

计算机组成原理

Homework 4

Made by TA

2023 年 3 月 29 日

注意：

1. 本次作业提交 DDL 为 4.5 下午 2:00 之前，与第三周作业一起提交。超过该时间的提交会被扣除一定的作业分数。
2. 不加特殊说明的情况下，所有的寄存器、地址均为 32 位宽度的。**以下题目均在 RV32I 指令集中讨论。**

题目 1. 计算题：

1. 将 13.34375 按照 IEEE 754 标准转换为双精度浮点数。
2. 计算 $9/(-4)$ 、 $(-9)/4$ 、 $(-9)/(-4)$ 、 $9\%(-4)$ 、 $(-9)\%4$ 、 $(-9)\%(-4)$ 。

注：可以先根据 PPT 上的信息自己猜测，再通过 C 代码进行验证。(事实上，这些式子的计算结果是符合直觉的。)

3. 在 8 位 有符号数意义下，计算 $51*5$ 、 $100*8$ 、 $51*(-5)$ 、 $(-100)*8$ ，要求用十进制写出结果 (更多拓展见思考题 1)。

题目 2. 指令题 (执行结果应写上所有影响的寄存器，包含 pc 寄存器)：

1. 将伪指令 `li x5, 0x00789abc` 翻译为两条真实指令的组合。
2. 当前 pc 在 `0x00003000` 位置，执行指令 `0xffffe297` 的结果是什么？

3. 执行完上一问后，下一条指令是 0x00c28067，执行结果是什么？
4. 考虑数组 `int r[8][8]`，假设 `int` 占四个字节，数组基址在 `x5`，请写出汇编程序，使得 `x8 = r[x6][x7]`。
注：不考虑越界，其它寄存器任意使用。

提示：前三问建议利用 RARS 验证答案 (用法简介：Edit 页面新建文件后写汇编代码，Run - Assemble 转化成二进制码，在 Execute 页面执行)。

实验题 1. 编译与执行 (本题只需要在提交的 pdf 中提供**第二问**的完整代码，但建议自行验证结果。RARS 默认数据段开头为 0x0000，可直接 Execute 页面双击修改)：

1. 假设 `int` 占四个字节，数组 `int a[100]` 基址在 `x5`，请写出代码使 `a[x6]>a[x7]` 时交换 `a[x6]` 与 `a[x7]` 的值，否则不变。
注：不要改变 `x5` 到 `x7` 的值，可使用 `x8` 到 `x11`，以方便下一问。
2. 如上问条件，请在框架下写出汇编程序，完成对 `a` 的冒泡排序 (从小到大)。

BEGIN:

```
addi x13, x0, 0
```

```
addi x12, x0, 99
```

LOOP1:

```
beq x12, x13, END
```

```
# TODO (可自行添加标签)
```

LOOP2:

```
# TODO (可自行添加标签)
```

LOOP2END:

```
addi x13, x13, 1
```

```
jal x0, LOOP1
```

SWAP:

```
# TODO (利用上一问)
```

END: nop

思考题 1. 补码的真相 (为方便, 以下均考虑 8 位有符号整数, 更多位时情况类似):

1. 众所周知, 有符号整数在计算机中是以**补码**的形式存储的。也即, 第一位为 0 时, 其看作无符号数的字面值 n 即为其值 x , 也即 $x = n$, 当第一位为 1 时, 需要按位取反后加 1, 再将得到的结果看作无符号数后添加负号。请写出此时 n 与 x 的关系。
注: 用十进制的简单算式表示。
2. 复习代数结构所学的内容: 若 $a \equiv c \pmod{m}, b \equiv d \pmod{m}$, 证明 $a + b \equiv c + d \pmod{m}, ab \equiv cd \pmod{m}$ 。(提示: 可写为 $a = mx + c, b = my + d$ 后计算)
3. 填空: x 是与 n 同余于 ____ 的, 在范围 [____, ____] 内的唯一整数。
4. 利用前两问说明, 将两操作数看作无符号数与看作有符号数进行加、减、乘时, 结果的字面值必然一样。(提示: 先计算不考虑溢出时的结果 k , 而最后显示的结果 x 与 k 的关系与上一问相同。)
5. 给出一个检测有符号数乘法溢出的可能方法。

思考题 2. 编译优化初探 (改编自陈意云、张昱《编译原理》习题 9.22):

1. 假设 `int r[20][10]` 的基址在 `x5`, `int` 占四个字节, 允许使用乘法 (规则见 ppt 或手册), 请写出下面代码的一个汇编表示:
`for (i = 0; i < 20; i++) for (j = 0; j < 10; j++) r[i][j] = 10 * i * j;`
2. 如何用不含乘法的指令集实现相同的功能? (这一编译优化方式称作**强度削弱**, 将复杂运算替代为简单运算)
提示: 注意相邻项之间简单的加法关系, 且每行相邻项的增量自身也等差。
3. 如果从每种不同的高级语言 (如 C、Python) 到课上学习的每种不同的指令集, 都需要设计一套编译方式, 新增高级语言/指令集的效率是过于低下的。有什么办法可能解决这个问题? (提示: 上一问的优化过程与指令集有关吗?)

Tips: 思考题不纳入作业评分, 但助教仍建议花一些时间思考以加深理解。我们鼓励大家进行思考, 对于有创意、有深度的答案我们会在习题课上进行分享。