Homework6 软中断测试

Problem

1. 编写下列函数,然后编写程序通过软中断方式调用，计算出结果。 ：

（1）64位带进位的加法运算（asm）；

（2） 两个32位数相乘，结果为64位的乘法运算（C 或 asm）。

Answer

程序

程序主要由get_swi_num.s，swi.c，swi.h和c_swi_handle.s文件组成

其中：

- get_swi_num.s代码：

```
        AREA        TopSwi, CODE, READONLY
        IMPORT      C_SWI_Handler
        EXPORT      SWI_Handler

SWI_Handler
        STMFD       sp!,{r0-r12,lr}
        LDR         r0, [lr,#-4]
        BIC         r0, r0,#0xff000000
        MOV         R1, SP
        BL          C_SWI_Handler
        END
```

- swi.c代码：

```
#include <stdio.h>
#include "swi.h"
unsigned *swi_vec = (unsigned *)0x08;
extern void SWI_Handler(void);

unsigned Install_Handler (unsigned *handlerloc, unsigned *vector)
{
    unsigned vec, oldvec;
    vec = ((unsigned)handlerloc - (unsigned)vector - 0x8)>>2;
    if ((vec & 0xFF000000) != 0)
     { return 0;}
    vec = 0xEa000000 | vec;
    oldvec = *vector;
    *vector = vec;
    return (oldvec);
}

int main( void )
{
    long long res;
```

```
        long long res1;

        long long a = 320000;
        long long b = 640000;

        int al = a;
        int ah = a >> 32;
        int bl = b;
        int bh = b >> 32;

    Install_Handler((unsigned *) SWI_Handler, swi_vec);

    res = add_two(al, ah, bl, bh);
        res1 = mut_two(al, bl);

    return 0;
}
```

- swi.h代码:

```
__swi(0) int add_two(int, int, int, int);

__swi(1) int mut_two(int, int);
```

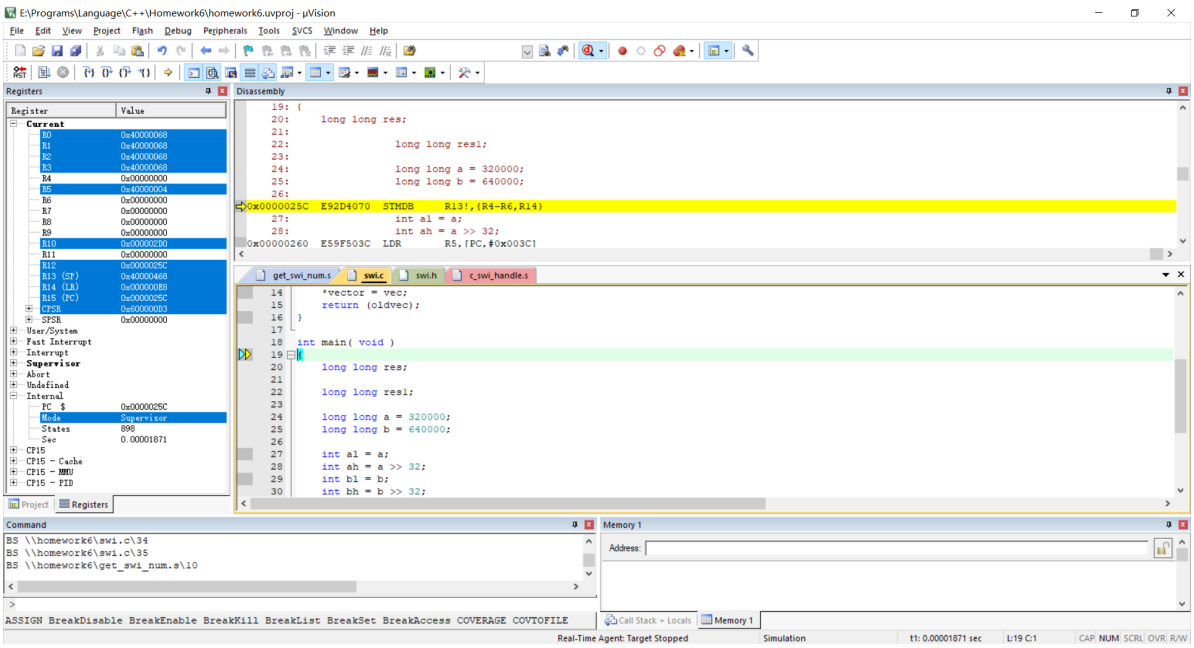
- c_swi_handle.s代码:

```
        AREA SecondSwi, CODE, READONLY
        EXPORT     C_SWI_Handler
C_SWI_Handler
        CMP    r0, #0x01000000          ; Range check?
        LDRLE  pc, [pc,r0,LSL #2]       ;(PC->DCD SWInum0)
        B       SWIOutOfRange
SWIJumpTable
        DCD    SWInum0
        DCD    SWInum1
SWInum0   ; SWI number 0 code
        B    EndofSWI
SWInum1   ; SWI number 1 code
        B    EndofSW

EndofSWI
        LDMFD   sp!, {r0-r3}
        ADDS    r0 , r0, r2
        ADC     r1 , r1, r3
        SUB     lr, lr, #4
        LDMFD   sp!, {r4-r12,pc}^

EndofSW
        LDMFD   sp!, {r0-r1}
        SMULL   r2, r3, r0, r1
        MOV     r0, r2
        MOV     r1, r3
        SUB     lr, lr, #4
        LDMFD   sp!, {r2-r12,pc}^
```

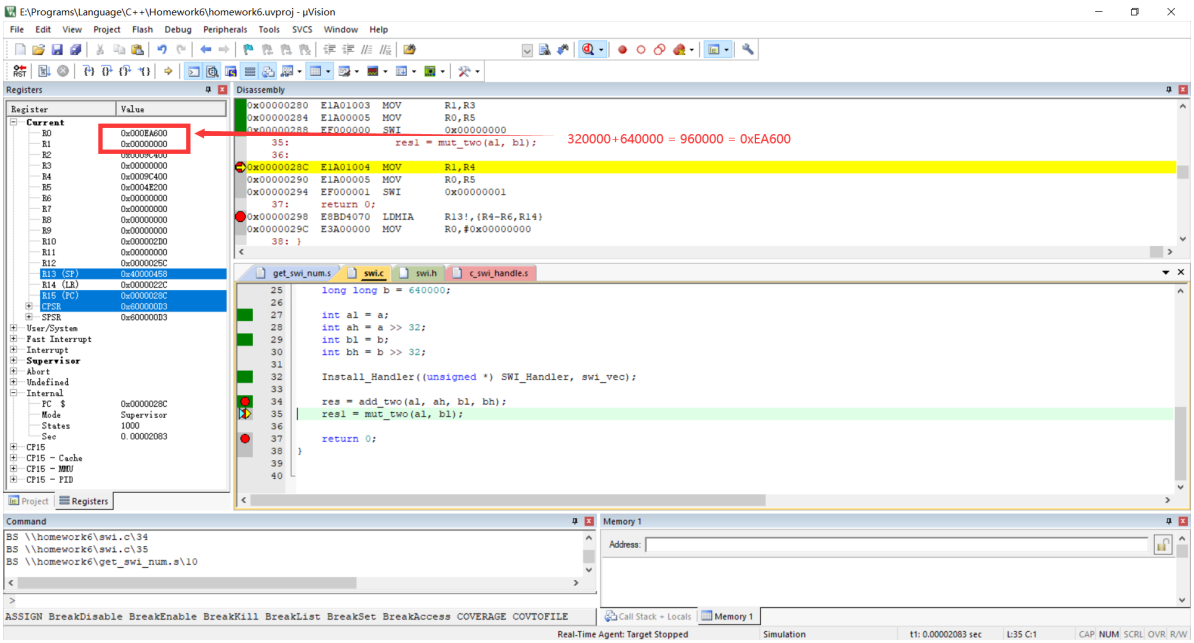
```
SWIOutOfRange
        SUB     lr, lr, #4
        LDMFD   sp!, {r0-r12,pc}^

        END
```

分析

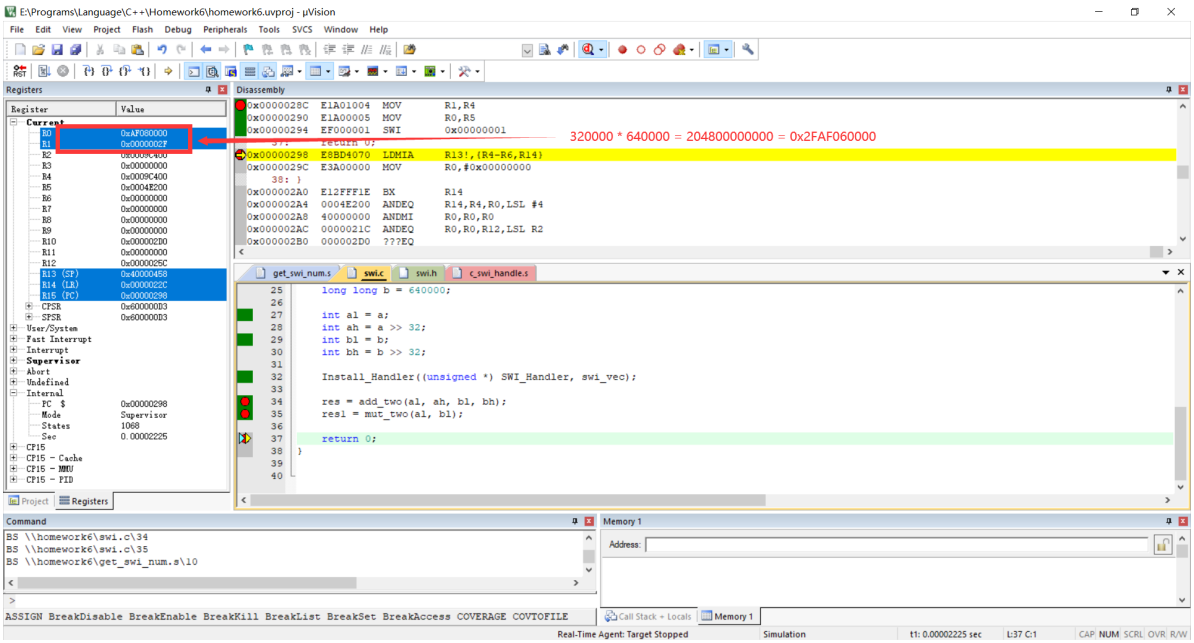
- 编译进行debug，得到：



- 首先，验证64位加法操作，传入两个64位数字a，b，用al，ah，bl，bh分别存储每个64位数字的高32位和低32位，然后进行赋值，传入中断号位0的加法操作函数中，输入的是320000和640000相加，得到低32位0x000EA600和高32位0x00000000，符合逻辑：

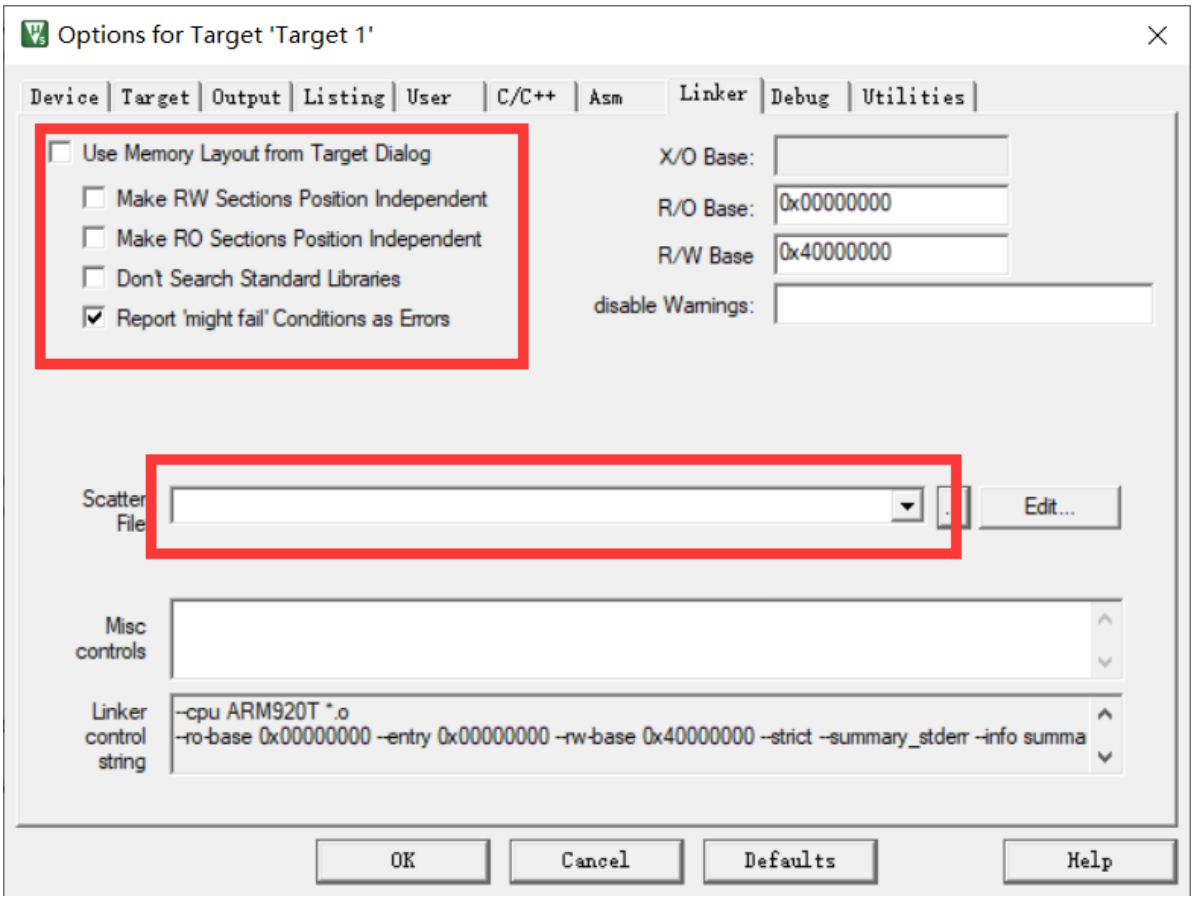


- 下面验证两个三十二位有符号数的乘法，还是传入320000和640000相乘，得到0x2FAF060000，即高32位存0x0000002F，低32位存0xAF060000，符合逻辑：



遇到的问题

- c代码的变量声明要在任意一个函数执行前全部声明完毕，否则报错
- 在执行代码过程中最好先再设置中选择Linker中去掉`Use Memory.....`的对勾，同时删除默认选中的`scatter file`



- 记得同第四次作业一样在配置中加入aplusb.ini配置初始化文件获取内存的**写权限**