**内蒙古农业大学**

**计算机与信息工程学院**

实 验 报 告

实验四

课程名称： 嵌入式基础实践

班 级： 21计科三班

姓 名： 周烨

学 号： 2021122156404

授课教师： 李慧旻

**实验四 ARM汇编指令**

1. **实验目的**
2. 通过实验掌握使用 ldm/stm，b，bl 等指令完成较为复杂的存储区访问和程序分支；
3. 学习使用条件码，加强对 CPSR 的认识；
4. **实验环境**
5. 硬件：PC 机。
6. 软件：µVision IDE for ARM 集成开发环境，Windows 98/2000/NT/XP。
7. **实验内容与结果**
8. 根据实验指导书3.1.6中 ARM汇编指令实验一的程序a和b，运行程序并观察寄存器的值。更改实验 A 中 X、Y 的值，观察执行结果。

在实验asml\_a.s中让X=45，Y=65

第一步：ldr sp, =stack\_top让sp=0x30200000

第二步：mov r0, #x 把x的值给 R0；

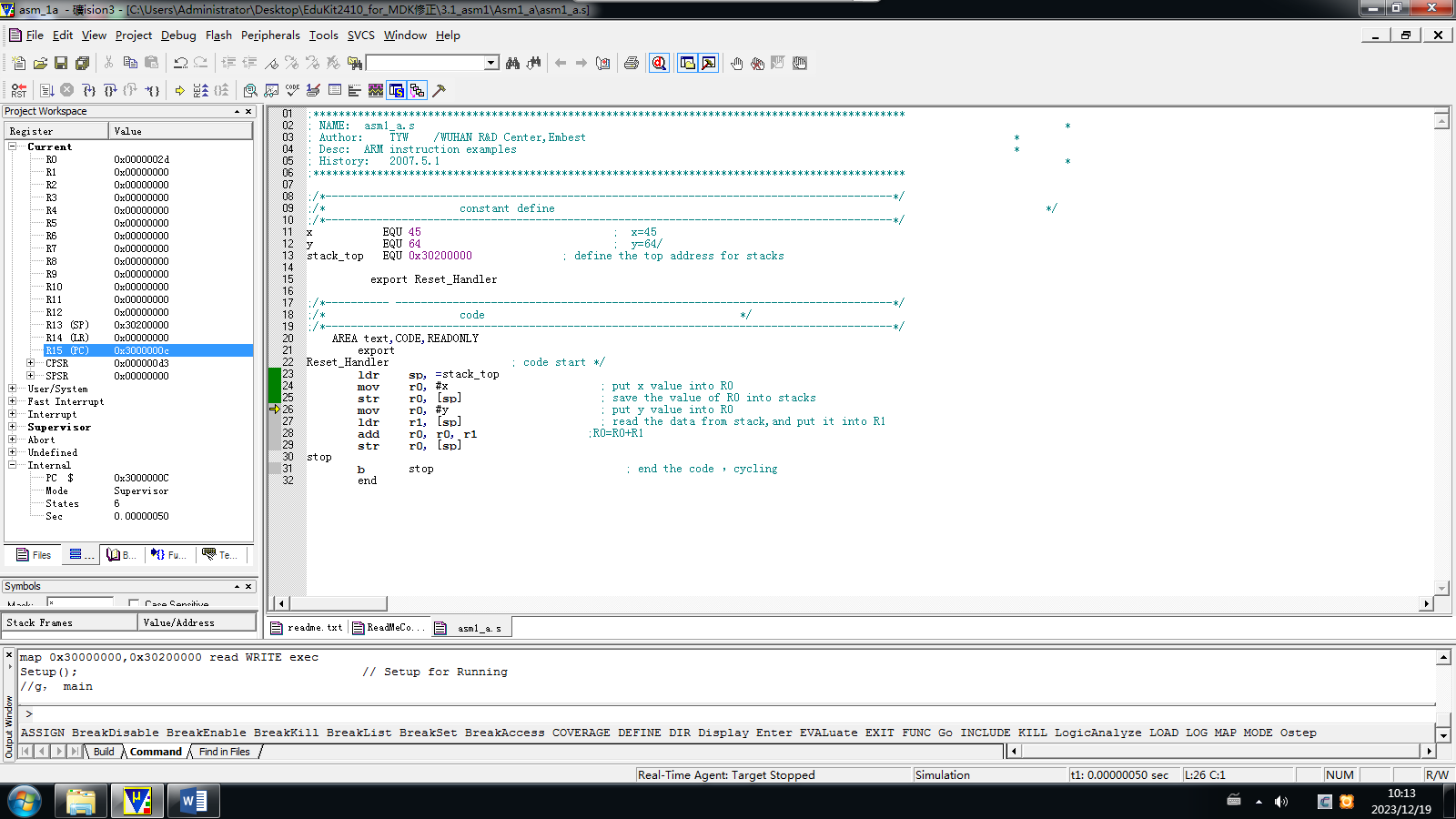
第三步：str r0, [sp] 将 R0 的值保存到栈中；把r0寄存器的值存到[0x30200000]这个地址的内存单元，因此这个内存单元中的值为0x0075。

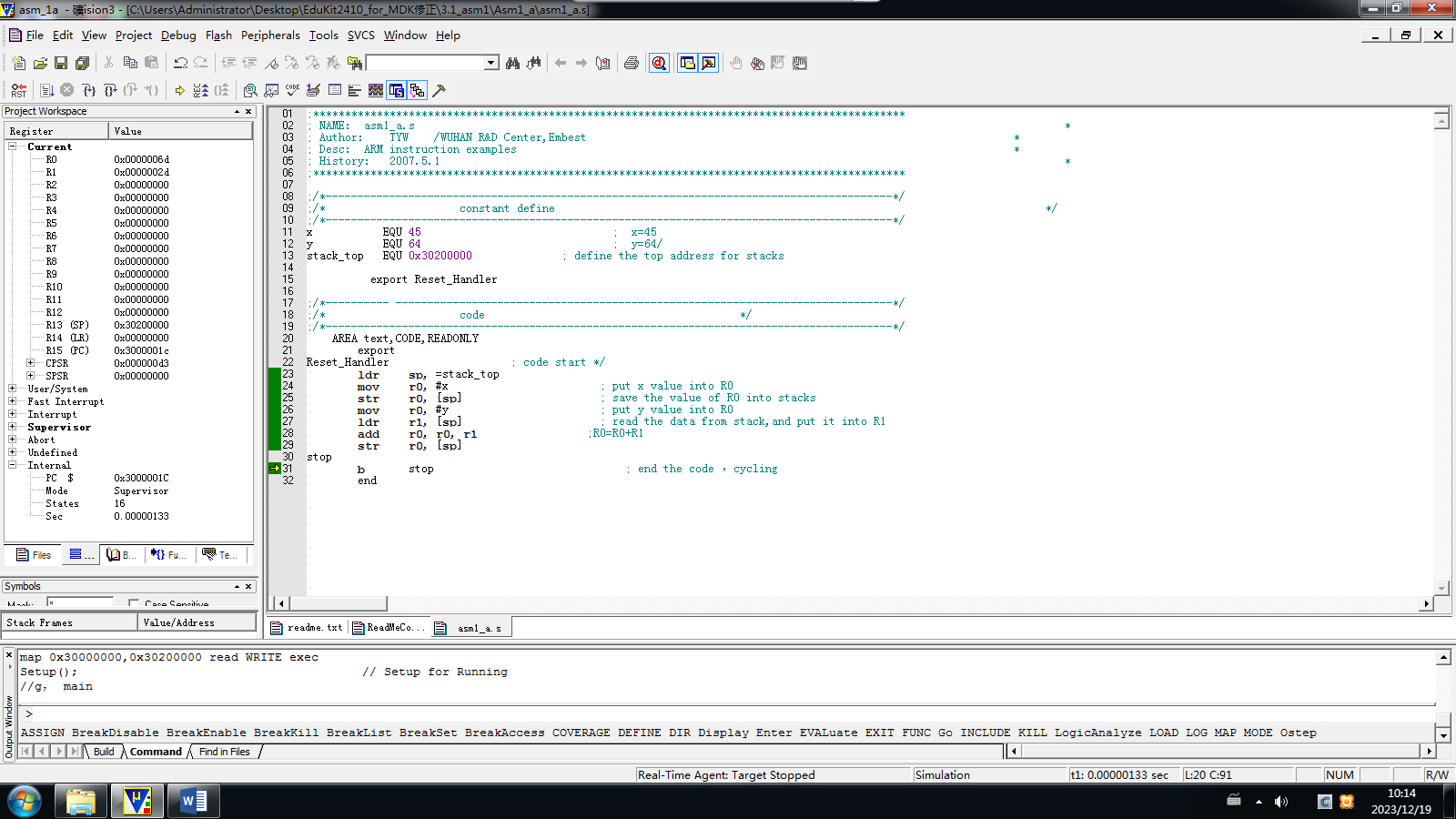
第四步：mov r0, #y把y的值给 R0；把Y的值变为16进制传送给r0寄存器。

第五步：ldr r1, [sp] 从堆栈中读取数据，并将其放入 R1；

第六步：add r0, r0, r1 把 R0 和 R1 的值加起来；

第七步：str r0, [sp]





在实验asm1\_b.s中让X=45 ,Y=64

第一步： step 1:mov r0, #x 把x的值放到 r0；X的值变为16进制给r0寄存器，之后r0寄存器并没有值。

第二步： mov r0, r0, lsl #8 x = x << 8 左移八位把r0的值右移8位也就是2进制数乘以256后变为16进制给r0寄存器.

第三步：mov r1, #y 把y的值放入 r1 中；把Y的值变为16进制赋值给r1寄存器.

第四步：add r2, r0, r1, lsr #1 R2 = (R1>>1) + R0；把r1寄存器的数右移1位与r0寄存器相加在存到r2寄存器。

第五步：ldr sp, =stack\_top让sp=0x30200000，栈顶地址为0x30200000。

第六步：str r2, [sp]把r2寄存器的值存放到0x30200000

第七步：mov r0, #z 把z的值放到 r0；把z的值变为16进制存到r0，

第八步：and r0, r0, #0xFF 从 r0 中获得低八位；

第九步：mov r1, #y y的值放入r1 中；把Y的值变为16进制存到r1，。

第十步：add r2, r0, r1, lsr #1 R2 =(R1>>1)+ R0；把r1寄存器的值右移一位与r0寄存器的相加存到r2寄存器

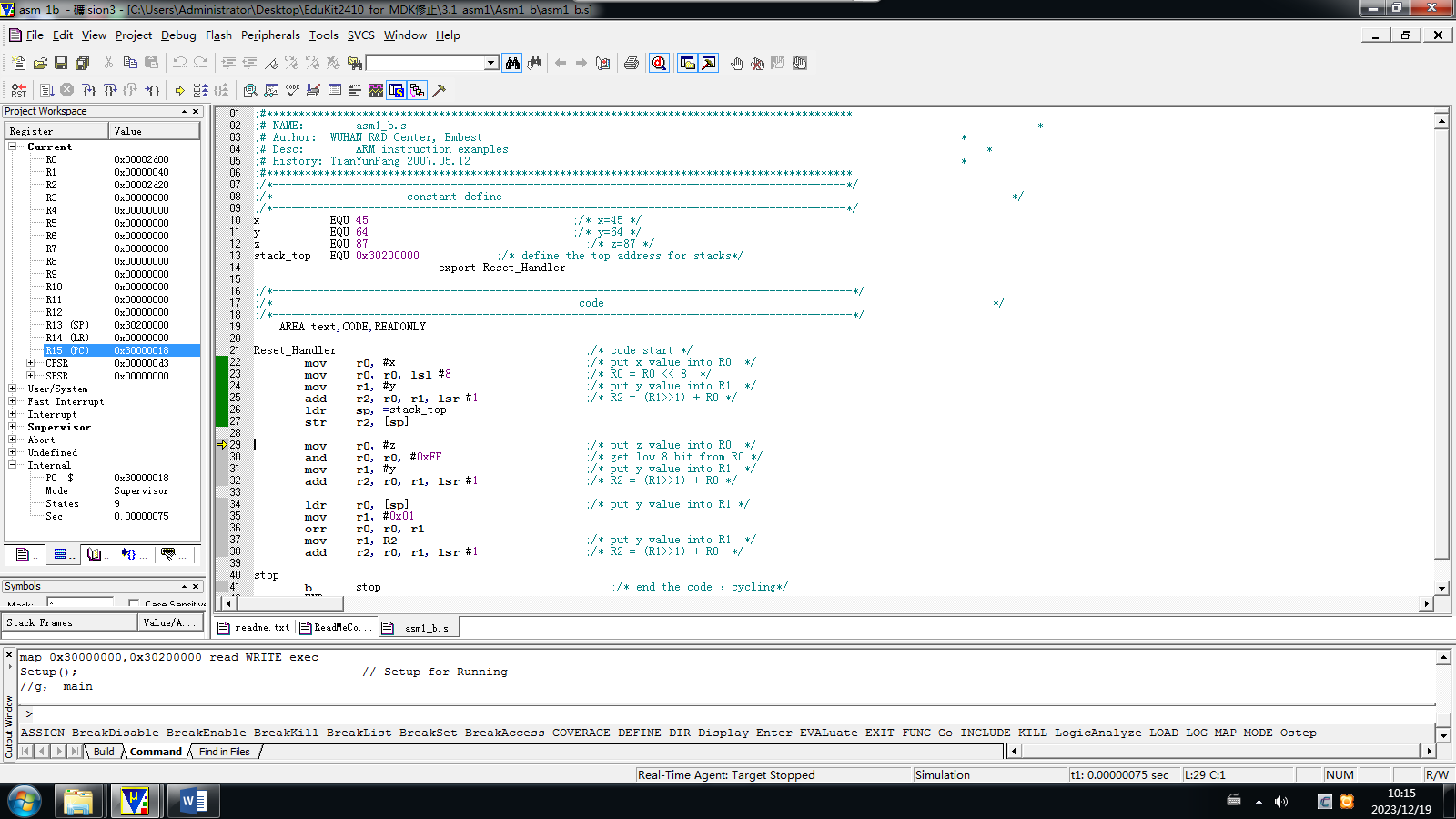
第十一步：ldr r0, [sp]把sp内存单元中的值加载到r0寄存器，sp的值为0x30200000

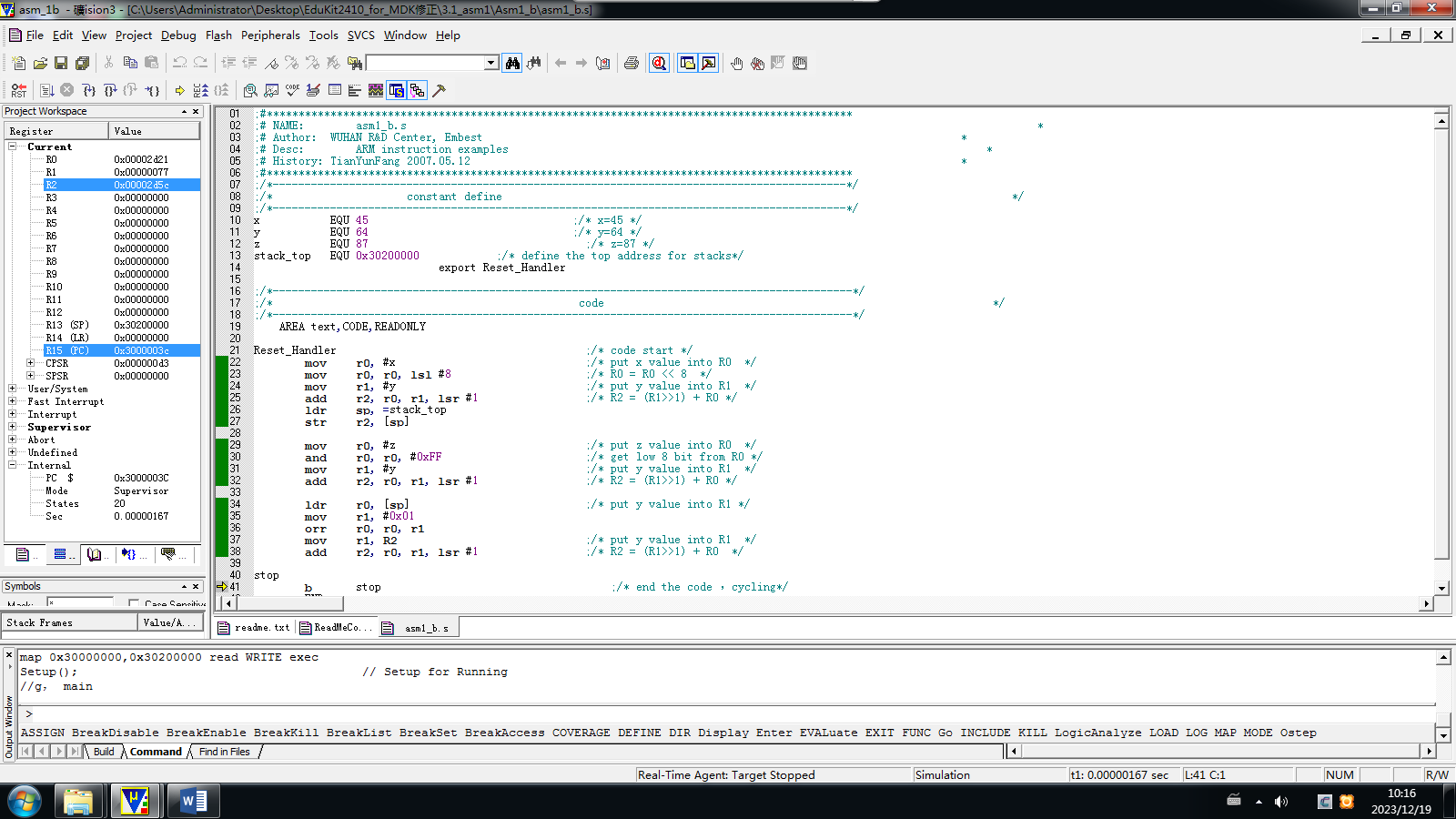
第十二步：mov r1, #0x01；0x01这个立即数传送到目标寄存器r1.

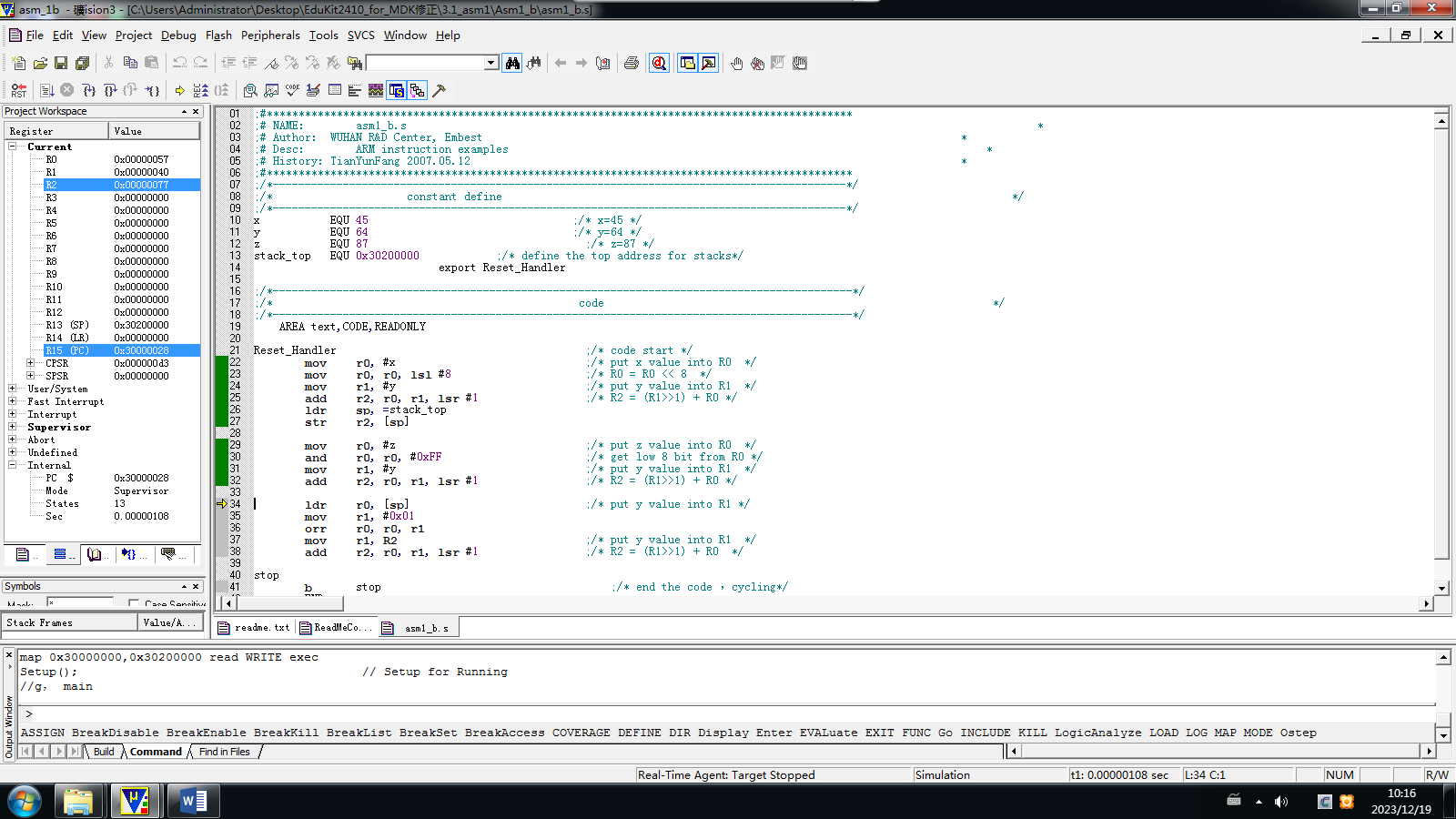
第十三步：orr r0, r0, r1；将r0寄存器的值与r1寄存器的值相或，最后在把值给到r0寄存器

第十四步：mov r1, R2 把y的值放入r1 中；把r2寄存器的值传送给r1寄存器

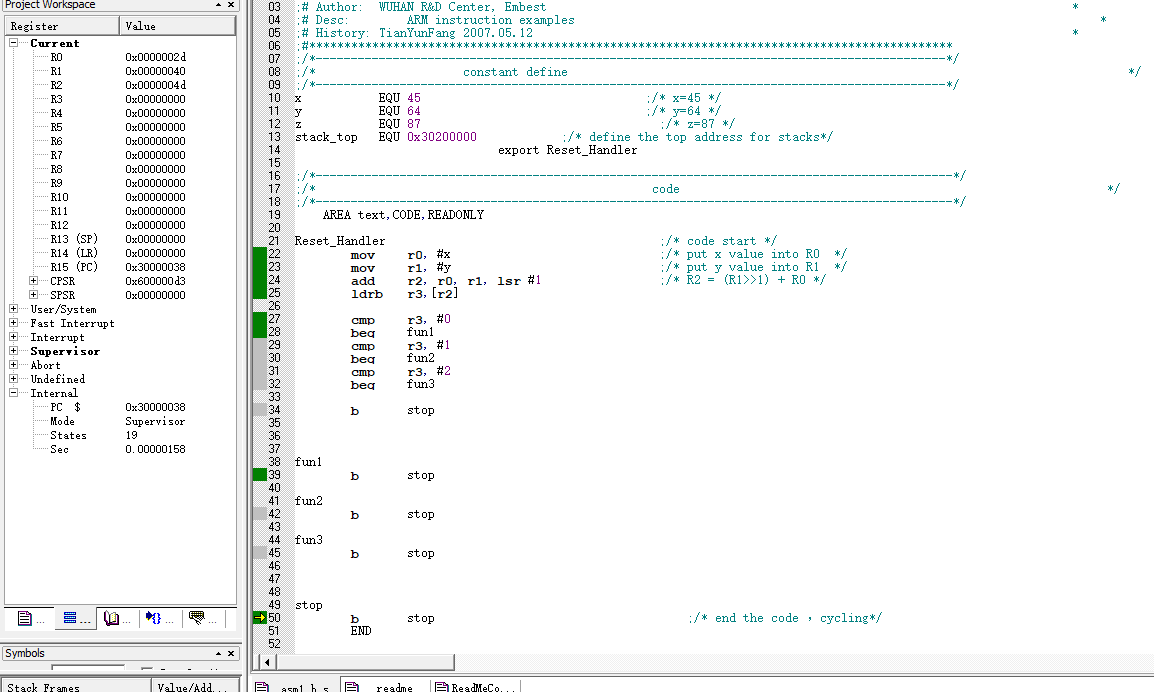
第十五步：add r2, r0, r1, lsr #1 R2 =(R1>>1) + R0；将r1寄存器的值右移1位与r0寄存器的值相加，赋值给r2寄存器。







2.完成分支程序设计，要求判断参数，根据不同参数，调用不同的子程序。



3.在程序中添加如下程序指令观察并写出结果

LDR R0,[R1]

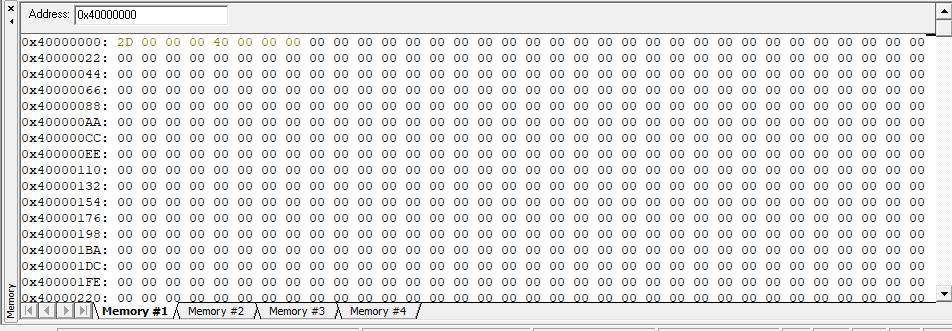
LDR R0,[R1,#4]

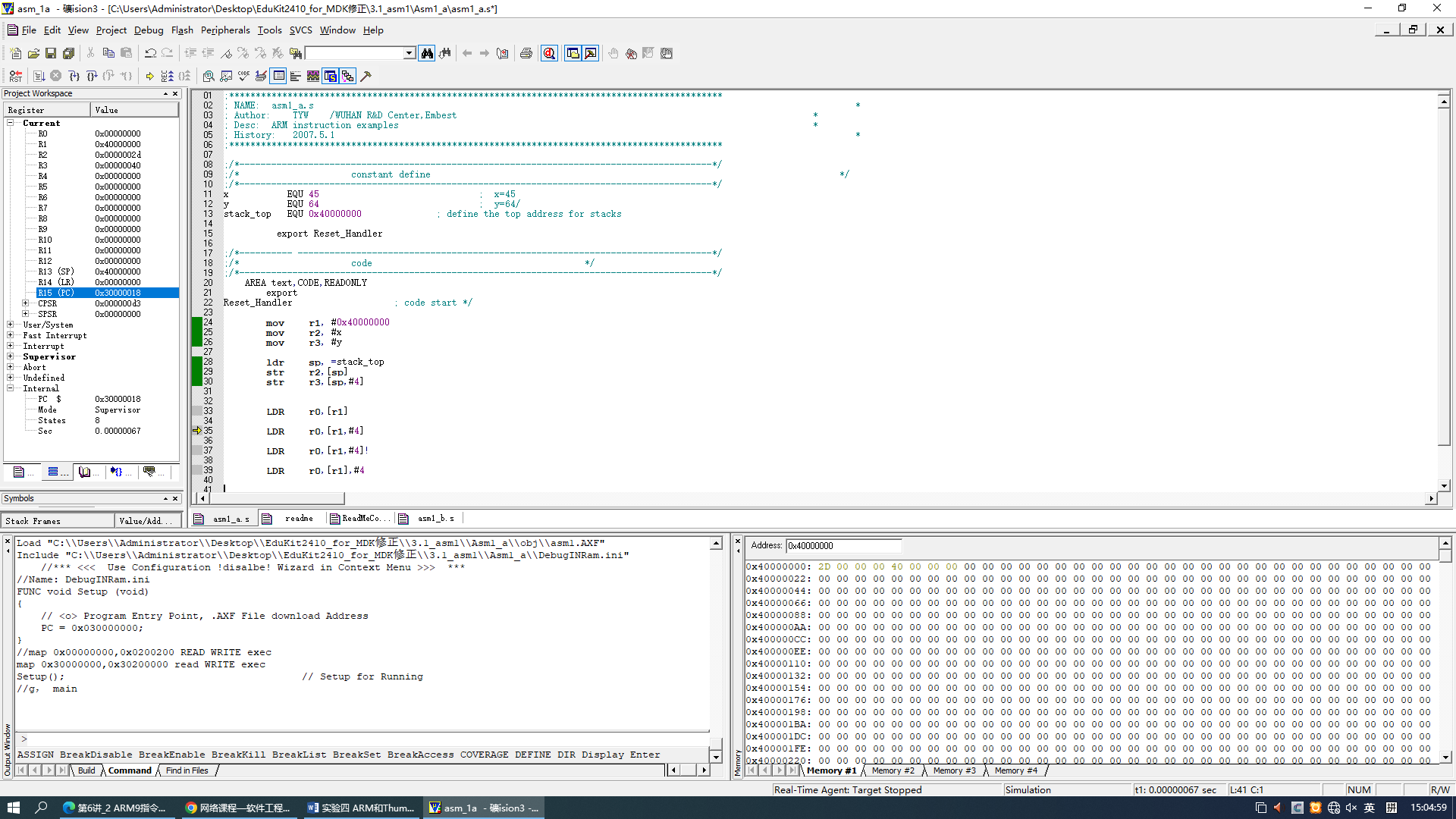
LDR R0,[R1,#4]!

LDR R0,[R1],#4

R1寄存器传送的值为0x40000000，r2寄存器传送的值为x，r3寄存器传送的值y。

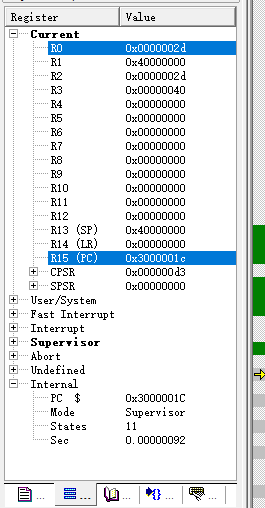
把栈顶sp置为0x40000000，把r2寄存器的值传送给sp这个地址，再把r3寄存器的值传送给sp+4这个地址。

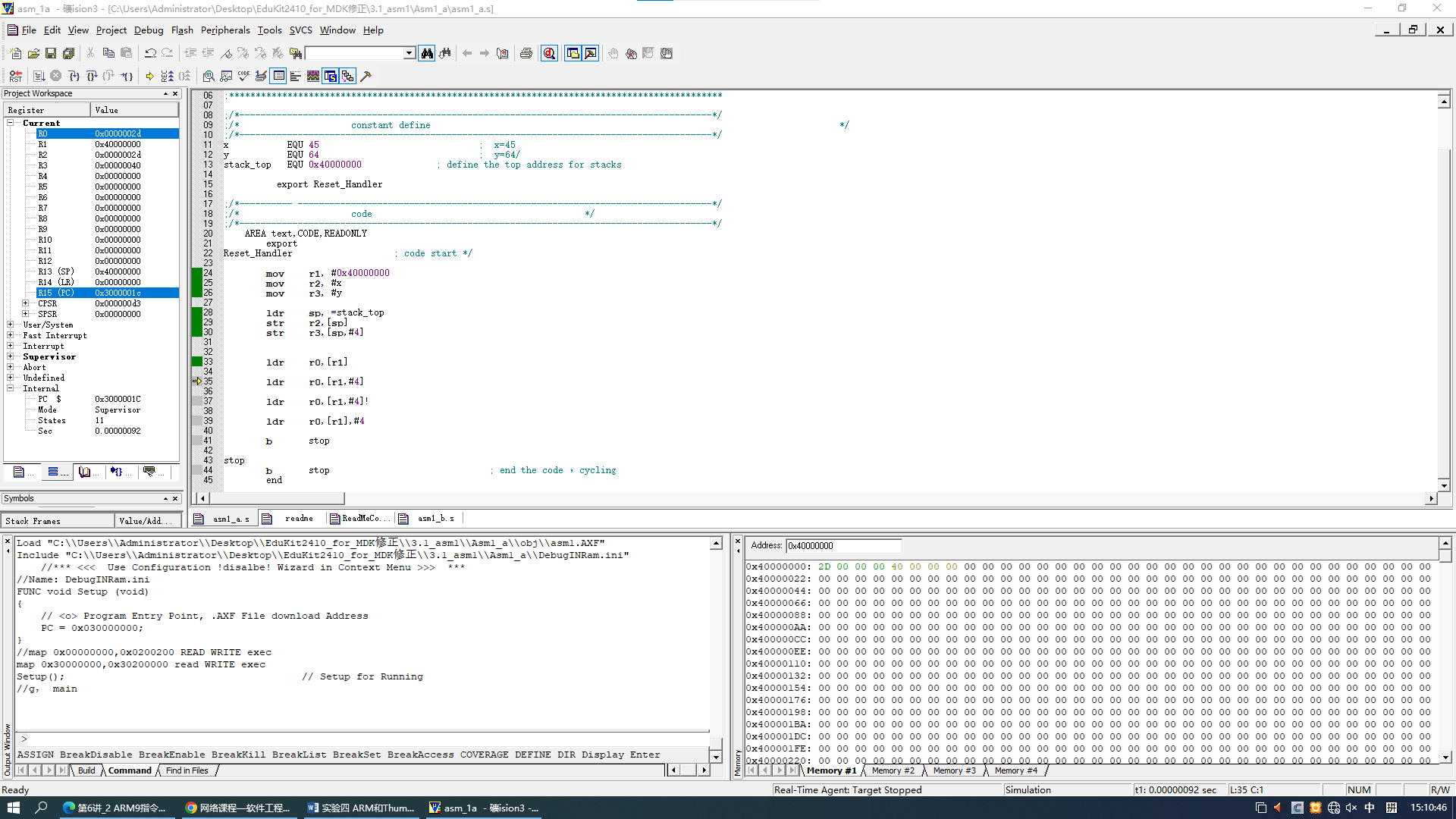




执行 ldr r0, [r1] 将存储器地址为r1的子数据读入寄存器r0

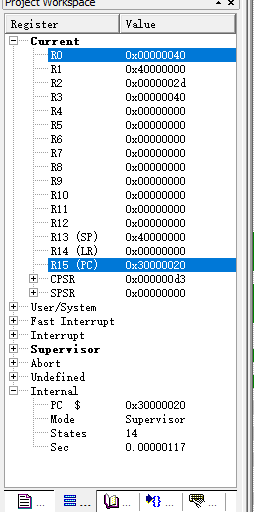
r1=0x40000000，这个地址的数据为0x2d，因此执行完这条指令后r0寄存器的值为0x0000002d。

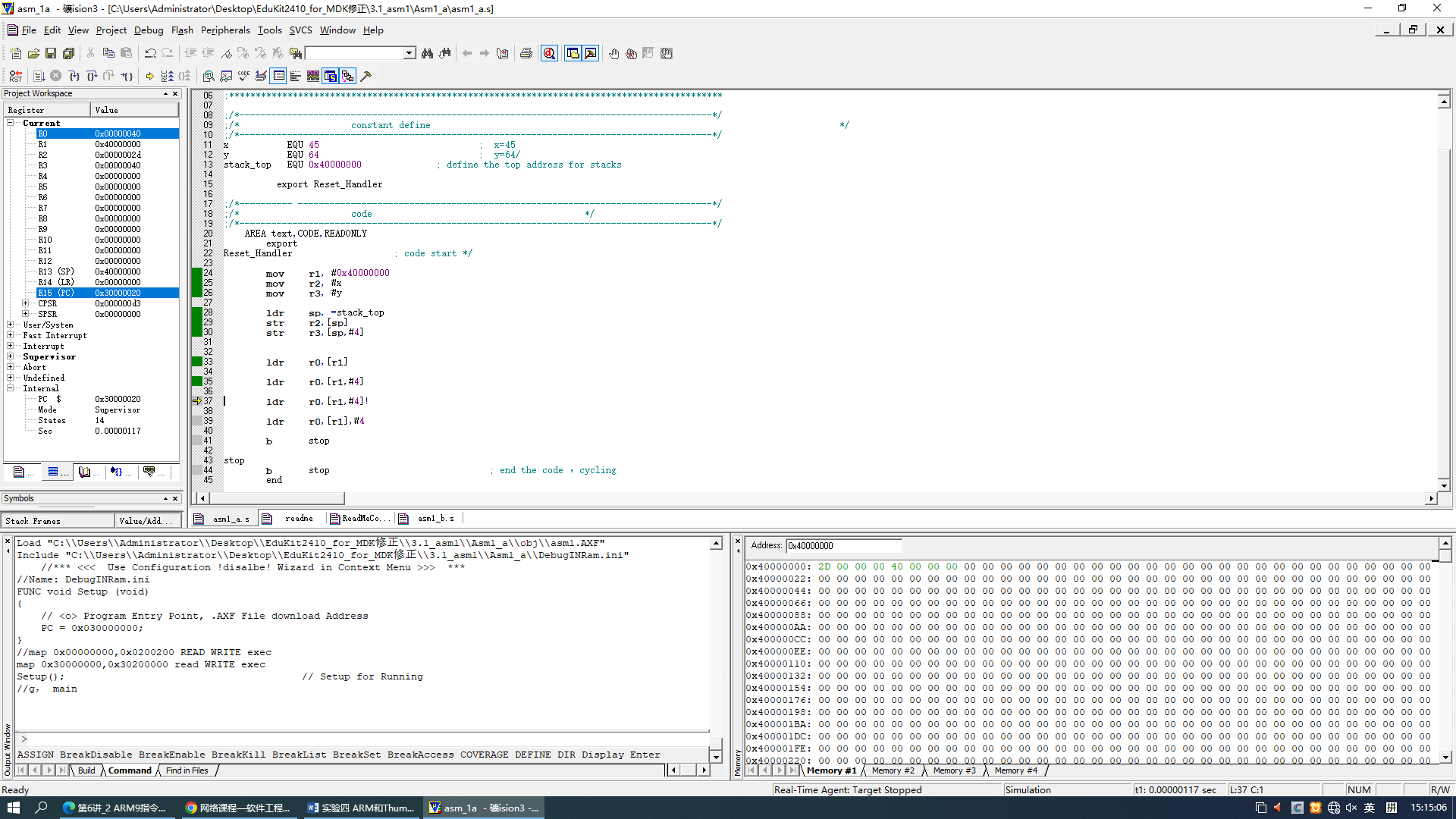




执行 ldr r0,[r1,#4] 将存储器地址为r1+4的子数据读入寄存器r0

r1+4这个地址的数据为为0x40，因此执行完这条指令后r0寄存器的值为0x00000040。

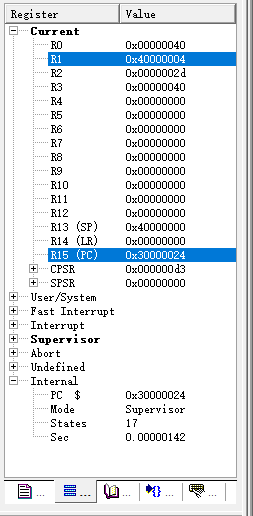


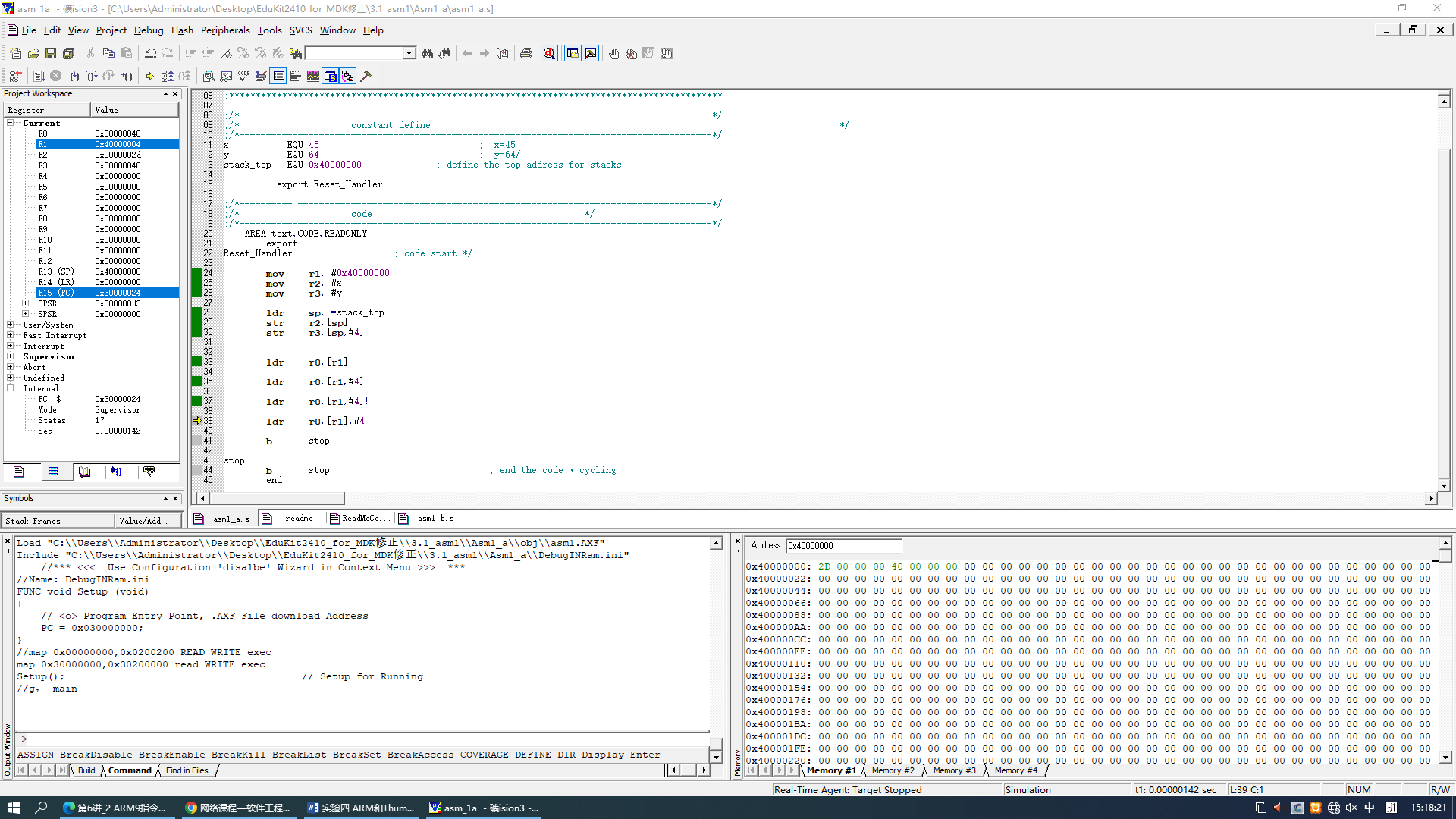


执行 ldr r0,[r1,#4]!将存储器地址为r1+4的子数据读入寄存器r0

r1+4这个地址的数据为为0x40，因此执行完这条指令后r0寄存器的值为0x00000040。并且要将新地址r1+4在写入r1.因此此时r1

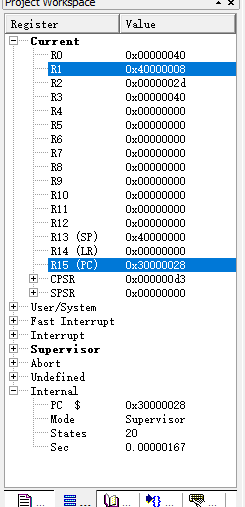
寄存器的值为0x40000004.

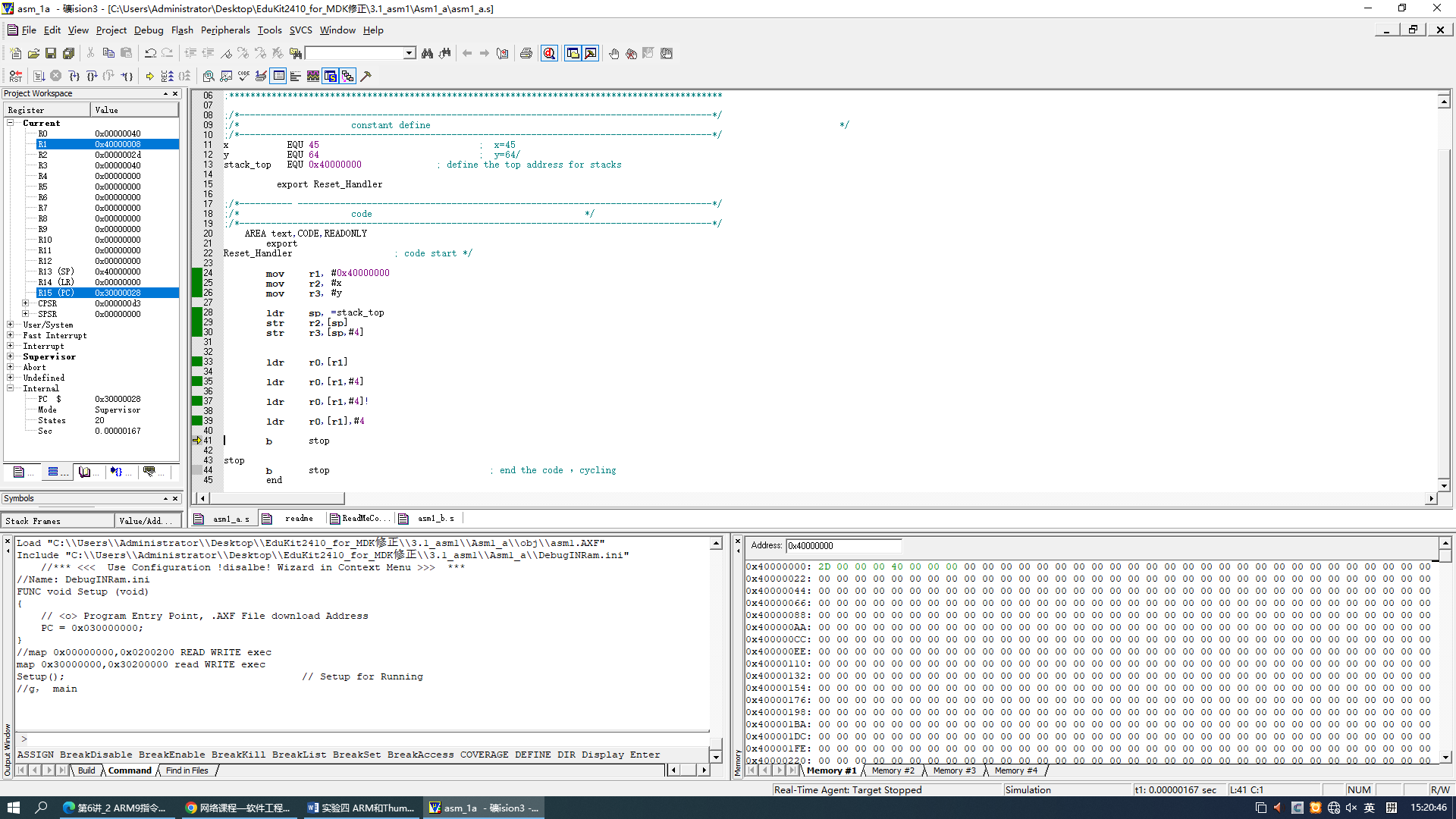




执行 ldr r0,[r1],#4将存储器地址为r1的子数据读入寄存器r0

r1这个地址的数据为为0x40，因此执行完这条指令后r0寄存器的值还是为0x00000040。并且要将新地址r1+4在写入r1.因为此时r1寄存器的值为0x40000004.因此执行完这条指令后r1寄存器的为0x40000008





4.观察并记录 ARM 指令下和 Thumb 指令状态下 stmfd，ldmfd，ldmia 和 stmia 指令执行的结果，指令的空间地址数值，数据存储的空间大小等等；

1. **实验总结**

**ARM和Thumb是一种指令集架构，用于处理器的指令执行。它们之间的主要区别在于指令长度和代码密度。**

**ARM指令集是32位指令集，每条指令占用4个字节。ARM指令集提供了更强大的功能和更高的执行效率，但指令长度较长，代码密度相对较低。**

**Thumb指令集是ARM指令集的一个压缩版本，每条指令占用2个字节。Thumb指令集可以提供更高的代码密度，节省存储空间，并且执行效率也相对较高。**

**总之对嵌入式汇编指令集掌握不是很好，运行ARM指令要仔细观察寄存器。子程序中调用跳转指令注意跳转条件。**