Lab 1 – ITITIU22240 – Đàm Nguyễn Trọng Lễ

Introduction to QtSpim

Procedures

P.1. Introduction to SPIM

A screenshot of a computer

AI-generated content may be incorrect.

P.2 Question and Tasks

3.

1. The code load in [00400004]
2. The one instruction is highlighted after every step is the next execution
3. the memory of the string is store on the Data window => User data segment [10000000]..[10040000] ,specifically [10010000]

A screenshot of a computer program

AI-generated content may be incorrect.4.

4.1:

A screenshot of a computer

AI-generated content may be incorrect.

A computer screen shot of a white screen

AI-generated content may be incorrect.

the output is: -148080296

the output is not correct

why: because the input is oversize the constrain of the data result in overflow value

4.2: Constraint = 1 word = 4 bytes = 2^31

The max and min input value is from -2,147,483,648 to 2,147,483,647

4.3:

A screenshot of a computer

AI-generated content may be incorrect.

4.4:

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

5.

A screen shot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

6.

    # -------------------- Data Declarations --------------------- #

.data

array:      .word   1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20

length:     .word   20

space:      .asciiz " "

    # ------------------- Text/Code section ---------------------- #

    # Basic approach: use Stack to reverse the array => First in, Last out

    # - loop to push each element onto the stack

    # - loop to pop each element off the stack

    # Final result is all elements reversed

.text

main:

    # ------------------------------------------------------------ #

    # Loop to read items from array and push to stack.

    la      $t0,    array                               # Load address A[0] of the array into $t0

    li      $t1,    0                                   # Init loop index, i = 0 into $t1

    lw      $t2,    length                              # Load length into $t2

    # $t0 = address of array[i]

    # $t1 = index i

    # $t2 = length of array

    # ------------------------------------------------------------ #

PushLoop:

    lw      $t4,    ($t0)                               # get array[i] into $t4

    addiu   $sp,    $sp,    -4                          # push array[i] onto stack | $sp = $sp - 4

    sw      $t4,    ($sp)                               # store/push array[i] into stack

    addi    $t1,    $t1,    1                           # i = i + 1 | increment loop index

    addi    $t0,    $t0,    4                           # update array address by 4 bytes = 1 word | ex: 0x1000 -> 0x1004

    blt     $t1,    $t2,    PushLoop                    # blt = branch if less than | if index i < length => continue to loop to line 34

    # ------------------------------------------------------------ #

    # Loop to pop items from stack and write into array.

    la      $t0,    array                               # array starting address

    li      $t1,    0                                   # reset the loop index, i=0

    lw      $t2,    length                              # length

    # ------------------------------------------------------------ #

PopLoop:

    lw      $t4,    ($sp)                               # get array[i] from stack into $t4

    addiu   $sp,    $sp,    4                           # increment stack pointer by 4 bytes | $sp = $sp + 4 | remove the top element from stack

    sw      $t4,    ($t0)                               # store array[i] from $t4 into array address $t0

    addi    $t1,    $t1,    1                           # i = i+1

    addi    $t0,    $t0,    4                           # update array address

    blt     $t1,    $t2,    PopLoop                     # if i<length, continue

    # ------------------------------------------------------------ #

    # re-init all registers

    la      $t0,    array

    li      $t1,    0

    lw      $t2,    length

    # ------------------------------------------------------------ #

LoopArray:

    lw      $a0,    ($t0)                               # Load the value at the address pointed to by $t0 into $t1

    # Print the value of Array[i]

    li      $v0,    1                                   # Set up syscall for print\_int

    syscall

    # Print a space

    li      $v0,    4                                   # Set up syscall for print\_string

    la      $a0,    space                               # Load address of space character

    syscall                                             # Print a space

    # Increment the array pointer and loop index

    addi    $t0,    $t0,    4                           # Increment the array pointer by 4 bytes

    addi    $t1,    $t1,    1                           # Increment the loop index by 1

    # Check if we have reached the end of the array

    blt     $t1,    $t2,    LoopArray                   # If i < length, continue looping

    # ------------------------------------------------------------ #

    li      $v0,    10                                  # terminate

    syscall

A screenshot of a computer

AI-generated content may be incorrect.

7.

.data

array:      .word   0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14                                               # array of integers

length:     .word   15                                                                                          # length of the array

prompt:     .asciiz "Enter your choice (1 or 2):\n"

choice1:    .asciiz "1. Print the value of the element from 1 to 14 \n"

choice2:    .asciiz "2. Print a sequence of values from the elements from i to j (0 <= i,j <= 14 || i != j) \n"

subprompt:  .asciiz "Enter the value of i: \n"

subprompt2: .asciiz "Enter the value of j: \n"

result:     .asciiz "The result is: \n"

error\_call:      .asciiz "Invalid choice. Terminate the program.\n"

space:      .asciiz " "

newline:    .asciiz "\n"

.text

main:

    # print out the prompt: "Enter an integer" --> line 6

    la      $a0,    prompt

    li      $v0,    4

    syscall

    # print out the prompt: "1. Print the value of the element from 1 to 14" --> line 7

    la      $a0,    choice1

    li      $v0,    4

    syscall

    # print out the prompt: "2. Print a sequence of values from the elements from i to j (0 <= i,j <= 14 || i != j)" --> line 8

    la      $a0,    choice2

    li      $v0,    4

    syscall

    # read in an integer

    li      $v0,    5                                       # input the integer from keyboard, user input it the keyboard, the number is stored into v0

    syscall                                                 # store the value from keyboard to v0 register

    move    $t0,    $v0                                     # copy the value of $v0 to $t0

    beq     $t0,    1,      Mode1

    beq     $t0,    2,      Mode2

    j      Error                                             # terminate the program if the user input is not 1 or 2

Mode1:

        # init the info into the registers for the looping in array

    la      $t1,    array                                   #A[0]

    li      $t2,    0                                       #index = 0

    lw      $t3,    length                                  #length = 14

    # print out the prompt: "The result is: " --> line 11

    la      $a0,    result

    li      $v0,    4

    syscall

    print\_loop:

        lw      $a0,    ($t1)                               # Load the value at the address pointed to by $t0 into $t1

        # Print the value of Array[i]

        li      $v0,    1                                   # Set up syscall for print\_int

        syscall

        # Print a space

        li      $v0,    4                                   # Set up syscall for print\_string

        la      $a0,    space                               # Load address of space character

        syscall                                             # Print a space

        # Increment the array pointer and loop index

        addi    $t1,    $t1,    4                           # Increment the array pointer by 4 bytes

        addi    $t2,    $t2,    1                           # Increment the loop index by 1

        # Check if we have reached the end of the array

        blt     $t2,    $t3,    print\_loop                  # If i < length, continue looping

    j       end                                             # terminate the program to prevent the program from going to Mode2

Mode2:

    # print out the prompt: "Enter the value of i: " --> line 9

    la      $a0,    subprompt

    li      $v0,    4

    syscall

    # read in an integer

    li      $v0,    5

    syscall

    move    $t1,    $v0                                     # copy the value of $v0 to $t1 => i

    # print out the prompt: "Enter the value of j: " --> line 10

    la      $a0,    subprompt2

    li      $v0,    4

    syscall

    # read in an integer

    li      $v0,    5

    syscall

    move    $t2,    $v0                                     # copy the value of $v0 to $t2 => j

    blt     $t1,    0,      Error                             # terminate the program if i < 0

    blt     $t2,    0,      Error                             # terminate the program if j < 0

    bge     $t1,    15,     Error                             # terminate the program if i >= 15

    bge     $t2,    15,     Error                             # terminate the program if j >= 15

    bgt     $t1,    $t2,    Error                             # terminate the program if i > j

    # Set up for printing range i to j

    la      $t3,    array              # Load array base address

    move    $t4,    $t1                # Initialize index with i

    sll     $t5,    $t4,    2          # Multiply index by 4

    add     $t3,    $t3,    $t5        # Add offset to array address

    print\_loop\_in\_range:

        # Print the value of Array[i]

        lw      $a0,    ($t3)                               # Load array element

        li      $v0,    1                                   # Set up syscall for print\_int

        syscall

        # Print a space

        li      $v0,    4                                   # Set up syscall for print\_string

        la      $a0,    space                               # Load address of space character

        syscall                                             # Print a space

        # Increment the array pointer and loop index

        addi    $t3,    $t3,    4                           # Increment the array pointer by 4 bytes

        addi    $t4,    $t4,    1                           # Increment the loop index by 1

        ble     $t4,    $t2,    print\_loop\_in\_range         # If index <= j, continue

    j       end                                             # terminate the program to prevent the program from going to another mode

Error:

    # print out the prompt: "Invalid choice. Terminate the program." --> line 12

    la      $a0,    error\_call

    li      $v0,    4

    syscall

end:

    # terminate the program

    li      $v0,    10

    syscall

Choice 1:

A screenshot of a computer

AI-generated content may be incorrect.

Choice 2:

A screenshot of a computer

AI-generated content may be incorrect.