

ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO
DISCIPLINA: LABORATÓRIO DE PROCESSADORES- PCS3732
1º QUADRIMESTRE/2021



Aula 4
27 de Maio de 2021

GRUPO 10

Arthur Pires da Fonseca
Bruno José Mório
Iago Soriano Roque Monteiro

NUSP: 10773096
NUSP: 10336852
NUSP: 8572921

Sumário

Exercício 4.5.1 - Assignments with operands in memory	3
Pós fixado	3
Pré fixado	4
Exercício 4.5.2 - Loads and stores	6
Pós fixado	6
Pré fixado	6
Exercício 4.5.3 - Array Assignment	7
Exercício 4.5.4 - Arrays and pointers	9
Índice	9
Pointers	11
Exercício 4.5.5 - The Fibonacci sequence	13
Exercício 4.5.6 - The nth Fibonacci number	15
Apêndice	17
Exercício 4.5.1	17
Pré indexado	17
Pós indexado	17
Exercício 4.5.2	18
Pós indexado	18
Pré indexado	18
Exercício 4.5.3	18
Exercício 4.5.4	20
4.5.4.A	20
4.5.4.B	20
Exercício 4.5.5	21
Exercício 4.5.6	22

Exercício 4.5.1 - Assignments with operands in memory

Pós fixado

- Início do processo

```
Register group: general-
r0      0x1      1      r1      0x1ffff8 2097144      r2      0xffffffff -1
r3      0xaa00   43520   r4      0x1      1      r5      0x1ffff8 2097144
r6      0x0      0      r7      0x0      0      r8      0x0      0
r9      0x0      0      r10     0x200100 2097408     r11     0x0      0
r12     0x1fffc 2097100   sp      0x1ffff8 2097144     lr      0x81fc 33276
pc      0x8218 33304     fps     0x0      0      cpsr    0x60000013 1610612755

B+> 4      MOV r1, #0x2
5      LDR r2, =arr
6      ADD r4, r2, #5
7      LDR r3, [r4], #0
8      ADD r0, r1, r3
9      SWI 0x0
10     arr:
11     .word 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3
12
13
14
```

- Antes do load

```
Register group: general-
r0      0x1      1      r1      0x2      2      r2      0x8230 33328
r3      0xaa00   43520   r4      0x8235 33333   r5      0x1ffff8 2097144
r6      0x0      0      r7      0x0      0      r8      0x0      0
r9      0x0      0      r10     0x200100 2097408     r11     0x0      0
r12     0x1fffc 2097100   sp      0x1ffff8 2097144     lr      0x81fc 33276
pc      0x8224 33316     fps     0x0      0      cpsr    0x60000013 1610612755

B+> 4      MOV r1, #0x2
5      LDR r2, =arr
6      ADD r4, r2, #5
7      LDR r3, [r4], #0
8      ADD r0, r1, r3
9      SWI 0x0
10     arr:
11     .word 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3
12
13
14
```

- Após o load apenas o valor de r3 sofre alteração. O registrador r4 permanece o mesmo.

```
Register group: general-
r0      0x1      1      r1      0x2      2      r2      0x8230 33328
r3      0x1000000 16777216 r4      0x8235 33333   r5      0x1ffff8 2097144
r6      0x0      0      r7      0x0      0      r8      0x0      0
r9      0x0      0      r10     0x200100 2097408     r11     0x0      0
r12     0x1fffc 2097100   sp      0x1ffff8 2097144     lr      0x81fc 33276
pc      0x8228 33320     fps     0x0      0      cpsr    0x60000013 1610612755

B+> 4      MOV r1, #0x2
5      LDR r2, =arr
6      ADD r4, r2, #5
7      LDR r3, [r4], #0
8      ADD r0, r1, r3
9      SWI 0x0
10     arr:
11     .word 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3
12
13
14
```

- Ao final, os registradores r1 e r2 são somados.

```

Activities Terminal mai 27 14:40
bruno@bruno-340XAA-350XA: ~/Documents/Poli/gcc-arm

Register group: general
r0 0x1000002 16777218 r1 0x2 2 r2 0x8230 33328
r3 0x1000000 16777216 r4 0x8235 33333 r5 0x1ffff8 2097144
r6 0x0 0 r7 0x0 0 r8 0x0 0
r9 0x0 0 r10 0x200100 2097408 r11 0x0 0
r12 0x1fffc 2097100 sp 0x1ffff8 2097144 lr 0x81fc 33276
pc 0x822c 33324 fps 0x0 0 cpsr 0x60000013 1610612755

B+ 4 MOV r1, #0x2
5 LDR r2, =arr
6 ADD r4, r2, #5
7 LDR r3, [r4], #0
8 ADD r0, r1, r3
> 9 SWI 0x0
10 arr:
11 .word 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3
12
13
14

```

Pré fixado

- Início do processo

```

Activities Terminal mai 27 14:43
bruno@bruno-340XAA-350XA: ~/Documents/Poli/gcc-arm

Register group: general
r0 0x1 1 r1 0x1ffff8 2097144 r2 0xffffffff -1
r3 0xa9fc 43516 r4 0x1 1 r5 0x1ffff8 2097144
r6 0x0 0 r7 0x0 0 r8 0x0 0
r9 0x0 0 r10 0x200100 2097408 r11 0x0 0
r12 0x1fffc 2097100 sp 0x1ffff8 2097144 lr 0x81fc 33276
pc 0x8218 33304 fps 0x0 0 cpsr 0x60000013 1610612755

B+ 4 MOV r1, #0x2
5 LDR r2, =arr
6 LDR r3, [r2, #5]
7 ADD r0, r1, r3
8 SWI 0x0
9
10 arr:
11 .word 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3
12
13
14

```

- Antes do load

```

Activities Terminal mai 27 14:44
bruno@bruno-340XAA-350XA: ~/Documents/Poli/gcc-arm

Register group: general
r0 0x1 1 r1 0x2 2 r2 0x822c 33324
r3 0xa9fc 43516 r4 0x1 1 r5 0x1ffff8 2097144
r6 0x0 0 r7 0x0 0 r8 0x0 0
r9 0x0 0 r10 0x200100 2097408 r11 0x0 0
r12 0x1fffc 2097100 sp 0x1ffff8 2097144 lr 0x81fc 33276
pc 0x8220 33312 fps 0x0 0 cpsr 0x60000013 1610612755

B+ 4 MOV r1, #0x2
5 LDR r2, =arr
> 6 LDR r3, [r2, #5]
7 ADD r0, r1, r3
8 SWI 0x0
9
10 arr:
11 .word 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3
12
13
14

```

- Após o load e após a soma

```
Activities Terminal mai 27 14:44
bruno@bruno-340XAA-350XA: ~/Documents/Poli/gcc-arm

Register group: general
r0 0x1000002 16777218 r1 0x2 2 r2 0x822c 33324
r3 0x1000000 16777216 r4 0x1 1 r5 0x1ffff8 2097144
r6 0x0 0 r7 0x0 0 r8 0x0 0
r9 0x0 0 r10 0x200100 2097408 r11 0x0 0
r12 0x1ffffc 2097100 sp 0x1ffff8 2097144 lr 0x81fc 33276
pc 0x8228 33320 fps 0x0 0 cpsr 0x60000013 1610612755

4 MOV r1, #0x2
5 LDR r2, =arr
6 LDR r3, [r2, #5]
7 ADD r0, r1, r3
> 8 SWI 0x0
9
10 arr:
11 .word 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3
12
13
14
```

Exercício 4.5.2 - Loads and stores

Pós fixado

- Resultado após execução, pode-se ver os valores atualizados do array na parte debaixo da imagem.

```
Activities Terminal mai 27 15:11
bruno@bruno-340XAA-350XA: ~/Documents/Polli/gcc-arm

Register group: general
r0      0x3      3          r1      0x2      2          r2      0x8238  33336
r3      0x8260  33376       r4      0x1      1          r5      0x1ffff8 2097144
r6      0x0      0          r7      0x0      0          r8      0x0      0
r9      0x0      0          r10     0x200100 2097408   r11     0x0      0
r12     0x1ffffc 2097100   sp      0x1ffff8 2097144   lr      0x81fc   33276
pc      0x8234   33332     fps      0x0      0          cpsr    0x60000013 1610612755

B+
14      MOV r1, #0x2        @ y qualquer
15      ADR r2, arr         @ salva end inicial em r2
16      ADD r3, r2, #20     @ salva o endereço de array[5] em r3
17      LDR r4, [r3], #0    @ salva o valor de array[5] em r4
18      ADD r0, r1, r4      @ salva array[5] + y em r0
19      ADD r3, r2, #40     @ salva o endereço de array[10] em r3
20      STR r0, [r3]        @ salve o valor de r0 no endereço de r3
21      SWI 0x0
22
23      arr:
24      .word 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3

sim process 42 In: main
(gdb) s
(gdb) s
(gdb) s
(gdb) s
(gdb) s
(gdb) s
(gdb) x/16 0x8238
0x8238 <arr>: 0x00000000 0x00000001 0x00000002 0x00000003
0x8248 <arr+16>: 0x00000000 0x00000001 0x00000002 0x00000003
0x8258 <arr+32>: 0x00000000 0x00000001 0x00000003 0x00000003
0x8268 <arr+48>: 0x00000000 0x00000001 0x00000002 0x00000003
(gdb) □
```

Pré fixado

- Resultado após execução, pode-se ver os valores atualizados do array na parte debaixo da imagem.

```
Activities Terminal mai 27 15:15
bruno@bruno-340XAA-350XA: ~/Documents/Polli/gcc-arm

Register group: general
r0      0x3      3          r1      0x2      2          r2      0x8230  33328
r3      0xa9fc  43516       r4      0x1      1          r5      0x1ffff8 2097144
r6      0x0      0          r7      0x0      0          r8      0x0      0
r9      0x0      0          r10     0x200100 2097408   r11     0x0      0
r12     0x1ffffc 2097100   sp      0x1ffff8 2097144   lr      0x81fc   33276
pc      0x822c   33324     fps      0x0      0          cpsr    0x60000013 1610612755

B+
14      MOV r1, #0x2        @ y qualquer
15      ADR r2, arr         @ salva end inicial em r2
16      LDR r4, [r2, #20]   @ salva o valor de array[5] em r4
17      ADD r0, r1, r4      @ salva array[5] + y em r0
18      STR r0, [r2, $40]   @ salve o valor de r0 no endereço de array[10]
19      SWI 0x0
20
21      arr:
22      .word 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3

sim process 42 In: main
Current language: auto; currently asm
(gdb) s
(gdb) s
(gdb) s
(gdb) s
(gdb) s
(gdb) s
(gdb) x/16 0x8230
0x8230 <arr>: 0x00000000 0x00000001 0x00000002 0x00000003
0x8240 <arr+16>: 0x00000000 0x00000001 0x00000002 0x00000003
0x8250 <arr+32>: 0x00000000 0x00000001 0x00000003 0x00000003
0x8260 <arr+48>: 0x00000000 0x00000001 0x00000002 0x00000003
(gdb) □
```

Exercício 4.5.3 - Array Assignment

$a[] = \{0xff00ff00, \dots\}$ e $b[] = \{0x12345678, 0x87654321, 0xff00ff00, \dots\}$, em que ambas as arrays tem 10 posições. $c=4$.

```
arthur@arthurpfonseca: ~/Documents/Poli/LabProc/gcc-arm
File Edit View Search Terminal Help
Register group: general
r0      0x3      3
r2      0x826c   33388
r4      0x4      4
r6      0x0      0
r8      0x0      0
r10     0x200100 2097408
r12     0x1fffc4 2097092
lr      0x81fc   33276
fps     0x0      0
r1      0x8244   33348
r3      0xaa20   43552
r5      0x1ffff0 2097136
r7      0x0      0
r9      0x0      0
r11     0x0      0
sp      0x1ffff0 2097136
pc      0x8228   33320
cpsr    0x60000013 1610612755

array_assign.s
10      adr r1, .a
11      adr r2, .b
12
13      LOOP:
14      @ condition
> 15      CMP r3, #9
16
17      @ body
18      LDR r5, [r2, r3, LSL #2] @ r5 = b[i]
19      add r5, r5, r4           @ r5 = b[i] + c
20      STR r5, [r1, r3, LSL #2] @ a[i] = r5
21

sim process 42 In: LOOP                                     Line: 15   PC: 0x8228

Breakpoint 1, main () at array_assign.s:7
Current language: auto; currently asm
(gdb) s
LOOP () at array_assign.s:15
(gdb) x/10 0x8244
0x8244 <.a>:  0xff00ff00  0xff00ff00  0xff00ff00  0xff00ff00
0x8254 <.a+16>: 0xff00ff00  0xff00ff00  0xff00ff00  0xff00ff00
0x8264 <.a+32>: 0xff00ff00  0xff00ff00
(gdb) x/10 0x826c
0x826c <.b>:  0x12345678  0x87654321  0xff00ff00  0xff00ff00
0x827c <.b+16>: 0xff00ff00  0xff00ff00  0xff00ff00  0xff00ff00
0x828c <.b+32>: 0xff00ff00  0xff00ff00
(gdb)
```

Início da execução.

```

arthur@arthurpfonseca: ~/Documents/Poli/LabProc/gcc-arm
File Edit View Search Terminal Help
Register group: general
r0      0x3      3          r1      0x8244   33348
r2      0x826c   33388      r3      0xa      10
r4      0x4      4          r5      0xff00ff04 -16711932
r6      0x0      0          r7      0x0      0
r8      0x0      0          r9      0x0      0
r10     0x200100 2097408    r11     0x0      0
r12     0x1fffc4 2097092    sp      0x1ffff0 2097136
lr      0x81fc   33276      pc      0x8240   33344
fps     0x0      0          cpsr    0x60000013 1610612755

array_assign.s
21
22      @ increment
23      ADD r3, r3, #1 @ i++
24      BLT LOOP
25
26      swi 0x0
27
28
29      .a:
30      .word 0xff00ff00 @ a[0]
31      .word 0xff00ff00 @ a[1]
32      .word 0xff00ff00 @ a[2]

sim process 42 In: LOOP                               Line: 26   PC: 0x8240
0x828c <.b+32>: 0xff00ff00      0xff00ff00
(gdb) c
Continuing.

Breakpoint 2, LOOP () at array_assign.s:26
(gdb) x/10 0x8244
0x8244 <.a>: 0x1234567c      0x87654325      0xff00ff04      0xff00ff04
0x8254 <.a+16>: 0xff00ff04      0xff00ff04      0xff00ff04      0xff00ff04
0x8264 <.a+32>: 0xff00ff04      0xff00ff04
(gdb) x/10 0x826c
0x826c <.b>: 0x12345678      0x87654321      0xff00ff00      0xff00ff00
0x827c <.b+16>: 0xff00ff00      0xff00ff00      0xff00ff00      0xff00ff00
0x828c <.b+32>: 0xff00ff00      0xff00ff00
(gdb)

```

Fim da execução. $a[i]=b[i]+4$, para n entre 0 e 9.

Exercício 4.5.4 - Arrays and pointers

Índice

```
arthur@arthurfonseca: ~/Documents/Poli/LabProc/gcc-arm
File Edit View Search Terminal Help
Register group: general
r0 0x0 0 r1 0x823c 33340
r2 0xffffffff -1 r3 0x0 0
r4 0x3 3 r5 0x1ffff0 2097136
r6 0x0 0 r7 0x0 0
r8 0x0 0 r9 0x0 0
r10 0x200100 2097408 r11 0x0 0
r12 0x1fffc4 2097092 sp 0x1ffff0 2097136
lr 0x81fc 33276 pc 0x8224 33316
fps 0x0 0 cpsr 0x6000013 1610612755

5      .globl main
6      main:
7          ldr r3, =0x0    @ i = 0
8          ldr r0, =0x0    @ aux = 0
9
10         adr r1, .a      @ a[]
11         mov r2, #10     @ s = 10
12
13     LOOP:
14         @ condition
15         CMP r3, r2      @ i VS s
16
17         @ body

sim process 42 In: main                               Line: 11  PC: 0x8224
Breakpoint 1, main () at arrays_n_ptrs.s:7
current language: auto; currently asm
(gdb) .byte 0x01 @ a[0]
Undefined command: ".". Try "help".
(gdb) .byte 0x02 @ a[1]
Undefined command: ".". Try "help".
(gdb) .byte 0x03 @ a[2]
Undefined command: ".". Try "help".
(gdb) s
(gdb) x/10 0x823c
0x823c <.a>: 0x04030201 0x08070605 0x00000a09 0xe1a0c00d
0x824c <atexit+4>: 0xe92dd830 0xe59f5080 0xe5953000 0xe5931148
0x825c <atexit+20>: 0xe3510000 0x02831f53
(gdb)
```

Início da execução, vê-se a array na memória.

```
arthur@arthurfonseca: ~/Documents/Poli/LabProc/gcc-arm
File Edit View Search Terminal Help
Register group: general
r0 0x0 0 r1 0x823c 33340
r2 0xa 10 r3 0x2 2
r4 0x3 3 r5 0x1ffff0 2097136
r6 0x0 0 r7 0x0 0
r8 0x0 0 r9 0x0 0
r10 0x200100 2097408 r11 0x0 0
r12 0x1fffc4 2097092 sp 0x1ffff0 2097136
lr 0x81fc 33276 pc 0x8228 33320
fps 0x0 0 cpsr 0x80000013 -2147483629

--arrays_n_ptrs.s--
13     LOOP:
14         @ condition
15         CMP r3, r2      @ i VS s
16
17         @ body
18         strb r0, [r1, r3] @ a[i] = 0
19
20         @ increment
21         ADD r3, r3, #1 @ i++
22
23         BLT LOOP
24
25         swi 0x0

sim process 42 In: LOOP                               Line: 15  PC: 0x8228
(gdb) .byte 0x03 @ a[2]
Undefined command: ".". Try "help".
(gdb) s
(gdb) x/10 0x823c
0x823c <.a>: 0x04030000 0x08070605 0x00000a09 0xe1a0c00d
0x824c <atexit+4>: 0xe92dd830 0xe59f5080 0xe5953000 0xe5931148
0x825c <atexit+20>: 0xe3510000 0x02831f53
(gdb) s
LOOP () at arrays_n_ptrs.s:15
(gdb) x/10 0x823c
0x823c <.a>: 0x04030000 0x08070605 0x00000a09 0xe1a0c00d
0x824c <atexit+4>: 0xe92dd830 0xe59f5080 0xe5953000 0xe5931148
0x825c <atexit+20>: 0xe3510000 0x02831f53
(gdb)
```

No meio da execução, alguns bytes são = 0.

```

arthur@arthurpfonseca: ~/Documents/Poli/LabProc/gcc-arm
File Edit View Search Terminal Help

Register group: general
r0      0x0      0      r1      0x823c   33340
r2      0xa     10     r3      0xb      11
r4      0x3     3      r5      0xfffff0 2097136
r6      0x0     0      r7      0x0      0
r8      0x0     0      r9      0x0      0
r10     0x200100 2097408 r11     0x0      0
r12     0x1fffc4 2097092 sp      0xfffff0 2097136
lr      0x81fc   33276  pc      0x8238   33336
fps     0x0     0      cpsr    0x60000013 1610612755

arrays_n_ptr.s
19
20      @ increment
21      ADD r3, r3, #1 @ i++
22
23      BLT LOOP
24
B+> 25      swi 0x0
26
27      .a:
28      .byte 0x01 @ a[0]
29      .byte 0x02 @ a[1]
30      .byte 0x03 @ a[2]
31      .byte 0x04 @ a[3]

sim process 42 In: LOOP                               Line: 25   PC: 0x8238
LOOP () at arrays_n_ptr.s:15
(gdb) x/10 0x823c
0x823c <-.a>:  0x04030000  0x08070605  0x00000a09  0xe1a0c00d
0x824c <atexit+4>:  0xe92dd830  0xe59f5080  0xe5953000  0xe5931148
0x825c <atexit+20>:  0xe3510000  0x02831f53  0xe5953000  0xe5931148
(gdb) c
Continuing.

Breakpoint 2, LOOP () at arrays_n_ptr.s:25
(gdb) x/10 0x823c
0x823c <-.a>:  0x00000000  0x00000000  0x00000000  0xe1a0c00d
0x824c <atexit+4>:  0xe92dd830  0xe59f5080  0xe5953000  0xe5931148
0x825c <atexit+20>:  0xe3510000  0x02831f53  0xe5953000  0xe5931148
(gdb)

```

Ao final da execução, toda a array = 0.

Pointers

```

arthur@arthurpfonseca: ~/Documents/Pol/LabProc/gcc-arm
File Edit View Search Terminal Help
--Register group: general--
r0 0x0 0 r1 0x8246 33350
r2 0xa 10 r3 0x823c 33340
r4 0x3 3 r5 0x1ffff0 2097136
r6 0x0 0 r7 0x0 0
r8 0x0 0 r9 0x0 0
r10 0x200100 2097408 r11 0x0 0
r12 0x1fffc4 2097092 sp 0x1ffff0 2097136
lr 0x81fc 33276 pc 0x8228 33320
fps 0x0 0 cpsr 0x60000013 1610612755

11      mov r2, #10      @ s = 10
12      add r1, r3, r2    @ r1 = &a[0] + s = &a[s]
13
14      LOOP:
15          @ condition
16      >+ CMP r3, r1      @ p VS &a[s]
17
18          @ body
19      strb r0, [r3]      @ *p = 0
20
21          @ increment
22      ADD r3, r3, #1     @ p++
23

sim process 42 In: LOOP                               Line: 16   PC: 0x8228
(gdb) r
Starting program: /home/student/src/Lab4/Lab de fato/a.out

Breakpoint 1, LOOP () at arrays_n_ptrs_b.s:16
Current language: auto; currently asm
(gdb) x/10 0x8246
0x8246 <.a+10>: 0xc00d0000 0xd830e1a0 0x5080e92d 0x3000e59f
0x8256 <atexit+14>: 0x1148e595 0x0000e593 0x1f53e351 0xb0040283
0x8266 <atexit+30>: 0xc004e24c 0x1148e591
(gdb) x/10 0x823c
0x823c <.a>: 0x04030201 0x08070605 0x00000a09 0xe1a0c00d
0x824c <atexit+4>: 0xe92dd830 0xe59f5080 0xe5953000 0xe5931148
0x825c <atexit+20>: 0xe3510000 0x02831f53
(gdb)

```

Início da execução, vê-se a array na memória.

```

arthur@arthurpfonseca: ~/Documents/Pol/LabProc/gcc-arm
File Edit View Search Terminal Help
--Register group: general--
r0 0x0 0 r1 0x8246 33350
r2 0xa 10 r3 0x823e 33342
r4 0x3 3 r5 0x1ffff0 2097136
r6 0x0 0 r7 0x0 0
r8 0x0 0 r9 0x0 0
r10 0x200100 2097408 r11 0x0 0
r12 0x1fffc4 2097092 sp 0x1ffff0 2097136
lr 0x81fc 33276 pc 0x8234 33332
fps 0x0 0 cpsr 0x80000013 -2147483629

--arrays_n_ptrs_b.s--
17
18      @ body
19      strb r0, [r3]      @ *p = 0
20
21      @ increment
22      ADD r3, r3, #1     @ p++
23
24      > BLT LOOP
25
26      swi 0x0
27
28      .a:
29      .byte 0x01 @ a[0]

sim process 42 In: LOOP                               Line: 24   PC: 0x8234
0x824c <atexit+4>: 0xe92dd830 0xe59f5080 0xe5953000 0xe5931148
0x825c <atexit+20>: 0xe3510000 0x02831f53
(gdb) s

Breakpoint 1, LOOP () at arrays_n_ptrs_b.s:16
(gdb) x/10 0x8246
0x8246 <.a+10>: 0xc00d0000 0xd830e1a0 0x5080e92d 0x3000e59f
0x8256 <atexit+14>: 0x1148e595 0x0000e593 0x1f53e351 0xb0040283
0x8266 <atexit+30>: 0xc004e24c 0x1148e591
(gdb) x/10 0x823c
0x823c <.a>: 0x04030000 0x08070605 0x00000a09 0xe1a0c00d
0x824c <atexit+4>: 0xe92dd830 0xe59f5080 0xe5953000 0xe5931148
0x825c <atexit+20>: 0xe3510000 0x02831f53
(gdb)

```

No meio da execução, alguns bytes são = 0.

```

arthur@arthurfonseca: ~/Documents/Pol/LabProc/gcc-arm
File Edit View Search Terminal Help
Register group: general
r0      0x0      0          r1      0x8246  33350
r2      0xa     10         r3      0x8247  33351
r4      0x3     3          r5      0x1ffff0 2097136
r6      0x2     2          r7      0x0     0
r8      0xa414  42004     r9      0xa23c  41532
r10     0x200100 2097408  r11     0x0     0
r12     0x1fffc4 2097092  sp      0x1ffff0 2097136
lr      0x81fc  33276    pc      0x8238  33336
fps     0x0     0          cpsr    0x6000013 1610612755

--arrays_n_ptrs_b.s--
21      @ increment
22      ADD r3, r3, #1 @ p++
23
24      BLT LOOP
25
8-> 26      swi 0x0
27
28      .a:
29      .byte 0x01 @ a[0]
30      .byte 0x02 @ a[1]
31      .byte 0x03 @ a[2]
32      .byte 0x04 @ a[3]
33

sim process 42 In: LOOP                               Line: 26   PC: 0x8238
Continuing.

Breakpoint 1, LOOP () at arrays_n_ptrs_b.s:16
Continuing.

Breakpoint 1, LOOP () at arrays_n_ptrs_b.s:16
Continuing.

Breakpoint 2, LOOP () at arrays_n_ptrs_b.s:26
(gdb) x/10 0x823c
0x823c <.a>: 0x00000000 0x00000000 0x00000000 0x1a0c00d
0x824c <atexit+4>: 0xe92dd830 0xe59f5080 0xe5953000 0xe5931148
0x825c <atexit+20>: 0xe3510000 0x02831f53
(gdb) 
```

Ao final da execução, toda a array = 0.

Exercício 4.5.5 - The Fibonacci sequence

```
arthur@arthurpfonseca: ~/Documents/Poli/LabProc/gcc-arm
File Edit View Search Terminal Help
Register group: general
r0      0x3      3      r1      0x1ffff0 2097136
r2      0xffffffff -1     r3      0xa9f4  43508
r4      0x3      3      r5      0x1ffff0 2097136
r6      0x0      0      r7      0x0      0
r8      0x0      0      r9      0x0      0
r10     0x200100 2097408 r11     0x0      0
r12     0x1fffc4 2097092 sp      0x1ffff0 2097136
lr      0x81fc   33276  pc      0x8218   33304
fps     0x0      0      cpsr    0x60000013 1610612755

2      @ Para debugar este codigo:
3      @ gcc fibo.s && gdb a.out
4      .text
5      .globl  main
6      main:
B+> 7      LDR      r7, =0x0
8      LDR      r8, =0x4000
9
10     STRB     r7, [r8]
11     LDR      r7, =0x1
12     STRB     r7, [r8, #1]
13
14     LDR      r0, =fibon

sim process 42 In: main                               Line: 7    PC: 0x8218
(gdb) b 30
Breakpoint 1 at 0x825c: file fibo.s, line 30.
(gdb) b main
Breakpoint 2 at 0x8218: file fibo.s, line 7.
(gdb) r
Starting program: /home/student/src/Lab4/Lab de fato/a.out

Breakpoint 2, main () at fibo.s:7
Current language: auto; currently asm
(gdb) x/10 0x8250
0x8250 <fibon>: 0x00000100      0x00000000      0x00000000      0xef123456
0x8260 <$d>:      0x00008250      0x0000825d      0xe1a0c00d      0xe92dd830
0x8270 <atexit+8>: 0xe59f5080      0xe5953000
(gdb)
```

Início da execução

```
arthur@arthurpfonseca: ~/Documents/Poli/LabProc/gcc-arm
File Edit View Search Terminal Help
Register group: general
r0      0x825d  33373      r1      0x1dc    476
r2      0xde   222         r3      0xa9f4   43508
r4      0x3    3           r5      0x1ffff0  2097136
r6      0x0    0           r7      0x825d    33373
r8      0x4000 16384       r9      0x1ffff0  2097136
r10     0x200100 2097408    r11     0x0      0
r12     0x1fffc4 2097092    sp      0x1ffff0  2097136
lr      0x81fc  33276      pc      0x825c  33372
fps     0x0     0           cpsr    0x60000013 1610612755

fibonacci.s
25      B      loop
26
27      fibon:
28      .byte  0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
29      out:
30      SWI 0x123456
31
32
33
34
35
36
37

sim process 42 In: out                                     Line: 30  PC: 0x825c
Start it from the beginning? (y or n)

Starting program: /home/student/src/Lab4/Lab de fato/a.out

Breakpoint 2, main () at fibo.s:7
(gdb) s
loop () at fibo.s:17

Breakpoint 1, out () at fibo.s:30
(gdb) x/10 0x8250
0x8250 <fibon>: 0x02010100      0x0d080503      0x59372215      0xef123456
0x8260 <$d>:      0x00008250      0x0000825d      0xe1a0c00d      0xe92dd830
0x8270 <atexit+8>:      0xe59f5080      0xe5953000
(gdb) 
```

Após a execução, o array armazena os 12 primeiros elementos da sequência de Fibonacci.

Exercício 4.5.6 - The nth Fibonacci number

```
arthur@arthurpfonseca: ~/Documents/Poli/LabProc/gcc-arm
File Edit View Search Terminal Help
Register group: general
r0      0x3      3      r1      0xa      10
r2      0xffffffff -1    r3      0x8258   33368
r4      0x3      3      r5      0x1ffff0 2097136
r6      0x0      0      r7      0x8262   33378
r8      0x4000   16384 r9      0x0      0
r10     0x200100 2097408 r11     0x0      0
r12     0x1fffc4 2097092 sp      0x1ffff0 2097136
lr      0x81fc   33276 pc      0x823c   33340
fps     0x0      0      cpsr    0x60000013 1610612755

15
16      LDR      r3, =fibon @ end atual
17      ADD      r7, r3, r1 @ end do ultimo
18      loop:
19
B+> 20      LDRB    r4, [r3], #1
21      LDRB    r2, [r3]
22      ADD     r0, r2, r4
23      STRB    r0, [r3, #1]
24
25      CMP     r3, r7
26      BLT     loop
27
```

Início da execução

```

arthur@arthurpfonseca: ~/Documents/Poli/LabProc/gcc-arm
File Edit View Search Terminal Help
Register group: general
r0      0x59      89          r1      0xa       10
r2      0x37      55          r3      0x8262    33378
r4      0x22      34          r5      0x1ffff0  2097136
r6      0x0       0           r7      0x8262    33378
r8      0x4000    16384       r9      0x0       0
r10     0x200100  2097408     r11     0x0       0
r12     0x1fffc4  2097092     sp      0x1ffff0  2097136
lr      0x81fc    33276       pc      0x8264   33380
fps     0x0       0           cpsr    0x60000013 1610612755

2_fibo.s
28      B      out
29
30      fibon:
31      .byte   0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
32      out:
B+> 33      SWI 0x123456
34
35
36
37
38
39
40

sim process 42 In: out                                     Line: 33   PC: 0x8264
Breakpoint 1, loop () at 2_fibo.s:20
Breakpoint 1, loop () at 2_fibo.s:20
Breakpoint 1, loop () at 2_fibo.s:20
Breakpoint 1, loop () at 2_fibo.s:20
Breakpoint 2, out () at 2_fibo.s:33
(gdb) x/10 0x8254
0x8254 <loop+24>: 0xea000002 0x02010100 0x0d080503 0x59372215
0x8264 <out>: 0xef123456 0x00008258 0xe1a0c00d 0xe92dd830
0x8274 <atexit+8>: 0xe59f5080 0xe5953000
(gdb) 
```

Após a execução, vê-se os resultados a partir da segunda palavra (na memória) da imagem acima.

Apêndice

1. Exercício 4.5.1

Pré indexado

```
@ Exercicio 4.5.1 do livro
@ Para debugar este codigo:
@ gcc assig_pre.s && gdb a.out

.text
.globl    main
main:
    MOV r1, #0x2
    LDR    r2, =arr
    LDR r3, [r2, #5]
    ADD    r0, r1, r3
    SWI 0x0
arr:
    .word  0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0,
0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3
```

Pós fixado

Pós indexado

```
@ Exercicio 4.5.1 do livro
@ Para debugar este codigo:
@ gcc assig_pos.s && gdb a.out

.text
.globl    main
main:
    MOV r1, #0x2
    LDR    r2, =arr
    ADD    r4, r2, #5
    LDR    r3, [r4], #0
    ADD    r0, r1, r3
    SWI 0x0
arr:
    .word  0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0,
0x1, 0x2, 0x3
```

2. Exercício 4.5.2

Pós indexado

```
@ Exercicio 4.5.1 do livro
@ Para debugar este codigo:
@ gcc load_store_pos.s && gdb a.out

.text
.globl      main
main:
    MOV r1, #0x2      @ y qualquer
    ADR r2, arr        @ salva end inicial em r2
    ADD r3, r2, #20    @ salva o endereço de array[5] em r3
    LDR r4, [r3], #0   @ salva o valor de array[5] em r4
    ADD r0, r1, r4     @ salva array[5] + y em r0
    ADD r3, r2, #40    @ salva o endereço de array[10] em r3
    STR r0, [r3]      @ salve o valor de r0 no endereço de r3
    SWI 0x0

arr:
    .word 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0,
    0x1, 0x2, 0x3
```

Pré indexado

```
@ Exercicio 4.5.1 do livro
@ Para debugar este codigo:
@ gcc load_store_pre.s && gdb a.out

.text
.globl      main
main:
    MOV r1, #0x2      @ y qualquer
    ADR r2, arr        @ salva end inicial em r2
    LDR r4, [r2, #20]  @ salva o valor de array[5] em r4
    ADD r0, r1, r4     @ salva array[5] + y em r0
    STR r0, [r2, #40]  @ salve o valor de r0 no endereço de array[10]
    SWI 0x0

arr:
    .word 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0, 0x1, 0x2, 0x3, 0x0,
    0x1, 0x2, 0x3
```

3. Exercício 4.5.3

@ Exercício 4.5.3 do livro
@ Para debugar este código:
@ gcc array_assign.s && gdb a.out

```
.text
.globl main
main:
    ldr r3, =0x0 @ i = 0
    ldr r4, =0x4 @ c = 4

    adr r1, .a
    adr r2, .b

LOOP:
    @ condition
    CMP r3, #10

    @ body
    LDR r5, [r2, r3, LSL #2] @ r5 = b[i]
    add r5, r5, r4 @ r5 = b[i] + c
    STR r5, [r1, r3, LSL #2] @ a[i] = r5

    @ increment
    ADD r3, r3, #1 @ i++

    BLT LOOP

    swi 0x0

.a:
    .word 0xff00ff00 @ a[0]
    .word 0xff00ff00 @ a[1]
    .word 0xff00ff00 @ a[2]
    .word 0xff00ff00 @ a[3]
    .word 0xff00ff00 @ a[4]
    .word 0xff00ff00 @ a[5]
    .word 0xff00ff00 @ a[6]
    .word 0xff00ff00 @ a[7]
    .word 0xff00ff00 @ a[8]
    .word 0xff00ff00 @ a[9]

.b:
    .word 0x12345678 @ b[0]
    .word 0x87654321 @ b[1]
    .word 0xff00ff00 @ b[2]
    .word 0xff00ff00 @ b[3]
    .word 0xff00ff00 @ b[4]
    .word 0xff00ff00 @ b[5]
    .word 0xff00ff00 @ b[6]
    .word 0xff00ff00 @ b[7]
    .word 0xff00ff00 @ b[8]
    .word 0xff00ff00 @ b[9]
```

4. Exercício 4.5.4

4.5.4.A

@ Exercicio 4.5.4.A do livro
@ Para debugar este codigo:
@ gcc arrays_n_ptrs.s && gdb a.out

```
.text
.globl main
main:
    ldr r3, =0x0    @ i = 0
    ldr r0, =0x0    @ aux = 0

    adr r1, .a      @ a[]
    mov r2, #10     @ s = 10

LOOP:
    @ condition
    CMP r3, r2      @ i VS s

    @ body
    strb r0, [r1, r3] @ a[i] = 0

    @ increment
    ADD r3, r3, #1   @ i++

    BLT LOOP

    swi 0x0

.a:
    .byte 0x01 @ a[0]
    .byte 0x02 @ a[1]
    .byte 0x03 @ a[2]
    .byte 0x04 @ a[3]

    .byte 0x05 @ a[4]
    .byte 0x06 @ a[5]
    .byte 0x07 @ a[6]
    .byte 0x08 @ a[7]

    .byte 0x09 @ a[8]
    .byte 0x0a @ a[9]

    .align 1
```

4.5.4.B

@ Exercicio 4.5.4.B do livro

```

@ Para debugar este codigo:
@ gcc arrays_n_ptrs_b.s && gdb a.out

        .text
        .globl main
main:
        adr r3, .a      @ p = &a[0]
        ldr r0, =0x0    @ aux = 0

        @adr r1, .a     @ a[]
        mov r2, #10     @ s = 10
        add r1, r3, r2  @ r1 = &a[0] + s = &a[s]

LOOP:
        @ condition
        CMP r3, r1      @ p VS &a[s]

        @ body
        strb r0, [r3]   @ *p = 0

        @ increment
        ADD r3, r3, #1   @ p++

        BLT LOOP

        swi 0x0

.a:
        .byte 0x01 @ a[0]
        .byte 0x02 @ a[1]
        .byte 0x03 @ a[2]
        .byte 0x04 @ a[3]

        .byte 0x05 @ a[4]
        .byte 0x06 @ a[5]
        .byte 0x07 @ a[6]
        .byte 0x08 @ a[7]

        .byte 0x09 @ a[8]
        .byte 0x0a @ a[9]

        .align 1

```

5. Exercício 4.5.5

```

@ Exercicio 4.5.5 do livro
@ Para debugar este codigo:
@ gcc fibo.s && gdb a.out

```

```

        .text
        .globl main

```

```

main:
    LDR        r7, =0x0
    LDR        r8, =0x4000

    STRB    r7, [r8]
    LDR    r7, =0x1
    STRB    r7, [r8, #1]

    LDR    r0, =fibon
    LDR    r7, =fibon + 13
loop:
    CMP    r7, r0
    BLE    out

    LDRB    r1, [r0], #1
    LDRB    r2, [r0]
    ADD    r1, r2, r1
    STRB    r1, [r0, #1]

    B      loop
out:
    SWI    0x123456

fibon:
    .byte    0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0

```

6. Exercício 4.5.6

```

@ Exercicio 4.5.6 do livro
@ Para debugar este codigo:
@ gcc 2_fibo.s && gdb a.out

.text
.globl main
main:
    LDR r7, =0x0
    LDR r8, =0x4000
    MOV r1, #11
    SUB r1, r1, #1 @ Tira 1 pro indice ficar certo

    STRB    r7, [r8]
    LDR    r7, =0x1
    STRB    r7, [r8, #1]

    LDR    r3, =fibon @ end atual
    ADD    r7, r3, r1 @ end do ultimo
loop:

    LDRB    r4, [r3], #1
    LDRB    r2, [r3]
    ADD    r0, r2, r4

```

STRB r0, [r3, #1]

CMP r3, r7

BLT loop

B out

out:

SWI 0x123456

fibon:

.byte 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0