Float values behaving differently across the release and debug builds

Asked 16 years, 2 months ago Modified 3 months ago Viewed 3k times

7

My application is generating different floating point values when I compile it in release mode and in debug mode. The only reason that I found out is I save a binary trace log and the one from the release build is ever so slightly off from the debug build, it looks like the bottom two bits of the 32 bit float values are different about 1/2 of the cases.

Would you consider this "difference" to be a bug or would this type of difference be expected. Would this be a compiler bug or an internal library bug.

For example:

```
LEFTPOS and SPACING are defined floating point values.
float def_x;
int xpos;

def_x = LEFTPOS + (xpos * (SPACING / 2));
```

The issue is in regards to the X360 compiler.

c++ c debugging floating-point

Share

Improve this question

Follow

edited Aug 28 at 21:31

Jonas
128k ● 100 ● 326 ● 405

asked Sep 26, 2008 at 20:26

KPexEA
16.8k ● 16 ● 63 ● 79

9 Answers

Sorted by: Highest score (default)

11

✓

Release mode may have a different FP strategy set. There are different floating point arithmetic modes depending on the level of optimization you'd like. MSVC, for example, has strict, fast, and precise modes.

Share Improve this answer Follow

answered Sep 26, 2008 at 20:29

Nick
13.4k ● 17 ● 66 ● 100

4

I know that on PC, floating point registers are 80 bits wide. So if a calculation is done entirely within the FPU, you get the benefit of 80 bits of precision. On the other hand, if an intermediate result is moved out into a normal register and back, it gets truncated to 32 bits, which gives different results.

Now consider that a release build will have optimisations which keep intermediate results in FPU registers, whereas a debug build will probably naively copy intermediate results back and forward between memory and registers - and there you have your difference in behaviour.

I don't know whether this happens on X360 too or not.

Share Improve this answer Follow

answered Sep 26, 2008 at 20:37

Luke Halliwell

7,352 ● 6 ● 32 ● 31

3

It's not a bug. Any floating point operation has a certain imprecision. In Release mode, optimization will change the order of the operations and you'll get a slightly different result. The difference should be small, though. If it's big you might have other problems.

Share Improve this answer Follow

answered Sep 26, 2008 at 20:29

Branan

1,819 ● 15 ● 21

3

I helped a co-worker find a compiler switch that was different in release vs. debug builds that was causing his differences.

Take a look at [/fp (Specify Floating-Point Behavior)](#).

Share Improve this answer Follow

answered Sep 26, 2008 at 20:33

crashmstr

28.5k ● 9 ● 64 ● 80

2

In addition to the different floating-point modes others have pointed out, SSE or similiar vector optimizations may be turned on for release. Converting floating-point arithmetic from standard registers to vector registers can have an effect on the lower bits of your results, as the vector registers will generally be more narrow (fewer bits) than the standard floating-point registers.

Share Improve this answer Follow

answered Sep 26, 2008 at 23:57
Derek Park
46.8k ● 16 ● 59 ● 76

1

Not a bug. This type of difference is to be expected.

For example, some platforms have float registers that use more bits than are stored in memory, so keeping a value in the register can yield a slightly different result compared to storing to memory and re-loading from memory.

Share Improve this answer Follow

answered Sep 26, 2008 at 20:33
Jamie
426 ● 2 ● 7

0

This discrepancy may very well be caused by the compiler optimization, which is typically done in the release mode, but not in debug mode. For example, the compiler may reorder some of the operations to speed up execution, which can conceivably cause a slight difference in the floating point result.

So, I would say most likely it is not a bug. If you are really worried about this, try turning on optimization in the Debug mode.

Share
Improve this answer
Follow

edited Sep 26, 2008 at 20:44

answered Sep 26, 2008 at 20:34
Dima
39.3k ● 14 ● 78 ● 116

0

Like others mentioned, floating point registers have higher precision than floats, so the accuracy of the final result depends on the register allocation.

If you need consistent results, you can make the variables volatile, which will result in slower, less precise, but consistent results.

Share Improve this answer Follow

answered Sep 26, 2008 at 23:51
tfinniga
6,831 ● 3 ● 34 ● 40

If you set a compiler switch that allowed the compiler to reorder floating-point operations, -- e.g. /fp:fast -- then obviously it's not a bug.

0

If you didn't set any such switch, then it's a bug -- the C and C++ standards don't allow the compilers to reorder operations without your permission.

Share Improve this answer Follow

answered Nov 25, 2008 at 21:52

Die in Sente

9,907 ● 4 ● 37 ● 44