北京大学信息科学技术学院考试试卷

考试科目：计算机系统导论 **姓名：**_____ **学号：**_____

考试时间：2017 年 11 月 6 日**小班号：**___**小班教师：**_____

题号	一	二	三	四	五	**总分**
分数						
阅卷人						

装订线内

不要答题

北京大学考场纪律

1、考生进入考场后，按照监考老师安排隔位就座，将学生证放在桌面上。无学生证者不能参加考试；迟到超过 15 分钟不得入场。在考试开始 30 分钟后方可交卷出场。

2、除必要的文具和主考教师允许的工具书、参考书、计算器以外，其它所有物品（包括空白纸张、手机、或有存储、编程、查询功能的电子用品等）不得带入座位，已经带入考场的必须放在监考人员指定的位置。

3、考试使用的试题、答卷、草稿纸由监考人员统一发放，考试结束时收回，一律不准带出考场。若有试题印制问题请向监考教师提出，不得向其他考生询问。提前答完试卷，应举手示意请监考人员收卷后方可离开；交卷后不得在考场内逗留或在附近高声交谈。未交卷擅自离开考场，不得重新进入考场答卷。考试结束时间到，考生立即停止答卷，在座位上等待监考人员收卷清点后，方可离场。

4、考生要严格遵守考场规则，在规定时间内独立完成答卷。不准交头接耳，不准偷看、夹带、抄袭或者有意让他人抄袭答题内容，不准接传答案或者试卷等。凡有违纪作弊者，一经发现，当场取消其考试资格，并根据《北京大学本科考试工作与学术规范条例》及相关规定严肃处理。

5、考生须确认自己填写的个人信息真实、准确，并承担信息填写错误带来的一切责任与后果。

学校倡议所有考生以北京大学学生的荣誉与诚信答卷，共同维护北京大学的学术声誉。

以下为试题和答题纸，共　　页。

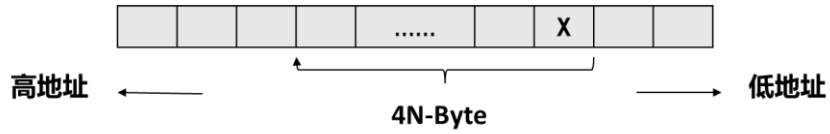
得分

第一题 单项选择题（每小题 2 分，共 30 分）

注：选择题的回答请填写在下表中。

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. 假定一个特殊设计的计算机，将 int 型数据的长度从 4-Byte 扩展为 4N-Byte，采用大端法（Big Endian）。现将该 int 型所能表示的最小负数写入内存中，如下图所示。其中每个小矩形代表一个 Byte，请问 X 位置这个 Byte 中的值是多少？



A. 00000000_2　B. 01111111_2　C. 10000000_2　D. 11111111_2

答案：C

解析：最小的负数为 1000000…，大端法将最高位放到最低地址

2. 以下说法正确的是：

A. 负数加上负数结果都为负数

B. 正数加上正数结果都为正数

C. 用&和~可以表示所有的逻辑与或非操作

D. 用&和|可以表示所有的逻辑与或非操作

答案：C

3. 若我们采用基于 IEEE 浮点格式的浮点数表示方法，阶码字段（exp）占据 k 位，小数字段（frac）占据 n 位，则最小的规格化（normalized）的正数是_____（结果用含有 n，k 的表达式表示）

A. $(1-2^{-n})*2^{-2^{k-1}+2}$　　B. $1*2^{-2^{k-1}+2}$

C. $2^{-n} * 2^{-2^{k-1}+2}$　　D. $(1-2^{-n}) * 2^{-2^{k}+1}$

答案： B

解析:阶码字段 exp:00...01;小数字段 frac:00...00;偏置值 bias: $2^{k-1}-1$

4. 在下列的 x86-64 汇编代码中，错误的是：

A. movq %rax, (%rsp)　　B. movl $0xFF, (%ebx)

C. movsbl (%rdi), %eax　　D. leaq (%rdx, 1), %rdx

答案：B/D 都对

解析：B:地址寄存器必须是 64 位寄存器;D:括号里少了一个逗号

5. 在下列关于条件传送的说法中，正确的是：

A. 条件传送可以用来传送字节、字、双字、和 4 字的数据

B. C 语言中的“?:”条件表达式都可以编译成条件传送

C. 使用条件传送总可以提高代码的执行效率

D. 条件传送指令不需要用后缀（例如 b, w, l, q）来表明操作数的长度

答案：D

解析：A.条件传送不支持单字节传送；B.如果“?:”涉及到的两个表达式中有一个出错或者有副作用，用条件传送会导致非法行为；C.如果被旁路的分支的计算量很大，计算就白做了；D.从目标寄存器的名字可以推断出条件传送指令的操作数长度

6. 有如下代码段：

```
int func(int x, int y);
int (*p) (int a, int b);
p = func;
p(0,0);
```

对应的下列 x86-64 过程调用正确的是：

A. call *%rax　　B. call (%rax)

C. call *(%rax)　　D. call func

答案：A/D 都对

解析：用 O0 编译可以得到 A，用 Og 编译可以得到 D。

7. 有 A 的定义：int A[3][2] = {{1,2}, {3,3}, {2,1}};

那么 A[2]的值为：

A. &A+16　　B. A+16　　C. *A+4　　D. *A+2

答案：C

解析：参见书 P177 页表格，机器在计算指针与常数的运算时，会将常数乘以指针指向的元素大小。&A 常数扩大的倍数为 sizeof(A[3][2]) = 3*2*4; A 常数扩大的倍数为 sizeof(A[0])=2*4; *A 常数扩大的倍数为 sizeof(int) = 4。正确的答案应为 A+2 或*A+4，故应选择 C。

程序验证如下：

```
int main(){
    int A[3][2] = {{1,2},{3,3},{2,1}};
    cout << "&A+16: " << &A+16 << endl;
    cout << "A+16: " << A+16 << endl;
    cout << "*A+4: " << *A+4 << endl;
    cout << "*A+2: " << *A+2 << endl;
    cout << A[2] << endl;
}
```

```
&A+16: 0xb5fee0
A+16: 0xb5fde0
*A+4: 0xb5fd70
*A+2: 0xb5fd68
0xb5fd70
```

8. 在 x86-64 架构下，有如下变量：

```
union {char c[8], int i;} x;
```

在 x.i=0x41424344 时，x.c[2]的值为多少（提示：'A'=0x41）：

A. 'A'　　B. 'B'　　C. 'C'　　D. 'D'

答案：B

9. 下面关于 RISC 和 CISC 的描述中，正确的是：

A. CISC 和早期 RISC 在寻址方式上相似，通常只有基址和偏移量寻址

B. CISC 指令集可以对内存和寄存器操作数进行算术和逻辑运算，而 RISC 只能寄存器操作数进行算术和逻辑运算

C. CISC 和早期的 RISC 指令集都有条件码，用于条件分支检测

D. CISC 机器中的寄存器数目较少，函数参数必须通过栈来进行传递；RISC 机器中的寄存器数目较多，只需要通过寄存器来传递参数，避免了不必要的存储访问

答案：B

10. 在 Y86 的 SEQ 实现中，对仅考虑 IRMMOVQ，ICALL，IPOPQ，IRET 指令，对 mem_addr 的 HCL 描述正确的是：

```
word mem_addr = [
    icode in { (1), (2) } : valE;
    icode in { (3), (4) } : valA;
];
```

A. (1) IRMMOVQ (2) IPOPQ (3) IRET (4) ICALL
B. (1) IRMMOVQ (2) IRET (3) IPOPQ (4) ICALL
C. (1) ICALL (2) IPOPQ (3) IRMMOVQ (4) IRET
D. (1) IRMMOVQ (2) ICALL (3) IPOPQ (4) IRET

答案： D

11. 关于流水线技术的描述，错误的是：

A. 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

B. 增加流水线级数，不一定能获得总体性能的提升

C. 指令间数据相关引发的数据冒险，不一定可以通过暂停流水线来解决。

D. 流水级划分应尽量均衡，吞吐率会受到最慢的流水级影响，均衡的流水线能提高吞吐量。

答案：C

12. 假设已有声明 int i, int j, const int n, int r, int a[n], int b[n], int mul(int, int)以下程序优化编译器一般不会进行的是：

	优化前	优化后

A.	`for (j = 0; j < n; ++j)`
	`  a[n * 8] += b[j];`
		`int tmp = (n << 3);`
		`for (j = 0; j < n; ++j)`
		`  a[tmp] += b[j];`

B.	`for (j = 0; j < n; ++j)` `  r = (r * a[j]) * b[j];`	`for (j = 0; j < n; ++j)` `  r = r * (a[j] * b[j]);`

C.	`for (j = 0; j < n; ++j)`
	`  a[mul(n, i) + j] = b[j];`
		`int ni = mul(n, i);`
		`for (j = 0; j < n; ++j)`
		`  a[ni + j] = b[j];`

D.	`for (j = 1; j < n; ++j)`
	`  a[0] += a[j];`
		`int tmp = 0;`
		`for (j = 1; j < n; ++j)`
		`  tmp += a[j];`
		`a[0] += tmp;`

答案：C

13．以下计算机部件中，通常不用于存储器层次结构（Memory Hierarchy）的是：

A．高速缓存　　B．内存　　C．硬盘　　D．优盘（U 盘）

答案：D

14．关于局部性（locality）的描述，正确的是：

A．数据的时间局部性或数据空间局部性，在任何有意义的程序中都能体现

B．指令的时间局部性或数据空间局部性，在任何有意义的程序中都能体现

C．数据的时间局部性，在任何循环操作中都能体现

D．数据的空间局部性，在任何数组操作中都能体现

答案：B

注：B 选项有笔误（“数据空间局部性”应为“指令空间局部性”），所以所有答案都不对。按任何选项都对计分。

15．在高速缓存存储器中，关于全相联和直接映射结构，以下论述正确的是：

A．如果配备同样容量、技术的高速缓存，配备全相联高速缓存的计算机总是比配备直接映射高速缓存的计算机性能低

B．如果配备同样容量、技术的高速缓存，配备全相联高速缓存的计算机总是比配备直接映射高速缓存的计算机性能高

C．如果配备同样容量、技术的高速缓存，当数据在缓存中时，配备全相联高速缓存的计算机总是比配备直接映射高速缓存的计算机读数据慢

D．如果配备同样容量、技术的高速缓存，当数据在缓存中时，配备全相联高速缓存的计算机总是比配备直接映射高速缓存的计算机读数据快

答案：C

得分

第二题（15 分）

考虑有一种基于 IEEE 浮点格式的 9 位浮点表示格式 A。格式 A 有 1 个符号位，k 个阶码位，n 个小数位。现在已知$\frac{-9}{16}$的位模式可以表示为“101100010”，请回答以下问题：（注：阶码偏移量为$2^{k-1}-1$）

1． 求 k 和 n 的值。（1 分）

答案：k = 4，n = 4

2． 基于格式 A，请填写下表。值的表示可以写成整数（如 16），或者写成分数（如 17/64）。（注:每格 2 分）

描述	二进制位表示	值
最大的非规格化数	0 0000 1111	15/1024
最小的正规格化数	0 0001 0000	1/64
最大的规格化数	0 1110 1111	248

3． 假设格式 A 变为 1 个符号位，k+1 个阶码位，n-1 个小数位，那么能表示的实数数量会怎样变化，数值的精度会怎样变化？（回答增加、降低或不变即可）（2 分）

答案：小数位变少后，NaN 的数量减少了，所以实数数量增加（1 分），数值精度降低（1 分）。

得分

第三题（15 分）

分析下面C语言程序和相应的x86-64汇编程序。其中缺失部分代码（被遮挡），请在对应的横线上填写缺失的内容。（注：每格1分）

```
#include <stdio.h>
#include "string.h"

void myprint(char *str)
{
    char buffer[16];
    [redacted](buffer,str);                    ①strcpy
    printf("%s \n",buffer);
}

void alert(void)
{
    printf("[redacted]\n");                    ②Where am I?
}

int main(int argc,char *argv[])
{
    myprint("12345671234567123456712345671234567\xaa\x84\x04\x08");
    return 0;
}

*************************************************************
    .section    .rodata
.LC0:
    .string "[redacted]"                       ③%s \n
    .text
    .globl  myprint
    .type   myprint, @function
myprint:
.LFB0:
    .cfi_startproc
    pushq   %rbp
    .cfi_def_cfa_offset 16
    .cfi_offset 6, -16
    movq    %rsp, %rbp
    .cfi_def_cfa_register 6
    subq    $48, [redacted]                    ④%rsp
    movq    %rdi, -40(%rbp)
    movq    %fs:40, [redacted]                                  ⑤%rax
    movq    %rax, -8(%rbp)
    xorl    %eax, %eax
    movq    [redacted], %rdx                   ⑥-40(%rbp)
    leaq    -32(%rbp), %rax
    movq    %rdx, [redacted]                                    ⑦%rsi
    [redacted]                                 ⑧movq  %rax, %rdi
    call    strcpy
    [redacted]  -32(%rbp), %rax                                 ⑨leaq
    movq    %rax, %rsi
    movl    $.LC0, %edi
    movl    $0, %eax
    call    printf
```

```
    nop
    [redacted], %rax          ⑩movq  -8(%rbp)
    xorq    [redacted]                    ⑪%fs:40, %rax
    je  [redacted]            ⑫.L2
    call    __stack_chk_fail
.L2:                          注：下面这种回答也对：
    Leave                     ⑩movq  %fs:40
    .cfi_def_cfa 7, 8         ⑪-8(%rbp), %rax
    ret
    .cfi_endproc
.LFE0:
    .size   myprint, .-myprint
    .section    .rodata
.LC1:
    .string "Where am I?"
    .text
    .globl  alert
    .type   alert, @function
alert:
.LFB1:
    .cfi_startproc
    pushq   %rbp
    .cfi_def_cfa_offset 16
    .cfi_offset 6, -16
    movq    %rsp, %rbp
    .cfi_def_cfa_register 6
    movl    $.LC1, %edi
    call    puts
    nop
    popq    %rbp
    .cfi_def_cfa 7, 8
    [redacted]                ⑬ret
    .cfi_endproc
.LFE1:
    .size   alert, .-alert
    .section    .rodata
    .align 8
.LC2:
    .string "123456712345671234567123456712345671234567\252\[redacted]\004\b"   ⑭204
    .text
    .globl  main
    .type   main, @function
main:
.LFB2:
    .cfi_startproc
    pushq   %rbp
    .cfi_def_cfa_offset 16
    .cfi_offset 6, -16
    movq    %rsp, %rbp
    .cfi_def_cfa_register 6
    subq    $16, %rsp
    movl    %edi, -4(%rbp)
    movq    %rsi, -16(%rbp)
    movl    $.LC2, %edi
    [redacted]                ⑮call  myprint
    movl    $0, %eax
    leave
    .cfi_def_cfa 7, 8
    ret
    .cfi_endproc
.LFE2:
    .size   main, .-main
```

得分

第四题（20 分）

分析64位的Y86 ISA中新加入的条件内存传送指令：crmmovqXX和cmrmovqXX。crmmovqXX和cmrmovqXX指令在条件码满足所需要的约束时，分别执行和rmmovq以及mrmovq同样的语义。其格式如下：

rmmovq	4	0	rA	rB	D（8字节）
crmmovqXX	4	fn	rA	rB	D（8字节）
mrmovq	5	0	rA	rB	D（8字节）
cmrmovqXX	5	fn	rA	rB	D（8字节）

1．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；寄存器堆R[]，存储器M[]，程序计数器PC，条件码CC。其中对存储器的引用必须标明字节数。（10分）

阶段	rmmovq rA,D(rB)	cmrmovqXX D(rB),rA
取指	icode:ifun ← M_1[PC] rA:rB ← M_1[PC+1] valC ← M_8[PC+2] valP ← PC + 10	
译码	valA ← R[rA] valB ← R[rB]	
执行	valE ← valB + valC	valE ← valB + valC Cnd ← Cond(CC, ifun)
访存	M_8[valE] ← valA	valM ← M_8[valE] 或 if(Cnd) valM ← M_8[valE]
写回	none	if(Cnd) R[rA] ← valM
更新PC	PC ← valP	

2.为了执行上述新增指令，我们需要改进教材所描述的PIPE处理器，在回写（W：Write Back）阶段引入寄存器以保持流水线信号___W_Cnd___（请参考教材对信号的命名规则书写），以便有条件地更新寄存器内容。在如此改进的PIPE处理器上，请写出如下信号的HCL代码。（6分）

信号	HCL代码
F_stall	(E_icode in {IMRMOVQ, IPOPQ} || (_____①_____)) && E_dstM in _____②_____ || IRET in _____③_____
E_bubble	(_____④_____) || (E_icode in {IMRMOVQ, IPOPQ} || (_____①_____)) && E_dstM in _____②_____
M_bubble	m_stat in _____⑤_____ || W_stat in _____⑤_____

附HCL描述中的常数值编码表如下：

IHALT	halt指令的代码	INOP	nop指令的代码
IRRMOVQ	rrmovq指令的代码	IIRMOVQ	irmovq指令的代码
IRMMOVQ	rmmovq指令的代码	IMRMOVQ	mrmovq指令的代码
ICRMMOVQ	crmmovqXX指令的代码	ICMRMOVQ	cmrmovqXX指令的代码
IOPL	整数运算指令的代码	IJXX	跳转指令的代码
ICALL	call指令的代码	IRET	ret指令的代码
IPUSHQ	pushq指令的代码	IPOPQ	popq指令的代码
FNONE	默认功能码	RNONE	表示没有寄存器文件访问
ALUADD	表示加法运算	RRSP	表示%rsp寄存器ID
SAOK	正常地址操作状态码	SADR	地址异常状态码
SINS	非法指令异常状态码	SHLT	halt状态码

① E_icode == ICMRMOVQ && e_Cond

② {d_srcA, d_srcB}

③ {D_icode, E_icode, M_icode}

④ E_icode == IJXX && !e_Cond

⑤ {SADR, SINS, SHLT}

3.对于下面的Y86汇编代码，请使用上述条件内存传送指令将其修改为不带跳转的汇编代码序列。假设下面的代码片段在教材所描述的PIPE处理器上运行，不考虑该片段前后代码的影响以及高速缓存（cache）失效的情况，假设%rsi初值为0，处理器设计使用总是选择（always taken）的预测策略。原始代码片段预计运行___11__周期，改进代码片段预计执行___9(或8)___周期。（4分）

注：由于第2小题的HCL描述，要求带Cond，因为这是一个更优化的设计，而且和教材上对IJXX的考虑一致。基于此，第3小题可以有两种指令序列，cmrmoveqne在前的序列执行需要8个周期，cmrmoveqe在前的指令序列需要9个周期，因此最后一空填8或9都对。

原始代码	改进代码
andq %rsi %rsi jne L1 mrmovq 8(%rdx), %rax jmp L2 L1: mrmovq 8(%rdx), %rbx L2: addq %rax, %rbx	（9个周期的版本） andq %rsi %rsi cmrmovqe 8(%rdx), %rax cmrmovqne 8(%rdx), %rbx addq %rax, %rbx （8个周期的版本） andq %rsi %rsi cmrmovqne 8(%rdx), %rbx cmrmovqe 8(%rdx), %rax addq %rax, %rbx

得分

第五题（20 分）

现有一个能够存储 4 个 Block 的 Cache，每一个 Cache Block 的大小为 2 Byte（即 B = 2）。内存空间的大小是 32 Byte，即内存空间地址范围如下：

0_{10} (00000_2) -- 31_{10} (11111_2)

现有一程序，访问内存地址序列如下所示，单位是 Byte。

0_{10} 3_{10} 4_{10} 7_{10} 16_{10} 19_{10} 21_{10} 22_{10} 8_{10} 10_{10} 13_{10} 14_{10} 24_{10} 26_{10} 29_{10} 30_{10}

1．Cache 的结构如下图所示（S=2， E=2），初始状态为空，替换策略 LRU。请在下图空白处填入上述数据访问后 Cache 的状态。（12 分）

（TAG 使用二进制格式；Data Block 使用十进制格式，例：M[6-7]表示地址 6_{10}-7_{10} 对应的数据）

	V	TAG	Data Block	V	TAG	Data Block
set0	1	110	M[24-25]	1	111	M[28-29]
set1	1	110	M[26-27]	1	111	M[30-31]

上述数据访问一共产生了多少次 Hit： __0__ （2 分）

2．在第 1 小题的基础上，现增加一条数据预取规则：每当 cache 访问出现 miss 时，被访问地址及其后续的一个 cache block 都会被放入缓存，即当 M[0-1]访问发生 miss，则把 M[0-1]和 M[2-3]都放入缓存中。那么，这 16 次数据访问一共产生了多少次 Hit ： __8___ （2 分）

3．在第 1 小题的基础上，如果每一个 Cache Block 的大小扩大为 4 Byte（即 B = 4，cache 大小变为原来的 2 倍），这 16 次数据访问一共产生了多少次 Hit ： __8___ （2 分）

4．在第 3 小题的基础上，考虑增加 2 中的数据预取规则，这 16 次数据访问一共产生了多少次 Hit ： __12___ （2 分）