

Zi Xuan Li

Professor Gertner

Csc 34200

25 March 2024

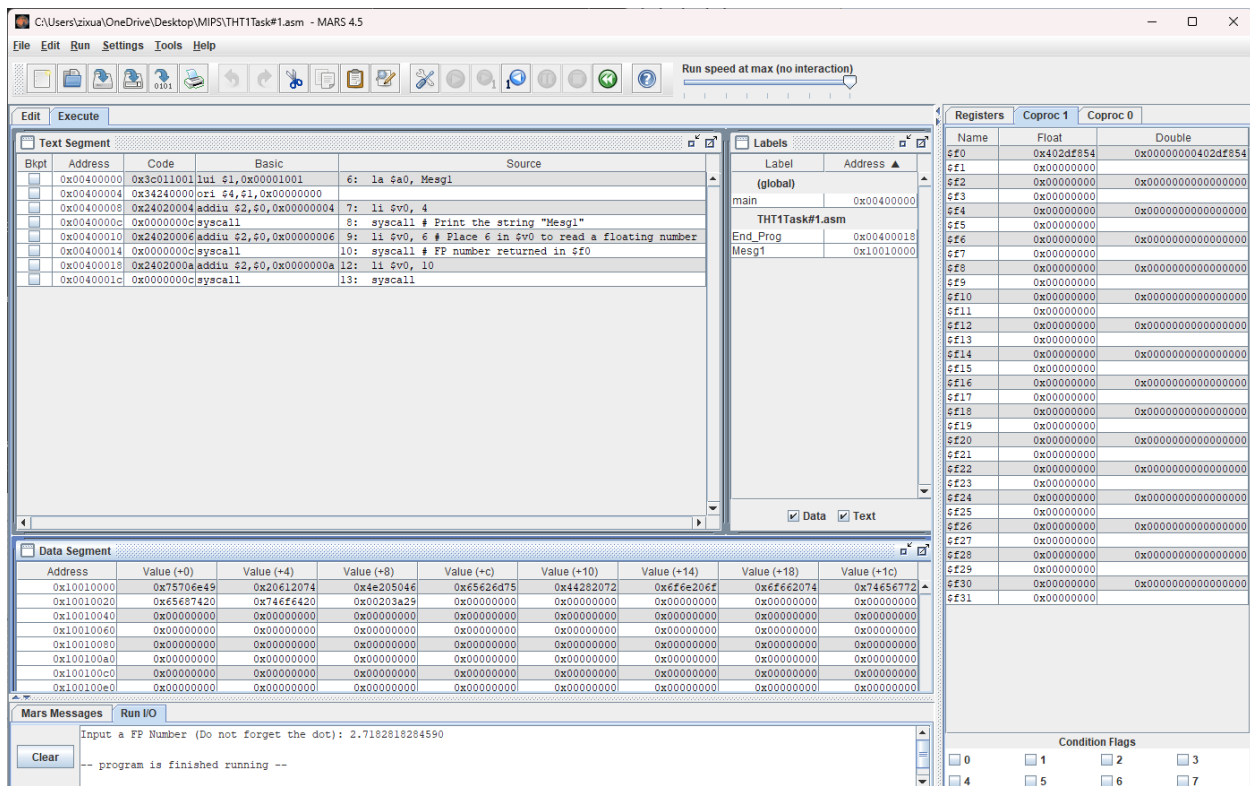
Take At Home Test

Task 1: Question #1

Input the value of $e = 2.7182818284590$ using SYSCALL (\$v0 = 6, which returns the value in \$f0).

Write the values:

- Hex Number in \$f0: 0x402df854
- Binary Value of \$f0: 01000000001011011111100001010100
- Sign: 0, Exponent: 1000 000, Mantissa: 0101 1011 1111 0000 1010 100
- Compute the decimal value stored in \$f0: 2.71828174591064453125

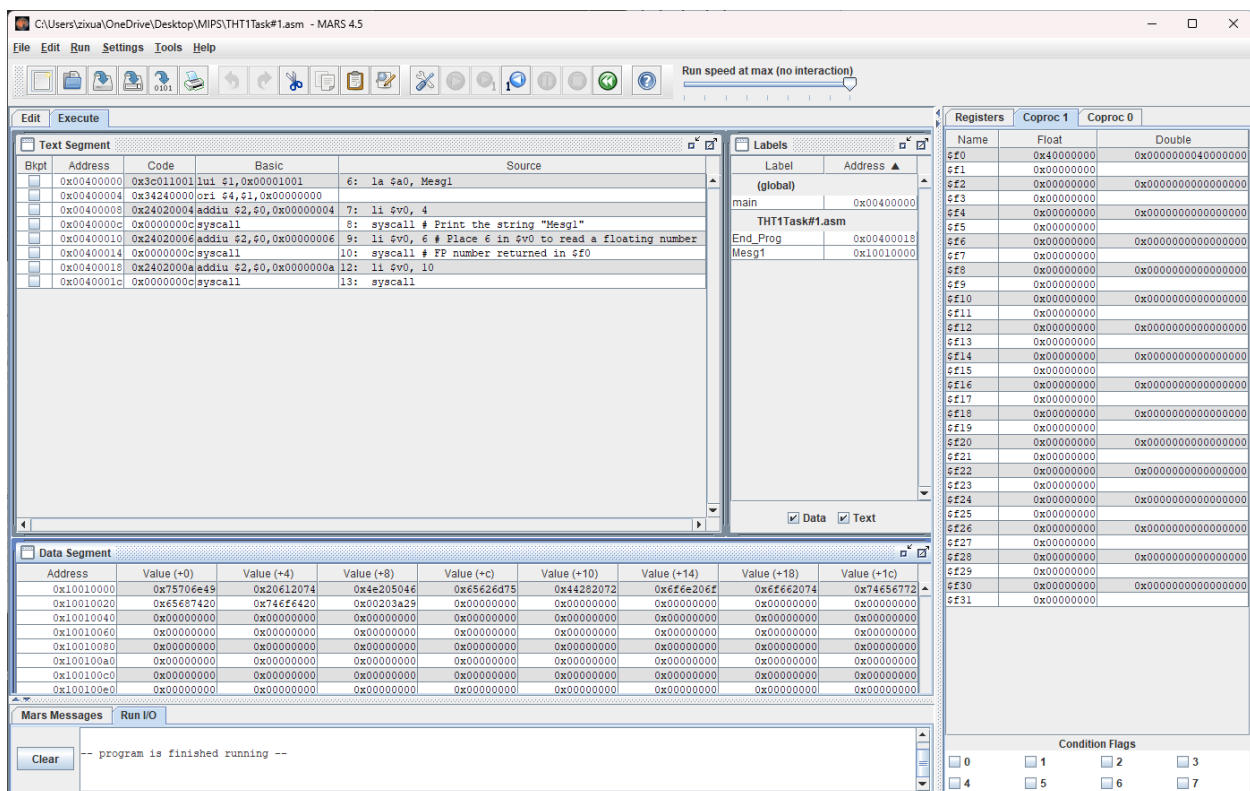


Task 1: Question #2

Input the value of $e = 2.0$ using SYSCALL (\$v0 = 6, which returns the value in \$f0).

Write the values:

- Hex Number in \$f0: 0x40000000
- Binary Value of \$f0: 01000000000000000000000000000000
- Sign: 0, Exponent: 1000 000, Mantissa: 0000 0000 0000 0000 0000 000
- Compute the decimal value stored in \$f0: 2



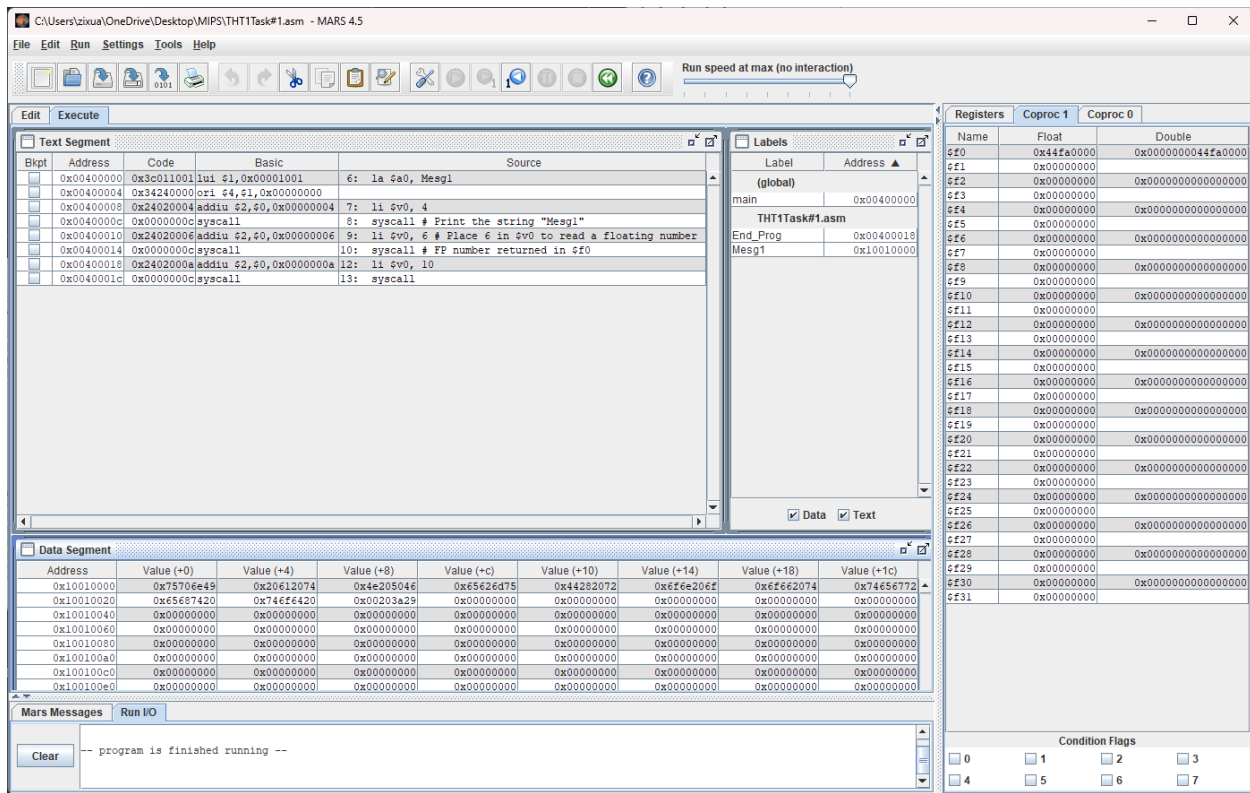
Task 1: Question #3

Input the value of $e = 2E3$ using SYSCALL (\$v0 = 6, which returns the value in \$f0).

Write the values:

- Hex Number in \$f0: 0x44fa0000
- Binary Value of \$f0: 01000100111110100000000000000000
- Sign: 0, Exponent: 1000 1001, Mantissa: 1111 0100 0000 0000 0000 000

- Compute the decimal value stored in \$f0: 2000

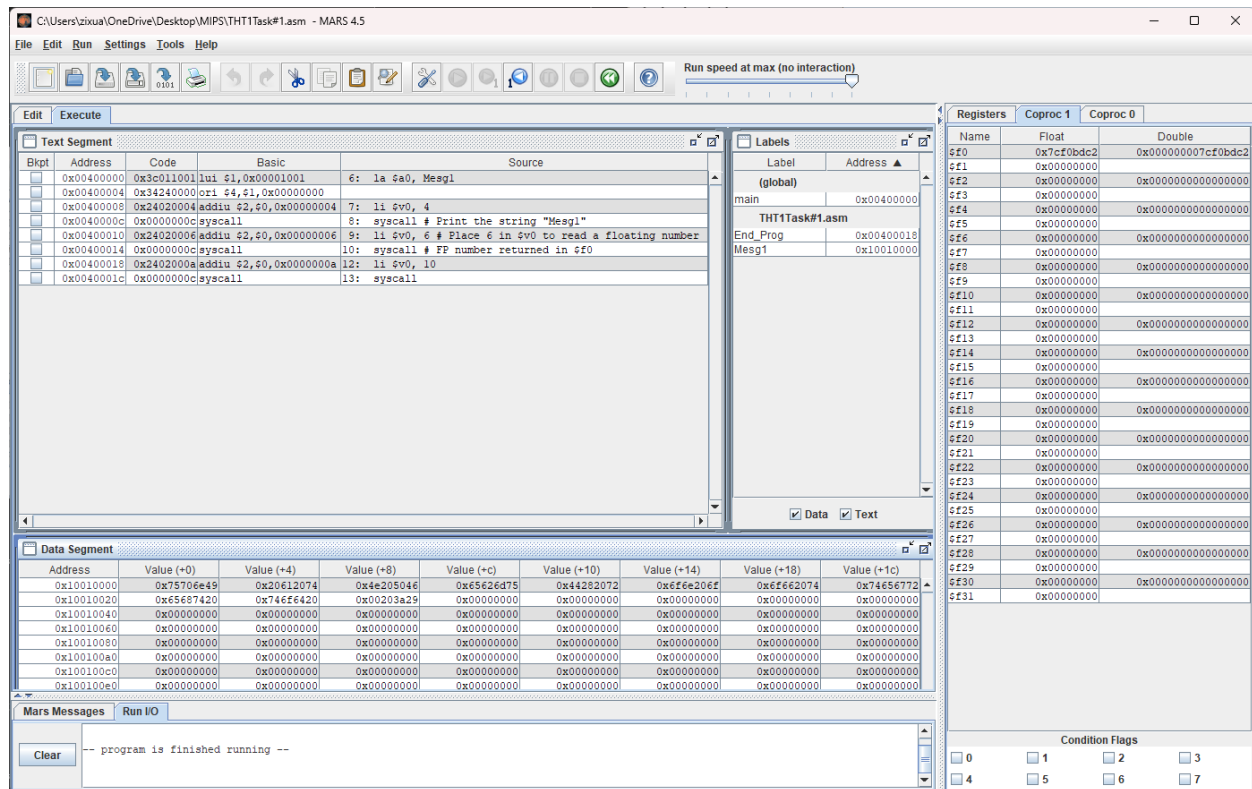


Task 1: Question #4

Input the value of $e = 1E37$ using SYSCALL (\$v0 = 6, which returns the value in \$f0).

Write the values:

- Hex Number in \$f0: 0x7cf0bdc2
- Binary Value of \$f0: 01111100111100001011110111000010
- Sign: 0, Exponent: 1111 1001, Mantissa: 1110 0001 0111 1011 1000 010
- Compute the decimal value stored in \$f0: $1 * 10^{37}$



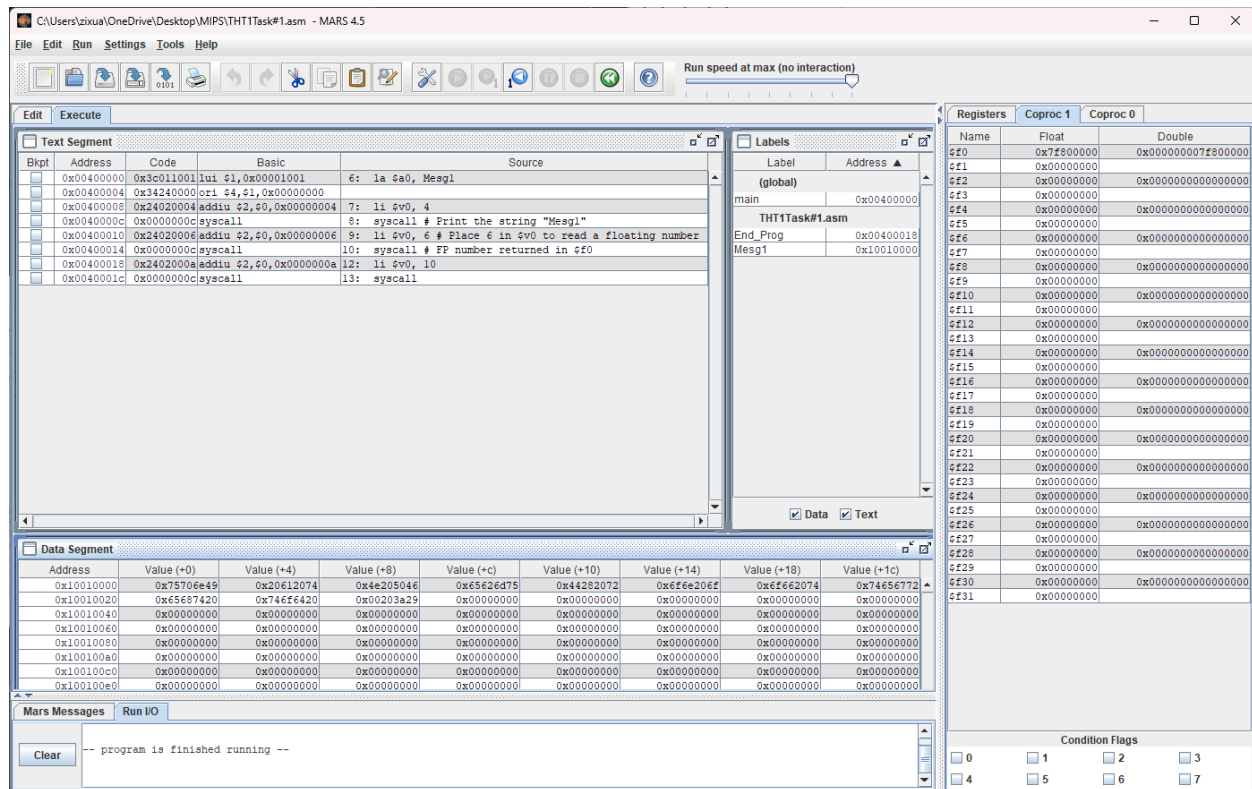
Task 1: Question #5

Input the value of $e = 1E39$ using SYSCALL (\$v0 = 6, which returns the value in \$f0).

Write the values:

- Hex Number in \$f0: 0x7f800000
- Binary Value of \$f0:

```
10111100000101000011111110100100111000100101000011101000000000000000
00000000000000000000000000000000000000000000000000000000000000000000
```
- Sign:, Exponent:, Mantissa: (Number is too large to represent)
- Compute the decimal value stored in \$f0: $1 * 10^{39}$

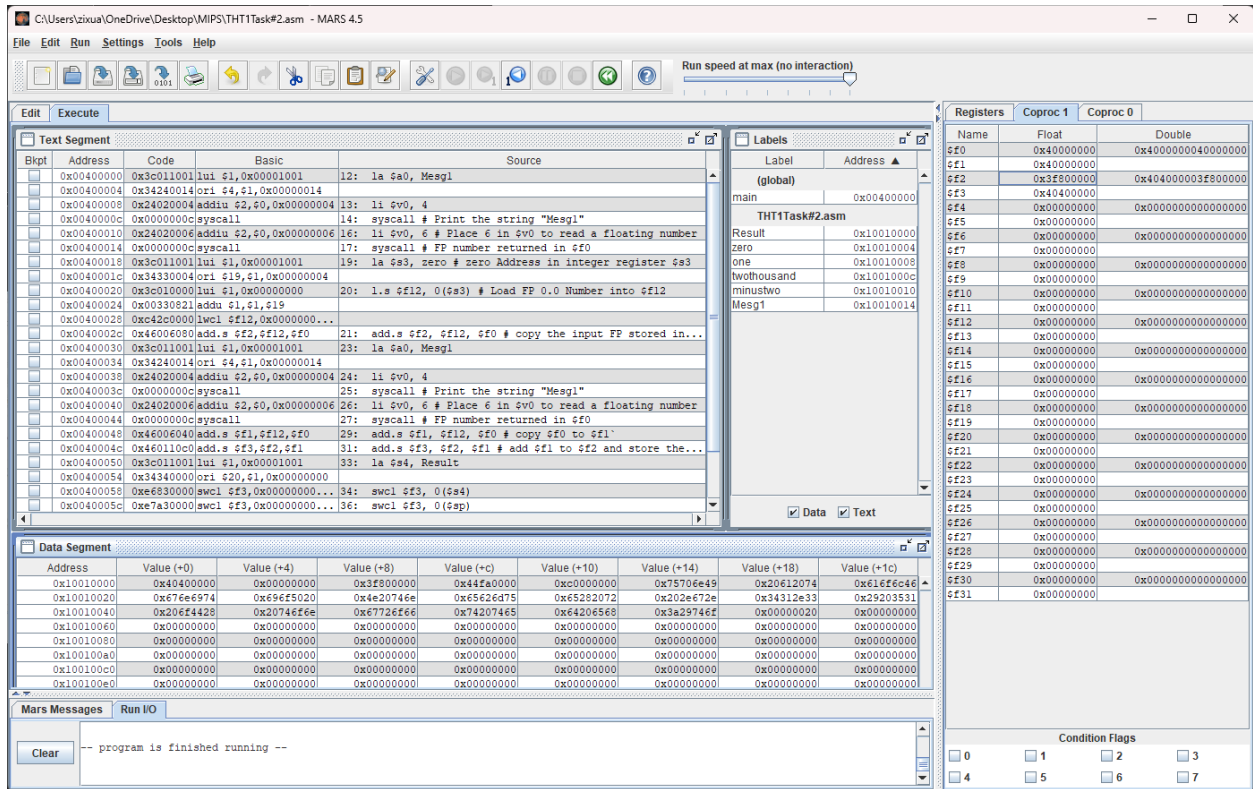


Task 2: Question #1

When running the program in MARS you inputted two floating point numbers in the INPUT WINDOW. The program computed the sum and stored it in register \$f3.

Write the values:

- Hex Number in \$f3: 0x40400000
- Binary Value of \$f3: 01000000010000000000000000000000
- Sign: 0, Exponent: 1000 0000, Mantissa: 1000 0000 0000 0000 0000 000
- Compute the decimal value stored in \$f3: 3
- Explain the result: The two floating point numbers I inputted were 1.0 and 2.0, the sum of these two floating points is 3.0 which is represented in hexadecimal in register \$f3.



Task 2: Question #2

Write address of variable Result in data segment where the sum in \$f3 is stored. Provide screenshots.

The sum of \$f3 is stored in the data segment address 0x10010000.

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0x40400000	0x00000000	0x3f800000	0x44fa0000	0xc0000000	0x75706e49	0x20612074	0x616f6c46
0x10010020	0x676e6974	0x696f5020	0x4e20746e	0x65626d75	0x65282072	0x202e672e	0x34312e33	0x29203531
0x10010040	0x206f4428	0x20746f6e	0x67726f66	0x74207465	0x64206568	0x3a29746f	0x00000020	0x00000000
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100e0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

Task 2: Question #3

Write address of Top of the Stack (Stack pointer) where the sum in \$f3 is stored. Provide screenshots of the stack.

The address of the top of the stack where the sum \$f3 is stored is 0x7ffeffc.

Registers	Coproc 1	Coproc 0
Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x10010000
\$v0	2	0x0000000a
\$v1	3	0x00000000
\$a0	4	0x10010014
\$a1	5	0x00000000
\$a2	6	0x00000000
\$a3	7	0x00000000
\$t0	8	0x00000000
\$t1	9	0x00000000
\$t2	10	0x00000000
\$t3	11	0x00000000
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x00000000
\$s1	17	0x00000000
\$s2	18	0x00000000
\$s3	19	0x10010004
\$s4	20	0x10010000
\$s5	21	0x00000000
\$s6	22	0x00000000
\$s7	23	0x00000000
\$t8	24	0x00000000
\$t9	25	0x00000000
\$k0	26	0x00000000
\$k1	27	0x00000000
\$gp	28	0x10008000
\$sp	29	0x7ffffc
\$fp	30	0x00000000
\$ra	31	0x00000000
pc		0x00400070
hi		0x00000000
lo		0x00000000

Task 2: Question #4

Write address on Stack at the offset +4 from stack pointer where the value \$f2 is stored. Display the value of \$f2 at this address. Provide screenshots of the stack.

The address on stack at the offset +4 from stack pointer where the value \$f2 is stored is 0x7ffffe0.

Data Segment								
Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x7ffffefe0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x40400000
0x7ffffef00	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x7ffffef20	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x7ffffef40	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

Task 2: Question #5

Write address on Stack at the offset +8 from stack pointer where the value \$f1 is stored. Display the value of \$f2 at this address. Provide screenshots of the stack.

The address on stack at the offset +8 from stack pointer where the value \$f1 is stored is 0x7ffff000.

Data Segment								
Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x7ffffefe0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x40400000
0x7ffffef00	0x3f800000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x7ffffef20	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x7ffffef40	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

Task 3: Question #1

Correct the code. Provide new, corrected listing and prove that it works.

The average of the numbers 200.86, -100.0, 300.0, -150.7258, 400.68562, -200.5686, 54.6789, 481.9827, -682.4012, 586.4512 is 89.096282. The result of running the program outputs “The average of the array is: 89.09628”.


```

.data
myArray: .float 200.86, -100.0, 300.0, -150.7258, 400.68562, -200.5686, 54.6789,
481.9827, -682.4012, 586.4512
Result: .asciiz "The average of the array is: "
.text
.globl main
main:
    la $s0, myArray

    li $t1, 0

    li $t2, 10

    li $t4, 0x41200000    # 10.0 in IEEE 754 single precision (hex representation)

    mtc1 $t4, $f13        # Move 10.0 to $f13

    li $t4, 0             # 0.0 in integer register
    mtc1 $t4, $f12        # Move 0.0 to $f12

loop:
    sll $t3, $t1, 2

    add $s1, $s0, $t3

    l.s $f2, 0($s1)

    add.s $f12, $f12, $f2

    addi $t1, $t1, 1

    bne $t1, $t2, loop

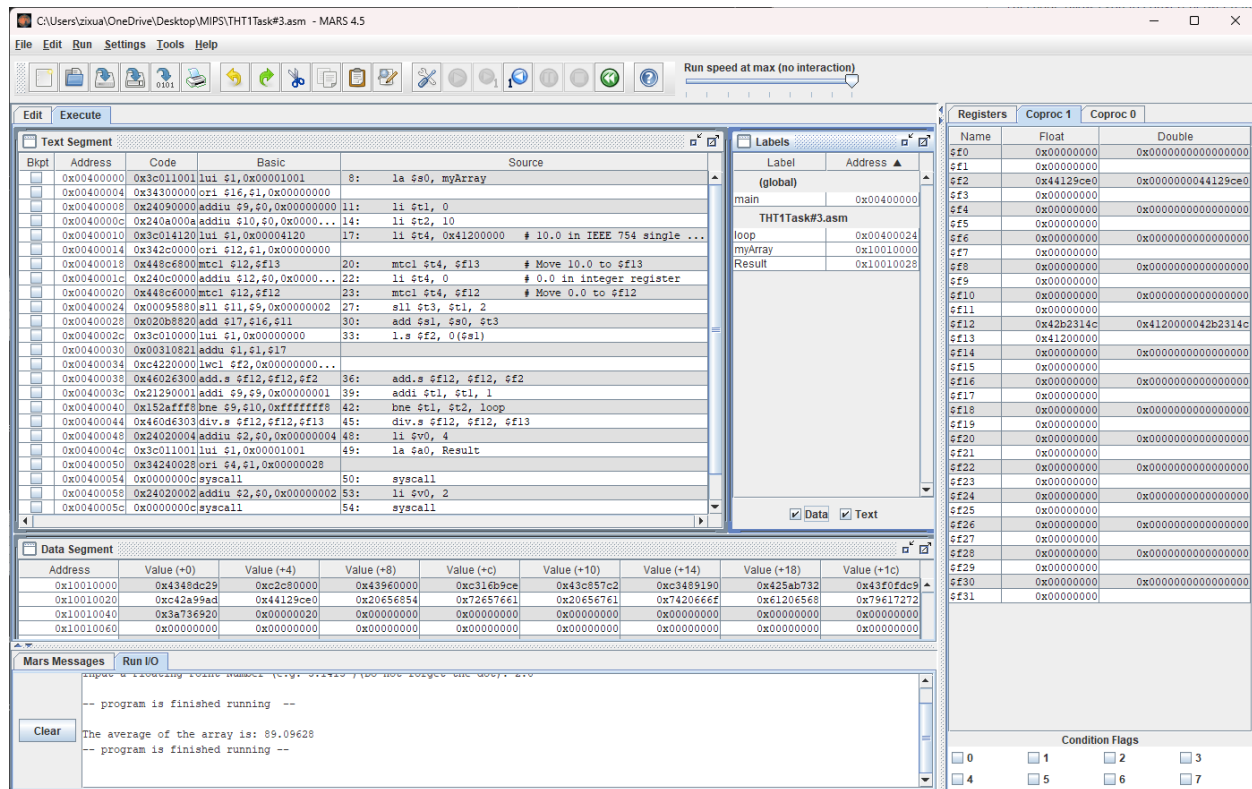
    div.s $f12, $f12, $f13

    li $v0, 4
    la $a0, Result
    syscall

    li $v0, 2
    syscall

    li $v0, 10
    syscall

```



Task 3: Question #2

Provide the address of the variable result and show its value.

The address of the variable result is 0x10010020 and the value is 0x20656854.

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0x4348dc29	0xc2c80000	0x43960000	0xc316b9ce	0x43c857c2	0xc3489190	0x425ab732	0x43f0fdc9
0x10010020	0xc42a99ad	0x4129ce0	0x20656854	0x72657661	0x20656761	0x7420666f	0x61206568	0x79617272
0x10010040	0x3a736920	0x00000020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

Task 4: Question #1

After correction, verify that your program works correctly. Compare your results with the temperature table. You must show screenshots with the results.

The program works correctly because when I type in 32, 212, 98.6, 122 the program outputs 0, 100, 37, and 50 respectively.

```

.data
Input: .asciiz "\nPlease Enter the Temperature in Fahrenheit: "
Output: .asciiz "\nThe Temperature in Celsius is: "
.text
.globl main

main:
    # Print Input String
    la $a0, Input
    li $v0, 4
    syscall

    # Read the temperature in Fahrenheit into $f0
    li $v0, 6
    syscall
    mov.s $f12, $f0

    # Call the procedure T_Convert with the input in $f12
    jal T_Convert

    # Print Output String
    li $v0, 4
    la $a0, Output
    syscall

    # Print the result contained in $f1
    li $v0, 2
    mov.s $f12, $f1
    syscall

    # Exit
    li $v0, 10
    syscall

T_Convert: # Procedure T_Convert
    li $t4, 0x3fe66666
    mtc1 $t4, $f5

    li $t4, 0x42000000
    mtc1 $t4, $f6

    sub.s $f1, $f12, $f6 # $f1 = Fahrenheit($f12) - 32
    div.s $f1, $f1, $f5 # Celsius ($f1) = (Fahrenheit - 32) / 1.8
    jr $ra # Return to caller

```

CAUsers\zixua\OneDrive\Desktop\WIPs\THT1Task#4.asm - MARS 4.5

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Registers Coproc 1 Coproc 0

Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x10010000
\$v0	2	0x0000000a
\$v1	3	0x00000000
\$a0	4	0x1001002e
\$a1	5	0x00000000
\$a2	6	0x00000000
\$a3	7	0x00000000
\$t0	8	0x00000000
\$t1	9	0x00000000
\$t2	10	0x00000000
\$t3	11	0x00000000
\$t4	12	0x42000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x00000000
\$s1	17	0x00000000
\$s2	18	0x00000000
\$s3	19	0x00000000
\$s4	20	0x00000000
\$s5	21	0x00000000
\$s6	22	0x00000000
\$s7	23	0x00000000
\$s8	24	0x00000000
\$s9	25	0x00000000
\$k0	26	0x00000000
\$k1	27	0x00000000
\$gp	28	0x10008000
\$sp	29	0x7fffffc0
\$fp	30	0x00000000
\$ra	31	0x00400020
PC		0x00400044
hi		0x00000000
lo		0x00000000

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400000	0x3c011001	lui \$1,0x000...	9: la \$a0, Input
	0x00400004	0x34240000	ori \$4,\$1,0x...	
	0x00400008	0x24020004	addiu \$2,\$0,...	10: li \$v0, 4
	0x0040000c	0x0000000c	syscall	11: syscall
	0x00400010	0x24020006	addiu \$2,\$0,...	14: li \$v0, 6
	0x00400014	0x0000000c	syscall	15: syscall
	0x00400018	0x46000036	mov.s \$f12,\$f0	16: mov.s \$f12, \$f0
	0x0040001c	0xc1000111	jnl 0x00400044	19: jnl T_Convert
	0x00400020	0x24020004	addiu \$2,\$0,...	22: li \$v0, 4
	0x00400024	0x3c011001	lui \$1,0x000...	23: la \$a0, Output
	0x00400028	0x3424002e	ori \$4,\$1,0x...	
	0x0040002c	0x0000000c	syscall	24: syscall
	0x00400030	0x24020002	addiu \$2,\$0,...	27: li \$v0, 2
	0x00400034	0x46000036	mov.s \$f12,\$f1	28: mov.s \$f12, \$f1

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0x656c500a	0x20657361	0xe5746e45	0x69742072	0x65542065	0x7265706d	0x72757461	0x6e692065
0x10010020	0x69614620	0x696e6572	0x3a746965	0x540a0020	0x54206568	0x65706d65	0x75746172	0x69206572
0x10010040	0x6543206e	0x7569736c	0x73692073	0x0000003a	0x00000000	0x00000000	0x00000000	0x00000000
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100e0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010100	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010120	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

Mars Messages Run IO

Please Enter the Temperature in Fahrenheit: 32

The Temperature in Celsius is: 0.0

-- program is finished running --

Reset: reset completed.

Clear

Please Enter the Temperature in Fahrenheit: 212

The Temperature in Celsius is: 100.0

-- program is finished running --

Reset: reset completed.

Please Enter the Temperature in Fahrenheit: 32

The Temperature in Celsius is: 0.0

-- program is finished running --

Reset: reset completed.

Please Enter the Temperature in Fahrenheit: 212

The Temperature in Celsius is: 100.0

-- program is finished running --

```
Please Enter the Temperature in Fahrenheit: 98.6
```

```
The Temperature in Celsius is: 37.0
```

```
-- program is finished running --
```

```
Reset: reset completed.
```

```
Please Enter the Temperature in Fahrenheit: 122
```

```
The Temperature in Celsius is: 50.0
```

```
-- program is finished running --
```

Task 4: Question #2

Temperature in Farenheit	Temperature in Celsius
32	0.0000
212	100.000
98.6	37.0000
122	50.0000

Write a function that prints the table as shown above. Print just temperatures, no need to draw the table.

The following code below prints the table as shown above printing just the temperatures without drawing the table.

```
Temperature in Fahrenheit Temperature in Celsius
32.0                      0.0
212.0                    100.0
98.6                     37.0
122.0                    50.0

-- program is finished running --
```

```
.data
Input:  .asciiz "Temperature in Fahrenheit "
Input2: .asciiz "Temperature in Celsius\n"
Output: .asciiz "                "
Output2: .asciiz "\n"
zero:   .float 0.0
onepointeight:.float 1.8
thirtytwo:.float 32.0
myArray: .float 32.0, 212.0, 98.6, 122.0
```

```
.text
.globl main
```

```
main:
```

```
    # Print initial messages
```

```
    li $v0, 4
```

```
    la $a0, Input
```

```
    syscall
```

```
    li $v0, 4
```

```
    la $a0, Input2
```

```
    syscall
```

```
    # Load constants
```

```
    l.s $f5, onepointeight
```

```
    l.s $f6, thirtytwo
```

```
    li $t1, 0
```

```
loop:
```

```
    # Load Fahrenheit from array
```

```
    l.s $f2, myArray($t1)
```

```
    # Convert Fahrenheit to Celsius
```

```
    sub.s $f1, $f2, $f6
```

```
    div.s $f1, $f1, $f5
```

```
    # Print Fahrenheit value
```

```
    li $v0, 2
```

```
    mov.s $f12, $f2
```

```
    syscall
```

```
    la $a0, Output
```

```
    li $v0, 4
```

```
    syscall
```

```
    # Print Celsius value
```

```
li $v0, 2
mov.s $f12, $f1
syscall
la $a0, Output2
li $v0, 4
syscall

addi $t1, $t1, 4
blt $t1, 16, loop
```

End_Prog:

```
li $v0, 10
syscall
```