(An assembly-language version of the original binary code.)

Count the occurrences of each letter (A to Z) in an ASCII string terminated by a NUL character. Lower case and upper case should be counted together, and a count also kept of all non-alphabetic characters (not counting the terminal NUL).

The string starts at x3000.

The resulting histogram (which will NOT be initialized in advance) should be stored starting at x3100, with the non-alphabetic count at x3100, and the count for each letter in x3101 (A) through x311A (Z).

R0 holds a pointer to the histogram (x3100); R1 holds a pointer to the current position in the string; and as the loop counter during histogram initialization R2 holds the current character being counted; and is also used to point to the histogram entry; R3 holds the additive inverse of ASCII '0' (0xFFFC0); R4 holds the difference between ASCII '0' and 'Z' (xFFE6); R5 holds the difference between ASCII '0' and 'Z' (xFFE0); R6 is used as a temporary register.

; starting address is x3000
LEA R0, RIST ; point R0 to the start of the histogram

; fill the histogram with zeroes
AND R6, R6, #0 ; put a zero into R6
LD R1, NUM_BINS ; initialize loop count to 27
ADD R2, R0, #0 ; copy start of histogram into R2

; loop to fill histogram starts here
HFLOOP STR R6, R2, #0 ; write a zero into histogram
ADD R5, R2, #1 ; point to next histogram entry
decrement loop count
BRp HFLOOP ; continue until loop count reaches zero

; initialize R1, R3, R4, and R5 from memory
LD R3, NEG_AT ; R3 holds additive inverse of ASCII '0'
LD R4, AT_MIN_Z ; R4 holds difference between ASCII '0' and 'Z'
LD R5, AT_MIN_BQ ; R5 holds difference between ASCII '0' and 'Z'
LD R1, STR_START ; point R1 to start of string

; the counting loop starts here
COUNTLOOP
LDR R2, R1, #0 ; read the next character from the string
BRz DONE ; found the end of the string
ADD R2, R2, R3 ; subtract '0' from the character
BRp AT_LEAST_A ; branch if > '0', i.e., >= 'A'
NON_ALPHA
LDR R6, R0, #0 ; load the non-alpha count
ADD R6, R6, #1 ; add one to it
STR R6, R0, #0 ; store the new non-alpha count
BRnzp GET_NEXT ; branch to end of conditional structure
AT_LEAST_A
ADD R6, R2, R4 ; compare with 'Z'
BRp MORE_THAN_Z ; branch if > 'Z'

; note that we no longer need the current character
; so we can reuse R2 for the pointer to the correct
; histogram entry for incrementing
ALPHA
ADD R2, R2, R0 ; point to correct histogram entry
LDR R6, R2, #0 ; load the count
ADD R6, R6, #1 ; add one to it
STR R6, R2, #0 ; store the new count
BRnzp GET_NEXT ; branch to end of conditional structure

; subtracting as below yields the original character minus ' '
MORE_THAN_Z
ADD R2, R2, R5 ; subtract ' ' - '0' from the character
BRnz NON_ALPHA ; if <= ' ', i.e., < 'a', go increment non-alpha
ADD R6, R2, R4 ; compare with 'z'
BRnz ALPHANZ ; otherwise, go increment non-alpha
ALPHANZ
ADD R6, R2, R4 ; add one to the number of non-alphas
BRnzp GET_NEXT ; branch to end of conditional structure

GET_NEXT
ADD R1, R1, #1 ; point to next character in string
BRnzp COUNTLOOP ; go to start of counting loop
DONE
HALT ; done

; the data needed by the program
NUM_BINS .FILL #27 ; 27 loop iterations
NEG_AT .FILL xFFFC0 ; the additive inverse of ASCII '0'
AT_MIN_Z .FILL xFFE6 ; the difference between ASCII '0' and 'Z'
AT_MIN_BQ .FILL xFFE6 ; the difference between ASCII '0' and 'Z'
STR_START .FILL STRING ; string stored below for simplicity
HIST .BLKW #27 ; space to store the histogram

STRING .STRINGZ "This is a test of the counting frequency code. AbCd...WxYz."