计算机组成与设计 2024春夏

选择

1. 10% 的指令 CPI 为 1，20% 的指令的 CPI 为 2，30% 的指令的 CPI 为 3，40% 的指令的 CPI 是 4，计算总的 CPI
2. 下面哪个运算结果不是NaN：A 是 0*infinity，B 是 infinity minus negative infinity，C 是 8 divided by 0，D 是任何涉及NaN的计算
3. 试卷开头的地方给了OpCode之类。问0xE2952023对应的指令是什么
4. 选出 -59.875 的单精度浮点数表示
5. IEEE754-2008定义了一种浮点数，1 bit sign，5 bit exponent，10 bit fraction，求其可以取值的范围
   四个选项形状be like：正负1.0000 0000 0 * 2^-14 ~ 正负1.1111 1111 1 * 2^15 ，正负0，正负无穷，NaN
   四个选项有区别浮点部分是9位还是10位、是$2^{-14~-2}15$还是$2^{-15~-2}14$
6. 在polling、DMA和interruption中，哪种要求processor与IO并行（或者都不需要）
7. 增大 cache 的 block size 可以减少什么，选项 A 是 conflict miss，B 是 hit time，C 是 miss penalty，D 是 compulsory miss
   （这道题目去年也有类似的，这几个名词意思可以自己查查）
8. seek time=8ms，转速7200RPM，每track有1000个sector，计算access time
   （track sector应该是干扰项）

大题

第一题：给branch load store指令占比以及Instruction miss rate、Data miss rate等，反正你见过的都有，求在它们共同作用下，CPI是多少

第二题：给出一段汇编代码，分析代码运行完成的时候，各寄存器的取值。

下面的内容是对这道题的模拟，大致是这个意思，不能保证题目正确、与原题目数值完全一致。作业或历年卷里面没见过特别像的题目。
Memory地址与内容：0x5004处内容为8，0x5008处内容为20，0x500C处内容为5
初始时刻x21寄存器的值为0x5004，x22寄存器的值为0x5，其余各寄存器值均为0
指令地址与内容：

```
loop:
0x8000: slli x20, x10, 2
0x8004: addi x23, x21, x20
0x8008: lw x24, 0(x23)
0x800C: beq x24, x22, exit
0x8010: addi x10, x10, 1
0x8014: jal x1 loop
exit:
```

没了
求最终的 x20、x23、x24、x1、PC

PageTableSize计算：
Virtual address 48 bit，DRAM 512GiB，SRAM 256GiB，PageSize 4 KiB，PageEntry 4 bytes
（1）求5并行进程需要多大的PageTable
（2）显然这个大小超出了内存可以容纳的范围，如果想要能塞进内存，PageSize应为多大

流水线：先给一张五级流水线CPU的图，再给一段汇编程序

```
I1: addi x5, x0, 1
I2: addi x6, x0, 1
I3: add x7, x5, x6
I4: add x5, x6, x7
I5: add x6, x7, x5
I6: add x7, x5, x6
```

如是作无限循环
（1）这段程序是干啥的
（2）运行这段程序会在哪里发生hazard，在五级流水线没有任何针对hazard的改进的情况下，求CPI，为什么
（3）求用前递解决数据冲突之后的 CPI
（4）分别问 I2 I3 I4 的两个操作数的前递来源，从ID/EX、EX/MEM、MEM/WB种选择
（5）给你一个新指令Jr r1, r2, imm，要求能实现r1<- PC+4，PC<- Memory[r2+imm]，指令的格式应当如何设计，参数都放到哪个位置，需要对流水线作何改动，在原图上改

汇编：
（1）将0x0123456789ABCDEF存进x10
（2）C转汇编，int为64位，参数从x10传入，从x11传出

```
int sum ( int num ) {
    if ( num < 10 )
        return num ;
    else return num%10 + sum( num/10 ) ;
}
```

这几个10都是十进制的，我考试的时候除法指令忘记记了，一度以为题目的意思是十六进制

最后一道Cache，25分，
（1）一个有 64 块，每块 64 bytes 的cache，主存有 8192 块，问 tag index offset 各有几位
（2）给出一个Cache，no write allocate+write back，LRU，2-way set associative，width=2，32bit block，最开始时候四个格子里面有一个是已经有的，另外三个空着
给出10个读写操作，这串读写设计得颇为精微，我回忆不上来，总之练习的时候尽量多练一些情况吧
（2.1）写出每一步是hit还是miss，以及有没有Write Cache、Write Back、Hit Replacement
这里 Write Back 的意思是写回内存，Write Cache 的意思是直接往 Cache 里面写包括读出来修改（因为不采取 Write allocate 所以与平常练习的略有不同）
（2.2）问最终的cache状态，需要写出四个格子的dirty bit、Tag 和 对应内存的内容（如Memory[20-23]）

2021-2022 春夏学期 计算机组成与设计 期末考试回忆卷

> By 瓜豪 & 贝拉拉贝拉 & 其他热心的98ers

> Note: 允许携带一张A4，不允许携带计算器

分布：10道选择，20分；4道大题

Name (Field size)	Field 7 bits	5 bits	5 bits	3 bits	5 bits	7 bits	Comments
R-type	funct7	rs2	rs1	funct3	rd	opcode	Arithmetic instruction format
I-type	immediate[11:0]		rs1	funct3	rd	opcode	Loads & immediate arithmetic
S-type	immed[11:5]	rs2	rs1	funct3	immed[4:0]	opcode	Stores
SB-type	immed[12,10:5]	rs2	rs1	funct3	immed[4:1,11]	opcode	Conditional branch format
UJ-type	immediate[20,10:1,11,19:12]				rd	opcode	Unconditional jump format
U-type	immediate[31:12]				rd	opcode	Upper immediate format

给出了需要用到的opcode和Fun3/7

Opcode：Beq/Bne 0x63

Fun3：Beq 0 Bne 1

选择题

> 1. 机器码`11111110000000000001011011100011`

A. `Beq x0, x0, -10` B. `Beq x0, x0, -20` C. `Bne x0, x0, -10` D.`Bne x0, x0, -20`

> 2. IEEE754规则化能表示的最小数。

> 3. PC最初为`0x20000000`，问`jal`指令所能跳转的范围

（似乎是）A. `[1FF00000, 200FFFFF]` B. `[1FF00000, 200FFFFE]` C. `[1FF80000, 200FFFFE]` D. `[1FF00000, 200FFFFF]`

> 4. 提高block size可以显著影响

A. cache miss B. Compulsory miss rate C. hit time D. miss penalty

> 5. Exception相关

> 6. float加法，0.12345×10^{-1}与0.12345×10^{3}在对齐后，指数位

A. both -1 B. both 3 C. both 0 D.忘记了

> 7. I/O相关：根据CPU efficiency，对3种I/O方式从高到低排序

选项就是polling，interupt，DMA的排列组合

> 8. 如何获得一个比较大的立即数地址存储的内容（原题干有些问题，这里稍作修改）

```
A. lui x5, higher_bits

   ori x5, x5, lower_bits

   lw x7, 0(x5)
```

```
B. lui x5, higher_bits

   or x5, x5, lower_bits

   lw x7, 0(x5)
```

```
C. lui x5, higher_bits

   lw x7, lower_bits(x5)
```

D忘记了。。。 Hint：各类型立即数范围

> 9. 判断以下指令发生了什么：WAW(Write After Write)，RAW(Read After Write)，WAR(Write After Read)

```
lw x1, 0(x1)
add x2, x1, x1
sub x1, x2, x3
```

还有一道选择题忘记了，欢迎楼下补充

大题

CPI相关

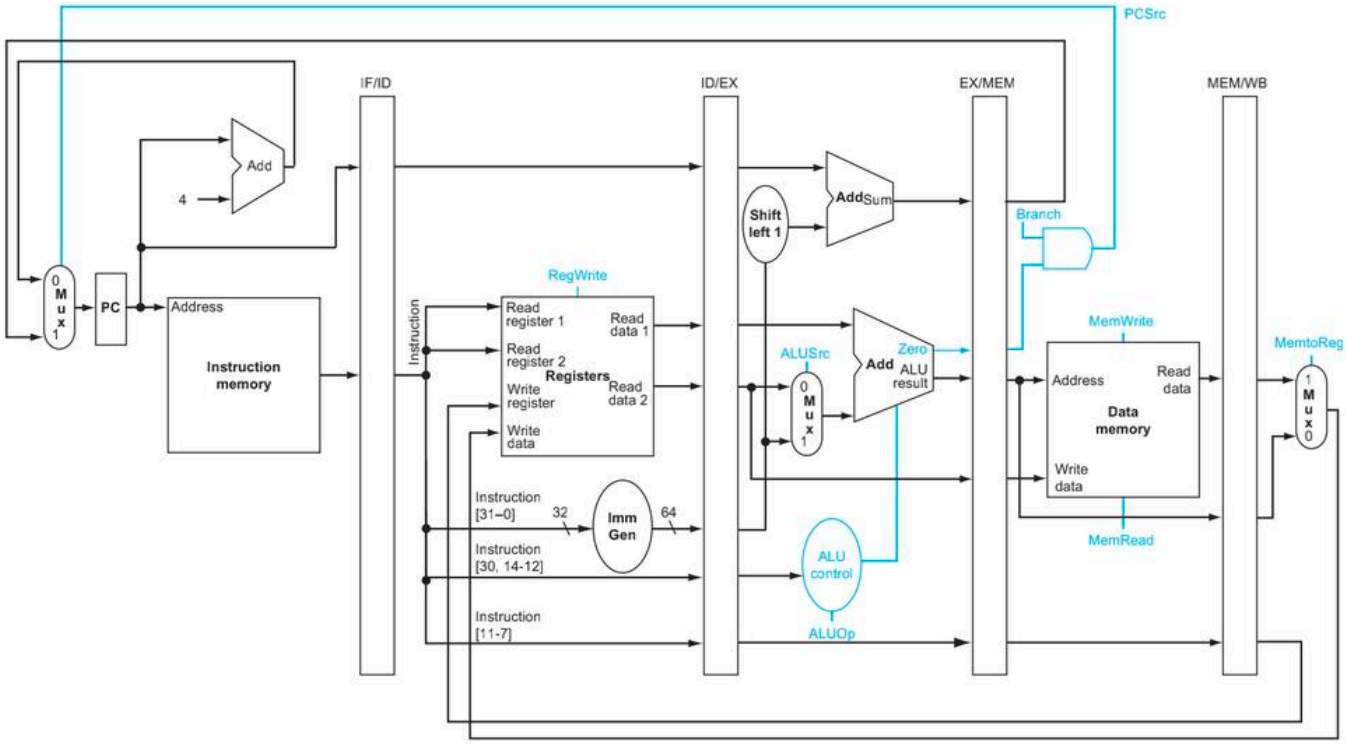
稍后我找一道类似的题目放在这里（没有找到很像的，这里我自己出一道，还是欢迎大佬补充更改数据）

ALU/Logic	Jump/Branch	Load	Store
45%	20%	20%	15%

已知：5-stage 200MHz流水线CPU，具有前递，stall detector；对跳转指令采取not-taken预判，实际上45%的跳转指令not-taken，跳转的判断在4th阶段（MEM）进行；每一次从memory中fetch需要75ns；I-MEM miss rate为90%，D-MEM miss rate 为98%；有25%的load指令会发生load-use hazard。

请求：该CPU的CPI值。

Datapath相关



1. SW指令，对应的CPU control信号：ALU_Src, RegWrite, MemtoReg, PCSrc

2. 给出一段代码，插入NOP（根据题目原图，无forwarding等）

```
I0  add t2, s1, sp
I1  lw  t1, 0(t1)
I2  addi t2, t1, 7
I3  add t1, s2, sp
I4  lw  t1, 0(t1)
I5  addi t1, t1, -9
I6  sub t1, t1, x2
```

3. 更改datapath，使其可以满足forwarding以解决算术指令的RAW竞争

4. 请在表格中填写ALU中A，B的来源（从ID/EX，EX/MEM，MEM/WB中进行选择）

Inst.	ALU_Src_A	ALU_Src_B
I0		
I1		N/A
I2		N/A
I3		
I4		N/A
I5		N/A
I6		

> 5. 求指令所用的Cycle数（有forwarding）
>
> 6. 请调整顺序，尽可能减少stall（有forwarding），并求需要多少Cycle

汇编

> (1) 用最少的指令实现 x20>x11 | x20 < 0的跳转
>
> (2) beq能跳转的范围比较小，请使用一对指令实现`Beq x0, x1, L`（L较大）
>
> (3)
>
> ```
> int find(int *mv, int *v, int n) {
> if(n > 0) { *mv = max(v, n); return 1; }
> return -1;
> }
> int max(int *p, int n) {
> int i, m = p[0];
> for(int i = 1; i < n; i++) {
> if(p[i] > m) m = p[i];
> }
> return m;
> }
> ```
>
> 将上述C语言代码转为RISCV汇编语言，要求进行保护（忘记题目的说法，总之要求在进入函数将要保护的寄存器push，离开前pop）。

存储层级相关

两部分，cache，TLB。

已知如下：

> 1. virtual memory address 54 bits，physical memory address 32 bits；
> 2. 4KB cache 2路组相连，block size 128B；
> 3. TLB 2路组相连，512 entries；
> 4. page size 8KB。

题目

1. （1）求cache中Tag，Index，Block offset的位数，求cache中一条data的位数。

   （2）求TLB中Tag，Index，Page offset的位数，求一条TLB中存储data的位数

2. 给出如下一组地址查询（10进制），请填入每一个地址hit/miss（使用LRU策略进行替换）

0	100	200	300	1024	2048	4096	250	100

根据上表求hit ratio；求LRU替换多少个block

3. 根据上边的查询，给出最后cache中的内容（仅需给出valid部分），以如下格式:`<index, tag, data[xx bytes-xx bytes]>`

最后一次更改时间：2022.06.20 13:23
在此感谢刘海风老师和郝家辉学长的帮助！郝助教毕业快乐！
同时感谢热心的98ers的补充，尤其感谢@贝拉拉贝拉提供的内容，极大丰富了初版回忆卷。

2022-2023春夏

单选

> 1.下面运算中，哪个 OF=1(Overflow), CO=1(Carry Output).

A. `0x12+0x78` B. `0x80+0x80` C. `0x12+0x34` D. `0x12+0x1F`

> 2.IEEE 单精度下能表示的最大规格化数(normalized) 是多少?

A. `FFFFFFFF` B. `FFFF7FFF` C. `7F7FFFFF` D. 忘了

> 3.我要把一个大整数 A(32 bit) 加载到 x29 中，如何做?

A.
lui x29, higher_bits
ori x29, x29, lower_bits
B.
lui x5, higher_bits
ld x29, lower_bits(x5)
C.
lui x5, higher_bits
ori x5, lower_bits
ld x29, 0(x5)
D.
addi x29, x0, higher_bits
slli x29, x29, 16
ori x29, x29, lower_bits

> 4.当前 PC 地址 0x30000000, jal 范围

A. [1FF00000, 200FFFFF] B. [1FF00000, 200FFFFE] C. [1FF80000, 200FFFFE] D. [1FF00000, 200FFFFF]

> 5.单周期 CPU 中，以下哪些操作是不能在一个时钟周期内完成的

A. 从内存里读，并写数据
B. ALU 计算，并写数据到内存
C. 更新 PC, 并写数据到内存
D. 从寄存器堆读值，进行 ALU 计算，并写数据到内存。

> 6.如果我们采用 a mixed cache 用来取指和读内存，会有什么冒险?

A. 啥都没有
B. Data hazard
C. Control hazard
D. Structural hazard

> 7.提高 cache 的组相联度可以提高(improve)

A. capacity miss
B. hit time
C. conflict miss
D. compulsory miss

> 8.下面的编程指令，有没有 RAW, WAR, WAW 依赖`x2` `x1` `x3`具体不记得了

> 9.关于异常的说法，哪个是不正确的

A. 进入处理程序前要把原因放入 cause register.
B. 要把当前指令的地址放到 mepc, 以便处理程序执行结束后返回
C. 要调到 mtvec 的位置
D. 处理程序执行结束后要 jalr x0, 0(x1)

> 10.RAID 中，Rundancy 是作用是什么

A. 提高速度
B. 忘了
C. 检验读写正确性
D. 提高硬盘的是 reliability

大题

CPI 计算

200 MHz. 采用 mixed cache. ALU/Logic 35%
Load 30%
Store 15%
Branch 20%
cache hit rate 98%, 内存读写 100ns. branch 采用 predict-not-taken. 假设跳转在 3rd 阶段决定。
branch not taken 的比例为 40%, load 后有 50% 的比例会有 load-use hazard. 求 CPI.

Pipeline
(1) LW 指令的 PCSrc ALUSrc MemToReg MemWrite RegWrite
(2) 插入 nop

```
addi
lw sp
lw t1, 0(t1)
lw t1,
sw t1, 0(t2)
```

(3) 添加 datapath 以支持 forwarding 解决 RAW 的竞争
(4) 对于添加了上一问 forwarding 的通路，画出此时 (2) 指令执行的流水线图，并标出数据前递，格式如下
例：MemWB.ALUOut -> ALU up port

汇编

(1) 用汇编实现 int B[k]=A[i-j] 其中 i,j,k 在 . A, B 在 x10, x11
(2) 实现（注意 RISC-V calling convention）

```
int sum(int *u, int n) {            // int is 32-bit
    int i, sum = 0;
    for (i = 0; i < n; i++)
        if (isOk(u[i]))sum += u[i];
    return sum;
}
bool isOk(int x) {
    if (x>=0 && x<=100) return true;
    else return false;
}
```

存储

已知 virtual memory address 54 bits，physical memory address 32 bits;
4KB cache 2路组相连，block size 128B, LRU
TLB fully associative, 128 entries
page size 4KB

（1）求 cache 的 data, index, tag, byte offset bits 以及放在缓存中的 tag/valid bits(NOTE: 包括 VALID bit)
求页表的 offset, 虚拟页号位，物理页号位. TLB 需要多少个 comparator.
（2）下面顺序访问(r 代表 read, w 代表 write)
0w 64r 96w 100r 400r 800r 1000r 16Cw
求 hit rate, replaced block?
求从 cache 中写到内存的块?
（3）根据上边的查询，给出最后cache中的内容（仅需给出valid部分），以如下格式: <index, tag, data[xx bytes-xx bytes]>

有些细节实在想不起来了，欢迎大家补充。

1. Different meaning of data.

The bits have no inherent meaning. Given the pattern:

0x8D100123

What does it represent, assuming that it is:

1)a two's complement integer	
2)signed and magnitude integer	
3)A IEEE754 single precision floating-point number	

2. IEEE 754, Carry-flag, Overflow-flag……

(1) (6%)Show the IEEE754 for the floating-point number **-12.3** in single precision. Write down the Hexadecimal representation.

0x_____

(2). (16%)Suppose we have four 8-bit registers $A, $B, $C and $D. The signed 8-bit integer is stored in 2's complement format. Calculate $C=$A+$B. and mark if it is Overflow(OF=1) or Carryout(CF).

$A	$B	$C(Hex) *$C=$A+$B*	CF	OF
121	107	0x		
98	-112	0x		
-87	-76	0x		
-67	123	0x		

3. please select the best option from option list and fill corresponding letter of A-K into blank below.

Question	(1)	(2)	(3)	(4)	(5)	(6)	(7)
Answer					不做	不做	不做

(1).A number is represented as 0x7F800000 in IEEE 754 Single precision number format, its value is _____.

(2). IEEE754 single precision representation for -1 is _____.

(3). A number is represented as 10101101_00010000_00000000_00000000 in 2's complement format, original value of this number is _____.

(4). A number is represented as 0xF0000320 in 1's complement format, original value of this number is _____.

~~(5).Value of machine code for instruction 'jr $ra' is _____~~. （本题不做）

~~(6).Value of machine code for instruction 'andi $t2, $t2, 4' is _____~~. （本题不做）

~~(7).Value of machine code for instruction 'sub $t0,$t1,$t2' is _____~~. （本题不做）

Options for above blanks are listed below:

(A) 0x03E00008 (B) 0XBF800000 (C) 0XBF100000 (D) -0x0FFFFCDF
(E) 0x314A0004 (F) +∞ (G) -0x52F00000 (H) 0x12A4022
(I) 0x12A4020 (J) -0x52700000 (K) None of above

1. A memory and cache subsystem uses byte-addressing。。。

【答案:】

32-bit **0x**22339ab memory address of following table contains 3 parts:

Tag, Index, Byte-Offset 0x22339ab= 0000 0010 0010 0011 0011 1001 1010 1011

16 13 3 0000 0010 0010 0011-0,011 1,001 1,010 1-011

13 13 6 0000 0010 0010 0-0,11 00,11 10,01 10-10 1011

14 15 3 0000 0010 0010 00-11 0,011 1,001 1,010 1-011

15 12 5 0000 0010 0010 001-1 001,1 100,1 101-0 1011

Cache feature	Index value(Hex)	Index size, bits	TAG size, bits	Tatal Size,KB
64KB cache data,Direct-mapped, 8 bytes/block	0x0735	13	16	8k*(1+16)/8+64KB=81KB
512KB cache data, Direct-mapped, 64 bytes/block	0xce6	13	13	8K*(1+13)/8+512KB=526KB
512KB cache data, 2-Way set associative 8 bytes/block	0x6735	15	14	64K*(1+14)/8+512KB=632KB
1024KB cache data, 8-Way set associative 32 bytes/block	0x9cd	12	15	32k*(1+15)/8+1024KB=1088KB

2. A computer with multi-cycle CPU runs at 5 GHz 。。。。

【答案:】

(1).Total Clock cycles of 1200 accesses in Part-A is (A) 1200 ;

(2).Total Clock cycles of 40 accesses in Part-B is (B) 40*(1+25)=1040 , these 40 accesses need more Clock cycles than 40 accesses in perfect cache mode, the increased cycle number is noted as J, value of J is (C) 40*25=1000 .

(3).Total Clock cycles of 10 accesses in Part-C is (D) 10*(1+25+400) =4260 , these 10 accesses need more Clock cycles than 10 accesses in perfect cache mode, the increased cycle number is noted as K, value of K is (E) 10*(25+400) =4250 .

(4).Average CPI of TP run in not perfect cache mode is noted as L, L may be written as an arithmetic expression about I,J and K, please fill following blank with such an arithmetic expression:

L= (F) (4.2*I+J+K)/I

1. A memory and cache subsystem uses byte-addressing, 32-bit address. Each row of the table below indicates a kind of cache, the "Cache feature" column of table describes the total size of cache data (not containing tag and valid bit), cache kind, cache block size. A 32-bit memory address 0x22339AB contains 3 fields: tag, index, byte offset, some of these field's size(bit number of this field) or value should be filled into table blank below. You should use Hex-decimal number to fill into column "Index value(Hex)". Total cache size containing data block, tag and valid bit should also be filled into column "Tatal Size,KB"(KB means unit is KB—1024 byte, not byte), only number (not expression) is allowed to be filled into this colomn.

Cache feature	Index value(Hex)	Index size, bits	TAG size, bits	Tatal Size,KB
64KB cache data,Direct-mapped, 8 bytes/block	0x0735	13	16	81
512KB cache data, Direct-mapped, 64 bytes/block				
512KB cache data, 2-Way set associative 8 bytes/block				
1024KB cache data, 8-Way set associative 32 bytes/block				

Example value has been given in first question row.

2. A computer with multi-cycle CPU runs at 5 GHz , it has 2 levels of cache: the first level cache C1 ,and the second level cache C2. When a test program TP runs, total I instructions is executed with M memory accesses, here I=1000, M=1250. 1250 memory accesses include fetching instruction from memory, loading memory operand

into register, …… etc. When this TP runs first in perfect cache mode that all instruction and data are located in C1, all M memory accesses hit in C1, and CPI is 4.2. Now TP runs in not perfect cache mode, 1250 memory accesses are divided into 3 parts:

Part-A: 1200 accesses hit in C1, they only need to access C1;

Part-B: 40 accesses hit in C2, they need to access C1 and C2;

Part-C: 10 accesses need to access C1, C2 and DRAM;

DRAM access time: 80ns, C2 access time: 5ns, C1 access time: 0.2ns. Please fill numbers （not expressions） into following blank (A)—(E).

(1).Total Clock cycles of 1200 accesses in Part-A is (A) ;

(2).Total Clock cycles of 40 accesses in Part-B is (B) , these 40 accesses need more Clock cycles than 40 accesses in perfect cache mode, the increased cycle number is noted as J, value of J is (C) .

(3).Total Clock cycles of 10 accesses in Part-C is (D) , these 10 accesses need more Clock cycles than 10 accesses in perfect cache mode, the increased cycle number is noted as K, value of K is (E) .

(4).Average CPI of TP run in not perfect cache mode is noted as L, L may be written as an arithmetic expression about I,J and K, please fill following blank with such an arithmetic expression:

L= (F)

1. Translate the following C code to RISC-V assembly code.

Answer

```
LOOPI:
    addi x7, x0, 0       // Init i = 0
    bge x7, x5, ENDI     // While i < a
    addi x30, x10, 0     // x30 = &D
    addi x29, x0, 0      // Init j = 0
LOOPJ:
    bge x29, x6, ENDJ    // While j < b
    add x31, x7, x29     // x31 = i+j
    sd x31, 0(x30)       // D[4*j] = x31
    addi x30, x30, 32    // x30 = &D[4*(j+1)]
    addi x29, x29, 1     // j++
    jal x0, LOOPJ
ENDJ:
    addi x7, x7, 1       // i++;
    jal x0, LOOPI
ENDI:
```

2. Implement the following C code in RISC-V assembly. Hint: Remember that the stack pointer must remain aligned on a multiple of 8（原来的16 应改成 8）.

------- -------

Answer

```
// IMPORTANT! Stack pointer must reamin a multiple of 16!!!!
fib:
beq x10, x0, done // If n==0, return 0
addi x5, x0, 1
beq x10, x5, done // If n==1, return 1
addi x2, x2, -16 // Allocate 2 words of stack
space
sd x1, 0(x2) // Save the return address
sd x10, 8(x2) // Save the current n
addi x10, x10, -1 // x10 = n-1
jal x1, fib // fib(n-1)
ld x5, 8(x2) // Load old n from the stack
sd x10, 8(x2) // Push fib(n-1) onto the stack
addi x10, x5, -2 // x10 = n-2
jal x1, fib // Call fib(n-2)
ld x5, 8(x2) // x5 = fib(n-1)
add x10, x10, x5 // x10 = fib(n-1)+fib(n-2)
// Clean up:
```

```
ld x1, 0(x2) // Load saved return address
addi x2, x2, 16 // Pop two words from the stack
done:
jalr x0, x1
```

1. Different meaning of data.

Answer:
-0x72EFFEDD
-0x0D100123
-(1.00100000000000100100011)two*2^-101

2. IEEE 754, Carry-flag，Overflow-flag……

Answer:
(1). 0xC144CCCC or 0xC144CCCD
(2). E4-0-1
F2-0-0
5D-1-1
38-1-0

3. please select the best option from option list and fill corresponding letter of A-K into blank below.

Question	（1）	（2）	（3）	（4）	（5）	（6）	（7）
Answer					不做	不做	不做

答案：

Question	（1）	（2）	（3）	（4）	（5）	（6）	（7）
Answer	F	B	G	D	不做	不做	不做

1. Different meaning of data.

The bits have no inherent meaning. Given the pattern:

0x8D100123

What does it represent, assuming that it is:

1)a two's complement integer	
2)signed and magnitude integer	
3)A IEEE754 single precision floating-point number	

2. IEEE 754, Carry-flag, Overflow-flag……

(1) (6%)Show the IEEE754 for the floating-point number **-12.3** in single precision. Write down the Hexadecimal representation.

0x_____

(2). (16%)Suppose we have four 8-bit registers $A, $B, $C and $D. The signed 8-bit integer is stored in 2's complement format. Calculate $C=$A+$B. and mark if it is Overflow(OF=1) or Carryout(CF).

$A	$B	$C(Hex) *$C=$A+$B*	CF	OF
121	107	0x		
98	-112	0x		
-87	-76	0x		
-67	123	0x		

3. please select the best option from option list and fill corresponding letter of A-K into blank below.

Question	(1)	(2)	(3)	(4)	(5)	(6)	(7)
Answer					不做	不做	不做

(1).A number is represented as 0x7F800000 in IEEE 754 Single precision number format, its value is _____.

(2). IEEE754 single precision representation for -1 is _____.

(3). A number is represented as 10101101_00010000_00000000_00000000 in 2's complement format, original value of this number is _____.

(4). A number is represented as 0xF0000320 in 1's complement format, original value of this number is _____.

~~(5).Value of machine code for instruction 'jr $ra' is _____~~. （本题不做）

~~(6).Value of machine code for instruction 'andi $t2, $t2, 4' is _____~~. （本题不做）

~~(7).Value of machine code for instruction 'sub $t0,$t1,$t2' is _____~~. （本题不做）

Options for above blanks are listed below:

(A) 0x03E00008 (B) 0XBF800000 (C) 0XBF100000 (D) -0x0FFFFCDF
(E) 0x314A0004 (F) +∞ (G) -0x52F00000 (H) 0x12A4022
(I) 0x12A4020 (J) -0x52700000 (K) None of above

浙江大学 2018～2019 学年 春夏 学期

《 计算机组成 》课程期末考试试卷（A）

课程号： 21186031__，开课学院：**计算机学院/软件学院**

考试试卷：√A 卷、B 卷（请在选定项上打√）

考试形式： 闭 卷，允许带 一页 A4 纸手写笔记 入场，笔记署名，不得互借

交卷方式： 试卷名字朝外对折整齐，草稿纸、笔记与试卷一起上交 。

考试日期： **2019 年 06 月 18 日(10:30～12:30)** ，考试时间： **120** 分钟

I. True or False(8x1%; √/×)

Statement	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
Answer								

(1). ( ) We can connect at most 7 disks to a cable of SCSI bus.

(2). ( ) There is a bit in status register, or cause register, this bit can be used to disable all kinds of hardware interrupt.

(3). ( ) In memory-mapped I/O system, dedicated I/O instruction can be used, these I/O instruction can specify both the device number and the command word.

(4). ( ) Write buffer is a small cache that can hold a few values waiting to go to main memory. It can be used to entirely eliminate any write stalls.

(5). ( ) When write-through cache scheme is used, some read misses still cause miss block in cache to be written back to main memory.

(6). ( ) C0(Coprocessor 0#) is used to process exception and float point number calculation.

(7). ( ) Floating point addition includes: [redacted], Adding Significands, Over/underflow processing, Rounding, Normalisation.

(8). ( ) Underflow means that the number is too small to be represented.

II. Choose 1 best answer. (10x2%)

(1) What is the sign extension (16 bit to 32 bit) result of number 0x89ab represented in 2's complement?

A. 0x000089ab B. 0xffff89ab C. 0x111189ab D. 0x800089ab

(2) MIPS addressing mode includes base addressing, immediate addressing, PC-relative addressing, etc, which instruction uses base addressing?

A. addi $s0,$s0,4 B. bne $s0,$s1, L1

C. j 4000 D. sw $s1,0($s0)

(3) A BNE instruction with machine code 0x1508000c is located at memory address 0x2004, suppose jump action occurs when executing this BNE instruction, the address of next instruction executed will be _____ .

A. 0x2038 B. 0x2034 C. 0x2014 D. none of above

(4) When write miss happens in a write-through cache system, the data is only written to main memory, it is not written into the cache, such cache technology is called _____.

A. Fetch-on-miss B. write back

C. write around D. none of above

[redacted]

(5) Which instruction will not cause overflow?

A. addu $t2,$t1,$t1 B. subiu $t2,$t1,0x23

C. addi $t2,$t1,0x23 D. nor $t2,$t1,$t1

(6) Which one is not a bus standard?

A.SCSI B. PCI C. DIMM D. EISA

(8) In virtual memory system, each TLB row (or called entry) contains serveral fields, which one is not such field?

A. Virtual page number

B. physical page number

C. Valid bit indicating TLB hit

D. A bit called D-bit indicating disk address is used.

(9) Which instruction can jump to an address at any position of 4GB memory?

A. jr $ra B. bne $t0,$t1,label

C. j label D. jal label

(10) Which instruction is not relative to floating point hardware?

A. bc0t B. mul.s

C. div.d D. bc1f

(7) For page table of virtual memory, what is the usage of dirty bit?
A. indicate the entry is filled with useless physical page address.
B. indicate the entry is filled with dirty physical page address.
C. indicate the TLB relative to this entry is filled with dirty data.
D. indicate new data has been written into the corresponding physical page.

III. (14%):

fill corresponding letter of A-K into table below.

Question	(1)	(2)	(3)	(4)	(5)	(6)	(7)
Answer							

(1).A number is representd as 0x7F800000 in IEEE 754 Single precision number format, its value is _____.

(2). IEEE754 single precision representation for -1 is _____.

(3). A number is represented as 10101101_00010000_00000000_00000000 in 2's complement format, original value of this number is _____.

(4). A number is represented as 0xF0000320 in 1's complement format, original value of this number is _____.

(5).Value of machine code for instruction 'jr $ra' is _____.

(6).Value of machine code for instruction 'andi $t2, $t2, 4'

(3). A number is represented as 10101101_00010000_00000000_00000000 in 2's complement format, original value of this number is _____.
(4). A number is represented as 0xF0000320 in 1's complement format, original value of this number is _____.

(5).Value of machine code for instruction 'jr $ra' is _____.

(6).Value of machine code for instruction 'andi $t2, $t2, 4'
is _____.
(7).Value of machine code for instruction 'sub $t0,$t1,$t2'
is _____.

Options for above blanks are listed below:

(A) 0x03E00008 (B) 0XBF800000 (C) 0XBF100000 (D) -0x0FFFFCDF

(E) 0x314A0004 (F) $+\infty$ (G) -0x52F00000 (H) 0x12A4022

(I) 0x12A4020 (J) -0x52700000 (K) None of above

浙江大学 2019～2020 学年 春夏 学期

《 计算机组成 》课程期末考试试卷（A）

课程号： 21186031__，开课学院：**计算机学院/软件学院**

考试试卷：√A 卷、B 卷（请在选定项上打√）

考试形式： 闭 卷，允许带 **一页 A4 纸手写笔记** 入场，**笔记署名，不得互借**

交卷方式： **试卷名字朝外对折整齐，草稿纸、笔记与试卷一起上交** 。

考试日期： **2020 年 09 月 11 日(10:30～12:30)** ，考试时间： **120** 分钟

OPcode: R-Type:0, Beq:4, Bne:5, J:2, Lw:35 SW:43
Function Code: Add:32, Sub:34, And:36, or:37, jr:8,
Register No. $s0:16, $t0:8

1. (20%)

1.1 (4%) 1) Assume that registers $s0 and $s1 hold the values 0x80000000 and 0xD0000000, respectively.

1) What is the value of $t0 for the following assembly code

```
add $t0,$s0,$1
```

A: 0x50000000 (overflow)

2) What if the value $t0 for the following assembly code

```
add $t0, $s0, $s1
add $t0, $t0, $s0
srl $t0,$t0,  2
```

A: 0x34000000

1.2 (2%) Assume the current PC is 0x1FFFFFFC, what's next value of PC after execution of "j 0x10"?

A: 0x10000100 ({PC[31:28],0x10(26 bits),00})

1.3 (2%)What is the hexadecimal value of 2020.

A: 0X7E4

1.4 (4%)Show the IEEE754 for the floating-point number **-32.6** and **0.64** in single precision. Write down the binary representation.

-32.6

1	10000100	00000100110011001100100

0.64

0	011 1111 110	10100011110101110000101

1.5(4%) For a processor with the clock rate 2.5Ghz, the instructions can be divided into four classes (Class A, B, C, D) according to their CPIs of 1, 2, 3,3. Given a program with instruction count of 1.0E6 instructions divided into classes as follows: 10% class A, 20% class B, 50% Class C, and 20% class D.

a. What is the global CPI for the program? 2.6;

b. The clock cycles required for executing the program? 2.6 $x10^6$;

1.6 (4%) Convert the MIPS instructions into machine code or Machine code into MIPS instructions.

Address(Hex)	MIPS Assembly Instruction	Machine Code(Hex)
100000	Loop: Beq $s0, $t8, L1	(0x12180030)
100004	(sub $s0, $s1, $s2)	0x02328022
100008	J Loop	(0x804000)
……		-
1000D4	L1: ……	-

2. (20%)Memory Hierarchy

2.1(10%) For a direct-mapped cache design with a 32-bit address, the following bits of the address are used to access the cache.

Tag	Index	Offset
31-11	10-6	5-0

1) What is the cache block size(in words)? 16 (1 分)

2) How many entries does the cache have? 32 (1 分)

3) The following byte-addressed cache reference are recorded.

0	4	16	132	160	1024	30	140	3000

How many blocks are replaced? 0 (2 分)

What is the hit ratio 5/9 (55.6%) (2 分)

4) List the final state of the cache after 3), with each valid entry represented as a record of <index, tag, data> (每个 Cache 状态 1 分)。
[0,0,0~63]。
[2,0,128~181]。
[16,0,1024~1087]。
[14,1,2944~3007]。
其他位置均为无效。

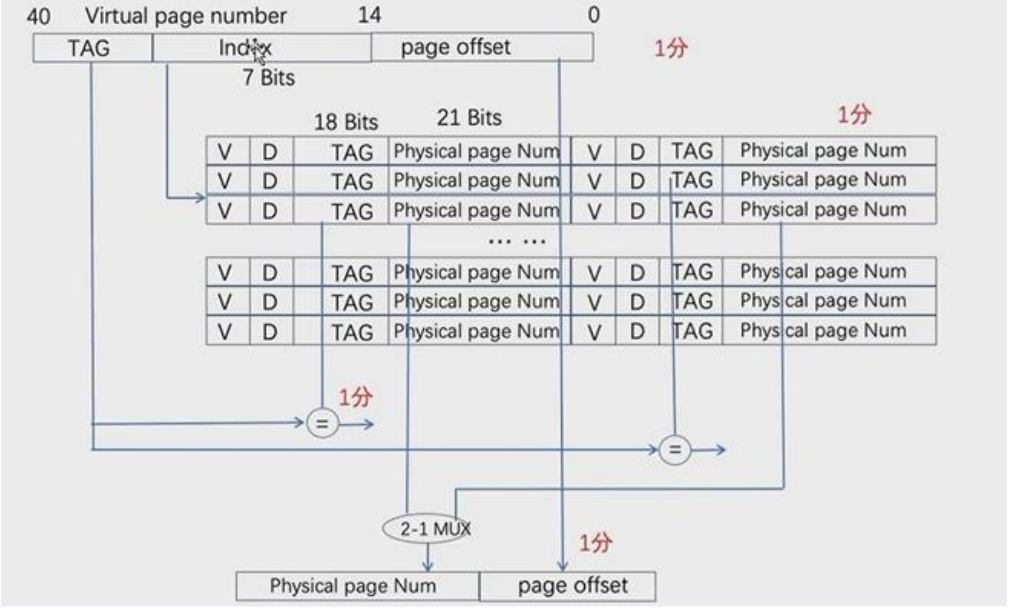
2.2(10%)Consider a virtual memory system with the following properties:
- 40-bit virtual byte address;
- Page size 32KB;
- 36-bit physical byte address.

1) What is the total size of the page table for each process on this processor? Assume that the valid and dirty take a total of 2 bits and that all the virtual pages are in use and the disk address are not stored in the page table. Each Entry of the page table should be byte aligned.

A: 1) Entry number: $2^{40}/32KB = 2^{25}$ (2 分)。
2) Entry size: 36 bits - 15 bits + 2 bits = 23 bits.
Byte aligned: 3 Bytes. (2 分)。
3) 3 Bytes x 2^{25} = 3 x 32M = 96MB; (2 分)。

2)Assume the virtual memory system is implemented with a two-way set associative TLB with 256 TLB entries. Show the virtual-to-physical mapping with a figure.

1) virtual memory address 分成 三部分： 1 分；
2) 2-way TLB 结构 ： 1 分；
3) index 过程： 1 分；
4) physical Address 过程： 1 分；



3(30%) Program

3.1(10%)

```
START: addi $s0, $zero, $zero
       addi  $s1, $zero, $zero
       addi  $s2, $zero, $zero
       addi  $t2, $zero,3
       la    $t1, LOOP1 #Load the addr of the Loop1 into $t1
       lw    $t3,4($t1)
       addi  $t3,$t3,4
       sw    $t3,4($t1)
       lw    $t3,8($t1)
       addi  $t3,$t3,8
       sw     $t3,8($t1)
       lw     $t3,($t1)

LOOP1:
       bne  $s1, $s2, LOOP2
       addi  $s0, 252.
```

```
       addi  $s0, 120
LOOP2:
       addi  $s0, 100
       addi  $s0, 50
       addi  $s1, 1
       addi  $t3,1
       sw    $t3,0($t1)
       subi $t2, 1
bne    $t2, $zero, LOOP1
END:   … …
```

After the instructions running, the content of register $s0 is (584(0x248)), Why? (Give all the different values of register $s0 during the program running)

1)First Loop: $s0=0; $s0=$s0+256; $s0=$s0+128; $s0=$s0+100; $s0=$s0+50;

2) Second Loop: $s0= $s0+50;

3) Thirt Loop: $s0 保持不变;

答案：4 分； 每个 LOOP 过程$s0 的变化： 2 分；

3.2 (20%) Implement the following C code in MIPS, assuming that the sort is the first function called:

```
void sort (int v[],int n){
    int i, j;
    for(i=1;i<n;i++){
       for(j=i-1;j>=0;j--){
          if(compare(v[j],v[j+1])){
             swap(v,j)
          }
       }
    }
}
int compare(int a, int b){
     if( a >= b)
        return 1;
     else
        return 0;
}
int swap(int v[], int k){
      int temp;
      temp = v[k];
      v[k] =v[k+1];
      v[k+1] =v[k];
}
```

Be sure to handle the stack and frame pointer appropriately. i corresponds to $s0, j corresponds to $s1 in function **sort**, temp correspond to $s2 in function **swap**.

1) Write the MIPS assembly code corresponding to this C function.

2) Draw the status of the stack before calling sort and during each function call. Indicate the names of registers and variables stored on the stack and mark the location of $sp and $fp.

```
sort:addi $sp, -20;
     sw  $ra, 16($sp)
     sw  $s3, 12 ($sp)
     sw  $s2, 8 ($sp)
     sw  $s1, 4($sp)
     sw  $s0,  0($sp)
(以上 Stack 保护 1 分，$s0,$s1 必须保护，其他根据代码情况)
     add $s2,$a0,$zero;//s1=V[]
     add $s3,$a1,$zero;//13=n
(以上参数传递 1 分)
     add $s0,$zero  //i=0
f1st: slt $t0,$s0,$s3
     beq  $t0, $zero exit1//i<n
     addi  $s1,-1;//j=i-1;
(第一个循环 1 分)
f2st:  slti $t0, $s1,0; //$j<0;
     bne   $t0,zero, exit2;
     sll   $t0,$s1,2// $t1=j*4;
     add   $t2, $s2,$t1 //v+j*4;
     lw     $a0, 0($t2) //v[j];
     lw     $a1, 4($t2) //v[j+4];
     jal    compare;
(准备 V[j],V[j+1]，正常跳转 3 分)
     beq    $v0, zero, exit2
     add    $a0, $a0, $s2;
     add    $a1, $a0, $$1;
     jal    swap
     addi   $s1,$s1,-1
     j      f2st.
(正常传递 swap 参数，跳转 2 分)
exit2:  addi.  $s0, $s0,-1;
     j      f1st
```

```
void sort (int v[],int n){
    int i, j;
    for(i=1;i<n;i++){
       for(j=i-1;j>=0;j--){
if(compare(v[j],v[j+1])){
            swap(v,j)
         }
       }
    }
}
```

	(正常判断结束 1 分)
	exit1: lw $s0, 0($sp)
	lw $s1, 4($sp)
	lw $s2, 8($sp)
	lw $s3, 12($sp)
	lw $ra, 16($sp)
	addi $sp, $sp,20
	jr $ra
	(正常返回 2 分)

int compare(int a, int b){
if(sub(a,b) >= 0)
return 1;
else
return 0;
}
	compare:
	slt $t0,$a0, $a1;//a<b
	beq $t0,zero, exit3;
	addi $v0, $zero,1
	exit3:
	addi $v0, $zero,$zero;
	jr $ra
	(compare 函数 2 分)

int swap(int v[], int k){
int temp;
temp = v[k];
v[k] =v[k+1];
v[k+1] =v[k];
}
	swap:
	sll $t1,$a1,2 //$t1=k*4;
	add $t1, $a0,$t1 //v+k*4
	lw $t0, 0($t1)
	lw $t2, 4($t1)
	sw $t2, 0($t1)
	sw $t0, 4($t1)
	jr. ra
	(swap 函数 2 分)

4.(20%)Multicycle CPU Design

1)(6%) Exception detection is an important aspect of exception handling. Try to identify the cycle in which the following exception can be detected for the multicycle DataPath. The steps of multicycle CPU is shown below.

Step name	Action for R-type instructions	Action for memory-reference instructions	Action for branches	Action for jumps
Instruction fetch	IR <= Memory[PC] PC <= PC + 4			
Instruction decode/register fetch	A <= Reg [IR[25:21]] B <= Reg [IR[20:16]] ALUOut <= PC + (sign-extend (IR[15:0]) << 2)			
Execution, address computation, branch/jump completion	ALUOut <= A op B	ALUOut <= A + sign-extend (IR[15:0])	if (A == B) PC <= ALUOut	PC <= (PC [31:28], (IR[25:0]),2'b00))
Memory access or R-type completion	Reg [IR[15:11]] <= ALUOut	Load: MDR <= Memory[ALUOut] or Store: Memory [ALUOut] <= B		
Memory read completion		Load: Reg[IR[20:16]] <= MDR		

a. Overflow exception (EX 执行完检测，有学生说在 MEM 阶段也可以，因为 EX 阶段得到结果，在 MEM 检测；)

b. Divide by zero exception. (Assume we use the ALU for division in on cycle) (EX，这个在计算前就知道，所以一定是 EX 阶段；)

c. Invalid instruction (ID)

d. External interrupt (Every stage is OK，我们设计中可以规定只在 IF 阶段或者结束执行检测)

e. Invalid instruction memory address (IF)

f. Invalid data memory address (EX 或者 MEM 阶段，看学生解释，因为 EX 阶段地址算出来了， 而 MEM 使用地址，都是合理的；)