Lab 7 Report

*Max Saunders*

*Msaunders3772@floridapoly.edu*

*EEL4768: Computer Architecture and Organization*

*Due Date: 25 October 2019*

1. **Project Description**

The primary objective of this lab is to increase understanding of data, addresses, memory contents, and strings. This lab should take a string input consisting of zero or more alphanumeric words and replace valid hexadecimal values with their decimal equivalent and reprint the string.

1. **Program Design**

At the beginning of the program we set $t7 to 0 as the max hex value. Then we $a0 to the input string from the user. Then we set the address of $s0 to the string. Then we begin the first while loop at label while1. Here we set $t3 to 1, to say that the characters are valid hexadecimals.

In label while2, we load the next character of the string into $t2 and we check for the null value in $t2, to check if the string is finished. If it is we exit to partB. Then we check if the character is a space or a period in order to signal the end of a word. If it passes this check then we determine if the value is not a valid hex and exit to label invalid if it’s not.

If it is a valid hex then we check if it’s a number or letter and jump to the corresponding label. Depending on the character type the value of $t8 is set to either 48 or 55.

Then we jump to while3 where we calculate the decimal value, increment the string and jump back to while2.

If we jump to the invalid label we set $t3 to 0, to say that it is not a valid hex value then we increment the string and jump back to while2.

In label while4 we check if $t3 is 0, to see if the value is not a valid hex if it isn’t a valid hex we branch to label notHex and if it is a valid hex then the program continues to label hex.

In label hex we print out the decimal value stored in $t4 and then branch to isPeriod if the stored character was a period in order to reprint the period. If it wasn’t a period then we jump over the isPeriod function. Then we print the space, and increment the string. Then we determine if the stored decimal in $t4 and determine if it’s a greater value than the stored value in $t7, which stores the greatest valid hex value as a decimal. If $t4 is greater than $t7 then we replace $t7 with the value in $t4. Then we reset $t4 and jump back to while1.

If the word was not a hexadecimal value then we arrive in label notHex. Here we load a ‘0’ into the string and print out the word. We also check for a period, and print one if it was a period then we increment the string and reset $t4 then we print a space in between the words. Then we jump back to while1.

In label partB, we print a ‘\n’ then determine if the string is sparse, intermediate, or plentiful based on the greatest valid hexadecimal value, and branch to the corresponding label.

Based on the label we load the address of the corresponding string into $a0 and jump to exit.

In exit we run syscall to print the loaded string from the previous step. Then we exit the program.

1. **Symbol Table**

**Table 1: Registers Used**

|  |  |
| --- | --- |
| Register | Use |
| $v0 | Tells Mars what to do when syscall is called. Used to print values |
| $a0 | Holds the output string of the program |
| $s0 | Holds the string and its address |
| $t0 | 1st pointer |
| $t1 | 2nd pointer |
| $t2 | Holds the next character of the string |
| $t3 | Boolean check for hexadecimal values |
| $t4 | Stores the decimal value of valid hex characters for printing |
| $t5 | Temporary storage |
| $t7 | Holds the max hexadecimal value for part B |
| $t8 | Holds the value with which we subtract from the hex to determine decimal value |

1. **Learning Coverage**

* Intermediate Operations in MIPS
* The usage of while loops
* The usage of labels
* If-statements equivalents
* Familiarizing myself with addresses, and string usage
* Determining hexadecimal values

1. **Prototype**

**5.1 Code in MARS MIPS**

# Max Saunders -- 10/25/19

# lab7

# Write a program to determine if a word in string is hexadecimal

# then print in decimal value

#===============================================================

#Registers used:

# $a0 - holds string and it's output

# $s0 - string address

# $t0 - 1st pointer

# $t1 - 2nd pointer

# $t2 - holds the characters

# $t3 - valid hex boolean

# $t4 - stores decimal value

# $t5 - temporary storage

# $t7 - max hex value for part b

.data

newLine: .asciiz "\n"

string: .space 300

space: .asciiz " "

sparse: .asciiz "Sparse"

inter: .asciiz "Intermediate"

plenti: .asciiz "Plentiful"

period: .byte '.'

.text

main:

li $t7, 0 #for max hex value

la $a0, string

li $a1, 300

li $v0, 8

syscall

la $s0, string

move $t0, $s0

while1:

li $t3, 1 #hex boolean

move $t1, $t0

while2:

lb $t2, 0($t0) #loads the next character

beqz $t2, partB #exits to partB if the next char is null

beq $t2, ' ', while4

beq $t2, '.', while4

bgt $t2, 'F', invalid

bgt $t2, '9', letter

##Determine type of char and calculate the decimal

num:

li $t8, 48

j while3

letter:

li $t8, 55

j while3

while3:

sll $t4, $t4, 4

add $t4, $t4, $t2

sub $t4, $t4, $t8

addi $t0, $t0, 1

j while2

invalid:

li $t3, 0 #checks the boolean to see if it's a hex

addi $t0, $t0, 1

j while2

while4:

beq $t3, 0, notHex

hex:

li $v0, 1

move $a0, $t4

syscall

beq $t2, '.', isPeriod

j jump1

isPeriod:

la $a0, period

li $v0, 4

syscall

jump1:

li $v0, 4

la $a0, space

syscall

addi $t0, $t0, 1

bge $t4, $t7, greatest

j reset

greatest:

move $t7, $t4 #sets new max value

reset:

li $t4, 0

j while1

notHex:

li $t8, 0

sb $t8, 0($t0)

move $a0, $t1

li $v0, 4

syscall

beq $t2, '.', isPeriod2

j jump2

isPeriod2:

la $a0, period

li $v0, 4

syscall

jump2:

addi $t0, $t0, 1

li $t4, 0

li $v0, 4

la $a0, space

syscall

j while1

partB:

la $a0, newLine

li $v0, 4

syscall

ble $t7, 255, isSparse

ble $t7, 65535, isInter

isPlentiful:

la $a0, plenti

j exit

isSparse:

la $a0, sparse

j exit

isInter:

la $a0, inter

j exit

exit:

syscall

li $v0, 10

syscall

1. **Test Plan and Results**

The test plan for this program is to use the three given test cases to check for accuracy.

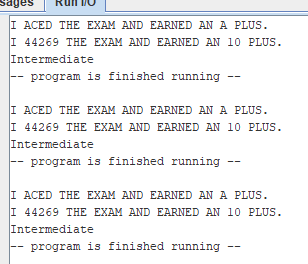
The three test case strings:

I ACED THE EXAM AND EARNED AN A PLUS.

BEEF BRISKET ON BED OF LETTUCE FOR 99 CENTS.

FADED OUT AND FAD3D IN.

**Conclusion:** The program successfully recognized the hexadecimal values in the strings and converted the strings from the input to the expected output.



1. **References**
   1. Lab Manual
      1. EEL4768-Lab7.pdf on CANVAS
   2. MARS Simulator
      1. The MARS simulator for MIPS processors