Subject: Intel iMac IDL performance
Posted by K. Bowman on Mon, 27 Feb 2006 21:09:53 GMT
View Forum Message <> Reply to Message

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.

Ken Bowman

Subject: Re: Intel iMac IDL performance
Posted by JD Smith on Tue, 28 Feb 2006 17:07:14 GMT
View Forum Message <> Reply to Message

On Mon, 27 Feb 2006 21:47:15 -0500, Robert Moss wrote:

> 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.
>
> 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

>   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.

Yes, IDL performance on G4's is pretty pathetic. Much better on G5's.
The excuse seems to be gcc, which I believe is used to compile IDL on all
Unix platforms. So really, the advantage for IDL from moving to
PowerPC->Intel will be larger than average, especially for laptop owners.

JD

Subject: Re: Intel iMac IDL performance
Posted by JD Smith on Tue, 28 Feb 2006 20:55:49 GMT
View Forum Message <> Reply to Message

On Tue, 28 Feb 2006 14:28:26 -0600, Kenneth Bowman wrote:

> 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.

Thanks, Ken. As anticipated, it seems the PBG4 is worse at running
IDL PPC code than an emulated Intel Core Duo at 20% higher clock
speed. Ouch. You might add (Rosetta) or something to that Intel iMac
column in case your link turns up on Google for the ever popular
"Intel iMac benchmark" search. Also, can you list the IDL version?
When 6.4 or 6.3.x or whatever comes compiled for Intel, we can re-do
things. My bet: faster than the quad-G5 in time_test3, slower (but
not by much) in my thread-heavy test. Makes me want to find someone
to revive the IDLSPEC of years past. Anyone?

JD

Subject: Re: Intel iMac IDL performance
Posted by K. Bowman on Tue, 28 Feb 2006 21:25:00 GMT

View Forum Message <> Reply to Message

In article <pan.2006.02.28.20.55.49.89627@as.arizona.edu>,
JD Smith <jdsmith@as.arizona.edu> wrote:

> You might add (Rosetta) or something to that Intel iMac
> column in case your link turns up on Google for the ever popular
> "Intel iMac benchmark" search. Also, can you list the IDL version?
>
> JD

Done. Thanks for the suggestion, JD.

Ken

Subject: Re: Intel iMac IDL performance
Posted by Richard French on Tue, 28 Feb 2006 22:53:27 GMT
View Forum Message <> Reply to Message

On 2/28/06 3:55 PM, in article pan.2006.02.28.20.55.49.89627@as.arizona.edu,
"JD Smith" <jdsmith@as.arizona.edu> wrote:

> When 6.4 or 6.3.x or whatever comes compiled for Intel, we can re-do
> things.

I was under the impression that getting IDL compiled for Intel (Mac) was not high on RSI's list of priorities - or am I thinking of something else? Does anyone have an authoritative answer on when we might expect this to appear?

Dick French

Subject: Re: Intel iMac IDL performance
Posted by K. Bowman on Wed, 01 Mar 2006 14:44:03 GMT
View Forum Message <> Reply to Message

In article <C02A3E97.2B61%rfrench@wellesley.edu>,
"Richard G. French" <rfrench@wellesley.edu> wrote:

> On 2/28/06 3:55 PM, in article pan.2006.02.28.20.55.49.89627@as.arizona.edu,
> "JD Smith" <jdsmith@as.arizona.edu> wrote:
>
>> When 6.4 or 6.3.x or whatever comes compiled for Intel, we can re-do
>> things.
>
> I was under the impression that getting IDL compiled for Intel (Mac) was not
> high on RSI's list of priorities - or am I thinking of something else? Does
> anyone have an authoritative answer on when we might expect this to appear?
>
> Dick French

From the RSI Platform Support FAQ

When Will RSI Support Mac OS X on Intel?

Now that the new Mac OS X machines running on Intel have been released, RSI can investigate the process of supporting this new system. Necessary software components need to first come together; namely, the third-party software that

RSI uses needs to be fully supported and tested on the new Mac. IDL 6.3 will not
be tested or supported on the Mac for Intel architecture.

IDL will support Mac on the Intel platform as soon as it is feasible.
Unfortunately, it is still too early to know exactly when that will be. We will
keep this FAQ up to date as we know more in the near future.

Cheers, Ken

Subject: Re: Intel iMac IDL performance
Posted by Wolf Schweitzer on Mon, 13 Mar 2006 17:59:00 GMT

View Forum Message <> Reply to Message

JD Smith wrote:

> 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

I did set the vector to 100000 (which rids me of depending on that
assumption with the threadpool minimal elements - setting being constant).

Then I vary TPOOL_MIN_ELTS until I find the fastest speed. I personally
would see no point in recording an artificially slow speed just because
for a given machine / task, the TPOOL_MIN_ELTS is suboptimal. So you'd
first seek the best speed, and record that.

Below my "tweaked" version.

Regards, Wolf.

```
pro jdstest

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

```
print, 'initial result ', ri, ' @ tpool ', !cpu.tpool_min_elts

bs= double(5)
p = 100
pool=0

for n = 0.,30. do begin

cpu,tpool_min_elts=bs^n
p=100
for i = 1,32 do begin

  a=randomu(sd,100L *100000)   ;*!CPU.TPOOL_MIN_ELTS)
  t=systime(1) & a=sqrt(a)/(a>0.5) & r =systime(1)-t

if r lt p then begin
  print, ' found new optimum at ',r, ' seconds @ tpool_min_elts ',bs^n
  p = r
  pool=n
end


end

end

print, ' final results ', p, ' @ tpoolminelts ', bs^pool
print, ' performance gain through tweaking tpool variable : new jd test
runs at percentage of ', p/ri * 100., ' %'

end
```

Subject: Re: Intel iMac IDL performance
Posted by JD Smith on Mon, 20 Mar 2006 16:19:38 GMT

View Forum Message <> Reply to Message

On Mon, 13 Mar 2006 18:59:00 +0100, Wolf Schweitzer wrote:

```
>  JD Smith wrote:
>>  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
>
> I did set the vector to 100000 (which rids me of depending on that
> assumption with the threadpool minimal elements - setting being constant).
>
> Then I vary TPOOL_MIN_ELTS until I find the fastest speed. I personally
> would see no point in recording an artificially slow speed just because
> for a given machine / task, the TPOOL_MIN_ELTS is suboptimal. So you'd
> first seek the best speed, and record that.

Because that's somewhat unrealistic, given how the optimum variable will change depending on the code executing. I think 100000 is a good conservative choice, though it might be worth trying half that and double that.

JD