## 一、实验目的

通过c语言作为主函数，利用汇编语言进行冒泡排序算法程序设计，从而实现C语言与汇编语言的混合使用，实现冒泡排序方法。

## 二、实验原理

2.1、arm developer suite是[ARM公司](https://www.baidu.com/s?wd=ARM%E5%85%AC%E5%8F%B8&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "_blank)的开发环境(IDE),也就是常说的ADS，是对于ARM开发环境，并利用arm developer suite带的仿真器AXD Debugger进行仿真调试。

## 三、实验环境

CoderWarrior arm developer suit、AXD Debugger

## 四、实验程序代码

4.1、 C语言代码

#include <stdio.h>

#define N 6

extern void sort(char \* ,int);

int main()

{

char a[N]={'a','z','c','q','w','e'};

int i;

printf("排序前：\n");

for(i=0;i<N;i++)

printf("%c\t",a[i]);

sort(a,N);

printf("\n 排序后： \n");

for(i=0;i<N;i++)

printf("%c\t",a[i]);

return 0;

}

代码实现步骤：

1. 声明静态子函数，用来实现汇编算法
2. 初始化char类型数组，用于排序
3. 在控制台上输出对应的排序前，排序后的数组
4. 利用sort函数，进行冒泡排序，并在控制台中输出

4.2、汇编语言代码

area sort ,code ,readonly

export sort

start

mov r3,r0

mov r7,r1 ;第一层计数,i=N

loop0

sub r7,r7,#1 ;r7=i=N-1,倒计数

cmp r7,#0

beq endh ;相等时跳转

mov r2,#0 ;第二层循环计数,j=0

loop

add r4,r2,#1

ldrb r5,[r3,r2]

ldrb r6,[r3,r4]

cmp r5,r6

bgt jh ;r5>r6,交换

b judge ;不交换

jh

strb r5,[r3,r4]

strb r6,[r3,r2]

b judge

judge

;add r3,r3,#1

add r2,r2,#1 ;j=j+1

cmp r2,r7 ;内循环次数, i与j

bne loop ;不相等，跳转

b loop0

endh

end

代码实现步骤：

1. 将数组地址r1，复制到r3中；将传入的数据个数N(r1)，复制到r7中
2. 在loop0处，是外层循环的开始，先进行减一操作，此时r7=i=N-1,r7进行倒计数来控制外层循环
3. 初始化r2，用于控制内层循环，r2=j
4. Loop处，为内层循环的开始，先初始化r4，作为r4=j+1，来控制数组第二个比较的位数
5. 用ldrb，从内存中装载到寄存器中，进行比较
6. 符合r5>r6，跳转到jh中，进行数组位置交换，将大的数据放到后面，然后进行跳转judge；不符合则跳转到judge中，进行加法，即r2=j=j+1,实现内层循环次数加1
7. 用r2与r7中数据来比较，判断是否完成内层循环次数，可参考如下图

正常情况下，冒泡排序采用

for(i=0;i<n-1;i++)

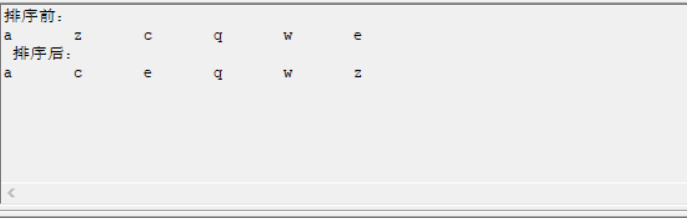
for(j=0;j< n-i-1;j++)

倒着计数与正常情况下的比较（假设共5个数据）

|  |  |  |
| --- | --- | --- |
| 倒着计数次数（外层） | 正着计数次数（外层） | 内层循环计数次数 |
| 4 | 1 | 4 |
| 3 | 2 | 3 |
| 2 | 3 | 2 |
| 1 | 4 | 1 |

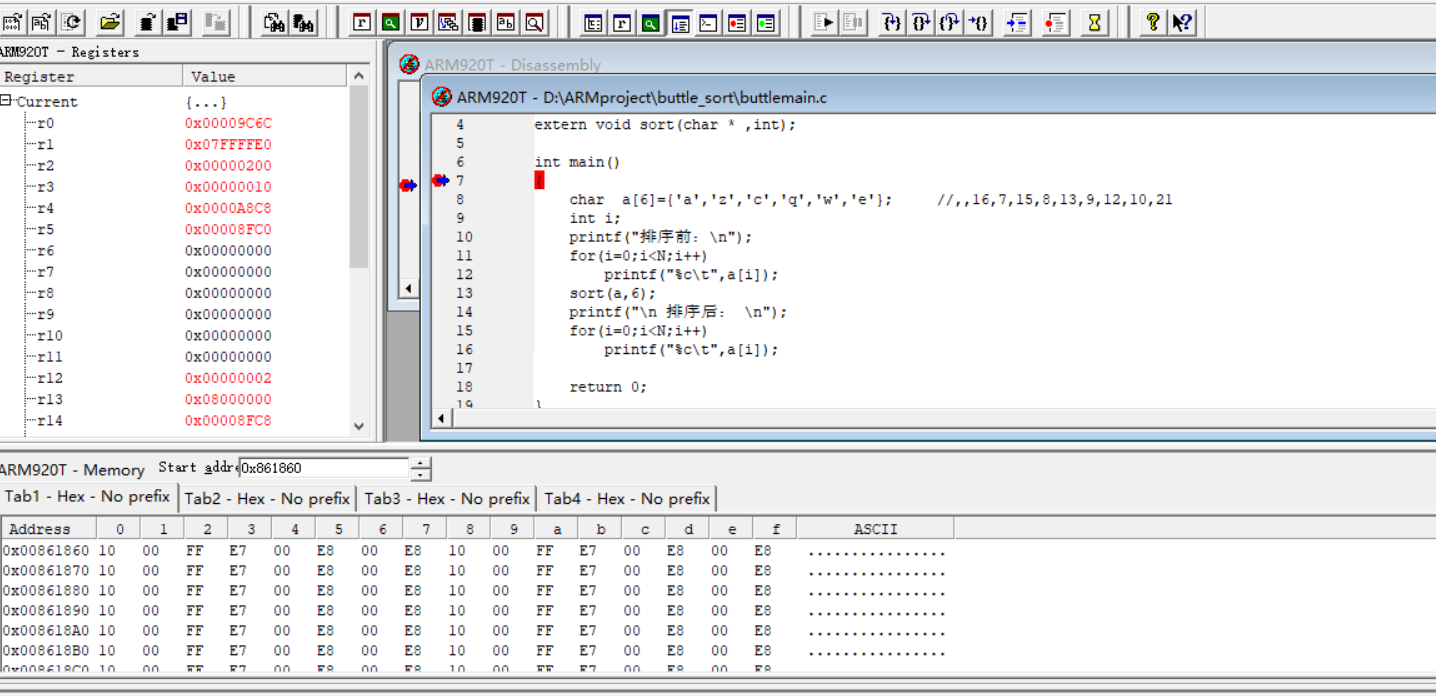
1. 内层循环结束，跳转到外层循环处，进行下一轮循环，每一次内层循环，都会将最大的数据放到循环所处的最高位。
2. 数据排序次数结束，跳转到endh,结束冒泡排序算法

## 五、实验结果

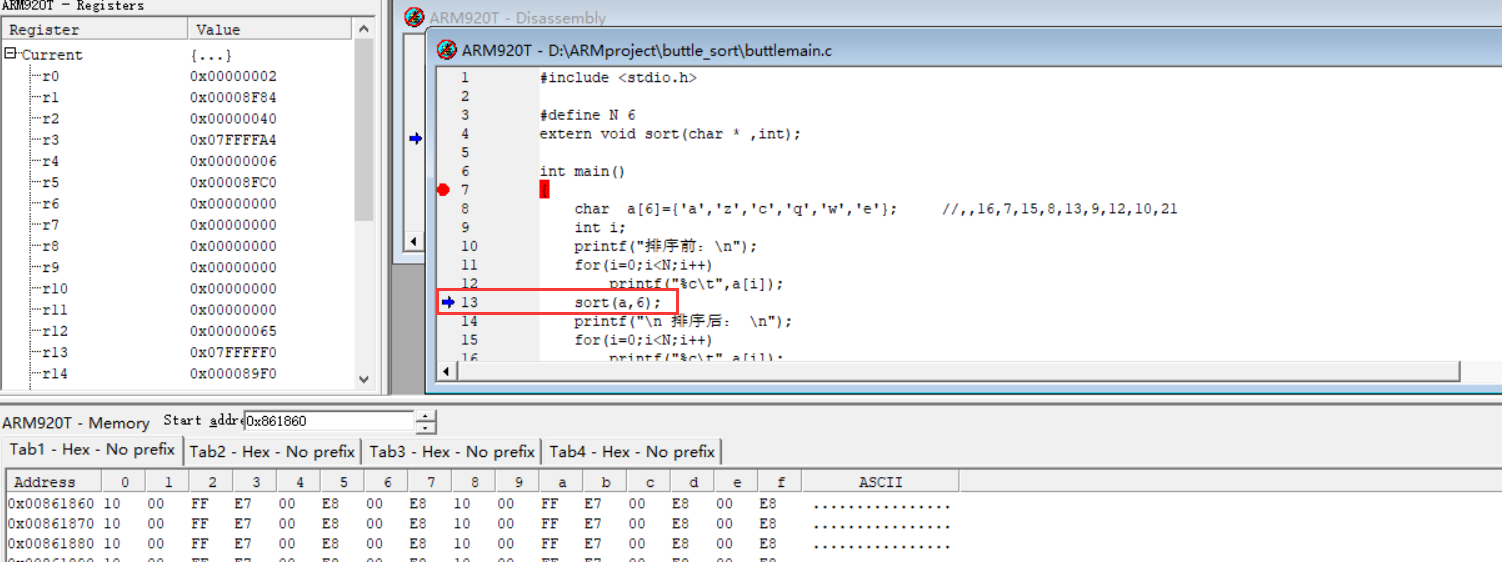


## 六、AXD调试

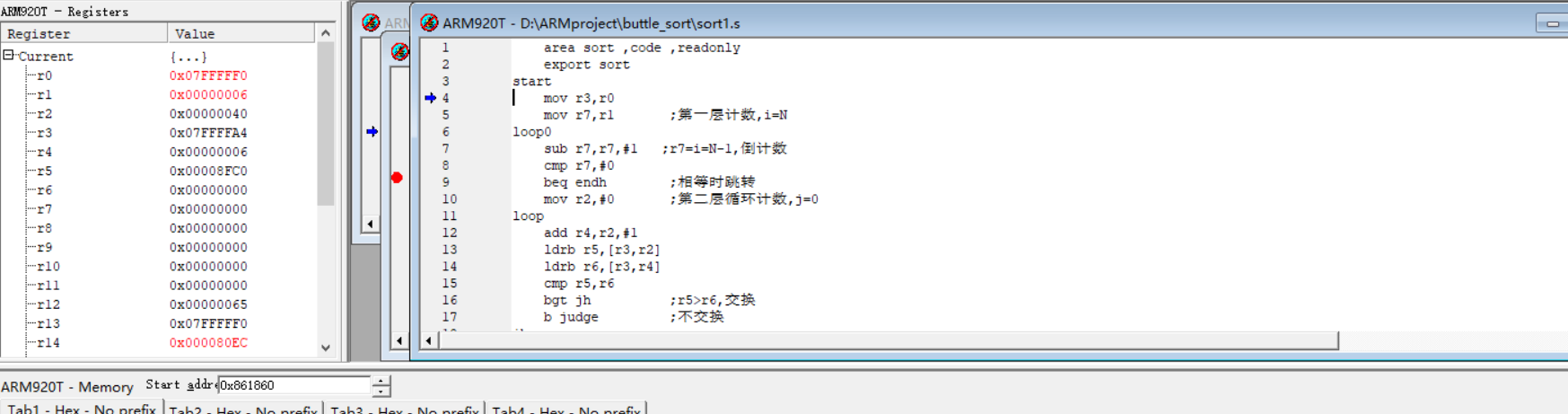
6.1、点击运行，进入到主函数处



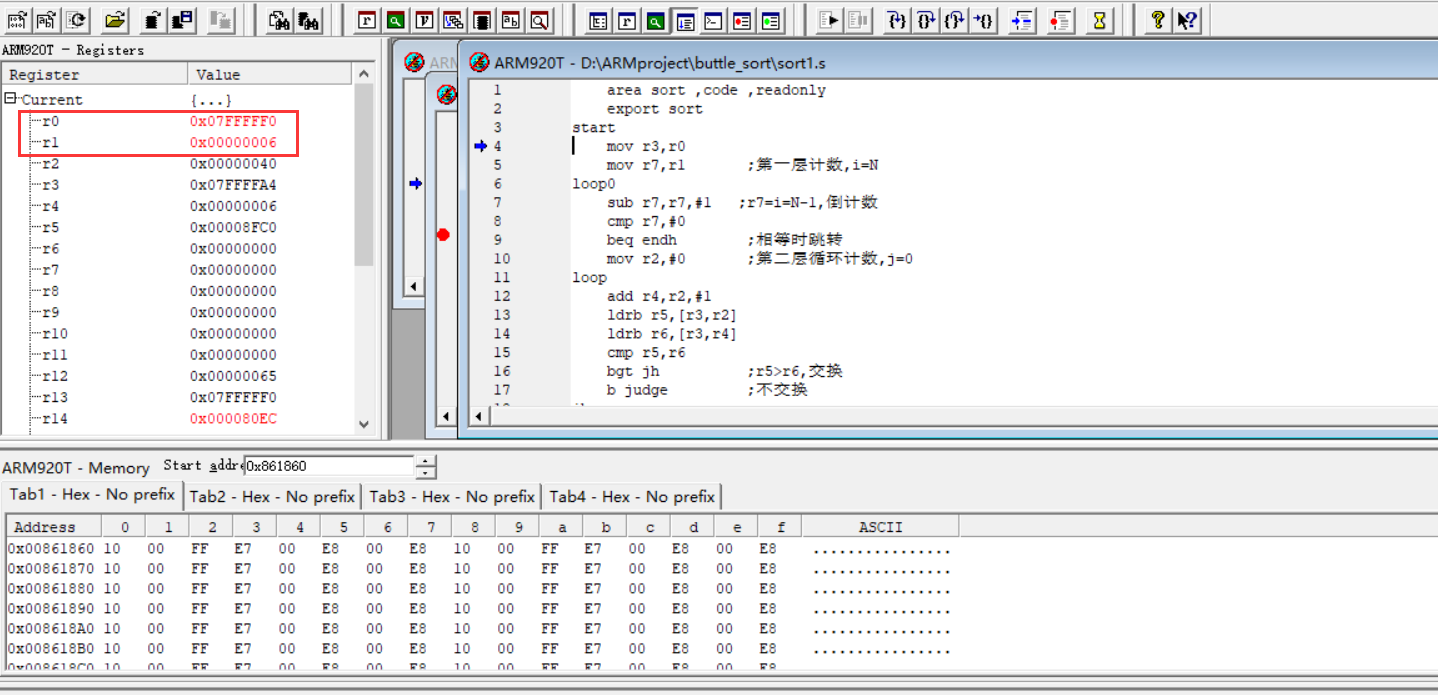
6.2、进行单步执行，进行数据初始化



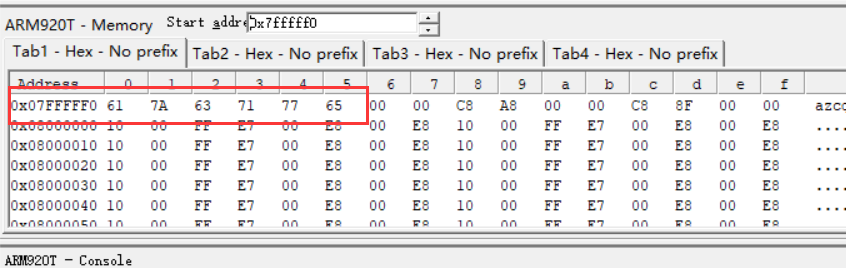
6.3、到sort函数处，step in进入此函数，进行算法调试



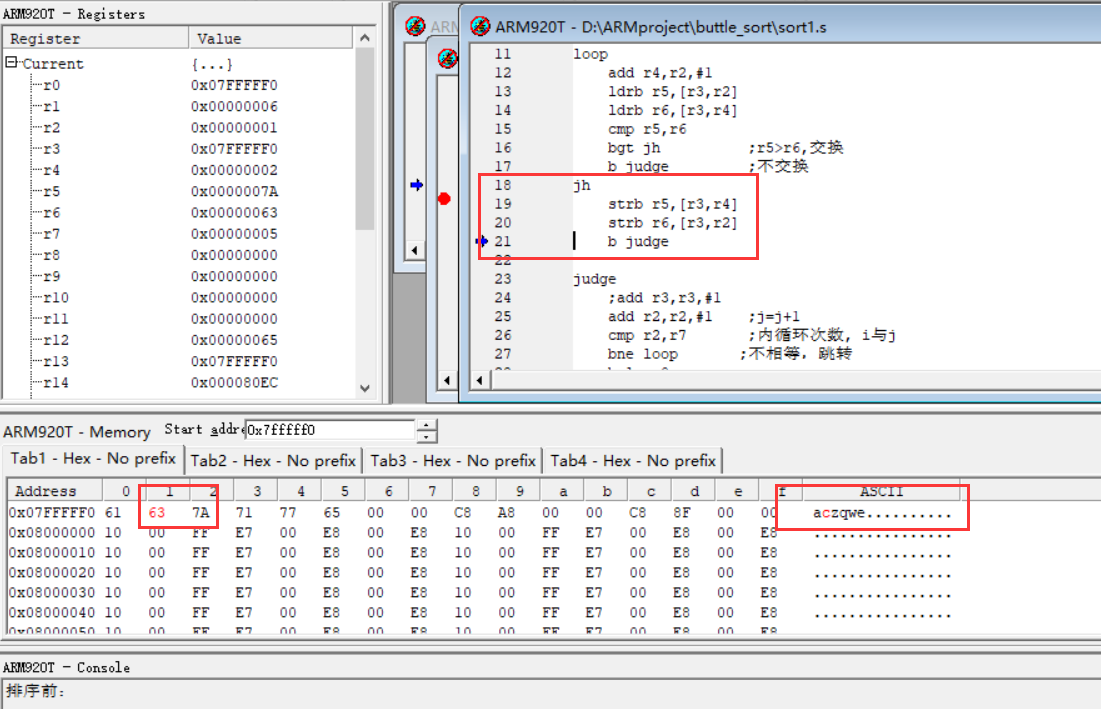
6.4、在途中，r0为0x07fffff0存储的是数组的首地址，r1为0x00000006存储的是数组的个数



6.5、以下在内存中，储存的是数组数据



6.6、可以看出，在原顺序'a','z','c','q','w','e'，z>c，因此跳转到jh中，进行数据交换，到内存中，可以看到数据已经交换完成



6.7、继续进行单步调试，内层循环后，进入到外层循环，进入下一轮内层循环，直到最后，数组排序完成结束。