
Subject: Re: inexplicable LONG() - behaviour
Posted by [steinhh](#) on Wed, 07 Sep 1994 09:19:08 GMT
[View Forum Message](#) <> [Reply to Message](#)

In article <34hrvh\$6sq@jurpool0.rz.uni-frankfurt.de>, frank@chaos.uni-frankfurt.dbp.de (Frank Hoffsuemmer) writes:

```
|> Hello,  
|>  
|> I'm using IDL 3.1.1 (no update in sight :-( ) under HP-UX.  
|> And there are some strange things happening....  
|> Of course, these are just things that I understand wrong :), so could someone  
|> please explain this behaviour:  
|>  
|>  
|> IDL. Version 3.1.1 (hp-ux hp_pa).  
|> Copyright 1989-1993, Research Systems, Inc.  
|> All rights reserved. Unauthorized reproduction prohibited.  
|> Installation number: 3063.  
|> Licensed for use by: Johann Wolfgang Goethe-Universitaet, HRZ  
|>  
|> X-IDL> print, long(1231231434.1)  
|> 1231231488  
|>  
|> The last two digits differ quite a bite. O.k. the number is too long,  
|> but why isn't there an error-message like in this example (same number but  
|> first digit):
```

Because the value 1231231434.1 isn't possible to store in a FLOAT
(not enough precision). (It's actually stored as 1231231488!)
The error occurs in converting the textual digits into a float,
which is not normally considered an error, as long as it's the
float precision causing it.

Try print,long(1231231434.1d)

Or, print,'\$(f12.1)',1231231434.1

```
|>  
|> X-IDL> print, long(2231231434.1)  
|> % Program caused arithmetic error: Floating overflow  
|> % Detected at $MAIN$ (LONG).  
|> 2139095040  
|>  
|> there still is an result, and the result is still wrong, but at least there's  
|> a message indicating that something went wrong.  
|>
```

Such a large number can be stored in a FLOAT (because of the FLOAT-ing

point exponent notation), but not in a LONG (just not enough bits).
Thus, a significant error occurs, and it's reported.

```
|> Furthermore, I programmed a little routine, that is supposed to return  
|> a rounded value of it's argument (included at the end of this article).  
|> The argument can be integer, long, double, and the result is always long. e.g.  
|>  
|> X-IDL> PRINT, ROUNDUP(12345.9867)  
|> 13000  
|> X-IDL> PRINT, ROUNDUP(999.67)  
|> 1000  
|> X-IDL> PRINT, ROUNDUP(53645)  
|> 54000  
|>  
|> - just to give you an impression.  
|> Using this function I have the following problem:  
|>  
|> X-IDL> PRINT, ROUNDUP(101.1)  
|> 109
```

Your problem is that what's calculated in your procedure **isn't** the
value double(110.0), its (change of the print lines to
PRINT, '\$(A,f19.15)', "DOUBLE: ", DOUBLE(result)
PRINT, "LONG : ", LONG(result+0.0001)

And voila:

```
RESULT:    110.00000  
DOUBLE: 109.999999999999970  
LONG :     110
```

Now, it's the DOUBLE that's wrong... :-)

It's all correct, if you allow for the finite-precision nature of
things.

Stein Vidar

Subject: Re: inexplicable LONG() - behaviour
Posted by [thompson](#) on Wed, 07 Sep 1994 13:51:09 GMT
[View Forum Message](#) <> [Reply to Message](#)

frank@chaos.uni-frankfurt.dbp.de (Frank Hoffsuemmer) writes:

> Hello,

> I'm using IDL 3.1.1 (no update in sight :-() under HP-UX.
> And there are some strange things happening....
> Of course, these are just things that I understand wrong :), so could someone
> please explain this behaviour:

```
> IDL. Version 3.1.1 (hp-ux hp_pa).  
> Copyright 1989-1993, Research Systems, Inc.  
> All rights reserved. Unauthorized reproduction prohibited.  
> Installation number: 3063.  
> Licensed for use by: Johann Wolfgang Goethe-Universitaet, HRZ  
>  
> X-IDL> print, long(1231231434.1)  
> 1231231488
```

This behavior exists in IDL 3.5 as well.

The reason for this is quite simple. The argument to the LONG function is a floating point number. The way you've phrased the statement, it's a *single* *precision* floating point number. Thus, it's already lost accuracy before you even get to converting it to a long integer. You can see this by entering the following command:

```
IDL> print, 1231231434.1, format='(f20.1)'  
1231231488.0
```

On the other hand, if you define the floating point constant to be double precision, then it works fine, i.e.

```
IDL> print, long(1231231434.1d0)  
1231231434
```

Bill Thompson
