

TP1: Máquina Virtual

1. Introdução

O trabalho prático propõe a criação de uma máquina virtual básica, que deve ser simulada pela interpretação da Máquina de Khattab. A criação deste emulador tem por objetivo simular o ciclo “Fetch-decode-execute”, realizado pelos processadores no nível de arquitetura do conjunto de instruções (ISA) dos computadores.

2. Implementação

O conjunto das instruções executadas pela Máquina Virtual está representado na tabela 1.

Cód	Símbolo	Operandos	Significado	Ação
01	LOAD	R M	Carrega Registrador	$Reg[R] \leftarrow Mem[M + PC]$
02	STORE	R M	Armazena Registrador	$Mem[M + PC] \leftarrow Reg[R]$
03	READ	R	Lê valor para registrador	$Reg[R] \leftarrow \text{“valor lido”}$
04	WRITE	R	Escreve conteúdo do registrador	“Imprime” $Reg[R]$
05	COPY	R1 R2	Copia registrador	$Reg[R1] \leftarrow Reg[R2] *$
06	NEG	R1	Inverte o sinal do registrador	$Reg[R1] \leftarrow -Reg[R1] *$
07	SUB	R1 R2	Subtrai dois registradores	$Reg[R1] \leftarrow Reg[R1] - Reg[R2] *$
08	ADD	R1 R2	Soma dois registradores	$Reg[R1] \leftarrow Reg[R1] + Reg[R2] *$
09	AND	R1 R2	AND (bit a bit) de dois registradores	$Reg[R1] \leftarrow Reg[R1] \text{ AND } Reg[R2] *$
10	OR	R1 R2	OR (bit a bit) de dois registradores	$Reg[R1] \leftarrow Reg[R1] \text{ OR } Reg[R2] *$
11	XOR	R1 R2	XOR (bit a bit) de dois registradores	$Reg[R1] \leftarrow Reg[R1] \text{ XOR } Reg[R2] *$
12	NOT	R1	NOT (bit a bit) de um registrador	$Reg[R1] \leftarrow \text{NOT } Reg[R1] *$
13	JMP	M	Desvio incondicional	$PC \leftarrow PC + M$
14	JZ	M	Desvia se zero	Se $PSW[zero]$, $PC \leftarrow PC + M$
15	JNZ	M	Desvia se não zero	Se $!PSW[zero]$, $PC \leftarrow PC + M$
16	JN	M	Desvia se negativo	Se $PSW[negativo]$, $PC \leftarrow PC + M$
17	JNN	M	Desvia se não negativo	Se $!PSW[negativo]$, $PC \leftarrow PC + M$
18	PUSH	R	Empilha valor do registrador	$SP \leftarrow SP - 1$ $Mem[SP] \leftarrow Reg[R]$
19	POP	R	Desempilha valor no registrador	$Reg[R] \leftarrow Mem[SP]$ $SP \leftarrow SP + 1$
20	CALL	M	Chamada de subrotina	$SP \leftarrow SP - 1$ $Mem[SP] \leftarrow PC$ $PC \leftarrow PC + M$
21	RET		Retorno de subrotina	$PC \leftarrow Mem[SP]$ $SP \leftarrow SP + 1$
22	HALT		Parada	

Tabela 1: Instruções da Máquina Virtual

Ao ser executado, é necessário que o emulador receba cinco **argumentos**. São eles, respectivamente: valor inicial do PC, valor inicial do SP, posição inicial do programa na memória, nome do arquivo de texto que contém o programa, e o modo de saída de dados.

O **quarto argumento**, ou seja, o arquivo que contém o programa a ser executado pela Máquina Virtual, deve conter uma instrução numérica por linha, de acordo com a tabela 1. Caso a instrução necessite de operandos, estes estarão presentes nas linhas seguintes, de forma que o acesso à memória seja contínuo.

Ao executar a Máquina Virtual, a memória e os registradores serão inicializados com zero. As instruções e os operandos contidos no arquivo texto serão lidos pelo programa, um a um, e armazenados nas posições de memória, a partir da posição fornecida pelo **terceiro argumento**. Dessa forma, a memória passa a conter as instruções que devem ser realizadas pela Máquina Virtual.

Para a criação da Máquina Virtual, foi implementada no corpo da função principal o comando “switch”, que permite ao emulador executar diferentes tarefas, executadas de acordo com a instrução lida pelo programa.

Cada caso presente no “switch” representa uma das 22 instruções presentes no conjunto de instruções da máquina e direciona o emulador para uma função, a qual faz a operação solicitada, seja ela uma adição, uma operação lógica de registradores ou até um desvio na leitura das instruções. Essas funções estão definidas no arquivo cabeçalho “func.h”, que contém outros comentários a respeito do objetivo de cada uma delas. A implementação dessas funções foi realizada no arquivo “func.c”. Além da criação destes dois arquivos, está presente ainda o arquivo “main.c”, no qual está implementada a função principal.

A compilação do programa foi feita por um Makefile. Para gerar o arquivo binário executável, basta entrar no diretório que contém o arquivo Makefile pelo terminal e digitar “make”. Assim, um executável chamado “emulador” será criado na pasta **/bin**. Para executá-lo pelo terminal, é necessário o comando:

```
./emulador [arg1] [arg2] [arg3] [arg4] [arg5]
```

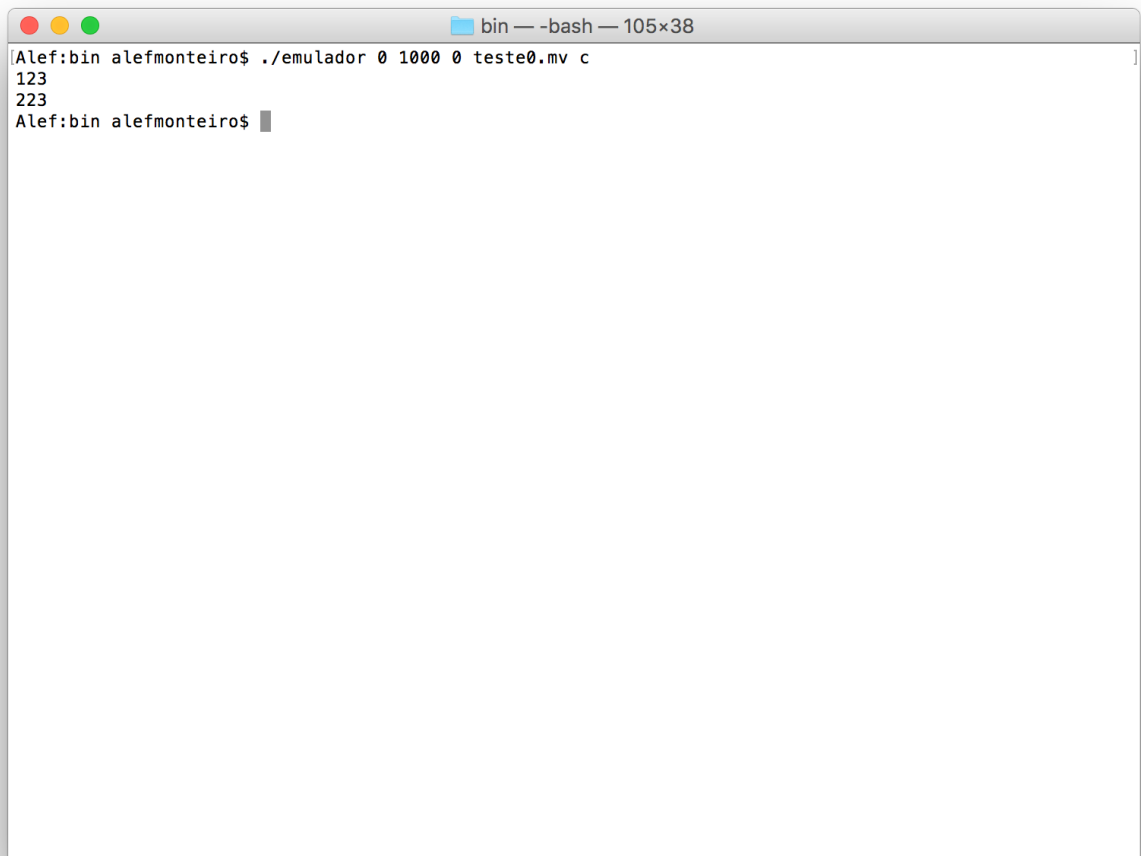
3. Testes

Quatro testes foram realizados na Máquina Virtual, onde cada um dos casos de teste representam um pequeno programa. Os arquivos de texto que contém os programas-teste estão no diretório **tp1_alefmon/tst** e estão nomeados como teste0.mv, teste1.mv, teste2.mv,

teste3.mv, respectivamente. O objetivo dos testes é verificar o funcionamento das diversas funções presentes no emulador.

3.1 Teste 0

O teste 0 foi realizado com o programa contido na especificação do TP.

A screenshot of a terminal window titled "bin — -bash — 105x38". The window shows the execution of a program. The prompt is "Alef:bin alefmonteiro\$". The user enters the command "./emulador 0 1000 0 teste0.mv c". The program outputs "123" and "223" on separate lines. The prompt returns to "Alef:bin alefmonteiro\$".

```
Alef:bin alefmonteiro$ ./emulador 0 1000 0 teste0.mv c
123
223
Alef:bin alefmonteiro$
```

```
bin — -bash — 105x38
Alef:bin alefmonteiro$ ./emulador 0 1000 0 teste0.mv v
Instrucao executada: READ
PC-> 0
SP-> 1000
PSW-> (0, 0)
Registradores: R0(0) R1(0) R2(0) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
123
Instrucao executada: LOAD
PC-> 2
SP-> 1000
PSW-> (0, 0)
Registradores: R0(123) R1(0) R2(0) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: ADD
PC-> 5
SP-> 1000
PSW-> (0, 0)
Registradores: R0(123) R1(100) R2(0) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: WRITE
PC-> 8
SP-> 1000
PSW-> (0, 0)
Registradores: R0(223) R1(100) R2(0) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
SAIDA--> 223
Instrucao executada: HALT
PC-> 10
SP-> 1000
PSW-> (0, 0)
Registradores: R0(223) R1(100) R2(0) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Alef:bin alefmonteiro$ █
```

3.2 Teste 1

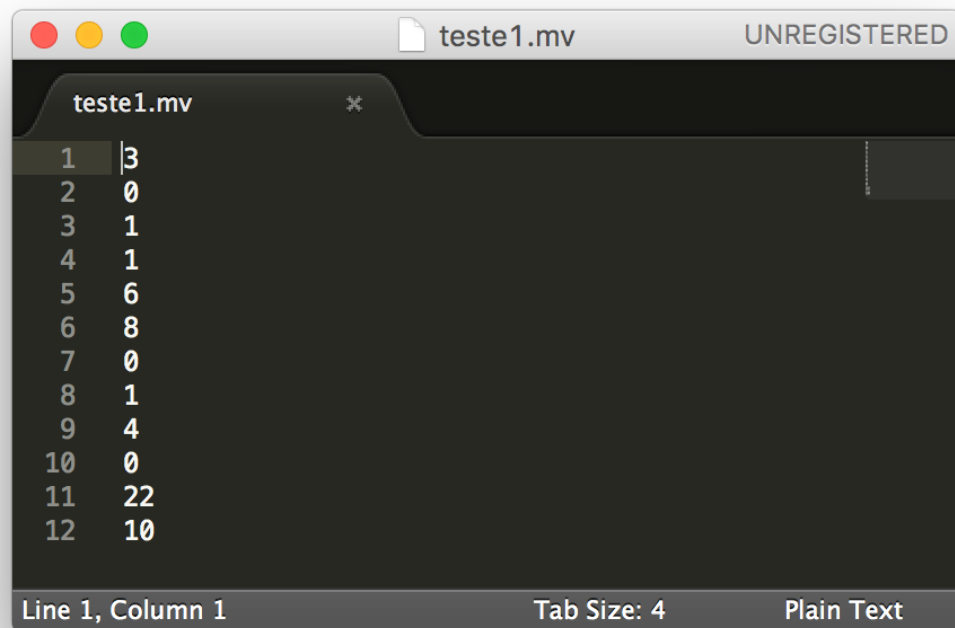
O teste 1 consiste em somar 10 no valor x inserido pelo usuário. Este teste verifica o funcionamento das instruções: READ, LOAD, ADD, WRITE, HALT.

Etapa 1: O valor x inserido pelo usuário é armazenado no registrador 0.

Etapa 2: O registrador 1 recebe o valor 10, presente na posição 11 da memória.

Etapa 3: Os registradores 0 e 1 são somados e o resultado é impresso na tela.

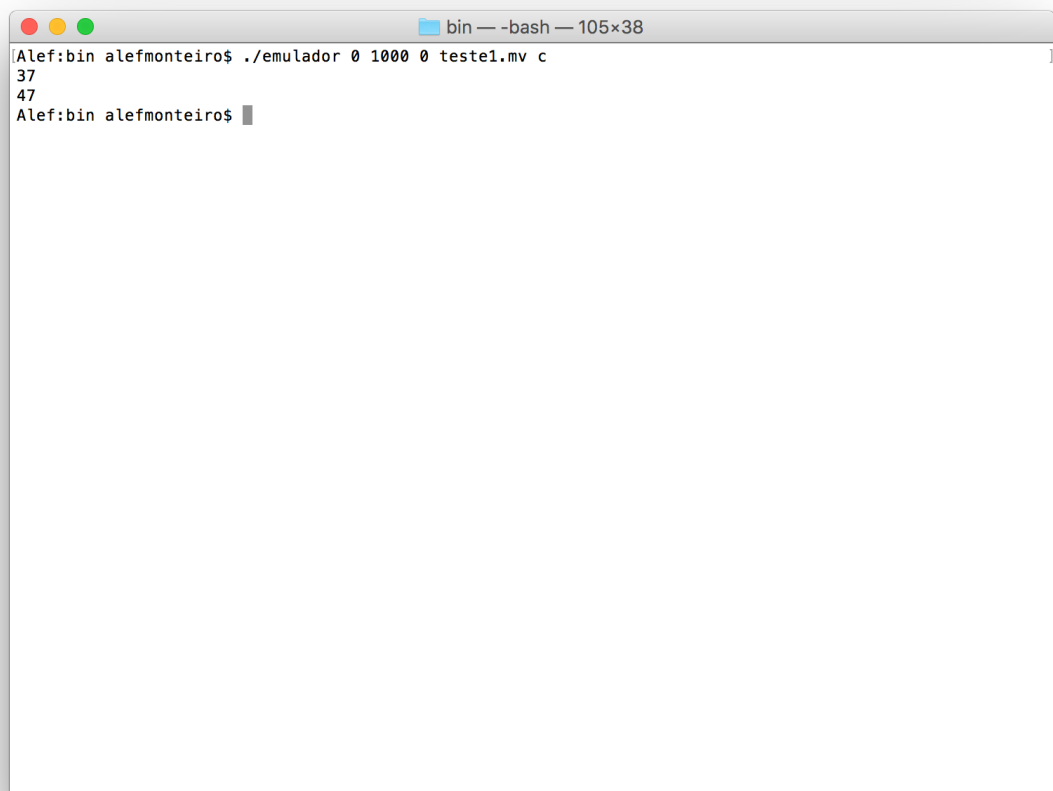
Etapa 4: A instrução Halt é executada e o programa é finalizado.



teste1.mv UNREGISTERED

```
1 |3
2 0
3 1
4 1
5 6
6 8
7 0
8 1
9 4
10 0
11 22
12 10
```

Line 1, Column 1 Tab Size: 4 Plain Text



bin — -bash — 105x38

```
Alef:bin alefmonteiro$ ./emulador 0 1000 0 teste1.mv c
37
47
Alef:bin alefmonteiro$
```

```
bin — -bash — 105x38
Alef:bin alefmonteiro$ ./emulador 0 1000 0 teste1.mv v
Instrucao executada: READ
PC-> 0
SP-> 1000
PSW-> (0, 0)
Registadores: R0(0) R1(0) R2(0) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
37
Instrucao executada: LOAD
PC-> 2
SP-> 1000
PSW-> (0, 0)
Registadores: R0(37) R1(0) R2(0) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: ADD
PC-> 5
SP-> 1000
PSW-> (0, 0)
Registadores: R0(37) R1(10) R2(0) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: WRITE
PC-> 8
SP-> 1000
PSW-> (0, 0)
Registadores: R0(47) R1(10) R2(0) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
SAIDA->-> 47
Instrucao executada: HALT
PC-> 10
SP-> 1000
PSW-> (0, 0)
Registadores: R0(47) R1(10) R2(0) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Alef:bin alefmonteiro$
```

3.3 Teste 2

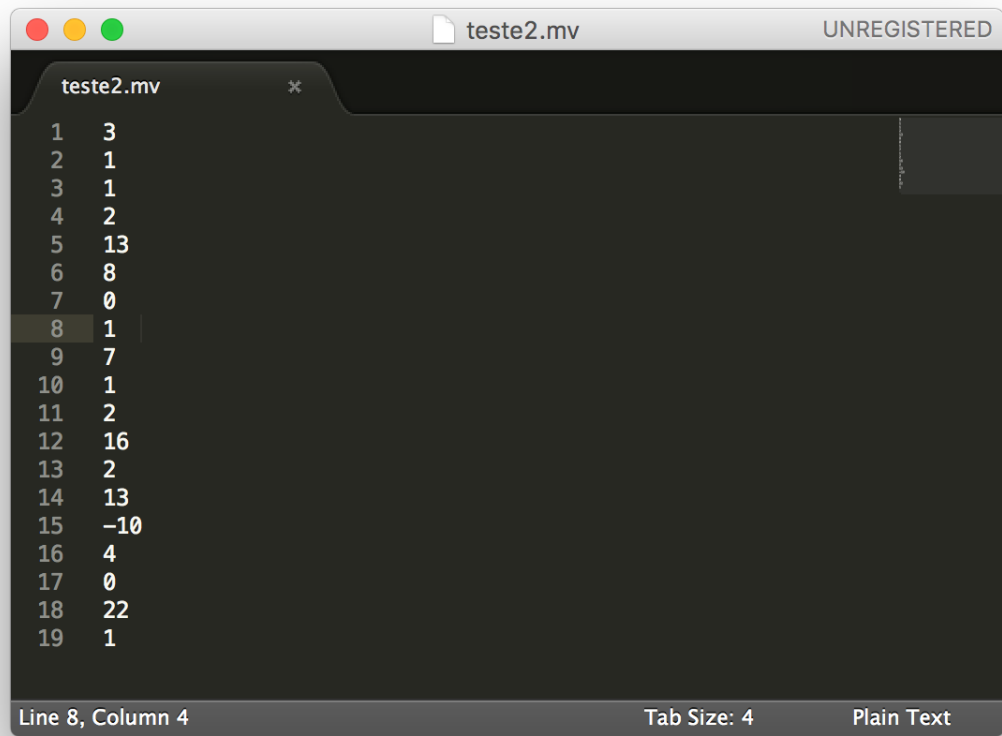
O teste 2, por sua vez, consiste em um programa que calcula o somatório de 0 até n, onde n é fornecido pelo usuário. Este teste verifica o funcionamento das instruções: READ, LOAD, ADD, SUB, JN, JMP, WRITE, HALT.

Etapa 1: O valor de n é armazenado no registrador 1.

Etapa 2: O valor contido no registrador 1 (no caso, valor é igual a “n”) é adicionado ao registrador 0.

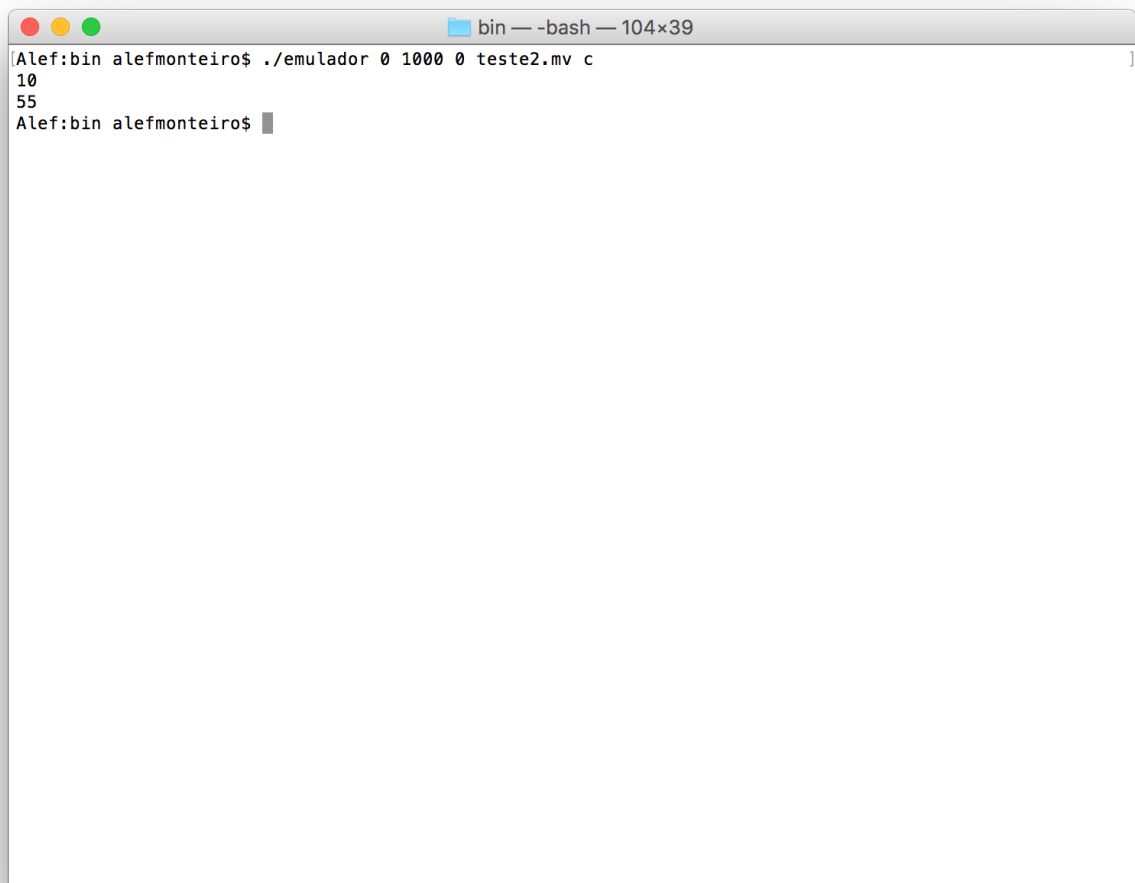
Etapa 3: O registrador 2, que recebe o valor (1), será utilizado para a decrementar o valor de n em (-1).

Etapa 4: Por fim, um desvio condicional foi implementado. Caso o valor do registrador 1 seja menor que zero, o programa imprime o valor do somatório e é finalizado. Caso contrário, o programa regressa à etapa 1.



```
teste2.mv
1 3
2 1
3 1
4 2
5 13
6 8
7 0
8 1
9 7
10 1
11 2
12 16
13 2
14 13
15 -10
16 4
17 0
18 22
19 1
```

Line 8, Column 4 Tab Size: 4 Plain Text



```
bin -- -bash -- 104x39
[Alef:bin alefmonteiro$ ./emulador 0 1000 0 teste2.mv c
10
55
Alef:bin alefmonteiro$ ]
```

```
bin — -bash — 131x50

Alef:bin alefmonteiro$ ./emulador 0 1000 0 teste2.mv v
Instrucao executada: READ
PC-> 0
SP-> 1000
PSW-> (0, 0)
Registradores: R0(0) R1(0) R2(0) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
3
Instrucao executada: LOAD
PC-> 2
SP-> 1000
PSW-> (0, 0)
Registradores: R0(0) R1(3) R2(0) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: ADD
PC-> 5
SP-> 1000
PSW-> (0, 0)
Registradores: R0(0) R1(3) R2(1) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: SUB
PC-> 8
SP-> 1000
PSW-> (0, 0)
Registradores: R0(3) R1(3) R2(1) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: JN
PC-> 11
SP-> 1000
PSW-> (0, 0)
Registradores: R0(3) R1(2) R2(1) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: JMP
PC-> 13
SP-> 1000
PSW-> (0, 0)
Registradores: R0(3) R1(2) R2(1) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: ADD
PC-> 5
SP-> 1000
PSW-> (0, 0)
Registradores: R0(3) R1(2) R2(1) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: SUB
PC-> 8
SP-> 1000
PSW-> (0, 0)
Registradores: R0(5) R1(2) R2(1) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: JN
PC-> 11
SP-> 1000
PSW-> (0, 0)
Registradores: R0(5) R1(1) R2(1) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: JMP
PC-> 13

bin — -bash — 131x50

PSW-> (0, 0)
Registradores: R0(5) R1(1) R2(1) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: JMP
PC-> 13
SP-> 1000
PSW-> (0, 0)
Registradores: R0(5) R1(1) R2(1) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: ADD
PC-> 5
SP-> 1000
PSW-> (0, 0)
Registradores: R0(5) R1(1) R2(1) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: SUB
PC-> 8
SP-> 1000
PSW-> (0, 0)
Registradores: R0(6) R1(1) R2(1) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: JN
PC-> 11
SP-> 1000
PSW-> (1, 0)
Registradores: R0(6) R1(0) R2(1) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: JMP
PC-> 13
SP-> 1000
PSW-> (1, 0)
Registradores: R0(6) R1(0) R2(1) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: ADD
PC-> 5
SP-> 1000
PSW-> (1, 0)
Registradores: R0(6) R1(0) R2(1) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: SUB
PC-> 8
SP-> 1000
PSW-> (0, 0)
Registradores: R0(6) R1(0) R2(1) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: JN
PC-> 11
SP-> 1000
PSW-> (0, 1)
Registradores: R0(6) R1(-1) R2(1) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: WRITE
PC-> 15
SP-> 1000
PSW-> (0, 1)
Registradores: R0(6) R1(-1) R2(1) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
SAIDA-->> 6
Instrucao executada: HALT
PC-> 17
```


3.4 Teste 3

O teste 3 foi utilizado para efetuar operações lógicas com dois valores, inseridos pelo usuário. Este teste verifica o funcionamento das instruções: READ, COPY, AND, OR, XOR, NOT, WRITE, HALT.

Etapa 1: O programa pede ao usuário que insira dois valores, armazenados nos registradores 0 e 1, respectivamente.

Etapa 2: O programa copia o valor do registrador 0 para o registrador 2. Também é feita a cópia do valor presente no registrador 1 para o registrador 3.

Etapa 3: A operação AND é feita com os registradores 2 e 3 e o resultado é impresso.

Etapa 4: Registrador 2 recebe a cópia do valor do registrador 0 novamente.

Etapa 5: A operação OR é feita com os registradores 2 e 3 e o resultado é impresso.

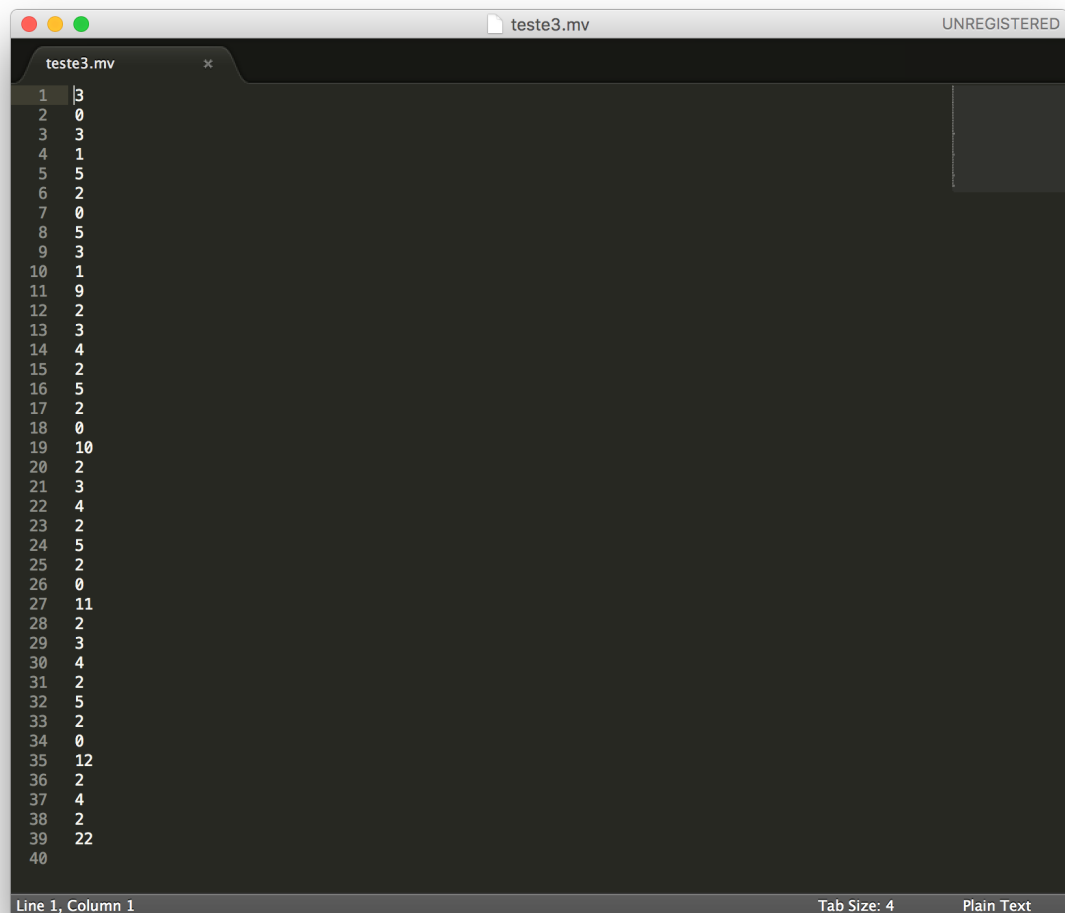
Etapa 6: Registrador 2 recebe a cópia do valor do registrador 0 novamente.

Etapa 7: A operação XOR é feita com os registradores 2 e 3 e o resultado é impresso.

Etapa 8: Registrador 2 recebe a cópia do valor do registrador 0 novamente.

Etapa 9: A operação NOT é feita com o registrador 2 e o resultado é impresso.

Etapa 10: A instrução Halt é executada e o programa é finalizado.



```
1 |3
2 0
3 3
4 1
5 5
6 2
7 0
8 5
9 3
10 1
11 9
12 2
13 3
14 4
15 2
16 5
17 2
18 0
19 10
20 2
21 3
22 4
23 2
24 5
25 2
26 0
27 11
28 2
29 3
30 4
31 2
32 5
33 2
34 0
35 12
36 2
37 4
38 2
39 22
40
```

```
bin — -bash — 103x33
Alef:bin alefmonteiro$ ./emulador 0 1000 0 teste3.mv c
2 4
0
6
6
-3
Alef:bin alefmonteiro$
```

```
bin — -bash — 131x50
Alef:bin alefmonteiro$ ./emulador 0 1000 0 teste3.mv v
Instrucao executada: READ
PC-> 0
SP-> 1000
PSW-> (0, 0)
Registradores: R0(0) R1(0) R2(0) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
2
Instrucao executada: READ
PC-> 2
SP-> 1000
PSW-> (0, 0)
Registradores: R0(2) R1(0) R2(0) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
4
Instrucao executada: COPY
PC-> 4
SP-> 1000
PSW-> (0, 0)
Registradores: R0(2) R1(4) R2(0) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: COPY
PC-> 7
SP-> 1000
PSW-> (0, 0)
Registradores: R0(2) R1(4) R2(2) R3(0) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: AND
PC-> 10
SP-> 1000
PSW-> (0, 0)
Registradores: R0(2) R1(4) R2(2) R3(4) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: WRITE
PC-> 13
SP-> 1000
PSW-> (1, 0)
Registradores: R0(2) R1(4) R2(0) R3(4) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
SAIDA->-> 0
Instrucao executada: COPY
PC-> 15
SP-> 1000
PSW-> (1, 0)
Registradores: R0(2) R1(4) R2(0) R3(4) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: OR
PC-> 18
SP-> 1000
PSW-> (0, 0)
Registradores: R0(2) R1(4) R2(2) R3(4) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: WRITE
PC-> 21
SP-> 1000
PSW-> (0, 0)
Registradores: R0(2) R1(4) R2(6) R3(4) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
SAIDA->-> 6
```

```
bin -- -bash -- 131x50
Registadores: R0(2) R1(4) R2(0) R3(4) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: OR
PC-> 18
SP-> 1000
PSW-> (0, 0)
Registadores: R0(2) R1(4) R2(2) R3(4) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: WRITE
PC-> 21
SP-> 1000
PSW-> (0, 0)
Registadores: R0(2) R1(4) R2(6) R3(4) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
SAIDA->-> 6
Instrucao executada: COPY
PC-> 23
SP-> 1000
PSW-> (0, 0)
Registadores: R0(2) R1(4) R2(6) R3(4) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: XOR
PC-> 26
SP-> 1000
PSW-> (0, 0)
Registadores: R0(2) R1(4) R2(2) R3(4) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: WRITE
PC-> 29
SP-> 1000
PSW-> (0, 0)
Registadores: R0(2) R1(4) R2(6) R3(4) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
SAIDA->-> 6
Instrucao executada: COPY
PC-> 31
SP-> 1000
PSW-> (0, 0)
Registadores: R0(2) R1(4) R2(6) R3(4) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: NOT
PC-> 34
SP-> 1000
PSW-> (0, 0)
Registadores: R0(2) R1(4) R2(2) R3(4) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Instrucao executada: WRITE
PC-> 36
SP-> 1000
PSW-> (0, 0)
Registadores: R0(2) R1(4) R2(-3) R3(4) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
SAIDA->-> -3
Instrucao executada: HALT
PC-> 38
SP-> 1000
PSW-> (0, 0)
Registadores: R0(2) R1(4) R2(-3) R3(4) R4(0) R5(0) R6(0) R7(0) R8(0) R9(0) R10(0) R11(0) R12(0) R13(0) R14(0) R15(0)
Alef:bin alefmonteiros$
```

4. Conclusão

A implementação do trabalho exigiu bastante esforço e dedicação. A tarefa, que no início parecia impossível, mostrou a importância do trabalho prático no aprendizado do aluno, uma vez que ele permite a verificação e a aplicação do conhecimento obtido em sala de aula. A montagem da Máquina Virtual exige que o aluno entenda as diversas instruções executadas e, dessa maneira, compreenda o funcionamento do ciclo de dados feito pelo processador.

A maior dificuldade encontrada foi o entendimento geral da função de cada registrador específico. Outro problema foi a compreensão de como o conteúdo da memória seria buscado pelo programa, mas foi solucionado pelo uso do "switch". Após a verificação de como o programa iria fluir e executar as instruções, o trabalho transcorreu sem maiores problemas.

O resultado, sem dúvidas, foi uma compreensão maior e mais detalhada do conteúdo abordado pelo professor. Além disso, a tarefa mostrou a importância da disciplina Software Básico para a formação acadêmica do estudante de computação.

5. Referências

[1] TANENBAUM, Andrew S.; AUSTIN, Todd. Organização Estruturada de Computadores. 6. ed. : Pearson, 2013.