

Magic Numbers & Un/Pack

a use.perl.org magical mystery tour

Boston.PM
2008-11-18
Bill Ricker
aka n1vux

HAKMEM 169

- How many bits are set in a word?

```
sub bc { ### SLOW O(N) ###  
  my $v=shift; my $c;  
  for ($c = 0; $v; $v >>= 1){ $c += $v & 1; }  
  return $c;}
```

- How fast can it be done?

- For some applications, including keyword search, this is critical. Constant time, space with small constant needed.

ITEM 169 (in order of one-ups-manship: Gosper, Mann, Lenard, [Root and Mann]):

To count the ones in a **PDP-6/10** word:

```
LDB B,[014300,,A] ;or MOVE B,A then LSH B,-1  
AND B,[333333,,333333]  
SUB A,B  
LSH B,-1  
AND B,[333333,,333333]  
SUBB A,B ;each octal digit is replaced by number of 1's in it  
LSH B,-3  
ADD A,B  
AND A,[070707,,070707]  
IDIVI A,77 ;casting out 63.'s
```

HAKMEM 169 (32)

011111111111

- Changes look in C and 32bit

- <http://blogs.msdn.com/jeuge/archive/2005/06/08/HAKMEM-Bit-Count.aspx>

```
int BitCount(unsigned int n)
{
    unsigned int uCount;

    uCount = n - ((n >> 1) & 033333333333) - ((n >> 2) & 011111111111);
    return ((uCount + (uCount >> 3)) & 030707070707) % 63;
}
```

$$n = a_{31} * 2^{31} + a_{30} * 2^{30} + \dots + a_k * 2^k + \dots + a_1 * 2 + a_0;$$

$$u = a_2 * 2^2 + a_1 * 2 + a_0;$$

$$u >> 1 = a_2 * 2 + a_1;$$

$$u >> 2 = a_2;$$

$$n - (n >> 1) - (n >> 2) = (4-2-1)a_2 + (2-1) a_1 + 1 a_0 = a_2 + a_1 + a_0$$

- With sums threes, the >>3 gives two sets of sums of six.
- 030...0707 discards the duplicates.
- sum digits: mod 63=x3f=077
 - casting out nines but in base64 ... but not for 64bit !\

HAKMEM 169

- This number octal 1111111111 is sacred to HAKMEM 169, bitcount.
- <http://blogs.msdn.com/jeuge/archive/2005/06/08/HAKMEM-Bit-Count.aspx>
<http://infolab.stanford.edu/~manku/bitcount/bitcount.html>
<http://www.setbb.com/phpbb/viewtopic.php?mforum=sudoku&p=7629>
<http://www.inwap.com/pdp10/hbaker/hakmem/hakmem.html>

Magic Numbers & Un/Pack

So far ...

- HAKMEM 169 011111111111

Popcount SWAR reinvented x55/xAA

- often

bm32.pl.html#ilya-

- rediscovered
- improved
- renamed
 - Pop(ulation)count
 - bitcount, bitcnt
 - SWAR: SIMD within a register
 - Sideways addition, sidesum
 - Hamming Weight
 - Ones Count

Popcount notes

- <http://aggregate.ee.engr.uky.edu/MAGIC/>
- http://en.wikipedia.org/wiki/Hamming_weight
- <http://www.math.uni-bielefeld.de/~sillke/PROBLEMS/bitcount>
=> <http://www.math.uni-bielefeld.de/~sillke/ALGORITHMMS/bitmani/>
- <http://www.ciphersbyritter.com/NEWS4/BITCT.HTM>
- http://www.dalkescientific.com/writings/diary/archive/2008/07/03/hakmem_and_other_popcounts.html
=> <http://wiki.cs.pdx.edu/forged/popcount.html>
- hackers delight
<http://safari.awprofessional.com/0201914654> <http://www.hackersdelight.org/>
http://www.pearsonhighered.com/academic/product/0,,0201914654,00%2Ben-USS_01DBC.html
- <http://popcnt.org/2007/09/magic-popcount-popcnt-command.html>
- http://www.dalkescientific.com/writings/diary/archive/2008/07/05/bitslice_and_popcount.html
- <http://graphics.stanford.edu/~seander/bithacks.html#CountBitsSetNaive>
corrects kernighan=>wegner+lehmer
- <http://groups.google.com/group/comp.graphics.algorithms/msg/43dfb1f4a0f08441>
- "Beautiful Code" <http://isbn.nu/9780596510046>
- "Programming Pearls" http://isbn.nu/authorx/bentley_jon_louis <http://www.cs.bell-labs.com/cm/cs/pearls/>

Magic Numbers & Un/Pack

So far ...

- HAKMEM 169 011111111111
- Popcount SWAR reinvented x55/xAA

Usenet Benchmark contest

bm32.pl.html

bm64.pl.html

bitcount.rtf

-
-

```
wdr@bill-laptop:~/perl/advent/bit-count$ bench=1 perl510 ./bm32.pl
```

	Rate	rec_n	direct	lslb	decr	ilya	sprtr	katz	table	ones32	unroll	unrmsk	parmod	hakmem_169	tabunr	packtut
rec_n	39172/s	--	-67%	-77%	-82%	-85%	-88%	-90%	-91%	-92%	-92%	-92%	-93%	-93%	-94%	-94%
direct	118266/s	202%	--	-29%	-44%	-54%	-63%	-71%	-72%	-76%	-76%	-77%	-80%	-80%	-81%	-83%
lslb	167166/s	327%	41%	--	-21%	-35%	-47%	-58%	-61%	-66%	-66%	-67%	-71%	-72%	-72%	-76%
decr	212850/s	443%	80%	27%	--	-18%	-33%	-47%	-50%	-56%	-56%	-58%	-63%	-65%	-65%	-70%
ilya	258673/s	560%	119%	55%	22%	--	-18%	-36%	-39%	-47%	-47%	-49%	-55%	-57%	-57%	-63%
sprtr	316291/s	707%	167%	89%	49%	22%	--	-21%	-25%	-35%	-35%	-38%	-45%	-47%	-48%	-55%
katz	402668/s	928%	240%	141%	89%	56%	27%	--	-5%	-17%	-17%	-21%	-30%	-33%	-34%	-42%
table	424511/s	984%	259%	154%	99%	64%	34%	5%	--	-13%	-13%	-17%	-27%	-29%	-30%	-39%
ones32	487116/s	1144%	312%	191%	129%	88%	54%	21%	15%	--	-0%	-4%	-16%	-19%	-20%	-30%
unroll	487480/s	1144%	312%	192%	129%	88%	54%	21%	15%	0%	--	-4%	-16%	-19%	-20%	-30%
unrmsk	509460/s	1201%	331%	205%	139%	97%	61%	27%	20%	5%	5%	--	-12%	-15%	-16%	-27%
parmod	579359/s	1379%	390%	247%	172%	124%	83%	44%	36%	19%	19%	14%	--	-4%	-5%	-17%
hakmem_169	601230/s	1435%	408%	260%	182%	132%	90%	49%	42%	23%	23%	18%	4%	--	-1%	-14%
tabunr	606683/s	1449%	413%	263%	185%	135%	92%	51%	43%	25%	24%	19%	5%	1%	--	-13%
packtut	699252/s	1685%	491%	318%	229%	170%	121%	74%	65%	44%	43%	37%	21%	16%	15%	--

```
# No tests run!
```

Magic Numbers & Un/Pack

So far ...

- HAKMEM 169 011111111111
- Popcount SWAR reinvented x55/xAA
- Usenet Benchmark contest

Bit banging in interpretive languages

- C rarely beats hardware
 - but `bitcnt` in best C can beat *naïve* microcode
- Perl userspace code *should* never beat libc and Perl's C guts
 - loops in C always cheaper than loops in Perl
- only reason to try
 - fun
 - can't find the right builtin yet
 - or doesn't exist and XS + Inline::C out of bounds

Optimization

- Rule 1 – Don't
- Rule 2 – Don't do it yet.
 - *(for experts only)*
- Rule 3 – Benchmark

Magic Numbers & Un/Pack

So far ...

- HAKMEM 169 011111111111
- Popcount SWAR reinvented x55/xAA
- Usenet Benchmark contest
- Bit banging in interpretive languages
- Rule 1 Don't

e.g. Log2

use Perl;

Search use Perl

Go

All the Perl that's Practical to Extract and Report

[FAQ](#) [About](#) [Submit Story](#) [Older Stories](#) [Topics](#) [Polls](#) [Journals](#) [Authors](#) [New Modules](#) [Preferences](#) [Logout](#)
[Logout](#) [Preferences](#) [Password](#) [~n1vux \(1492\)](#) [~Ovid \(2709\)](#)

Journal of Ovid (2709) 😊 😊 😊

[Info](#) [Relation](#) [Journal](#) [Firehose](#) [Friends](#) [Fans](#) [Foes](#) [Freaks](#) [Tags](#) [Bookmarks](#)
[n1vux's Journal](#) [Write in Journal](#) [Delete/Edit Entries](#) [Top 10](#) [Edit Preferences](#) [Friend's Journals](#)

Wednesday September 03, 2008

 09:49 AM **Heatmap Ranges** [[#37337](#)]

I'd like to take a series of values and convert them to numbers between 0 and 255, but on a logarithmic scale. My math studies were a long time ago and for the life of me I can't remember how to do this. I'm hoping to take a 2D table of values and project them onto an HTML table in a 'heatmap' fashion.

[List all Journal entries](#)

Ovid (2709)

[Ovid](#) 😊 😊 😊

(email not shown publicly)

<http://publius-ovidius.livejournal.com/>
AOL IM: ovidperl ([Add Buddy](#), [Send Message](#))

Stuff with the Perl Foundation. A couple of patches in the Perl core. A few CPAN modules. That about sums it up.

User Space

[Comments](#)
[Entries](#)
[Write](#)
[Heatmap Ranges](#) | [Preferences](#) | [Top](#) | [3 comments](#) | [Search Discussion](#)

Scripts Partially Allowed, 1/3 (perl.org) | <SCRIPT>: 19 | <OBJECT>: 0

Options...

 Find: [Next](#) [Previous](#) [Highlight all](#)

Done

e.g. Log2

The screenshot shows a Mozilla Firefox browser window with the title "Journal of Ovid (2709) - Mozilla Firefox". The address bar contains "http://use.perl.org/~". The page content is a forum post titled "scale linear then log (Score:4)" by user "slanning (5049)". The post text explains a scaling process: "Maybe I'm misunderstanding, but I think you want to first scale logarithmically: log(\$val) if it's log base 'e'. Then linearly: so find the max value in the logged sequence, then multiply each value by 255 and divide by the max. Here's a kind of generic script, if I got it right:". Below the text is a Perl script. A yellow security warning bar at the bottom of the page reads "Scripts Partially Allowed, 1/3 (perl.org) | <SCRIPT>: 19 | <OBJECT>: 0". The browser's status bar at the bottom shows "Done".

```
#!/usr/bin/perl
# convert @VALS to a log scale base $LOG_BASE
# and scaled linearly to $SCALE_MAX

use strict;
use warnings;
use List::Util qw(max);

my $SCALE_MAX = 1000;
my $LOG_BASE = 10;
my @VALS = (1, 10, 100, 1000);

main();

sub main {
    my @logvals = map { logN($_, $LOG_BASE) } @VALS;
```

e.g. Log2

use Perl;

Search use Perl

Go

All the Perl that's Practical to Extract and Report

[FAQ](#) [About](#) [Submit Story](#) [Older Stories](#) [Topics](#) [Polls](#) [Journals](#) [Authors](#) [New Modules](#) [Preferences](#) [Logout](#)
[Logout](#) [Preferences](#) [Password](#) [~n1vux \(1492\)](#) [~Ovid \(2709\)](#)

Journal of Ovid (2709) 😊 😊 😊

[Info](#) [Relation](#) [Journal](#) [Firehose](#) [Friends](#) [Fans](#) [Foes](#) [Freaks](#) [Tags](#) [Bookmarks](#)
[n1vux's Journal](#) [Write in Journal](#) [Delete/Edit Entries](#) [Top 10](#) [Edit Preferences](#) [Friend's Journals](#)

Wednesday September 03, 2008

 09:49 AM **Heatmap Ranges** [[#37337](#)]

I'd like to take a series of values and convert them to numbers between 0 and 255, but on a logarithmic scale. My math studies were a long time ago and for the life of me I can't remember how to do this. I'm hoping to take a 2D table of values and project them onto an HTML table in a 'heatmap' fashion.

[List all Journal entries](#)
[Heatmap Ranges](#) | [Preferences](#) | [Top](#) | [3 comments](#) | [Search Discussion](#)

Ovid (2709)

Ovid 😊 😊 😊

(email not shown publicly)

<http://publius-ovidius.livejournal.com/>

 AOL IM: ovidperl ([Add Buddy](#), [Send Message](#))

Stuff w with the Perl Foundation. A couple of patches in the Perl core. A few CPAN modules. That about sums it up.

User Space

[Comments](#)
[Entries](#)
[Write](#)

Scripts Partially Allowed, 1/3 (perl.org) | <SCRIPT>: 19 | <OBJECT>: 0

Options...

 Find: [Next](#) [Previous](#) [Highlight all](#)

Done

Journal of Ovid (2709) - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://use.perl.org/~() Google

Log2 for heatmap, fun with pack (Score:2)

by [n1vux \(1492\)](#) on Wednesday September 10, @07:02PM (#64834) [Homepage Journal](#)

i think both of the prior comments are on target.

If you have such quantities of data that scaling and calling log repeatedly is a problem - and only if - there are old integer bitbang routines for log2 that could be done with XS or Inline::C or PDL.

Or you could (ab)use Perl 5.10 pack() to grab the floating point representation's exponent

Note that 256 buckets is a lot, hi res, for loglinear data unless it already was floating point or Math::BigFloat - as $\log_2(\text{MAXLONG}) - \log_2(1)$ 256 or 8 bits -- it's the aize of plain float's exponent. BUT grabbing it is hard since it's offset one bit by the sign.

If your data is not log clean but may include the gamut from -INF to 0 to +INF, if you grab the sign and the top 7 bits of exponent, that's nice, but it's not a single numeric range -- exponent is UCHAR biased by 127 [or by 63 after we nip one bit taking a byte with the sign] -- while the sign prefixed to it inverts direction.) A crass fix -

```
my $s;  
my $n= unpack("C",pack("f>", $data)); #> on x86 cpu  
$n ^= 0xff if $s=$n & 0x80;  
$n ^= 0x80 if !$s;
```

if its all positive, and you want full 256 buckets without XS or PDL or log(), the best I can see is unpack with B9, discard the leading sign bit, repack B8, unpack C. But if you're going to do that might as well get full dynamic range from 11 bit 'double' exponent.

Scripts Partially Allowed, 1/3 (perl.org) | <SCRIPT>: 19 | <OBJECT>: 0 Options...

Find: Next Previous Highlight all

Done

Journal of Ovid (2709) - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://use.perl.org/~(Google

```
sub log2{
    require 5.010; # assumes x86 too...
    # should die if arg <= 0 ...
    # instead gives log2(abs())
    my $str =unpack("B12",pack("F>", shift));
    $str =~ s/^[01]/00000/; # drop sign and pad
    my $exp = unpack("s>",pack("B*", $str ));
    return -1023 + $exp;
}
```

With a bit of magic number abuse, we can easily squeeze out a single fractional bit as well, which could give upto 4096 buckets for positive numbers, and as many more for negatives if you rescale somewhere.

Why 5.10? Because on any commodity platform (x86) we need to coerce to a sensible bit/byte order. I am sure I could work out a x86 specific way with older pack but life is too short.

--
Bill
I had a sig when sigs were cool
use Sig;

[Reply to This](#)

[Prefs](#)


use Perl; If you want to see useful Perl examples, we can certainly arrange to have comp.lang.misc flooded with them, but I don't think that would help the advance of civilization. :-)
--Larry Wall in <1992Mar5.180926.19041@netlabs.com>

[download @ Perl %](#)

Stories, comments, journals, and other submissions on use Perl; are Copyright 1998-2006, their respective

Scripts Partially Allowed, 1/3 (perl.org) | <SCRIPT>: 19 | <OBJECT>: 0 [Options...](#)

Find: [Next](#) [Previous](#) [Highlight all](#)

Done 

Log2 perf

log2.pl.html

- SWAR version uses bitcnt and thus 011111111111
- perl510 log2.pl bench

	Rate	log2_xx	log2_2	log2_3	log2_1	log2_0	floor_log2	floor_log2_pt	log2_LL
log2_xx	406/s	--	-4%	-4%	-5%	-16%	-35%	-45%	-62%
log2_2	422/s	4%	--	-0%	-2%	-12%	-33%	-43%	-61%
log2_3	423/s	4%	0%	--	-1%	-12%	-33%	-42%	-60%
log2_1	429/s	6%	2%	1%	--	-11%	-32%	-42%	-60%
log2_0	481/s	18%	14%	14%	12%	--	-23%	-35%	-55%
floor_log2	628/s	55%	49%	48%	46%	31%	--	-15%	-41%
floor_log2_pt	736/s	81%	74%	74%	71%	53%	17%	--	-31%
log2_LL	1070/s	163%	153%	153%	149%	122%	70%	45%	--

	Rate	log2_Lcs	log2_Lc	log2_LL
log2_Lcs	1060/s	--	-2%	-3%
log2_Lc	1078/s	2%	--	-1%
log2_LL	1088/s	3%	1%	--

my \$n=shift; cost 4%

Magic Numbers & Un/Pack

So far ...

- HAKMEM 169 011111111111
- Popcount SWAR reinvented x55/xAA
- Usenet Benchmark contest
- Bit banging in interpretive languages
- Rule 1 Don't
- Log2 – can use bitcnt but builtin still fastest

Perl 5.10 pack F>

- Byte-order modifiers for pack() and unpack()
 - There are two new byte-order modifiers, > (big-endian) and < (little-endian), that can be appended to most pack() and unpack() template characters and groups to force a certain byte-order for that type or group. See pack and perlpacktut for details.
 - <http://perldoc.perl.org/perldelta.html#Byte-order-modifiers-for-pack%28%29-and-unpack%28%29>
 - <http://perldoc.perl.org/perlpacktut.html#Dealing-with-Endian-ness>
- <http://perldoc.perl.org/functions/pack.html>
- Must be enabled
 - 5.10 - <http://perldoc.perl.org/perldelta.html#The-%27feature%27-pragma>

Magic Numbers & Un/Pack

So far ...

- HAKMEM 169 011111111111
- Popcount SWAR reinvented x55/xAA
- Usenet Benchmark contest
- Bit banging in interpretive languages
- Rule 1 Don't
- Eg, Log2
- Perl 5.10 pack F>

More fun with unpack

- `ACK_PAGER_COLOR='less -R' ack --perl '(?![$])\b(un)?pack\b' ~/perl ~/localperl`
-

Magic Numbers & Un/Pack

So far ...

- HAKMEM 169 011111111111
- Popcount SWAR reinvented x55/xAA
- Usenet Benchmark contest
- Bit banging in interpretive languages
- Rule 1 Don't
- Eg, Log2
- Perl 5.10 pack F>
- More fun with unpack

Bill'o'clock

Journal of n1vux (1492) - Mozilla Firefox

File Edit View History Bookmarks Tools Help

Search **use Perl** Go

use Perl;

All the Perl that's Practical to Extract and Report

[FAQ](#) [About](#) [Submit Story](#) [Older Stories](#) [Topics](#) [Polls](#) [Journals](#) [Authors](#) [New Modules](#) [Preferences](#) [Logout](#)

[Logout](#) [Preferences](#) [Password](#) [~n1vux \(1492\)](#) [~Ovid \(2709\)](#)

Journal of n1vux (1492)

[Info](#) [Journal](#) [Firehose](#) [Messages](#) [Friends](#) [Fans](#) [Foes](#) [Freaks](#) [Tags](#) [Bookmarks](#)

[n1vux's Journal](#) [Write in Journal](#) [Delete/Edit Entries](#) [Top 10](#) [Edit Preferences](#) [Friend's Journals](#)

Tuesday March 29, 2005
08:03 AM **Happy "BILL" o'clock** [[Edit](#) | [Delete](#) | [#23904](#)]

[Elaborating on the original report at <http://use.perl.org/comments.pl?sid=25547&cid=39001>]

This morning, at 123836GMT/ 7:38:36 EST, the Unix time_t value was (in Network standard byte order) "BILL", that being the ASCII representation of the 32bit number of seconds since New Years 1970.

BILL Tue Mar 29 12:38:36 2005 GMT . Tue Mar 29 07:38:36 2005 ET

So happy BILL-o'clock to all the Bill's in my address book.

Also upcoming is Bill-o'clock and a bunch of other 4-letter words.

n1vux (1492)
[n1vux](#)
Bill Ricker
n1vux@yahoo.com
(email not shown publicly)
<http://boston.pm...x.cgi?BillRicker>
Karma: 25
AOL IM: n1vux ([Add Buddy](#), [Send Message](#))
Yahoo! ID: [n1vux](#) ([Add User](#), [Send Message](#))

Only started with Perl4 and Perl5 in 1995. I was doing AWK etc for 12 years before that, and resisted switching. I've been doing OO since before C++ hit bigtime, with Objective-C and SmallTalk, so I really like the (no longer new) Perl5 OO style; and the Lispish Map style is also an old friend. What do I hack with Perl? All data that passes my way; systems monitoring scripts at \$DayJob, weather data at night, and I cheat on NPR word puzzles.
Member: [Boston.pm.org](#) [pm.org] [BLU.org](#) [blu.org] / [LinkedIn](#) [linkedin.com]

N1 VUX is my FCC-issued ham radio callsign.

[Logout](#) | [Preferences](#) | [Password](#) | [~n1vux \(1492\)](#) | [~Ovid \(2709\)](#)

Journal of n1vux (1492)

[Info](#) | [Journal](#) | [Firehose](#) | [Messages](#) | [Friends](#) | [Fans](#) | [Foes](#) | [Freaks](#) | [Tags](#) | [Bookmarks](#)

[n1vux's Journal](#) | [Write in Journal](#) | [Delete/Edit Entries](#) | [Top 10](#) | [Edit Preferences](#) | [Friend's Journals](#)

Tuesday March 29, 2005

08:03 AM **Happy "BILL" o'Clock** [[Edit](#) | [Delete](#) | [#23904](#)]

[Elaborating on the original report at <http://use.perl.org/comments.pl?sid=25547&cid=39001>]

This morning, at 123836GMT/ 7:38:36 EST, the Unix time_t value was (in Network standard byte order) "BILL", that being the ASCII representation of the 32bit number of seconds since New Years 1970.

BILL Tue Mar 29 12:38:36 2005 GMT . Tue Mar 29 07:38:36 2005 ET

So happy BILL-o'clock to all the Bill's in my address book.

Also upcoming is Bill-o-clock and a bunch of other 4-letter words.

BYTE Sun Apr 10 16:28:53 2005 GMT . Sun Apr 10 12:28:53 2005 ET
Beer Tue Apr 19 20:09:22 2005 GMT . Tue Apr 19 16:09:22 2005 ET
Bill Fri Apr 22 21:28:12 2005 GMT . Fri Apr 22 17:28:12 2005 ET
Byte Thu May 5 01:18:29 2005 GMT . Wed May 4 21:18:29 2005 ET

And CAAA-CZZZ, Caaa-Czzz are due in the fall, starting with

CABS Mon Oct 3 14:38:11 2005 GMT . Mon Oct 3 10:38:11 2005 ET
CAFE Mon Oct 3 14:55:01 2005 GMT . Mon Oct 3 10:55:01 2005 ET

This runs about 2 letters a year thru

Zoos Mon Jan 29 19:01:07 2018 GMT . Mon Jan 29 14:01:07 2018 ET

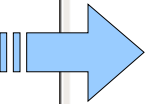
[n1vux](#)
Bill Ricker
n1vux@yahoo.com
(email not shown publicly)
<http://boston.pm...x.cgi?BillRicker>
Karma: 25
AOL IM: n1vux ([Add Buddy](#), [Send Message](#))
Yahoo! ID: [n1vux](#) ([Add User](#), [Send Message](#))

Only started with Perl4 and Perl5 in 1995. I was doing AWK etc for 12 years before that, and resisted switching. I've been doing OO since before C++ hit bigtime, with Objective-C and SmallTalk, so I really like the (no longer new) Perl5 OO style; and the Lispish Map style is also an old friend. What do I hack with Perl? All data that passes my way; systems monitoring scripts at \$Dayjob, weather data at night, and I cheat on NPR word puzzles.
Member: [Boston.pm.org](#) [pm.org] [BLU.org](#) [blu.org] / [LinkedIn](#) [linkedin.com]

N1 VUX is my FCC-issued ham radio callsign.

User Space

[Comments](#)
[Entries](#)
[Write](#)



BILL'o'Clock

- <http://use.perl.org/comments.pl?sid=25547&cid=>
- <http://use.perl.org/~n1vux/journal/23904>

time_t2.pl.html

time_t

time_t2.pl.html

- `$ perl time_t2.pl -s iaaa izzz`

```
IAAA Thu Dec 11 16:35:13 2008 GMT . Thu Dec 11 11:35:13 2008 ET
IZZZ Tue Dec 30 17:28:58 2008 GMT . Tue Dec 30 12:28:58 2008 ET
Iaaa Mon Jan 5 01:24:49 2009 GMT . Sun Jan 4 20:24:49 2009 ET
Izzz Sat Jan 24 02:18:34 2009 GMT . Fri Jan 23 21:18:34 2009 ET
```

- `$ perl time_t2.pl --sort I > I_time_t.txt`

```
ICED Sat Dec 13 05:16:52 2008 GMT . Sat Dec 13 00:16:52 2008 ET
..
IRES Wed Dec 24 14:21:07 2008 GMT . Wed Dec 24 09:21:07 2008 ET
IRIS Wed Dec 24 14:38:11 2008 GMT . Wed Dec 24 09:38:11 2008 ET
IRKS Wed Dec 24 14:46:43 2008 GMT . Wed Dec 24 09:46:43 2008 ET
IRON Wed Dec 24 15:03:42 2008 GMT . Wed Dec 24 10:03:42 2008 ET
ISLE Thu Dec 25 09:03:01 2008 GMT . Thu Dec 25 04:03:01 2008 ET
ITCH Fri Dec 26 02:36:56 2008 GMT . Thu Dec 25 21:36:56 2008 ET
ITEM Fri Dec 26 02:45:33 2008 GMT . Thu Dec 25 21:45:33 2008 ET ..
Iced Tue Jan 6 14:06:28 2009 GMT . Tue Jan 6 09:06:28 2009 ET ..
Ires Sat Jan 17 23:10:43 2009 GMT . Sat Jan 17 18:10:43 2009 ET
Iris Sat Jan 17 23:27:47 2009 GMT . Sat Jan 17 18:27:47 2009 ET
Irks Sat Jan 17 23:36:19 2009 GMT . Sat Jan 17 18:36:19 2009 ET
Iron Sat Jan 17 23:53:18 2009 GMT . Sat Jan 17 18:53:18 2009 ET
Isle Sun Jan 18 17:52:37 2009 GMT . Sun Jan 18 12:52:37 2009 ET
Itch Mon Jan 19 11:26:32 2009 GMT . Mon Jan 19 06:26:32 2009 ET
Item Mon Jan 19 11:35:09 2009 GMT . Mon Jan 19 06:35:09 2009 ET
```

Magic Numbers & Un/Pack

So far ...

- HAKMEM 169 011111111111
- Popcount SWAR reinvented x55/xAA
- Usenet Benchmark contest
- Bit banging in interpretive languages
- Rule 1 Don't
- Eg, Log2
- Perl 5.10 pack F>
- More fun with unpack
- Bill'o'clock

Obfu

- What does this do?

```
unlink${0}if$^T>1x10;
```

<http://use.perl.org/~cog/journal/23711>

Obfu

- What does this do?

```
unlink${0}if$^T>1x10;
```

I first noticed that about 6 months ago, when I was thinking (just for fun) on [logic bombs](#) and came up with this one:

```
unlink${0}if$^T>1x10;
```

Nothing really fancy there :-)

Move along now :-)

<http://use.perl.org/~cog/journal/23711>

Obfu

- What does this do?

```
unlink${0}if$^T>1x10;
```

[Logout](#) | [Preferences](#) | [Password](#) | [~n1vux\(1492\)](#) | [~cog\(4665\)](#)
Journal of cog (4665) 😊 😊

[Info](#) | [Relation](#) | **[Journal](#)** | [Firehose](#) | [Friends](#) | [Fans](#) | [Foes](#) | [Freaks](#) | [Tags](#) | [Bookmarks](#)

[n1vux's Journal](#) | [Write in Journal](#) | [Delete/Edit Entries](#) | [Top 10](#) | [Edit Preferences](#) | [Friend's Journals](#)

Friday March 18, 2005
04:39 AM **Have a nice 1111111111 day** [#23711]

That'll be [today](#).

I first noticed that about 6 months ago, when I was thinking (just for fun) on [logic bombs](#) and came up with this one:

```
unlink${0}if$^T>1x10;
```

Nothing really fancy there :-)

Move along now :-)

Obfu

- What does this do?

```
unlink${0}if$^T>1x10;
```

[Logout](#) | [Preferences](#) | [Password](#) | [~n1vux \(1492\)](#) | [~cog \(4665\)](#)
[Journal of cog \(4665\)](#) 😊 😊

[Info](#) | [Relation](#) | [Journal](#) | [Firehose](#) | [Friends](#) | [Fans](#) | [Foes](#) | [Freaks](#) | [Tags](#) | [Bookmarks](#)

[n1vux's Journal](#) | [Write in Journal](#) | [Delete/Edit Entries](#) | [Top 10](#) | [Edit Preferences](#) | [Friend's Journals](#)

Friday March 18, 2005

04:39 AM **Have a nice 1111111111 day** [[#23711](#)]

That'll be [today](#).

I first noticed that about 6 months ago, when I was thinking (just for fun) on [logic bombs](#) and came up with this one:

```
unlink${0}if$^T>1x10;
```

Nothing really fancy there :-)

Move along now :-)

/i. celebrated `date +%s` eq 1111111111 too.

<http://slashdot.org/article.pl?sid=05/03/17/169200&tid=130> with mention of 1234567890 as well <http://use.perl.org/comments.pl?sid=25547&cid=39001>

Magic Numbers & Un/Pack

So far ...

- HAKMEM 169 011111111111
- Popcount SWAR reinvented x55/xAA
- Usenet Benchmark contest
- Bit banging in interpretive languages
- Rule 1 Don't
- Eg, Log2
- Perl 5.10 pack F>
- More fun with unpack
- Bill'o'clock
- Obfu

Today (UTC)

- Obfu was for 1x10 base ten.
- When is 1x11 *base 8* aka 011111111111 ?
- octal 1111111111
= decimal 1227133513
= hex 0x49249249
= (as time_t) Wed Nov 19 22:25:13 UTC 2008
= Wednesday, November 19, 2008 5:25 PM ET
- it was after UTC Midnight when the meeting started, in POSIX standard systime it's later today!

Magic Numbers & Un/Pack

So far ...

- HAKMEM 169 011111111111
- Popcount SWAR reinvented x55/xAA
- Usenet Benchmark contest
- Bit banging in interpretive languages
- Rule 1 Don't
- Eg, Log2 ... 011111111111
- Perl 5.10 pack F>
- More fun with unpack
- Bill'o'clock
- Obfu 1111111111
- Today (UTC) 011111111111

ta da

Bonus slides

Y2K038

- <http://www.google.com/search?q=2038+date>

```
5.8.0/386 RH3
$ perl y2k038.pl
-7 Tue Jan 19 03:14:01 2038
-6 Tue Jan 19 03:14:02 2038
-5 Tue Jan 19 03:14:03 2038
-4 Tue Jan 19 03:14:04 2038
-3 Tue Jan 19 03:14:05 2038
-2 Tue Jan 19 03:14:06 2038
-1 Tue Jan 19 03:14:07 2038
 0 Fri Dec 13 20:45:52 1901
+1 Fri Dec 13 20:45:52 1901
+2 Fri Dec 13 20:45:52 1901
```

```
5.8.85/x86_64 RH4
$ perl ./y2k038.pl
-7 Tue Jan 19 03:14:01 2038
-6 Tue Jan 19 03:14:02 2038
-5 Tue Jan 19 03:14:03 2038
-4 Tue Jan 19 03:14:04 2038
-3 Tue Jan 19 03:14:05 2038
-2 Tue Jan 19 03:14:06 2038
-1 Tue Jan 19 03:14:07 2038
 0 Tue Jan 19 03:14:08 2038
+1 Tue Jan 19 03:14:09 2038
+2 Tue Jan 19 03:14:10 2038
```

```
1 #! /usr/bin/perl
2
3 # http://www.google.com/search?q=2038+date
4
5
6 use POSIX;
7 $ENV{'TZ'} = "GMT";
8
9 my $eow= 2147483641;
10 for ($clock = $eow; $clock < 2147483651;
      $clock++) {
11     my $dt=$clock-$eow-7;
12     print "@{[$dt > 0 ? '+:']}$dt
      @{{ctime($clock)}}";
13 }
14
```

y2k038.pl.html

Y2K038 fix

- **y2038** <http://use.perl.org/~schwern/journal/37688>
 - <http://y2038.googlecode.com/svn/trunk/patches/Time-Local-Extended.patch>
 - <http://use.perl.org/~schwern/journal/37688?from=rss>
- **and the Y3001 bug.**
<http://use.perl.org/~schwern/journal/37586?from=rss>

Who needs Bignums?

- <http://use.perl.org/~davorg/journal/34739>
 - <http://use.perl.org/comments.pl?cid=58619&sid=373>
- We all do eventually ..
 - <http://www.forthgo.com/blog/2008/10/19/fast-factoring-for-64-bit-integers/#comment-6319>
 - <http://projecteuler.net/index.php?section=problems>
 -

More fun with dates

exif-date.pl.html

- Exif dates

- reset file system date of cropped foto to exposure date
- date arithmetic is very plain
- but has a magic number ... find and understand

- dst.pl

dst.pl.html

- check Libc TZ definitions

- 1234567890

- <http://use.perl.org/comments.pl?sid=25547&cid=39001>
- <http://www.perladvent.org/2008/12/>
- <http://www.perladvent.org/2008/16/>

