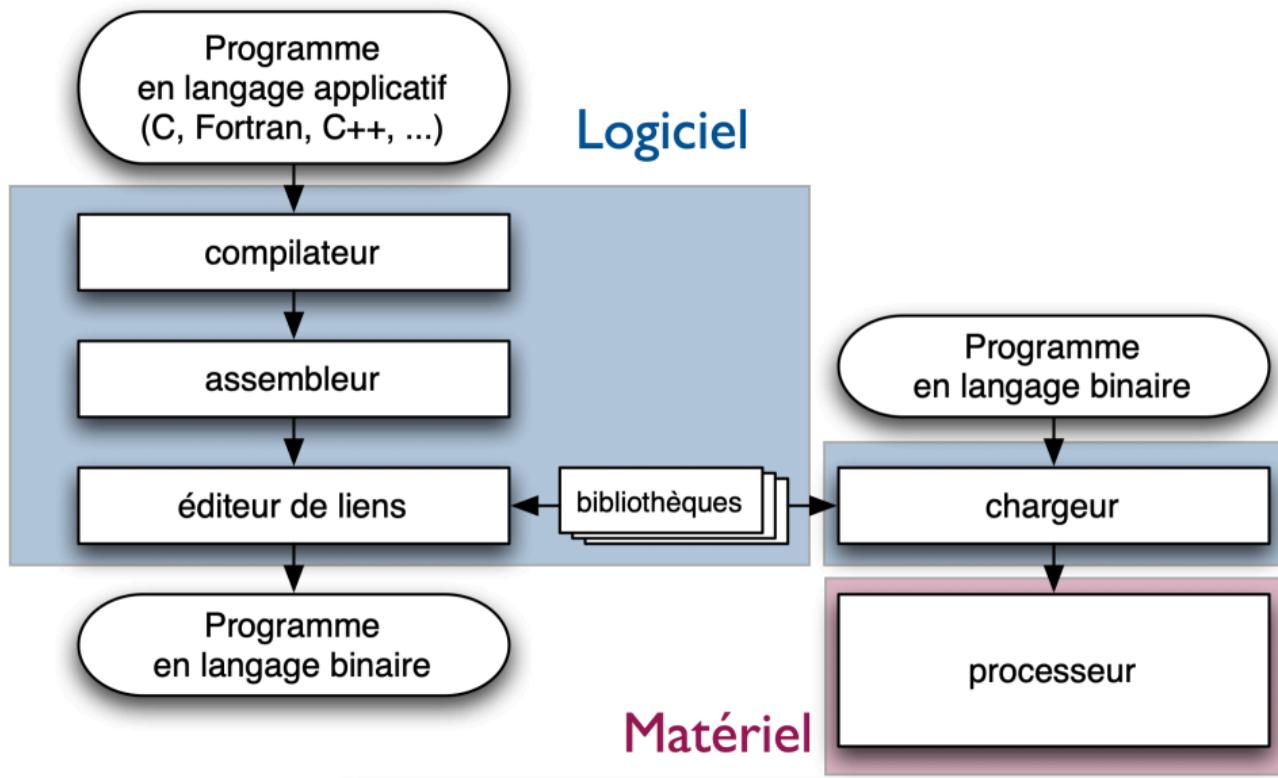


Introduction au développement logiciel

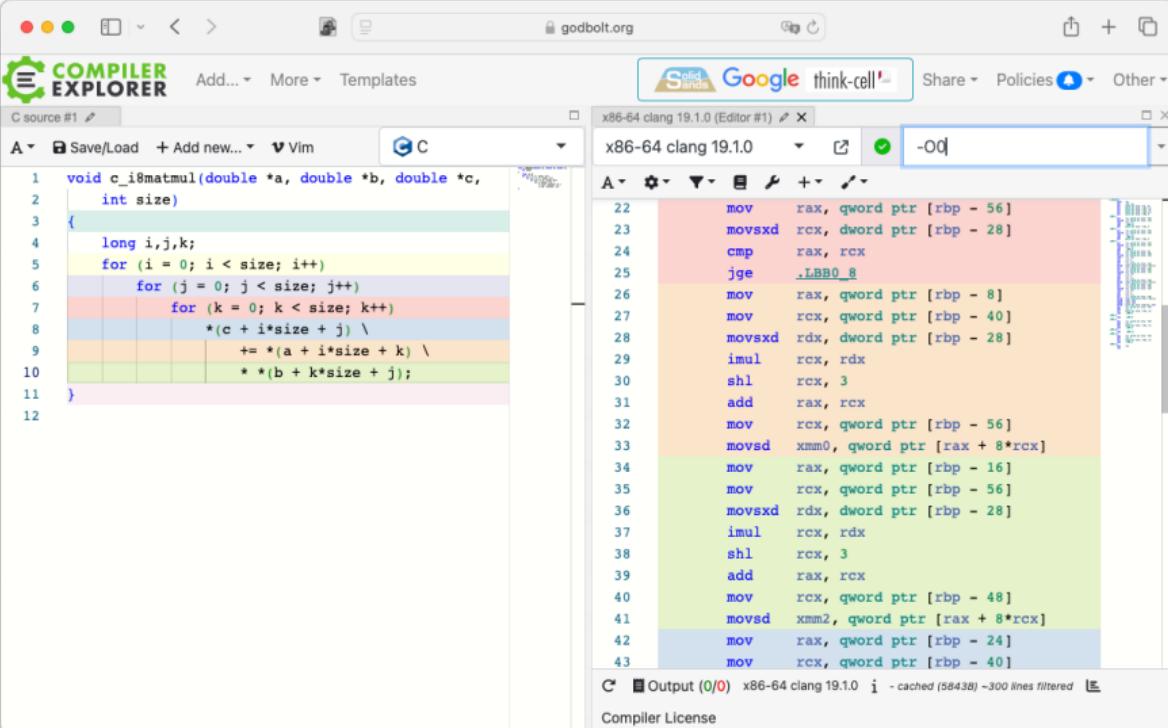
Langage machine et langage d'assemblage

Georges-André Silber, Centre de recherche en informatique

Mines Paris – PSL, octobre 2025



Compilation sans optimisation



The image shows a screenshot of the Compiler Explorer interface on godbolt.org. On the left, the C source code for a 16x16 matrix multiplication is displayed:

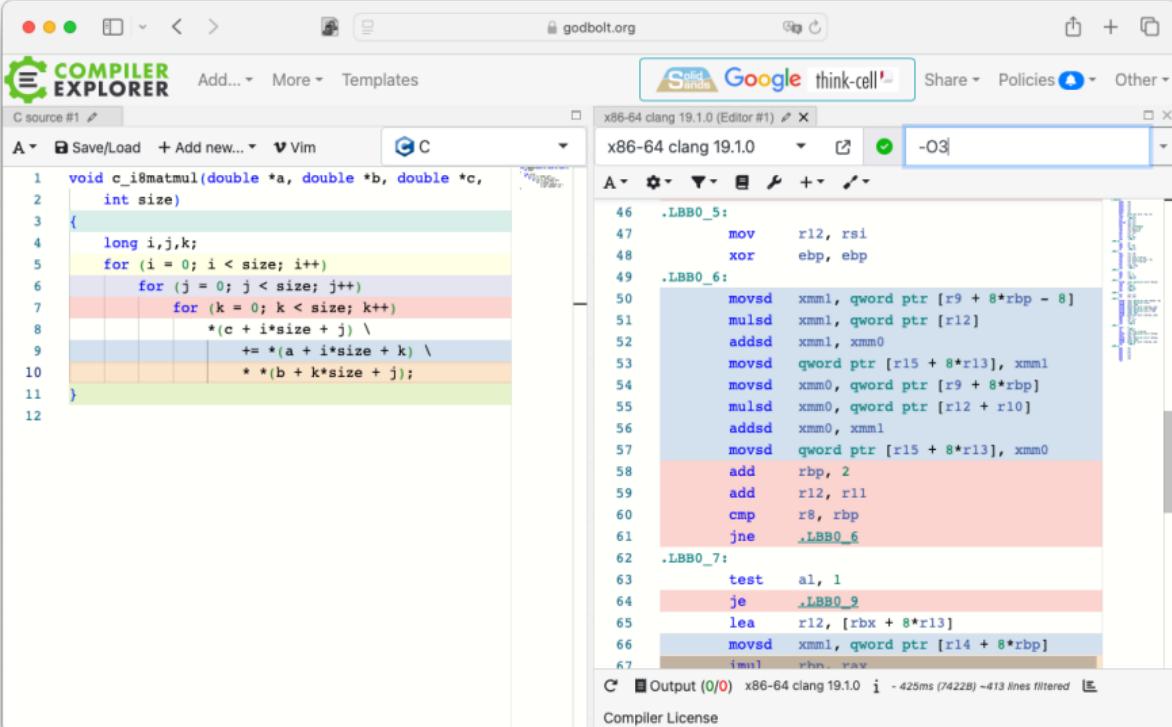
```
1 void c_16matmul(double *a, double *b, double *c,
2                   int size)
3 {
4     long i,j,k;
5     for (i = 0; i < size; i++)
6         for (j = 0; j < size; j++)
7             for (k = 0; k < size; k++)
8                 *(c + i*size + j) \
9                 += *(a + i*size + k) \
10                * *(b + k*size + j);
11 }
```

On the right, the generated assembly code for x86-64 clang 19.1.0 is shown:

```
22     mov    rax, qword ptr [rbp - 56]
23     movsxd rcx, dword ptr [rbp - 28]
24     cmp    rax, rcx
25     jge   .LBB0_8
26     mov    rax, qword ptr [rbp - 8]
27     mov    rcx, qword ptr [rbp - 40]
28     movsxd rdx, dword ptr [rbp - 28]
29     imul   rcx, rdx
30     shl    rcx, 3
31     add    rax, rcx
32     mov    rcx, qword ptr [rbp - 56]
33     movsd  xmm0, qword ptr [rax + 8*rcx]
34     mov    rax, qword ptr [rbp - 16]
35     mov    rcx, qword ptr [rbp - 56]
36     movsxd rdx, dword ptr [rbp - 28]
37     imul   rax, rdx
38     shl    rax, 3
39     add    rax, rdx
40     mov    rax, qword ptr [rbp - 48]
41     movsd  xmm2, qword ptr [rax + 8*rcx]
42     mov    rax, qword ptr [rbp - 24]
43     mov    rax, qword ptr [rbp - 40]
```

The assembly code is color-coded by register: rax (red), rcx (blue), rdx (green), and xmm0/xmm2 (orange). The compiler has generated 43 assembly instructions to perform the matrix multiplication. The output is for clang 19.1.0 with optimization level -O0.

Compilation avec optimisation -O3 (x86-64 "classique")



The image shows the Compiler Explorer interface on godbolt.org. On the left, the C source code for a 16x16 matrix multiplication is displayed:

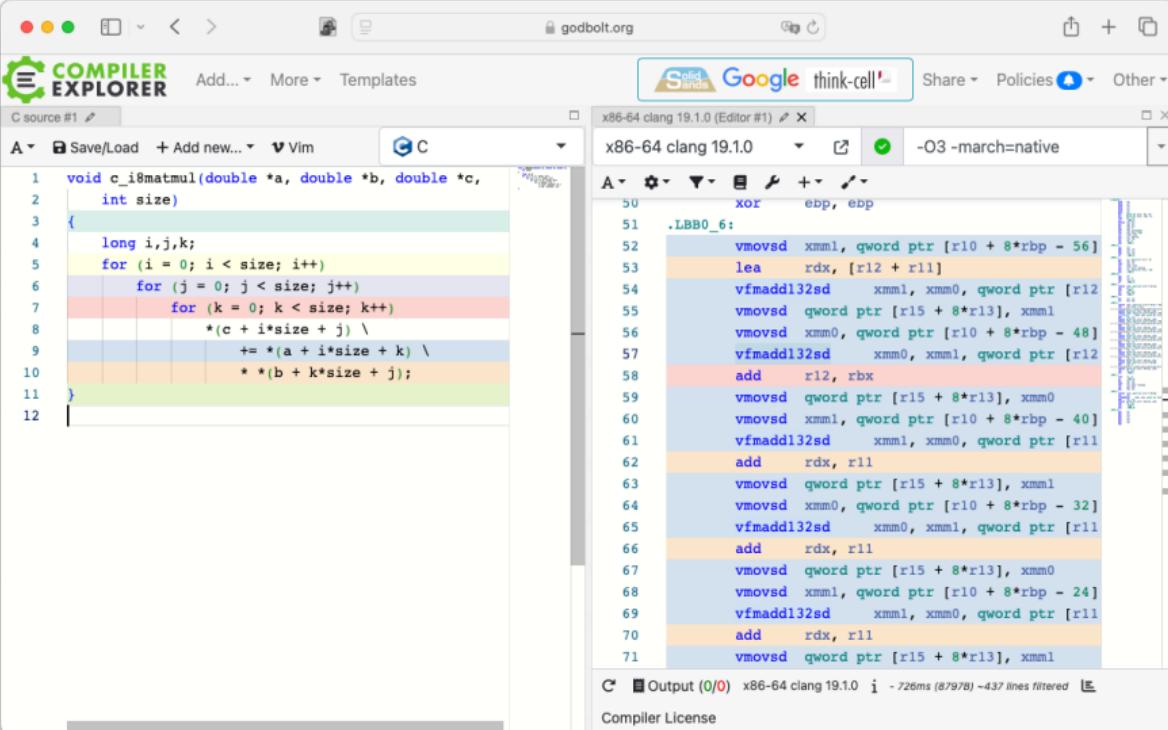
```
1 void c_16matmul(double *a, double *b, double *c,
2                   int size)
3 {
4     long i,j,k;
5     for (i = 0; i < size; i++)
6         for (j = 0; j < size; j++)
7             for (k = 0; k < size; k++)
8                 *(c + i*size + j) \
9                 += *(a + i*size + k) \
10                * *(b + k*size + j);
11 }
```

On the right, the assembly output for x86-64 clang 19.1.0 is shown, generated with the -O3 optimization level. The assembly code is color-coded to match the source code:

```
46 .LBB0_5:
47     mov    r12, rsi
48     xor    ebp, ebp
49 .LBB0_6:
50     movsd xmml, qword ptr [r9 + 8*rbp - 8]
51     mulsd xmml, qword ptr [r12]
52     addsd xmml, xmml0
53     movsd qword ptr [r15 + 8*r13], xmml
54     movsd xmml0, qword ptr [r9 + 8*rbp]
55     mulsd xmml0, qword ptr [r12 + r10]
56     addsd xmml0, xmml1
57     movsd qword ptr [r15 + 8*r13], xmml0
58     add    rbp, 2
59     add    r12, r11
60     cmp    r8, rbp
61     jne    .LBB0_6
62 .LBB0_7:
63     test   al, 1
64     je     .LBB0_9
65     lea    r12, [rbx + 8*r13]
66     movsd xmml, qword ptr [r14 + 8*rbp]
67     imul   rbn, rax
```

The assembly code implements a loop unrolled by 16, using SIMD (xmm registers) for the innermost loop. The compiler has optimized the code to use only 16 registers (r9 to r15, r17 to r23) and has removed the outermost loop's conditionals.

Compilation avec optimisation -O3 (x86-64 spécifique)



Compiler Explorer - godbolt.org

C source #1

```
1 void c_i8matmul(double *a, double *b, double *c,
2                   int size)
3 {
4     long i,j,k;
5     for (i = 0; i < size; i++)
6         for (j = 0; j < size; j++)
7             for (k = 0; k < size; k++)
8                 *(c + i*size + j) \
9                 += *(a + i*size + k) \
10                * *(b + k*size + j);
11 }
```

x86-64 clang 19.1.0 (Editor #1) -O3 -march=native

```
50     xor    ebp, ebp
51     .LBB0_6:
52     vmovsd  xmml, qword ptr [r10 + 8*rbp - 56]
53     lea     rdx, [r12 + r11]
54     vfmadd132sd xmml, xmm0, qword ptr [r12
55     vmovsd  qword ptr [r15 + 8*r13], xmml
56     vmovsd  xmm0, qword ptr [r10 + 8*rbp - 48]
57     vfmadd132sd xmml, xmm0, qword ptr [r12
58     add    r12, rbp
59     vmovsd  qword ptr [r15 + 8*r13], xmm0
60     vmovsd  xmml, qword ptr [r10 + 8*rbp - 40]
61     vfmadd132sd xmml, xmm0, qword ptr [r11
62     add    rdx, r11
63     vmovsd  qword ptr [r15 + 8*r13], xmml
64     vmovsd  xmm0, qword ptr [r10 + 8*rbp - 32]
65     vfmadd132sd xmml, xmm0, qword ptr [r11
66     add    rdx, r11
67     vmovsd  qword ptr [r15 + 8*r13], xmml
68     vmovsd  xmml, qword ptr [r10 + 8*rbp - 24]
69     vfmadd132sd xmml, xmm0, qword ptr [r11
70     add    rdx, r11
71     vmovsd  qword ptr [r15 + 8*r13], xmml
```

C Output (0/0) x86-64 clang 19.1.0 i - 726ms (87978) ~437 lines filtered

Compiler License