Pending ASSIGNMENTS as of May 12, 1988

1. (HARD!) Explain why, for IEEE 754,

$$\left[\left[i/(2^j+2^k)\right]_{\text{rounded}} \times (2^j+2^k)\right]_{\text{rounded}} = i$$

for all moderate sized integers i, j, k.
But not for other ways to round!

2. How to orchestrate a program that solves $f(x)=0$ when $f(x)$ is like $\ln(x)\cdot\sqrt{10-x}$.

3. Recommend an expedient remedy to cure CRAY's negative AMOD (positive, positive) .

4. In a vanilla higher-level language like FORTRAN, program a way to discover the full range of any machine's INTEGER format.

5. Program for arbitrary continued fractions that survives divide-by-zero.

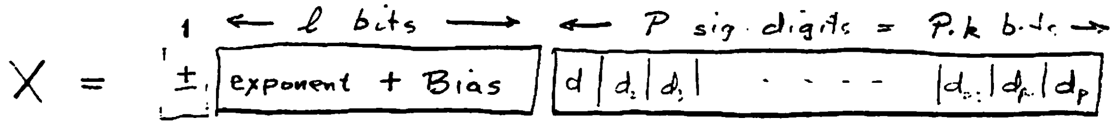
Kahan
12 May

FLOATING-POINT
RANGE / PRECISION TRADEOFF FOR RADICES $\beta = 2^k$, $k = 1, 2, 3, \ldots$.

WHICH RADIX IS BEST?

Name	β	k	Who?
BINARY	2	1	IEEE 754, DEC VAX, CDC, CRAY, ...
QUATERNARY	4	2	--- no more ---
OCTAL	8	3	Burroughs B65xx
HEXADECIMAL	16	4	IBM 370, Amdahl, ...

Floating-point word:



1
← ℓ bits →
← P sig. digits = P.k bits →
X =
±
exponent + Bias
d | d_2 | d_3 | - - - - | d_{p-2} | d_{p-1} | d_p

value $\quad x = \pm \beta^{\text{exponent}} \times [d_1 d_2 d_3 \cdots d_{p-2} d_{p-1} d_p]$

where $\quad 0 \leq \text{exponent} + \text{Bias} \leq 2^{\ell} - 1$

$0 \leq [d_1 d_2 d_3 \cdots d_{p-2} d_{p-1} d_p] \leq \beta^{p} - 1$

and $\beta = 2^k$ for some fixed k

Total wordsize $\quad w = 1 + \ell + p.k$ bits

Let $\rho = \beta - 1$ so $[00\ldots00] \leq [d_1 d_2 \ldots d_{p-1} d_p] \leq [\rho\rho \quad \rho\rho]$

$\updownarrow$

Normally X is NORMALIZED : $d_1 \geq 1$ unless $x = 0$.

$$X = \boxed{\pm}\boxed{\text{exponent + Bias}}\boxed{d.d_2 \;----\; d_{p-1}\;d_p}$$

$$x = \pm\, \beta^{\text{exponent}} \times [d_1 d_2 \cdots d_{p-1}\, d_p] \qquad \text{Normalized} \neq 0,$$

$$\beta = 2^k \qquad \rho = \beta - 1 \qquad d_1 \geq 1.$$

$$0 \leq \text{exponent+Bias} \leq 2^\ell - 1$$

RANGE:

$$\frac{\text{Max. } x}{\text{Min. } x > 0} = \frac{\beta^{2^\ell - 1 - \text{Bias}} \times [\rho\rho \cdots \rho\rho]}{\beta^{0 - \text{Bias}} \times [10 \cdots 00]}$$

$$= \frac{\beta^{2^\ell - 1} \times (\beta^p - 1)}{\beta^{p-1}} \doteqdot \beta^{2^\ell} = 2^{k \cdot 2^\ell}$$

WORST-CASE PRECISION:

$$\underset{x>0}{\text{Max.}} \frac{(\text{Successor of } x) - x}{x} = \frac{[100\cdots001] - [100\cdots000]}{[100\cdots000]}$$

$$= 1/\beta^{p-1} = 2^{k \cdot (p-1)}$$

What BINARY format has the same RANGE and WORST-CASE PRECISION?

Say ℓ' exponent bits, where $2^{1 \cdot 2^{\ell'}} = 2^{k \cdot 2^\ell}$

p' significant bits, where $2^{1 \cdot (p'-1)} = 2^{k \cdot (p-1)}$

i.e. $\ell' = \ell + \log_2 k$, $p' = 1 + k \cdot (p-1)$

For "same" RANGE & WORST-CASE PRECISION

	$\beta = 2^k$	$\beta' = 2$
Exponent field #bits	ℓ	$\ell' = \ell + \log_2 k$
Sig. dig. field #bits	pk	$p' = 1 + k \cdot (p-1)$
Total wordsize	$w = 1 + \ell + pk$	$w' = 1 + \ell' + p'$

Hence

$$\begin{aligned} w - w' &= \ell - \ell' + pk - p' \\ &= -\log_2 k + k - 1 \\ &\geq 0 \quad \text{for all} \quad k \geq 1. \end{aligned}$$

Name	k	lost bits $w - w' = -\log_2 k + k - 1$
BINARY	1	0 $\quad -1$ for Hidden Bit!
QUATERNARY	2	0
OCTAL	3	$2 - \log_2 3 = 0.415$
HEX.	4	1

WITHOUT HIDDEN BIT (Goldberg's variation),

BINARY matches QUATERNARY's RANGE/PRECISION.

WITH HIDDEN BIT,

BINARY beats QUATERNARY by 1 bit
OCTAL by 1.415 bits
HEX by 2 bits.

And then there is WOBBLING PRECISION

```
10   '  WHICHINT.BAS  is a BASIC program to discover which integers the
20   '  computer on which it runs can handle in its  INTEGER  format.
30     DEFINT A-Z ' ...  or  INTEGER ...  in other BASIC dialects.
40     O1 = 1 :  IF (O1>0 AND O1*O1=O1) THEN 60
50        PRINT "Something is  VERY  wrong with  1 ." :  STOP
60     O2 = O1+O1 ' ... Test the hypothesis that the machine is  BINARY :
70     P = O2 :  J = O2+O1 ' ...  j = 2^P - 1
80     ON ERROR GOTO 220 ' ... and resume at 120
90        P = P+O1 :  I = J :  J = I+I+O1 :  D = (J-I) - I
100       IF D><O1 THEN PRINT "FLOATING-POINT is used for INTEGERS." :  STOP
110       IF J>I THEN 90 ' ...  else now  i = 2^(P-1)-1 >= j = i+i+1 . !
120    ON ERROR GOTO 230 ' ... and resume at 140
130    J = I+O1 :  IF J>I THEN 300 ' ...  else now the machine IS binary.
140    ON ERROR GOTO 240 ' ... and resume at 160
150    M = -I :  IF M<0 THEN 170 ' ...  This ought not to overflow,  but ...
160       PRINT "Negative integers malfunction!" :  STOP
170    ON ERROR GOTO 250 ' ... and resume at 200
180    J = M-O1 :  IF J>=M THEN 200
190       PRINT P;" digits of Twos' complement"; :  GOTO 210
200       PRINT P;" digits of either  Sign-Magnitude or Ones' complement";
210       PRINT "  BINARY  (B = 2)." :  STOP
220                  RESUME 120    ' ...   IBM PC  BASIC  requires these
230                   RESUME 140   ' ...   RESUME statements to prevent
240                    RESUME 160  ' ...   subsequent  "ERRORS"  from
250                     RESUME 200 ' ...   terminating the program.
300    O3 = O2+O1 ' ... Test the hypothesis that the machine is  TERNARY :
310    P = O2 :  J = O3+O1 ' ...  j = (3^P - 1)/2
320    ON ERROR GOTO 490 ' ... and resume at 350
330       P = P+O1 :  I = J :  J = I+I+I+1
340       IF J>I THEN 330 ' ...  else now  i = (3^(P-1)-1)/2 >= j = 3i+1 . !
350    ON ERROR GOTO 500 ' ... and resume at 370
360    J = I+O1 :  IF J>I THEN 410 ' ...  else now  i = 111...111  is maximal.
370    ON ERROR GOTO 240 ' ... and resume at 160
380    M = -I ' ...  This ought not to overflow,  but ...
390       IF M>=0 THEN 160 ' ... else now  m = 222...222 or TTT...TTT < 0 .
400       PRINT P;" digits of  Threes' complement  or  Balanced";: GOTO 480
410    ON ERROR GOTO 510 ' ... and resume at 600
420    J = I+I :  IF J<=I THEN 600 ' ... else  j = 222...222 > 0 .
430    ON ERROR GOTO 240 ' ... and resume at 160
440    M = -J :  IF M>= 0 THEN 160 ' ... else now  m = -222...222 < 0 .
450    ON ERROR GOTO 510 ' ... and resume at 600
460    J = J+1 :  IF J>I THEN 600 ' ...  else now  222...222  is maximal.
470       PRINT P;" digits and a sign for  Sign-Magnitude";
480       PRINT "  TERNARY  (B = 3) ." :  STOP
490                 RESUME 350
500                  RESUME 370
510                   RESUME 600
600    N = O3*O3 :  T = N+1 ' ... Check that the machine really is  DECIMAL :
610    P = O1 :  J = O3+O1 ' ...  j = 5*10^P - 1
620    ON ERROR GOTO 810 ' ... and resume at 650
630       P = P+1 :  I = J :  J = T*I+N
640       IF J>I THEN 630 ' ... else now i = 499...999 >= j = 10*i+9 . !
650    ON ERROR GOTO 820 ' ... and resume at 670
660    K = I+O1 :  IF K>I THEN 700 ' ... else now  i  is maximal.
670    ON ERROR GOTO 240 ' ... and resume at 160
680    M = -I-O1 :  IF M>=0 THEN 160 ' ... else now  m = -500...000 < 0 .
690       PRINT P;" digits of  Tens' complement"; :  GOTO 770
700    ON ERROR GOTO 800 ' ... and stop
710    K = I+K :  IF K<=I THEN 800 ' ... else  k = 999...999 > 499...999 .
720    ON ERROR GOTO 830 ' ... and resume at 740
730    J = K+O1 :  IF J>K THEN 800 ' ... else  k  is maximal.
740    ON ERROR GOTO 240 ' ... and resume at 160
750    M = -K :  IF M>=0 THEN 160
760       PRINT P;" digits and a sign for  Sign-Magnitude";
770       PRINT "  DECIMAL  (B = 10) ." :  STOP
800    PRINT "This program can't tell what happens to integers > ";I :  STOP
810               RESUME 650
820                RESUME 670
830                 RESUME 740 :  END
```