程。请结合进程与虚拟存储管理的知识，分析上述程序的执行过程中：

- (1) 在取指令时发生的缺页异常次数为 0。
- (2) 写出已恢复的故障指令序号与故障类型 1、2；缺页故障 (page fault)。
- (3) 写出没有恢复的故障指令序号与故障类型 7；保护违例或“段错误”或段故障 (segmentation fault)。

(1) 指令位于起始地址 0x8048000 的页面 (4k 大小)，执行指令前，该页面已调入内存。系统无其他进程，不会发生因执行其他进程而使该页面调出到磁盘。故不会发生缺页故障。

(2) 1, 2；对数组 b[1000] 即 0x80497d0 的访问发生缺页，对数组 b[2000] 即地址 0x804a324 (另一个页面) 访问也发生缺页故障，6 不发生缺页。



(3) 7, 访问 `b[10000]` 时地址 `0x804de20` 可能超出可读写范围, 发生保护违例。

49. 考虑下面的程序, 它由两个模块组成:

```
/*main.c*/
#include <stdio.h>
int x = 100;
int y;
void p1(void);
int main()
{
    int z=0;
    p1();
    y = 2000;
    printf("x=%d,y=%d\n",x,y);
    return 0;
}
```

```
/*p1.c*/
int x;
int y;
void p1()
{
    x=1000;
    y=200;
}
```

请指出 `main.o` 中属于强符号的是? x, main  
 程序最后的输出是什么? x=1000, y=2000

## 六、综合设计题 (每小题 10 分, 共 20 分)

50. Y86-64 的部分指令序列如下图所示, 写出 Y86-64 顺序结构 CPU 中, 执行指令 `rmmovq %rsp, 100(%rbx)` 时各阶段的操作及操作的具体结果。

1	0x000:30f30900000000000000		<code>irmovq \$9,%rbx</code>
2	0x00a:30f41500000000000000		<code>irmovq \$21,%rsp</code>
3	0x014:40436400000000000000		<code>rmmovq %rsp,100(%rbx)</code>
...	...		.....

(提示: 指令 `rmmovq rA, D(rB)` 的编码规则如下)

字节 0		字节 1		字节 2...9
4	0	rA	rB	D

取指  $icode: ifun \leftarrow M_1[PC=0x014] = 4:0$   
 $rA:rB \leftarrow M_1[0x015] = 4:3$   
 $valC \leftarrow M_8[0x016] = 100 (0x00..64),$   
 $valP \leftarrow PC + 10 = 0x014 + 0xa = 0x01e$

译码  $valA \leftarrow R[\%rsp] = 21,$   
 $valB \leftarrow R[\%rbx] = 9$

执行  $valE \leftarrow valB + valC = 9 + 100 = 109$

访存  $M_8[valE] = M_8[109] \leftarrow valA = 21$

写回

更新 PC  $PC \leftarrow valP = 0x01e$ . 指令将 21 写入地址 109 处, PC 加 10

51. 一段 C 语言程序如下:

```
#include "csapp.h"
int counter = 0;
jmp_buf buf;
void handler_alarm(int sig){
}
void handler_usr1(int sig) {
    counter +=1;
    siglongjmp(buf,1);
    counter +=2;
}
int main(void)
{
    signal(SIGUSR1, handler_usr1);
    signal(SIGALRM, handler_alarm);
    if (!sigsetjmp(buf,1)) {
        sleep(10);
        printf("A");
        counter +=3;
    } else {
        printf("B");
    }
    exit(0);
}
```

假设: 上述 C 语言程序运行后, 进程 ID 为 12345, 且程序执行完语句 sigsetjmp 后收到信号, 且 printf()不会被信号中断。

(1) 如另一个程序给正在运行的进程 12345 发送了 1 个 SIGALRM 信号, 屏幕输出是什么? 在退出 main 函数之前一刻, 变量 counter 的数值是多少?

输出: A

counter 的数值是: 3

(2) 如另一个程序给正在运行的进程 12345 发送了 1 个 SIGUSR1 信号, 可能的屏幕输出有哪些, 在退出 main 函数之前一刻, 变量 counter 的数值是多少, 并对每种情况作出解释。

输出: B

counter 的数值是: 1

输出: AB

答案错误

counter 的数值是: 3

输出: AB

counter 的数值是: 1

输出: AB

counter 的数值是: 4