

Re: OT : Approximate / fast math libraries ?

Source: <http://linux.derkeiler.com/Mailing-Lists/Fedora/2007-08/msg04925.html>

- *From:* Chris Jones <jonesc@xxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Fri, 31 Aug 2007 20:41:56 +0100
-

Hi,

Thanks for your feedback.

What exactly is your need? Contact me off-list and maybe I can help. Have you profiled your code? I have found that people often do not actually know where their code is spending its time. I once sped up an app which was universally acknowledged to be slow "because it uses floating point." I sped it up 3x.

Yes, I have profiled the code, quite extensively, using the valgrind/calltree application. From this I know this that I'm know I've tidying up this to the point where its hard to find big improvements, the cpu time is fairly well spread around, not isolated in a few places. So am now looking a a few places where math calls are taking more time than I would hope. I'm not going to get factors in speed in the overall application, but I hope in a few places things can be improved a lot locally.

Also, the project is not small, massive in fact, and I'm only writting one small part. If you are interested you can find it here

<http://lhcb-release-area.web.cern.ch/LHCb-release-area/DOC/brunel/releases/latest/doxygen/index.html>

It also has to be supported on a *lot* of hardware. Basically gcc 3.2.3 based Scientific Linux 3 machines, gcc 3.4.6 SL4 machines (32 and 64 bit) and (not my decision), windows VC 7.1. I cannot rely on for instance SSE math calls etc.

Taking an example from another thread, one place I'm trying to understand is where I use atan2 see

<http://www.hep.phy.cam.ac.uk/~jonesc/atan2.png>

for the profiler output. atan2 is taking 50% of the time of this method. Not here I don't need that much precision on the result – say $\pm O(2\pi/100)$. Anything you can suggest here – The code is here

Re: OT : Approximate / fast math libraries ?

<http://www.hep.phy.cam.ac.uk/~jonesc/RichPhotonRecoUsingCKEstiFromRadius.cpp>

(note though its full of internal classes etc...)

cheers Chris

I modified the parsing routines it used, not the floating point.

Mike

--

```
p="p=%c%s%c;main(){printf(p,34,p,34);}";main(){printf(p,34,p,34);}
```

Oppose globalization and One World Governments like the UN.

This message made from 100% recycled bits.

You have found the bank of Larn.

I can explain it for you, but I can't understand it for you.

I speak only for myself, and I am unanimous in that!

--

fedora-list mailing list

fedora-list@xxxxxxxxxx

To unsubscribe: <https://www.redhat.com/mailman/listinfo/fedora-list>