

Pile et appels de fonction (suite)

```
# int f(int *t, int n) {
#     if (n==0) return 0;
#     else return t[0]+f(t+1,n-1);
# }
# int t[] = {3,5,2,6};
# int main(void) {
#     return f(t, 4);
# }

main:  irmovl Stack,%esp
       irmovl 4,%eax      # empiler le deuxième paramètre 4
       pushl %eax
       irmovl t,%eax     # empiler le premier paramètre t
       pushl %eax
       call  f
       halt

f:     pushl %ebp
       rrmovl %esp,%ebp

       mrmovl 12(%ebp), %eax # n
       andl  %eax,%eax     # n == 0 ?
       je    fin          # on a déjà 0 dans eax

       isubl 1, %eax      # n-1
       pushl %eax        # deuxième paramètre

       mrmovl 8(%ebp), %eax # t
       iaddl 4, %eax     # t+1
       pushl %eax        # premier paramètre

       call f
       # résultat dans %eax

       # popl  %ecx      # enlever les paramètres, inutile
       # popl  %ecx      # puisque %esp est restauré plus bas

       mrmovl 8(%ebp), %ecx # t
       mrmovl (%ecx), %ecx  # t[0]
       addl  %ecx, %eax     # t[0]+f(t+1,n-1)

fin:   rrmovl %ebp,%esp
       popl  %ebp
       ret

       .pos 0x100
```

```

t:      .long 3
        .long 5
        .long 2
        .long 6

        .pos 0x200
Stack:

```

Compilée par gcc -O4 -S, on obtient :

```

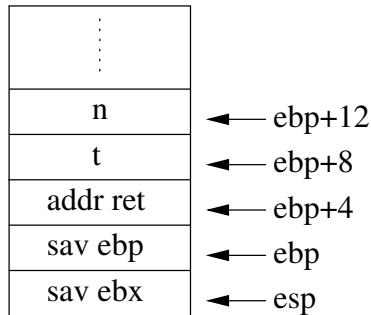
f:
    pushl %ebp
    rrmovl %esp, %ebp
    mrmovl 12(%ebp), %edx
    pushl %ebx
    xorl %ebx, %ebx
    mrmovl 8(%ebp), %ecx
    andl %edx, %edx
    jz L3

L6:
    mrmovl (%ecx), %eax
    addl 4, %ecx
    addl %eax, %ebx
    subl 1, %edx
    jnz L6

L3:
    rrmov %ebx, %eax
    pop %ebx
    pop %ebp
    ret

```

Pendant la boucle, la pile a cet aspect :



Juste pour le fun, compilé en optimisé pour Pentium4

```
f:
    pushl   %ebp
    movl   %esp, %ebp
    pushl   %edi
    pushl   %esi
    pushl   %ebx
    subl   $4, %esp
    movl   8(%ebp), %ebx
    movl   12(%ebp), %ecx
    xorl   %edx, %edx
    testl  %ecx, %ecx
    je     .L3
    movl   %ecx, -16(%ebp)
    movl   %ecx, %esi
    shrl   $2, %esi
    movl   %ecx, %edi
    andl   $-4, %edi
    cmpl   $6, %ecx
    jbe   .L9
    testl  %edi, %edi
    jne   .L4
.L9:
    xorl   %edx, %edx
.L10:
    movl   (%ebx), %eax
    addl   $4, %ebx
    addl   %eax, %edx
    subl   $1, %ecx
    jne   .L10
.L3:
    movl   %edx, %eax
    addl   $4, %esp
    popl   %ebx
    popl   %esi
    popl   %edi
    popl   %ebp
    ret

.L4:
    movl   %ebx, %edx
    pxor   %xmm2, %xmm2
    xorl   %eax, %eax
.L7:
    movdqu (%edx), %xmm0
    padd  %xmm0, %xmm2
    addl   $1, %eax
    addl   $16, %edx
    cmpl  %eax, %esi
    ja    .L7
    movdqa %xmm2, %xmm1
    psrldq $8, %xmm1
    padd  %xmm2, %xmm1
    movdqa %xmm1, %xmm0
    psrldq $4, %xmm0
    padd  %xmm1, %xmm0
    movd  %xmm0, %edx
    leal  (%ebx,%edi,4), %ebx
    subl  %edi, %ecx
    cmpl  %edi, -16(%ebp)
    je    .L3
    movl  (%ebx), %eax
    addl  $4, %ebx
    addl  %eax, %edx
    subl  $1, %ecx
    jne  .L10
    jmp  .L3
```