

ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO
Graduação em Engenharia Computação

PCS3732 - Laboratório de Processadores

Professor Jorge Kinoshita



Grupo 10 - Planejamento E6

Arthur Pires da Fonseca NUSP: 10773096

Como a função `imprime()` está usando a pilha para fazer a chamada recursiva? Apresente a explicação de como a pilha está sendo usada (simule desenhando a pilha usando o `fp`, `ip`, `sp`) para fazer a chamada recursiva.

A cada chamada da rotina `imprime()`, o ponteiro para a pilha é salvo no registrador `$ip`, cujo valor acaba também sendo posto na pilha em seguida, e em seguida `$fp` é inicializado com o valor de `$ip` decrescido de 4.

A pilha fica sempre com a seguinte configuração, logo antes de uma chamada à label `.L2` do código Assembly gerado:

```
sp ->          0x1fffd4 -> o topo da pilha
r11 / fp ->    0x1fffe4 -> o topo da pilha anterior - 1 word
r12 / ip ->    0x1fffe8 -> o topo da pilha anterior
```

Na última execução da função, quando o registrador `$r3` fica com o valor "-1", a pilha (`$sp`) e o `$ip` encontram-se da seguinte forma:

(gdb) x/30x \$sp

```
0x1fff5c:  0xffffffff 0x001fff80 0x001fff70 0x00008258
0x1fff6c:  0x00008224 0x00000000 0x001fff94 0x001fff84
0x1fff7c:  0x00008258 0x00008224 0x00000001 0x001fffa8
0x1fff8c:  0x001fff98 0x00008258 0x00008224 0x00000002
0x1fff9c:  0x001fffb0 0x001fffac 0x00008258 0x00008224
0x1fffac:  0x00000003 0x001fffd0 0x001fffc0 0x00008258
0x1fffb0:  0x00008224 0x00000004 0x001fffe4 0x001fffd4
0x1fffcc:  0x00008258 0x00008224
```

(gdb) x/30x \$r12

```
0x1fff70:  0x00000000 0x001fff94 0x001fff84 0x00008258
0x1fff80:  0x00008224 0x00000001 0x001fffa8 0x001fff98
0x1fff90:  0x00008258 0x00008224 0x00000002 0x001fffb0
0x1fffa0:  0x001fffac 0x00008258 0x00008224 0x00000003
0x1fffb0:  0x001fffd0 0x001fffc0 0x00008258 0x00008224
0x1fffc0:  0x00000004 0x001fffe4 0x001fffd4 0x00008258
0x1fffd0:  0x00008224 0x00000005 0x001ffff4 0x001fffe8
0x1fffe0:  0x00008274 0x00008224
```

```

arthur@arthurpfonseca: ~/Documents/Poli/LabProc/gcc-arm
File Edit View Search Terminal Help
Register group: general
r0      0xffffffff -1
r2      0x10484 66692
r4      0x1 1
r6      0x0 0
r8      0x0 0
r10     0x200100 2097408
r12     0x1fff70 2097008
lr      0x8258 33368
fps     0x0 0
r1      0x0 0
r3      0xffffffff -1
r5      0x1ffff8 2097144
r7      0x0 0
r9      0x0 0
r11     0x1fff6c 2097004
sp      0x1fff5c 2096988
pc      0x8258 33368
cpsr    0xa0000013 -1610612717

-imprime.s
27      sub    r3, r3, #1
28      mov    r0, r3
29      bl     imprime
30      .L1:
> 31      ldmfd  sp, {r3, fp, sp, pc}
32      .L4:
33      .align 2
34      .L3:
35      .word   .LC0
36      .size   imprime, .-imprime

sim process 42 In: imprime Line: 31 PC: 0x8258
0x200020: 0 0 0 0
0x200030: 0 0
(gdb) x/30x $r12+80
0x1fffc0: 0x00000004 0x001fffe4 0x001fffd4 0x00008258
0x1fffd0: 0x00008224 0x00000005 0x001ffff4 0x001fffe8
0x1fffe0: 0x00008274 0x00008224 0x00000000 0x001ffff8
0x1ffff0: 0x000081fc 0x0000826c 0x000100bc 0x00000000
0x200000: 0x00000000 0x00000000 0x00000000 0x00000000
0x200010: 0x00000000 0x00000000 0x00000000 0x00000000
0x200020: 0x00000000 0x00000000 0x00000000 0x00000000
0x200030: 0x00000000 0x00000000 0x00000000 0x00000000
(gdb)

```

Momento final do programa, quando os valores de antes da primeira chamada da função `imprime()`.

Como o valor `N` é passado como parâmetro para `imprime()` em `main`? via registrador ou pilha?

O valor de `N` é passado para a função `imprime()` através do registrador `$r0`.

Para que serve o `frame pointer`?

O registrador `$fp` guarda o valor do `stack pointer` (`$sp`) em cada uma das chamadas de uma função ou série de funções.

Como o valor `N` é referenciado dentro de `imprime.s` (vindo do compilador)?

Dentro do código Assembly da função `imprime()`, o valor de `N` alterna entre a pilha e os registradores `$r3` e `$r0`.

No momento de verificar a condição da recursão, o `N` é referenciado sempre pelo registrador `$r3`.

Outra imagem que achei relevante colocar no relatório:

```
arthur@arthurpfonseca: ~/Documents/Poli/LabProc/gcc-arm
File Edit View Search Terminal Help
--Register group: general--
r0      0xfd84 64900
r2      0xffffffff -1
r4      0x1 1
r6      0x0 0
r8      0x0 0
r10     0x200100 2097408
r12     0x1fffe8 2097128
lr      0x8274 33396
fps     0x0 0
r1      0x1ffff8 2097144
r3      0x5 5
r5      0x1ffff8 2097144
r7      0x0 0
r9      0x0 0
r11     0x1fffe4 2097124
sp      0x1fffd4 2097108
pc      0x8240 33344
cpsr    0x20000013 536870931

--imprime.s--
21      b .L1
22      .L2:
23      ldr r0, .L3
> 24      ldr r1, [fp, #-16]
25      bl printf
26      ldr r3, [fp, #-16]
27      sub r3, r3, #1
28      mov r0, r3
29      bl imprime
30      .L1:

sim process 42 In: imprime
(gdb) s
(gdb) x/10c $r0
0xfd84 <$a+8>: 110 'n' 117 'u' 109 'm' 101 'e' 114 'r' 111 'o' 32 ' ' 61 '='
0xfd8c <$a+16>: 32 ' ' 37 '%'
(gdb) x/2-c $r0
No symbol "c" in current context.
(gdb) x/30c $r0
0xfd84 <$a+8>: 110 'n' 117 'u' 109 'm' 101 'e' 114 'r' 111 'o' 32 ' ' 61 '='
0xfd8c <$a+16>: 32 ' ' 37 '%' 100 'd' 10 '\n' 0 '\0' 0 '\0' 0 '\0' 0 '\0'
0xfd94 <$a+24>: 67 'C' 0 '\0' 0 '\0' 0 '\0' 58 ':' 116 't' 116 't' 0 '\0'
0xfd9c <blanks.0>: 32 ' ' 32 ' ' 32 ' ' 32 ' ' 32 ' ' 32 ' '
(gdb)
```

Registrador \$r0 recebendo o endereço para a string que será usada na chamada da função printf().