

Relatório - Laboratório 04

Disciplina: INE5411 – Organização de Computadores I

Integrantes: Lucas Pastre de Souza e Rodrigo Martins dos Santos

Introdução:

O presente relatório busca mostrar o funcionamento dos exercícios 1 e 2, referentes à quarta atividade de laboratório da disciplina Organização e Arquitetura de Computadores do curso de Ciências da Computação na Universidade Federal de Santa Catarina (UFSC).

Exercício 1:

O objetivo do exercício 1 era percorrer uma matriz 16x16 linha após linha, armazenando na memória os valores 0 a 255 na ordem correta. Veja abaixo as imagens do código e também dos valores armazenados na memória.

```
1  .data
2      matriz:.space 1024      # aloca espaço para 256 words
3      espaco:.asciiz " "
4      quebra:.asciiz "\n"
5
6  .text
7      move $s0, $zero
8      li $s1, 16
9      li $s7, 0
10
11     loop_linha:
12         beq $s0, $s1, fim
13         move $s2, $zero # reseta o índice da coluna
14
15     loop_coluna:
16         li $s3, 16
17         beq $s2, $s3, proxima_linha
18
19         # calculo do endereço de matriz[linha][coluna]
20         # endereço do elemento = endereço base + (linha x num de colunas + coluna) x 4 bytes
21
22         mul $s4, $s0, $s3      # linha x num de colunas
23         add $s5, $s4, $s2      # (linha x num de colunas) + coluna
24
25         #deslocamento de 2 p/ esquerda p/ multiplicar por 4
26         sll $s6, $s5, 2      # ((linha x num de colunas) + coluna) x 4
27
28         la $t0, matriz      # carrega endereço base
29         add $t1, $t0, $s6    # endereço base + deslocamento
30         sw $s7, 0($t1)      # armazena o valor na memória
31
32         li $v0, 1
33         move $a0, $s7
34         syscall
35         li $v0, 4
36         la $a0, espaco
37         syscall
38
39         addi $s2, $s2, 1
40         addi $s7, $s7, 1
41
42     j loop_coluna
43
```

```

44     proxima_linha:
45         li $v0, 4
46         la $a0, quebra
47         syscall
48
49         addi $s0, $s0, 1
50
51     j loop_linha
52
53 fim:
54     li $v0, 10      # encerra o programa
55     syscall
56
57

```

Address	Value (+0)	Value (+4)	Value (+8)	Value (+12)	Value (+16)	Value (+20)	Value (+24)	Value (+28)
268500952	0	1	2	3	4	5	6	7
268501024	8	9	10	11	12	13	14	15
268501056	16	17	18	19	20	21	22	23
268501088	24	25	26	27	28	29	30	31
268501120	32	33	34	35	36	37	38	39
268501152	40	41	42	43	44	45	46	47
268501184	48	49	50	51	52	53	54	55
268501216	56	57	58	59	60	61	62	63
268501248	64	65	66	67	68	69	70	71
268501280	72	73	74	75	76	77	78	79
268501312	80	81	82	83	84	85	86	87
268501344	88	89	90	91	92	93	94	95
268501376	96	97	98	99	100	101	102	103
268501408	104	105	106	107	108	109	110	111

Address	Value (+0)	Value (+4)	Value (+8)	Value (+12)	Value (+16)	Value (+20)	Value (+24)	Value (+28)
268501312	80	81	82	83	84	85	86	87
268501344	88	89	90	91	92	93	94	95
268501376	96	97	98	99	100	101	102	103
268501408	104	105	106	107	108	109	110	111
268501440	112	113	114	115	116	117	118	119
268501472	120	121	122	123	124	125	126	127
268501504	128	129	130	131	132	133	134	135
268501536	136	137	138	139	140	141	142	143
268501568	144	145	146	147	148	149	150	151
268501600	152	153	154	155	156	157	158	159
268501632	160	161	162	163	164	165	166	167
268501664	168	169	170	171	172	173	174	175
268501696	176	177	178	179	180	181	182	183
268501728	184	185	186	187	188	189	190	191

Address	Value (+0)	Value (+4)	Value (+8)	Value (+12)	Value (+16)	Value (+20)	Value (+24)	Value (+28)
268501568	144	145	146	147	148	149	150	151
268501600	152	153	154	155	156	157	158	159
268501632	160	161	162	163	164	165	166	167
268501664	168	169	170	171	172	173	174	175
268501696	176	177	178	179	180	181	182	183
268501728	184	185	186	187	188	189	190	191
268501760	192	193	194	195	196	197	198	199
268501792	200	201	202	203	204	205	206	207
268501824	208	209	210	211	212	213	214	215
268501856	216	217	218	219	220	221	222	223
268501888	224	225	226	227	228	229	230	231
268501920	232	233	234	235	236	237	238	239
268501952	240	241	242	243	244	245	246	247
268501984	248	249	250	251	252	253	254	255

```
Messages Run I/O
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47
48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63
64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79
80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95
96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111
112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127
128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143
144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159
160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175
176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191
192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207
208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223
224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239
240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255

-- program is finished running --
```

Exercício 2:

O objetivo do exercício 2 era percorrer uma matriz 16x16 coluna após coluna, armazenando na memória os valores 0 a 255 na ordem correta. Confira abaixo as imagens do código e dos valores armazenados na memória.

```
1 .data
2     matriz:.space 1024      # aloca espaço para 256 words
3     espaco:.asciiz " "
4     quebra:.asciiz "\n"
5
6 .text
7     move $s0, $zero
8     li $s1, 16
9     li $s7, 0      # contador valor
10
11     loop_linha:
12         beq $s0, $s1, print
13         move $s2, $zero # reseta o índice da coluna
14
15     loop_coluna:
16         li $s3, 16
17         beq $s2, $s3, proxima_linha
18
19         # calculo do endereço de matriz[linha][coluna]
20
21         # endereço do elemento = endereço base + (coluna x num de linhas + linha) x 4 bytes
22
23         mul $s4, $s2, $s3      # coluna x num de linhas
24         add $s5, $s4, $s0      # (coluna x num de linhas) + linha
25
26         #deslocamento de 2 p/ esquerda p/ multiplicar por 4
27         sll $s6, $s5, 2      # ((coluna x num de linhas) + linha) x 4
28
29         la $t0, matriz        # carrega endereço base
30         add $t1, $t0, $s6      # endereço base + deslocamento
31         sw $s7, 0($t1)        # armazena o valor na memória
32
33         addi $s2, $s2, 1
34         addi $s7, $s7, 1
35
36         j loop_coluna
37
38     proxima_linha:
39         addi $s0, $s0, 1
40
41         j loop_linha
42
```

```

43     print:
44         la $t0, matriz
45         li $t2, 16
46         move $t3, $zero
47         li $t6, 16
48         li $t5, 1
49
50     loop_print:
51         beq $t3, $t2, proxima_linha_print
52         lw $t4, 0($t0)
53
54         li $v0, 1
55         move $a0, $t4
56         syscall
57         li $v0, 4
58         la $a0, espaco
59         syscall
60
61         addi $t0, $t0, 4      # incrementa para acessar o proximo elemento dessa linha
62         addi $t3, $t3, 1
63         j loop_print

```

```

64
65     proxima_linha_print:
66         beq $t5, $t6, end
67         li $v0, 4
68         la $a0, quebra
69         syscall
70
71         move $t3, $zero
72         addi $t5, $t5, 1
73
74         j loop_print
75
76     end:
77         li $v0, 10          # encerra o programa
78         syscall
79

```

Address	Value (+0)	Value (+4)	Value (+8)	Value (+12)	Value (+16)	Value (+20)	Value (+24)	Value (+28)
268500992	0	16	32	48	64	80	96	112
268501024	128	144	160	176	192	208	224	240
268501056	1	17	33	49	65	81	97	113
268501088	129	145	161	177	193	209	225	241
268501120	2	18	34	50	66	82	98	114
268501152	130	146	162	178	194	210	226	242
268501184	3	19	35	51	67	83	99	115
268501216	131	147	163	179	195	211	227	243
268501248	4	20	36	52	68	84	100	116
268501280	132	148	164	180	196	212	228	244
268501312	5	21	37	53	69	85	101	117
268501344	133	149	165	181	197	213	229	245
268501376	6	22	38	54	70	86	102	118
268501408	134	150	166	182	198	214	230	246

Address	Value (+0)	Value (+4)	Value (+8)	Value (+12)	Value (+16)	Value (+20)	Value (+24)	Value (+28)
268501568	9	25	41	57	73	89	105	121
268501600	137	153	169	185	201	217	233	249
268501632	10	26	42	58	74	90	106	122
268501664	138	154	170	186	202	218	234	250
268501696	11	27	43	59	75	91	107	123
268501728	139	155	171	187	203	219	235	251
268501760	12	28	44	60	76	92	108	124
268501792	140	156	172	188	204	220	236	252
268501824	13	29	45	61	77	93	109	125
268501856	141	157	173	189	205	221	237	253
268501888	14	30	46	62	78	94	110	126
268501920	142	158	174	190	206	222	238	254
268501952	15	31	47	63	79	95	111	127
268501984	143	159	175	191	207	223	239	255

Messages	Run I/O
0	16 32 48 64 80 96 112 128 144 160 176 192 208 224 240
1	17 33 49 65 81 97 113 129 145 161 177 193 209 225 241
2	18 34 50 66 82 98 114 130 146 162 178 194 210 226 242
3	19 35 51 67 83 99 115 131 147 163 179 195 211 227 243
4	20 36 52 68 84 100 116 132 148 164 180 196 212 228 244
5	21 37 53 69 85 101 117 133 149 165 181 197 213 229 245
6	22 38 54 70 86 102 118 134 150 166 182 198 214 230 246
7	23 39 55 71 87 103 119 135 151 167 183 199 215 231 247
8	24 40 56 72 88 104 120 136 152 168 184 200 216 232 248
9	25 41 57 73 89 105 121 137 153 169 185 201 217 233 249
10	26 42 58 74 90 106 122 138 154 170 186 202 218 234 250
11	27 43 59 75 91 107 123 139 155 171 187 203 219 235 251
12	28 44 60 76 92 108 124 140 156 172 188 204 220 236 252
13	29 45 61 77 93 109 125 141 157 173 189 205 221 237 253
14	30 46 62 78 94 110 126 142 158 174 190 206 222 238 254
15	31 47 63 79 95 111 127 143 159 175 191 207 223 239 255
	-- program is finished running --