ICS 2015 期中试题

## 第一题 单项选择题（每小题1分，共20分）

注：请将选择题的回答填写在下表中

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 题号 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 回答 |  |  |  |  |  |  |  |  |  |  |
| 题号 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 回答 |  |  |  |  |  |  |  |  |  |  |

1. **给定一个实数，会因为该实数表示成单精度浮点数而发生误差。不考虑NaN和Inf的情况，该绝对误差的最大值为:**

**A. B. C. D.**

参考信息：单精度浮点数阶码8位，尾数23位

1. **以下说法错误的是:**
   1. **负数加上负数结果可能为正数**
   2. **正数加上正数结果可能为负数**
   3. 用&和~可以表示所有的逻辑与或非操作
   4. **用&和|可以表示所有的逻辑与或非操作**
2. **在32位平台上，按C90标准以下语句中，结果为假的是:**
   1. **return INT\_MIN < INT\_MAX;**
   2. return -2147483648 < 2147483647;
   3. **int a =** -2147483648; return a < 2147483647;
   4. **return -**2147483647-1 < 2147483647;

参考信息：

C90的转换顺序：int -> long -> unsigned -> unsigned long

1. **关于浮点数，以下说法正确的是:**
   1. **给定任意浮点数a，b和x，如果a>b成立（求值为1），则一定a+x>b+x成立**
   2. **不考虑结果为NaN、Inf或运算过程发生溢出的情况，高精度浮点数一定得到比低精度浮点数更精确或相同的结果**
   3. **不考虑输入为NaN、Inf的情况，高精度浮点数一定得到比低精度浮点数更精确或相同的结果**
   4. **给定任意浮点数a, b和x，如果a>b不成立（求值为0），则一定a+x>b+x不成立。**
2. 已知下面的数据结构，假设在Linux/IA32下要求对齐，这个结构的总的大小是多少个字节？如果重新排列其中的字段，最少可以达到多少个字节？

struct {

char a;

double \*b;

double c;

short d;

long long e;

short f;

};

A. 32, 28 B. 36, 32 C. 28, 26 D. 26, 26

1. 下列寻址模式中，正确的是：

A. (%eax, , 4) B. (%eax, %esp, 3)

C. 123 D. $1(%ebx, %ebp, 1)

1. 假设存储器按“大端法”存储数据对象，已知如下的C语言数据结构：union { char c[2]; int i; }; 当c的值为0x01, 0x23时，i的值为：

A. 0x0123 B. 0x2301 C. 0x01230000 D.不确定

1. 假设某条C语言switch语句编译后产生了如下的汇编代码及跳转表：

|  |  |
| --- | --- |
| movl 8(%ebp), %eax  subl $48, %eax  cmpl $8, %eax  ja .L2  jmp \*.L7(, %eax, 4) | .L7:  .long .L3  .long .L2  .long .L2  .long .L5  .long .L4  .long .L5  .long .L6  .long .L2  .long .L3 |

在源程序中，下面的哪些（个）标号出现过：

A. ‘2’, ‘7’

B. 1

C. ‘3’

D. 5

1. x86体系结构中，下面哪个说法是正确的？

A. leal指令只能够用来计算内存地址

B. x86\_64机器可以使用栈来给函数传递参数

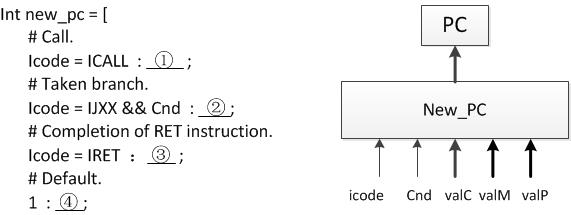
C. 在一个函数内，改变任一寄存器的值之前必须先将其原始数据保存在栈内

D. 判断两个寄存器中值大小关系，只需要SF和ZF两个条件码

1. 下列的指令组中，哪一组指令只改变条件码，而不改变寄存器的值？

A. CMP, SUB   
B. TEST, AND   
C. CMP, TEST   
D. LEAL, CMP

1. 下面有关指令系统设计的描述正确的是：
2. 采用CISC指令比RISC指令代码更长。
3. 采用CISC指令比RISC指令运行时间更短
4. 采用CISC指令比RISC指令译码电路更加复杂
5. 采用CISC指令比RISC指令的流水线吞吐更高
6. 一个功能模块包含组合逻辑和寄存器，组合逻辑单元的总延迟是100ps，单个寄存器的延时是20ps，该功能模块执行一次并保存执行结果，理论上能达到的最短延时和最大吞吐分别是多少？
7. 20ns, 50GIPS
8. 120ns，50GIPS
9. 120ns，10GIPS
10. 20ps,10GIPS
11. 关于流水线技术的描述，错误的是：
12. 流水线技术能够提高执行指令的吞吐率，但也同时增加单条指令的执行时间。
13. 减少流水线的级数，能够减少数据冒险发生的几率。
14. 指令间数据相关引发的数据冒险，都可以通过data forwarding来解决。
15. 现代处理器支持一个时钟内取指、执行多条指令，会增加控制冒险的开销。
16. 在Y86的SEQ实现中，PC（Program Counter，程序计数器）更新的逻辑结构如下图所示，请根据HCL描述为①②③④选择正确的数据来源。



其中：Icode为指令类型，Cnd为条件是否成立，valC表示指令中的常数值，valM表示来自返回栈的数据，valP表示PC自增。

1. valC, valM, valP, valP
2. valC, valC, valP, valP
3. valC, valC, valM, valP
4. valM, valC, valC, valP
5. 下面关于存储器的说法，错误的是\_\_\_\_
6. SDRAM的速度比FPM DRAM快
7. SDRAM的RAS和CAS请求共享相同的地址引脚
8. 磁盘的寻道时间和旋转延迟大致在一个数量级
9. 固态硬盘的随机读写性能基本相当
10. 某磁盘的旋转速率为 7200 RPM，每条磁道平均有400扇区，则一个扇区的平均传送时间为\_\_\_\_
11. 0.02 ms
12. 0.01 ms
13. 0.03 ms
14. 0.04 ms
15. 某高速缓存 （E=2, B=4, S=16），地址宽度14，当引用地址0x9D28处的1个字节时，tag位应为：
16. 01110100
17. 001110000
18. 1110000
19. 10011100
20. 关于高速缓存的说法正确的是 \_\_\_\_\_
21. 直写（write through）比写回（write back）在电路实现上更复杂
22. 固定的高速缓存大小，较大的块可提高时间局部性好的程序的命中率
23. 随着高速缓存组相联度的不断增大，失效率不断下降
24. 以上说法全不正确
25. 关于局部性（locality）的描述，不正确的是：

A. 循环通常具有很好的时间局部性

B. 循环通常具有很好的空间局部性

C. 数组通常具有很好的时间局部性

D. 数组通常具有很好的空间局部性

1. 以下哪些程序优化编译器总是可以自动进行？（假设int i, int j, int A[N], int B[N], float m都是局部变量，N是一个整数型常量，int foo(int) 是一个函数）

|  |  |  |
| --- | --- | --- |
|  | 优化前 | 优化后 |
| A. | for (j = 0 ; j < N ; j ++)  m + = i\*N\*j; | int temp = i\*N;  for (j= 0 ; j < N ; j ++)  m + = temp \* j; |
| B. | for (j = 0 ; j < N ; j ++)  B[i] \*= A[j]; | int temp = B[i];  for (j= 0 ; j < N ; j ++)  temp \*= A[j];  B[i] = temp; |
| C. | for (j = 0 ; j < N ; j ++)  m = (m + A[j]) + B[j]; | for (j = 0 ; j < N ; j ++)  m = m + (A[j] + B[j]); |
| D. | for (j = 0 ; j < foo(N) ; j ++)  m++; | int temp = foo(N);  for (j= 0 ; j < temp ; j ++)  m++; |

|  |
| --- |
| 得分 |
|  |

## **第二题（20分）**

**1.对于下面的每一个表达式，请选择以下选项中的一个或多个（即“不定项”），**使得该表达式恒成立，如果没有满足条件的选项则选E。

1. < B. > C. == D. != E. none

题目中出现的变量定义如下（浮点数保证不是NaN或者Inf）:

int x, y;

unsigned ux = x;

double d;

1. 如果x > 0, 则 x + 1 0
2. 如果 x > y, 则 ux y
3. 如果 ((x << 31) >> 31) < 0, 则x & 1 0
4. 如果((unsigned char)x >> 1) < 64， 则(char)x 0
5. 如果d < 0， 则d \* 2 0
6. 如果d < 0, 则 d \* d 0
7. x ˆ y ˆ (˜x) - y y ˆ x ˆ (˜y) - x
8. (((!!ux)) << 31) >> 31) (((!!x) << 31) >> 31)

2.考虑一种12-bit长的浮点数，此浮点数遵循IEEE浮点数的格式原理，其字段划分如下：

符号位（s）：1-bit；阶码字段（exp）：4-bit；小数字段（frac）：7-bit。

1. 请写出在下列区间中包含多少个用上面规则精确表示的浮点数

A： [1,2)

B: [2,3)

1. 请写出下面浮点数的二进制表示

|  |  |
| --- | --- |
| 数字 | 二进制表示 |
| 最小的正规格化数 |  |
| 最大的非规格化数 |  |
| 17 |  |
| - |  |
| 20 |  |
| -∞ |  |

|  |
| --- |
| 得分 |
|  |

## **第三题（20分）**

1.考虑下面的union的声明，回答后面的问题。

union ELE {

struct {

int x;

int \*p;

}e1;

struct {

union ELE \* next;

int y;

}e2;

};

(注：32位机器)

这个union总共大小为多少\_\_\_\_字节。

2.假设编译器为process的主体产生了如下了代码，请补充完整下面的过程:

（只有一个不需要任何强制类型转换且不违反任何类型限制的答案）

movl 8(%ebp),%eax

movl (%eax),%ecx

movl 4(%ecx),%edx

movl (%edx),%edx

subl 4(%eax),%edx

movl %edx,(%ecx)

void process(union ELE \* up)

{

up->\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_=\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ - \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;

}

3.请查看下文完成如下功能的汇编代码，定位错误语句并进行更正:

给出 n(在%ebp+8位置,n>=1),up(在%ebp+12位置,ELE\* 类型),假设以\*up为头元素(设\*up为第0个),由声明中的next连接形成了一个链表,请将第n个元素(假设链表足够长)的x的值放入%eax中

 X86代码

xorl %ecx,%ecx

movl 8(%edx),%ebp

movl 12(%ebp),%eax

LOOP:

movl (%eax),%eax

add $1,%ecx

test %ecx,%edx

jne LOOP

movl (%eax),%eax

Y86代码

mrmovl 8(%edx),%ebp

mrmovl 12(%ebp),%eax

irmovl $1,%ecx

LOOP:

mrmovl (%eax),%eax

test %ecx,%edx

jne LOOP

mrmovl (%eax),%eax

4.阅读下列代码，回答后面的问题

typedef struct {

short x[A][B];

int y;

}str1;

typedef struct {

char array[B];

int t;

short s[B];

int u;

}str2;

void setVal(str1 \*p,str2 \*q) {

int v1=q->t;

int v2=q->u;

p->y=v1+v2;

}

(short以2字节计算)

GCC为setVal的主体产生下面的代码:

movl 12(%ebp),%eax

movl 28(%eax),%edx

addl 8(%eax),%edx

movl 8(%ebp),%eax

movl %edx,44(%eax)

请直接写出A和B的值各是多少?

|  |
| --- |
| 得分 |
|  |

## 第四题（20分）

请分析Y86 ISA中新加入的一条指令：NewJE，其格式如下。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| NewJE | C | 0 | rA | rB | Dest |

其功能为：如果R[rA]= R[rB]，则跳转到Dest继续执行，否则顺序执行。

1. 若在教材所描述的SEQ处理器上执行这条指令，请按下表补全每个阶段的操作。需说明的信号可能会包括：icode, ifun, rA, rB, valA, valB, valC, valE, valP, Cnd；the register file R[], data memory M[], Program counter PC, condition codes CC。其中对存储器的引用必须标明字节数。如果在某一阶段没有任何操作，请填写none指明。

|  |  |
| --- | --- |
| Stage | NewJE rA, rB, Dest |
| Fetch | icode:ifun ← M1[PC]  rA:rB ← M1[PC+1]  valC ←  valP ← |
| Decode | valA ← R[rA]  valB ← R[rB] |
| Execute | valE ← |
| Memory |  |
| Write back | none |
| PC update | PC ← valE==0 ? : |

2. 若在教材所描述的PIPE处理器上执行NewJE指令，如果跳转条件不满足，一共会错误执行\_\_\_\_条指令。

为了减小错误预测的代价，现将教材所描述的PIPE处理器做如下改进：在Decode阶段增加一个比较器，用于判断（R[rA] = R[rB]）条件，比较器的输出信号为d\_equal。如果相等，则d\_equal = 1，反之 d\_equal = 0。

此时，如果执行NewJE指令时跳转条件不满足，一共会错误执行\_\_\_\_条指令。

3.在教材所描述的PIPE处理器上执行JXX指令时，发生转移预测错误的判断条件和各级流水线寄存器的控制信号如下所示：

|  |  |
| --- | --- |
| Condition | Trigger |
| Mispredicted Branch | E\_icode = IJXX & !e\_Cnd |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Condition | F | D | E | M | W |
| Mispredicted Branch | normal | bubble | bubble | normal | normal |

在第（2）小题所述的改进后的处理器上执行NewJE指令，发生转移预测错误的判断条件和各级流水线寄存器的控制信号应如何设置？

|  |  |
| --- | --- |
| Condition | Trigger |
| Mispredicted Branch | = INewJE |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Condition | F | D | E | M | W |
| Mispredicted Branch |  |  |  | normal | normal |

4.在第（2）小题所述的改进后的处理器上执行如下代码，

0x000: mrmovl 0(%eax), %edx

0x006: NewJE %edx, %eax, t

0x00c: irmovl $1, %eax # Fall through

0x012: nop

0x013: nop

0x014: nop

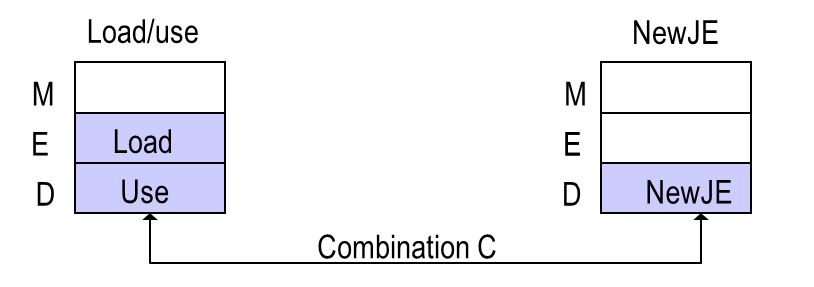
0x015: halt

0x016: t:irmovl $3, %edx # Target (Should not execute)

0x01c: irmovl $4, %ecx # Should not execute

0x022: irmovl $5, %edx # Should not execute

当发生load-use 和misprediction组合的hazard情况, 如下图所示



请问此时，各级流水线寄存器的控制信号应如何设置？

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Condition | F | D | E | M | W |
| Combination C |  |  |  | normal | normal |

5. 在教材PIPE处理器设计中，data memory实际是高速缓存(cache)。假设在执行上述(4)中代码时，0x000指令中的0(%eax)地址中的数据不在data memory 中，则data memory会将输出信号 m\_datamiss置为1，直到数据从内存中取回到data memory，再将 m\_datamiss置为0。（m\_datamiss的默认值为0）。这种情况的判断条件如下，请问各级流水线寄存器的控制信号应如何设置？

|  |  |
| --- | --- |
| Condition | Trigger |
| Data Miss | M\_icode in { IMRMOVL, IPOPL } && m\_datamiss |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Condition | F | D | E | M | W |
| data miss |  |  |  |  |  |

|  |
| --- |
| 得分 |
|  |

## **第五题（20分）**

1.仔细阅读下面的程序，根据条件回答下列各题

* 地址宽度为7，数组的起始地址为0x1000000
* Block Size = 4 Byte，Set = 4，两路组相连.
* 替换算法位LRU（最近最少使用）

#define LENGTH 8

void clear4x4 ( char array[LENGTH][LENGTH] ) {

int row, col ;

for ( col = 0 ; col < 4; col++ ) {

for ( row = 0; row < 4 ; row++ ) {

array[ row] [ col] = 0;

}

}

}

1. 以上程序执行会引起多少次失效？
2. 如果LENGTH改为16，会引起多少次失效？
3. 如果LENGTH变为17，与2)相比，下面描述正确的是：（ ）。  
   此时会引起 次失效。
4. 16×16 比17×17产生更多的失效次数
5. 16×16 和17×17产生的失效次数相同
6. 16×16 比17×17产生更少的失效次数
7. 请画出3）运行后cache中set0和set1的最终状态。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| V | Tag | Data | V | Tag | Data |
| 1 |  |  | 1 |  |  |
| 1 |  |  | 1 |  |  |

2.改变假设条件，回答下列各题

* 地址宽度为8，数组的起始地址为0x10000000
* Cache容量为16 Byte，Block Size = 4 Byte，全相联Cache
* 替换算法位LRU（最近最少使用）

1. Tag的位数为
2. 如果执行上述程序，当LENGTH=8时，会引起多少次失效？

1. 如果LENGTH改为16，会引起多少次失效？

1. 如果LENGTH变为17，与3)相比，下面描述正确的是：（ ）
2. 16×16 比17×17产生更多的失效次数
3. 16×16 和17×17产生的失效次数相同
4. 16×16 比17×17产生更少的失效次数
5. 请画出4）执行后cache的最终状态

|  |  |  |
| --- | --- | --- |
| V | Tag | Data |
| 1 |  |  |
| V | Tag | Data |
| 1 |  |  |
| V | Tag | Data |
| 1 |  |  |
| V | Tag | Data |
| 1 |  |  |