

Relatório

Responda

1. O que é um arquivo fonte?

A. um arquivo de texto que contém instruções de linguagem de programação.

B. um subdiretório que contém os programas.

C. um arquivo que contém dados para um programa.

D. um documento que contém os requisitos para um projeto.

2. O que é um registro?

A. parte do sistema de computador que mantém o controle dos parâmetros do sistema.

B. uma parte do processador que possui um padrão de bits.

C. parte do processador que contém o seu número de série único.

D. parte do bus de sistema que contém dados.

3. Qual o caracter que, na linguagem assembly do SPIM, inicia um comentário?

A. #

B. \$

C. //

D. *

4. Quantos bits há em cada instrução de máquina MIPS?

A. 8

B. 16

C. 32

D. instruções diferentes possuem diferentes comprimentos.

5. Quando você abre um arquivo de origem a partir do menu Arquivo do SPIM, quais as duas coisas que acontecem?
- A. O arquivo está carregado na memória e começa a execução.
 - B. SPIM é iniciado e o arquivo é aberto no editor.**
 - C. O arquivo é montado em instruções de máquina, e as instruções de máquina são carregados na memória do SPIM.
 - D. O programa é executado e os resultados são salvos em disco.
6. O que é o contador de programa?
- A. um registrador que mantém a conta do número de erros durante a execução de um programa.
 - B. uma parte do processador que contém o endereço da primeira palavra de dados.**
 - C. uma variável na montadora que os números das linhas do arquivo de origem.
 - D. parte do processador que contém o endereço da próxima instrução de máquina para ser obtida.
7. Ao pressionar a tecla F10 para executar uma instrução, quanto será adicionado ao contador de programa?
- A. 1
 - B. 2
 - C. 4**
 - D. 8
8. O que é uma diretiva, tal como a diretiva .text?
- A. uma instrução em linguagem assembly que resulta em uma instrução em linguagem de máquina.
 - B. uma das opções de menu do sistema SPIM.
 - C. uma instrução em linguagem de máquina que faz com que uma operação sobre os dados ocorra.
 - D. uma declaração que diz o montador algo sobre o que o programador quer, mas não corresponde diretamente a uma instrução de máquina.**
9. O que é um endereço simbólico?
- A. um local de memória que contém dados simbólicos.
 - B. um byte na memória que contém o endereço de dados.
 - C. símbolo dado como argumento para uma directiva.**
 - D. um nome usado no código-fonte em linguagem assembly para um local na memória.
10. Em qual endereço o simulador SPIM coloca a primeira instrução de máquina quando ele está sendo executado com a opção Bare Machine ligada?
- A. 0x00000000
 - B. 0x00400000**
 - C. 0x10000000
 - D. 0xFFFFFFFF
11. Algumas instruções de máquina possuem uma constante como um dos operandos. Como é chamado tal operando?
- A. operando imediato**
 - B. operando embutido
 - C. operando binário
 - D. operando de máquina
12. Como é chamada uma operação lógica executada entre bits de cada coluna dos operandos para produzir um bit de resultado para cada coluna?
- A. operação lógica

B. operação bitwise

C. operação binária

D. operação coluna

13. Quando uma operação é de fato executada, como estão os operandos na ALU?

A. Pelo menos um operando deve ser de 32 bit.

B. Cada operando pode ser de qualquer tamanho.

C. Ambos operandos devem vir de registros.

D. Cada um dos registradores deve possuir 32 bit.

14. Dezesesseis bits de dados de uma instrução de ori são usados como um operando imediato. Durante execução, o que deve ser feito primeiro?

A. Os dados são estendidos em zero à direita por 16 bits.

B. Os dados são estendidos em zero à esquerda por 16 bits.

C. Nada precisa ser feito.

D. Apenas 16 bits são usados pelo outro operando.

15. Qual o nome para um padrão de bits copiados em um registrador?

A. load.

B. filled.

C. stuffed.

D. fixed.

16. Qual das instruções seguintes armazenam no registrador \$5 um padrão de bits que representa positivo 48?

A. ori \$5,\$0,0x48

B. ori \$5,\$5,0x48

C. ori \$5,\$0,48

D. ori \$0,\$5,0x48

17. A instrução de ori pode armazenar o complemento de dois de um número em um registrador?

A. Não.

B. Sim.

18. Qual das instruções seguintes limpa todos os bits no registrador \$8 com exceção do byte de baixa ordem que fica inalterado?

A. **ori \$8,\$8,0xFF**

B. ori \$8,\$0,0x00FF

C. xori \$8,\$8,0xFF

D. andi \$8,\$8,0xFF

19. Qual é o resultado de um ou exclusivo de padrão sobre ele mesmo?

A. Todos os bits em zero.

B. Todos os bits em um.

C. O padrão original utilizado.

D. O resultado é o contrário do original.

20. Todas as instruções de máquina têm os mesmos campos?

A. Não. Diferentes de instruções de máquina possuem campos diferentes.

B. Não. Cada instrução de máquina é completamente diferente de qualquer outra.

C. Sim. Todas as instruções de máquina têm os mesmos campos na mesma ordem.

D. Sim. Todas as instruções de máquina têm os mesmos campos, mas eles podem estar em ordens diferentes.

Programa01

PCSpim

File Simulator Window Help

Status = 3000fff10 HI = 00000000 LO = 00000000

General Registers

R0 (r0) = 0	R8 (t0) = 5	R16 (s0) = 2	R24 (t8) = 0
R1 (at) = 0	R9 (t1) = 9	R17 (s1) = 3	R25 (t9) = 0
R2 (v0) = 0	R10 (t2) = -1	R18 (s2) = 4	R26 (k0) = 0
R3 (v1) = 0	R11 (t3) = 0	R19 (s3) = 5	R27 (k1) = 0
R4 (a0) = 0	R12 (t4) = 0	R20 (s4) = -4	R28 (gp) = 0
R5 (a1) = 0	R13 (t5) = 0	R21 (s5) = -5	R29 (sp) = 2147479548

Assembly code:

```
[0x00400008] 0x20120004 addi $18, $0, 4 ; 16: addi $s2, $zero, 4 #c=4
[0x0040000c] 0x20130005 addi $19, $0, 5 ; 17: addi $s3, $zero, 5 #d=5
[0x00400010] 0x02114020 add $8, $16, $17 ; 18: add $t0, $s0, $s1 #t0=(a+b)
[0x00400014] 0x02534820 add $9, $18, $19 ; 19: add $t1, $s2, $s3 #t1=(c+d)
[0x00400018] 0x0109a022 sub $20, $8, $9 ; 20: sub $s4, $t0, $t1 #x=t0-t1
[0x0040001c] 0x02115022 sub $10, $16, $17 ; 21: sub $t2, $s0, $s1 #t2=a-b
[0x00400020] 0x028aa820 add $21, $20, $10 ; 22: add $s5, $s4, $t2 #y=t2+x
[0x00400024] 0x02958822 sub $17, $20, $21 ; 23: sub $s1, $s4, $s5 #b=x-y
```

DATA

[0x10000000]...[0x10040000] 0x00000000

STACK

[0x7ffffeffc] 0x00000000

[0x7ffff000]...[0x80000000] 0x00000000

KERNEL DATA

Assembly code:

```
[0x00400004] 0x20110003 addi $17, $0, 3 ; 15: addi $s1, $zero, 3 #b=3
[0x00400008] 0x20120004 addi $18, $0, 4 ; 16: addi $s2, $zero, 4 #c=4
[0x0040000c] 0x20130005 addi $19, $0, 5 ; 17: addi $s3, $zero, 5 #d=5
[0x00400010] 0x02114020 add $8, $16, $17 ; 18: add $t0, $s0, $s1 #t0=(a+b)
[0x00400014] 0x02534820 add $9, $18, $19 ; 19: add $t1, $s2, $s3 #t1=(c+d)
[0x00400018] 0x0109a022 sub $20, $8, $9 ; 20: sub $s4, $t0, $t1 #x=t0-t1
[0x0040001c] 0x02115022 sub $10, $16, $17 ; 21: sub $t2, $s0, $s1 #t2=a-b
[0x00400020] 0x028aa820 add $21, $20, $10 ; 22: add $s5, $s4, $t2 #y=t2+x
```

For Help, press F1

PC=0x00400024 EPC=0x00000000 Cause=0x00000000

BARE DELAY BR DELAY LD

08:38

Programa02

PCSpim

File Simulator Window Help

PC = 0040001c EPC = 00000000 Cause = 00000000 BadVAddr= 00000000

Status = 3000fff10 HI = 00000000 LO = 00000000

General Registers

R0 (r0) = 0	R8 (t0) = 4	R16 (s0) = 1	R24 (t8) = 0
R1 (at) = 0	R9 (t1) = 0	R17 (s1) = 20	R25 (t9) = 0
R2 (v0) = 0	R10 (t2) = 0	R18 (s2) = 0	R26 (k0) = 0
R3 (v1) = 0	R11 (t3) = 0	R19 (s3) = 0	R27 (k1) = 0
R4 (a0) = 0	R12 (t4) = 0	R20 (s4) = 0	R28 (gp) = 0

Assembly code:

```
[0x00400000] 0x20100001 addi $16, $0, 1 ; 9: addi $s0, $zero, 1 #x=1
[0x00400004] 0x00104020 add $8, $0, $16 ; 10: add $t0, $zero, $s0 #t0=x
[0x00400008] 0x01084020 add $8, $8, $8 ; 11: add $t0, $t0, $t0 #t0=2x
[0x0040000c] 0x01084020 add $8, $8, $8 ; 12: add $t0, $t0, $t0 #t0=4x
[0x00400010] 0x00088820 add $17, $0, $8 ; 13: add $s1, $zero, $t0 #y=t0
[0x00400014] 0x02088820 add $17, $16, $8 ; 14: add $s1, $s0, $t0 #y=5x
[0x00400018] 0x2231000f addi $17, $17, 15 ; 15: addi $s1, $s1, 15 #y=5x+15
```

DATA

[0x10000000]...[0x10040000] 0x00000000

STACK

[0x7ffffeffc] 0x00000000

[0x7ffff000]...[0x80000000] 0x00000000

KERNEL DATA

C:\Users\Lu\Documents\Faculdade\AC II\Relatório7\programa2.asm successfully loaded

Assembly code:

```
[0x00400000] 0x20100001 addi $16, $0, 1 ; 9: addi $s0, $zero, 1 #x=1
[0x00400004] 0x00104020 add $8, $0, $16 ; 10: add $t0, $zero, $s0 #t0=x
[0x00400008] 0x01084020 add $8, $8, $8 ; 11: add $t0, $t0, $t0 #t0=2x
[0x0040000c] 0x01084020 add $8, $8, $8 ; 12: add $t0, $t0, $t0 #t0=4x
[0x00400010] 0x00088820 add $17, $0, $8 ; 13: add $s1, $zero, $t0 #y=t0
[0x00400014] 0x02088820 add $17, $16, $8 ; 14: add $s1, $s0, $t0 #y=5x
[0x00400018] 0x2231000f addi $17, $17, 15 ; 15: addi $s1, $s1, 15 #y=5x+15
```

For Help, press F1

PC=0x0040001c EPC=0x00000000 Cause=0x00000000

BARE DELAY BR DELAY LD

15:38 27/04/2018

Programa03

PCSpin

File Simulator Window Help

PC = 0040005c EPC = 00000000 Cause = 00000000 BadVAddr= 00000000
Status = 3000fff10 HI = 00000000 LO = 00000000

General Registers

R0 (r0) = 0	R8 (t0) = 48	R16 (s0) = 3	R24 (t8) = 0
R1 (at) = 0	R9 (t1) = 45	R17 (s1) = 4	R25 (t9) = 0
R2 (v0) = 0	R10 (t2) = 256	R18 (s2) = 1252	R26 (k0) = 0
R3 (v1) = 0	R11 (t3) = 12	R19 (s3) = 0	R27 (k1) = 0
R4 (a0) = 0	R12 (t4) = 268	R20 (s4) = 0	R28 (gp) = 0

< >

```
[0x0040003c] 0x014a5020 add $10, $10, $10 ; 27: add $t2,$t2,$t2 #t2=32y
[0x00400040] 0x014a5020 add $10, $10, $10 ; 28: add $t2,$t2,$t2 #t2=64y
[0x00400044] 0x000a6020 add $12, $0, $10 ; 29: add $t4,$zero,$t2 #t4=64y
[0x00400048] 0x018b6020 add $12, $12, $11 ; 30: add $t4,$t4,$t3 #t4=67y
[0x0040004c] 0x000c9020 add $18, $0, $12 ; 32: add $s2,$zero,$t4 #s2=67y
[0x00400050] 0x02499020 add $18, $18, $9 ; 33: add $s2,$s2,$t1 #s2=67y+15x
[0x00400054] 0x02529020 add $18, $18, $18 ; 35: add $s2,$s2,$s2 #s2=2(67y+15x)
[0x00400058] 0x02529020 add $18, $18, $18 ; 36: add $s2,$s2,$s2 #s2=4(67y+15x)
```

< >

DATA
[0x10000000]...[0x10040000] 0x00000000

STACK
[0x7fffffc] 0x00000000
[0x7ffff00]...[0x80000000] 0x00000000

KERNEL DATA

< >

```
[0x00400040] 0x014a5020 add $10, $10, $10 ; 28: add $t2,$t2,$t2 #t2=64y
[0x00400044] 0x000a6020 add $12, $0, $10 ; 29: add $t4,$zero,$t2 #t4=64y
[0x00400048] 0x018b6020 add $12, $12, $11 ; 30: add $t4,$t4,$t3 #t4=67y
[0x0040004c] 0x000c9020 add $18, $0, $12 ; 32: add $s2,$zero,$t4 #s2=67y
[0x00400050] 0x02499020 add $18, $18, $9 ; 33: add $s2,$s2,$t1 #s2=67y+15x
[0x00400054] 0x02529020 add $18, $18, $18 ; 35: add $s2,$s2,$s2 #s2=2(67y+15x)
[0x00400058] 0x02529020 add $18, $18, $18 ; 36: add $s2,$s2,$s2 #s2=4(67y+15x)
```

< >

For Help, press F1

PC=0x0040005c EPC=0x00000000 Cause=0x00000000

BARE DELAY BR DELAY LD

15:40
27/04/2018

Programa04

PCSpin

File Simulator Window Help

PC = 00400034 EPC = 00000000 Cause = 00000000 BadVAddr= 00000000
Status = 3000fff10 HI = 00000000 LO = 00000000

General Registers

R0 (r0) = 0	R8 (t0) = 12	R16 (s0) = 45	R24 (t8) = 0
R1 (at) = 0	R9 (t1) = 0	R17 (s1) = 268	R25 (t9) = 0
R2 (v0) = 0	R10 (t2) = 0	R18 (s2) = 1252	R26 (k0) = 0
R3 (v1) = 0	R11 (t3) = 0	R19 (s3) = 0	R27 (k1) = 0
R4 (a0) = 0	R12 (t4) = 0	R20 (s4) = 0	R28 (gp) = 0

< >

```
[0x00400014] 0x00114020 add $8, $0, $17 ; 17: add $t0,$zero,$s1 #t0=y
[0x00400018] 0x01084020 add $8, $8, $8 ; 18: add $t0,$t0,$t0 #t0=2y
[0x0040001c] 0x00114020 add $8, $8, $17 ; 19: add $t0,$t0,$s1 #t0=3y
[0x00400020] 0x00118980 sll $17, $17, 6 ; 20: sll $s1,$s1,6 #s1=64y
[0x00400024] 0x02288820 add $17, $17, $8 ; 21: add $s1,$s1,$t0 #s1=67y
[0x00400028] 0x00109020 add $18, $0, $16 ; 23: add $s2,$zero,$s0 #z=15x
[0x0040002c] 0x02519020 add $18, $18, $17 ; 24: add $s2,$s2,$s1 #z=15x+67y
[0x00400030] 0x00129080 sll $18, $18, 2 ; 26: sll $s2,$s2,2 #z=4(15x+67y)
```

< >

DATA
[0x10000000]...[0x10040000] 0x00000000

STACK
[0x7fffffc] 0x00000000
[0x7ffff00]...[0x80000000] 0x00000000

KERNEL DATA

< >

```
[0x00400014] 0x00114020 add $8, $0, $17 ; 17: add $t0,$zero,$s1 #t0=y
[0x00400018] 0x01084020 add $8, $8, $8 ; 18: add $t0,$t0,$t0 #t0=2y
[0x0040001c] 0x00114020 add $8, $8, $17 ; 19: add $t0,$t0,$s1 #t0=3y
[0x00400020] 0x00118980 sll $17, $17, 6 ; 20: sll $s1,$s1,6 #s1=64y
[0x00400024] 0x02288820 add $17, $17, $8 ; 21: add $s1,$s1,$t0 #s1=67y
[0x00400028] 0x00109020 add $18, $0, $16 ; 23: add $s2,$zero,$s0 #z=15x
[0x0040002c] 0x02519020 add $18, $18, $17 ; 24: add $s2,$s2,$s1 #z=15x+67y
[0x00400030] 0x00129080 sll $18, $18, 2 ; 26: sll $s2,$s2,2 #z=4(15x+67y)
```

< >

For Help, press F1

PC=0x00400034 EPC=0x00000000 Cause=0x00000000

BARE DELAY BR DELAY LD

15:41
27/04/2018

Programa05

PCSpin

File Simulator Window Help

PC = 00400014 EPC = 00000000 Cause = 00000000 BadVAddr= 00000000
Status = 3000ff10 HI = 00000000 LO = 00000000

General Registers

R0 (r0) = 0	R8 (t0) = 0	R16 (s0) = 100000	R24 (t8) = 0
R1 (at) = 0	R9 (t1) = 0	R17 (s1) = 200000	R25 (t9) = 0
R2 (v0) = 0	R10 (t2) = 0	R18 (s2) = 300000	R26 (k0) = 0
R3 (v1) = 0	R11 (t3) = 0	R19 (s3) = 0	R27 (k1) = 0
R4 (a0) = 0	R12 (t4) = 0	R20 (s4) = 0	R28 (gp) = 0

<

```
[0x00400000] 0x2010186a addi $16, $0, 6250      ; 12: addi $s0,$zero,0x186A    #x=0x186A
[0x00400004] 0x00108100 sll $16, $16, 4        ; 13: sll $s0,$s0, 4      #x=0x186A0
[0x00400008] 0x201130d4 addi $17, $0, 12500     ; 15: addi $s1,$zero,0x30D4  #y=0x30D4
[0x0040000c] 0x00118900 sll $17, $17, 4        ; 16: sll $s1,$s1,4      #y=0x30D40
[0x00400010] 0x02309020 add $18, $17, $16      ; 18: add $s2,$s1,$s0    #z=x+y
```

KERNEL

>

DATA

[0x10000000]...[0x10040000] 0x00000000

STACK

[0x7fffffc] 0x00000000

[0x7ffff00]...[0x80000000] 0x00000000

KERNEL DATA

<

See the file README for a full copyright notice.
C:\Users\Lu\Documents\Faculdade\AC II\Relatório7\programa5.asm successfully loaded

```
[0x00400000] 0x2010186a addi $16, $0, 6250      ; 12: addi $s0,$zero,0x186A    #x=0x186A
[0x00400004] 0x00108100 sll $16, $16, 4        ; 13: sll $s0,$s0, 4      #x=0x186A0
[0x00400008] 0x201130d4 addi $17, $0, 12500     ; 15: addi $s1,$zero,0x30D4  #y=0x30D4
[0x0040000c] 0x00118900 sll $17, $17, 4        ; 16: sll $s1,$s1,4      #y=0x30D40
[0x00400010] 0x02309020 add $18, $17, $16      ; 18: add $s2,$s1,$s0    #z=x+y
```

>

For Help, press F1

PC=0x00400014 EPC=0x00000000 Cause=0x00000000

BARE DELAY BR DELAY LD

15:43 27/04/2018

Programa06

C:\Users\Lu\Documents\Faculdade\AC II\Relatório7\programa6.asm - MARS 4.5

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400000	0x20107fff	addi \$16,\$0,0x00007fff	11: addi \$s0,\$zero,0x7FFF #x=0x7FFF
	0x00400004	0x00108400	sll \$16,\$16,0x00000010	12: sll \$s0,\$s0, 16 #x=16 bits para a esquerda
	0x00400008	0x3c010000	lui \$1,0x00000000	13: addi \$s0,\$s0,0xFFFF #x=0x7FFFFFFF
	0x0040000c	0x3421ffff	ori \$1,\$1,0x0000ffff	
	0x00400010	0x02018020	add \$16,\$16,\$1	
	0x00400014	0x2011493e	addi \$17,\$0,0x0000493e	15: addi \$s1,\$zero,0x493E #y=0x493E
	0x00400018	0x00118900	sll \$17,\$17,0x00000004	16: sll \$s1,\$s1,4 #y=0x493E0
	0x0040001c	0x00114020	add \$8,\$0,\$17	18: add \$t0,\$zero,\$s1 #t0=y
	0x00400020	0x00088880	sll \$17,\$8,0x00000002	19: sll \$s1,\$t0,2 #s1=4y
	0x00400024	0x02119022	sub \$18,\$16,\$17	21: sub \$s2,\$s0,\$s1 #z=x-y

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	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

Mars Messages Run I/O

Clear

-- program is finished running (dropped off bottom) --

Registers Coproc 1 Coproc 0

Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x0000ffff
\$v0	2	0x00000000
\$v1	3	0x00000000
\$a0	4	0x00000000
\$a1	5	0x00000000
\$a2	6	0x00000000
\$a3	7	0x00000000
\$t0	8	0x0000493e0
\$t1	9	0x00000000
\$t2	10	0x00000000
\$t3	11	0x00000000
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x7fffff7f
\$s1	17	0x00124930
\$s2	18	0x7ffefb7f
\$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	0x10000800
\$sp	29	0x7fffffc
\$fp	30	0x00000000
\$ra	31	0x00000000
pc		0x00400028
hi		0x00000000
lo		0x00000000

15:47 27/04/2018

Programa07

PCSpin

File Simulator Window Help

PC = 0040000c EPC = 00000000 Cause = 00000000 BadVAddr= 00000000
Status = 3000ff10 HI = 00000000 LO = 00000000

General Registers

R0 (r0) = 00000000	R8 (t0) = ffffffff	R16 (s0) = 00000000	R24 (t8) = 00000000
R1 (at) = 00000000	R9 (t1) = 00000000	R17 (s1) = 00000000	R25 (t9) = 00000000
R2 (v0) = 00000000	R10 (t2) = 00000000	R18 (s2) = 00000000	R26 (k0) = 00000000
R3 (v1) = 00000000	R11 (t3) = 00000000	R19 (s3) = 00000000	R27 (k1) = 00000000
R4 (a0) = 00000000	R12 (t4) = 00000000	R20 (s4) = 00000000	R28 (gp) = 00000000

<

```
[0x00400000] 0x34080001 ori $8, $0, 1          ; 7: ori $8, $0, 0x01    #colocar valor 1
[0x00400004] 0x000847c0 sll $8, $8, 31        ; 8: sll $8,$8,31     #shift até ficar na 1a posição
[0x00400008] 0x000847c3 sra $8, $8, 31        ; 9: sra $8,$8,31     #shift para encher de 1s
```

KERNEL

>

DATA

[0x10000000]...[0x10040000] 0x00000000

STACK

[0x7ffffcfc] 0x00000000

[0x7ffff000]...[0x80000000] 0x00000000

KERNEL DATA

<

All Rights Reserved.
SPIM is distributed under a BSD license.
See the file README for a full copyright notice.
C:\Users\Lu\Documents\Faculdade\AC II\Relatório7\programa7.asm successfully loaded

```
[0x00400000] 0x34080001 ori $8, $0, 1          ; 7: ori $8, $0, 0x01    #colocar valor 1
[0x00400004] 0x000847c0 sll $8, $8, 31        ; 8: sll $8,$8,31     #shift até ficar na 1a posição
[0x00400008] 0x000847c3 sra $8, $8, 31        ; 9: sra $8,$8,31     #shift para encher de 1s
```

>

For Help, press F1

PC=0x0040000c EPC=0x00000000 Cause=0x00000000

BARE DELAY BR DELAY LD

Digite aqui para pesquisar

Programa08

PCSpin

File Simulator Window Help

PC = 00400044 EPC = 00000000 Cause = 00000000 BadVAddr= 00000000
Status = 3000ff10 HI = 00000000 LO = 00000000

General Registers

R0 (r0) = 00000000	R8 (t0) = 12345678	R16 (s0) = 00000000	R24 (t8) = 00000000
R1 (at) = 00000000	R9 (t1) = 00000012	R17 (s1) = 00000000	R25 (t9) = 00000000
R2 (v0) = 00000000	R10 (t2) = 00000034	R18 (s2) = 00000000	R26 (k0) = 00000000
R3 (v1) = 00000000	R11 (t3) = 00000056	R19 (s3) = 00000000	R27 (k1) = 00000000
R4 (a0) = 00000000	R12 (t4) = 00000078	R20 (s4) = 00000000	R28 (gp) = 00000000

<

```
[0x00400024] 0x01485024 and $10, $10, $8      ; 22: and $10,$10,$8    #fazer um and para ter resultado requerido
[0x00400028] 0x000a5402 srl $10, $10, 16       ; 23: srl $10,$10,16    #reg9=0x12
[0x0040002c] 0x200b00ff addi $11, $0, 255      ; 25: addi $11,$zero,0xFF #reg9=0xFF
[0x00400030] 0x000b5a00 sll $11, $11, 8        ; 26: sll $11,$11,8     #coloca os FF no inicio
[0x00400034] 0x01685824 and $11, $11, $8      ; 27: and $11,$11,$8    #fazer um and para ter resultado requerido
[0x00400038] 0x000b5a02 srl $11, $11, 8        ; 28: srl $11,$11,8     #reg9=0x12
[0x0040003c] 0x200c00ff addi $12, $0, 255      ; 30: addi $12,$zero,0xFF #reg12=0xFF
[0x00400040] 0x01886024 and $12, $12, $8      ; 31: and $12,$12,$8    #fazer um and para ter resultado requerido
```

DATA

[0x10000000]...[0x10040000] 0x00000000

STACK

[0x7ffffcfc] 0x00000000

[0x7ffff000]...[0x80000000] 0x00000000

KERNEL DATA

<

C:\Users\Lu\Documents\Faculdade\AC II\Relatório7\programa8.asm successfully loaded

```
[0x00400000] 0x20081234 addi $8, $0, 4660      ; 11: addi $8, $0, 0x1234 #reg8=0x1234
[0x00400004] 0x00084400 sll $8, $8, 16         ; 12: sll $8,$8,16      #reg8=0x12340000
[0x00400008] 0x21085678 addi $8, $8, 22136     ; 13: addi $8,$8,0x5678 #reg8=0x12345678
[0x0040000c] 0x200900ff addi $9, $0, 255      ; 15: addi $9,$zero,0xFF #reg9=0xFF
[0x00400010] 0x00094e00 sll $9, $9, 24         ; 16: sll $9,$9,24     #coloca os FF no inicio
[0x00400014] 0x01284824 and $9, $9, $8        ; 17: and $9,$9,$8     #fazer um and para ter resultado requerido
[0x00400018] 0x00094e02 srl $9, $9, 24         ; 18: srl $9,$9,24     #reg9=0x12
```

>

For Help, press F1

PC=0x00400044 EPC=0x00000000 Cause=0x00000000

BARE DELAY BR DELAY LD

Digite aqui para pesquisar