# 第六章 ARM汇编与C混合程序设计

实验目的：利用ARM指令集编写汇编程序并实现与C的混合程序设计，掌握ARM汇编程序设计的基本方法并了解不同语言撰写的函数之间相互调用（mix calls）的基本规则（ATPCS）。

## 6.1 简单汇编新工程的建立及编译、链接、调试过程

MDK可以编写基于C和ARM 汇编的工程，下面我们先建立一个简单的汇编工程并进行编译、链接和调试（MDK为5.16版本）。

1. 打开Keil软件，进入软件，点击菜单栏上的“Project”→“New uVision Project…”。

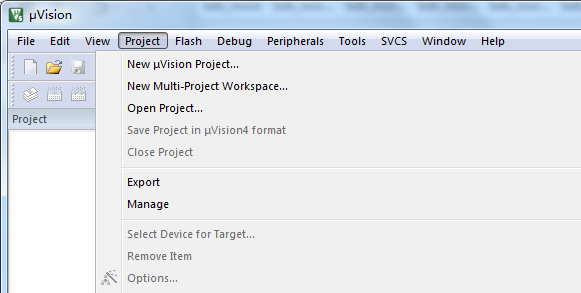


图6.1.1 Keil-MDK建立新工程

1. 设定保存工程路径和工程名（汇编工程比较简单，不需要设置库函数目录，可以直接将工程文件存在根目录下，如图6.1.2中即工程名为test2存在test目录中），扩展名为\*.uvproj或者\*.uvprojx。

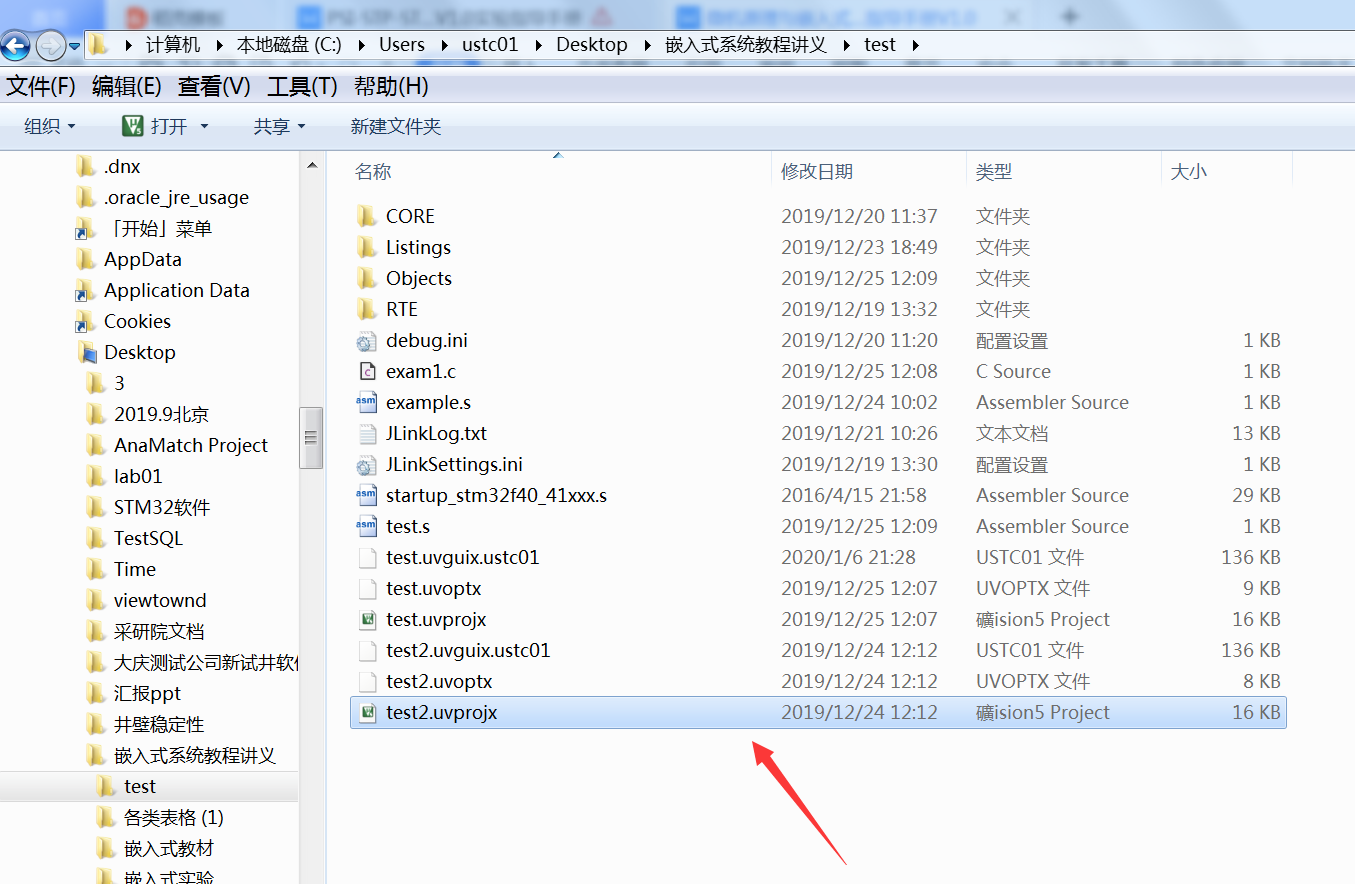


图6.1.2 新建工程保存文件夹和工程名设置

1. 选择所需要的MCU型号，本实验箱的核心处理器芯片是STM32F407ZGT6，在此我们选择STM公司的STM32F407ZG芯片。

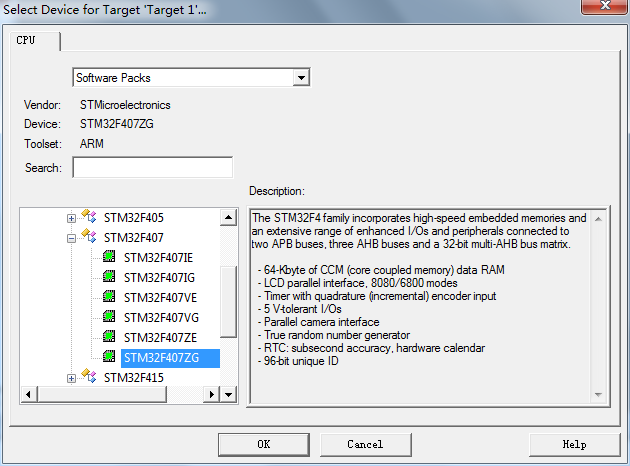


图6.1.3 选择处理器芯片型号

1. 点击 OK选定后，MDK会弹出Manage Run-Time Environment（管理运行环境）对话框，这是 MDK5 新增的一个功能，在这个界面，我们可以添加自己需要的组件，从而方便构建开发环境，其中最重要的一个功能是可以选择Device的Startup功能，勾选后系统会自动帮你把启动文件（后缀名为\*.s的汇编文件）拷贝到工程目录下，启动文件会做很多系统初始化的设置，包括指令指针初始化、堆栈初始化、各种变量的存放等待功能。

**注意：启动文件一般在最后会找C中的main函数做为入口点，所以汇编程序中一般不会选择系统启动，这也意味着这些初始化功能我们必须自己选择最简单模式实现。**

这里我们直接点击Cancel，这样一个新的工程就建立完成了。

1. 利用Manage Project Items修改Target名为“ASM”和Group名为“1”如图6.1.4所示。

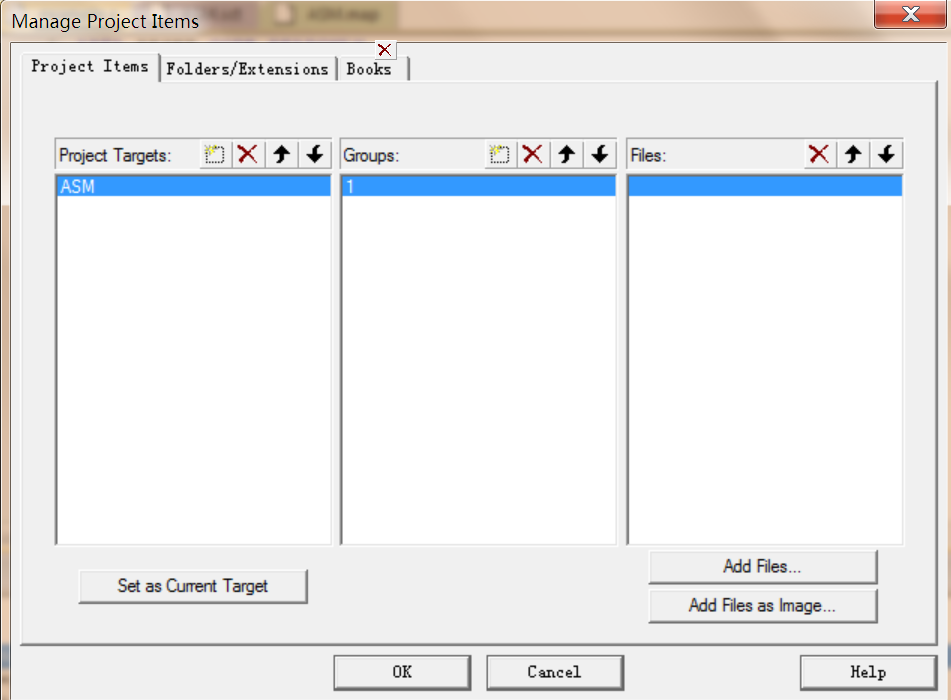


图6.1.4 修改Target和Group名

1. 下一步需要为工程添加文件（因为工程文件\*.uvproj或者\*.uvprojx只包含工程结构信息，不包括源码，所以需要添加源码文件，汇编工程要添加汇编文件，C工程要添加main.c文件等），因为这个项目是一个简单的汇编工程，所以只需要添加一个汇编文件即可，File->New

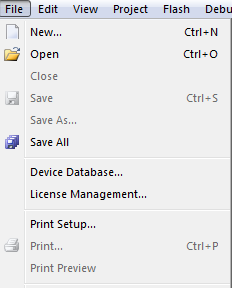


图6.1.5 工程添加新文件

添加完毕后将文件另存为任意文件名并添加扩展名，假设存为example.s，并如图6.1.6所示加入工程，图左边标注可以看到example.s已被加入工程结构中。

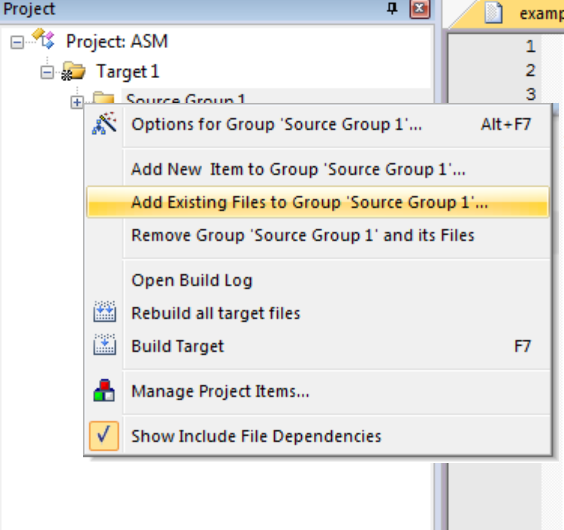
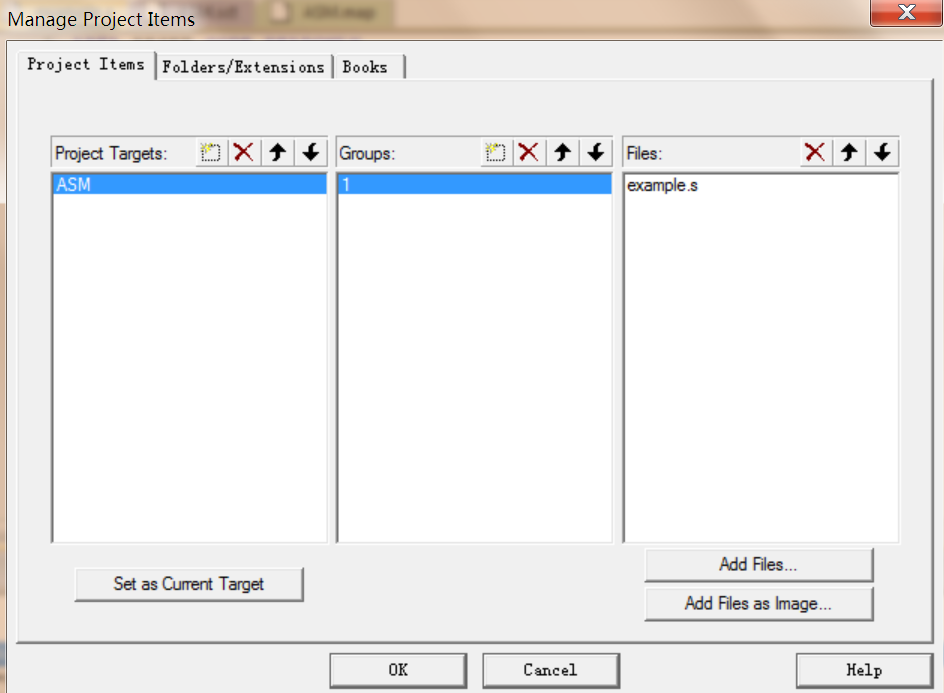


图6.1.6 添加新文件到工程中



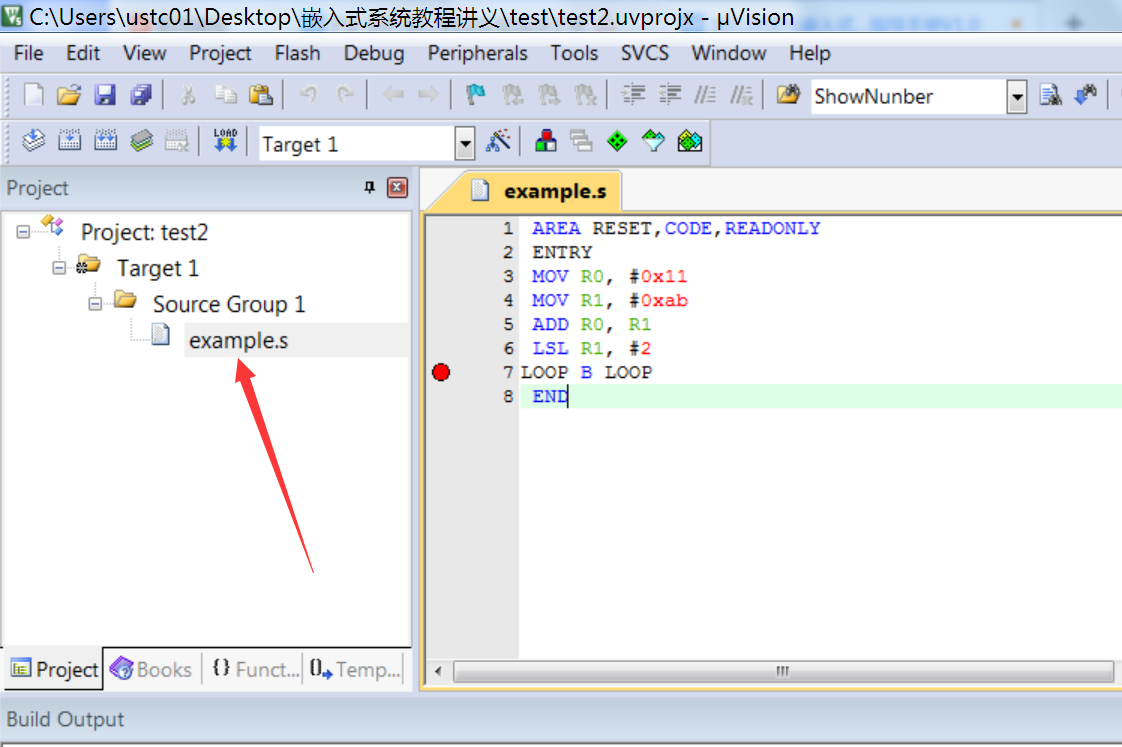


图6.1.7 添加的汇编源文件并添加到工程结构中

同时这里需要对工程进行一些设置，Options for Target重要的几个设置如下图。

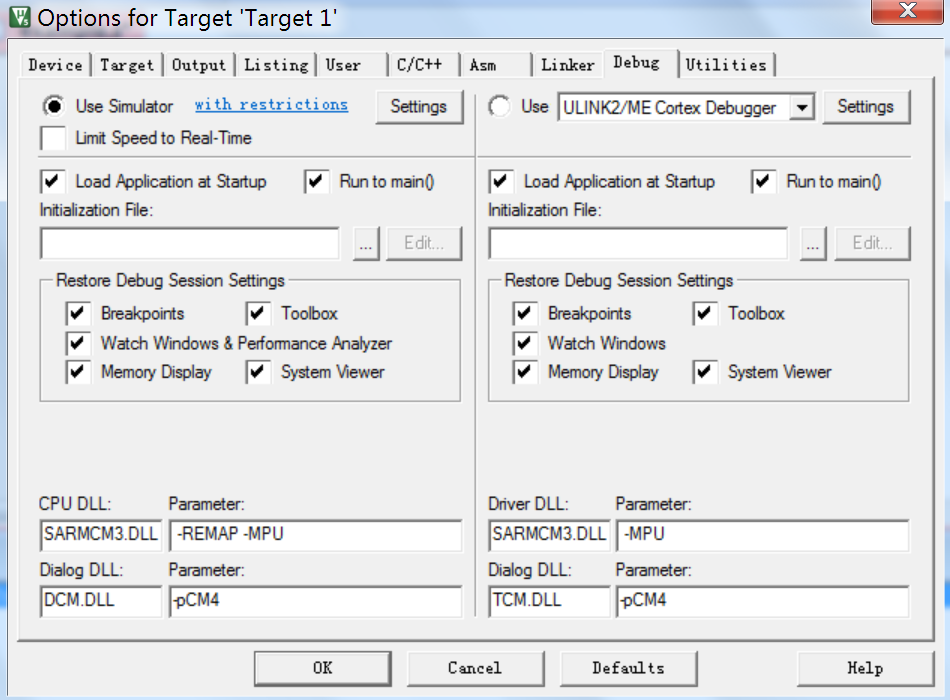
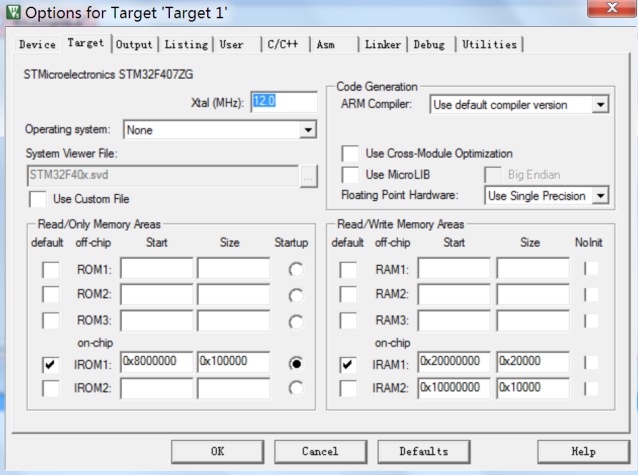


图6.1.8 汇编工程设置

这里设置的是IROM和IRAM的起始地址和大小，也就是系统的代码区和数据区，这和我们开始选择的芯片型号相关，本工程IROM从0x08000000开始，IRAM从0x20000000开始，一般选定了芯片起始地址和大小就是默认值。另一个设置是Debug选型里的仿真类型，我们用软件仿真Use Simulator，另一个选项是仿真器仿真。

1. 至此设置完毕，可以在example.s中开始代码编写。

AREA RESET,CODE,READONLY

ENTRY

MOV R0, #0x11

MOV R1, #0xab

ADD R0, R1

LSL R1, #2

LOOP B LOOP

END

**注意：根据ARM汇编指令语法，程序中伪指令AREA ENTRY END以及各条指令不能左边顶格书写，而标号LOOP必须顶格书写，否则编译会报错。**

这个程序代码实现简单的寄存器赋值功能，并对寄存器内容进行左移操作，最后进入死循环状态。代码编辑完毕后点击进行编译，通过后即可进行调试。

选择Debug->Start/Stop Debug Session，如图6.1.9所示。

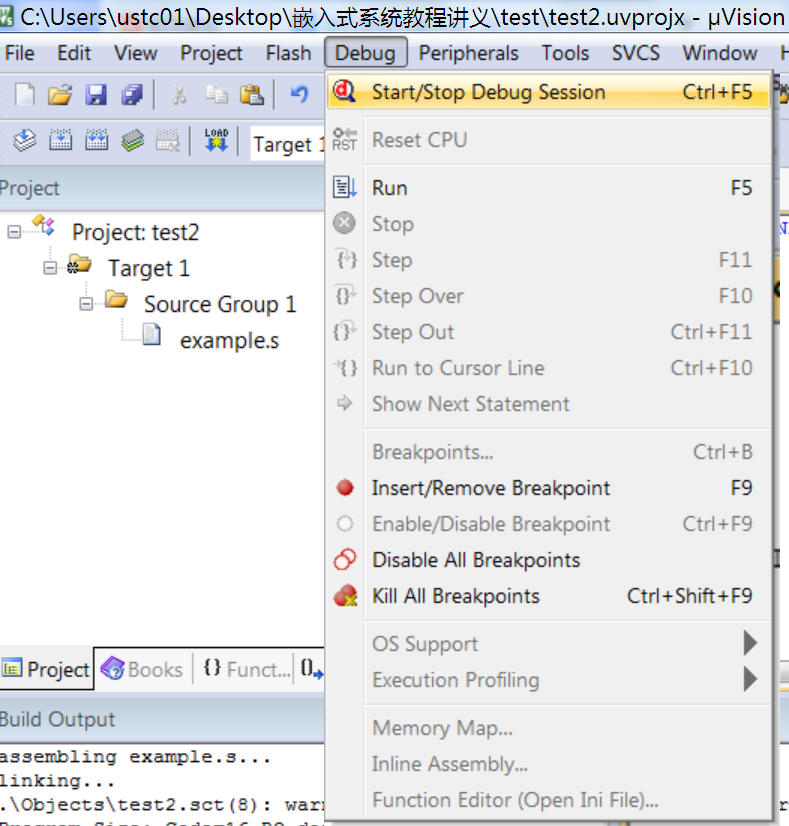


图6.1.9 程序进入调试模式菜单选项

进入调试状态后如图6.1.10所示，包括寄存器状态、反汇编、代码区、内存观察等窗口。

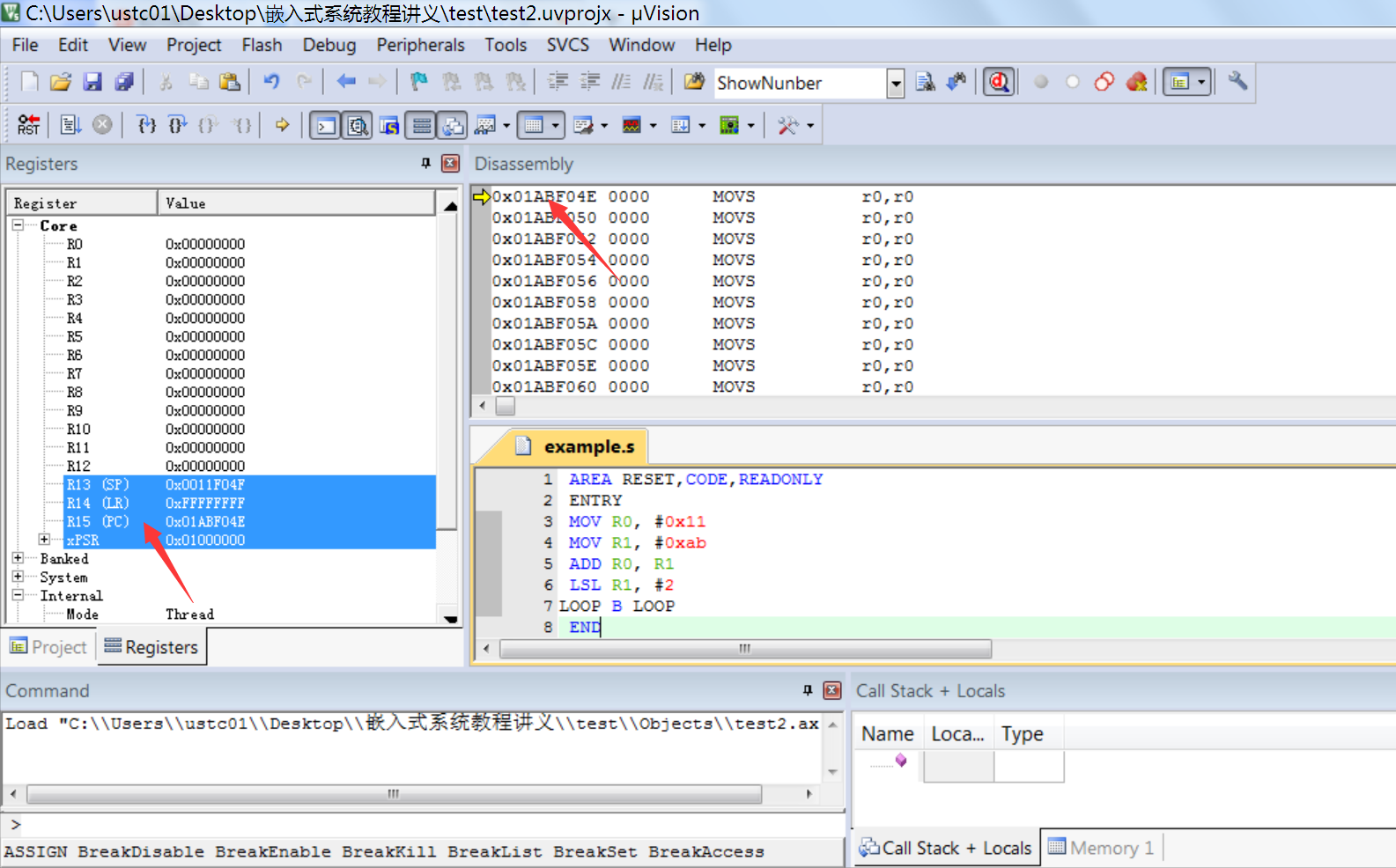


图6.1.10 程序进入调试模式

**注意：前面说过，汇编程序没有用系统自带的启动程序，所以系统初始化等功能都没有执行，R13R14R15内容为随机数，R15如图所示也就是PC并未指向我们的代码，我们必须通过手动修改的形式将PC指向我们需要执行的代码，根据我们所用芯片，应将R15赋值为0x08000000，修改后如图6.1.11所示。**

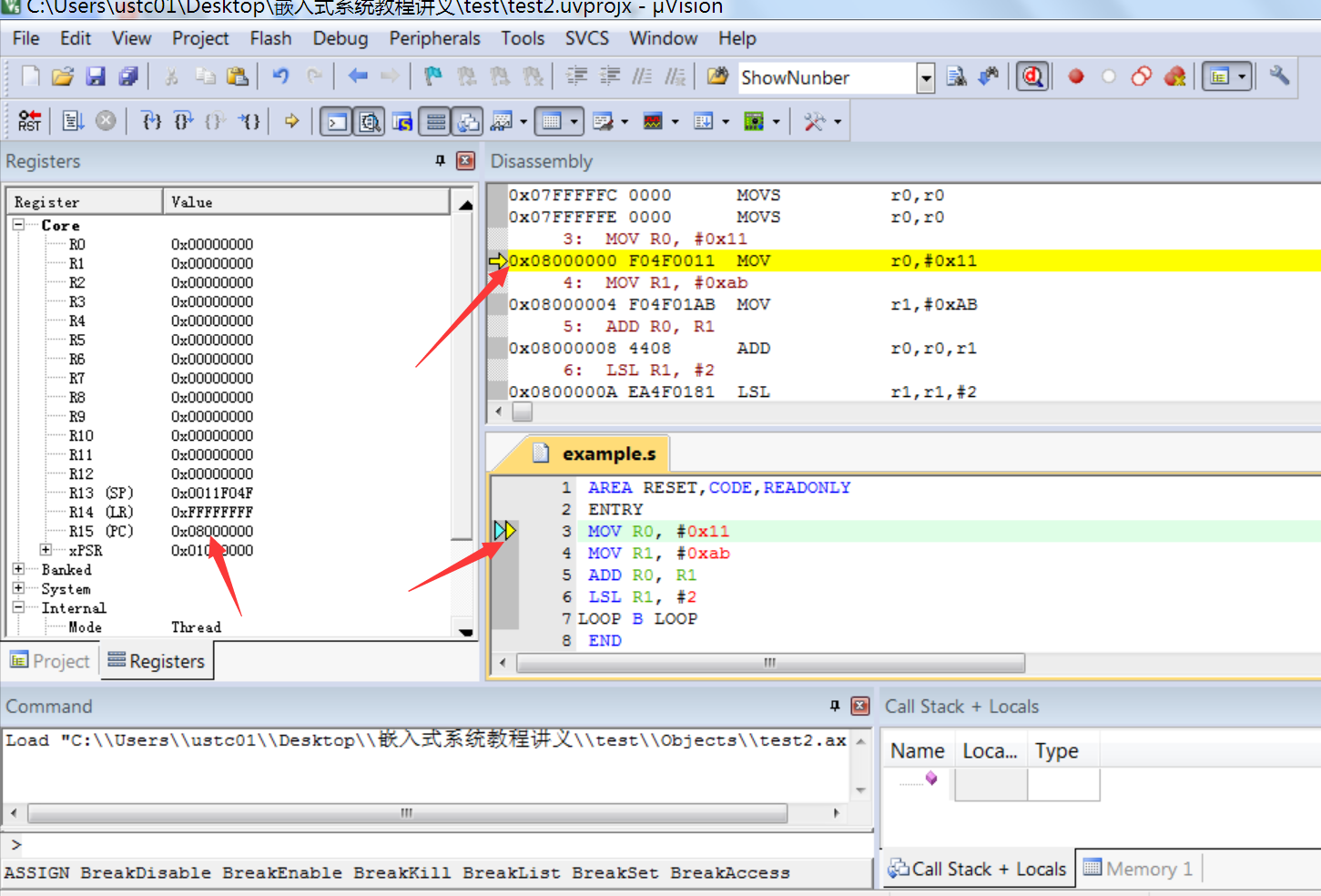


图6.1.11 修改PC寄存器后的调试状态

此时，PC指向我们刚刚写好的汇编代码，程序可以进入正常调试状态。调试主要有两种模式，单步执行和设断点，这两种模式结合使用效果更好。单步执行点击选择快捷键F10（step over）或F11（step in）进行单步跟踪并观察寄存器或者内存状态；设短点则在相应代码行设置断点，程序到此处会停止执行。本汇编代码中，我们可以将断点设在LOOP标号死循环处，并查看程序运行结果，如图6.1.12所示。

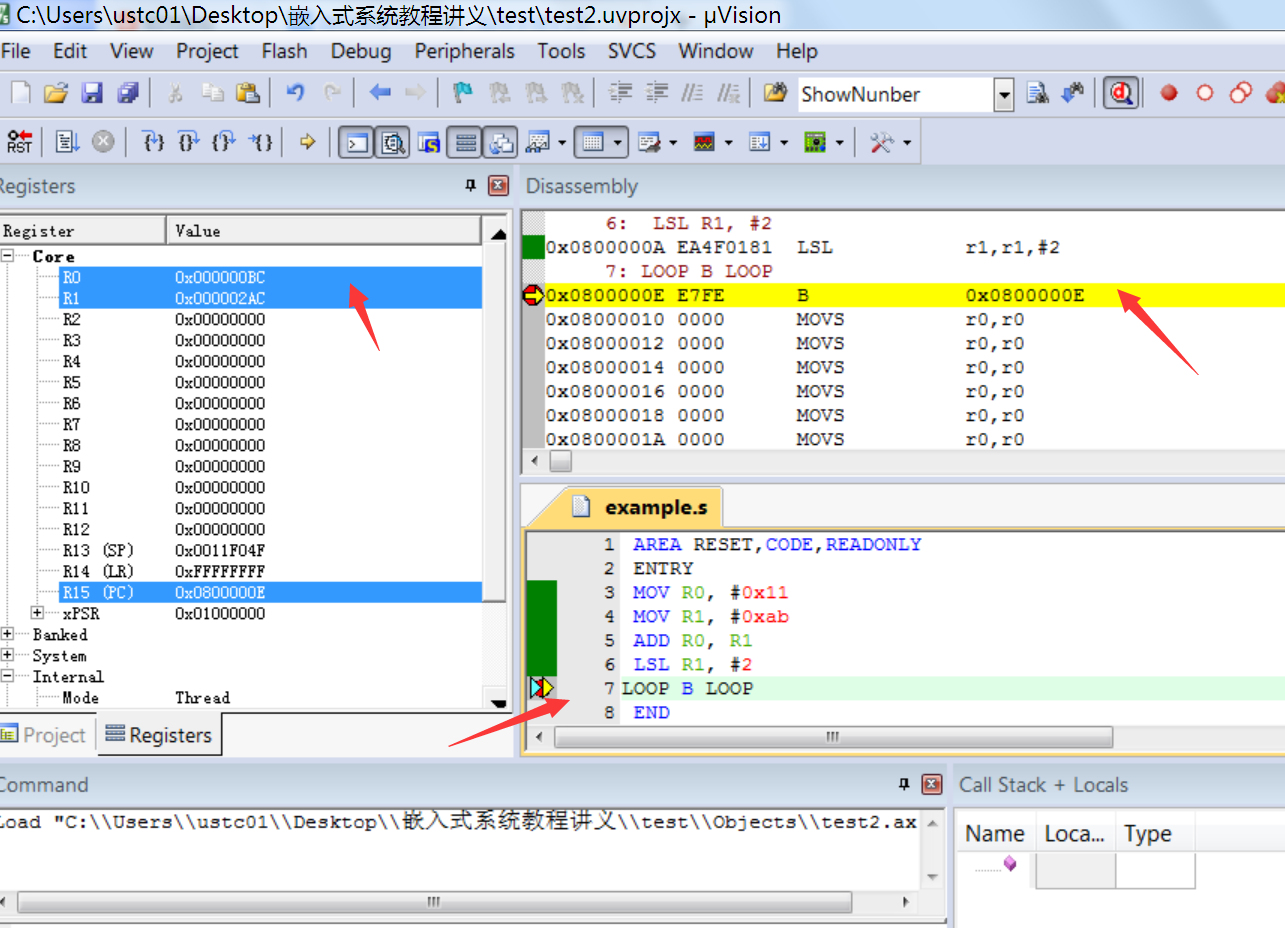


图6.1.12 设置断点后程序执行结果

我们可以看到程序执行结束后R0=0xAB+0x11=0xBC,R1=0xAB左移两位后为0x000

002AC。

汇编程序具有执行效率高、占用内存资源少等优点，但是对于新手来说上手门槛较高，同时因为不能利用系统自带初始化程序进行初始化，自己实现这些过程操作过于繁琐，所以对于绝大多数开发者来说都是利用C语言来开发项目，在必要时可以通过调用汇编子程序或者嵌入汇编代码的方式使用汇编，后面开始介绍利用C语言开发嵌入式应用程序的基本流程。

## 6.2 C程序中调用汇编函数

ARM编译器使用的函数调用规则就是 ATPCS标准，也是设计可被C程序调用的汇编函数的编写规则。为了保证程序调用时参数传递正确，C程序调用的汇编函数时必须严

格按照ATPCS规则。

如果汇编函数和调用函数的C程序不在同一个文件中，则需要在汇编程序中用**EXPORT**声明汇编语言起始处的标号为外部可引用符号，该标号应该为C语言中所调用函数的名称。

以实验1中LED工程做为模板，测试C程序调用汇编函数。修改**main.c中代码。**

**void TestAsm(void);**

**int main ()**

**{**

**TestAsm();**

**return 0;**

**}**

**添加汇编源文件test.s保存到USER目录，并将test.s添加到Groups USER下，如图6.2.1所示：**

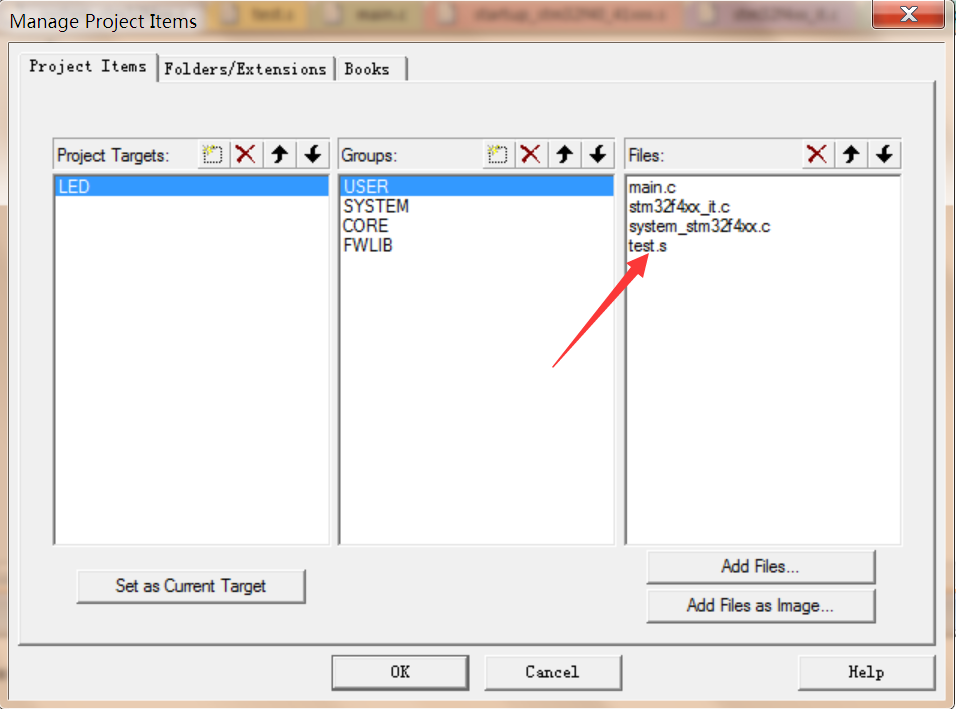


图6.2.1 工程中添加汇编源文件

test.s代码如下：

**AREA Buf,DATA,READWRITE;**

**Num1 DCD 0x11;**

**Nums DCD 0x22;**

**AREA BUF,CODE,READONLY**

**EXPORT TestAsm**

**ENTRY**

**TestAsm**

**LDR R0,=Num1**

**LDR R1,[R0]**

**ADD R1,#0x9A**

**STR R1,[R0]**

**LDR R0,=Nums**

**LDR R2,[R0]**

**ADD R2,#0xAB**

**STR R2,[R0]**

**LOOP B LOOP**

**NOP**

**END**

**编译、链接通过后，进入调试状态**Debug->Start/Stop Debug Session**，在图6.2.2所示设定断点，运行程序（F5）。**

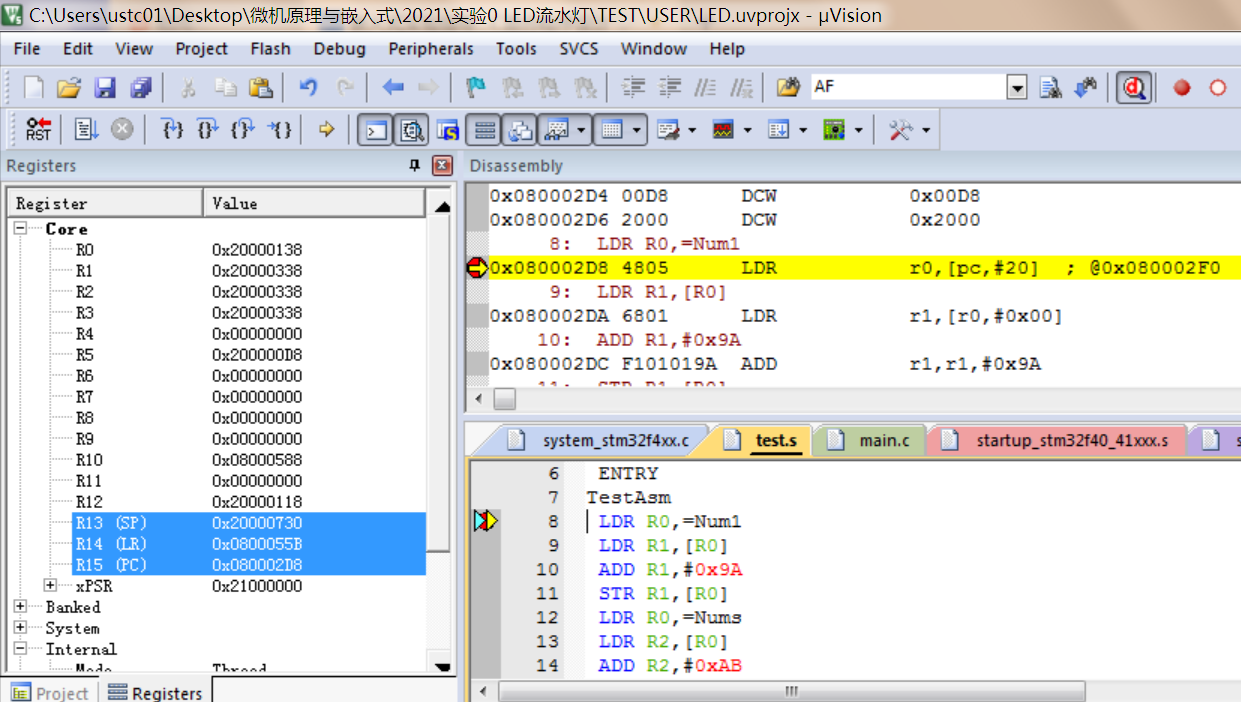


图6.2.2 运行程序至断点处

**通过内存窗口观察初始状态Num1和Nums存储位置和内容，单步跟踪程序执行，并在内存窗口观察内存状态变化，如图6.2.3所示：**

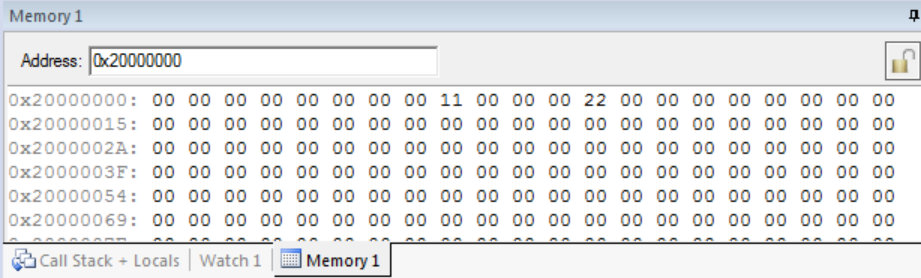


图6.2.3 初始状态下内存单元存储内容

## 6.3 实验步骤及练习

1. 在6.1工程下修改代码，实现1+2+3+...+N，N=100的累加求和，计算结果存入R0寄存器。

2. 在6.1工程下修改代码如下：

Stack\_Size EQU 0x0000010

AREA STACK, READWRITE

\_\_stack\_limit

Stack\_Mem SPACE Stack\_Size

\_\_initial\_sp

AREA RESET,CODE,READONLY

Vectors DCD \_\_initial\_sp

ENTRY

LDR R0,= Vectors

LDR R1,[R0]

MOV R2, #0x11

PUSH {R2}

LOOP B LOOP

END

编译后调试代码，进入调试状态先观察内存0x08000000中内容及SP寄存器内容，再修改PC寄存器为0x08000004，xPSR为0x01000000，开始单步执行程序并从内存单元0x20000000开始观察内存变化，并解释变化结果。

1. 以6.2工程为模板实现讲义7.4.1中例7.48 C程序调用汇编函数实例，在调试模式下运行并观察结果。
2. 以6.1工程为模板实现讲义7.4.1中例7.49汇编程序调用C函数实例，在调试模式下运行并观察结果。