# jboner / latency.txt

Last active 6 hours ago • Report abuse

Star

Code Revisions 18 Stars 5,000+ Forks 1,608

Latency Numbers Every Programmer Should Know

latency.txt

```
Latency Comparison Numbers (~2012)
----------------------------------
L1 cache reference                           0.5 ns
Branch mispredict                            5   ns
L2 cache reference                           7   ns                      14x L1 cache
Mutex lock/unlock                           25   ns
Main memory reference                      100   ns                      20x L2 cache, 200
Compress 1K bytes with Zippy             3,000   ns        3 us
Send 1K bytes over 1 Gbps network       10,000   ns       10 us
Read 4K randomly from SSD*             150,000   ns      150 us          ~1GB/sec SSD
Read 1 MB sequentially from memory     250,000   ns      250 us
Round trip within same datacenter      500,000   ns      500 us
Read 1 MB sequentially from SSD*     1,000,000   ns    1,000 us    1 ms  ~1GB/sec SSD, 4X
Disk seek                           10,000,000   ns   10,000 us   10 ms  20x datacenter ro
Read 1 MB sequentially from disk    20,000,000   ns   20,000 us   20 ms  80x memory, 20X S
Send packet CA->Netherlands->CA    150,000,000   ns  150,000 us  150 ms

Notes
-----
1 ns = 10^-9 seconds
1 us = 10^-6 seconds = 1,000 ns
1 ms = 10^-3 seconds = 1,000 us = 1,000,000 ns

Credit
------
By Jeff Dean:               http://research.google.com/people/jeff/
Originally by Peter Norvig: http://norvig.com/21-days.html#answers

Contributions
-------------
'Humanized' comparison:  https://gist.github.com/hellerbarde/2843375
Visual comparison chart: http://i.imgur.com/k0t1e.png
```

**dominictarr** commented on May 31, 2012

need a solar system type visualization for this, so we can really appreciate the change of scale.

**jboner** commented on May 31, 2012 Owner Author

I agree, would be fun to see. :-)

**pmanvi** commented on May 31, 2012

useful information & thanks

**marianposaceanu** commented on May 31, 2012

Looks nice kudos !
One comment about the Branch mispredict: if the cpu architecture is based on P4 or Bulldozer that would result in 20-30+ cycles on a mispredict that would translate to a much bigger number (and they do mispredict) :)

For SSD's would be something like:
Disk seek: 100 000 ns

**preinheimer** commented on May 31, 2012

Latency numbers between large cities: https://wondernetwork.com/pings/

**alexismo** commented on May 31, 2012

@**preinheimer** Asia & Australasia have it bad.

**gandalfar** commented on May 31, 2012

From the same author: http://videolectures.net/wsdm09_dean_cblirs/

**Eronarn** commented on May 31, 2012

"Latency numbers every programmer should know" - yet naturally, it has no information about humans!

http://biae.clemson.edu/bpc/bp/lab/110/reaction.htm

**hellerbarde** commented on May 31, 2012

maybe you want to incorporate some of this: https://gist.github.com/2843375

**christopherscott** commented on May 31, 2012

Curious to see numbers for SSD read time

**klochner** commented on May 31, 2012

I think the reference you want to cite is here: http://norvig.com/21-days.html#answers

**lucasces** commented on May 31, 2012

This remind me of this Grace Hopper's video about Nanoseconds. Really worthy.
http://www.youtube.com/watch?v=JEpsKnWZrJ8

**mikea** commented on May 31, 2012

I find comparisons much more useful than raw numbers: https://gist.github.com/2844130

**briangordon** commented on May 31, 2012

I'm surprised that mechanical disk reads are only 80x the speed of main memory reads.

**marianposaceanu** commented on May 31, 2012

my version : https://gist.github.com/2842457 includes SSD number, would love some more

**newphoenix** commented on May 31, 2012

Does L1 and L2 cache latency depends on processor type? and what about L3 cache.

**marianposaceanu** commented on May 31, 2012

Ofc it does ... those are averages I think.

**cayblood** commented on May 31, 2012

Would be nice to right-align the numbers so people can more easily compare orders of magnitude.

**jboner** commented on May 31, 2012 Owner Author

Good idea. Fixed.

**jhclark** commented on May 31, 2012

And expanded even a bit more: https://gist.github.com/2845836 (SSD numbers, relative comparisons, more links)

**nicowilliams** commented on May 31, 2012

TLB misses would be nice to list too, so people see the value of large pages...

Context switches (for various OSes), ...

Also, regarding packet sends, that must be latency from send initiation to send completion -- I assume.

If you're going to list mutex lock/unlock, how about memory barriers?

Thanks! This is quite useful, particularly for flogging at others.

**lry** commented on Jun 1, 2012

Quick pie chart of data with scales in time (1 sec -> 9.5 years) for fun.

Spreadsheet with chart

**vickychijwani** commented on Jun 1, 2012

"Read 1 MB sequentially from disk - 20,000,000 ns". Is this with or without disk seek time?

**pgroth** commented on Jun 1, 2012

I made a fusion table for this at:
https://www.google.com/fusiontables/DataSource?snapid=S523155yioc

Maybe be helpful for graphing, etc. Thanks for putting this together

**jboner** commented on Jun 1, 2012 Owner Author

Cool. Thanks.
Thanks everyone for all the great improvements.

**ayshen** commented on Jun 2, 2012

Here is a chart version. It's a bit hard to read, but I hope it conveys the perspective.
http://i.imgur.com/k0t1e.png

**gchatelet** commented on Jun 2, 2012

It would also be very interesting to add memory allocation timings to that : )

**PerWiklander** commented on Jun 5, 2012

How long does it take before this shows up in XKCD?

**talltyler** commented on Jun 5, 2012

You guys are talking about is the powers of ten http://vimeo.com/819138

**BillKress** commented on Jun 5, 2012

If it does show up on xkcd it will be next to a gigantic "How much time it takes for a human to react to any results", hopefully with the intent to show people that any USE of this knowledge should be tempered with an understanding of what it will be used for--possibly showing how getting a bit from the cache is pretty much identical to getting a bit from china when it comes to a single fetch of information to show a human being.

**hellerbarde** commented on Jun 5, 2012

**@BillKress** yes, this is specifically for Programmers, to make sure they have an understanding about the bottlenecks involved in programming. If you know these numbers, you know that you need to cut down on disk access before cutting down on in-memory shuffling.
If you don't properly follow these numbers and what they stand for, you will make programs that don't scale well. That is why they are important on their own and (in this context) should not be dwarfed by human reaction times.

**PerWiklander** commented on Jun 5, 2012

**@BillKress** If we were only concerned with showing information to a single human being at a time we could just as well shut down our development machines and go out into the sun and play. This is about scalability.

**klochner** commented on Jun 5, 2012

this is getting out of hand, how do i unsubscribe from this gist?

**gemclass** commented on Jun 7, 2012

Saw this via @smashingmag . While you guys debate the fit for purpose, here is another visualization of your quick reference latency data with Prezi ow.ly/bnB7q

**briangordon** commented on Jul 3, 2012

Does anybody know how to stop receiving notifications from a gist's activity?

**colin-scott** commented on Dec 25, 2012

Here's a tool to visualize these numbers over time:
http://www.eecs.berkeley.edu/~rcs/research/interactive_latency.html

**JensRantil** commented on Jan 6, 2013

I just created flash cards for this: https://ankiweb.net/shared/info/3116110484 They can be downloaded using the Anki application: http://ankisrs.net

**JensRantil** commented on Jan 14, 2013

I'm also missing something like "Send 1MB bytes over 1 Gbps network (within datacenter over TCP)". Or does that vary so much that it would be impossible to specify?

**kofemann** commented on Feb 9, 2013

If L1 access is a second, then:

L1 cache reference : 0:00:01
Branch mispredict : 0:00:10
L2 cache reference : 0:00:14
Mutex lock/unlock : 0:00:50
Main memory reference : 0:03:20
Compress 1K bytes with Zippy : 1:40:00
Send 1K bytes over 1 Gbps network : 5:33:20
Read 4K randomly from SSD : 3 days, 11:20:00
Read 1 MB sequentially from memory : 5 days, 18:53:20
Round trip within same datacenter : 11 days, 13:46:40
Read 1 MB sequentially from SSD : 23 days, 3:33:20
Disk seek : 231 days, 11:33:20
Read 1 MB sequentially from disk : 462 days, 23:06:40
Send packet CA->Netherlands->CA : 3472 days, 5:20:00

**kofemann** commented on Feb 9, 2013

You can add LTO4 tape seek/access time, ~ 55 sec, or 55.000.000.000 ns

**metakeule** commented on Jul 29, 2013

I'm missing things like sending 1K via Unix pipe/ socket / tcp to another process.
Has anybody numbers about that?

**shiplunc** commented on Nov 27, 2013

@**metakeule** its easily measurable.

**mnem** commented on Jan 9, 2014

Related page from "Systems Performance" with similar second scaling mentioned by @**kofemann**:
https://twitter.com/rzezeski/status/398306728263315456/photo/1

**izard** commented on May 29, 2014

L1D hit on a modern Intel CPU (Nehalem+) is at least 4 cycles. For a typical server/desktop at 2.5Ghz it is at least 1.6ns.
Fastest L2 hit latency is 11 cycles(Sandy Bridge+) which is 2.75x not 14x.
May be the numbers by Norwig were true at some time, but at least caches latency numbers are pretty constant since Nehalem which was 6 years ago.

**richa03** commented on Aug 22, 2014

Please note that Peter Norvig first published this expanded version (at that location - http://norvig.com/21-days.html#answers) ~JUL2010 (see wayback machine). Also, note that it was "Approximate timing for various operations on a typical PC".

**pdjonov** commented on Oct 3, 2014

One light-nanosecond is roughly a foot, which is considerably less than the distance to my monitor right now. It's kind of surprising to realize just how much a CPU can get done in the time it takes light to traverse the average viewing distance...

**junhe** commented on Jan 16, 2015

@**jboner**, I would like to cite some numbers in a formal publication. Who is the author? Jeff Dean? Which url should I cite? Thanks.

**weidagang** commented on Jan 26, 2015

I'd like to see the number for "Append 1 MB to file on disk".

**dhartford** commented on Mar 11, 2015

The "Send 1K bytes over 1 Gbps network" doesn't feel right, if you were comparing the 1MB sequential read of memory, SSD, Disk, the Gbps network for 1MB would be faster than disk (x1024), that doesn't feel right.

**leotm** commented on May 2, 2015

A great solar system type visualisation: http://joshworth.com/dev/pixelspace/pixelspace_solarsystem.html

**ali** commented on Sep 14, 2015

I turned this into a set of flashcards on Quizlet: https://quizlet.com/_1iqyko

**misgeatgit** commented on Dec 11, 2015

Can you update the the Notes section with the following
1 ns = 10^-9 seconds
1 ms = 10^-3 seconds

Thanks.

**jboner** commented on Dec 13, 2015 Owner Author

@**misgeatgit** Updated

**juhovuori** commented on Dec 25, 2015

Zippy is nowadays called snappy. Might be worth updating. Tx for the gist.

**georgevreilly** commented on Jan 10, 2016

Several of the recent comments are spam. The links lead to sites in India which have absolutely nothing to do with latency.

**wenjianhn** commented on Jan 12, 2016

Are there any numbers about latency between NUMA nodes?

**vitaut** commented on Jan 31, 2016

Sequential SSD speed is actually more like 500 MB/s, not 1000 MB/s for SATA drives (http://www.tomshardware.com/reviews/ssd-recommendation-benchmark,3269.html).

**BruceGooch** commented on Mar 9, 2016

You really should cite the folks at Berkeley. Their site is interactive, has been up for 20 years, and it is where you "sourced" your visualization. http://www.eecs.berkeley.edu/~rcs/research/interactive_latency.html

**julianeden** commented on Mar 10, 2016

Question~ do these numbers not vary from one set of hardware to the next? How can these be accurate for all different types of RAM, CPU, motherboard, hard drive, etc?

(I am primarily a front-end JS dev, I know little-to-nothing about this side of programming, where one must consider numbers involving RAM and CPU. Forgive me if I'm missing something obvious.)

consider numbers involving RAM and CPU. Forgive me if I'm missing something obvious.)

**jlleblanc** commented on Mar 21, 2016

The link to the animated presentation is broken, here's the correct one: http://prezi.com/pdkvgys-r0y6/latency-numbers-for-programmers-web-development

**keenkit** commented on Aug 16, 2016

Love this one.

**profuel** commented on Oct 5, 2016

Mentioned `gist` : https://gist.github.com/2843375 is private or was removed.
can someone restore it?
Thanks!

**trans** commented on Oct 9, 2016

It would be nice to be able to compare this to computation times -- How long to do an add, xor, multiply, or branch operation?

**mpron** commented on Oct 12, 2016 • edited

Last year, I came up with this concept for an infographic illustrating these latency numbers with time analogies (if 1 CPU cycle = 1 second). Here was the result: http://imgur.com/8LIwV4C

**pawel-dubiel** commented on Jan 29, 2017 • edited

Most of these number were valid in 2000-2001, right now some of these numbers are wrong by an order of magnitude. ( especially reading from main memory, as DRAM bandwidth doubles every 3 years )

**maranomynet** commented on Jan 31, 2017

µs , not us

 **GLMeece** commented on Jan 31, 2017 • edited

I realize this was published some time ago, but the following URLs are no longer reachable/valid:

- https://gist.github.com/2843375
- http://prezi.com/pdkvgys-r0y6/latency-numbers-for-programmers-web-development/latency.txt

However, the second URL should now be: https://prezi.com/pdkvgys-r0y6/latency-numbers-for-programmers-web-development/

Oh, and **@mpron** - nice!

 **JustinNazari** commented on Jan 31, 2017

Thank you **@jboner**

 **GLMeece** commented on Jan 31, 2017

Note: I created my own "fork" of this.

 **ValerieAnne563** commented on May 2, 2017

Thank you **@GLMeece**

 **orestotel** commented on Jun 11, 2017

Google it

 **knbknb** commented on Jun 24, 2017 • edited

Median human reaction time (to some stimulus showing up on a screen): 270 ms
(value probably increases with age)
https://www.humanbenchmark.com/tests/reactiontime/statistics

**SonalJha** commented on Aug 15, 2017

Awesome info. Thanks!

**deadjdona** commented on Sep 8, 2017

https://people.eecs.berkeley.edu/~rcs/research/interactive_latency.html

**keynan** commented on Sep 22, 2017 • edited

Could you please add printf & fprintf to this list

**awilkins** commented on Oct 13, 2017

Heh, imagine this transposed into human distances.

1ns = 1 step, or 2 feet.

L1 cache reference = reaching 1 foot across your desk to pick something up
Datacentre roundtrip = 94 mile hike.
Internet roundtrip (California to Netherlands) = Walk around the entire earth. Wait! You're not done. Then walk from London, to Havana. Oh, and then to Jacksonville, Florida. Then you're done.

**benirule** commented on Oct 23, 2017

The last link is giving a 404

**ahartmetz** commented on Nov 16, 2017

The numbers "Read 1 MB sequentially from memory" mean a memory bandwidth of 4 GB/s. That is a very old number. Can you update it? The time should be roughly 1/5th - one core can do about 20 GB/s today, all cores of a 4 or 8 core about 40 GB/s together. I remember seeing 18-19 GB/s in memtest86 for single core on my Ryzen 1800X and there are several benchmarks floating around where all cores do about 40 GB/s. It is very hard to find anything on the web about single core memory bandwidth...

**jamalahmedmaaz** commented on Jan 25, 2018

Good information, thanks.

**ryazo** commented on Jan 28, 2018 • edited

http://ram.userbenchmark.com/
Ram has gotten slightly faster. It is 70 ns now.

Edit: I was wrong. https://developers.redhat.com/blog/2016/03/01/reducing-memory-access-times-with-caches/

**ldavide** commented on Feb 14, 2018

there is an updated version of the latency table?

**rcosnita** commented on Mar 21, 2018

Nice gist. Thanks **@jboner**.

**connecttobn** commented on Mar 30, 2018

Links are dead..

https://gist.github.com/2843375
http://prezi.com/pdkvgys-r0y6/latency-numbers-for-programmers-web-development/latency.txt

**@jboner** lets remove them

**calimeroteknik** commented on Apr 9, 2018 • edited

> https://prezi.com/pdkvgys-r0y6/latency-numbers-for-programmers-web-development/

This prezi presentation is reversed: the larger numbers are inside the smaller ones, instead of the logical opposite.

**achiang** commented on Apr 17, 2018

Humanized version can be found: https://gist.github.com/hellerbarde/2843375

**jboner** commented on Apr 22, 2018 Owner Author

Thanks. Updated.

**amirouche** commented on Apr 28, 2018

Where is the xkcd version?

**amirouche** commented on Apr 28, 2018

This one is nice https://gist.github.com/hellerbarde/2843375#gistcomment-1896153

**dsuoch-intel** commented on Jul 10, 2018

http://ithare.com/infographics-operation-costs-in-cpu-clock-cycles/

**negrinho** commented on Jul 17, 2018

Just use logarithms directly: https://gist.github.com/negrinho/8a8b45a8958a8653054aa2b349b4cb05

**eleztian** commented on Oct 22, 2018

Thanks

**AnatoliiStepaniuk** commented on Dec 25, 2018

Is there any resources when one can test himself with a tasks involving these numbers?
E.g. calculate how much time will it take to read 5Mb from DB in another datacenter and get it back?
That would be a great test of applying those numbers in some real use cases.

**bhaavanmerchant** commented on Dec 26, 2018

I think given increased use of GPUs / TPUs it might be interesting numbers to add here now. Like: 1MB over PCIexpress to GPU memory, Computing 100 prime numbers per core of CPU compared to CPU, reading 1 MB from GPU memory to GPU etc.

**sergekukharev** commented on Jan 11, 2019

Markdown version https://gist.github.com/sergekukharev/ccdd49d23a5078f108175dc71ad3c06c

**binbinlau** commented on Jan 25, 2019

useful information & thanks



**bpmf** commented on Feb 22, 2019 • edited

Some data of the Berkeley interactive version (https://people.eecs.berkeley.edu/~rcs/research/interactive_latency.html ) is estimated, eg: 4 µs in 2019 to read 1 MB sequentially from memory; it seems too fast.

**speculatrix** commented on Mar 25, 2019

this is a great idea.
how about the time to complete a DNS request - UDP packet request and response with a DNS server having, say, 1ms response time, with the DNS server being 5ms packet time-of-flight away?

**schemacs** commented on Apr 12, 2019

https://computers-are-fast.github.io

**joelkraehemann** commented on Apr 15, 2019

What effect on latency has the use multiple native threads on doing operations possible due to proper mutex locking? Assumed you have:

- an operation 1024 ns operation in 1st level cache
- 2 x lock unlock mutex (50 ns)
- move it from/to main memory (200 ns)

Now, I wonder about malloc latency, can you tell about it? It is definitely missing because I can compute data without any lock as owning the data.

**haai** commented on Sep 4, 2019

interesting when you see in a glance. but would't it be good to use one unit in the comparison e.g. memory page 4k?

**acuariano** commented on Sep 11, 2019

> Nanoseconds

It's an excellent explanation. I had to search the video because the account was closed. Here's the result I got: https://www.youtube.com/watch?v=9eyFDBPk4Yw

**KevinZhou92** commented on Jan 30, 2020

`Send 1K bytes over 1 Gbps network 10,000 ns 10 us`
This doesn't look right to me. 1 Gbps = 125, 000 KB/s, the time should be 1 / 125,000 = 8 * 10^-6 seconds which is 8000ns

**andaru** commented on Apr 4, 2020

> `Send 1K bytes over 1 Gbps network 10,000 ns 10 us`
> This doesn't look right to me. 1 Gbps = 125, 000 KB/s, the time should be 1 / 125,000 = 8 * 10^-6 seconds which is 8000ns

For a direct host-to-host connection with 1000BaseT interfaces, a wire latency of 8µs is correct.

However, if the hosts are connected using SGMII, the Serial Gigabit Media Independent Interface, data is 8b10b encoded, meaning 10 bits are sent for every 8 bits of data, leading to a latency of 10µs.

Jeff may also have been referring to the fact that in a large cluster you'll have a few switches between the hosts, so even where 1000BaseT is in use, the added switching latency (even for switches operating in cut-through mode) for, say, 2 switches can approach 2µs.

In any event, the main thing to take away from these numbers are the orders of magnitude differences between latency for various methods of I/O.

**moon-chilled** commented on Apr 27, 2020

Fancy unicode version:

```
Latency Comparison Numbers (~2012)
┌──────────────────────────────────────────────────────────────────────────────────
│L1 cache reference                            0.5 ns
│
├──────────────────────────────────────────────────────────────────────────────────
│Branch mispredict                             5   ns
│
├──────────────────────────────────────────────────────────────────────────────────
│L2 cache reference                            7   ns
14× L1 cache                    │
├──────────────────────────────────────────────────────────────────────────────────
│Mutex lock/unlock                            25   ns
│
├──────────────────────────────────────────────────────────────────────────────────
│Main memory reference                       100   ns
20× L2 cache, 200× L1 cache     │
├──────────────────────────────────────────────────────────────────────────────────
│Compress 1K bytes with Zippy                3,000   ns         3 µs
│
├──────────────────────────────────────────────────────────────────────────────────
│Send 1K bytes over 1 Gbps network          10,000   ns        10 µs
│
├──────────────────────────────────────────────────────────────────────────────────
│Read 4K randomly from SSD*                150,000   ns       150 µs
~1GB/sec SSD                    │
├──────────────────────────────────────────────────────────────────────────────────
│Read 1 MB sequentially from memory        250,000   ns       250 µs
```

```
|
├──────────────────────────────────────────────────────────────────────────────────────────────

|Round trip within same datacenter          500,000   ns       500 µs
|
├──────────────────────────────────────────────────────────────────────────────────────────────

|Read 1 MB sequentially from SSD*         1,000,000   ns     1,000 µs       1 ms
~1GB/sec SSD, 4× memory         |
├──────────────────────────────────────────────────────────────────────────────────────────────

|Disk seek                               10,000,000   ns    10,000 µs      10 ms
20× datacenter roundtrip        |
├──────────────────────────────────────────────────────────────────────────────────────────────

|Read 1 MB sequentially from disk        20,000,000   ns    20,000 µs      20 ms
80× memory, 20× SSD             |
├──────────────────────────────────────────────────────────────────────────────────────────────

|Send packet CA→Netherlands→CA          150,000,000   ns   150,000 µs     150 ms
|
└──────────────────────────────────────────────────────────────────────────────────────────────
```

**arunkumaras10** commented on May 20, 2020

Are these numbers still relevant in 2020? Or this needs an update?

**maning711** commented on Jun 9, 2020

> Are these numbers still relevant in 2020? Or this needs an update?

I think hardwares are so expensive that can't update them~

**vladimirvs** commented on Jul 21, 2020

One thing that is misleading is that different units are used for send over 1Gbps versus read 1 MB from RAM. RAM is at least x20 times faster, but it ranks below send over network which is misleading. They should have used the same 1MB for network and RAM.

**amresht** commented on Aug 6, 2020 • edited

> need a solar system type visualization for this, so we can really appreciate the change of scale.

Hi
I liked your request and made an comparison. One unit is Mass of earth not radius.

| Operation | Time in Nano Seconds | Astronomical Unit of Weight |
|---|---|---|
| L1 cache reference | 0.5 ns | 1/2 Earth or Five times Mars |
| Branch mispredict | 5 ns | 5 Earths |
| L2 cache reference | 7 ns | 7 Earths |
| Mutex lock/unlock | 25 ns | Roughly [Uranus +Neptune] |
| Main memory reference | 100 ns | Roughly Saturn + 5 Earths |
| Compress 1K bytes with Zippy | 3,000 ns | 10 Jupiters |
| Send 1K bytes over 1 Gbps network | 10,000 ns | 20 Times All the Planets of the Solar System |
| Read 4K randomly from SSD* | 150,000 ns | 1.6 times Red Dwarf Wolf 359 |
| Read 1 MB sequentially from memory | 250,000 ns | Quarter of the Sun |
| Round trip within same datacenter | 500,000 ns | Half of the Mass of Sun |
| Read 1 MB sequentially from SSD* | 1,000,000 ns | Sun |
| Disk seek | 10,000,000 ns | 10 Suns |
| Read 1 MB sequentially from disk | 20,000,000 ns | Red Giant R136a2 |
| Send packet CA->Netherlands->CA | 150,000,000 ns | An Intermediate Sized Black Hole |

https://docs.google.com/spreadsheets/d/13R6JWSUry3-TcCyWPbBhD2PhCeAD4ZSFqDJYS1SxDyc/edit?usp=sharing

**asimilon** commented on Oct 4, 2020

> need a solar system type visualization for this, so we can really appreciate the change of scale.
>
> Hi
> I liked your request and made an comparison. One unit is Mass of earth not radius

> I liked your request and made an comparison. One unit is mass of earth not radius.

For me the best way of making this "more human relatable" would be to treat nanoseconds as seconds and then convert the large values.

eg. 150,000,000s = ~4.75 years