HP Forum Archive 16

[Return to Index | Top of Index]

Mini-Challenge: Twiddling the bits

*Message #1 Posted by **Valentin Albillo** on 23 June 2006, 7:04 a.m.*

Hi, all:

Since I'm just too busy right now to post my more-or-less-monthly, full-fledged S&SMC (#17 this time), which I expect to do soon, meanwhile you can try this mini-challenge (actually, more of a *curiosity* than a real challenge), which can be attempted in most nearly every HP calc out there.

In fact, it's so easy that my HP-71B version is only *4 lines* long, but you can do it in any HP model from the HP-25 onwards. Now, this is the

Mini-challenge

For a nice surprise, try and write a short program to do the following, in **FIX 7** display mode:

- - compute the first **19 bits** of **e (= 2.71828...)**
- - append them in reverse order to "11**.** 00 10" and convert back to base 10

Nice, uh ? :-)

I strongly suggest you *avoid* re-reading this thread till you've succeeded, lest someone else's posted solution would spoil the surprise for you.

Best regards from V.

Edited: 23 June 2006, 8:20 a.m.

Re: Mini-Challenge: Twiddling the bits

*Message #2 Posted by **Gerson W. Barbosa** on 23 June 2006, 10:05 p.m.,*
in response to message #1 by Valentin Albillo

Hello Valentin,

Here is an RPL solution (HP-48GX):

```
%%HP: T(3)A(D)F(.);
\<< e 2 - 0 0 1 1 { }
\-> n ns x p s l
  \<< 1 17
    START .5 'p'
STO* p s * 'x' STO+
      IF x n <
      THEN { 1 } 1
      ELSE { 0 } -1
      END 's' STO
'l' STO+
    NEXT { 0 0 1 0
} l { 0 1 } + + 1
'p' STO REVLIST
LIST\-> 1 SWAP
    START .5 'p'
STO* p * 'ns' STO+
    NEXT ns 3 +
  \>>
\>>
```

You may find this an exotic solution, to say the least, but it does work:

3.1415926218

or

3.1415926 in FIX 7 display mode.

Really interesting! (Besides, this is the kind of challenge I can solve :-)

Best regards,

Gerson.

Re: Mini-Challenge: Twiddling the bits

*Message #3 Posted by **Valentin Albillo** on 23 June 2006, 10:13 p.m.,*
in response to message #2 by Gerson W. Barbosa

Congratulations, Gerson !

The very first solution, and it does produce the correct result !

I hope you enjoyed this mini-challenge and yes, it's quite unexpected, isn't it ? :-)

Best regards from V.

Re: Mini-Challenge: Twiddling the bits

*Message #4 Posted by **Gerson W. Barbosa** on 23 June 2006, 10:57 p.m.,*
in response to message #3 by Valentin Albillo

Thanks Valentin,

I was too lazy to add remarks. But it's easy to see the first START NEXT loop performs base 10 to base 2 conversion for fractional numbers. Actually, the line

```
IF x n <
```

should be

```
IF x n <=
```

so that the routine works for n=0.5 (otherwise it would return 0.011111111111 instead of 0.10000000000).

The second loop does the opposite.

But, as you know, there's another way to relate **pi** to **e**. How's that anyway?

Best regards,

Gerson.

Re: Mini-Challenge: Twiddling the bits

*Message #5 Posted by **Valentin Albillo** on 23 June 2006, 11:32 p.m.,*
in response to message #4 by Gerson W. Barbosa

That one's *very* easy, Gerson !

Best regards from V.

Re: Mini-Challenge: Twiddling the bits

*Message #6 Posted by **Kiyoshi Akima** on 24 June 2006, 2:41 a.m.,*
in response to message #5 by Valentin Albillo

Ah, a bit-twiddling exercise. This calls for the trusty HP-16C. A straightforward implementation of the problem results in the following program. (Multi-digit constants are shown on one line.)

```
(001) LBL A
(002) FLOAT 0
(003) 2.71828  ; more digits aren't necessary
(010) HEX      ; convert to bits
(011) x<>y
(012) d        ; eliminate
(013) RRn      ;  extra
(014) 13       ;  bits
(016) MASKR    ;  and
(017) AND      ;  reposition
(018) 32       ; 32h = 110010b
(020) x<>y
(021) LBL 1
(022) SR       ; put rightmost bit into carry
(023) x<>y
(024) RLC      ; append bit to the right
(025) x<>y
(026) x!=0     ; repeat until
(027) GTO 1    ;  no more bits
(028) R down
(029) 7        ; reposition
(030) RLn      ;  (accounting for extra bits added in steps 018-019)
(031) x<>y
(032) FLOAT 9  ; convert to floating point
(033) RTN
```

33 bytes long and takes about fifteen seconds.

Now for the HP-48GX (probably works on all the 48 series, as well as the 49). May even work on the 38/39 series for all I know about them.

```
<<
 1 EXP 2 17 ^ * IP R->B #32
 1 19 START
  2 * OVER #1h AND + SWAP 2 / SWAP
```

```
 NEXT
 SWAP DROP B->R 2 23 ^ /
>>
```

The 48 doesn't do rotates and shifts through carry so I fake them using * / AND. 130 bytes and takes about half a second, with the same answer as above (to 25 bits).

I wasn't going to give the answers, but since they've already been presented: On the HP-16C: 3.141592622 On the HP-48GX: 3.1415926218

Interesting challenge. Looking forward to your 71B program, V. (Though I doubt it's as short as 33 bytes :-))

Re: Mini-Challenge: Twiddling the bits

*Message #7 Posted by **Gerson W. Barbosa** on 24 June 2006, 11:56 a.m.,*
in response to message #6 by Kiyoshi Akima

Congratulations!

Very concise programs! If you accept a simple suggestion, replace **1 EXP** with the built-in constant **e** to make your 48GX program even shorter: only 127.5 bytes (precisely 200 bytes shorter than mine... and about 2 seconds faster!).

Regards,

Gerson.

Edited: 24 June 2006, 12:07 p.m.

Re: Mini-Challenge: Twiddling the bits

*Message #8 Posted by **Kiyoshi Akima** on 24 June 2006, 1:29 p.m.,*
in response to message #7 by Gerson W. Barbosa

Gerson, you're absolutely right about using the built-in constant for **e** instead of evaluating it. I was just so used to the older calculators that hitting 1 e^x was automatic.

Another ten bytes could be saved by hard-coding values for 2^17 and 2^23. And maybe more by replacing the definite loop with an indefinite loop as in the 16C version. I'm sure a few more bytes could be squeezed out of the 16C version as well.

Both of these programs were first drafts, just blindly following the statement of the challenge. No attempts at optimization. Never thought about putting the bits into a list and REVLISTing it. Nope, it was always "shift bits from the right end of word 1 onto the right end of word

2" from the beginning. Pure brute force.

Re: Mini-Challenge: Twiddling the bits

*Message #9 Posted by **Gerson W. Barbosa** on 24 June 2006, 6:26 p.m.,*
in response to message #8 by Kiyoshi Akima

> Quote:
>
> Never thought about putting the bits into a list and REVLISTing it.

Actually, contrary to what I expected, the bits came out already in reversed order after the the first conversion, that's why I appended {0 1} to include the integer part (2). However, I ended up reversing the final list before putting its elements on the stack to proceed to the conversion to base 10.

I didn't attempt to use the BASE functions on the 48GX because I imagined they woudn't handle fractional numbers, but you have proved they do. The 42S is another calculator capable of base conversions and some logical functions. Perhaps someone comes up with a 42S solution using an approach similar to yours.

Regards,

Gerson.

Re: Mini-Challenge: Twiddling the bits

*Message #10 Posted by **Gerson W. Barbosa** on 24 June 2006, 10:56 a.m.,*
in response to message #5 by Valentin Albillo

You're right, Valentin:

Euler's formula: $\mathbf{e^{i\,pi} = -1}$

Best regards,

Gerson.

Re: Mini-Challenge: Twiddling the bits

*Message #11 Posted by **Crawl** on 24 June 2006, 12:06 p.m.,*
in response to message #10 by Gerson W. Barbosa

What about

```
  _
 / \+inf
 |
 |       e^(-x^2) dx  = sqrt(pi)
 |
\_/-inf
```

Actually, this was the basis for the near identity in SSMC#13, when a sum looked surprisingly close to (but not exactly equal to) pi. The reason was because the sum was a numerical approximation of the above integral; its similarity to pi was not a coincidence (I don't believe anyone pointed that out in the older thread).

Actually, the simpliest relationship between pi and e is

pi + e = pie

:)

Re: Mini-Challenge: Twiddling the bits

*Message #12 Posted by **Gerson W. Barbosa** on 24 June 2006, 12:13 p.m.,*
in response to message #11 by Crawl

> Quote:
>
> pi + e = pie

Very funny! :-)

And the first part of your posting is interesting indeed: a relationship between **e** and **pi** in the realm of real numbers. Like Valentin, I have an approximation for **pi** involving **e** that works in FIX 7 display mode:

e^{pi} - 20 + 1/1111

:-)

Edited: 24 June 2006, 5:43 p.m.

Re: Mini-Challenge: Twiddling the bits

*Message #13 Posted by **Crawl** on 24 June 2006, 11:18 p.m.,*
in response to message #12 by Gerson W. Barbosa

I don't purposefully try to come up with near identities, but every once in a while I notice them anyway. One I noticed not too long ago was

exp((3/2)^(1/3)) ~ pi [5 digits]

Re: Mini-Challenge: Twiddling the bits

*Message #14 Posted by **Valentin Albillo** on 25 June 2006, 1:24 a.m.,*
in response to message #13 by Crawl

One of the very best Pi-related near-identities is:

```
        3
     _____  Ln(640320)
   V 163
```

which agrees with Pi to 15 places no less ...

Best regards from V.

Re: Mini-Challenge: Twiddling the bits

*Message #15 Posted by **Antonio Maschio (Italy)** on 28 June 2006, 5:06 a.m.,*
in response to message #14 by Valentin Albillo

A simpler Pi-related near-identities is:

```
355/113
```

With more or less 6 decimals right. Simple, isn't it? But I guess you all already know it...

-- Antonio

Re: Mini-Challenge: Twiddling the bits

*Message #16 Posted by **Valentin Albillo** on 28 June 2006, 5:58 a.m.,*
in response to message #15 by Antonio Maschio (Italy)

Hi, Antonio:

You might want to try and compute the following integral, to the maximum precision you can:

$$\text{Integral between 0 and 1 of } \frac{x^8(1-x)^8(25+816x^2)}{3164(1+x^2)}\ .\ dx$$

and see if you can relate it to both Pi and your approximation (which, by the way, is correct to 8 digits to within 2 ulps, and was discovered 1500+ years ago by a chinese mathematician).

Best regards from V.

Re: Mini-Challenge: Twiddling the bits (pi error)

*Message #17 Posted by **Karl Schneider** on 28 June 2006, 11:43 p.m.,*
in response to message #16 by Valentin Albillo

Valentin --

Nice little integration problem!

I calculated the value of the integral on an HP-42S with ACC = 0.00001 (meaning that the acutal value of the integrand function can be assumed to lie within 0.001% of the calculated value).

The answer was $2.66764189313 \times 10^{-7}$

And, to 12 significant digits, $355/113 - 2.66764 \times 10^{-7}$ equals pi.

To those of you who evaluate the integral on an HP calculator other than an HP-42S or HP-71B: The value of the integrand is between 0 and 10^{-6} over the entire range, as far as I can tell. It's probably better to use SCI or ENG to specify the input accuracy.

The indefinite (symbolic) integral should exist, but there's a lot of tedious algebra to do. Does anyone know offhand if the CAS in the HP-49G or HP-49G+ can reduce the integrand to a 16th-degree polynomial and a term

$$\frac{(ax+b)}{3164(1+x^2)} \; ?$$

-- KS

Edited: 28 June 2006, 11:48 p.m.

Re: Mini-Challenge: Twiddling the bits (pi error)

*Message #18 Posted by **Valentin Albillo** on 29 June 2006, 4:41 a.m.,*
in response to message #17 by Karl Schneider

Hi, Karl:

Karl posted:

"Nice little integration problem!"

Thank you very much. It is indeed very nice, and easy too !

"I calculated the value of the integral on an HP-42S [...] The answer was 2.66764189313 x 10-7, and, to 12 significant digits, 355/113 - 2.66764x10-7 equals pi."

That's right. The *exact* value of this simple integral is indeed **355/113-Pi**. Nice, uh ? :-)

"The indefinite (symbolic) integral should exist, but there's a lot of tedious algebra to do. Does anyone know offhand if the CAS in the HP-49G or HP-49G+ can reduce the integrand to a 16th-degree polynomial and a term [...]"

Yes again. Being a rational function of two polynomials, it's elementarily integrable. This kind of rational functions usually result in a sum of polynomial terms plus some logarithms and some arctangents. The usual

approach is to perform synthetic division till the numerator's degree is less than the denominator's, then find the denominator's roots in order to break the remaining expression into a sum of fractions with denominators of degrees 1 and 2, then integrate each of them separately. All of this is very easily automated, even in a handheld calculator or computer.

That being so, I would expect the 49G/49G+ to have no problem at all to symbolically integrate it.

Thanks for your interest and

Best regards from V.

Re: Mini-Challenge: Twiddling the bits (pi error)

*Message #19 Posted by **Crawl** on 30 June 2006, 11:30 a.m.,*
in response to message #18 by Valentin Albillo

Indeed, the integral is pretty simple, a polynomial and an arctan term:

```
12/791 x^17
-102/791 x^16
+3151/6780 x^15
-703/791 x^14
+393/452 x^13
-23/113 x^12
-145/452 x^11
-5/791 x^10
+1409/3164 x^9
-4/7 x^7
+4/5 x^5
-4/3 x^3
+4 x
-4 atan(x)
```

The arctan term gives the pi, the polynomial the 355/113

Re: Mini-Challenge: Twiddling the bits (pi error)

*Message #20 Posted by **Karl Schneider** on 1 July 2006, 1:16 p.m.,*
in response to message #19 by Crawl

"Crawl" --

Thanks for posting the symbolic integral. What tool or software did you use? Expanding the numerator was easy enough on my HP-49G, but I didn't see any means of performing long division by the denominator. I haven't yet tried to do symbolic integration of this integrand on the 49G.

There is no natural-logarithm term in the solution. Did a = 0 in the remainder expression

$$\frac{(ax + b)}{3164(1+x^2)} \quad ?$$

Thanks,

-- KS

Edited: 1 July 2006, 1:19 p.m.

Re: Mini-Challenge: Twiddling the bits (pi error)

*Message #21 Posted by **Valentin Albillo** on 1 July 2006, 2:14 p.m.,*
in response to message #20 by Karl Schneider

Hi, Karl:

Karl posted:

"There is no natural-logarithm term in the solution."

> Informally speaking, exponentials and trigonometrics are essentially the same kind of functions, trigonometrics being just exponentials of imaginary arguments or exponentials being trigonometrics of imaginary arguments, whichever way you choose to look at it.
>
> Likewise with their inverses, logarithms and inverse trigonometrics, like arctangents for instance. They're both essentially the same, depending on the nature of their arguments.
>
> When dealing with rational functions of polynomials, you can always decompose the whole expression P(x)/Q(x) into a polynomial part (absent if P(x)'s degree is less than Q(x)'s), and a sum of fractions having either linear terms (degree 1) or irreductible quadratic terms (degree 2), where irreductible in this case means their roots are complex.

Every linear term, upon integration, results in a logarithm while every irreductible quadratic term results in arctangents. For this example, there are no linear terms in the denominator, just the irreductible quadratic $(1+x^2)$, which results in no logarithmic terms and just the one arctangent.

Best regards from V.

Re: Mini-Challenge: Twiddling the bits (pi error)

*Message #22 Posted by **Karl Schneider** on 2 July 2006, 5:55 p.m.,*
in response to message #21 by Valentin Albillo

Valentin --

Yes, you are absolutely right about the fundamental mathematics.

I was wondering why there was no "x" term in the *numerator* of the remainder of polynomial long division. That would have integrated to a natural-logarithm term, by substitution of variables $u = x^2$; $du = x\,dx$.

The x^9 term of the full numerator polynomial (prior to integration) was neatly eliminated by the long division, leaving only an x^8 remainder from the last term of the original polynomial. Thus, further division by (x^2+1) yielded only even-powered terms down to x^0, which yielded an arctan term after integration, but no logarithmic term that would have resulted from an x^1 term in the remainder.

Crawl's answer was exactly correct. I was able to obtain it by long division, an HP-49G to expand the original numerator, and a HP-32SII for fractions. Unfortunately, "INTVX" on the HP-49G did not return an exact answer of the symbolic integral for me -- it gave a polynomial with lengthy floating-point coefficients for each term. For all even powers of x, the coefficients were of the order 10^{-14} instead of 0. Such small values were also present as real-part components added to 2i in the cumbersome expression

$2i\ \ln(x-i) - 2i\ \ln(x+i)$,

which is 4 times arctan (1/x).

Setting "Exact mode" didn't help; the HP-49G refused to use it for this problem.

Crawl may have used some capable software to quickly obtain the answer (or, possibly -- *gasp!* -- a TI-89).

Best regards,

-- KS

Edited: 2 July 2006, 7:08 p.m.

Re: Mini-Challenge: Twiddling the bits

*Message #23 Posted by **Karl Schneider** on 25 June 2006, 6:03 p.m.,*
in response to message #1 by Valentin Albillo

Gerson has already posted the numerical solution, which is easy enough to do interactively:

The first 19 bits of *e* are

```
10 . 10 11 01 11 11 10 00 01 0

= 2.71827697754
```

The first 25 bits of pi are

```
11 . 00 10 01 00 00 11 11 11 01 10 10 1

= 3.14159262182
```

So, what is the probability that a given 19-bit string within the first 25 bits of two numbers will be present -- backwards or forwards -- in both numbers?

There are 2^{19} = 524,288 possible bit-string sequences. There are 7 x 7 = 49 possible combinations of starting bits in the two numbers. Allowing for forward or backward matching doubles the chances. So,

524288 / (49 * 2) = 5349.87755.

If this is a random coincidence, its probability of occurrence is about 1 in 5350 -- small, but not astoundingly so.

Many thanks to Kiyoshi for posting an HP-16C solution, as well as a concise HP-48GX solution. The HP-16C would seem to be the ideal tool for such a problem. I own one, but don't comprehend its function set well enough. The HP-16C doesn't even *have* the functions I would typically employ to tackle a similar

problem. As for RPL, I probably never will learn and understand it well enough to do anything very useful on my 28C, 48G, or 49G.

Thanks, Valentin.

-- KS

Edited: 25 June 2006, 11:42 p.m. after one or more responses were posted

Re: Mini-Challenge: Twiddling the bits

*Message #24 Posted by **Trent Moseley** on 25 June 2006, 10:52 p.m.,*
in response to message #23 by Karl Schneider

Karl,

Amen. So I gave my wife's grandnephew the 48GX and I've lived happily ever after.

tm

Re: Mini-Challenge: My Solution & Comments

*Message #25 Posted by **Valentin Albillo** on 26 June 2006, 8:40 a.m.,*
in response to message #1 by Valentin Albillo

Hi all,

Thanks for your interest and most specially your superb posted solutions and comments. I know I repeat myself but I continue to be amazed by your unfailing ingenuity. Now for my original solution and a few comments:

Kiyoshi Akima posted:

"Interesting challenge. Looking forward to your 71B program, V. (Though I doubt it's as short as 33 bytes :-))"

> Thanks a lot. My solution for the HP-71B is just an straightforward *2-liner*, using flags, but it isn't as short as your *awesome* 33-byte HP-16C solution, of course. HP-71B's BASIC takes more bytes to encode operations, which is hardly a problem when even the bare-bones machine has more than 80 times as much RAM as an HP-16C (and my 160 Kb one has 800+ times as much). Still, my 2-line solution stands at a meager 120 bytes, namely

```
1 FIX 7 @ CFLAG ALL @ E=EXP(1) @ T=2 @ FOR I=1 TO 19 @ IF E>=T THEN SFLAG I @ E=E-T
2 T=T/2 @ NEXT I @ T=1/2^23 @ S=3.125 @ FOR I=1 TO 19 @ S=S+T*FLAG(I) @ T=T*2 @ NEXT I @ DISP S
```

```
>RUN
      3.1415926
```

It could be made somewhat shorter by using Math ROM's base-conversion keywords, BVAL and BSTR$, but in this case the star of the show's actually the nice, unexpected result, not the program.

Karl Schneider posted:

"If this is a random coincidence, its probability of occurrence is about 1 in 5350 -- small, but not astoundingly so. [...] As for RPL, I probably never will learn and understand it well enough to do anything very useful on my 28C, 48G, or 49G."

It *is* a random coincidence, but besides its fairly low probabilty, the most astounding thing to me is the fact that, once again, the most important transcendental constants of all are involved. It couldn't be Sqrt(5) and Ln(2) or something, it had to be Pi and e ! :-)

As for your RPL comment, I agree with the spirit of what you say, which I interpret as being:

"These machines are so complicated that it doesn't pay spending vast amounts of time to get to know them at the same level of proficiency we can achieve with RPN models, even the most advanced ones (or the 71B), then try and remember what you've so expensively learned so that next time you need it, the necessary knowledge will instantly come to your mind.

Also, where you could write a quick program on the fly, by sheer, simple keystroke programming in RPN models (or the 71B), you'll find it utterly complicated to do likewise in RPL ones, where you'll find yourself forced to do things the way the RPL system enforces upon you, and where you'll find the vast majority of operations to be performed or programmed have to do with the stack and housekeeping, instead of with the algorithmic details of your particular problem at hand."

See you all in S&SMC#17, due next week (... and it won't be *that* simple. You've been warned :-) ...)

Best regards from V.

Edited: 26 June 2006, 8:46 a.m.

Re: Mini-Challenge: My Solution & Comments

*Message #26 Posted by **Kiyoshi Akima** on 26 June 2006, 8:22 p.m.,*
in response to message #25 by Valentin Albillo

I hope this post doesn't show up twice; my browser just hiccuped.

V, why all the flags and *two* loops? The problem statement only involved one loop, the solution should do the same.

What about something like:

```
1 E=EXP(1) @ P=3.125 @ F=2 @ R=2^-23 @ FOR I=1 TO 19 @ IF F<E THEN P=P+R @ E=E-F
2 F=F/2 @ R=R*2 @ NEXT I @ FIX 7 @ DISP P
```

It still won't fit on one line, but...

Or, quick and dirty in RPL for the HP48:

```
<<
 e ->NUM 3.125 2 DUP -23 ^ -> E P F R
 <<
  1 19 START
  IF F E <
  THEN
   'P' R STO+
   'E' F STO-
  END
  'F' 2 STO/
  'R' 2 STO*
  NEXT P
 >>
>>
```

Unfortunately, this takes twice as long as my original program and is about a third again as big.

The same algorithm, but this time on the stack:

```
<<
 2 DUP -23 ^ e ->NUM 3.125
 1 19 START
  IF OVER 5 PICK >
  THEN
   3 PICK + SWAP 4 PICK - SWAP
  END
  4 ROLL 2 / 4 ROLL 2 * 4 ROLL 4 ROLL
 NEXT
 4 ROLLD 3 DROPN
>>
```

Doing everything on the stack is faster and more compact, but it's still about ten percent larger and ten percent slower than my original.

I guess this is just another example of the rule stating that the slowest parts of a program are usually the conditionals. My original program didn't have any **IF**s, just straight-line code inside a loop.

Re: Mini-Challenge: My Solution & Comments

*Message #27 Posted by **Valentin Albillo** on 27 June 2006, 4:17 a.m.,*
in response to message #26 by Kiyoshi Akima

Hi,

Kiyoshi Akima wrote:

"V, why all the flags and two loops?"

> It wasn't actually a 'program optimization' or 'best-program' challenge so I never gave it a thought, I coded the first idea that passed through my mind, as my only intention was to demonstrate the nice coincidence and have others find it by themselves. How efficiently or inefficiently they managed to do it was absolutely irrelevant as long as the shocking, correct answer was returned.
>
> For real 'optimized' programming, stay tuned for my next S&SMC#17, you'll have a fairly decent opportunity to show off your programming abilities ! That is, if you dare try and solve it ... :-)

Thanks for your clever comments and code and

Best regards from V.

Re: Mini-Challenge: My Solution & Comments

*Message #28 Posted by **Kiyoshi Akima** on 27 June 2006, 1:25 p.m.,*
in response to message #26 by Kiyoshi Akima

Oh, I'm so stupid!

My first attempts showed that it wasn't necessary to test individual bits. My later attempts showed that it wasn't necessary to wholesale conversions to/from binary. So, even though this wasn't an optimization contest or a shortest program contest, by combining the two ideas above, we get this RPL program for the 48:

```
<<
 e ->NUM 2 / 0
 DUP 18 START
```

```
  OVER IP + 2 / SWAP FP DUP + SWAP
 NEXT
 SWAP DROP 16 / 3.125 +
>>
```

Under a hundred bytes and runs in under a third of a second.

For those that prefer RPN, a straightforward translation yields this for the 41:

```
LBL 'BT
19
1
E^X
2
/
.
LBL 00
RCL Y
INT
+
2
/
X<>Y
FRC
ENTER
+
X<>Y
DSE Z
GTO 00
50
+
16
/
FIX 7
END
```

Other, "pure" RPN machines will need to put the loop counter into a register and manipulate the stack to get around the **RCL Y**.

And finally, for the 71, by eliminating the bit tests we get V's ideal program, this one-liner:

```
1 E=EXP(1)/2 @ P=0 @ FOR I=1 TO 19 @ P=(P+IP(E))/2 @ E=FP(E)*2 @ NEXT I @ P=(50+P)/16 @ DISP P
```

Unfortunately, line-length limitations preclude the use of a **FIX 7** step.

Re: Mini-Challenge: My Solution & Comments

*Message #29 Posted by **Valentin Albillo** on 27 June 2006, 5:21 p.m.,*
in response to message #28 by Kiyoshi Akima

Hi,

Kiyoshi Akima posted:

"Unfortunately, line-length limitations preclude the use of a FIX 7 step. "

> Nope. Not only can you put in that **FIX 7** "step" (statement, more like) in your ingenious concoction, but you can add a necessary **DESTROY ALL** statement as well, which would render the following **P=0** assignment rather pointless, but what the heck, we'll let that in as well, just for fun. This is how you enter and run it: at the command prompt key in:

```
>1FIX7@DESTROYALL@E=EXP(1)/2@P=0@FORI=1TO19@P=(P+IP(E))/2@E=FP(E)*2@NEXTI@P=(50+P)/16@P [press ENTER here]

>LIST

1 FIX 7 @ DESTROY ALL @ E=EXP(1)/2 @ P=0 @ FOR I=1 TO 19 @ P=(P+IP(E))/2 @ E=FP(E)*2 @ NEXT I @
P=(50+P)/16 @ DISP P

>RUN

 3.1415926
```

Best regards from V.

Re: Mini-Challenge: My Solution & Comments

*Message #30 Posted by **Chris Dean** on 29 June 2006, 2:24 a.m.,*
in response to message #29 by Valentin Albillo

```
Another RPL solution (HP49G+) for the problem which takes a
slightly different approach.


<<
```

```
   3.125 'S' STO
   0 'T' STO
   1 -17 FOR N
      IF T 2 N ^ + e <= THEN
         2 N ^ 'T' STO+
         2 N 22 + NEG ^ 'S' STO+
      END
   -1 STEP
    S 7 FIX
>>


'MINI'
STO


Running the program gives 3.1415926 as required.


The loop 1 to -17 represents the 19 bits 1, 0, -1, ... -17 and the
value 22 (17 + 5) represents 19 bits plus a right bit shift of 5 places to
account for 11.0010.


A nice little problem.


Regards


Chris Dean
```

Edited: 29 June 2006, 8:16 a.m.

RPL vs. RPL-based models

*Message #31 Posted by **Karl Schneider** on 27 June 2006, 12:02 a.m.,*
in response to message #25 by Valentin Albillo

Valentin stated,

> Quote:
>
> As for your RPL comment, I agree with the spirit of what you say, which I interpret as being: [...]

Well, that might be a tad more expansive and eloquent than what I had in mind. I should probably refine my original statement

> Quote:
>
> *As for RPL, I probably never will learn and understand it well enough to do anything very useful on my 28C, 48G, or 49G.*

to the following:

I probably never will learn and understand the RPL programming paradigm well enough to develop an application program of substance on my 28C, 48G, or 49G.

Now, that does not preclude my using the built-in functions individually (perhaps with some help from the manual) in order to exploit a capability not found on any RPN-based calculator (excluding the 71B) -- e.g., compound unit conversions (28C), multiple integrals (48G/49G), equation reference library (48G), or primes and factorization (49G).

That also does not preclude me from downloading programs by hardwire I/O to my 48G or 49G, or writing my own simple RPL programs.

My "beef" isn't really with the RPL-based calculators, as the models introduced through 1993 included some good ideas, nice improvements, and new functions. Rather, it's with RPL itself.

Just wanted to make that distinction, not to start that old thread again...

-- KS

Edited: 27 June 2006, 12:03 a.m.

[Return to Index | Top of Index]

Go back to the main exhibit hall