Subject: Re: Intel iMac IDL performance
Posted by JD Smith on Mon, 27 Feb 2006 22:35:31 GMT
View Forum Message <> Reply to Message

On Mon, 27 Feb 2006 15:09:53 -0600, Kenneth Bowman wrote:

```
>  Apple loaned us an Intel Dual-Core iMac for a few days for testing.  Here is a
>  quick comparison:
>
>  Intel system specs:
>     2 GHz Intel Core Duo (2 cpus)
>     2 GB DDR2 SDRAM
>     667 MHz bus
>     OS X 10.4.5
>
>  PowerPC system specs:
>     2.5 GHz PowerPC G5 (4 cpus)
>     2 GB DDR2 SDRAM
>     1.25 GHz bus
>     OS X 10.4.5
>
>  We installed the Mac (PowerPC) version of IDL on both.  The Intel runs IDL via
>  emulation software (Rosetta).
>
>  My IDL benchmark code (dominated by 3-D interpolation, random memory access):
>     PowerPC     31 s
>     Intel iMac  61 s
>
>
>  I played with the IDL demo programs on the Intel iMac and everything that I
>  tried ran fine.  Basic interactive IDL performance is very quick.
>
>  All in all, IDL seems to run fine.  Performance is quite respectable for an
>  emulated system.  Native IDL performance (when available) could be comparable to
>  the G5.
```

Good news. Can you try running your benchmark a few time, Ken?
Rosetta is not an emulator, but a caching code translator. When it
encounters code it has already translated, it simply uses its cached
version of that, which should run somewhat faster, so it's not unusual
to have the second and later runs of a given benchmark speed up. Can
you also run:

```
IDL> time_test3
```

a few times? On my PB G4, that takes 3.6s/0.13s total/geom. mean.
Sadly, I expect the iBook Intel/MacBook Pro to beat these numbers even
under Rosetta. One other good one to try:

```
IDL> a=randomu(sd,100L*!CPU.TPOOL_MIN_ELTS)
IDL> t=systime(1) & a=sqrt(a)/(a>0.5) & print,systime(1)-t
```

which shows how well the threading is working on ~40MB of data. On my
PBG4, this takes 1.8s.

Thanks,

JD

Subject: Re: Intel iMac IDL performance
Posted by Robert Moss on Tue, 28 Feb 2006 02:47:15 GMT

View Forum Message <> Reply to Message

JD Smith wrote:
```
>  On Mon, 27 Feb 2006 15:09:53 -0600, Kenneth Bowman wrote:
>
>>  Apple loaned us an Intel Dual-Core iMac for a few days for testing.  Here is a
>>  quick comparison:
>>
>>  Intel system specs:
>>     2 GHz Intel Core Duo (2 cpus)
>>     2 GB DDR2 SDRAM
>>     667 MHz bus
>>     OS X 10.4.5
>>
>>  PowerPC system specs:
>>     2.5 GHz PowerPC G5 (4 cpus)
>>     2 GB DDR2 SDRAM
>>     1.25 GHz bus
>>     OS X 10.4.5
>>
>>  We installed the Mac (PowerPC) version of IDL on both.  The Intel runs IDL via
>>  emulation software (Rosetta).
>>
>>  My IDL benchmark code (dominated by 3-D interpolation, random memory access):
>>     PowerPC     31 s
>>     Intel iMac  61 s
>>
>>
>>  I played with the IDL demo programs on the Intel iMac and everything that I
>>  tried ran fine.  Basic interactive IDL performance is very quick.
>>
>>  All in all, IDL seems to run fine.  Performance is quite respectable for an
>>  emulated system.  Native IDL performance (when available) could be comparable to
>>  the G5.
```

>
> Good news. Can you try running your benchmark a few time, Ken?
> Rosetta is not an emulator, but a caching code translator. When it
> encounters code it has already translated, it simply uses its cached
> version of that, which should run somewhat faster, so it's not unusual
> to have the second and later runs of a given benchmark speed up. Can
> you also run:
>
> IDL> time_test3
>
> a few times? On my PB G4, that takes 3.6s/0.13s total/geom. mean.
> Sadly, I expect the iBook Intel/MacBook Pro to beat these numbers even
> under Rosetta. One other good one to try:
>
> IDL> a=randomu(sd,100L*!CPU.TPOOL_MIN_ELTS)
> IDL> t=systime(1) & a=sqrt(a)/(a>0.5) & print,systime(1)-t
>
> which shows how well the threading is working on ~40MB of data. On my
> PBG4, this takes 1.8s.
>
> Thanks,
>
> JD
>

Hmm. Maybe your PB is dialed back to save battery power. My Pentium 4m @
2.2 GHz and 512 MB RAM gives this:

```
IDL> a=randomu(sd,100L*!CPU.TPOOL_MIN_ELTS)
IDL> t=systime(1) & a=sqrt(a)/(a>0.5) & print,systime(1)-t
      0.62500000
IDL> time_test3
|TIME_TEST3 performance for IDL 6.2:
|       OS_FAMILY=Windows, OS=Win32, ARCH=x86
| Mon Feb 27 21:45:17 2006
        1    0.0160000 Empty For loop, 2000000 times
        2    0.0309999 Call empty procedure (1 param) 100000 times
        3    0.0470002 Add 200000 integer scalars and store
        4    0.0469999 50000 scalar loops each of 5 ops, 2 =, 1 if)
        5    0.0630000 Mult 512 by 512 byte by constant and store, 30 times
        6     0.140000 Shift 512 by 512 byte and store, 300 times
        7    0.0779998 Add constant to 512x512 byte array, 100 times
        8     0.172000 Add two 512 by 512 byte arrays and store, 80 times
        9     0.109000 Mult 512 by 512 floating by constant, 30 times
       10     0.219000 Shift 512 x 512 array, 60 times
       11     0.187000 Add two 512 by 512 floating images, 40 times
       12    0.0469999 Generate 1000000 random numbers
       13    0.0320001 Invert a 192^2 random matrix
```

```
14    0.0150001 LU Decomposition of a 192^2 random matrix
15    0.0309999 Transpose 384^2 byte, FOR loops
16    0.0790000 Transpose 384^2 byte, row and column ops x 10
17    0.0780001 Transpose 384^2 byte, TRANSPOSE function x 100
18    0.0470002 Log of 100000 numbers, FOR loop
19    0.0469999 Log of 100000 numbers, vector ops 10 times
20     0.203000 131072 point forward plus inverse FFT
21    0.0780001 Smooth 512 by 512 byte array, 5x5 boxcar, 10 times
22    0.0160000 Smooth 512 by 512 floating array, 5x5 boxcar, 5 times
23     0.141000 Write and read 512 by 512 byte array x 40
1.92300=Total Time,      0.062429919=Geometric mean,      23 tests.
```

I did run these a couple of times to remove the memory allocation time
you typically see the first time through. Still, I'm surprised.

Robert Moss, PhD

Subject: Re: Intel iMac IDL performance
Posted by K. Bowman on Tue, 28 Feb 2006 20:28:26 GMT

View Forum Message <> Reply to Message

In article <pan.2006.02.27.22.35.29.385927@as.arizona.edu>,
JD Smith <jdsmith@as.arizona.edu> wrote:

```
>  Good news.  Can you try running your benchmark a few time, Ken?
>  Rosetta is not an emulator, but a caching code translator.  When it
>  encounters code it has already translated, it simply uses its cached
>  version of that, which should run somewhat faster, so it's not unusual
>  to have the second and later runs of a given benchmark speed up.  Can
>  you also run:
>
>  IDL> time_test3
>
>  a few times?  On my PB G4, that takes 3.6s/0.13s total/geom. mean.
>  Sadly, I expect the iBook Intel/MacBook Pro to beat these numbers even
>  under Rosetta.  One other good one to try:
>
>  IDL> a=randomu(sd,100L*!CPU.TPOOL_MIN_ELTS)
>  IDL> t=systime(1) & a=sqrt(a)/(a>0.5) & print,systime(1)-t
>
>  which shows how well the threading is working on ~40MB of data.  On my
>  PBG4, this takes 1.8s.
>
>  Thanks,
>
>  JD
```

Hi, JD.

I ran JD's benchmark, along with time_test3 and my personal benchmark. The
results are summarized here:

   http://idl.tamu.edu/mac_bench.php

I ran all tests 3 times. Variations between individual runs was at the 10%
level. (Re-running did not produce significant changes in speed.)

The Intel iMac is faster than my (relatively new) PowerBook G4, but slower than
a high end G5 desktop.

Multi-threading on the quad-processor G5 seems to work quite well.

I ran a few other non-IDL tests. TeX, with the TeXshop front-end, is amazingly
fast.

Ken

Subject: Re: Intel iMac IDL performance
Posted by Wolf Schweitzer on Wed, 08 Mar 2006 12:47:08 GMT
View Forum Message <> Reply to Message

On my dual-AMD Opteron 254, the minimal amount of time for this test,
replacing !CPU.TPOOL_MIN_ELTS with the standard preset 100000 and
setting the tpool_min_elts to 25, is 0.16 sec.

Wolf.

JD Smith wrote:
> a few times? On my PB G4, that takes 3.6s/0.13s total/geom. mean.
> Sadly, I expect the iBook Intel/MacBook Pro to beat these numbers even
> under Rosetta. One other good one to try:
>
> IDL> a=randomu(sd,100L*!CPU.TPOOL_MIN_ELTS)
> IDL> t=systime(1) & a=sqrt(a)/(a>0.5) & print,systime(1)-t
>
> which shows how well the threading is working on ~40MB of data. On my
> PBG4, this takes 1.8s.

Subject: Re: Intel iMac IDL performance
Posted by JD Smith on Fri, 10 Mar 2006 17:19:03 GMT
View Forum Message <> Reply to Message

On Wed, 08 Mar 2006 13:47:08 +0100, Wolf Schweitzer wrote:

> On my dual-AMD Opteron 254, the minimal amount of time for this test,
> replacing !CPU.TPOOL_MIN_ELTS with the standard preset 100000 and setting
> the tpool_min_elts to 25, is 0.16 sec.

This assumes TPOOL_MIN_ELTS=100000. Setting tpool_min_elts with CPU will
reset this, which will make the size of the vector much smaller, and make
this somewhat artificial (though I don't doubt a factor of 10, really). I
guess I should have put a:

cpu,tpool_min_elts=100000

first, to even the playing field.

JD