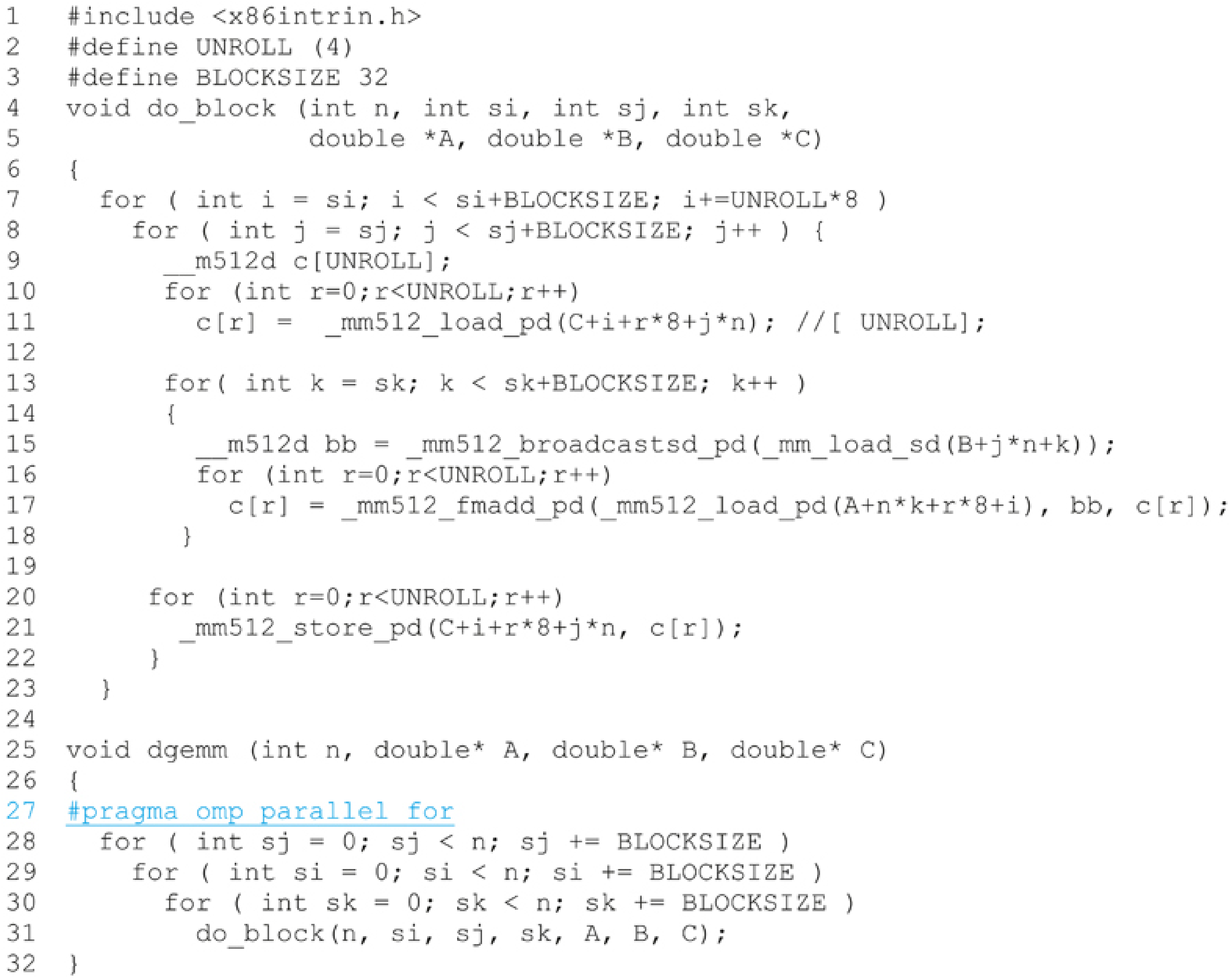

```
1   #include <x86intrin.h>
2   #define UNROLL (4)
3   #define BLOCKSIZE 32
4   void do_block (int n, int si, int sj, int sk,
5                  double *A, double *B, double *C)
6   {
7     for ( int i = si; i < si+BLOCKSIZE; i+=UNROLL*8 )
8       for ( int j = sj; j < sj+BLOCKSIZE; j++ ) {
9         __m512d c[UNROLL];
10        for (int r=0;r<UNROLL;r++)
11          c[r] =  _mm512_load_pd(C+i+r*8+j*n); //[ UNROLL];
12
13        for( int k = sk; k < sk+BLOCKSIZE; k++ )
14        {
15          __m512d bb = _mm512_broadcastsd_pd(_mm_load_sd(B+j*n+k));
16          for (int r=0;r<UNROLL;r++)
17            c[r] = _mm512_fmadd_pd(_mm512_load_pd(A+n*k+r*8+i), bb, c[r]);
18         }
19
20       for (int r=0;r<UNROLL;r++)
21         _mm512_store_pd(C+i+r*8+j*n, c[r]);
22       }
23    }
24
25  void dgemm (int n, double* A, double* B, double* C)
26  {
27  #pragma omp parallel for
28    for ( int sj = 0; sj < n; sj += BLOCKSIZE )
29      for ( int si = 0; si < n; si += BLOCKSIZE )
30        for ( int sk = 0; sk < n; sk += BLOCKSIZE )
31          do_block(n, si, sj, sk, A, B, C);
32  }
```

FIGURE 6.31 OpenMP version of DGEMM from Figure 5.48. Line 27 is the only OpenMP code, making the outermost for loop operate in parallel. This line is the only difference from Figure 5.48.