

# 深圳大学实验报告

课程名称: 计算机系统(3)

实验项目名称: MIPS64 乘法器模拟实验

学 院: 计算机与软件学院

专 业: 计算机与软件学院所有专业

指导教师: 刘刚

报告人: 林宪亮 学号: 2022150130 班级: 国际班

实 验 时 间: 2024 年 10 月 10 日~10 月 13 日

实验报告提交时间: 2024 年 10 月 13 日

## 一、实验目标：

实际运用 WinMIPS64 进行试验，以期更了解 WinMIPS64 的操作；  
更加深入地了解 MIPS 程序的语法；  
深入地了解在计算机中乘法的实现以及加法与乘法之间的关系。

## 二、实验内容

按照下面的实验步骤及说明，完成相关操作记录实验过程的截图：

首先，我们使用加法操作设计一个不检测溢出的乘法操作；完成后，我们对此进行优化，以期获得一个可以对溢出进行检测的乘法操作。（100 分）

## 三、实验环境

硬件：桌面 PC

软件：Windows，WinMIPS64 仿真器

## 四、实验步骤及说明

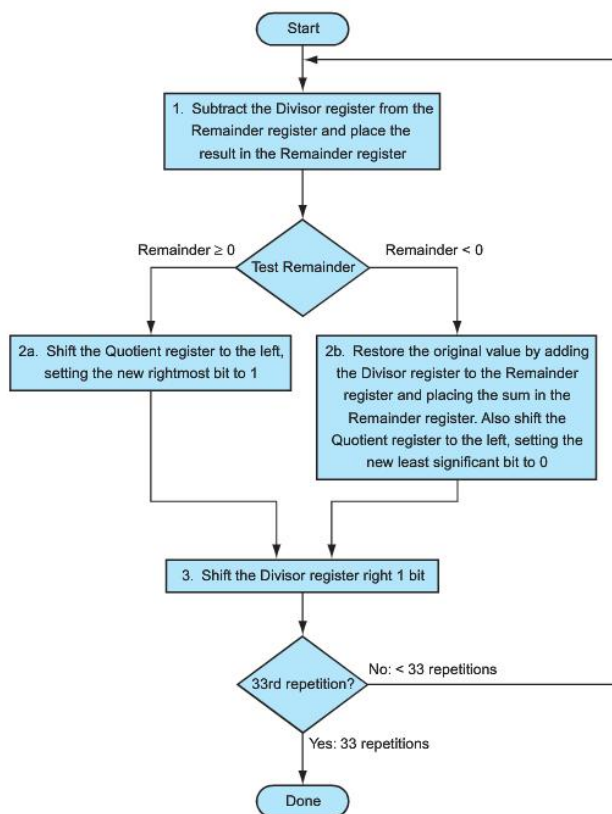
本次试验分为两个部分：第一部分、用加法器设计一个不考虑溢出的乘法器；第二部分、用加法器设计一个考虑溢出的乘法器（编程熟练的同学，也可以用除法器、浮点加法器等替代）。

### 1、忽略溢出的乘法器

首先，我们得了解乘法器如何由加法器设计得到，此处，我们以 32 位乘法为例。

总共分为 4 步：

1. 测试乘数最低位是否为 1，是则给乘积加上被乘数，将结果写入乘积寄存器；
2. 被乘数寄存器左移 1 位；
3. 乘数寄存器右移一位；
4. 判断是否循环了 32 次，如果是，则结束，否则返回步骤 1。



运行显示运行结果的例子如下，由于我们这里展示的是忽略了溢出的乘法，所以结果有两种：1、小于 32 位；2、大于 32 位。

第一种情况截图：

```

Terminal
please enter two numbers:
12
12
result:
144
  
```

第二种情况截图：

```

Terminal
please enter two numbers:
10000000
10000000
result:
276447232
  
```

根据上面的程序代码和截图，我们可以很清楚的看出，当结果小于32位时，结果正常；当结果大于32位时，结果只截取了低32位的结果，而高32位的结果直接忽略掉了。

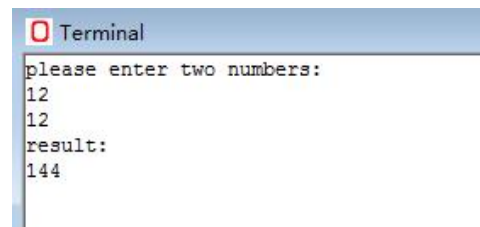
## 2、溢出提示的乘法器

上述的程序，用加法实现了 32 位乘法，但是，其中，对溢出情况没有进行考虑是其中的弊端。这里，我们来完善上述的乘法器，使得该乘法器会在结果溢出时候提示。

其实，这个小优化是十分简单的，只需要对 64 位的寄存器中的高 32 位进行检测即可。当高 32 位为 0 时，说明结果没有溢出，否则，结果溢出。

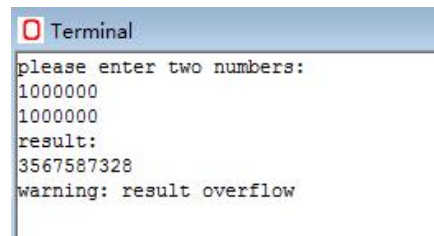
上述代码运行结果也有两个，一个是没有溢出的情况下的结果，一个是溢出了的情况下的结果。

首先，我们看没有溢出的情况结果：



```
Terminal
please enter two numbers:
12
12
result:
144
```

结果正确，其次，我们看溢出的情况结果如何：



```
Terminal
please enter two numbers:
1000000
1000000
result:
3567587328
warning: result overflow
```

可以看到，当结果溢出时，程序会给出提示“warning: result overflow”。

## 4 结束语

本实验介绍了通过加法器来设计乘法器的原理，并且在编写该实验程序的时候，我们更加了解了：1、计算机乘法器工作原理的内容；2、进一步熟练 MIPS 的编程方法；3、WinMIPS64 的使用方法。当然，如果想要更加深入的学习，我们也可以课外继续编写对除法的模拟。Perf 软件的使用让学生初步熟悉性能测评的主要工具。

## 五、实验结果

### 1. 忽略溢出的加法器

首先，我们可以通过加法器来设计乘法器，以 32 位乘法为例，具体步骤如下：

- **检测乘数最低位：**如果乘数的最低位为 1，则将被乘数加到当前乘积，并将结果写入乘积寄存器；

- **左移被乘数：**将被乘数寄存器中的值左移 1 位；

- **右移乘数：**将乘数寄存器中的值右移 1 位；

- **循环判断：**检查是否已经循环了 32 次，如果是，则运算结束；如果不是，则返回步骤 1，继续执行。

该过程的流程图可以参考图 1。

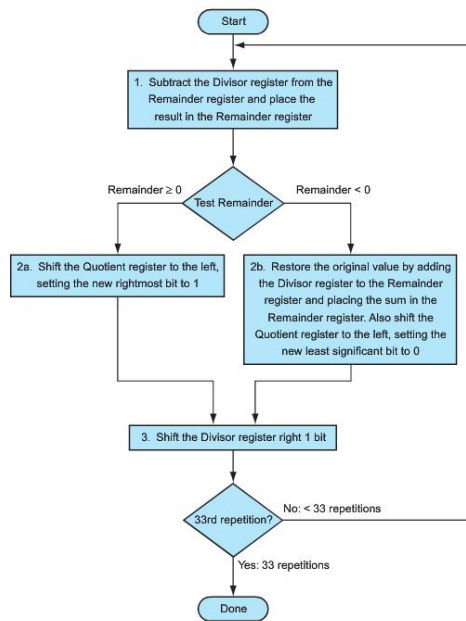


图 1 流程图

代码解释:

首先初始化一些数据,包括要输出的两个字符串提示信息和两个内容映射地址,如图 2 所示。

```

.data
CONTROL: .word32 0x10000
DATA: .word32 0x10008
str1: .asciiz "LXL,please enter two numbers:\n"
str2: .asciiz "results:\n"

```

图 2 数据初始化

然后写代码输出这两个字符串,代码如图 3 所示。

```

.text
daddi r1,r0,str1      # 将提示字符串地址加载到r1中 ("LXL,please enter two numbers:")
lw r2,DATA(r0)        # 从DATA寄存器读取地址
sd r1,0(r2)           # 将提示字符串的地址存入数据寄存器,显示提示用户输入数字
daddi r1,r0,4         # r1 = 4,表示下一个操作
lw r2,CONTROL(r0)     # 从CONTROL寄存器读取地址
sd r1,0(r2)           # 将4写入控制寄存器,通知硬件等待用户输入

```

图 3 输出字符串

然后需要获取被乘数和乘数,将 CONTROL 的值改为 8,获取整数输入,如图 4 所示。

```

daddi r1,r0,8         # r1 = 8,表示即将读取第一个输入 (a)
lw r2,CONTROL(r0)     # 从CONTROL寄存器读取地址
sd r1,0(r2)           # 将8写入控制寄存器,通知硬件准备读取输入
lw r2,DATA(r0)        # 从DATA寄存器读取第一个输入的地址
lw r3,0(r2)           # 将用户输入的第一个数加载到r3寄存器 (a)

daddi r1,r0,8         # r1 = 8,表示即将读取第二个输入 (b)
lw r2,CONTROL(r0)     # 从CONTROL寄存器读取地址
sd r1,0(r2)           # 将8写入控制寄存器,通知硬件准备读取输入
lw r2,DATA(r0)        # 从DATA寄存器读取第二个输入的地址
lw r4,0(r2)           # 将用户输入的第二个数加载到r4寄存器 (b)

```

图 4 获取被乘数和乘数

接着开始乘法计算过程，如图 5 所示，首先测试乘数的最低位是否为 1。如果是，则将被乘数加到当前乘积，并将结果写入乘积寄存器；如果不是，则跳过这一加法操作，直接进入下一步。下一步是将被乘数寄存器左移 1 位，同时将乘数寄存器右移 1 位。最后，判断是否已经完成了 32 次循环，如果是，计算结束；如果不是，则返回第一步，继续执行这一过程。

```

loop:   andi r2,r4,1      # 检查r4的最低位是否为1 (r4[-1])
        beq r2,r0,zero    # 如果r4[-1] == 0, 跳转到zero
        dadd r5,r5,r3     # 如果r4[-1] == 1, 将r3加到r5中 (累加被乘数)

zero:   dsll r3,r3,1      # 将被乘数左移1位 (r3 << 1)
        dsra r4,r4,1      # 将乘数右移1位 (r4 >> 1)
        daddi r1,r1,-1    # r1减1, 循环计数器减一
        bne r1,r0,loop    # 如果r1 != 0, 继续循环

```

图 5 乘法计算

最后把 CONTROL 改为 2，输出乘积结果，如图 6 所示。

```

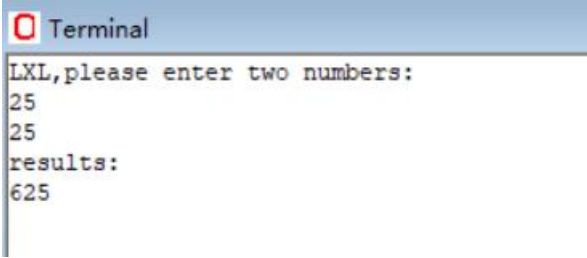
lw r2,DATA(r0)          # 从DATA寄存器读取地址，用于输出乘积
sd r5,0(r2)              # 将乘积r5的值存入数据寄存器，输出a * b的结果
daddi r1,r0,2            # r1 = 2, 表示操作结束
lw r2,CONTROL(r0)       # 从CONTROL寄存器读取地址
sd r1,0(r2)              # 将2写入控制寄存器，通知硬件输出结束

halt                     # 停止程序执行

```

图 6 输出乘积

代码测试结果：



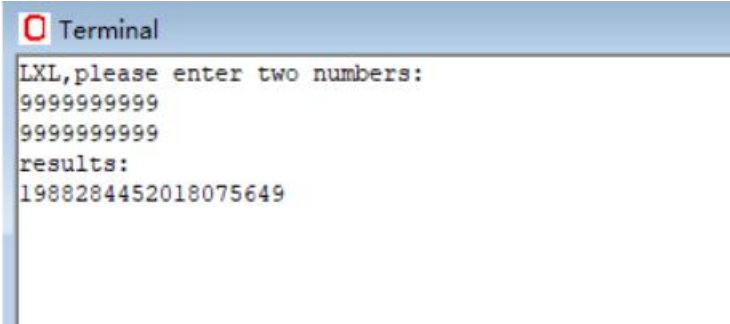
```

Terminal
LXL,please enter two numbers:
25
25
results:
625

```

图 7 正常答案

如图 7，当乘积不大至溢出时，答案正确。



```

Terminal
LXL,please enter two numbers:
999999999
999999999
results:
1988284452018075649

```

图 8 错误答案

如图 8，当答案大至溢出时，会输出错误的答案。

## 2. 检查溢出的乘法器

上述程序通过加法实现了 32 位乘法，但存在一个缺陷，即没有考虑结果溢出的情况。为了完善这个乘法器，我们可以优化它，使其在发生溢出时给出提示。  
这个小优化非常简单，只需要检测用于存储结果的 64 位寄存器的高 32 位。当高 32 位为 0 时，表示结果未溢出；如果高 32 位不为 0，则表示结果溢出，此时程序会输出溢出警告。

我们可以在数据段中添加一条提示字符串，如图 9 所示，用于在溢出时显示相关信息。

```
.data
CONTROL: .word32 0x10000    # 控制寄存器地址, 用于与硬件通信
DATA:    .word32 0x10008    # 数据寄存器地址, 用于输入和输出数据
str1:    .asciiiz ",LXL,please enter two numbers:\n"    # 提示用户输入两个数字的字符串
str2:    .asciiiz "results:\n"    # 输出结果的提示字符串
str3:    .asciiiz "warning: result overflow\n"    # 溢出警告的提示字符串
```

图 9 输出信息

我们可以通过对乘积寄存器进行算术右移 32 位来提取其高 32 位，以检测是否溢出。具体操作如图 10 所示，由于不能一次性右移 32 位，因此需要分两次移位：每次右移 16 位，最终得到高 32 位的值。接下来，判断这个高 32 位是否为 0。如果为 0，说明乘积未溢出；如果不为 0，则表示结果已经溢出，并输出提示字符串。

```
dsra r1,r5,16    # 将r5右移16位 (r5[0:31]) , 检查高位
dsra r1,r1,16    # 再次右移16位, 提取高16位
beq r1,r0,end    # 如果r1 == 0, 说明没有溢出, 跳转到end

daddi r1,r0,str3    # 将溢出警告字符串地址加载到r1中 ("warning: result overflow")
lw r2,DATA(r0)    # 从DATA寄存器读取地址
sd r1,0(r2)    # 将溢出警告字符串存入数据寄存器, 显示溢出警告
daddi r1,r0,4    # r1 = 4, 表示下一个操作
lw r2,CONTROL(r0)    # 从CONTROL寄存器读取地址
sd r1,0(r2)    # 将4写入控制寄存器, 通知硬件输出溢出警告

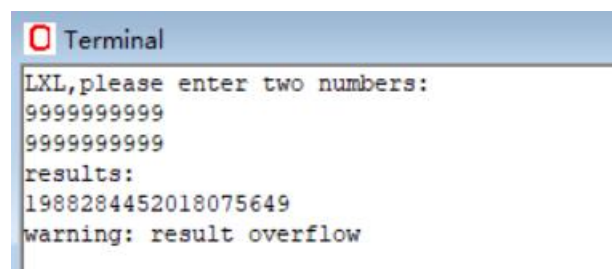
end:    halt    # 停止程序执行
```

图 10 溢出检测

通过这一优化，程序能够有效检测乘法结果是否超出 32 位范围，确保更完善的运算过程。

### 代码测试结果：

如图 11，当结果溢出时，会输出提示信息。



```
Terminal
LXL,please enter two numbers:
9999999999
9999999999
results:
1988284452018075649
warning: result overflow
```

图 11 溢出提示

## 六、实验总结与体会

在本次实验中，我通过设计 32 位乘法器深入理解了乘法的硬件实现方式。首先，我使用加法器逐位计算乘法，依次检测乘数最低位，再通过左移被乘数和右移乘数的操作完成整个乘法计算。在初步实现过程中，程序能够正常输出乘积结果，但当乘积超出 32 位时，未能检测到溢出，从而导致错误的结果输出。

为了解决这个问题，我对乘法器进行了优化。通过检测存储结果的 64 位寄存器的高 32 位，判断是否发生溢出。如果高 32 位不为 0，则提示结果溢出。这个改进有效提升了程序的健壮性，确保乘法器在处理大数时也能正确反馈溢出情况，并输出警告信息。

在实验过程中，我实现了以下关键功能：

- 32 位乘法的实现：通过加法和移位操作，使用逐位乘法算法实现乘法器，并成功输出乘积。
- 溢出检测优化：通过两次右移操作提取 64 位寄存器的高 32 位，判断是否发生溢出，确保运算结果的准确性。
- 提示信息的输出：当乘法结果溢出时，程序能够及时输出提示信息，避免错误的结果输出。

通过这次实验，我不仅掌握了乘法器的基本设计流程，还进一步理解了溢出处理的重要性。在硬件设计中，考虑到计算结果的溢出情况，可以提升计算的可靠性和准确性，这对复杂运算尤为关键。



指导教师批阅意见：

成绩评定：

指导教师签字： 刘刚  
2024 年 10 月 日

备注：

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。  
2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。