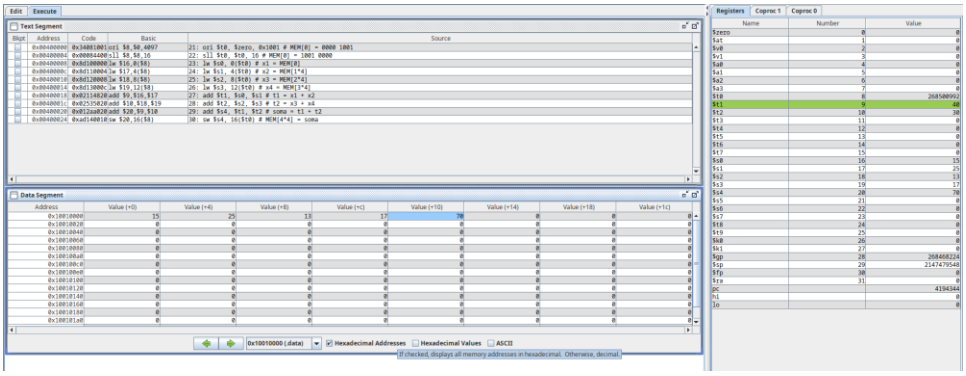
Felipe Augusto Morais Silva

Programa 09

mips9.asm

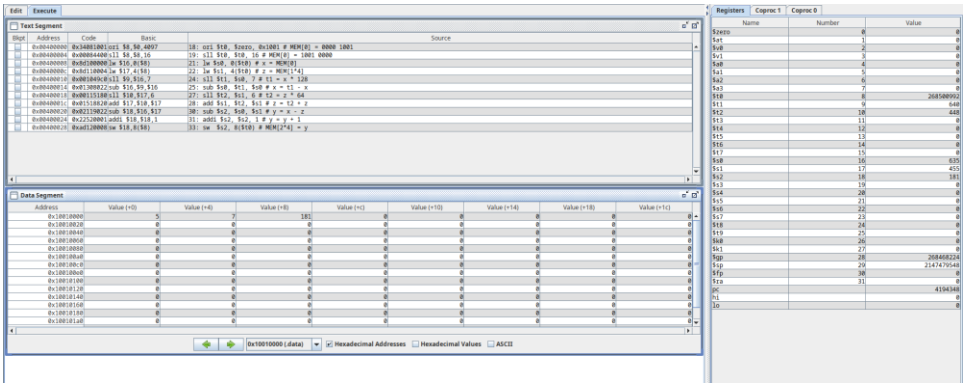
```
# programa 9
# .data
# x1: .word 15
# x2: .word 25
# x3: .word 13
# x4: .word 17
# soma: .word -1

# Associações :
# t0 = MEM[0]
# x1 = $s0
# x2 = $s1
# x3 = $s2
# x4 = $s3
# soma = $s4

#inicio
.text
.globl main
main:
ori $t0, $zero, 0x1001 # MEM[0] = 0000 1001
sll $t0, $t0, 16 # MEM[0] = 1001 0000
lw $s0, 0($t0) # x1 = MEM[0]
lw $s1, 4($t0) # x2 = MEM[1*4]
lw $s2, 8($t0) # x3 = MEM[2*4]
lw $s3, 12($t0) # x4 = MEM[3*4]
add $t1, $s0, $s1 # t1 = x1 + x2
add $t2, $s2, $s3 # t2 = x3 + x4
add $s4, $t1, $t2 # soma = t1 + t2
sw $s4, 16($t0) # MEM[4*4] = soma
.data
x1: .word 15
x2: .word 25
x3: .word 13
x4: .word 17
soma: .word -1
#fim

```



Programa 10

mips9.asm | mips10.asm

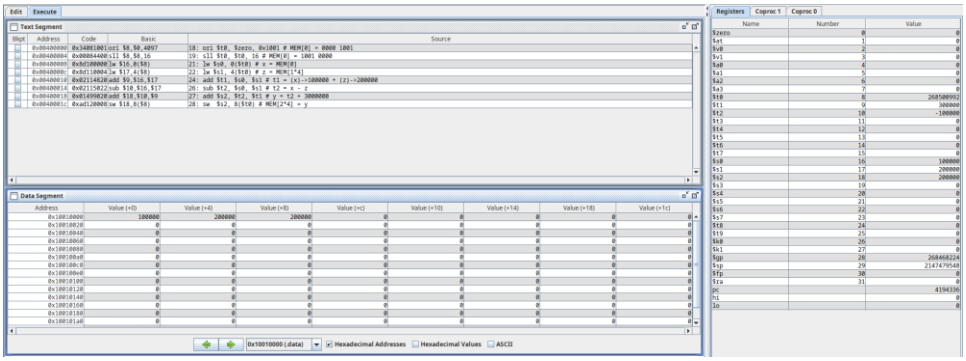
```
1  # programa 10
2  #  y = 127x - 65z + 1
3  # .data
4  # x: .word 5
5  # z: .word 7
6  # y: .word 0
7
8  # Associações :
9  # t0 = MEM[0]
10 # x = $s0
11 # z = $s1
12 # y = $s2
13
14 #inicio
15 .text
16 .globl main
17 main:
18 ori $t0, $zero, 0x1001 # MEM[0] = 0000 1001
19 sll $t0, $t0, 16 # MEM[0] = 1001 0000
20
21 lw $s0, 0($t0) # x = MEM[0]
22 lw $s1, 4($t0) # z = MEM[1*4]
23
24 sll $t1, $s0, 7 # t1 = x * 128
25 sub $s0, $t1, $s0 # x = t1 - x
26
27 sll $t2, $s1, 6 # t2 = z * 64
28 add $s1, $t2, $s1 # z = t2 + z
29
30 sub $s2, $s0, $s1 # y = x - z
31 addi $s2, $s2, 1 # y = y + 1
32
33 sw  $s2, 8($t0) # MEM[2*4] = y
34
35 .data
36 x: .word 5
37 z: .word 7
38 y: .word 0
39 #fim
40
```



Programa 11

mips9.asm | mips10.asm | mips11.asm

```
1  # programa 11
2  # y = x - z + 300000
3  # x: .word 100000
4  # z: .word 200000
5  # y: .word 0
6
7  # Associações :
8  # t0 = MEM[0]
9  # x = $s0
10 # z = $s1
11 # y = $s2
12
13 #inicio
14 .text
15 .globl main
16 main:
17
18 ori $t0, $zero, 0x1001 # MEM[0] = 0000 1001
19 sll $t0, $t0, 16 # MEM[0] = 1001 0000
20
21 lw $s0, 0($t0) # x = MEM[0]
22 lw $s1, 4($t0) # z = MEM[1*4]
23
24 add $t1, $s0, $s1 # t1 = (x)->100000 + (z)->200000
25
26 sub $t2, $s0, $s1 # t2 = x - z
27 add $s2, $t2, $t1 # y = t2 + 3000000
28 sw  $s2, 8($t0) # MEM[2*4] = y
29
30 .data
31 x: .word 100000
32 z: .word 200000
33 y: .word 0
34 #inicio
35
```

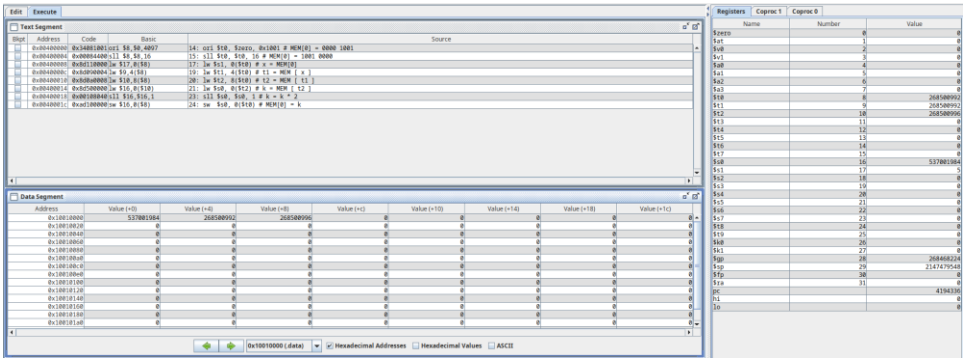


Programa 12

Edit Execute

mips9.asm | mips10.asm | mips11.asm | mips12.asm

```
1  # programa 12
2  # k = MEM [ MEM [ MEM [ x ] ] ].
3
4  # Associações :
5  # t0 = MEM[0]
6  # k = $s0
7  # x = $s1
8
9  #inicio
10 .text
11 .globl main
12 main:
13
14 ori $t0, $zero, 0x1001 # MEM[0] = 0000 1001
15 sll $t0, $t0, 16 # MEM[0] = 1001 0000
16
17 lw $s1, 0($t0) # x = MEM[0]
18
19 lw $t1, 4($t0) # t1 = MEM [ x ]
20 lw $t2, 8($t0) # t2 = MEM [ t1 ]
21 lw $s0, 0($t2) # k = MEM [ t2 ]
22
23 sll $s0, $s0, 1 # k = k * 2
24 sw  $s0, 0($t0) # MEM[0] = k
25
26 .data
27 x: .word 5
28 y: .word x
29 z: .word y
30 #fim
31
```

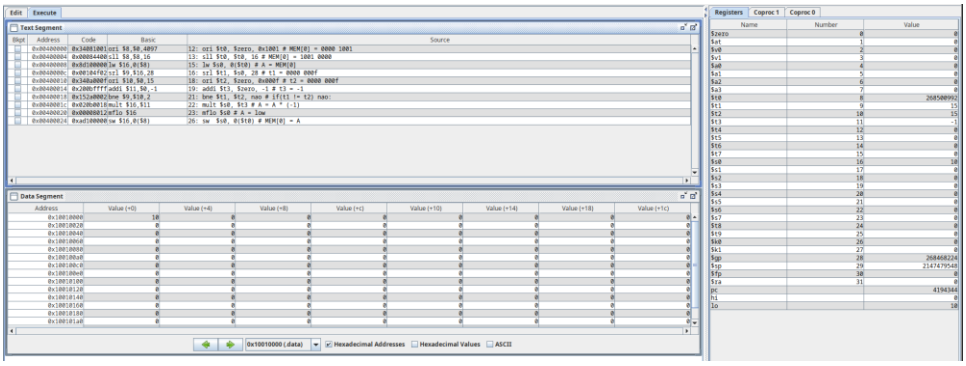


Programa 13

Edit Execute

mips9.asm | mips10.asm | mips11.asm | mips12.asm | mips13.asm

```
1  # programa 13
2
3  # Associações :
4  # t0 = MEM[0]
5  # A = $s0
6
7  #inicio
8  .text
9  .globl main
10 main:
11
12 ori $t0, $zero, 0x1001 # MEM[0] = 0000 1001
13 sll $t0, $t0, 16 # MEM[0] = 1001 0000
14
15 lw $s0, 0($t0) # A = MEM[0]
16 srl $t1, $s0, 28 # t1 = 0000 000f
17
18 ori $t2, $zero, 0x000f # t2 = 0000 000f
19 addi $t3, $zero, -1 # t3 = -1
20
21 bne $t1, $t2, nao # if(t1 != t2) nao:
22 mult $s0, $t3 # A = A * (-1)
23 mflo $s0 # A = low
24
25 nao:
26 sw  $s0, 0($t0) # MEM[0] = A
27
28 .data
29 A: .word -10
30 #fim
31
```

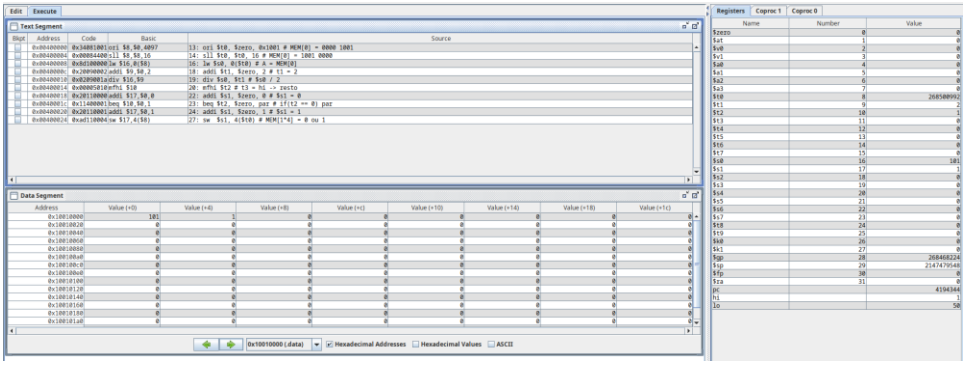


Programa 14

Edit Execute

mips9.asm | mips10.asm | mips11.asm | mips12.asm | mips13.asm | mips14.asm

```
1  # programa 14
2
3  # Associações :
4  # t0 = MEM[0]
5  # MEM[1] = $s1
6  # A = $s0
7
8  #inicio
9  .text
10 .globl main
11 main:
12
13 ori $t0, $zero, 0x1001 # MEM[0] = 0000 1001
14 sll $t0, $t0, 16 # MEM[0] = 1001 0000
15
16 lw $s0, 0($t0) # A = MEM[0]
17
18 addi $t1, $zero, 2 # t1 = 2
19 div $s0, $t1 # $s0 / 2
20 mfhi $t2 # t3 = hi -> resto
21
22 addi $s1, $zero, 0 # $s1 = 0
23 beq $t2, $zero, par # if(t2 == 0) par
24 addi $s1, $zero, 1 # $s1 = 1
25
26 par:
27 sw  $s1, 4($t0) # MEM[1*4] = 0 ou 1
28
29 .data
30 A: .word 101
31 #fim
```

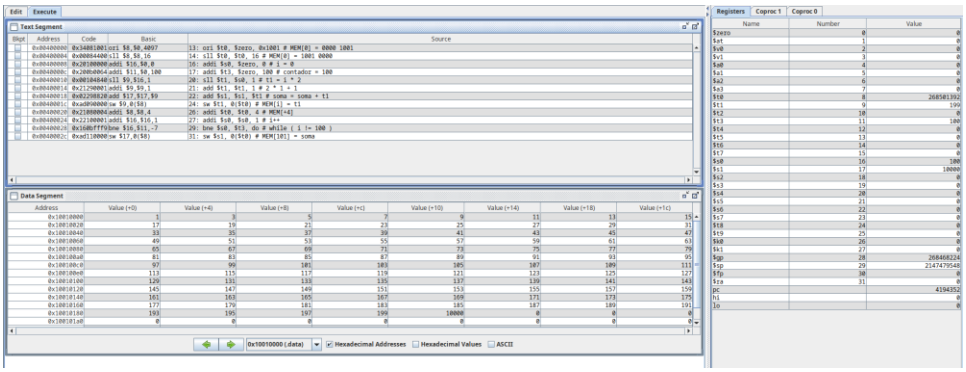


Programa 15

Edit Execute

mips9.asm | mips10.asm | mips11.asm | mips12.asm | mips13.asm | mips14.asm | mips15.asm

```
1  # programa 15
2
3  # Associações :
4  # t0 = MEM[0]
5  # i = $s0
6  # soma = $s1
7
8  #inicio
9  .text
10 .globl main
11 main:
12
13 ori $t0, $zero, 0x1001 # MEM[0] = 0000 1001
14 sll $t0, $t0, 16 # MEM[0] = 1001 0000
15
16 addi $s0, $zero, 0 # i = 0
17 addi $t3, $zero, 100 # contador = 100
18
19 do:
20 sll $t1, $s0, 1 # t1 = i * 2
21 add $t1, $t1, 1 # 2 * 1 + 1
22 add $s1, $s1, $t1 # soma = soma + t1
23
24 sw $t1, 0($t0) # MEM[i] = t1
25
26 addi $t0, $t0, 4 # MEM[+4]
27 addi $s0, $s0, 1 # i++
28
29 bne $s0, $t3, do # while ( i != 100 )
30
31 sw $s1, 0($t0) # MEM[101] = soma
32 #fim
33
```

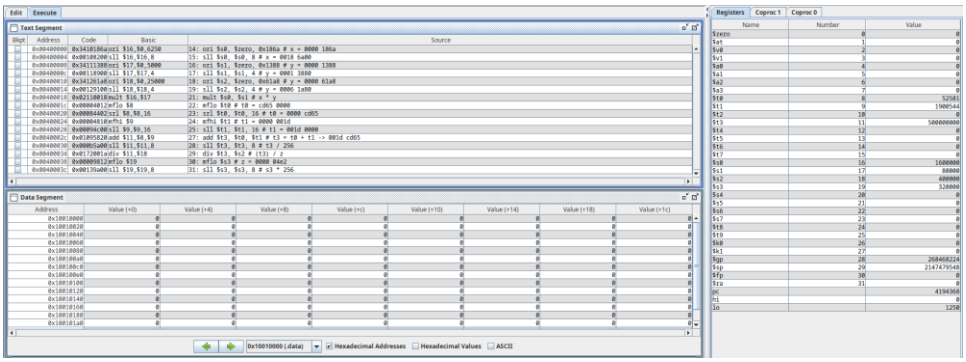


Programa 16

Edit Execute

mips9.asm | mips10.asm | mips11.asm | mips12.asm | mips13.asm | mips14.asm | mips15.asm | mips16.asm

```
# ((0x186A00*0x13880)/0x61A80)

# Associações :
# x = $s0
# y = $s1
# z = $s2
# resultado = $s3

#incio
.text
.globl main
main:
ori $s0, $zero, 0x186a # x = 0000 186a
sll $s0, $s0, 8 # x = 0018 6a00
ori $s1, $zero, 0x1388 # y = 0000 1388
sll $s1, $s1, 4 # y = 0001 3880
ori $s2, $zero, 0x61a8 # y = 0000 61a8
sll $s2, $s2, 4 # y = 0006 1a80

mult $s0, $s1 # x * y
mflo $t0 # t0 = cd65 0000
srl $t0, $t0, 16 # t0 = 0000 cd65
mfhi $t1 # t1 = 0000 001d
sll $t1, $t1, 16 # t1 = 001d 0000

add $t3, $t0, $t1 # t3 = t0 + t1 -> 001d cd65
sll $t3, $t3, 8 # t3 / 256
div $t3, $s2 # (t3) / z
mflo $s3 # z = 0000 04e2
sll $s3, $s3, 8 # s3 * 256
#fim
```

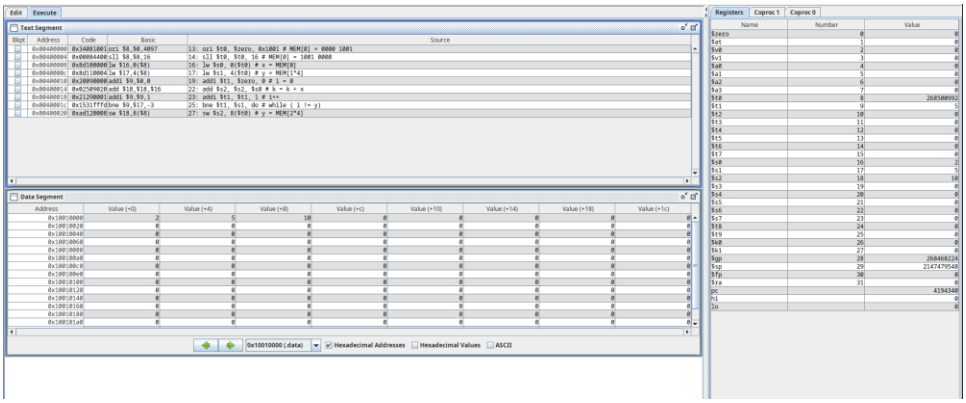


Programa 17

Edit Execute

mips9.asm | mips10.asm | mips11.asm | mips12.asm | mips13.asm | mips14.asm | mips15.asm | mips16.asm | mips17.asm

```
1  # programa 17
2
3  # Associações :
4  # t0 = MEM[0]
5  # x = $s0
6  # y = $s1
7  # k = $s2
8
9  #inicio
10 .text
11 .globl main
12 main:
13 ori $t0, $zero, 0x1001 # MEM[0] = 0000 1001
14 sll $t0, $t0, 16 # MEM[0] = 1001 0000
15
16 lw $s0, 0($t0) # x = MEM[0]
17 lw $s1, 4($t0) # y = MEM[1*4]
18
19 addi $t1, $zero, 0 # i = 0
20
21 do:
22 add $s2, $s2, $s0 # k = k + x
23 addi $t1, $t1, 1 # i++
24
25 bne $t1, $s1, do # while ( i != y)
26
27 sw $s2, 8($t0) # y = MEM[2*4]
28
29 .data
30 x: .word 2
31 y: .word 5
32 k: .word -1
33 #fim
34
```

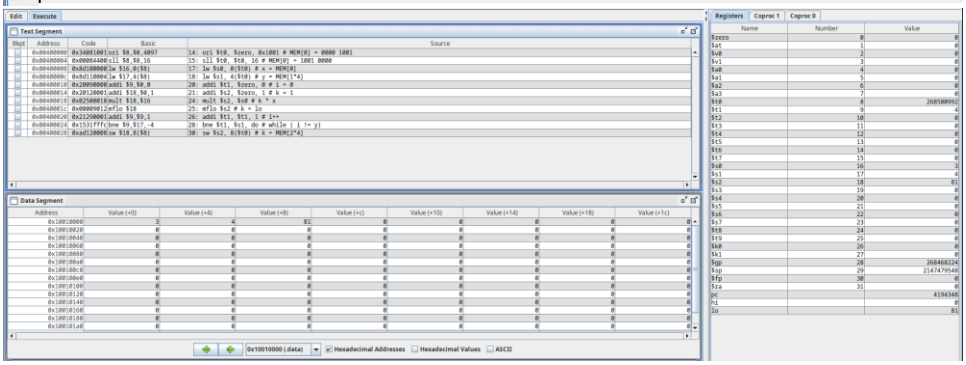


Programa 18

Edit Execute

mips9.asm | mips10.asm | mips11.asm | mips12.asm | mips13.asm | mips14.asm | mips15.asm | mips16.asm | mips17.asm | mips18.asm

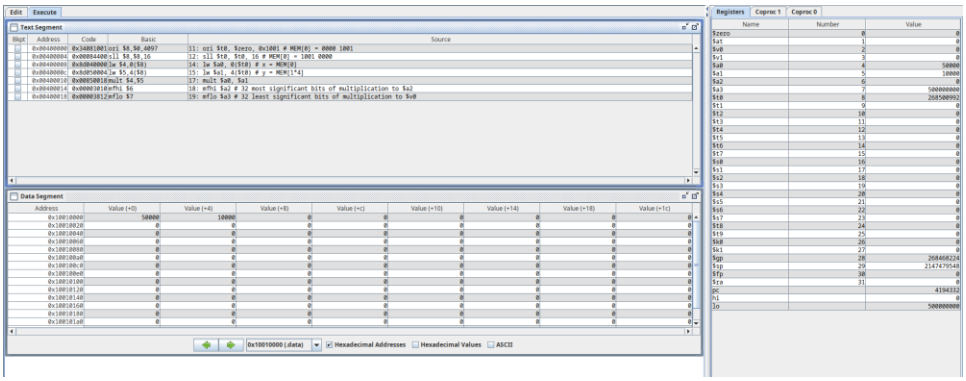
```
# programa 18

# Associações :
# t0 = MEM[0]
# x = $s0
# y = $s1
# k = $s2

#inicio
.text
.globl main
main:

ori $t0, $zero, 0x1001 # MEM[0] = 0000 1001
sll $t0, $t0, 16 # MEM[0] = 1001 0000

lw $s0, 0($t0) # x = MEM[0]
lw $s1, 4($t0) # y = MEM[1*4]

addi $t1, $zero, 0 # i = 0
addi $s2, $zero, 1 # k = 1

do:
mult $s2, $s0 # k * x
mflo $s2 # k = lo
addi $t1, $t1, 1 # i++

bne $t1, $s1, do # while ( i != y)

sw $s2, 8($t0) # k = MEM[2*4]

.data
x: .word 3
y: .word 4
k: .word -1
#fim
```



Programa desafio

mips9.asm | mips10.asm | mips11.asm | mips12.asm | mips13.asm | mips14.asm | mips15.asm | mips16.asm | mips17.asm | mips18.asm | desafio.asm

```
# programa desafio

#li $a0, 5 #load imediato
#li $a1, 3

#inicio
.text
.globl main
main:

ori $t0, $zero, 0x1001 # MEM[0] = 0000 1001
sll $t0, $t0, 16 # MEM[0] = 1001 0000

lw $a0, 0($t0) # x = MEM[0]
lw $a1, 4($t0) # y = MEM[1*4]

mult $a0, $a1
mfhi $a2 # 32 most significant bits of multiplication to $a2
mflo $a3 # 32 least significant bits of multiplication to $a3

.data
x: .word 50000
y: .word 10000

#fim
```


Edit
Execute
Text Segment
Bkpt Address Code Basic Source
0x00400000 0x3408100 ori $8,$0,4097 11: ori $t0, $zero, 0x1001 # MEM[0] = 0000 1001
0x00400004 0x00084400 sll $8,$8,16 12: sll $t0, $t0, 16 # MEM[0] = 1001 0000
0x00400008 0x8d040000 lw $4,0($8) 14: lw $a0, 0($t0) # x = MEM[0]
0x0040000c 0x8d050004 lw $5,4($8) 15: lw $a1, 4($t0) # y = MEM[1*4]
0x00400010 0x00850018 mult $4,$5 17: mult $a0, $a1
0x00400014 0x00003818 mfhi $6 18: mfhi $a2 # 32 most significant bits of multiplication to $a2
0x00400018 0x00003812 mflo $7 19: mflo $a3 # 32 least significant bits of multiplication to $v0
Data Segment
Address Value (+0) Value (+4) Value (+8) Value (+c) Value (+10) Value (+14) Value (+18) Value (+1c)
0x10010000 50000 10000 0 0 0 0 0 0
0x10010020 0 0 0 0 0 0 0 0
0x10010040 0 0 0 0 0 0 0 0
0x10010060 0 0 0 0 0 0 0 0
0x10010080 0 0 0 0 0 0 0 0
0x100100a0 0 0 0 0 0 0 0 0
0x100100c0 0 0 0 0 0 0 0 0
0x100100e0 0 0 0 0 0 0 0 0
0x10010100 0 0 0 0 0 0 0 0
0x10010120 0 0 0 0 0 0 0 0
0x10010140 0 0 0 0 0 0 0 0
0x10010160 0 0 0 0 0 0 0 0
0x10010180 0 0 0 0 0 0 0 0
0x100101a0 0 0 0 0 0 0 0 0
0x10010000 [.data)
Hexadecimal Addresses
Hexadecimal Values
ASCII
Registers
Coproc 1
Coproc 0
Name Number Value
$zero 0 0
$at 1 0
$v0 2 0
$v1 3 0
$a0 4 50000
$a1 5 10000
$a2 6 0
$a3 7 500000000
$t0 8 268500992
$t1 9 0
$t2 10 0
$t3 11 0
$t4 12 0
$t5 13 0
$t6 14 0
$t7 15 0
$s0 16 0
$s1 17 0
$s2 18 0
$s3 19 0
$s4 20 0
$s5 21 0
$s6 22 0
$s7 23 0
$t8 24 0
$t9 25 0
$k0 26 0
$k1 27 0
$gp 28 268468224
$sp 29 2147479548
$fp 30 0
$ra 31 0
pc 4194332
hi 0
lo 500000000