CS-35101

Thomas Moore

3/6/22

Summary:

For lab2 I implemented a single if else statement on lines 52 and 47 using the branch greater than or equal function. This allowed me to only have to use subtraction on line 44 and store the value to compare to w on line 45. Problem a on lab2 was able to be done with a simple swapping idiom between lines 29 and 31. Problem b took a bit more thinking and required arithmetic to swap without a third variable on lines 29 through 31 of lab2-b.asm. The core of the code of lab2-a and lab2-b are the same as lab2.asm as the only things that really needed to be changed were the inputs for the values of the registers and to add a swapping mechanism.

Conclusion:

The main problem I faced was in problem b where it required a swap to be done with no other registers available. It took some time and help from a tutor to figure out a way to mathematically do the same thing as swapping the variables, but it is indeed possible. This has taught me a valuable lesson for the future when I am coding in assembly due to the limited number of registers available. If I ever run into a situation where I need to swap register values with no available registers I am confident that I will be able to do so now.

lab2.asm

1. #Thomas Moore
2. .data
3. w: .asciiz "Enter value for w: "
4. x: .asciiz "Enter value for x: "
5. y: .asciiz "Enter value for y: "
6. z: .asciiz "Enter value for z: "
7. printx: .asciiz "The value of x is: "
9. .text
10. #input w
11. li $v0, 4
12. la $a0, w
13. syscall
14. li $v0, 5
15. syscall
16. add $t1, $0, $v0 #$t1 = w
18. #input x
19. li $v0, 4
20. la $a0, x
21. syscall
22. li $v0, 5
23. syscall
24. add $t2, $0, $v0 #$t2 = x
26. #input y
27. li $v0, 4
28. la $a0, y
29. syscall
30. li $v0, 5
31. syscall
32. add $t3, $0, $v0 #$t3 = y
34. #input z
35. li $v0, 4
36. la $a0, z
37. syscall
38. li $v0, 5
39. syscall
40. add $t4, $0, $v0 #$t4 = z
42. #subtraction and if statment
43. sub $t5, $t2, $t3 # x-y = $t5
44. bge $t5, $t1, if # if [(x-y)>= w] go to if:
46. else:
47. #set x to z
48. move $t2, $t4
49. j print
51. if:
52. #set x to y
53. move $t2, $t3
54. j print
56. print:
57. li $v0, 4
58. la $a0, printx
59. syscall
60. move $a0, $t2 #moving x to $a0
61. li $v0, 1
62. syscall
64. terminate:
65. li $v0, 10
66. Syscall

lab2-a.asm

1. #Thomas Moore
2. .data
3. r1: .asciiz "Enter value for register1: "
4. r2: .asciiz "Enter value for register2: "
5. printswap: .asciiz "\nThe values have been swapped "
6. printr1: .asciiz "\nThe value stored in r1 is: "
7. printr2: .asciiz "\nThe value stored in r2 is: "
9. .text
10. #input r1
11. li $v0, 4
12. la $a0, r1
13. syscall
14. li $v0, 5
15. syscall
16. add $s1, $0, $v0 #$s1 = r1
18. #input r2
19. li $v0, 4
20. la $a0, r2
21. syscall
22. li $v0, 5
23. syscall
24. add $s2, $0, $v0 #$s2 = r2
26. #swap r1 and r2
27. move $t0, $s2 #set $t0 to value of r2
28. move $s2, $s1 #set r2 to r1
29. move $s1, $t0 #set r1 to value of r2 via temp register
30. print:
31. li $v0, 4
32. la $a0, printswap
33. syscall
35. li $v0, 4
36. la $a0, printr1
37. syscall
38. move $a0, $s1 #moving r1 to $a0
39. li $v0, 1 #print int
40. syscall
41. li $v0, 4
42. la $a0, printr2
43. syscall
44. move $a0, $s2 #moving r2 to $a0
45. li $v0, 1
46. syscall
48. terminate:
49. li $v0, 10
50. Syscall

lab2-b.asm

1. #Thomas Moore
2. .data
3. r1: .asciiz "Enter value for register1: "
4. r2: .asciiz "Enter value for register2: "
5. printswap: .asciiz "\nThe values have been swapped "
6. printr1: .asciiz "\nThe value stored in r1 is: "
7. printr2: .asciiz "\nThe value stored in r2 is: "
9. .text
10. #input r1
11. li $v0, 4
12. la $a0, r1
13. syscall
14. li $v0, 5
15. syscall
16. add $s1, $0, $v0 #$s1 = r1
18. #input r2
19. li $v0, 4
20. la $a0, r2
21. syscall
22. li $v0, 5
23. syscall
24. add $s2, $0, $v0 #$s2 = r2
26. #swap via mathmatics
27. add $s1, $s1, $s2 # r1 = r1 + r2
28. sub $s2, $s1, $s2 # r2 = r1 - r2
29. sub $s1, $s1, $s2 # r1 = r1 - r2
31. print:
32. li $v0, 4
33. la $a0, printswap
34. syscall
36. li $v0, 4
37. la $a0, printr1
38. syscall
39. move $a0, $s1 #moving r1 to $a0
40. li $v0, 1 #print int
41. syscall
42. li $v0, 4
43. la $a0, printr2
44. syscall
45. move $a0, $s2 #moving r2 to $a0
46. li $v0, 1
47. syscall
49. terminate:
50. li $v0, 10
51. Syscall

Results:

lab2.asm

Table

Description automatically generated

lab2-a.asm

Text, letter

Description automatically generated

lab2-b.asm

Text, letter

Description automatically generated