ARMAssembly:LdrStr:C

Prg3.s

```
01 .text
02 valB:
03 .word 2257
04
05 numStr:
06 .ascii "2097\0"
07 .align 2
08
09 numStrLoc:
10 .word numStr
11
12 .global _start
13 _start:
14
15 ldr R0, valB          @ (Pseudoinstruction) ldr R0, [pc, #-16]
16 mvn R1, #199         @ R1 = -200
17 add R0, R0, R1       @ R0 = valB + -200
18
19 mov R3, #0           @ Will be integer value of "2097\0" when done
20 ldr R1, numStrLoc    @ (Pseudo instruction) R1 gets address of '2' in "2097\0"
21
22 mov R4, #10          @ 10 constant for mul
23
24 loopTop:                                   ldr r4, [r5, #4]!  vs ldr r4, [r5], #4
25 ldrb R2, [R1], #1    @ R2 gets value at R1, and R1 is then incremented
26 cmp R2, #0
27 beq allDone          @ Stop loop if R2 == '\0'
28 sub R2, R2, #48      @ R2 = R2 - '0'
29 mul R5, R3, R4       @ R5 = R3*10
30 add R2, R5, R2       @ R2 = R5 + R2
31 @ Or, mla R2, R3, R4, R2  @ R2 = (R3*10) + R2
32 mov R3, R2
33
34 b loopTop            @ Another time through loop
35
36 allDone:
37 sub R3, R3, R0       @ R3 = atoi("2097") - (valB + -200)
38 rsb R3, R3, #143     @ R3 = 143 - R3
39 ldr R2, =valA        @ Create and load a .word holding the address of valA
40 str R3, [r2]         @ valA = R3
41
42 mov R0, R3           @ Write value as exit code, too
43 mov R7, #1
44 SWI 0
45
46 .data
47 valA:
48 .word 0
```