```
1               .file     sum.c
2               .comm     A,4000,32
3               .section  .rodata
4 .LC0:
5               .string   sum = %d\n
6               .text
7 .globl main
8               .type     main, @function
9 main:
10              pushl     %ebp
11              movl      %esp, %ebp
12              andl      $-16, %esp
13              subl      $32, %esp
14              movl      $0, 28(%esp)
15              jmp       .L2
16 .L3:
17              movl      28(%esp), %eax
18              movl      28(%esp), %edx
19              movl      %edx, A(,%eax,4)
20              addl      $1, 28(%esp)
21 .L2:
22              cmpl      $999, 28(%esp)
23              jle       .L3
24              movl      $1000, 4(%esp)
25              movl      $A, (%esp)
26              call      sum
27              movl      %eax, 4(%esp)
28              movl      $.LC0, (%esp)
29              call      printf
30              leave
31              ret
32              .size     main, .-main
33 .globl sum
34              .type     sum, @function
35 sum:
36              pushl     %ebp
37              movl      %esp, %ebp
38              subl      $56, %esp
39              movl      $0, -12(%ebp)
40              movl      8(%ebp), %eax
41              movl      %eax, -36(%ebp)
42              movl      12(%ebp), %eax
43              movl      %eax, -32(%ebp)
44              movl      -12(%ebp), %eax
45              movl      %eax, -28(%ebp)
46              movl      $0, 8(%esp)             ifが無いので0
47              leal      -36(%ebp), %eax
48              movl      %eax, 4(%esp)           引数のポインタをセット
                                                  #pragmaで指定したブロックの関数をセ
49              movl      $sum.omp_fn.0, (%esp)   ット
50              call      GOMP_parallel_start     OpenMPのスレッドを生成
51              leal      -36(%ebp), %eax
52              movl      %eax, (%esp)
53              call      sum.omp_fn.0
```

```
                                                    OpenMPが生成した全スレッドがjoinして
54              call    GOMP_parallel_end           一つに
55              movl    -36(%ebp), %eax
56              movl    %eax, 8(%ebp)
57              movl    -32(%ebp), %eax
58              movl    %eax, 12(%ebp)
59              movl    -28(%ebp), %eax
60              movl    %eax, -12(%ebp)
61              movl    -12(%ebp), %eax
62              leave
63              ret
64              .size   sum, .-sum
65              .type   sum.omp_fn.0, @function
                                                    #pragmaで指定したブロックが、関数とし
66 sum.omp_fn.0:                                    て独立させられている
67              pushl   %ebp
68              movl    %esp, %ebp
69              pushl   %esi
70              pushl   %ebx
71              subl    $16, %esp
72              movl    $0, -12(%ebp)
73              movl    8(%ebp), %eax
74              movl    4(%eax), %ebx
75              call    omp_get_num_threads         全スレッド数を得る
76              movl    %eax, %esi
77              call    omp_get_thread_num          自スレッド番号を得る
78              movl    %eax, %ecx
79              movl    %ebx, %edx
80              movl    %edx, %eax
81              sarl    $31, %edx
82              idivl   %esi                        n(ここでは1000)をスレッド数で割る
83              movl    %eax, %edx
84              imull   %esi, %edx
85              cmpl    %ebx, %edx
86              setne   %dl
87              movzbl  %dl, %edx
88              addl    %eax, %edx
89              movl    %edx, %eax

90              imull   %ecx, %eax                  スレッドが担当するiの初期値を作る
91              leal    (%eax,%edx), %edx
92              cmpl    %ebx, %edx
93              cmovg   %ebx, %edx
94              cmpl    %edx, %eax
95              jge     .L8
96              movl    %eax, -16(%ebp)
97 .L9:                                             ここが加算のループ
98              movl    -16(%ebp), %eax
99              leal    0(,%eax,4), %ecx
100             movl    8(%ebp), %eax
101             movl    (%eax), %eax
102             addl    %ecx, %eax
103             movl    (%eax), %eax
104             addl    %eax, -12(%ebp)             スレッド・ローカルなs+=A[i]
```

```
105             addl    $1, -16(%ebp)           i=i+1 (iはスレッド・ローカル)
106             cmpl    %edx, -16(%ebp)
107             jl      .L9                     iが終値でなければ .L9へ
108 .L8:
109             movl    8(%ebp), %eax
110             leal    8(%eax), %edx
111             movl    -12(%ebp), %eax
112             lock addl %eax, (%edx)          アトミックに、sに、スレッドの加算結果を加える。
113             call    GOMP_barrier            全スレッドがここに到達するまで待ち合わせ
114             addl    $16, %esp
115             popl    %ebx
116             popl    %esi
117             popl    %ebp
118             ret
119             .size   sum.omp_fn.0, .-sum.omp_fn.0
120             .ident  GCC: (Ubuntu/Linaro 4.4.4-14ubuntu5) 4.4.5
121             .section .note.GNU-stack,"",@progbits
122
```