**深 圳 大 学 实 验 报 告**

|  |
| --- |
| **课程名称 计算机系统1**  **项目名称 实验一 LC-3仿真器安装和使用**  **学 院 计算机与软件学院**  **专 业 计算机科学与技术**  **指导教师**  **报 告 人 学号**  **实验时间 2024年4月11日**  **提交时间 2024年4月17日** |

**教务处制**

# 一、实验目的与要求

（1）掌握处理器仿真工具LC-3软件的安装和使用方法。

（2）学会在LC-3仿真环境下编辑程序和转换成可执行目标程序的方法 。

（2）学会在LC-3仿真环境下运行和调试程序的方法 。

# 二、实验内容与方法

利用提供的安装软件包和软件使用说明文档，完成以下试验内容：

（1）安装LC-3仿真器

（2）利用LC3EDIT输入机器代码程序（0/1模式）并创建可执行目标程序。

（3）利用LC3EDIT输入机器代码程序（hex模式）并创建可执行目标程序。

（4）利用LC3EDIT输入汇编代码程序并创建可执行目标程序。

（5）利用仿真器运用对应目标程序。

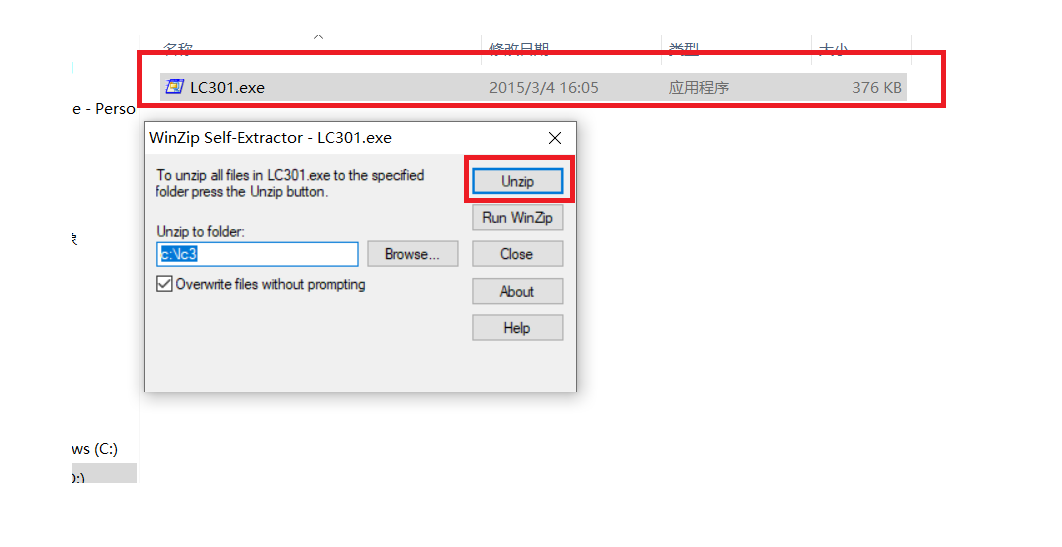
（6）学习和掌握断点，单步执行等调试方法和手段。

# 三、实验步骤与过程

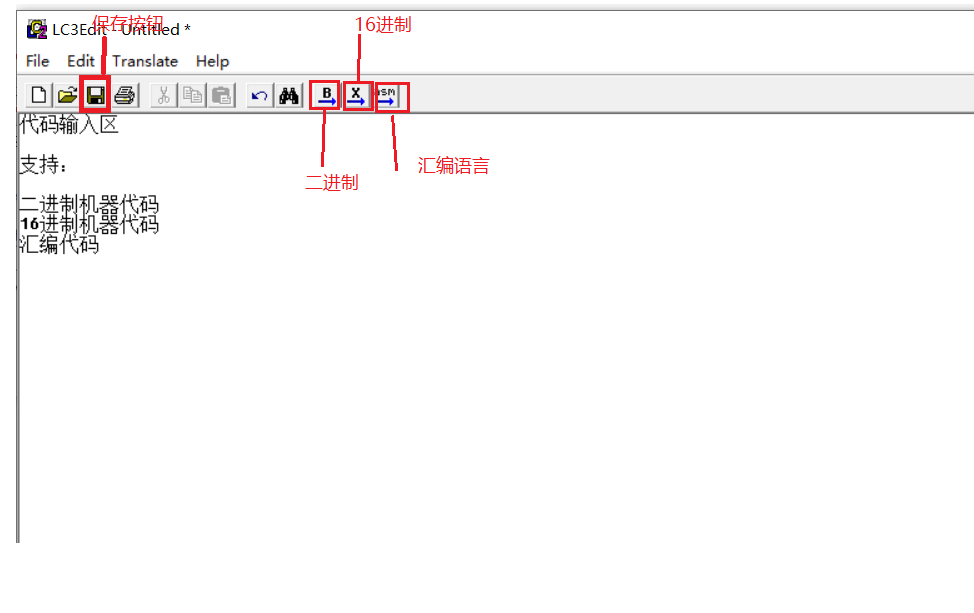
（依照实验内容，逐条撰写实验过程与实验所得结果：包括程序总体设计，核心数据结构及算法流程，调试过程。请附上核心代码，及注意格式排版的美观。实验提交时，以上为评分依据，请不删除本行）

**（一）、安装LC-3仿真器**

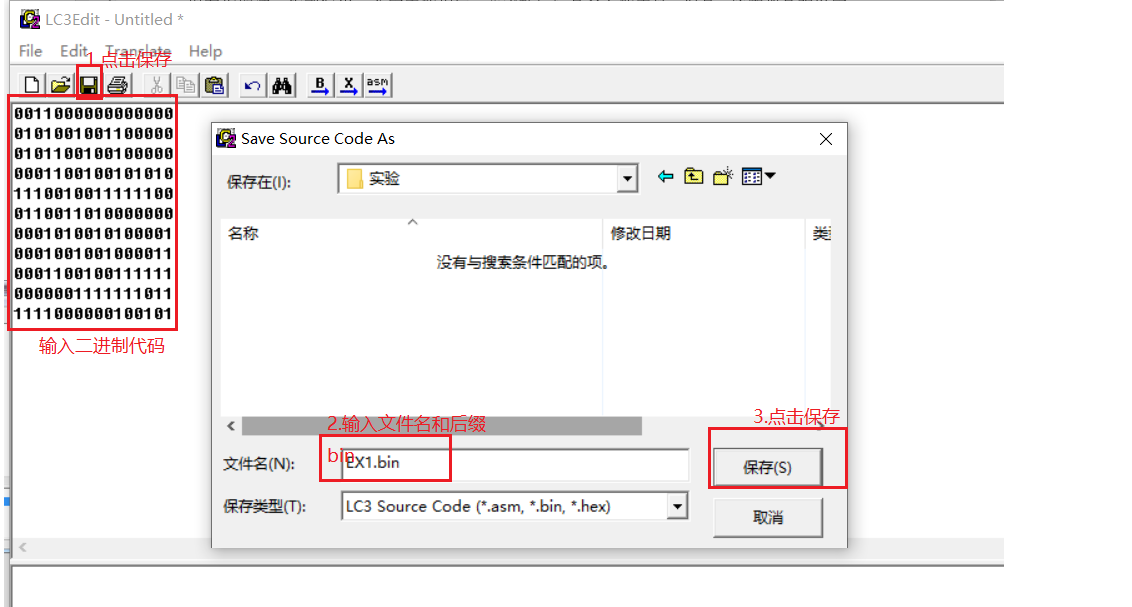
运行”仿真器” 目录下的LC301.exe文件，其实质是个winzip压缩包，选择目录，点击“Unzip”即可

****

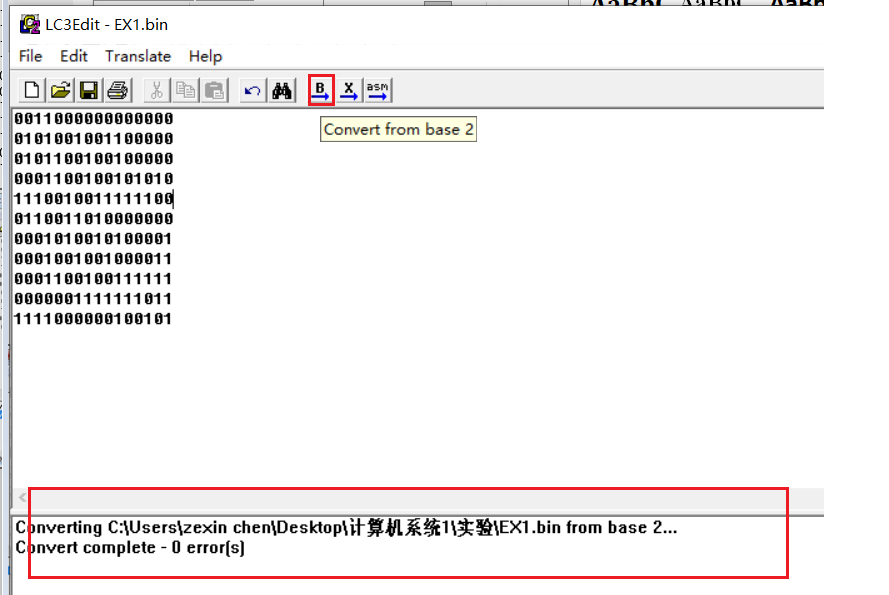
运行安装目录下的LC3edit.exe文件

**（二）、利用LC3EDIT输入机器代码程序（0/1模式）并创建可执行目标程序。**

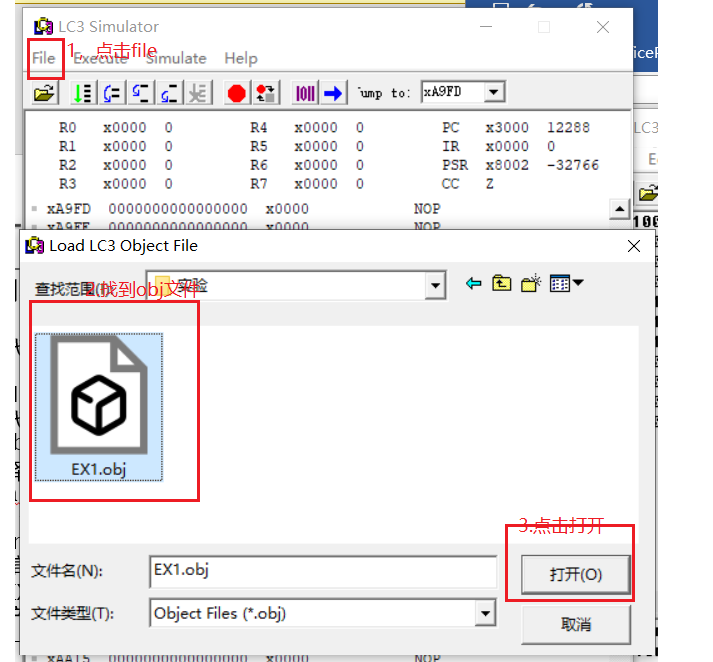
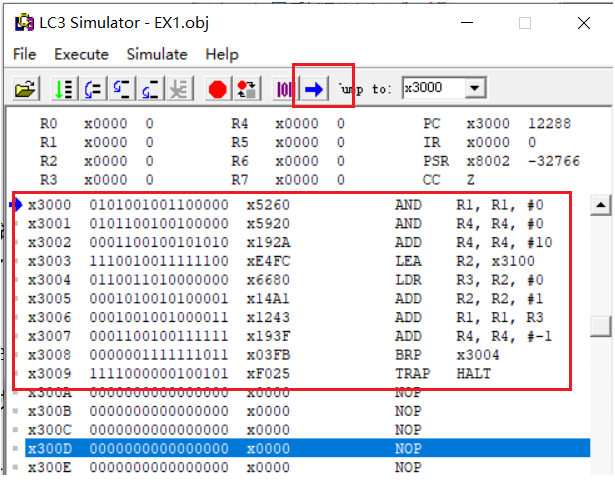
* 1. 在代码输入区，输入右侧代码。选择File->Save，命名EX1.bin文件。（二进制机器代码程序的后缀必需为.bin）



* 1. 编译，选择Translate🡪Convert Base 2。信息区域提示Convert Complete-0 error(s).的成功信息

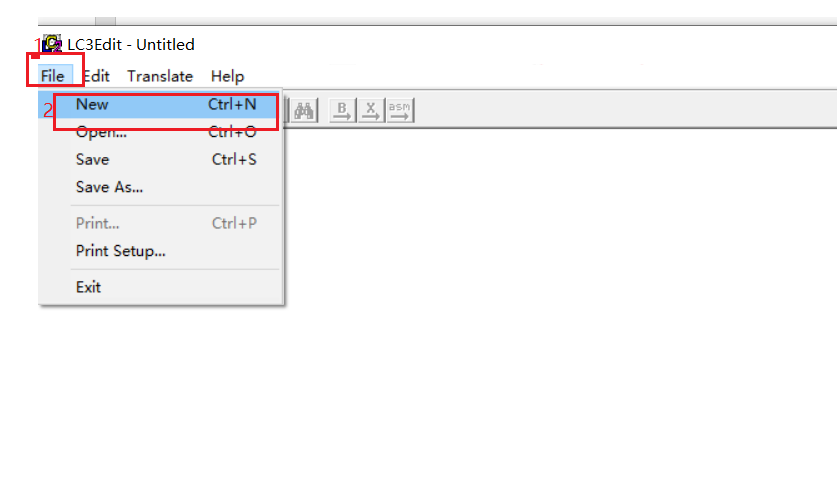
****

c)在EX1.bin对应的文件夹中会产生EX1.obj文件，可在LC-3 Simulator 中执行

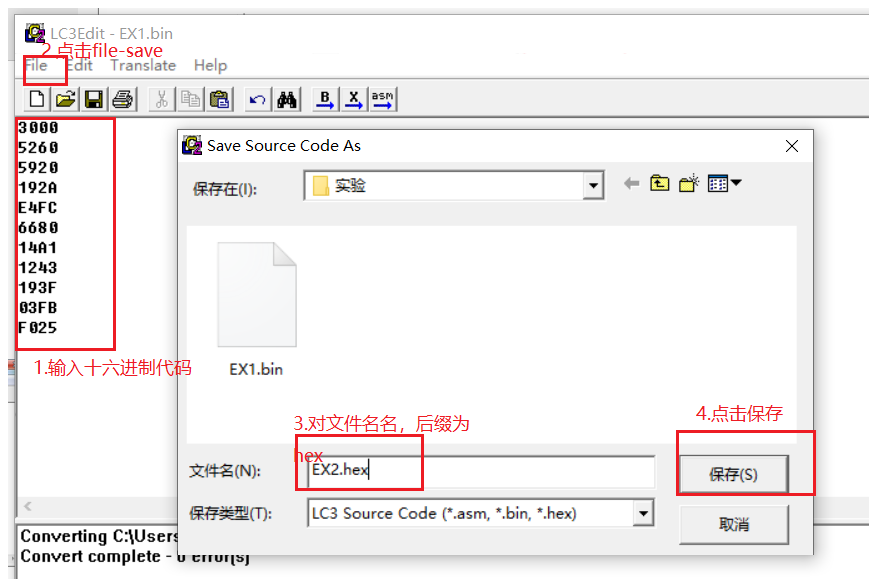
****

**（三）、利用LC3EDIT输入机器代码程序（hex模式）并创建可执行目标程序。**

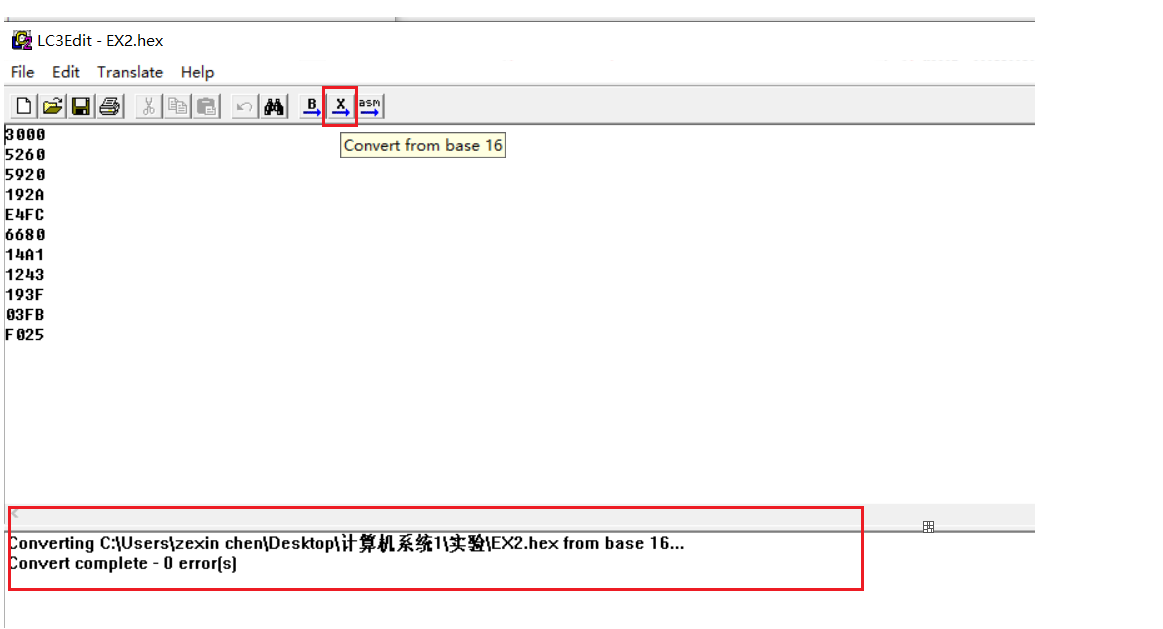
* 1. 新建文件，File->new

****

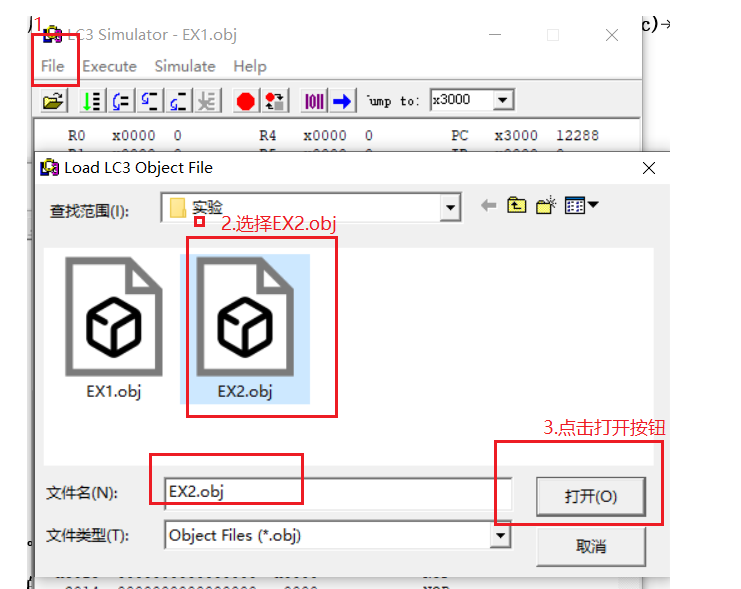
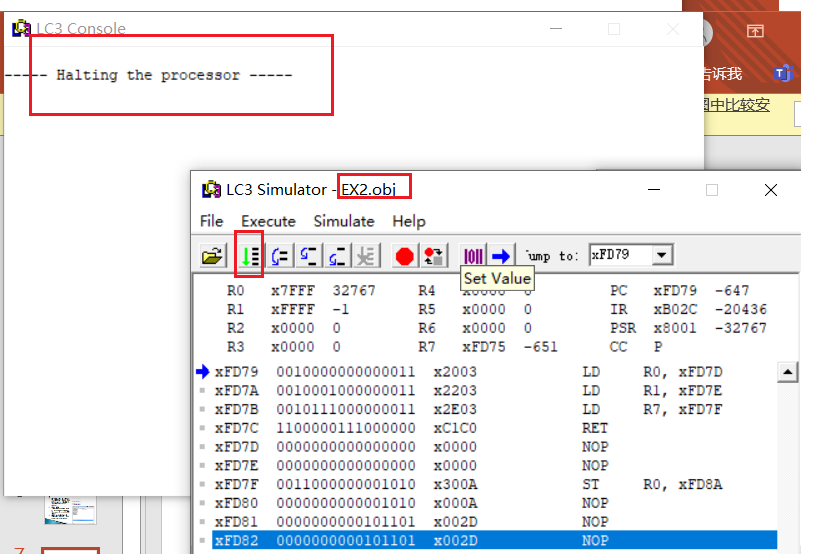
* 1. 在代码输入区，输入右侧代码。选择File->Save，命名EX2.hex文件。（二进制机器代码程序的后缀必需为.hex）

****

* 1. 编译，选择Translate🡪Convert Base 16。信息区域提示Convert Complete-0 error(s).的成功信息

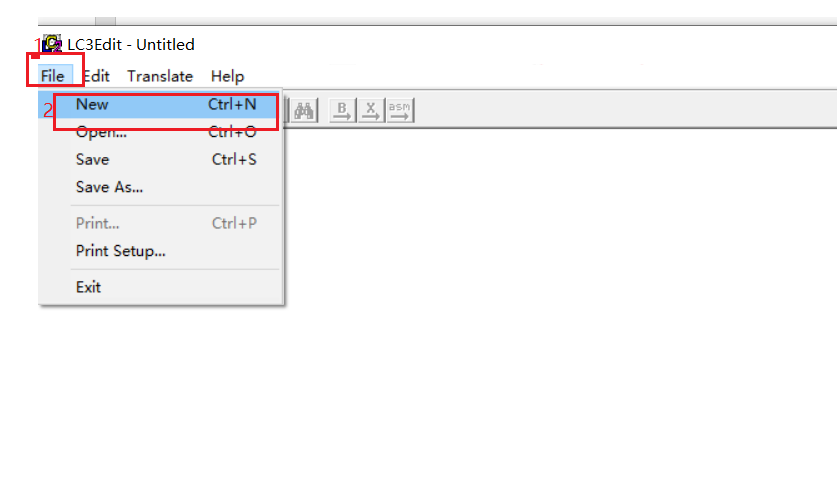
****

在EX2.hex对应的文件夹中会产生EX2.obj文件，可在LC-3 Simulator 中执行

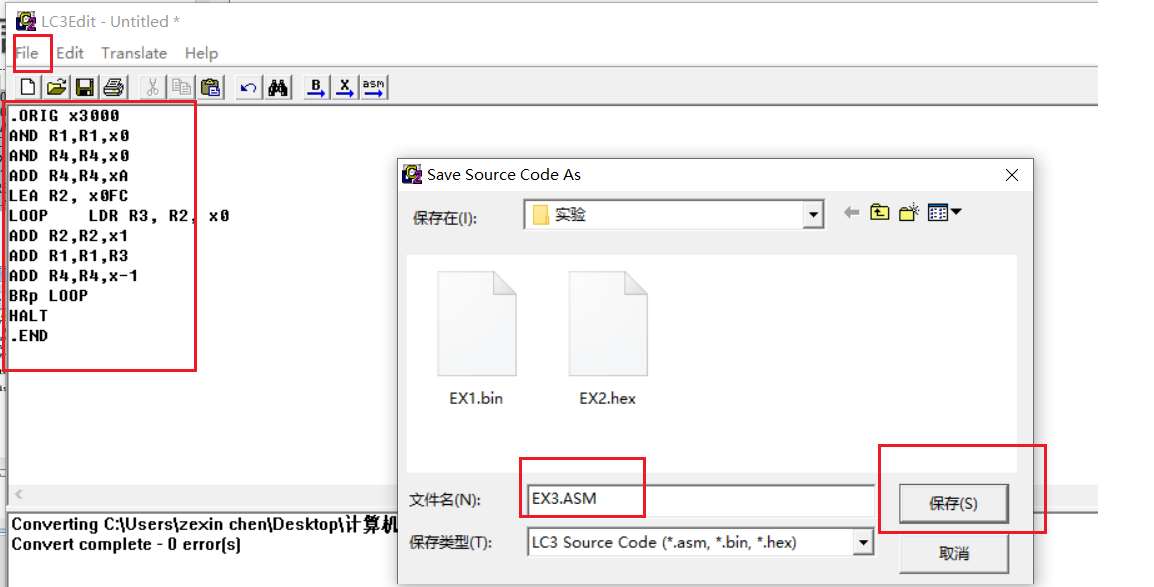
****

**（四）、利用LC3EDIT输入汇编代码程序并创建可执行目标程序。**

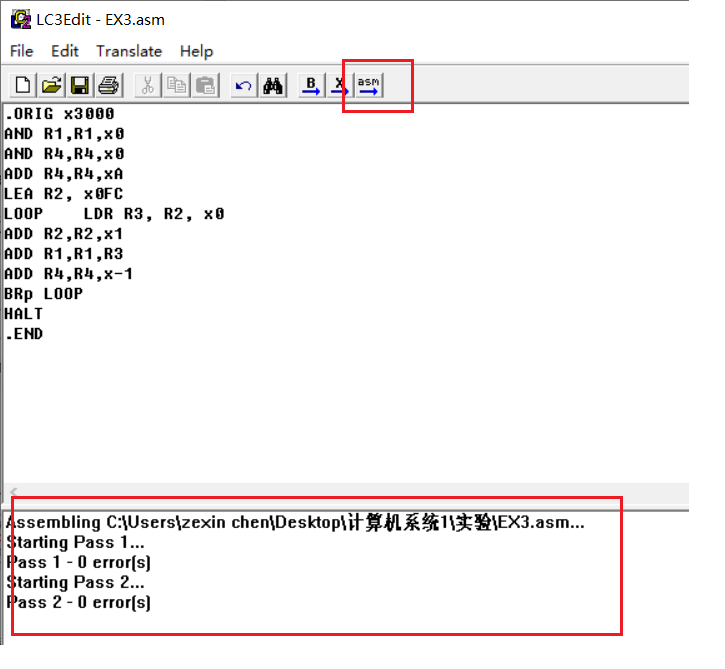
* 1. 新建文件，File->new

****

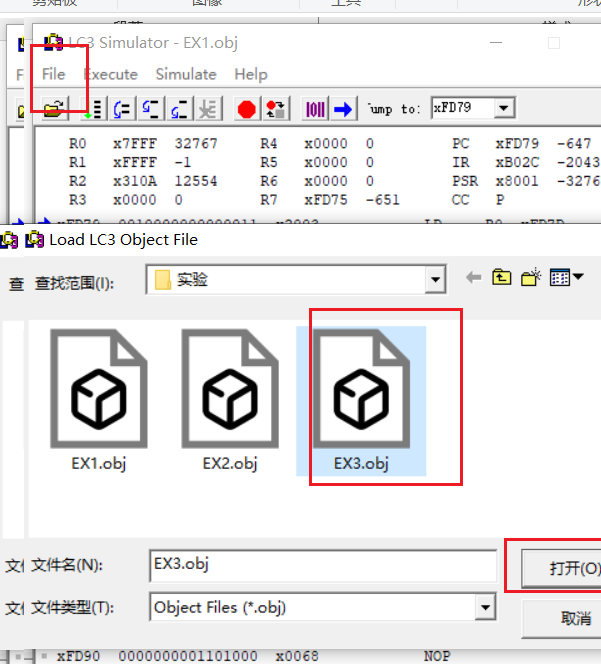
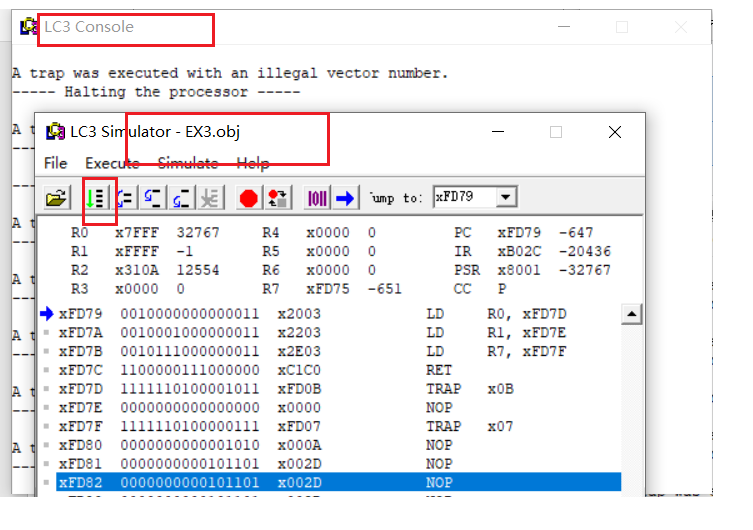
* 1. 在代码输入区，输入右侧代码。选择File->Save，命名EX3.ASM文件。（二进制机器代码程序的后缀必需为.asm）

****

* 1. 编译，选择Translate🡪Convert assemble。观察信息区域提示的成功信息。如果出错，请检查代码

****

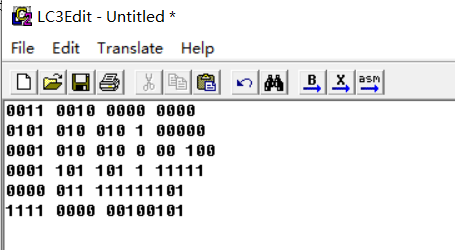
**在EX3.asm对应的文件夹中会产生EX3.obj文件，可在LC-3 Simulator 中执行**

****

**（五）、利用仿真器运用对应目标程序。**

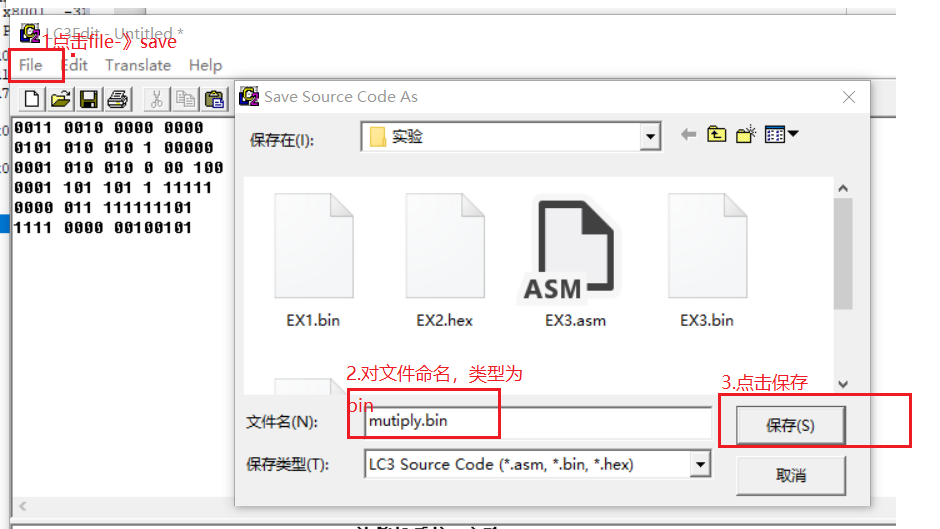
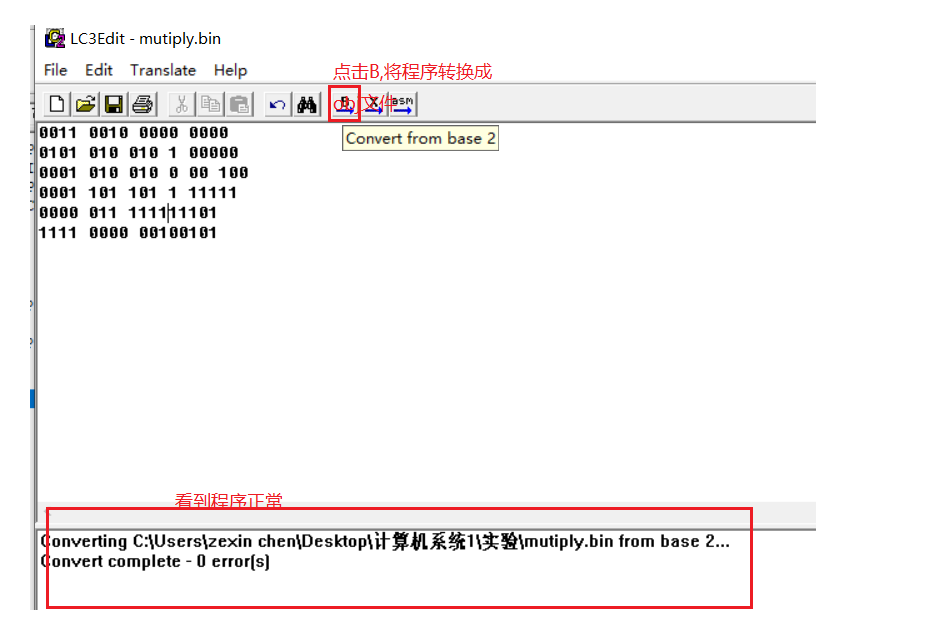
1. **Example1：调试这个不适用乘法指令的乘法程序**

**（1）输入程序： 首先需要在LC3Edit 中输入程序，如下显示：**

****

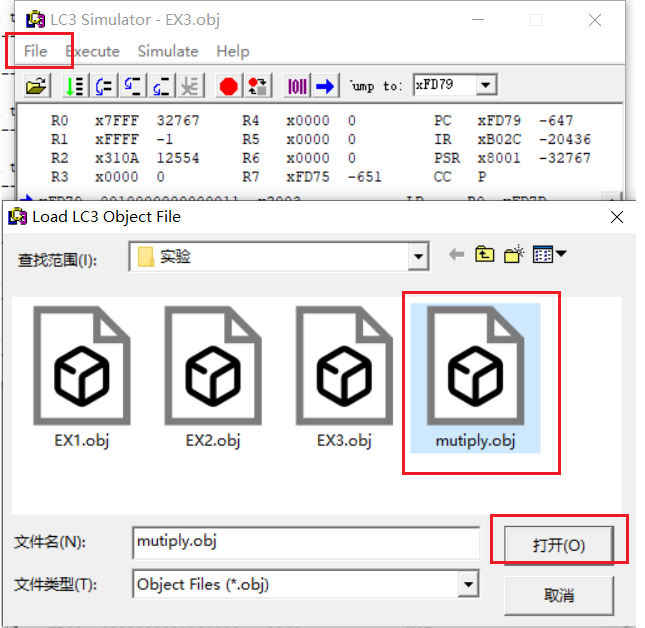
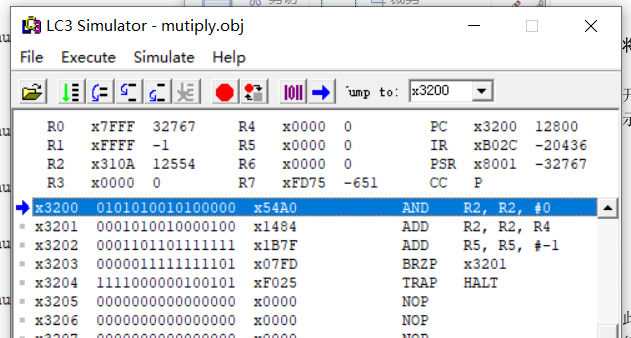
**（2）将程序转换为.obj格式**

当你 在LC3Edit 中输入完程序 并保存为如mutiply.bin 后，单击B 将程序加载如模拟器 转化为.obj 文件

****

**（3）将程序加载如模拟器**

启动模拟器，然后单击FILE加载你的程序：multiply.obj,现在模拟器的内存中部分内容显 示如下：

****

此时PC 值为x3200，蓝色箭头指向的那一行，即下一条即将执行的指令，因为程序尚未执 行，该指令也为你的程序的第一条指令。

**（4）在halt 指令处设置断点**

断点有很多中用途，我们将会解除到其中一些，最好养成在halt除设置断点的习惯，如果 不设置，程序就会运行到halt子程序然后结束，这时有可能会改变寄存器的值。因此，首 先在行x3204处设置断点，双击改行最前面的灰色方框：

未设置断点：

****

设置断点：

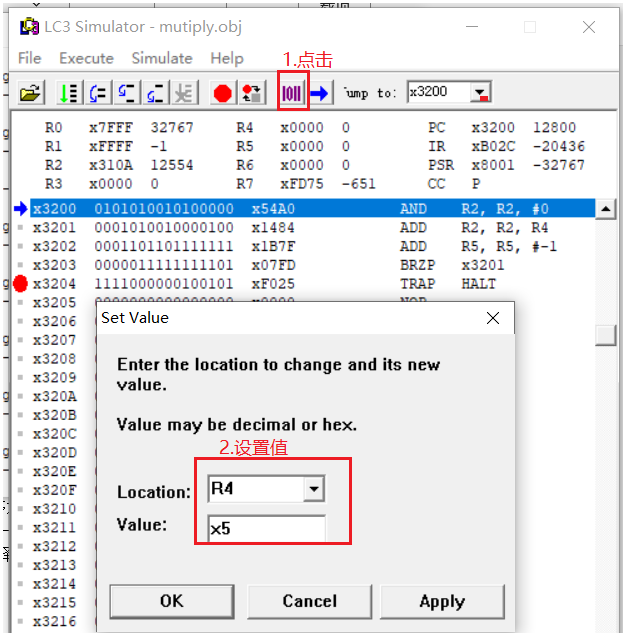
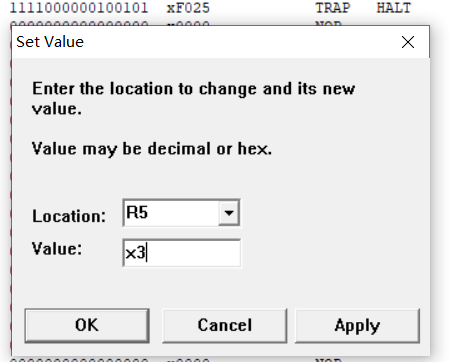
****

红色的标识表示该行存在断点，如果一个程序运行时，当PC值为x3204时，模拟器就 会暂停，等待进一步处理。

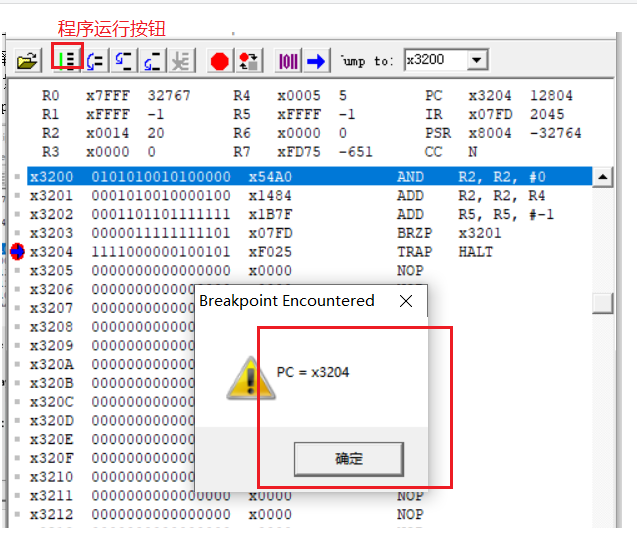
**（5）运行乘法程序**

在第一次运行程序之前，需要为R4、R5设置相应的值，以便他们相乘。怎样选择合适的值 以便利于测试，一般而言，0和1不是好的选择，如果为R5设置很大的值，后面就会循环 好多次，因此选择两个较小的不同的值，如5和3.

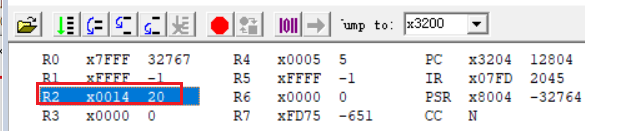
点击下图吧标出的紫色按钮，弹出Set Value 窗口，在Location字段中选择R4，在Value处输入“x5”， 点击Apply，然后选择R5，输入x3，点击OK，接下来运行程序。如下图

****

点击 程序运行程序，稍后，会弹出如下窗口：

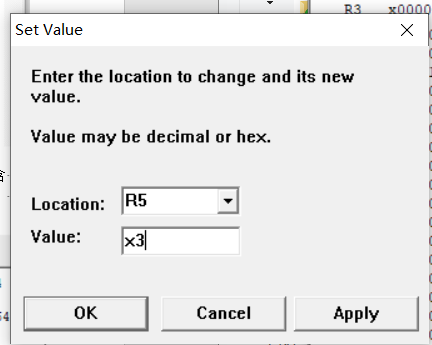
****

该窗口会弹出是因为在halt行处设置的断点，点击OK关闭窗口，查看R2，应该包含 最后的结果，十进制3\*5=15，但是R2中包含十进制20（十六进制x14)，程序存在问题， 下面发现问题所在

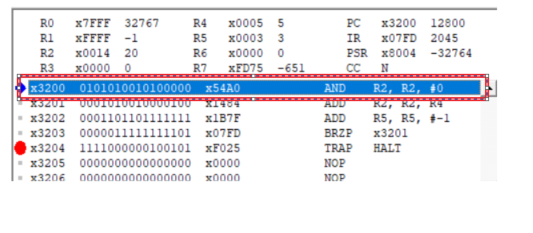
****

**（6）逐步调试乘法程序**

双击R5，然后在弹出的窗口中设置R5为x3，然后点击OK。

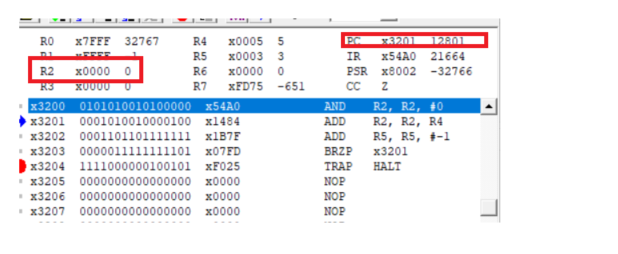


然后点击内存区域x3200处，接着点击set PC to select location按钮，设置PC值为x3200。

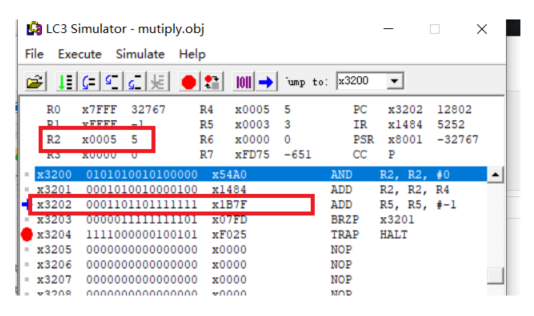


现在蓝色的箭头指向第一行，两个寄存器也已经初始化为目标值，接下来调试程序。

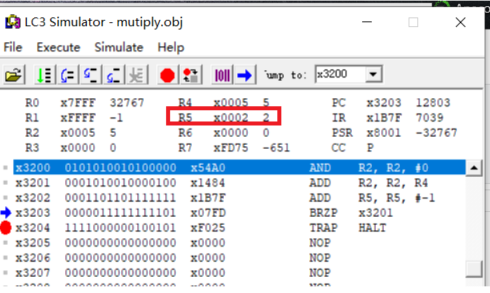
点击，Step Over，这时PC指向下一条指令X3201，IR中内容为第一条指令， X54A0，R2 被清零。



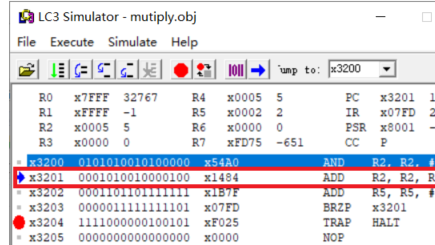
这些正是程序的期望目标值，下一步接着调试。 点击Step Over，PC和IR值如约改变，此时R2包含5（十进制和十六进制相同），运行正 确，继续。



接着点击Step Over，R5从x3变为x2，R5有两个作用，既是乘数，又是计数器(告诉模 拟器还要执行多少次循环，因此每次循环结束，R5自减），继续。



点击Step Over，触发分支指令的执行，每次分支指令执行，二者必选其一，此时，分支 被执行，因为状态码被add指令设置，add之后结果为x2，正数，因此状态码P为1，如果 状态码中Z或P为1，分支执行，因此分支被执行，此时PC指向X3201，等待下一次循环。

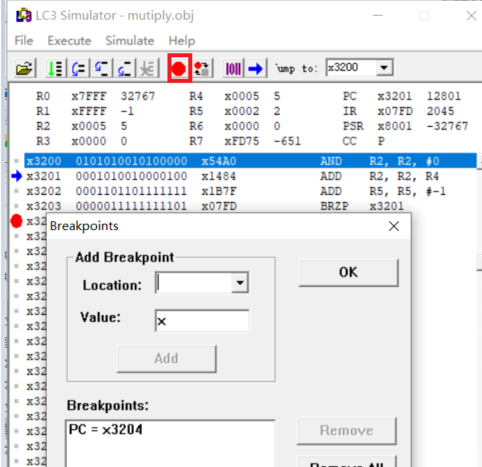


经过对程序的一次循环调试发现每条指令都没有问题，这样问题有可能在循环设置的地 方。

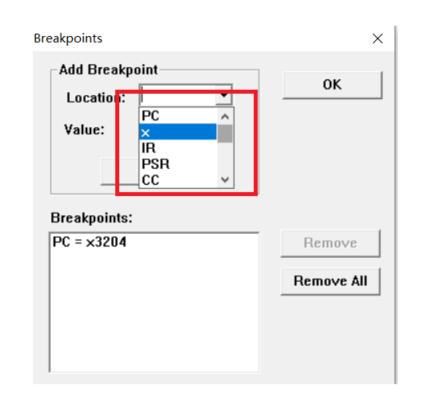
**（7）利用断点调试循环**

发现一个循环是否被过多执行的好方法就是在分支指令处设置断点。这样在每次循环迭代的 结尾处都会暂停，此时有利于查看寄存器的状态。

这里尝试另一种设置断点的方法，点击红色点 ，弹出如下窗口：

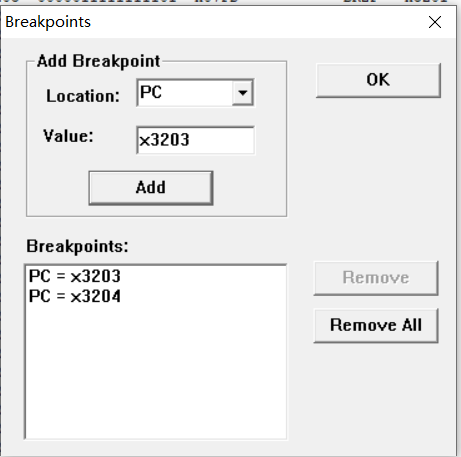


在Location 处点击下拉箭头，可以看到所有选项：PC X PSR IR CC R0 R1 R2 R3 R4 R5 R6 R7。

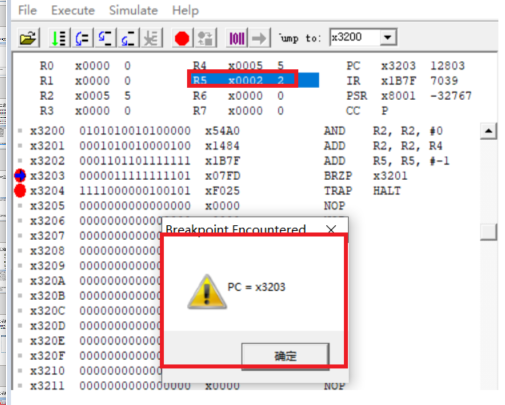


基本上，可以让模拟器在上述任意地方达到预设值时暂停运行，这样在R0的值为 X00FF 时，或者状态码Z为0时，或者内存区域x4000值为x1234时设置断点，程序就会 相应暂停。

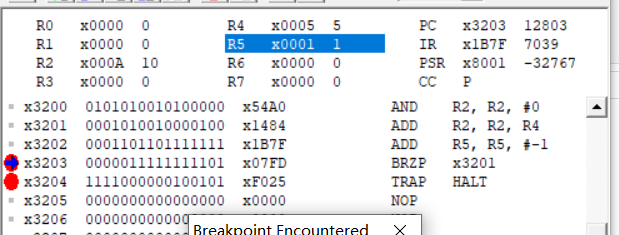
在本次调试中，设置PC为x3203，然后点击Add，这样断点列表中就有两条，两条都 和PC有关，在PC为x3203时或x3204时，模拟器都会暂停，然后点击ok即可。



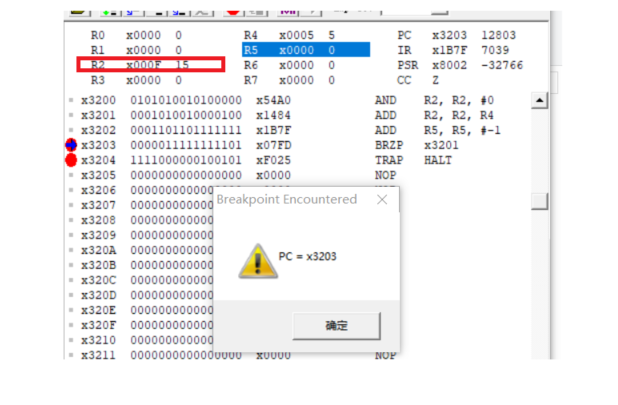
现在设置PC为x3200，R5为x3，然后点击运行程序窗口：点击OK，注意观察寄存器的值，蓝色箭头和PC均指向x3203，R4未变，R5则变为 x2，R2 变为x5,。状态码p为1，意味着继续执行程序时，分支会跳转。



点击 运行按钮，关闭弹出的窗口，如上次一样，观察寄存器，尤其是R2和R5，目前已 经循环两次，R2内容为x1，R2为十进制10，状态码P=1，因此循环将继续执行一次。

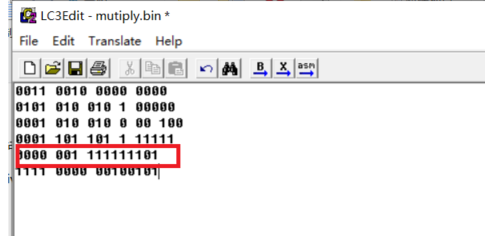


点击运行按钮，然后点击OK，此时R5为0，R2为十进制15，因为3\*5=15，此时应该停 止，但是状态码Z=1，分支指令将继续执行，多做一次，这里错了问题。

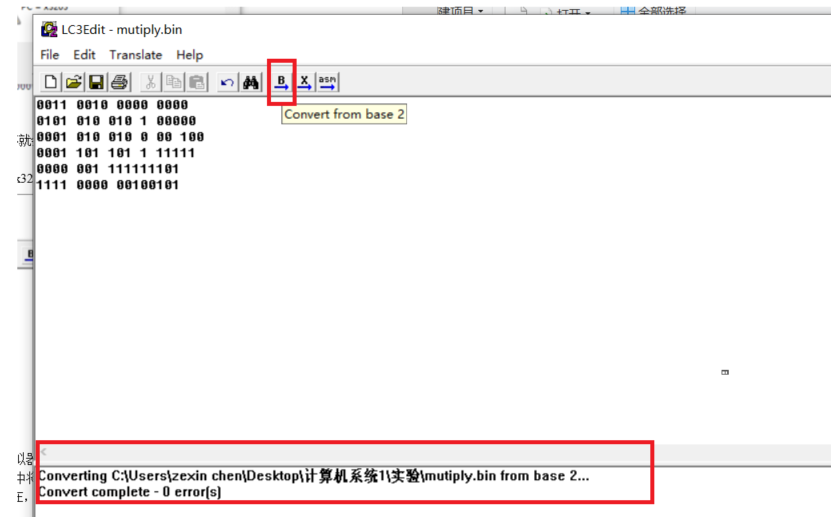


通过修改分支指令使只有当P=1时，循环就会执行正确的次数，为验证其正确性，用 LC3Edit 修改分支指令如下：

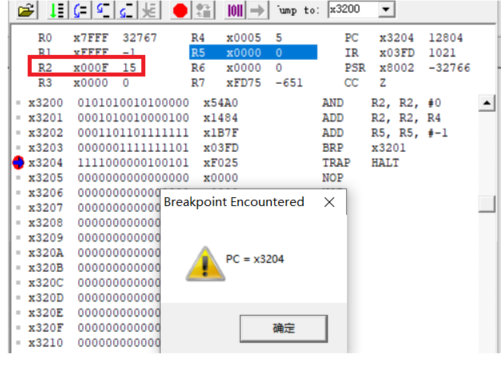
0000 001 111111101 ;跳转到 location x3201 如果结果为正值



保存并转成.obj格式，装载程序到模拟器，如果不想修改源代码，可以在模拟器中直 接修改，双击行x3203，在Set Value窗口中将其值从x0601改为x0201（此种方法仅对此 次装载有效，下次装载时，该bug仍然存在，因此还要修改源代码）。



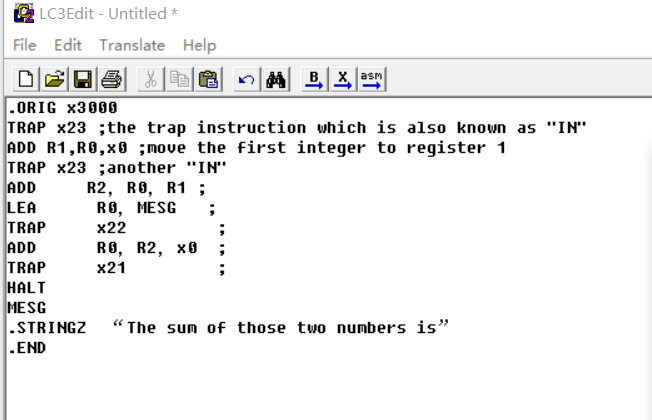
现在把PC值重设为x3200，R5设为x3，双击行x3202取消断点，点击，关闭 断点弹窗后，可以看到十进制15出现在R2中，程序调试成功。

****

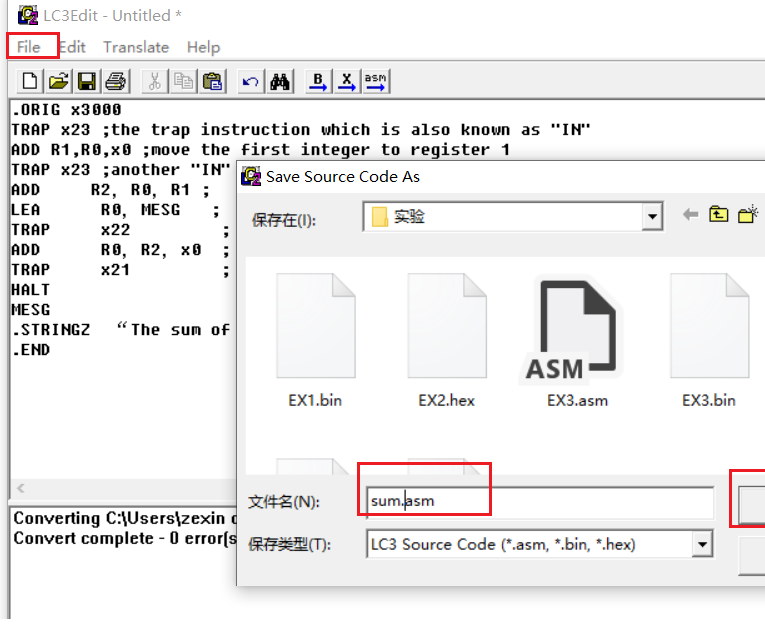
**Example 2：调试程序使其输入输入并求和**

**(1)在LC3Edit 中输入程序**

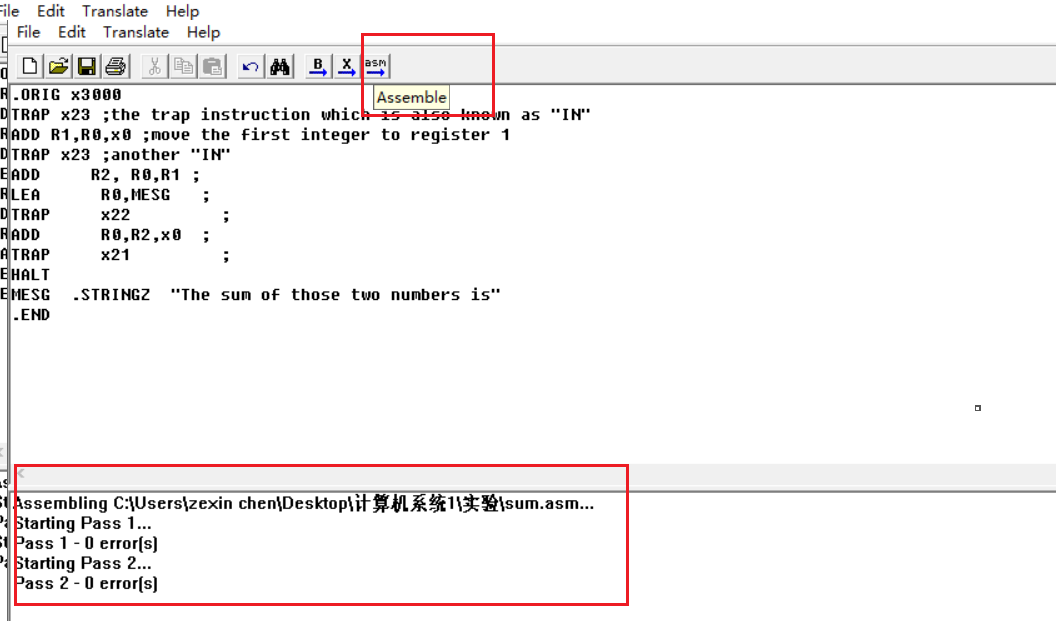
本程序的目的是让用户输入两个数（0到9），然后求和，然后打印（同样介于0和9） 在Console 窗口中，程序如下:



保存文件



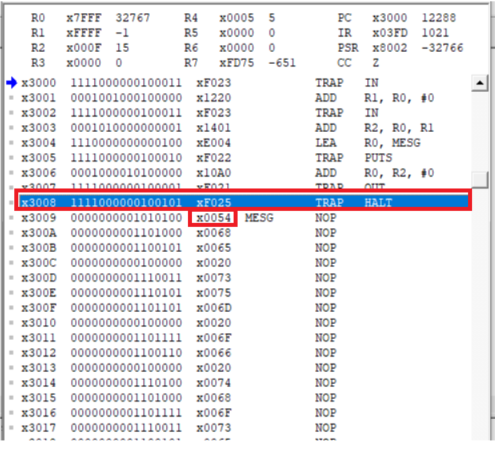
程序保存在在LC3Edit中，并通过点击ASM 来编译这个程序。



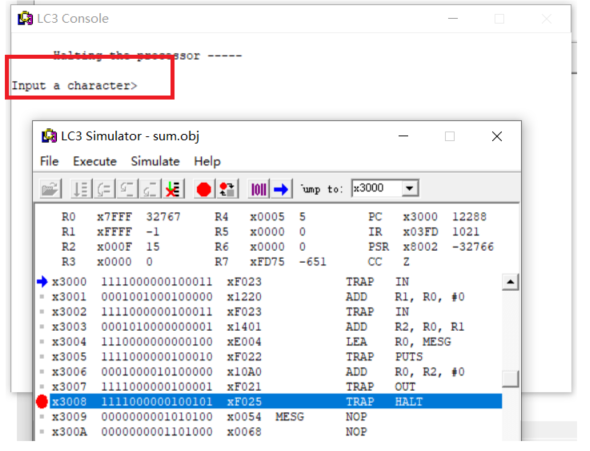
**（2）在simulator 中运行错误的程序**

**打开simulator，载入程序。**

注意：halt在x3008行，从x3009行开始，你会在每行都能看 到一个ASCII码值。在x3009行，你会看到x54，这个是表示字母T的ASCII码。在x300A 行，你会看到x68，代表“h”的ASCII码。整个字符串，“The sum of those two number is”存储在内存地址从x3009到x3028中，最后一个地址存储着空格。



双击x3008行前的小灰色方块来设置断点。现在点击 运行你的程序。第一个指令是陷阱 程序，提示你在console窗口输入字符，就像下面：

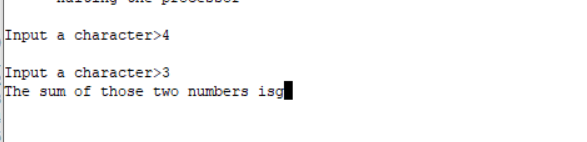


Simulator 在这个陷阱程序中会一直等待。（注意：“\_\_\_\_\_instructions executed”信息会 出现在simulator 窗口的下方，并且\_\_\_\_\_是快速增长的数字。知道你输入一个字符，它才 会停止无期限的变化。）通过点击console窗口，你可以使它变成活动窗口，然后输入0~9。 试试输入“4”。

注当输入“4”，R0中会出现x34。（寄存器R0保存的是你在IN陷阱程序中键盘的输 入。查看课本的附录E中的ASCII表，你会发现整数4在ASCII码中就是用x34表 示的。

程序的第二个指令是将x34传递给R1，然后提醒再次输入一个字符。因为这是一个 非常非常简单的程序，需要指定另一个整数使得它们相加的和最多为 9。所以再次点击 console 窗口，输入“3”。

一旦你输入了第二个数字，就可以在console 窗口中看到下面的信息：



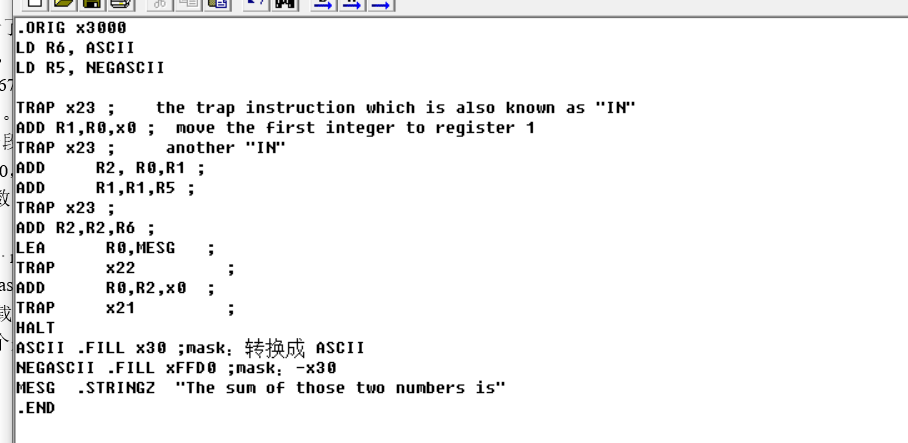
调试程序 关于为什么程序给了错误的结果，上面的一些段落给了一个大的暗示。记住，当在console 窗口输入“4”时，R0中给出的值是x34.当你输入的是“3”时，显示的是x33。把这些 值相加，结果是x67。查看ASCII表，x67代表的是“g”。所以你的输出也是有道理的，只是不是我们希望的。

只需要在程序中围绕数据段添加几行代码，确保它正确。对数字0~9的ASCII码需要做一 些窍门。“0”被表示为x30，“1”被表示为 x31。这种模式一直延续到“9”被表示为x39。 所以，我们怎么将x30从整数的ASCII值提取出来，得到它原来的数值呢？ 你只需要下面的数据段：

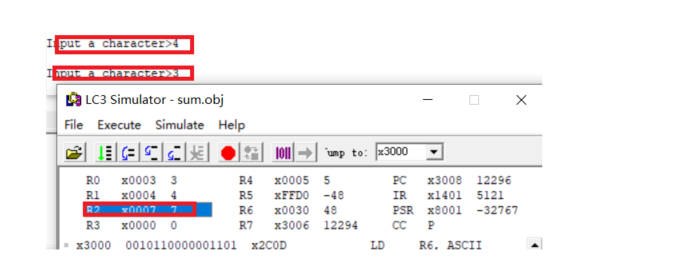
ASCII .FILL x30 ； mask 转换成 ASCII ；mask：ASCII

MEGASCII .FILL xFFD0 ;mask -30：

需要添加5个指令：两个是载入两个mask，一个是将-x30加到第一个数中，一个是对另一 个数做同样的操作，最后一个是在输出前将结果加上x30。程序现在是这样的



在LC3Edit 中保存程序，并且编译。在simulator中尝试这一版本，得到正确的结果。为3+4=7，在R2寄存器里面，如下图所示。



# 四、实验结论或体会

（撰写实验收获及思考）

通过“**LC-3仿真器安装和使用”**这个实验熟悉了LC-3仿真器的安装、编辑、转换和调试过程。通过输入不同格式的程序并转换成可执行目标程序，学会了如何利用LC3EDIT进行机器代码程序和汇编代码程序的编辑，并将其转换为可执行目标程序。同时，通过调试乘法程序和求和程序，学会了使用断点设置和逐步执行的调试方法，同时观察寄存器和内存中的值来排查问题。这个实验加深我对计算机体系结构和汇编语言的理解，同时也加深对程序执行过程的认识。

|  |
| --- |
| 指导教师批阅意见：   1. 寄存器写错 ：R4写成R22 ， 2. BR计算错误，本来应该跳转到x3005的，然后跳到了x2ffc 3. 最后计算的结果存储地址应该在x3101,计算偏移量的时候出错，存储在了x3102      1. 将代码修改正确后，设置x3100的值为x0011,即1的个数为8，个，但是运行后，发现存储在x3101位置的值为7个，预计是判断条件出现错误。     应该需要的是间接LDR,    成绩评定：  指导教师签字：  年 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。