字符串比较

郑悟强 PB22051082

2023.11.29

1 实验目的

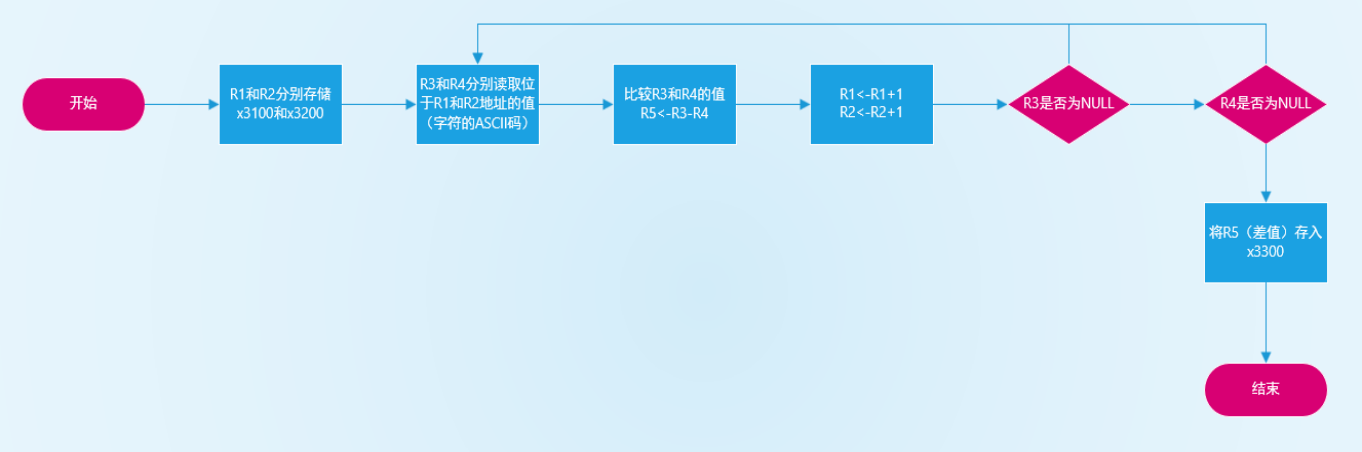
本实验需要通过 lc3 汇编实现 c 语言中 strcmp 函数的功能。

给定两个字符串 S1 和 S2，这两个字符串的起始地址分别是 x3100 和 x3200 。字符串中的每个字符都存储在连续的内存位置，字符串是以 NULL 字符结尾。可以假定 S1 和 S2 仅包含 a-z、A-Z 中的字符和作为终止符的 NULL 字符。请注意，0 < strlen（S1）、strlen（S2）< 100 。

实验目的：读取字符串 s1,s2，将 strcmp（s1, s2）的返回值存储在 x3300 中。

2 程序设计

2.1 总体思路



2.2 核心操作 1：计算每一个字符的差值

因为字符以 ASCII 码值的格式存储在内存中，所以直接读入对应地址的内容即为对应字符的 ASCII 码的二进制形式，只需对两值进行减法即可。

减法原理：R0 存储 R4（s2 对应字符的内容）的取反 +1，R5<-R3（s1 对应字符的内容）+ R0，R5 即为差值。

具体代码：

```
LOAD        LDR R3, R1, #0  ;store the num of an character of s1
            LDR R4, R2, #0  ;store the num of an character of s2
            AND R5, R5, #0  ;R5 initialize to 0
JUDGE       NOT R0, R4
            ADD R0, R0, #1  ;R0<- -s2
            ADD R5, R3, R0  ;R5<- s1 - s2
            BRnp STORE      ;if R5!=0 , the result of strcmp is R5
```

2.3 核心操作 2：判断字符串是否到头

由于字符串的结尾会有一个 NULL 作为表示，NULL 的 ASCII 码为 x0000，所以读入到寄存器，寄存器内容为 0，只需判断两个字符内容寄存器 R3 和 R4 的值，若为 0，则说明一个字符串已经结尾，直接跳转到存储部分结尾。

具体代码：

```
        ADD R3, R3, #0  ;if R3 has moverd to the NULL
        BRz STORE
        ADD R4, R4, #0  ;if R4 has moved to the NULL as well
        BRz STORE
        BRnzp LOAD
```

2.4 完整代码

```
        .ORIG x3000
;
;Initialize
;
        LD  R1, S1_ADDR ;store the address of an character of s1
        LD  R2, S2_ADDR ;store the address of an character of s2
;
;Circulation
;
LOAD    LDR R3, R1, #0  ;store the num of an character of s1
        LDR R4, R2, #0  ;store the num of an character of s2
        AND R5, R5, #0  ;R5 initialize to 0
JUDGE   NOT R0, R4
        ADD R0, R0, #1  ;R0<- -s2
        ADD R5, R3, R0  ;R5<- s1 - s2
        BRnp STORE      ;if R5!=0 , the result of strcmp is R5
        ADD R1, R1, #1  ;R0<-R0+1 store the next character
        ADD R2, R2, #1  ;R2<-R2+1 store the next character
        ADD R3, R3, #0  ;if R3 has moverd to the NULL
        BRz STORE
        ADD R4, R4, #0  ;if R4 has moved to the NULL as well
        BRz STORE
        BRnzp LOAD
;
;Store the answer
;
STORE   STI R5, RESULT  ; write back the result
        HALT


S1_ADDR .FILL x3100
S2_ADDR .FILL x3200
RESULT  .FILL x3300
        .END
```

3 过程中遇到的错误

3.1 动态读取地址的问题

关于读取一连串字符每一个的内容，一开始出现了问题，无法将使用.FILL 存储的内容进行改动。

解决方案：使用.FILL 操作存储两个地址内容为 x3100 和 x3200。然后先通过 LD 操作，将 R1 和 R2 分别读入

x3100 和 x3200，通过 LDR R3, R1, #0 的方法读取到 R1 内容的值。最后在每次循环中，操作结束后再将 R1 和 R2 加一。

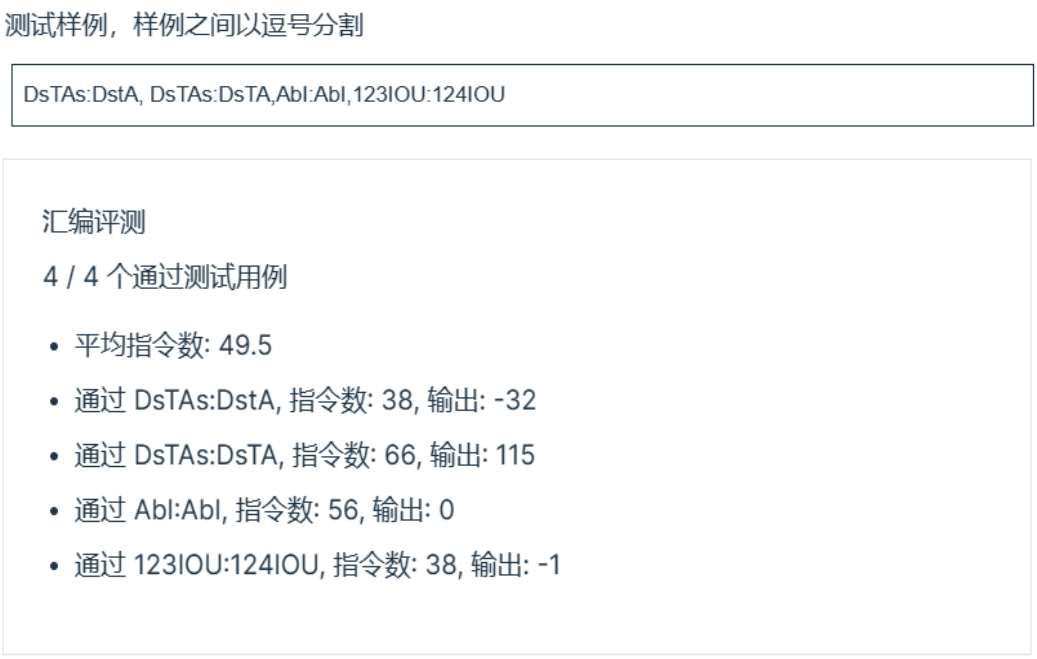
3.2 寄存器内容变动问题

由于需要将 R3 和 R4 的内容作差时，需要将 R4 内容取反加一，之后在判断 R4 内容是否为 0 时，结果会出现错误。

解决方案：使用其他寄存器来存储 R4 取反加 1 的值，这样来保护 R4 不收变动。

4 调试结果

使用助教提供的自测网站。



5 代码效率

本程序由于每次判断是否 R3R4 为 0，所以当一个为 0 时就结束循环，时间复杂度为 $\mathbb{O}(min(len(s1), len(s2)))$。

6 实验体会与收获

通过本实验我熟悉了关于 lc3 存储字符串的原理，及对于字符串的操作方法，同时对于动态读取内存内容有了更深的了解。