



PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS

Instituto de Ciências Exatas e Informática – ICEI

Departamento de Ciência da Computação

TRABALHO PRÁTICO 05

Belo Horizonte

2022

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS

Ciência da Computação

Hernane Velozo Rosa

hernane.rosa@sga.pucminas.br

Este trabalho apresenta respostas às perguntas teóricas envolvendo a linguagem Assembly no MIPS, além de conter os códigos dos exercícios práticos propostos.

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

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

RELATÓRIO: PARTE 01 - TEÓRICO

1. A) Um arquivo de texto que contém instruções de linguagem de programação.
2. B) Uma parte do processador que possui um padrão de bits.
3. A) #
4. C) 32 bits
5. D) parte do processador que contém o endereço da próxima instrução de máquina para ser obtida.
6. D) 8
7. A) Uma instrução em linguagem assembly que resulta em uma instrução em linguagem de máquina.
8. B) Um byte na memória que contém o endereço de dados.
9. A) 0x00000000
10. A) Operando imediato
11. A) Operação lógica
12. C) Ambos os operandos devem vir de registros.
13. B) Os dados são estendidos em zero à esquerda por 16 bits.
14. C) ori \$5,\$0,48
15. A) Não.
16. D) andi \$8,\$8,0xFF
17. A) Todos os bits em zero.
18. D) Sim. Todas as instruções de máquina têm os mesmos campos, mas eles podem estar em ordens diferentes.

RELATÓRIO: PARTE 02 - PRÁTICA

Programa 1 – Código

```
# Programa 1 - By Hernane V | Data 18/11/2022
# Arquitetura de Computadores II - Romanelli

# a = 2
# b = 2
# c = 4
# d = 5
# x = (a+b) - (c + d)
# y = a - b + x
# b = x - y

.text
.globl main
main:

addi $s0, $zero, 2      # a = 2
addi $s1, $zero, 2      # b = 2
addi $s2, $zero, 4      # c = 4
addi $s3, $zero, 5      # d = 5

add $t0, $s0, $s1       # t0 = a+b
add $t1, $s2, $s3       # t1 = c+d
sub $s4, $t0, $t1       # s4: x = (a+b) - (c+d)

sub $t0, $s0, $s1       # t0: y = a - b
add $s5, $t0, $s4       # s5: y = (a-b) + x

sub $s1, $s4, $s5       # s1: b = x - y
.data
#fim
```

Programa 1 – Execução via MARS

The screenshot displays the MARS MIPS simulator interface. The **Text Segment** window shows the assembly code with addresses and comments. The **Data Segment** window shows memory addresses from 0x10010000 to 0x100100e0. The **Registers** window shows the state of registers \$zero through \$t7, with \$s0 through \$s5 containing values 2, 2, 4, 5, 8, and -5 respectively. The **MARS Messages** window shows the message "program is finished running (dropped off bottom) --".

Name	Number	Value
\$zero	0	0
\$at	1	0
\$v0	2	0
\$v1	3	0
\$a0	4	0
\$a1	5	0
\$a2	6	0
\$a3	7	0
\$t0	8	0
\$t1	9	0
\$t2	10	0
\$t3	11	0
\$t4	12	0
\$t5	13	0
\$t6	14	0
\$t7	15	0
\$s0	16	2
\$s1	17	2
\$s2	18	4
\$s3	19	5
\$s4	20	-5
\$s5	21	-5
\$s6	22	0
\$s7	23	0
\$t8	24	0
\$t9	25	0
\$k0	26	0
\$k1	27	0
\$gp	28	268469224
\$sp	29	2147479544
\$fp	30	0
\$ra	31	0
pc		4194344
hi		0
lo		0

Programa 2 – Código

```
# Programa 2 - By Hernane V | Data 18/11/2022
# Arquitetura de Computadores II - Romanelli

# x = 1
# y = 5 * x + 15

.text
.globl main
main:

addi $s0, $zero, 1      # x = 1

add  $t0, $s0, $s0      # t0 = 2 * x
add  $t1, $t0, $t0      # t1 = 4 * x
add  $t0, $t1, $s0      # t0 = 5 * x

addi $s1, $t0, 15       # s1 = y = 5*x+15

.data

#fim
```

Programa 2 – Execução via MARS

The screenshot displays the MARS MIPS simulator interface. The **Text Segment** window shows the assembly code with addresses and comments. The **Data Segment** window shows memory addresses from 0x10010000 to 0x100100e0, all containing 0. The **Registers** window shows the state of registers, with \$s1 highlighted at 26. The **MARS Messages** window shows the message: "-- program is finished running (dropped off bottom) --".

Blkpt	Address	Code	Basic	Source
	0x00400000	0x20100001	addi \$t0, \$zero, 1	11: addi \$t0, \$zero, 1 # x = 1
	0x00400004	0x02104020	add \$t0, \$t0, \$t0	13: add \$t0, \$t0, \$t0 # t0 = 2 * x
	0x00400008	0x01084820	add \$t1, \$t0, \$t0	14: add \$t1, \$t0, \$t0 # t1 = 4 * x
	0x0040000c	0x01304020	add \$t0, \$t1, \$s0	15: add \$t0, \$t1, \$s0 # t0 = 5 * x
	0x00400010	0x2111000f	addi \$s1, \$t0, 15	17: addi \$s1, \$t0, 15 # s1 = y = 5*x+15

Name	Number	Value
\$zero	0	0
\$at	1	0
\$v0	2	0
\$v1	3	0
\$a0	4	0
\$a1	5	0
\$a2	6	0
\$a3	7	0
\$t0	8	5
\$t1	9	4
\$t2	10	0
\$t3	11	0
\$t4	12	0
\$t5	13	0
\$t6	14	0
\$t7	15	0
\$s0	16	1
\$s1	17	26
\$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		4194324
hi		0
lo		0

MARS Messages: -- program is finished running (dropped off bottom) --

Programa 3 – Código

```
# Programa 3 - By Hernane V | Data 18/11/2022
# Arquitetura de Computadores II - Romanelli

# x = 3
# y = 4
# z = (15*x + 67*y) * 4

.text
.globl main
main:
addi $s0, $zero, 3      # x = 3
addi $s1, $zero, 4      # y = 4

add $t0, $s0, $s0       # t0 = 2 * x
add $t0, $t0, $t0       # t0 = 4 * x
add $t0, $t0, $t0       # t0 = 8 * x
add $t0, $t0, $t0       # t0 = 16 * x
sub $t1, $t0, $s0       # t1 = 15 * x

add $t0, $s1, $s1       # t0 = 2 * y
add $t2, $t0, $t0       # t2 = 4 * y
add $t2, $t2, $t2       # t2 = 8 * y
add $t2, $t2, $t2       # t2 = 16 * y
add $t2, $t2, $t2       # t2 = 32 * y
add $t2, $t2, $t2       # t2 = 64 * y
add $t2, $t0, $t2       # t2 = 66 * y
add $t2, $t2, $s1       # t2 = 67 * y

add $t0, $t1, $t2       # t0 = t1 + t2

add $t0, $t0, $t0       # t0 = 2 * t0
add $s2, $t0, $t0       # s2 = 4 * t0

.data
#fim
```

Programa 3 – Execução via MARS

The screenshot displays the MARS MIPS simulator interface. The **Text Segment** window shows the assembly code with addresses and comments. The **Data Segment** window shows memory addresses from 0x10010000 to 0x100100e0. The **Registers** window shows the state of registers, with \$s2 highlighted at 1258. The **MARS Messages** window shows the message: "-- program is finished running (dropped off bottom) --".

Registers	Coproc 1	Coproc 0
Name	Number	Value
\$zero	0	0
\$at	1	0
\$v0	2	0
\$v1	3	0
\$a0	4	0
\$a1	5	0
\$a2	6	0
\$a3	7	0
\$t0	8	626
\$t1	9	45
\$t2	10	268
\$t3	11	0
\$t4	12	0
\$t5	13	0
\$t6	14	0
\$t7	15	0
\$s0	16	3
\$s1	17	4
\$s2	18	1258
\$s3	19	0
\$s4	20	0
\$s5	21	0
\$s6	22	0
\$s7	23	0
\$s8	24	0
\$s9	25	0
\$k0	26	0
\$k1	27	0
\$gp	28	268469224
\$sp	29	2147479549
\$fp	30	0
\$ra	31	0
pc		4194376
hi		0
lo		0

Programa 4 – Código

```
# Programa 4 - By Hernane V | Data 18/11/2022
# Arquitetura de Computadores II - Romanelli

# x = 3
# y = 4
# z = (15*x + 67*y) * 4

.text
.globl main
main:

addi $s0, $zero, 3      # x = 3
addi $s1, $zero, 4      # x = 4

sll $t0, $s0, 4          # t0 = x * 2 ^ 4 = 16 * x
sub $t0, $t0, $s0        # t0 = 16 * x - x = 15 * x

sll $t1, $s1, 6          # t1 = y * 2 ^ 6 = 64 * y
sll $t2, $s1, 2          # t2 = y * 2 ^ 2 = 4 * y
add $t1, $t1, $t2        # t2 = 64 * y + 4 * y = 68 * y
sub $t1, $t1, $s1        # t1 = 68 * y - y = 67 * y

add $t2, $t0, $t1        # t2 = 15x + 67y
add $t2, $t2, $t2        # t2 = (15x + 67y) * 2
add $s2, $t2, $t2        # z = (15x + 67y) * 4

.data

#fim
```

Programa 4 – Execução via MARS

The screenshot displays the MARS MIPS simulator interface. The **Text Segment** window shows the assembly code with line numbers 13 to 25. The **Data Segment** window shows memory addresses from 0x10010000 to 0x1001000F. The **Registers** window shows the state of registers \$zero through \$lo, with \$s2 containing the final value 4194348. The **MARS Messages** window shows the message "program is finished running (dropped off bottom)".

Name	Number	Value
\$zero	0	0
\$at	1	0
\$v0	2	0
\$v1	3	0
\$a0	4	0
\$a1	5	0
\$a2	6	0
\$a3	7	0
\$t0	8	45
\$t1	9	268
\$t2	10	626
\$t3	11	0
\$t4	12	0
\$t5	13	0
\$t6	14	0
\$t7	15	0
\$s0	16	3
\$s1	17	4
\$s2	18	4194348
\$s3	19	0
\$s4	20	0
\$s5	21	0
\$s6	22	0
\$s7	23	0
\$s8	24	0
\$s9	25	0
\$k0	26	0
\$k1	27	0
\$gp	28	268468224
\$sp	29	2147479548
\$fp	30	0
\$ra	31	0
\$pc		4194348
\$hi		0
\$lo		0

Programa 5 – Código

```
# Programa 5 - By Hernane V | Data 18/11/2022
# Arquitetura de Computadores II - Romanelli

# x = 3
# y = 4
# z = (15*x + 67*y) * 4

.text
.globl main
main:

# = x
# = y
# = z

addi $t0, $zero, 32767      # t0 = 32767
sll  $t0, $t0, 2            # t0 = t0 * 2 ^ 2
addi $t1, $zero, 31068     # t1 = 31068
sub  $s0, $t0, $t1         # s0 = t0 - t1
sll  $s1, $s0, 1           # s1 = s0 * 2 ^ 1
add  $s2, $s0, $s1         # s2 = s0 + s1
.data

#fim
```

Programa 5 – Execução via MARS

The screenshot displays the MARS MIPS simulator interface. The main window is divided into several panes:

- Text Segment:** Shows the assembly code with addresses and comments. The code is as follows:

Bkpt	Address	Code	Basic	Source
	0x00400000	addi \$t0,\$zero,32767	17: addi \$t0, \$zero, 32767	# t0 = 32767
	0x00400004	sll \$t0,\$t0,2	18: sll \$t0, \$t0, 2	# t0 = t0 * 2 ^ 2
	0x00400008	addi \$t1,\$zero,31068	19: addi \$t1, \$zero, 31068	# t1 = 31068
	0x0040000c	sub \$s0,\$t0,\$t1	20: sub \$s0, \$t0, \$t1	# s0 = t0 - t1
	0x00400010	sll \$s1,\$s0,1	21: sll \$s1, \$s0, 1	# s1 = s0 * 2 ^ 1
	0x00400014	add \$s2,\$s0,\$s1	22: add \$s2, \$s0, \$s1	# s2 = s0 + s1
- Data Segment:** Shows memory addresses from 0x10010000 to 0x1001000F, all containing 0.
- Registers:** Shows the state of MIPS registers. The \$s2 register is highlighted with a value of 300000.
- Mars Messages:** Shows the message "program is finished running (dropped off bottom) --".

Programa 6 – Código

```
# Programa 6 - By Hernane V | Data 18/11/2022
# Arquitetura de Computadores II - Romanelli

# x = 32767
# y = 300000
# z = x - 4y

.text
.globl main
main:

addi $s0, $zero, 32767 # s0 = 32767

sll $t0, $s0, 3          # t0 = t0 * 2^3
addi $t1, $zero, 5097    # t1 = 5097

add $t2, $s0, $t1        # t2 = s0 + t1
add $s1, $t0, $t2        # s1 = t0 + t2

sll $s1, $s1, 2          # s1 = s1 * 2^2
sub $s2, $s0, $s1        # s2 = s0 + s1
.data

#fim
```

Programa 6 – Execução via MARS

The screenshot displays the MARS MIPS simulator interface. The **Text Segment** window shows the assembly code with addresses and comments. The **Data Segment** window shows memory addresses from 0x10010000 to 0x1001000f. The **Registers** window shows the state of MIPS registers, with \$s2 highlighted in green and containing the value -1167233. The **Mars Messages** window shows the message "program is finished running (dropped off bottom) --".

Name	Number	Value
\$zero	0	0
\$at	1	0
\$v0	2	0
\$v1	3	0
\$a0	4	0
\$a1	5	0
\$a2	6	0
\$a3	7	0
\$t0	8	262136
\$t1	9	5097
\$t2	10	37864
\$t3	11	0
\$t4	12	0
\$t5	13	0
\$t6	14	0
\$t7	15	0
\$s0	16	32767
\$s1	17	1200000
\$s2	18	-1167233
\$s3	19	0
\$s4	20	0
\$s5	21	0
\$s6	22	0
\$s7	23	0
\$s8	24	0
\$s9	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		0

Programa 7 – Código

```
# Programa 7 - By Hernane V | Data 18/11/2022
# Arquitetura de Computadores II - Romanelli

# ori $8, $0, 0x01

.text
.globl main
main:
ori $8, $0, 0x01      # t0 = 1
sll $8, $8, 31        # t0 = 0x80000000
sra $8, $8, 31        # t0 = 0xffffffff

.data

#fim
```

Programa 7 – Execução via MARS

The screenshot shows the MARS MIPS simulator interface. The main window is divided into several panes:

- Text Segment:** Displays the assembly code with comments. The code is as follows:

```
0x00400000: ori $8, $0, 0x00000001 # t0 = 1
0x00400004: sll $8, $8, 31 # t0 = 0x80000000
0x00400008: sra $8, $8, 31 # t0 = 0xffffffff
```
- Data Segment:** Shows memory addresses and values. The values are all 0x00000000.
- Registers:** Shows the state of MIPS registers. The register \$t0 is highlighted in green, indicating its value is 0xffffffff.
- Mars Messages:** Shows the message "program is finished running (dropped off bottom)".

Programa 8 – Código

```
# Programa 7 - By Hernane V | Data 18/11/2022
# Arquitetura de Computadores II - Romanelli

.text
.globl main
main:

addi, $8, $zero, 0x12345678

srl $9, $8, 24          # $9 = 0x12

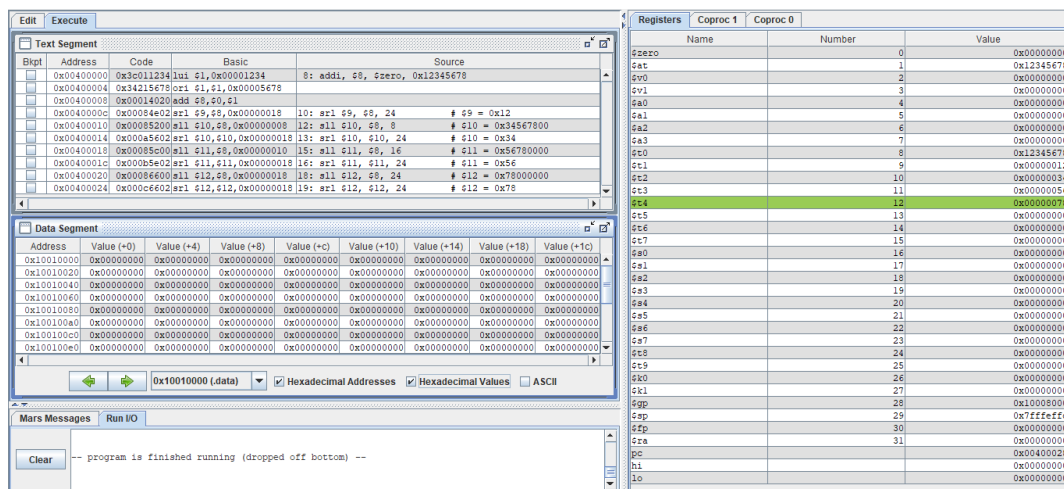
sll $10, $8, 8          # $10 = 0x34567800
srl $10, $10, 24        # $10 = 0x34

sll $11, $8, 16         # $11 = 0x56780000
srl $11, $11, 24        # $11 = 0x56

sll $12, $8, 24         # $12 = 0x78000000
srl $12, $12, 24        # $12 = 0x78

.data
#fim
```

Programa 8 – 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.