Subject: Windows/Linux reading binary data - sign problem
Posted by russell.grew on Wed, 09 Jan 2008 03:57:16 GMT
View Forum Message <> Reply to Message

Hello,

Interesting scenario here. I have some code (not written by myself) that reads a bunch of data series from a binary file.

Most of the data series contain positive floating point numbers. One of them contains negative numbers. On windows, this works fine. On linux, whenever the negative numbers should appear, the values have reverted to -2147.48. The series with only positive numbers reads fine on both machines.

There is some manipulation to produce the data series.

Both machines are little endian, checked with
http://www.dfanning.com/tips/endian_machines.html,
using IDL6.3 in both cases. Linux is 64bit gentoo, windows is a 64bit processor running 32 bit windows.

Ideas? Perhaps there is some obvious difference between platforms that I am unaware of?

Thanks.

Russell.

Subject: Re: Windows/Linux reading binary data - sign problem
Posted by russell.grew on Thu, 10 Jan 2008 02:17:38 GMT
View Forum Message <> Reply to Message

Note that if I start IDL in linux with "idl -32" I still have the same problem.

Thanks.

On Jan 10, 10:36 am, RussellGrew <russell.g...@gmail.com> wrote:
> Hi Nigel,
>
> The number is the result of further manipulation. I think the
> manipulation may be the problem here.
>
> I have tried the code on IDL 6.3 64 bit and IDL 6.1 32 bit in linux
> [both little endian machines] - both with the same problem!

>
> Code extract follows. The final values are stored in the 'a' matrix. I
> dont have the documentation for the binary file format handy.
>
> openr,u,fnm, /get_lun
> status = FSTAT(u)
> dd = status.size / (4*60)
>
> a=lonarr(dd,60)
> bdat=byte(1)
> dat=bytarr(4)
>
> for j=0,dd-1 do $;Loop to count total data rows and
> begin
> for i=0,59 do $
> begin
> fdat=double(0.0)
> sgn=1.
> for aa=0,3 do $;data component rows in file.
> begin
> READU,u,bdat
> dat(aa)=bdat
> end
> dat(0)=dat(0)-64 ; take off 2^30
> fdat=dat(0)*256.*256.*256.+dat(1)*256.*256.+dat(2)*256.+dat(3)
> fdat=sgn*fdat
> a[j,i]=fdat
> end
> endfor
> free_lun, u
>
> Any ideas? I assume Linux must handle some part of the above
> differently.
>
> Thanks.
>
> Russell.
>
> On Jan 9, 9:06 pm, Nigel Wade <n...@ion.le.ac.uk> wrote:
>
>> RussellGrew wrote:
>>> Hello,
>
>>> Interesting scenario here. I have some code (not written by myself)
>>> that reads a bunch of data series from a binary file.
>
>>> Most of the data series contain positive floating point numbers. One
>>> of them contains negative numbers. On windows, this works fine. On

>>> linux, whenever the negative numbers should appear, the values have
>>> reverted to -2147.48. The series with only positive numbers reads fine
>>> on both machines.
>
>> That looks very suspicious. It's 32bit -MAXINT, with a decimal scaling factor.
>
>>> There is some manipulation to produce the data series.
>
>> Is the number above what is read by IDL, or the result of the "manipulation"?
>
>>> Both machines are little endian, checked with
>
>> http://www.dfanning.com/tips/endian_machines.html,
>
>>> using IDL6.3 in both cases. Linux is 64bit gentoo, windows is a 64bit
>>> processor running 32 bit windows.
>
>>> Ideas? Perhaps there is some obvious difference between platforms that
>>> I am unaware of?
>
>> There shouldn't be. What method are you using to read the floating point binary
>> data? Are you running a 32bit or 64bit version of IDL on Linux, and are the
>> floating point numbers 32bit or 64bit (float or double)? READU should work the
>> same on all platforms provided the data is in the correct machine format and
>> you ask it to read floats/doubles.
>
>> For example, this writes a 32bit and 64bit floating point values to a file, then
>> reads them back. The platform is 64bit Linux and IDL is 64bit.
>
>> IDL Version 6.4 (linux x86_64 m64). (c) 2007, ITT Visual Information Solutions
>> IDL> a=float(-32.0)
>> IDL> b=double(-64)
>> IDL> openw,1,'tmp.tmp'
>> IDL> writeu,1,a,b
>> IDL> close,1
>> IDL> openr,2,'tmp.tmp'
>> IDL> c=float(1)
>> IDL> d=double(1)
>> IDL> readu,2,c,d
>> IDL> print,c,d
>>   -32.0000   -64.000000
>> IDL> exit
>
>> I can also read it back using 32bit IDL:
>
>> IDL Version 6.4 (linux x86 m32). (c) 2007, ITT Visual Information Solutions
>> IDL> c=float(1)
>> IDL> d=double(1)

>> IDL> openr,2,'tmp.tmp'
>> IDL> readu,2,c,d
>> IDL> print,c,d
>>   -32.0000   -64.000000
>
>> If I transfer the binary file tmp.tmp to a 32bit Windows machine I can still
>> read it using the same code.
>
>> --
>> Nigel Wade, System Administrator, Space Plasma Physics Group,
>>   University of Leicester, Leicester, LE1 7RH, UK
>> E-mail :   n...@ion.le.ac.uk
>> Phone :   +44 (0)116 2523548, Fax : +44 (0)116 2523555

Subject: Re: Windows/Linux reading binary data - sign problem
Posted by Nigel Wade on Thu, 10 Jan 2008 16:42:57 GMT
View Forum Message <> Reply to Message

RussellGrew wrote:

> Hi Nigel,
>
> The number is the result of further manipulation. I think the
> manipulation may be the problem here.
>
> I have tried the code on IDL 6.3 64 bit and IDL 6.1 32 bit in linux
> [both little endian machines] - both with the same problem!
>
> Code extract follows. The final values are stored in the 'a' matrix. I
> dont have the documentation for the binary file format handy.
>
>
> openr,u,fnm, /get_lun
> status = FSTAT(u)
> dd = status.size / (4*60)
>
> a=lonarr(dd,60)
> bdat=byte(1)
> dat=bytarr(4)
>
> for j=0,dd-1 do $;Loop to count total data rows and
> begin
> for i=0,59 do $
>   begin
>   fdat=double(0.0)

This line is redundant.

> sgn=1.

This line is redundant (sgn is never assigned other than here).

> for aa=0,3 do $;data component rows in file.
> begin
> READU,u,bdat
> dat(aa)=bdat
> end

This loop is redundant, just read the byte array.

> dat(0)=dat(0)-64 ; take off 2^30

What is the purpose of this line? What data format does your binary file contain? It's not masking a bit in dat(0), it is subtracting a fixed value and creating a (possibly negative) INT.

> fdat=dat(0)*256.*256.*256.+dat(1)*256.*256.+dat(2)*256.+dat(3)

This appears to be building a floating point number from a 32bit big-endian integer (apart from the "take off 2^30" part). Why make a float when a long would do the job with no inaccuracy?

> fdat=sgn*fdat

This line is redundant, sgn is always 1.0.

> a[j,i]=fdat

Now you convert your floating point number back to an integer...

I don't see how the above code could generate the floating point number -2147.48. There is no fractional part.

> end
> endfor
> free_lun, u
>
> Any ideas? I assume Linux must handle some part of the above
> differently.

I would handle the entire process differently...

--
Nigel Wade, System Administrator, Space Plasma Physics Group,
  University of Leicester, Leicester, LE1 7RH, UK

E-mail : nmw@ion.le.ac.uk
Phone : +44 (0)116 2523548, Fax : +44 (0)116 2523555

Subject: Re: Windows/Linux reading binary data - sign problem
Posted by David Fanning on Thu, 10 Jan 2008 17:46:09 GMT
View Forum Message <> Reply to Message

Nigel Wade writes:

> I would handle the entire process differently...

Welcome to my world! ;-)

Cheers,

David
--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming (www.dfanning.com)
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Windows/Linux reading binary data - sign problem
Posted by russell.grew on Thu, 10 Jan 2008 23:02:24 GMT
View Forum Message <> Reply to Message

If I had the format of the binary files I would also probably handle
the process differently. I didn't write the above code.

At this stage, I am just trying to get the thing working.

Thanks anyway.

Russell.

On Jan 11, 4:46 am, David Fanning <n...@dfanning.com> wrote:
> Nigel Wade writes:
>> I would handle the entire process differently...
>
> Welcome to my world! ;-)
>
> Cheers,
>
> David

> --
> David Fanning, Ph.D.
> Fanning Software Consulting, Inc.
> Coyote's Guide to IDL Programming (www.dfanning.com)
> Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Windows/Linux reading binary data - sign problem
Posted by Nigel Wade on Fri, 11 Jan 2008 09:10:29 GMT
View Forum Message <> Reply to Message

RussellGrew wrote:

> If I had the format of the binary files I would also probably handle
> the process differently. I didn't write the above code.

I appreciate that, I do wonder what the person who did write it was thinking when they wrote it...

I don't see where there is anything in the above code which would perform differently in Windows than Linux. I also don't see how it could produce the results you are getting. Are you sure that is really the code which is being executed?

>
> At this stage, I am just trying to get the thing working.
>
> Thanks anyway.
>

Getting code to correctly read binary data without knowing what format the binary data is in is a futile task. How do you know when you have got it right if you don't know what right is?

--
Nigel Wade, System Administrator, Space Plasma Physics Group,
University of Leicester, Leicester, LE1 7RH, UK
E-mail : nmw@ion.le.ac.uk
Phone : +44 (0)116 2523548, Fax : +44 (0)116 2523555
