Assignment A P1-2 Assembly Program Lab 2. Assign the values into the Array

GitHub Link to your file directly. (Please make it a link, not a text)

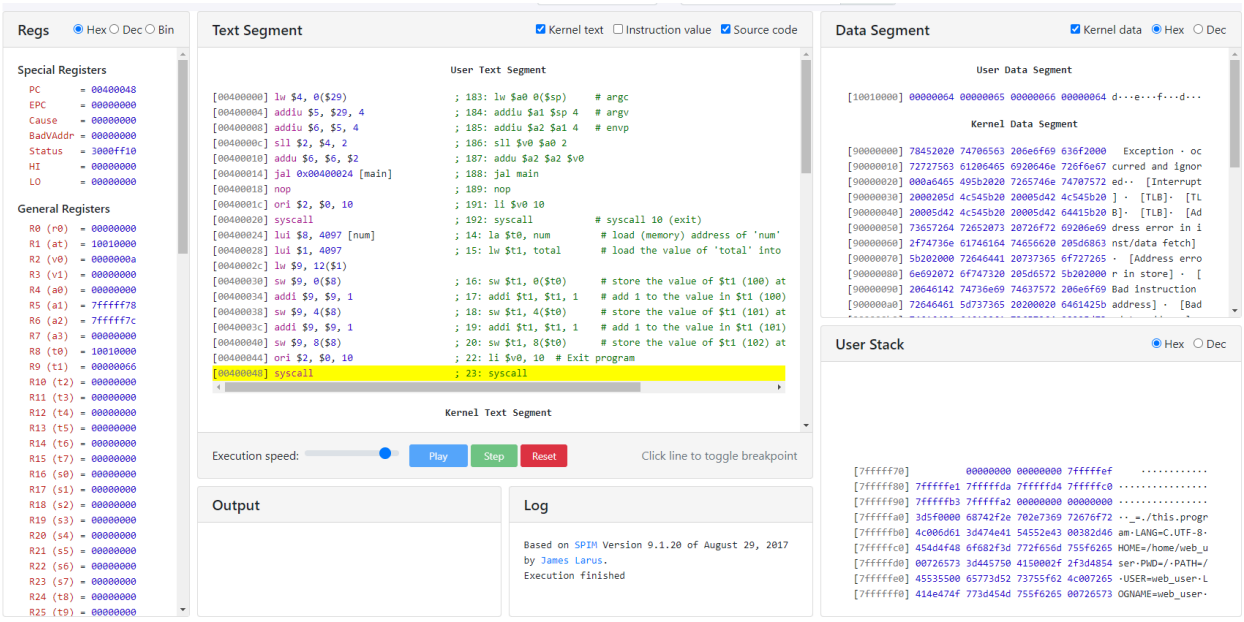
https://github.com/AVC-CS/p1a2-mtvonbargen/blob/0fbf5b58b7af34afefe81d8b46919a6f6999df84/P1A2.asm

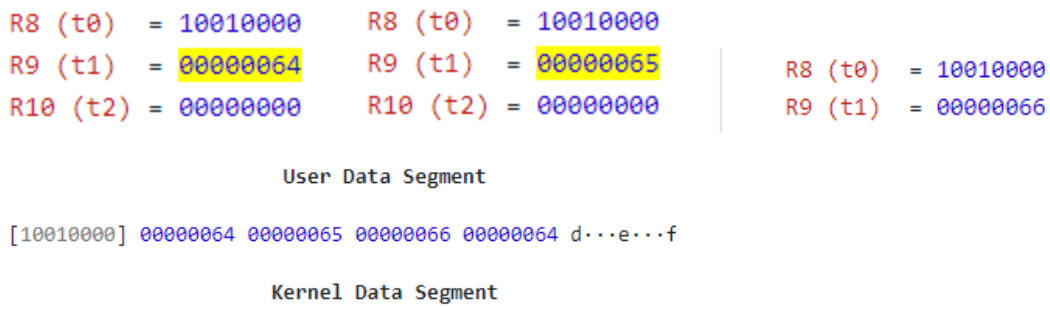
Elaborate on your program

```
################## Data segment ####################
.data
        num:    .space 12   # allocate 12 bytes of space for num
        total:  .word 100
################## Code segment ####################
.text
.globl main
main:
        la $t0, num         # load (memory) address of 'num' into $t0
        lw $t1, total       # load the value of 'total' into $t1
        sw $t1, 0($t0)      # store the value of $t1 (100) at offset 0 of 'num'
        addi $t1, $t1, 1    # add 1 to the value in $t1 (100), and save it in $t1 (new = 101)
        sw $t1, 4($t0)      # store the value of $t1 (101) at offset 4 of 'num'
        addi $t1, $t1, 1    # add 1 to the value in $t1 (101), and save it in $t1 (new = 102)
        sw $t1, 8($t0)      # store the value of $t1 (102) at offset 8 of 'num'

li $v0, 10  # Exit program
syscall
```

Program Result





Start of program ($t1) value is 100, which is 00000064

After one increment value is 101 00000065

End of program value is 102, which is 00000066

Offset can be seen in play under User Data Segment.

Register Value. What is the purpose of this register? How is it updated throughout your program?

$t0 = contains the memory address 0x1001000 of ‘num’

0x10010000 - 1st stored $t1 value location

0x10010004 - 2nd stored and incremented $t1 value location

0x10010008 - 3rd stored and incremented $t1 value location

The interaction between register and memory space

Memory Space - num: is used to set the array size, which is 12 bytes

Memory Space - total: is used to store the value ‘100’ (our starting value to increment)

$t0 is used to store the memory address of num

$t1 is used to store the value of total after every incrementation

Program logic

Store an initial value ‘total’ into the first position of a memory address/block and incrementally store sequentially increasing values in subsequent positions.