

（深圳）

实验报告

开课学期： 2022春季

课程名称：计算机组成原理（实验）

实验名称： 从C语言到机器码

实验性质： 综合设计型

实验学时： 2 地点：

学生班级： 计算机类4班

学生学号： 200110428

学生姓名： 杨杰睿

作业成绩：

实验与创新实践教育中心制

2022年3月

|  |
| --- |
| 1、实验结果截图 |
|  |
| 2、汇编代码注释（只需写主程序和子程序即可） |
| 示例：    将堆栈指针寄存器sp与立即数(-16)相加，再存入堆栈指针寄存器sp，即sp = sp + (-16)  **注：更为规整的格式请参见out/main.s代码行末注释！**      .file   "main.c"      .option pic      .text      .align  1      .globl  multiply      .type   multiply, @function  multiply:      mv  a5,a0      # 将参数寄存器a0的值复制到参数寄存器a5，即x保存到a5      li  a0,0      # 加载立即数0到寄存器a0，即result初始化为0      j   .L2      # 无条件跳转到.L2标签  .L3:      addw    a0,a0,a4      # 将参数寄存器a4与参数寄存器a0的值相加，结果复制到a0中，即计算result加上a4寄存器中的值，a4寄存器的值取决于(y&1)的结果，详见.L2标签      slli    a0,a0,48      # 将参数寄存器a0的值逻辑左移48位，结果复制到a0中，即丢弃寄存器a0从17位到64位的值，即保证下一步计算后result的结果是uint16\_t      srli    a0,a0,48      # 将参数寄存器a0的值逻辑右移48位，结果复制到a0中，即将寄存器a0的高48位写为0，结果保存在低16位中，即保证result的结果是uint16\_t      srli    a1,a1,1      # 将参数寄存器a1的值逻辑右移1位，结果复制到a1中，即将a1寄存器的值整除2，即源代码中语句 y >>= 1      slliw   a5,a5,1      # 将参数寄存器a5的值逻辑左移1位，结果复制到a5中，即将a5寄存器的值乘以2，即源代码中语句 x <<= 1      slli    a5,a5,48      # 将参数寄存器a5的值逻辑左移48位，结果复制到a5中，即丢弃寄存器a5从17位到64位的值，即保证下一步计算后x的类型是uint16\_t      srli    a5,a5,48      # 将参数寄存器a5的值逻辑右移48位，结果复制到a5中，即将寄存器a5的高48位写为0，结果保存在低16位中，即保证x的类型是uint16\_t  .L2:      beqz    a1,.L6      # 若参数寄存器a1的值等于0，就跳转到.L6标签，即y等于0的时候跳转到.L6      andi    a4,a1,1      # 将a1的值和立即数1进行“按位与”运算，将结果复制到a4寄存器中，即计算(y&1)的值      beqz    a4,.L3      # 若寄存器a4的内容值等于0，就直接跳转到.L3标签，即(y&1)等于0的时候跳转到.L3      mv  a4,a5      # 将寄存器a5的值复制到寄存器a4中，即将x的值保存到寄存器a4      j   .L3  # 无条件跳转到.L3标签  .L6:      ret # 函数返回，跳转到上层调用者处，返回值在参数寄存器a0中，即返回值为result      .size   multiply, .-multiply      .section    .rodata.str1.8,"aMS",@progbits,1      .align  3           # 对齐为8 bytes  .LC0:      .string "%u\n"      # 字符串"%u\n"      .text      .align  1           # 对齐为1 byte      .globl  main      .type   main, @function  main:      addi    sp,sp,-16      # 将堆栈指针寄存器sp与立即数(-16)相加，再存入堆栈指针寄存器sp，即sp = sp + (-16)      sd  ra,8(sp)      # 将ra寄存器的内容写入sp所指向地址加8的偏移量，即将上级调用者的返回地址写入方才分配的栈空间最高的双字中（栈空间总共分配了2个双字，当前最高位的空间写入了ra）      li  a1,28      # 将立即数28加载到参数寄存器a1中，即本例源码中的调用multiply(x ,x)的第2个x参数      li  a0,28      # 将立即数28加载到参数寄存器a0中，即本例源码中的调用multiply(x ,x)的第1个x参数      call    multiply      # 将main部分下一条需要执行的指令地址写入ra寄存器，调用函数multiply，返回值y保存在a0参数寄存器中      li  a1,28      # 将立即数28加载到参数寄存器a1中，即本例源码中的调用multiply(y ,x)的第2个x参数，y的值保存在a0中      call    multiply      # 将main部分下一条需要执行的指令地址写入ra寄存器，调用函数multiply，返回值result保存在a0参数寄存器中      sext.w  a1,a0      # 将a0参数寄存器的值复制到a1中，进行32位符号扩展(sign extend word)，此处应该是打印时的格式化指示符%u所致      lla a0,.LC0      # 将.LC0标签地址加载到参数寄存器a0中，即将字符串"%u\n"地址复制到a0中      call    printf@plt      # 将main部分下一条需要执行的指令地址写入ra寄存器，调用函数printf@plt，返回值读取的参数个数保存在a0参数寄存器中      li  a0,0      # 将立即数0加载到参数寄存器a0中，即main返回值为0，代表正常退出。      ld  ra,8(sp)      # 将堆栈指针寄存器sp所指向高8字节地址的值写入ra寄存器中，即将保存的上级调用者的返回地址从栈空间的最高双字取出复制到ra中      addi    sp,sp,16      # 将堆栈指针寄存器sp与立即数16相加，再存入堆栈指针寄存器sp，即sp = sp + 16      jr  ra      # 返回上级调用者，返回地址为ra，返回值为a0，即main函数返回上级并返回0      .size   main, .-main      .ident  "GCC: (Ubuntu 9.4.0-1ubuntu1~20.04) 9.4.0"      .section    .note.GNU-stack,"",@progbits |
| 3、机器码注释（只需写主程序和子程序即可） |
| 示例：    1141：二进制为0001 0001 0100 0001  fun3：000，imm：110000，rd/rs1：00010，op：01  c.addi指令：sp = sp + (-16) |