PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS

Instituto de Ciências Exatas e Informática – ICEI

Departamento de Ciência da Computação

TRABALHO PRÁTICO 06

Belo Horizonte

2022

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS

Ciência da Computação

Hernane Velozo Rosa
hernane.rosa@sga.pucminas.br

Este trabalho apresenta os códigos dos exercícios práticos propostos.

Professor: Dr. Romanelli Lodron Zuim
Disciplina: Arquitetura de Computadores II

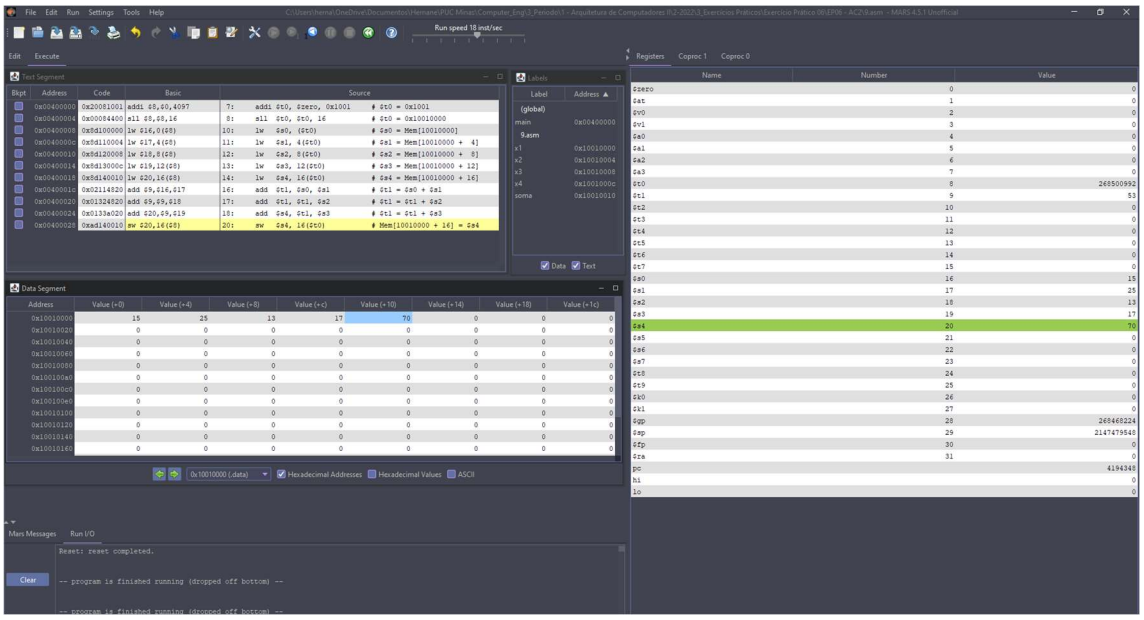
Palavras chaves: Instruções, ULA, programação de baixo-nível e MIPS.

RELATÓRIO: CÓDIGOS E EXECUÇÃO

Programa 09 – Código

```
# Programa 09 - By Hernane V | Data: 04/12/2022
# Arquitetura de Computadores II - Romanelli
.text
.globl main

main:
    addi $t0, $zero, 0x1001      # $t0 = 0x1001
    sll  $t0, $t0, 16            # $t0 = 0x10010000

    lw   $s0, ($t0)              # $s0 = Mem[10010000]
    lw   $s1, 4($t0)             # $s1 = Mem[10010000 +  4]
    lw   $s2, 8($t0)             # $s2 = Mem[10010000 +  8]
    lw   $s3, 12($t0)            # $s3 = Mem[10010000 + 12]
    lw   $s4, 16($t0)            # $s4 = Mem[10010000 + 16]

    add  $t1, $s0, $s1           # $t1 = $s0 + $s1
    add  $t1, $t1, $s2           # $t1 = $t1 + $s2
    add  $s4, $t1, $s3           # $t1 = $t1 + $s3

    sw   $s4, 16($t0)            # Mem[10010000 + 16] = $s4

.data
    x1:     .word 15
    x2:     .word 25
    x3:     .word 13
    x4:     .word 17
    soma:   .word -1
```

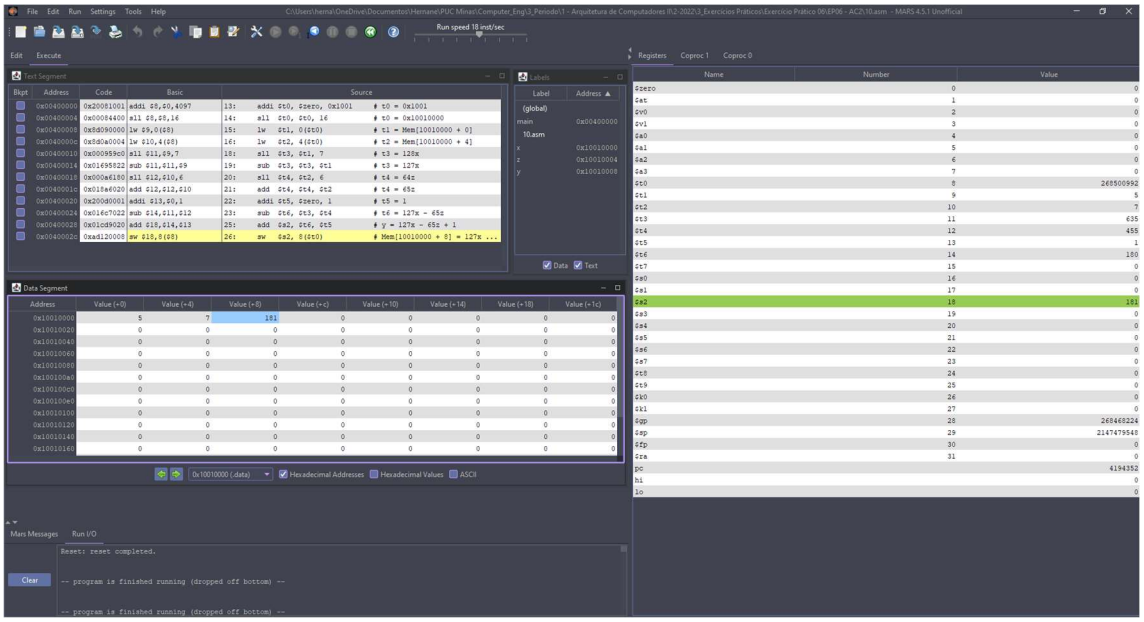
Programa 09 – Execução via MARS



Programa 10 – Código

```
# Programa 10 - By Hernane V | Data: 04/12/2022
# Arquitetura de Computadores II - Romanelli

    # y = 127x – 65z + 1
    # x ? $s0
    # y ? $s1
    # z ? $s2

.text
.globl main

main:
    addi $t0, $zero, 0x1001      # t0 = 0x1001
    sll  $t0, $t0, 16            # t0 = 0x10010000
    lw   $t1, 0($t0)             # t1 = Mem[10010000 + 0]
    lw   $t2, 4($t0)             # t2 = Mem[10010000 + 4]

    sll  $t3, $t1, 7             # t3 = 128x
    sub  $t3, $t3, $t1           # t3 = 127x
    sll  $t4, $t2, 6             # t4 = 64z
    add  $t4, $t4, $t2           # t4 = 65z
    addi $t5, $zero, 1           # t5 = 1
    sub  $t6, $t3, $t4           # t6 = 127x - 65z

    add  $s2, $t6, $t5           # y = 127x - 65z + 1
    sw   $s2, 8($t0)             # Mem[10010000 + 8] = 127x - 65z + 1

.data
    x: .word 5
    z: .word 7
    y: .word 0
```

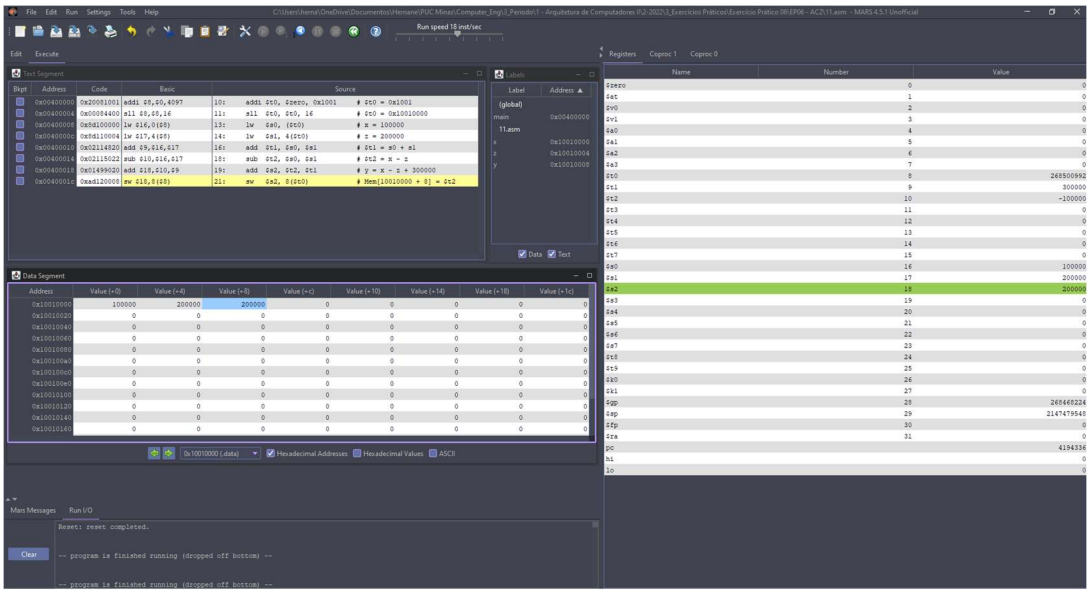
Programa 10 – Execução via MARS



Programa 11 – Código

```
# Programa 11 - By Hernane V | Data: 04/12/2022
# Arquitetura de Computadores II - Romanelli

    # y = x-z+300000

.text
.globl main

main:
    addi $t0, $zero, 0x1001      # $t0 = 0x1001
    sll  $t0, $t0, 16            # $t0 = 0x10010000

    lw   $s0, ($t0)              # x = 100000
    lw   $s1, 4($t0)             # z = 200000

    add  $t1, $s0, $s1           # $t1 = s0 + s1

    sub  $t2, $s0, $s1           # $t2 = x - z
    add  $s2, $t2, $t1           # y = x - z + 300000

    sw   $s2, 8($t0)             # Mem[10010000 + 8] = $t2

.data
    x: .word 100000
    z: .word 200000
    y: .word 0                   # valor a ser sobrescrito
```

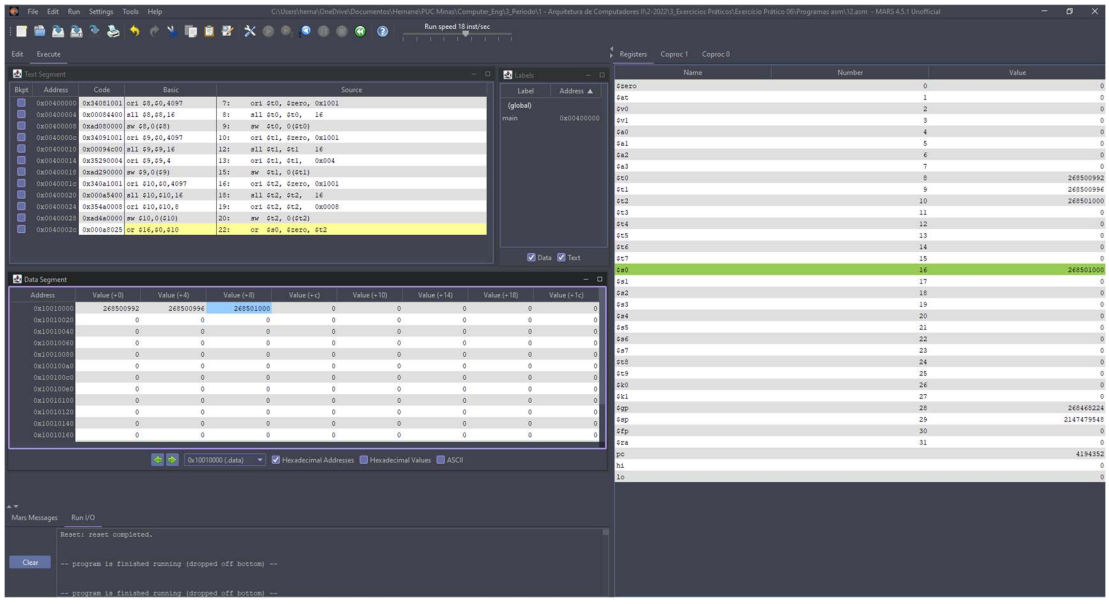
Programa 11 – Execução via MARS



Programa 12 – Código

```
# Programa 12 - By Hernane V | Data: 04/12/2022
# Arquitetura de Computadores II - Romanelli
.text
.globl main

main:
    ori $t0, $zero, 0x1001
    sll $t0, $t0,   16
    sw  $t0, 0($t0)
    ori $t1, $zero, 0x1001

    sll $t1, $t1    16
    ori $t1, $t1,   0x004

    sw  $t1, 0($t1)
    ori $t2, $zero, 0x1001

    sll $t2, $t2,   16
    ori $t2, $t2,   0x0008
    sw  $t2, 0($t2)

    or  $s0, $zero, $t2
```

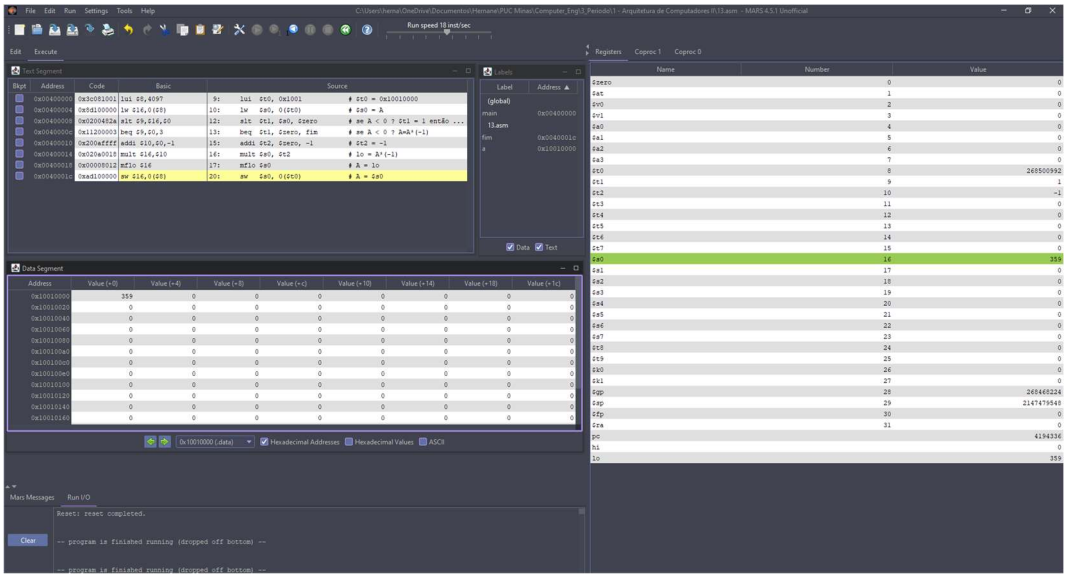
Programa 12 – Execução via MARS



Programa 13 – Código

```
# Programa 13 - By Hernane V | Data: 04/12/2022
# Arquitetura de Computadores II - Romanelli
.text
.globl main

        # A → $s0

main:
    lui  $t0, 0x1001            # $t0 = 0x10010000
    lw   $s0, 0($t0)            # $s0 = A

    slt  $t1, $s0, $zero        # se A < 0 → $t1 = 1 então $t1 = 0
    beq  $t1, $zero, fim        # se A < 0 → A=A*(-1)

    addi $t2, $zero, -1         # $t2 = -1
    mult $s0, $t2               # lo = A*(-1)
    mflo $s0                    # A = lo

fim:
    sw   $s0, 0($t0)            # A = $s0

.data
    a: .word -359               # 0x10010000
```

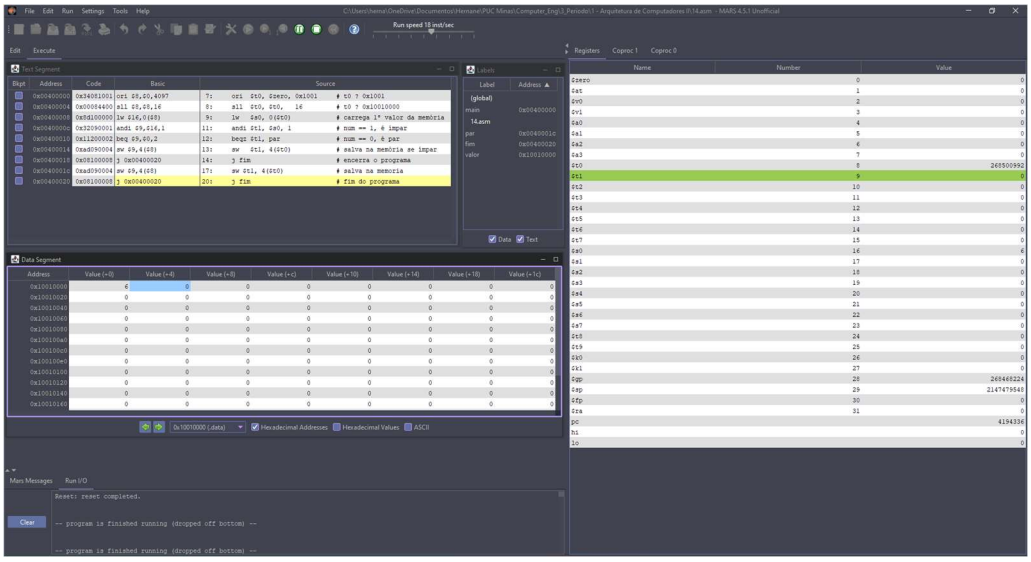
Programa 13 – Execução via MARS



Programa 14 – Código

```
# Programa 14 - By Hernane V | Data: 04/12/2022
# Arquitetura de Computadores II - Romanelli
.text
.globl main

main:
    ori  $t0, $zero, 0x1001      # t0 → 0x1001
    sll  $t0, $t0,   16          # t0 → 0x10010000
    lw   $s0, 0($t0)             # carrega 1º valor da memória

    andi $t1, $s0, 1             # num == 1, é ímpar
    beqz $t1, par                # num == 0, é par
    sw   $t1, 4($t0)             # salva na memória se ímpar
    j fim                        # encerra o programa

par:
    sw $t1, 4($t0)               # salva na memoria

fim:
    j fim                        # fim do programa

.data
    valor: .word 6
```

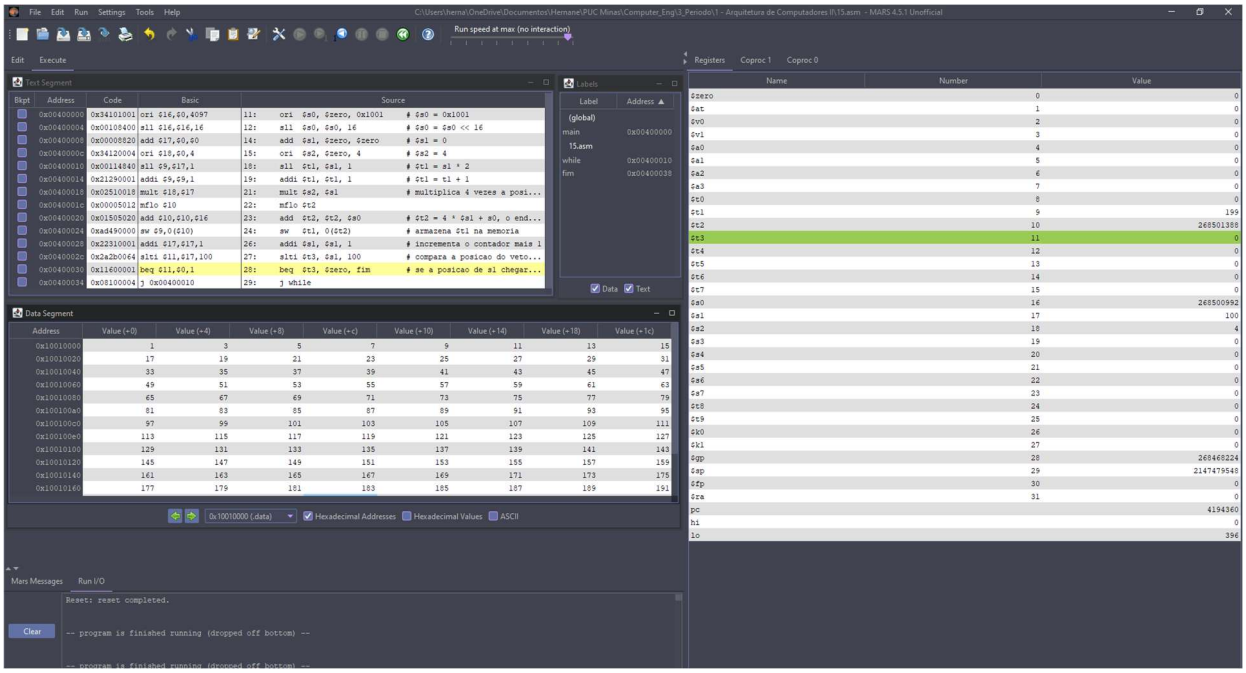
Programa 14 – Execução via MARS



Programa 15 – Código e quantidade de Instruções conforme o tipo

```
# Programa 15 - By Hernane V | Data: 04/12/2022
# Arquitetura de Computadores II - Romanelli
.text
.globl main

    # $s0 → V[0]
    # $s1 → i
    # $s2 → valor fixo de 4

main:
    ori  $s0, $zero, 0x1001     # $s0 = 0x1001
    sll  $s0, $s0, 16           # $s0 = $s0 << 16

    add  $s1, $zero, $zero      # $s1 = 0
    ori  $s2, $zero, 4          # $s2 = 4

while:
    sll  $t1, $s1, 1            # $t1 = s1 * 2
    addi $t1, $t1, 1            # $t1 = t1 + 1

    mult $s2, $s1               # multiplica 4 vezes a posição no vetor
    mflo $t2
    add  $t2, $t2, $s0          # $t2 = 4 * $s1 + s0, o endereco do elemento na memoria
    sw   $t1, 0($t2)            # armazena $t1 na memoria

    addi $s1, $s1, 1            # incrementa o contador mais 1
    slti $t3, $s1, 100          # compara a posicao do vetor com a ultima posição
    beq  $t3, $zero, fim        # se a posicao de s1 chegar a 100, encerra
    j while
fim:

.data
```

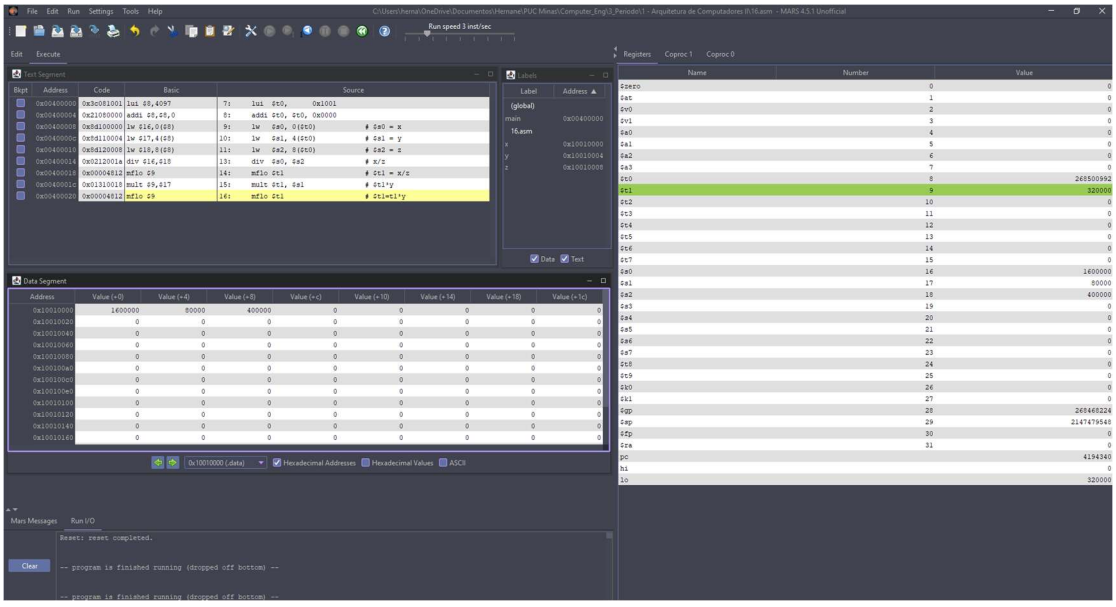
Programa 15 – Execução via MARS



Programa 16 – Código

```
# Programa 16 - By Hernane V | Data: 04/12/2022
# Arquitetura de Computadores II - Romanelli
.text
.globl main

main:
    lui  $t0,      0x1001
    addi $t0, $t0, 0x0000
    lw   $s0, 0($t0)            # $s0 = x
    lw   $s1, 4($t0)            # $s1 = y
    lw   $s2, 8($t0)            # $s2 = z

    div  $s0, $s2               # x/z
    mflo $t1                    # $t1 = x/z
    mult $t1, $s1               # $t1*y
    mflo $t1                    # $t1=t1*y

.data
    x: .word 0x186A00           # 1600000
    y: .word 0x13880            # 80000
    z: .word 0x61A80            # 400000
```

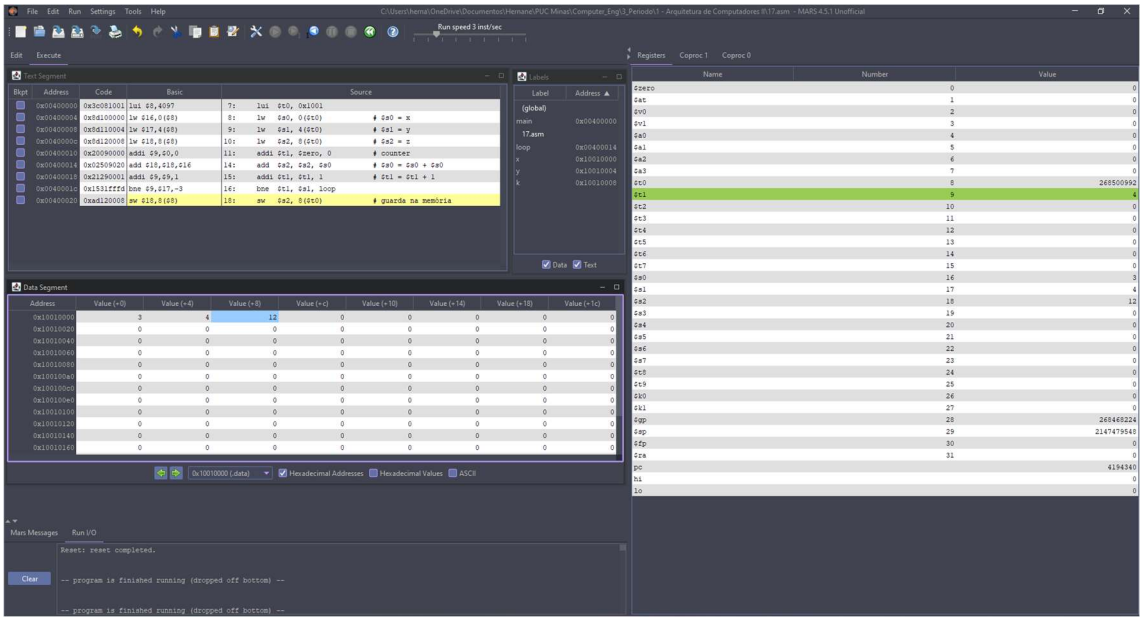
Programa 16 – Execução via MARS



Programa 17 – Código

```
# Programa 17 - By Hernane V | Data: 04/12/2022
# Arquitetura de Computadores II - Romanelli
.text
.globl main

main:
    lui  $t0, 0x1001
    lw   $s0, 0($t0)            # $s0 = x
    lw   $s1, 4($t0)            # $s1 = y
    lw   $s2, 8($t0)            # $s2 = z
    addi $t1, $zero, 0          # counter

loop:
    add  $s2, $s2, $s0          # $s0 = $s0 + $s0
    addi $t1, $t1, 1            # $t1 = $t1 + 1
    bne  $t1, $s1, loop

    sw   $s2, 8($t0)            # guarda na memória

.data
    x: .word 3
    y: .word 4
    k: .word 0
```

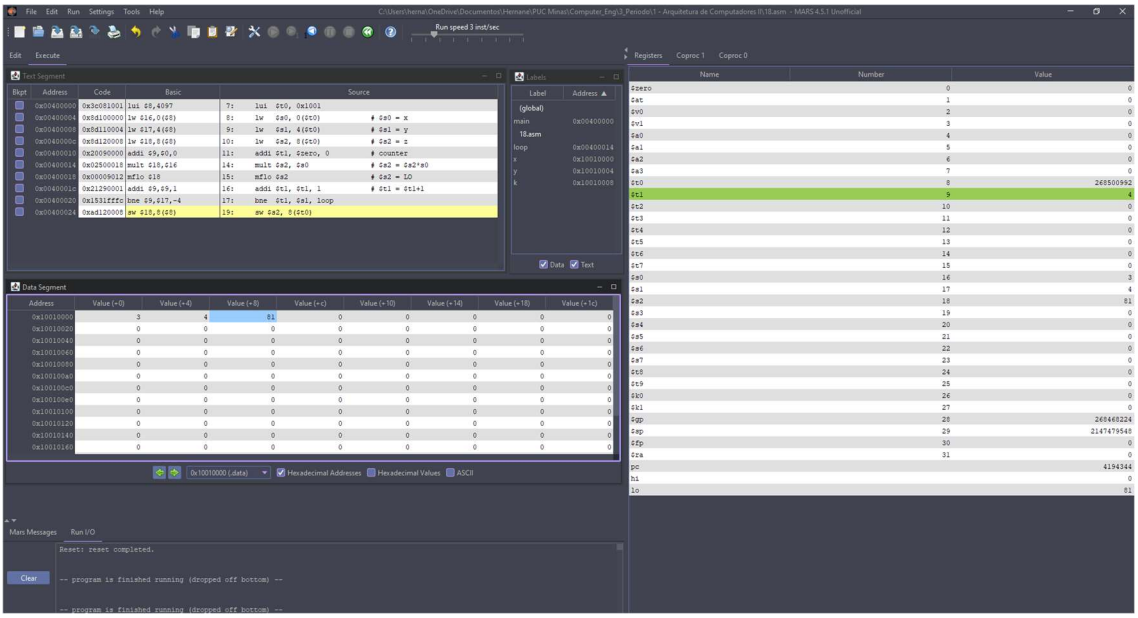
Programa 17 – Execução via MARS



Programa 18 – Código e quantidade de Instruções conforme o tipo

```
# Programa 18 - By Hernane V | Data: 04/12/2022
# Arquitetura de Computadores II - Romanelli
.text
.globl main

main:
    lui  $t0, 0x1001
    lw   $s0, 0($t0)            # $s0 = x
    lw   $s1, 4($t0)            # $s1 = y
    lw   $s2, 8($t0)            # $s2 = z
    addi $t1, $zero, 0          # counter

loop:
    mult $s2, $s0               # $s2 = $s2*s0
    mflo $s2                    # $s2 = LO
    addi $t1, $t1, 1            # $t1 = $t1+1
    bne  $t1, $s1, loop

    sw $s2, 8($t0)

.data
    x: .word 3
    y: .word 4
    k: .word 1
```

Programa 18 – Execução via MARS



REPOSITÓRIO DE PMG-AC2

Disponível em: https://github.com/hernanevelozo/PMG-AC2

REFERÊNCIAS:

HENNESSY, John L. **Arquitetura de computadores uma abordagem quantitativa**. Rio de Janeiro **GEN LTC 2019 1** recurso online ISBN 9788595150669.

SPRINGERLINK (ONLINE SERVICE). **The MIPS-X RISC Microprocessor**. 1st **ed. 1989. XXIV**, 232 p ISBN9781475767629.