

Re: Why does python break IEEE 754 for 1.0/0.0 and 0.0/0.0?

Source: <http://coding.derkeiler.com/Archive/Python/comp.lang.python/2005-07/msg02159.html>

- *From:* Tim Peters <tim.peters@xxxxxxxx>
 - *Date:* Thu, 14 Jul 2005 16:20:04 -0400
-

[Tim Peters]

....
>> What does your platform C return for the integer expression
>> 42/0? Is any other outcome "wrong"?

[Grant Edwards]

> I guess I thought it was obvious from my reference to IEEE 754
> that I was referring to floating point operations.

Yes, that was obvious. Since I thought my point would be equally obvious, I won't spell it out <0.7 wink>.

....

>> Note that support for 754 was rare on Python platforms at the
>> time Python was designed, and nobody mentioned 754 support as
>> even a vague desire in those days.

> I often forget how old Python is. Still, I've been using IEEE
> floating point in C programs (and depending on the proper
> production and handling of infinities and NaNs) for more than
> 20 years now. I had thought that Python might have caught up.

It has not. Please see the other thread I mentioned.

>> In the absence of user interest, and in the absence of HW
>> support for NaNs or infinities on most Python platforms,

> Really?

Yes, but looks like you didn't finish reading the sentence. Here's the rest, with emphasis added:

>> the decision to raise an exception was quite sensible **AT THE TIME**.

You may have forgotten how much richer the "plausible HW" landscape was at the time too. I was deeply involved in implementing Kendall Square Research's HW and SW 754 story at the time, and it was all

Re: Why does python break IEEE 754 for 1.0/0.0 and 0.0/0.0?

quite novel, with little prior art to draw on to help resolve the myriad language issues 754 didn't address (e.g., what should Fortran's 3-branch Arithmetic IF statement do if fed a NaN? there were hundreds of headaches like that, and no cooperation among compiler vendors since the language standards ignored 754). The C standards didn't mention 754 until C99, and then left all support optional (up to the compiler implementer whether to do it). That didn't help much for a bigger reason: major C vendors (like Microsoft and Borland) are still ignoring C99. "Subset" HW implementations of 754 were also common, like some that didn't support denorms at all, others that didn't implement the non-default rounding modes, some that ignored signed zeroes, and several that implemented 754 endcases by generating kernel traps to deal with infinities and NaNs, making them so much slower than normal cases that users avoided them like death.

If I had to bet at the time, I would have put my money on 754 dying out due to near-universal lack of language support, and incompatible HW implementations. Most programming languages still have no sane 754 story, but the remarkable dominance of the Pentium architecture changed everything on the HW side.

> I would have guessed that most Python platforms are
> '586 or better IA32 machines running either Windows or Linux.

Today, yes, although there are still Python users on many other OSes and architectures. Most of the latter support 754 too now.

> They all have HW support for NaNs and Infinities.

Yes, Intel-based boxes certainly do (and have for a long time), and so do most others now.

.

• *Follow-Ups:*

- ◆ [Re: Why does python break IEEE 754 for 1.0/0.0 and 0.0/0.0?](#)
◇ From: Grant Edwards

• *References:*

- ◆ [Why does python break IEEE 754 for 1.0/0.0 and 0.0/0.0?](#)
◇ From: Grant Edwards
- ◆ [Re: Why does python break IEEE 754 for 1.0/0.0 and 0.0/0.0?](#)
◇ From: Tim Peters
- ◆ [Re: Why does python break IEEE 754 for 1.0/0.0 and 0.0/0.0?](#)
◇ From: Grant Edwards

- Prev by Date: [Re: all possible combinations](#)
- Next by Date: [Re: Tkinter Button widget](#)
- Previous by thread: [Re: Why does python break IEEE 754 for 1.0/0.0 and 0.0/0.0?](#)
- Next by thread: [Re: Why does python break IEEE 754 for 1.0/0.0 and 0.0/0.0?](#)

Re: Why does python break IEEE 754 for 1.0/0.0 and 0.0/0.0?

- Index(es):

- ◆ *Date*

- ◆ *Thread*