ICS-lab3 实验报告

孔浩宇 PB20000113

2022 年 12 月 13 日

目录

1 实验目的

对于一个储存在从 x3101 开始的连续的内存位置的、长度为 N 的字符串，求出其最长的连续重复字符子串的长度，并将结果存储在 x3050 中.

2 实验原理

2.1 实现 if 语句

(1) if A==B.

```
        ADD     R, A, #0
        NOT     R, R
        ADD     R, R, #1     ; R=-A
        ADD     R, R, B      ; R=B-A
        BRnp     AFTER       ;
        ...                  ; if(B-A==0) do this
    AFTER
```

(2) if A>B.

```
        ADD     R, A, #0
        NOT     R, R
        ADD     R, R, #1     ; R=-A
        ADD     R, R, B      ; R=B-A
        BRzp     AFTER       ;
        ...                  ; if(B-A<0) do this
AFTER
```

(3) if A≥B.

```
        ADD     R, A, #0
        NOT     R, R
        ADD     R, R, #1     ; R=-A
        ADD     R, R, B      ; R=B-A
        BRp     AFTER       ;
        ...                  ; if(B-A<=0) do this
AFTER
```

(4) 其余 A<B, A≤B 类似，不再赘述.

2.2 判断当前重复子串长度及更新最大重复子串长度

用 R_4 来存储当前正在判断的重复字符串的长度，用 R_7 来存储历史最长重复字符串的长度，R_2 为已判断的字符串末尾，R_3 为子串末尾后一位

```
        NOT     R2, R2
        ADD     R2, R2, #1
        ADD     R6, R2, R3
        BRnp    Neq             ; if R2 != R3 , do Neq
        ADD     R4, R4, #1
        BRnzp   Nor             ; if R2 == R3, R4++, do Nor
Neq     ADD     R6, R4, #0
        NOT     R4, R4
        ADD     R4, R4, #1
        ADD     R4, R4, R7
        BRzp    Notre
        ADD     R7, R6, #0      ; if now > max, max <= now
Notre   AND     R4, R4, #0
        ADD     R4, R4, #1      ; R4 <= 1
Nor     ADD     R1, R1, #1
        LDR     R2, R1, #0      ; R2 is S[i]
        LDR     R3, R1, #1      ; R3 is S[i+1]
        ADD     R0, R0, #-1
        BRnzp   Loop
```

3 实验步骤

3.1 初始化

(0) 标号

```
RESULT  .FILL   x3050
NUM     .FILL   x3100
DATA    .FILL   x3101
```

(1) 读入 NUM 及 $DATA$ 等变量

```
LDI     R0, NUM
LD      R1, DATA        ; R1 is the pointer of the string
```

(2) 初始化其他变量

```
LDR     R2, R1, #0      ; R2 is S[i]
LDR     R3, R1, #1      ; R3 is S[i+1]
ADD     R4, R4, #1      ; R4 <= 1   (now)
ADD     R7, R7, #1      ; R7 <= 1   (max)
```

3.2 特殊情况

若 $NUM == 0$，则最长重复子串为 0

```
Zero    ADD     R6, R0, R0      ; judge if R0 == 0
        BRnp    Loop
        AND     R7, R7, #0      ; if NUM ==0, R7 <= 0
        BRnzp   End             ; JUMP to End
```

3.3 循环

当判断完 NUM 个字符后，结束循环。

```
Loop    ADD     R6, R0, #-1     ; judge if R0 == 1
        BRz     Store
        NOT     R2, R2
        ADD     R2, R2, #1
        ADD     R6, R2, R3
        BRnp    Neq             ; if R2 != R3 , do Neq
        ADD     R4, R4, #1
        BRnzp   Nor             ; if R2 == R3, R4++, do Nor
Neq     ADD     R6, R4, #0
        NOT     R4, R4
        ADD     R4, R4, #1
        ADD     R4, R4, R7
        BRzp    Notre           ; if now > max, max <= now
        ADD     R7, R6, #0
Notre   AND     R4, R4, #0
        ADD     R4, R4, #1      ; R4 <= 1
Nor     ADD     R1, R1, #1
        LDR     R2, R1, #0      ; R2 is S[i]
        LDR     R3, R1, #1      ; R3 is S[i+1]
        ADD     R0, R0, #-1
        BRnzp   Loop
```

3.4 储存结果

先判断是否应更新 max，再将 max 存储到对应位置

```
Store   ADD     R6, R4, #0
        NOT     R4, R4
        ADD     R4, R4, #1
        ADD     R4, R4, R7
        BRzp    End
        ADD     R7, R6, #0
End     STI     R7, RESULT
```

3.5 结束

```
    HALT.
```

3.6 代码

```
    .ORIG x3000
        LDI     R0, NUM
        LD      R1, DATA        ; R1 is the pointer of the string
        LDR     R2, R1, #0      ; R2 is S[i]
        LDR     R3, R1, #1      ; R3 is S[i+1]
        ADD     R4, R4, #1      ; R4 <= 1   (now)
        ADD     R7, R7, #1      ; R7 <= 1   (max)
Zero    ADD     R6, R0, R0      ; judge if R0 == 0
        BRnp    Loop
        AND     R7, R7, #0      ; R7 <= 0
        BRnzp   End
Loop    ADD     R6, R0, #-1     ; judge if R0 == 1
        BRz     Store
        NOT     R2, R2
        ADD     R2, R2, #1
        ADD     R6, R2, R3
        BRnp    Neq
        ADD     R4, R4, #1
        BRnzp   Nor
Neq     ADD     R6, R4, #0
        NOT     R4, R4
        ADD     R4, R4, #1
        ADD     R4, R4, R7
        BRzp    Notre
        ADD     R7, R6, #0
Notre   AND     R4, R4, #0
        ADD     R4, R4, #1      ; R4 <= 1
Nor     ADD     R1, R1, #1
        LDR     R2, R1, #0      ; R2 is S[i]
        LDR     R3, R1, #1      ; R3 is S[i+1]
        ADD     R0, R0, #-1
        BRnzp   Loop
Store   ADD     R6, R4, #0
        NOT     R4, R4
        ADD     R4, R4, #1
        ADD     R4, R4, R7
        BRzp    End
```

```
        ADD     R7, R6, #0
End     STI     R7, RESULT
        HALT
RESULT  .FILL   x3050
NUM     .FILL   x3100
DATA    .FILL   x3101
.END
```

4 实验结果