

Zi Xuan Li

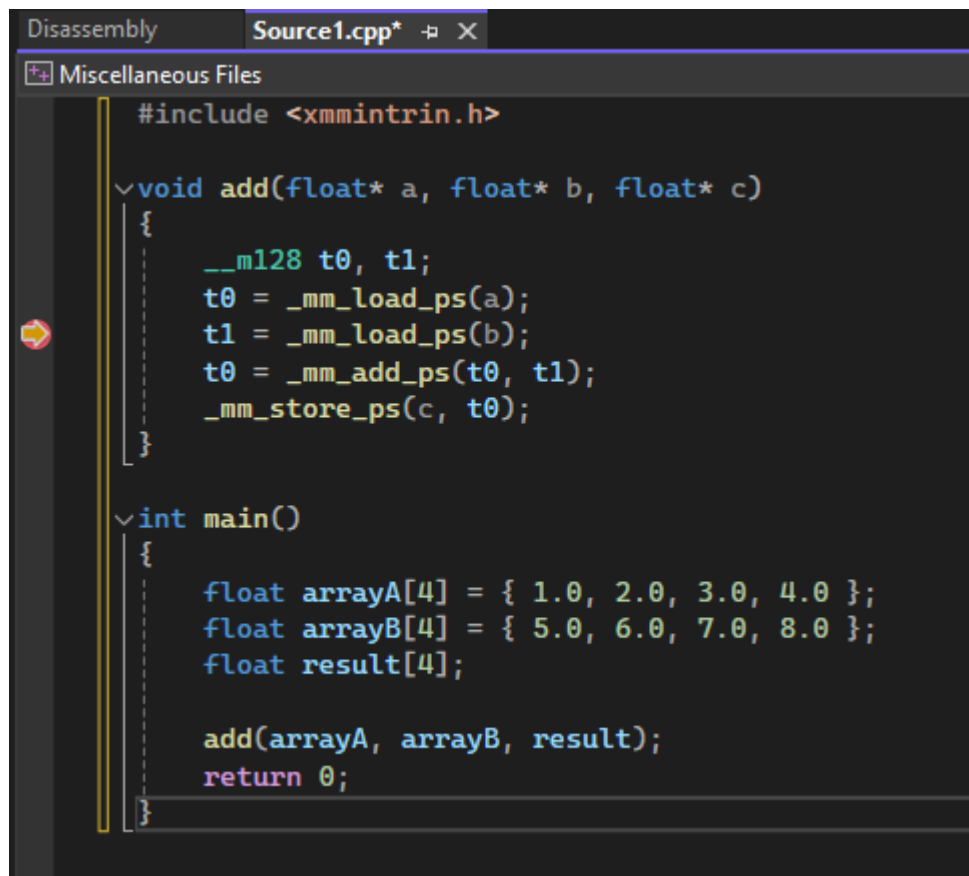
Professor Gertner

CSC 34200/34300

4/14/2024

Homework Intrinsic

The screenshot below shows the source code that I debugged with the Simple Four-Iteration Loop Coded with Intrinsics as shown in the homework guidelines.



The screenshot shows a code editor window with the title 'Source1.cpp*'. The editor displays C++ source code that includes the SSE intrinsic header <xmmintrin.h>. A function 'void add(float* a, float* b, float* c)' is defined, which uses SSE intrinsics to load two float arrays into __m128 registers, add them, and store the result. The 'main' function initializes two float arrays, 'arrayA' and 'arrayB', and calls the 'add' function to compute their sum into 'result'. A yellow vertical bar on the left side of the editor indicates the current line of execution, and a red arrow icon points to the first line of the 'add' function.

```
#include <xmmintrin.h>

void add(float* a, float* b, float* c)
{
    __m128 t0, t1;
    t0 = _mm_load_ps(a);
    t1 = _mm_load_ps(b);
    t0 = _mm_add_ps(t0, t1);
    _mm_store_ps(c, t0);
}

int main()
{
    float arrayA[4] = { 1.0, 2.0, 3.0, 4.0 };
    float arrayB[4] = { 5.0, 6.0, 7.0, 8.0 };
    float result[4];

    add(arrayA, arrayB, result);
    return 0;
}
```

The screenshot below shows the disassembly code for the vector instructions that implement the intrinsic functions.

Disassembly Source1.cpp*

Address: main(void)

Viewing Options

--- C:\Users\zixua\OneDrive\Desktop\C++\CS 342\test\test\Source1.cpp -----

1: #include <xmmintrin.h>

2:

3: void add(float* a, float* b, float* c)

4: {

```
00007FF777ACA1790 4C 89 44 24 18      mov     qword ptr [rsp+18h],r8
00007FF777ACA1795 48 89 54 24 10      mov     qword ptr [rsp+10h],rdx
00007FF777ACA179A 48 89 4C 24 08      mov     qword ptr [rsp+8],rcx
00007FF777ACA179F 55                push    rbp
00007FF777ACA17A0 57                push    rdi
00007FF777ACA17A1 48 81 EC D8 01 00 00 sub     rsp,1D8h
00007FF777ACA17A8 48 8D 6C 24 20      lea     rbp,[rsp+20h]
00007FF777ACA17AD 48 8D 0D 55 F8 00 00 lea     rcx,[__0B5CCB5D_Source1@cpp (07FF77ACB1009h)]
00007FF777ACA17B4 E8 A8 FB FF FF      call    __CheckForDebuggerJustMyCode (07FF77ACA1361h)
```

5: __m128 t0, t1;

6: t0 = _mm_load_ps(a);

```
00007FF777ACA17B9 48 8B 85 D0 01 00 00 mov     rax,qword ptr [a]
00007FF777ACA17C0 0F 10 00           movups  xmm0,xmmword ptr [rax]
00007FF777ACA17C3 0F 29 85 30 01 00 00 movaps  xmmword ptr [rbp+130h],xmm0
00007FF777ACA17CA 0F 28 85 30 01 00 00 movaps  xmm0,xmmword ptr [rbp+130h]
00007FF777ACA17D1 0F 29 45 10       movaps  xmmword ptr [t0],xmm0
```

7: t1 = _mm_load_ps(b);

```
00007FF777ACA17D5 48 8B 85 D8 01 00 00 mov     rax,qword ptr [b]
00007FF777ACA17DC 0F 10 00           movups  xmm0,xmmword ptr [rax]
00007FF777ACA17DF 0F 29 85 60 01 00 00 movaps  xmmword ptr [rbp+160h],xmm0
00007FF777ACA17E6 0F 28 85 60 01 00 00 movaps  xmm0,xmmword ptr [rbp+160h]
00007FF777ACA17ED 0F 29 45 40       movaps  xmmword ptr [t1],xmm0
```

8: t0 = _mm_add_ps(t0, t1);

```
00007FF777ACA17F1 0F 28 45 10       movaps  xmm0,xmmword ptr [t0]
00007FF777ACA17F5 0F 58 45 40       addps   xmm0,xmmword ptr [t1]
00007FF777ACA17F9 0F 29 85 90 01 00 00 movaps  xmmword ptr [rbp+190h],xmm0
00007FF777ACA1800 0F 28 85 90 01 00 00 movaps  xmm0,xmmword ptr [rbp+190h]
00007FF777ACA1807 0F 29 45 10       movaps  xmmword ptr [t0],xmm0
```

9: _mm_store_ps(c, t0);

```
00007FF777ACA180B 48 8B 85 E0 01 00 00 mov     rax,qword ptr [c]
00007FF777ACA1812 0F 28 45 10       movaps  xmm0,xmmword ptr [t0]
00007FF777ACA1816 0F 11 00         movups  xmmword ptr [rax],xmm0
```

10: }

```
00007FF777ACA1819 48 8D A5 B8 01 00 00 lea     rsp,[rbp+1B8h]
00007FF777ACA1820 5F                pop     rdi
00007FF777ACA1821 5D                pop     rbp
00007FF777ACA1822 C3                ret
```

--- No source file -----

```
00007FF777ACA1823 CC      int 3
00007FF777ACA1824 CC      int 3
00007FF777ACA1825 CC      int 3
00007FF777ACA1826 CC      int 3
00007FF777ACA1827 CC      int 3
00007FF777ACA1828 CC      int 3
00007FF777ACA1829 CC      int 3
00007FF777ACA182A CC      int 3
00007FF777ACA182B CC      int 3
00007FF777ACA182C CC      int 3
00007FF777ACA182D CC      int 3
```