电子科技大学信息与软件工程学院

**实 验 报 告**

学 号 2018091602007

姓 名 王乐卿

（实验）课程名称《ARM处理器体系结构及

应用》课程实验

理论教师 兰刚

实验教师 兰刚

**电 子 科 技 大 学**

**实 验 报 告**

**学生姓名：王乐卿 学号：2018091602007 指导教师：兰刚**

**实验地点：家 实验时间：2020.5.19**

1. 实验名称：子程序及调用
2. 实验学时：4
3. 实验目的：
4. 了解ARM汇编程序函数和C语言程序函数相互调用时，遵循的ATPCS标准；
5. 了解和掌握ARM汇编程序函数和C语言程序函数相互调的基本方法；
6. 了解ARM应用程序框架。
7. 实验原理：

由于C语言便于理解，有大量的支持库，所以它是当前ARM程序设计所使用的主要编程语言。

对硬件系统的初始化、CPU状态设定、中断使能、主频设定 以及RAM控制参数初始化等C程序力所不能及的底层操作，还是要由汇编语言程序 来完成。

ARM工程 的各种源文件之间的关系，以及最后形成可执行文件的过程如下图1所示：

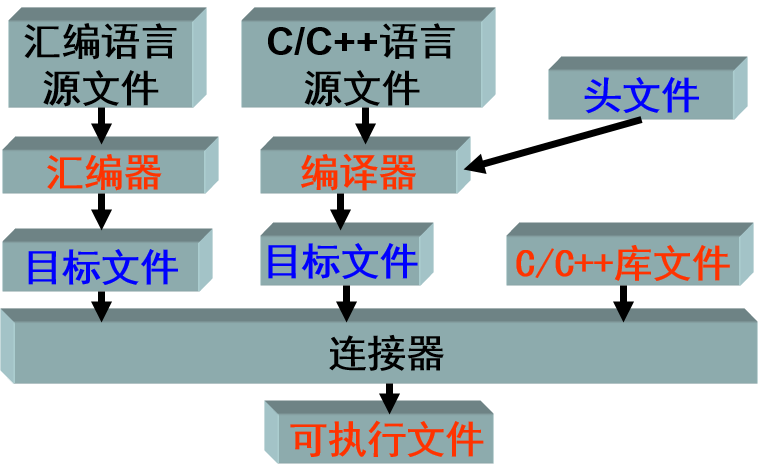


图1 汇编语言和C语言混合编译链接示意图

在应用系统的程序设计中，若所有的编程任务均用汇编语言 来完成，其工作量是可想而知的，这样做也不利于系统升级或应用软件移植。

通常汇编语言部分完成系统硬件的初始化；高级语言部分完成用户的应用。

执行时，首先执行初始化部分，然后再跳转到C/C++部分。整个程序结构显得清晰明了，容易理解。为方便工程开发，ARM公司的开发环境ARM ADS为用户提供了一个可以选用的应用程序框架。该框架把为用户程序做准备工作的程序分成了： 启动代码 和 应用程序初始化 两部分。

用于硬件初始化的汇编语言部分叫做 启动代码；用于应用程序初始化的C部分叫做初始化部分。整个程序如下图2所示：

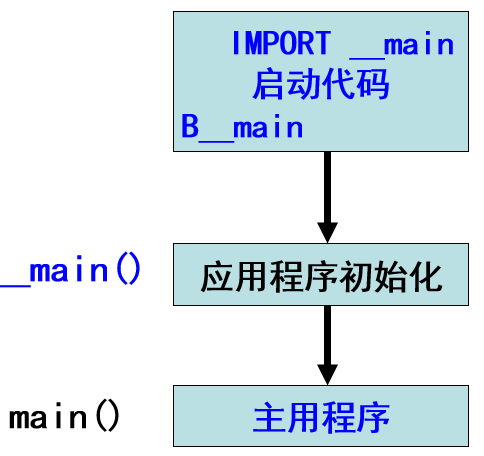


图2 ARM应用程序框架

在ARM工程中，C 程序调用汇编函数和汇编程序调用C函数是经常发生的事情。为此人们制定了 ARM-Thumb 过程调用标准 ATPCS（ARM-Thumb Procedure Call Standard）。

1. ATPCS规定，ARM的数据堆栈为FD型堆栈，即递减满堆栈。
2. ATPCS标准规定，对于参数个数不多于4的函数，编译器必须按参数在列表中的顺序，自左向右 为它们分配寄存器 R0~R3。其中函数返回时，R0 还被用来存放函数的返回值。
3. 如果函数的参数多于4个，那么多余的参数则按自右向左的顺序压入数据堆栈，即参数入栈顺序与参数顺序相反。
4. 根据 ATPCS 的 C 语言程序调用汇编函数，参数 由左向右 依次传递给寄存器 R0~R3 的规则。
5. 实验内容：

(1) ARM指令主程序调用ARM指令子程序；

(2) ARM指令主程序调用C语言子程序；

(3) C语言主程序调用ARM指令子程序；

1. 实验器材（设备、元器件）：

(1) PC机一台；

(2) Keil MDK-ARM uVision4开发工具。

1. 实验步骤：

(1) 打开Keil MDK-ARM uVision4开发工具；

(2) 新建一个工程文件；

(3) 在新建的工程文件中，添加新的源程序文件

(4) 编写代码

(5) 选择“Build target”菜单对编写好的工程文件进行编译链接。

(6) 点击““Start/Stop Debug Section””按键，对程序进行跟踪调试，在调试界面，单步执行，对CPU各寄存器的值的变化、以及相关内存的变化进行分析比较，判断程序的执行是否符合预期要求。

1. 实验结果与分析（含重要数据结果分析或核心代码流程分析）
2. **ARM指令主程序调用ARM指令子程序；**
3. **程序代码**

程序代码如代码1所示：

代码1 ARM指令主程序调用ARM指令子程序

|  |
| --- |
| AREA TT,CODE,READONLY  ENTRY  MOV R0,#20 ;赋值R0=20  MOV R1,#1  MOV R2,#2  MOV R3,#3  MOV R4,#4  MOV R5,#5  BL ADD1 ;使用BL调用子程序ADD1，同时把下一条指令的地址保存到LR中  B LOOP ;跳转到程序结尾处。  ADD1 SUBS R0,R0,R1 ;减法  SBCS R0,R0,R2 ;带进位减法  SBCS R0,R0,R3  SBCS R0,R0,R4  SBCS R0,R0,R5  BX LR ;使用BX从子程序返回，bit0等于0，程序还是处于ARM状态  LOOP MOV R0,R0  END |

1. **运行过程及结果界面截图**

图1显示了程序运行的结果：R0=0x00000005=5，即成功运行了20-1-2-3-4-5的操作，程序运行结果符合期望的结果。

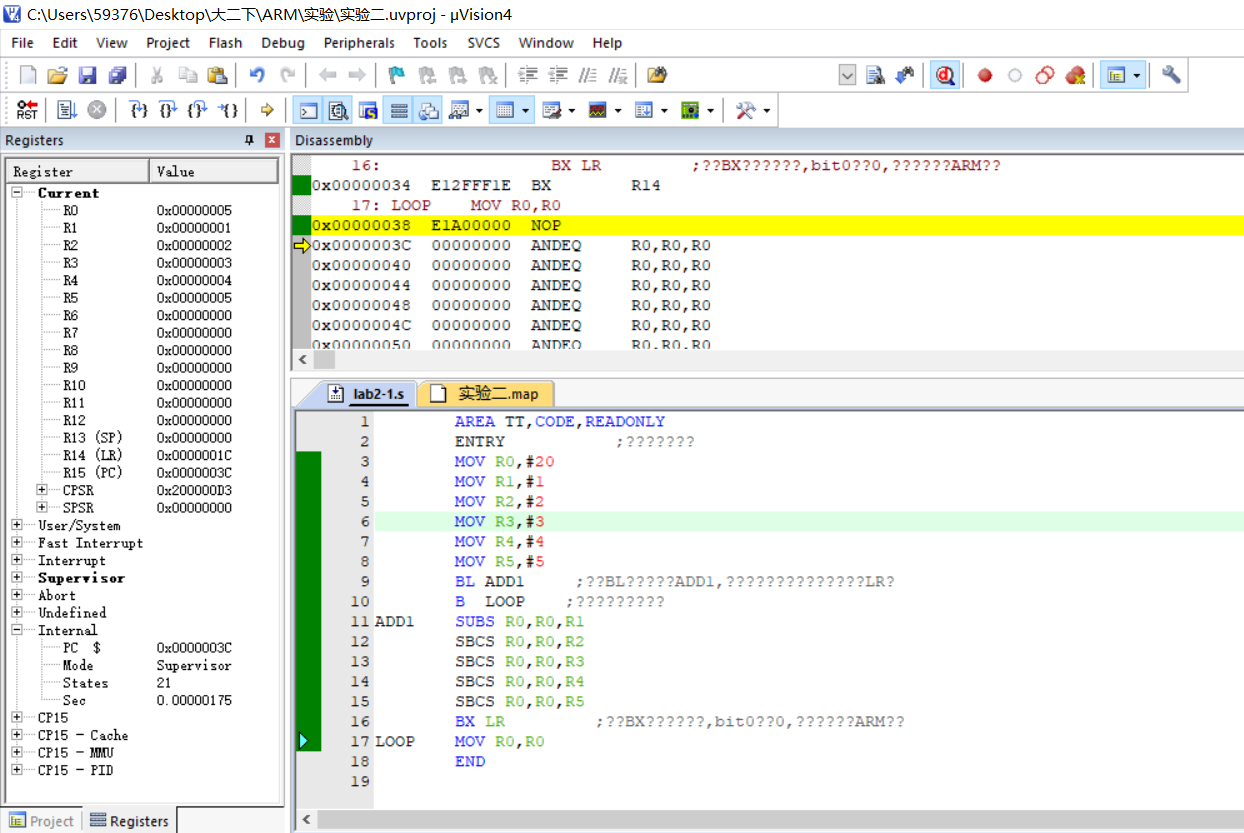


图1 ARM指令主程序调用ARM指令子程序运行结果截图

1. **实验结论**

通过BL可以正确的调用子程序，并且可以通过BX RL指令从子程序宏正确返回。

1. **ARM指令主程序调用C语言子程序；**
2. **程序代码**

ARM指令主程序如代码2所示

代码2 RM指令主程序调用C语言子程序的RM指令主程序

|  |
| --- |
| PRESERVE8  IMPORT TC1  AREA HUIC,CODE ,READONLY  ENTRY  LDR SP,=0X40000400  MOV R0,#20  MOV R1,#5  MOV R2,#4  MOV R3,#3  STR LR, [SP, #-4]!  MOV R4,#2  STR R4, [SP, #-4]!  MOV R4,#1  STR R4, [SP, #-4]!  BL TC1  ADD SP,SP,#8  LDR PC, [SP], #4    END |

C语言子程序代码如代码3所示

代码3 RM指令主程序调用C语言子程序的C语言子程序

|  |
| --- |
| int TC1(int i1,int i2,int i3,int i4,int i5,int i6)  { //c程序，i1减去i2减去i3减去i4加上i5加上i6  int i0;  i0=i1-i2-i3-i4+i5+i6;  return i0;  } |

1. **运行过程及结果界面截图**

图2显示程序通过执行STR LR, [SP, #-4]!实现把LR入栈的操作（注：此时LR的值为0）。图中的Memory窗口灰色的4个0表示刚入栈的LR的值。

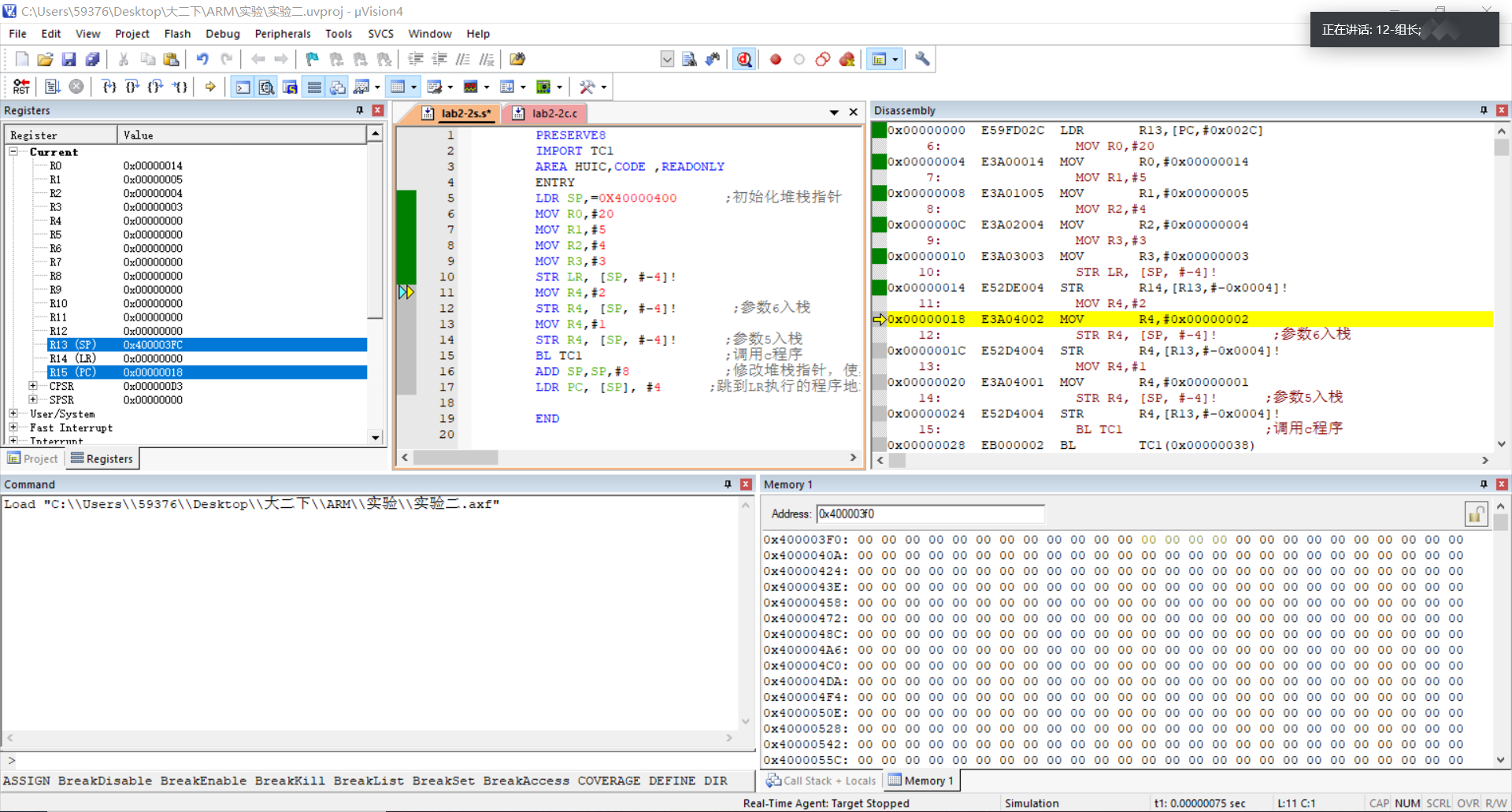


图2 程序执行完STR LR, [SP, #-4]!的界面截图

图3是程序即将调用C子程序的截图。

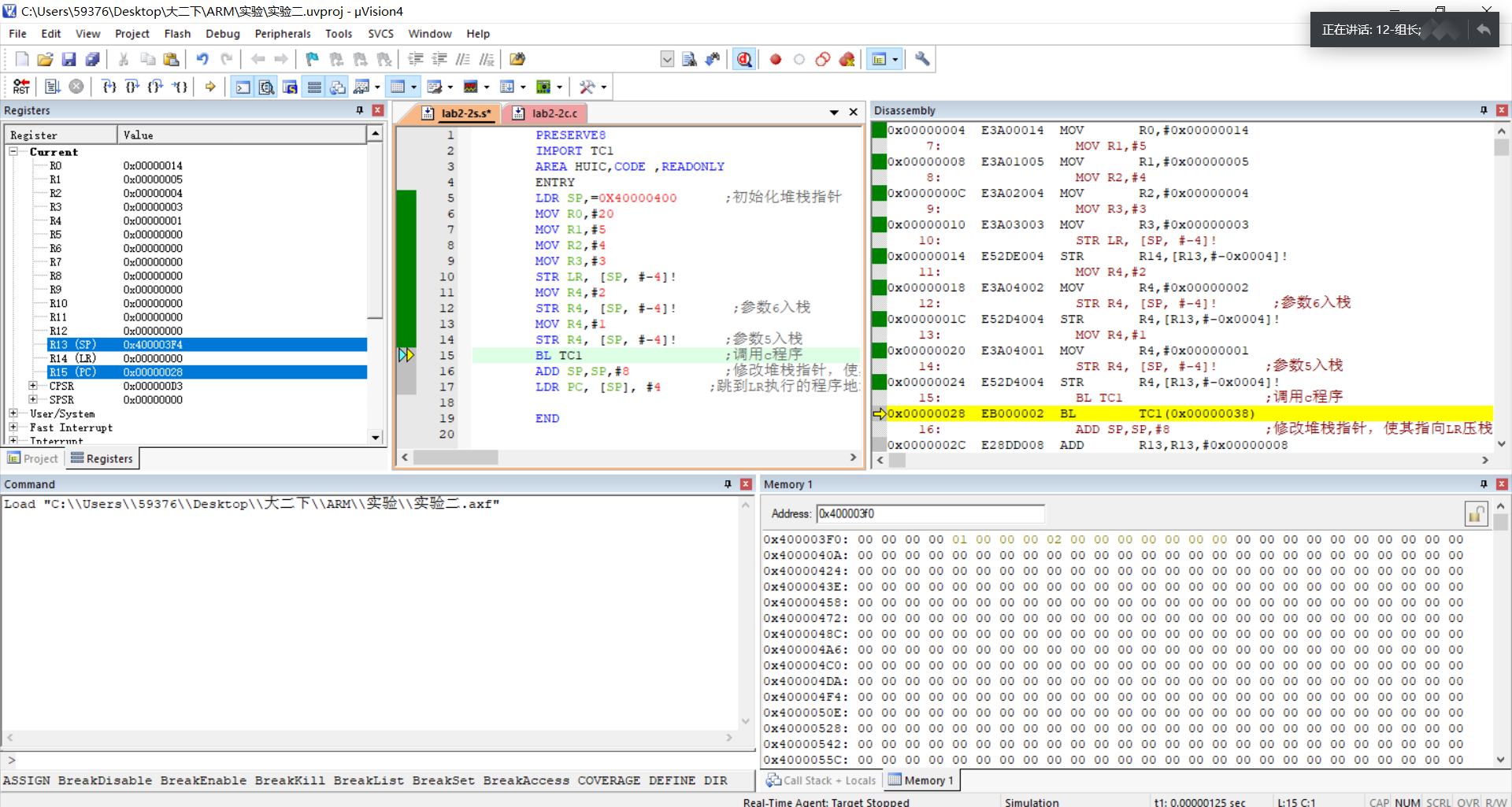


图3 程序即将调用C子程序的截图

图4是程序跟踪进入C语言子程序的截图。

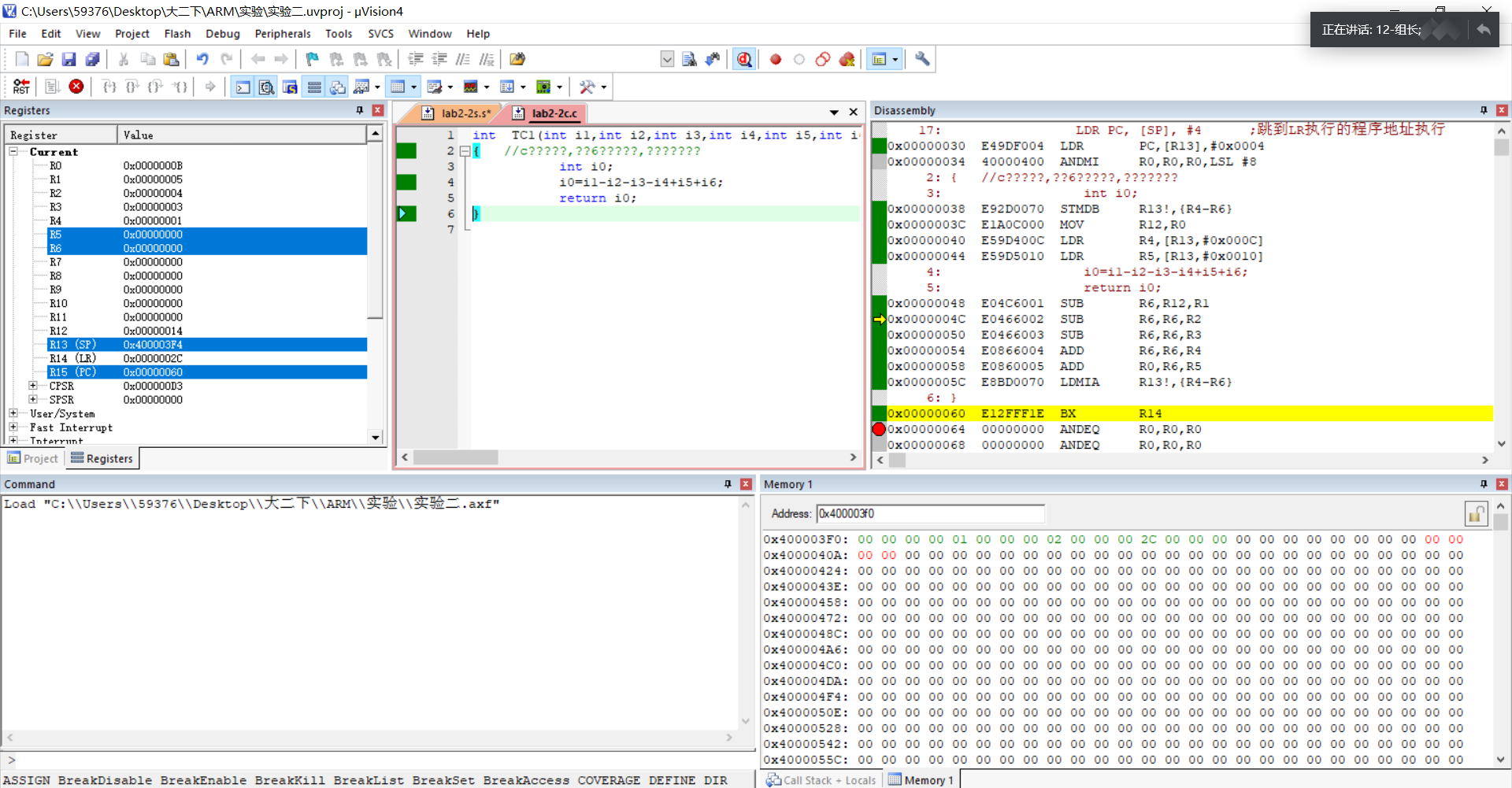


图4 程序跟踪进入C语言子程序的截图

1. **实验结论**

符合ATPCS的ARM汇编程序，可以正确的调用同样符合ATPCS标准的C语言子程序。

1. **C语言主程序调用ARM指令子程序；**
2. **程序代码**

C语言代码如代码4所示。

代码4 C语言主程序调用ARM指令子程序的C语言程序部分

|  |
| --- |
| #include <stdio.h>  extern int Call\_Arm(int, int, int, int, int, int, int);  int main()  {  int a,b,c1, c2,c3,c4,c5,c6,c7;  a=15;  a = Call\_Arm(22,1,2,3,4,5,6);  b=14;  b = Call\_Arm(22,1,2,3,4,5,6);  c1 = 22;  c2 = 1;  c3 = 2;  c4 = 3;  c5 = 4;  c6 = 5;  c7 = 6;  b = Call\_Arm(c1,c2,c3,c4,c5,c6,c7);  c1 =a + b;  return 0;  } |

汇编程序代码如代码5所示。

代码5

|  |
| --- |
| EXPORT Call\_Arm    Call\_Arm SUB R0,R0,R1  SUB R0,R0,R2  SUB R0,R0,R3  STMFD SP!,{R4-R6}  LDR R4,[SP,#12]  SUB R0,R0,R4  LDR R5,[SP,#16]  SUB R0,R0,R5  LDR R6,[SP,#20]  SUB R0,R0,R6  LDMFD SP!,{R4-R6}  MOV PC,LR |

1. **运行过程及结果界面截图**

图5显示，程序即将开始调用汇编子程序，从黄色箭头开始到BL指令部分，是调用前，准备参数的过程。目前SP指针为：0x40000428,此时a= 0x0000000F=15

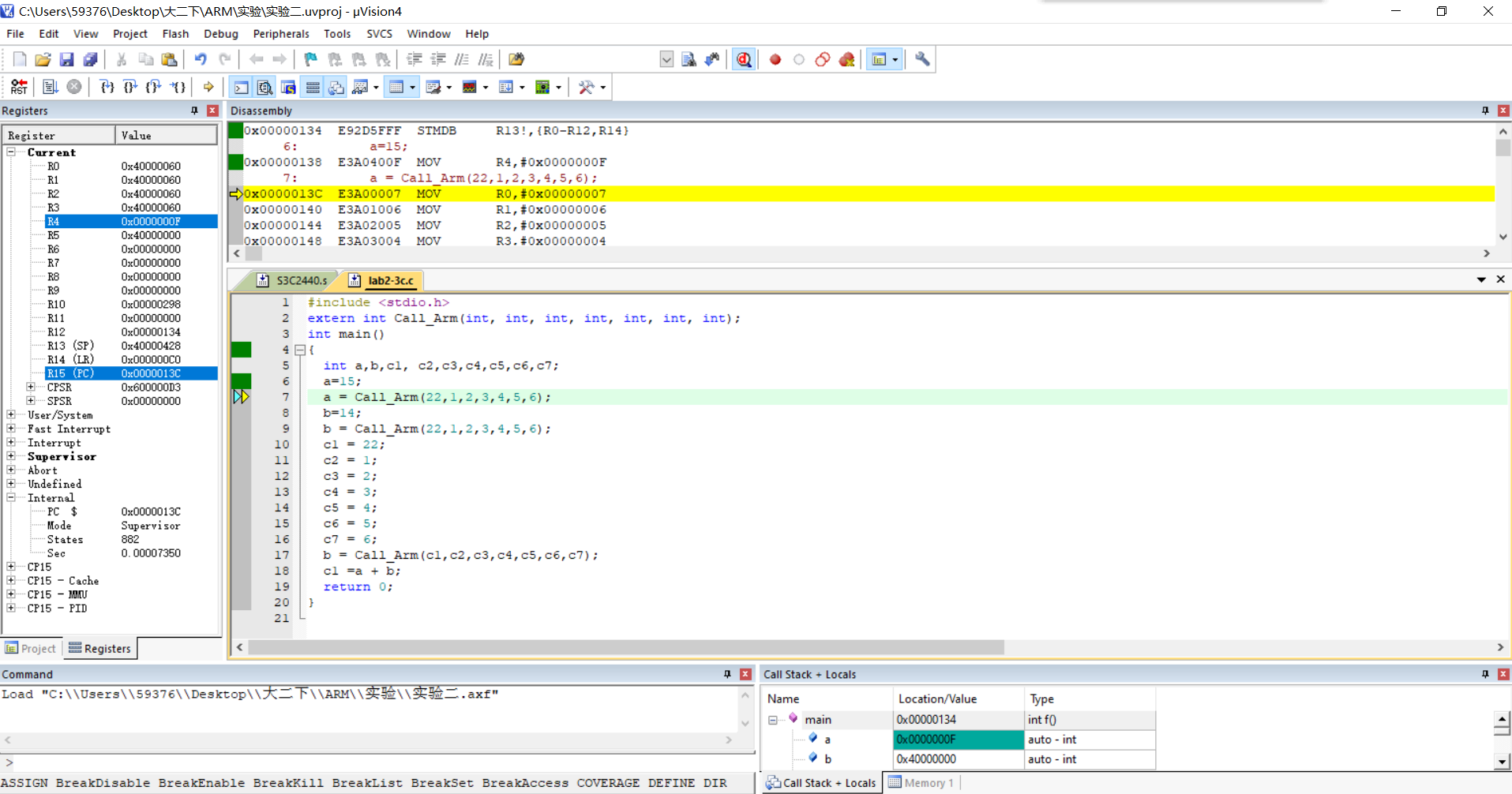


图5 程序即将调用汇编子程序Call\_Arm

下图6显示执行完子程序调用后，从子程序中返回，返回结果存放在R0中，结果为0x00000001=1，符合预期结果，子程序调用正确。

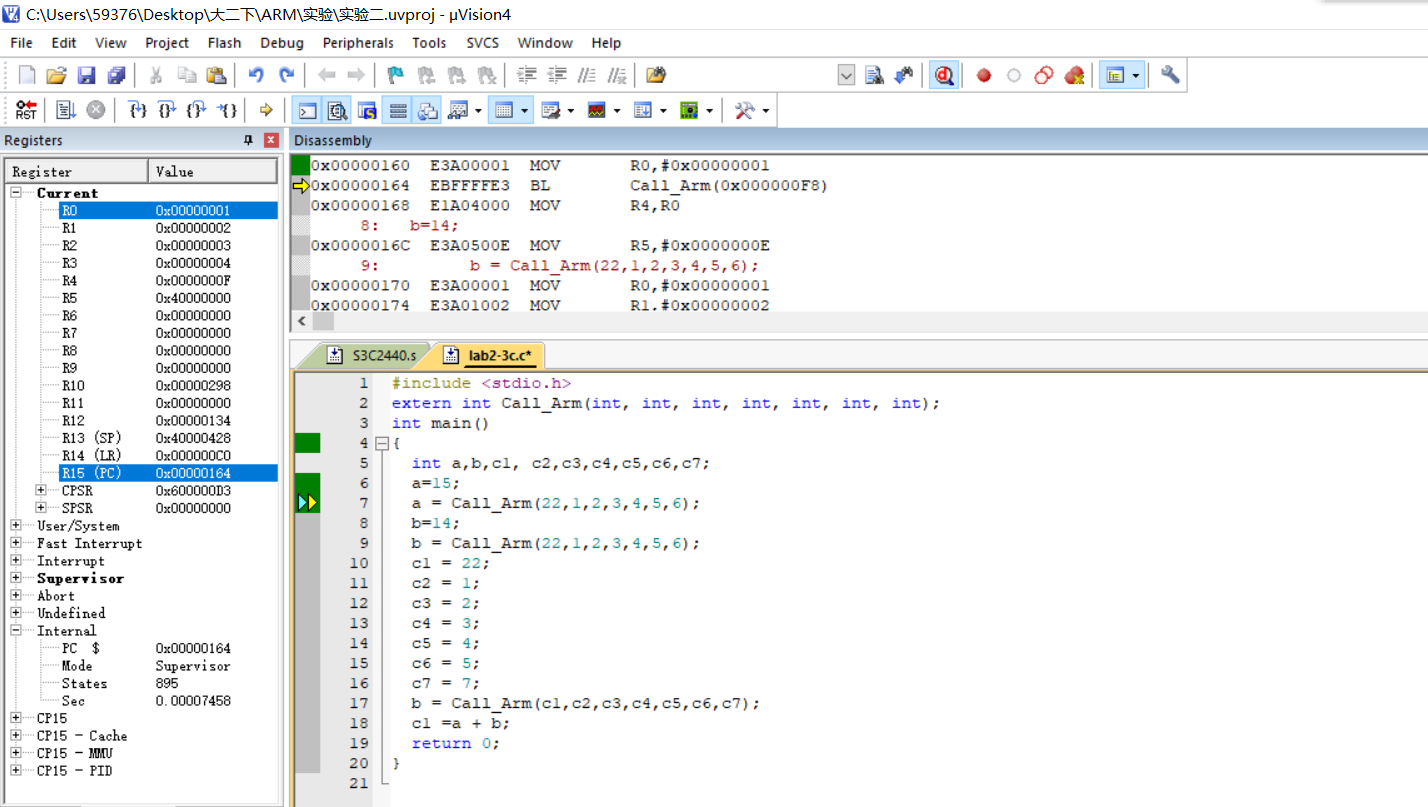


图6 程序从汇编子程序Call\_Arm中返回

下图7显示了程序跟踪进入汇编子程序Call\_Arm中的截图。

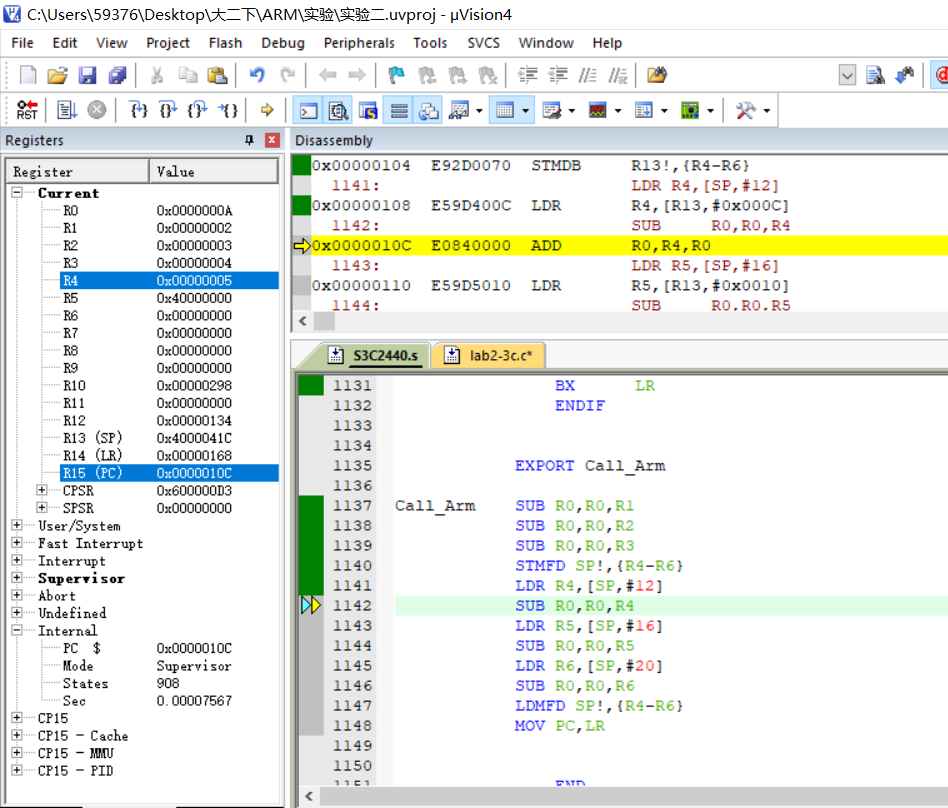


图7 程序跟踪进入汇编子程序Call\_Arm中

**3、实验结论**

符合ATPCS的C语言程序，可以正确的调用同样符合ATPCS标准的汇编语言子程序。

1. 总结及心得体会：

通过本次实验，我更加深入的了解了ARM汇编程序函数和C语言程序函数相互调用时，遵循的ATPCS标准，掌握了ARM汇编程序函数和C语言程序函数相互调的基本方法和ARM应用程序框架。

在实验过程中，遇到了许多的问题，比如linker的设置中没有去除Use Menery Layout from Target Dialog，而导致了一系列报错，以及在ARM主程序调用C子程序时存在首先运行C程序的问题，需要加入-first，还有在C主程序调用ARM子程序时，存在类似未定义符号问题，需要在asm中加入\_\_EVAL以解决问题。好在虽然困难很多，但是在合理运用搜索引擎和咨询老师同学之后都得到了很好的解决。

通过本次实验，我也更加深入的掌握了keil软件的使用和使用keil软件debug的能力也有了很大程度的提升，可以说是收获满满。

1. 对本实验过程及方法、手段的改进建议：

实验过程中会遇到很多奇奇怪怪的问题，虽然有些错误在指导书中已经给出，但是还是有很多没有给出，希望能更加完善实验指导书，提供更加完善的指导。

报告评分：

指导教师签字：