SPHERE
NEWSLETTER

APRIL 1980 *VOLUME IV ISSUE 5*

EDITORS:

JEFF BROWNSTEIN
ROGER J. SPOTT

SOFTWARE

HARDWARE

EDITOR'S MAILBAG

PLEASE SEND TYPED MATERIAL FOR NEXT ISSUE TO: *JEFFREY BROWNSTEIN*
2 TOR ROAD
WAPPINGERS, N.Y. 12590

WE ARE ABSOLUTELY OUT OF MATERIAL

PLEASE SEND ARTICLES SOON !

Jeff + Roger

```
0010 LINE= 0
0020 STRING= 72
0030 REM HAIKU MODIFIED FROM PROGRAM IN AUG 79 KILOBAUD
0040 PRINT : DIM R(0)
0050 PRINT
0060 LET A1=4 : REM ARTICLE NUMBER
0070 LET A2=50: REM ADJECTIVE NUMBER
0080 LET N1=50: REM NOUN NUMBER
0090 LET V1=13: REM VERB NUMBER
0100 LET P1=7 : REM PREPOSITION NUMBER
0110 GOTO 140
0120 LET L$=""
0130 RESTORE
0140 PRINT : PRINT
0150 LET R1=INT(RND(0)*4+1)
0160 ON R1 GOTO 470, 690, 910, 1130
0170 REM * ARTICLE-CHECKING SUBROUTINE
0180 FOR I=1 TO LEN(L$)-2
0190 IF MID$(L$,I,3)=" A " THEN B$=MID$(L$,I+3,1) : GOTO 210
0200 GOTO 260
0210 IF B$="A" THEN L$=LEFT$(L$,I+1)+"N"+MID$(L$,I+2)
0220 IF B$="E" THEN L$=LEFT$(L$,I+1)+"N"+MID$(L$,I+2)
0230 IF B$="I" THEN L$=LEFT$(L$,I+1)+"N"+MID$(L$,I+2)
0240 IF B$="O" THEN L$=LEFT$(L$,I+1)+"N"+MID$(L$,I+2)
0250 IF B$="U" THEN L$=LEFT$(L$,I+1)+"N"+MID$(L$,I+2)
0260 NEXT I
0270 FOR I=1 TO LEN(L$)-2
0280 IF MID$(L$,I,4)=" AN " THEN B$=MID$(L$,I+4,1) : GOTO 300
0290 GOTO 360
0300 IF B$="A" THEN 360
0310 IF B$="E" THEN 360
0320 IF B$="I" THEN 360
0330 IF B$="O" THEN 360
0340 IF B$="U" THEN 360
0350 LET L$=LEFT$(L$,I+1)+MID$(L$,I+3)
0360 NEXT I
0370 RETURN
0380 REM WORD-CHOOSING SUBROUTINE
0390 LET R=RND(0)*N+1+P
0400 FOR I=1 TO R
0410 READ W$
0420 NEXT I
0430 LET L$=L$+" "+W$
0440 RESTORE
0450 RETURN
0460 REM *       FIRST HAIKU PATTERN
0470 LET N=A1 : P=0  : GOSUB 390
0480 LET N=A2 : P=A1 : GOSUB 390
0490 LET N=N1 : P=A1+A2 : GOSUB 390
0500 GOSUB 180
0510 LET L$=L$+"..."
0520 PRINT L$
0530 LET L$=""
0540 LET N=A1 : P=0 : GOSUB 390
0550 LET N=N1 : P=A1+A2 : GOSUB 390
```

```
0560 LET N=V1 : P=A1+A2+N1 : GOSUB 390
0570 LET N=P1 : P=A1+A2+N1+V1 : GOSUB 390
0580 LET N=A1 : P=0 : GOSUB 390
0590 LET N=N1 : P=A1+A2 : GOSUB 390
0600 GOSUB 180
0610 PRINT L$;""
0620 LET L$=""
0630 LET N=A2 : P=A1 : GOSUB 390
0640 GOSUB 390
0650 LET N=N1 : P=A1+A2 : GOSUB 390
0660 PRINT L$
0670 GOTO 120
0680 REM *       SECOND HAIKU PATTERN
0690 LET N=N1 : P=A1+A2 : GOSUB 390
0700 LET N=P1 : P=A1+A2+N1+V1 : GOSUB 390
0710 LET N=A1 : P=0 : GOSUB 390
0720 LET N=N1 : P=A1+A2 : GOSUB 390
0730 GOSUB 180
0740 LET L$=L$+";"
0750 PRINT L$;""
0760 LET L$=""
0770 LET N=A1 : P=0  : GOSUB 390
0780 LET N=A2 : P=A1 : GOSUB 390
0790 LET N=N1 : P=A1+A2 : GOSUB 390
0800 LET N=P1 : P=A1+A2+N1+V1 : GOSUB 390
0810 LET N=A1 : P=0  : GOSUB 390
0820 LET N=N1 : P=A1+A2 : GOSUB 390
0830 GOSUB 180
0840 PRINT L$;""
0850 LET L$=""
0860 LET N=A2 : P=A1 : GOSUB 390
0870 LET N=N1 : P=A1+A2 : GOSUB 390
0880 PRINT L$
0890 GOTO 120
0900 REM *       THIRD HAIKU PATTERN
0910 LET N=A1 : P=0  : GOSUB 390
0920 LET N=A2 : P=A1 : GOSUB 390
0930 GOSUB 390
0940 LET N=N1 : P=A1+A2 : GOSUB 390
0950 GOSUB 180
0960 LET L$=L$+";"
0970 PRINT L$;""
0980 LET L$=""
0990 LET N=P1 : P=A1+A2+N1+V1 : GOSUB 390
1000 LET N=A1 : P=0  : GOSUB 390
1010 LET N=A2 : P=A1 : GOSUB 390
1020 LET N=N1 : P=A1+A2 : GOSUB 390
1030 GOSUB 180
1040 PRINT L$;""
1050 LET L$=""
1060 LET N=A1 : P=0  : GOSUB 390
1070 LET N=N1 : P=A1+A2 : GOS B 390
1080 LET N=V1 : P=A1+A2+N1 : GOSUB 390
1090 GOSUB 180
1100 PRINT L$
1110 GOTO 120
```

```
1120 REM *       FOURTH HAIKU PATTERN
1130 LET N=A1 : P=0  : GOSUB 390
1140 LET N=A2 : P=A1 : GOSUB 390
1150 LET N=N1 : P=A1+A2 : GOSUB 390
1160 LET N=V1 : P=A1+A2+N1 : GOSUB 390
1170 GOSUB 180
1180 LET L$=L$+";"
1190 PRINT L$
1200 LET L$=""
1210 LET N=A1 : P=0  : GOSUB 390
1220 LET N=A2 : P=A1 : GOSUB 390
1230 GOSUB 390
1240 LET N=N1 : P=A1+A2 : GOS B 390
1250 GOSUB 180
1260 PRINT L$
1270 LET L$=""
1280 LET N=P1 : P=A1+A2+N1+V1 : GOSUB 390
1290 LET N=A1 : P=0  : GOSUB 390
1300 LET N=A2 : P=A1 : GOSUB 390
1310 LET N=N1 : P=A1+A2 : GOSUB 390
1320 GOSUB 180
1330 PRINT L$
1340 GOTO 120
1350 REM *       ARTICLES
1360 DATA "A","THE","AN","THE"
1370 REM *       ADJECTIVES
1380 DATA "AUTUMN","HIDDEN","BITTER","MISTY","SILENT","EMPTY"
1390 DATA "DRY","DARK","SUMMER","ICY","DELICATE","QUIET"
1400 DATA "WHITE","SUDDEN","COOL","SPRING","WINTER","DAPPLED"
1410 DATA "TWILIGHT","DAWN","CRIMSON","WISPY","AZURE"
1420 DATA "BLUE","BILLOWING","BROKEN","COLD","DAMP","FALLING"
1430 DATA "FROSTY","GREEN","LONG","LATE","LINGERING","LIMPID"
1440 DATA "LITTLE","MORNING","MUDDY","ORANGE","OLD","RED"
1450 DATA "STILL","SMALL","SPARKLING","THROBBING","WANDERING"
1460 DATA "WITHERED","WILD","BLACK","YOUNG"
1470 REM *       NOUNS
1480 DATA "SCARECROW","WATERFALL","RIVER","BREEZE","MOON"
1490 DATA "RAIN","WIND","SEA","MORNING","SNOW","LAKE","SUNSET"
1500 DATA "PINE","SHADOW","LEAF","DAWN","GLITTER","FOREST"
1510 DATA "HILL","CLOUD","MEADOW","SUN","GLADE","BIRD"
1520 DATA "BUTTERFLY","BUSH","CROW","DEW","DUST","FIELD"
1530 DATA "FLOWER","FIREFLY","FEATHER","GRASS","MOUNTAIN"
1540 DATA "NIGHT","POND","PINE-CONE","SHADE","SNOWFLAKE"
1550 DATA "SILENCE","SOUND","SKY","SHAPE","SURF","THUNDER"
1560 DATA "VIOLET","WATER","WILDFLOWER","WAVE"
1570 REM *       VERBS
1580 DATA "SHAKES","DRIFTS","HAS TURNED","STRUGGLES"
1590 DATA "HAS FALLEN","HAS PASSED","SLEEPS","CREEPS"
1600 DATA "FLUTTERS","HAS RISEN","IS FALLING","IS TRICKLING"
1610 DATA "IS FLOATING"
1620 REM *       PREPOSITIONS
1630 DATA "ON","IN","WITH","OF","THROUGH","BEHIND","UNDER"
1640 END
```

```
I'M READY
```

```
0010 REM CHECKR From SBVC RSTS DISC
0020 LINE= 80
0030 PRINT "    This program plays checkers. The computer is X,"
0040 PRINT "and you are O.  The computer will go first  *NOTE:"
0050 PRINT "Squares are in the form-(X,Y) and SQ. 1,1 is the bottom"
0060 PRINT "left*  Do not attempt a double jump or your piece might"
0070 PRINT "just disappear (same for a triple jump!)"
0080 PRINT "    Wait for me to move first !!!!"
0090 LET G=-1
0100 DIM R(50)
0110 LET L=-1
0120 REM
0130 DATA 1,0,1,0,0,0,-1,0,0,1,0,0,0,-1,0,-1,15
0140 FOR X=1 TO 8
0150 FOR Y=1 TO 8
0160 READ J
0170 IF J=15 THEN 200
0180 LET S(X,Y)=J
0190 GOTO 220
0200 RESTORE
0210 READ S(X,Y)
0220 NEXT Y
0230 NEXT X
0240 REM
0250 LET L=-L
0260 FOR X=1 TO 8
0270 FOR Y=1 TO 8
0280 IF S(X,Y)=0 THEN 370
0290 IF G>0 THEN 320
0300 IF S(X,Y)>0 THEN 370
0310 GOTO 330
0320 IF S(X,Y)<0 THEN 370
0330 IF ABS(S(X,Y))<>1 THEN 350
0340 GOSUB 450
0350 IF ABS(S(X,Y))<>2 THEN 370
0360 GOSUB 2000
0370 IF X<>8 THEN 400
0380 IF L=1 THEN 400
0390 RETURN
0400 NEXT Y
0410 NEXT X
0420 PRINT
0430 GOSUB 1160
0440 GOTO 250
0450 FOR A=-1 TO 1 STEP 2
0460 LET U=X+A
0470 LET V=Y+G
0480 IF U<1 THEN 650
0490 IF U>8 THEN 650
0500 IF V<1 THEN 650
0510 IF V>8 THEN 650
0520 IF S(U,V)<>0 THEN 550
0530 GOSUB 930
0540 GOTO 650
0550 IF S(U,V)=G THEN 650
```

```
0560 IF S(U,V)=2*G THEN 650
0570 LET U=U+A
0580 LET V=V+G
0590 IF U<1 THEN 650
0600 IF U>8 THEN 650
0610 IF V<1 THEN 650
0620 IF V>8 THEN 650
0630 IF S(U,V)<>0 THEN 650
0640 GOSUB 930
0650 NEXT A
0660 RETURN
0670 REM KING MOVES
0680 FOR A=-1 TO 1 STEP 2
0690 FOR B=-1 TO 1 STEP 2
0700 LET U=X+A
0710 LET V=Y+B
0720 IF U<1 THEN 890
0730 IF U>8 THEN 890
0740 IF V<1 THEN 890
0750 IF V>8 THEN 890
0760 IF S(U,V)<>0 THEN 790
0770 GOSUB 930
0780 GOTO 890
0790 IF S(U,V)=G THEN 890
0800 IF S(U,V)=2*G THEN 890
0810 LET U=U+A
0820 LET V=V+B
0830 IF U<1 THEN 890
0840 IF U>8 THEN 890
0850 IF V<1 THEN 890
0860 IF V>8 THEN 890
0870 IF S(U,V)<>0 THEN 890
0880 GOSUB 930
0890 NEXT B
0900 NEXT A
0910 RETURN
0920 GOTO 1450
0930 REM *
0940 LET P=P+1
0950 IF P=K THEN 1260
0960 IF V<>(4.5+(3.5*G)) THEN 980
0970 LET Q=Q+2
0980 IF X<>(4.5-(3.5*G)) THEN 1000
0990 LET Q=Q-2
1000 REM *
1010 IF U<>1 THEN 1030
1020 LET Q=Q+1
1030 IF U<>8 THEN 1050
1040 LET Q=Q+1
1050 FOR C=-1 TO 1 STEP 2
1060 IF S(U+C,V+G)<1 THEN 1100
1070 LET Q=Q+1
1080 IF S(U-C,V-G)<>0 THEN 1100
1090 LET Q=Q-1
1100 REM THIS WAS THE EVALUATION SECTION
1110 REM
```

```
1120 NEXT C
1130 LET R(P)=Q
1140 LET Q=0
1150 RETURN
1160 IF P=0 THEN 1960
1170 FOR J=10 TO -10 STEP -1
1180 FOR F=1 TO P
1190 IF R(F)=J THEN 1230
1200 NEXT F
1210 NEXT J
1220 LET F=F+1
1230 LET K=F+P
1240 GOSUB 250
1250 RETURN
1260 PRINT "I move from (";X;Y;") to (";U;V;")"
1270 LET F=0
1280 LET P=0
1290 LET K=0
1300 IF V<>(4.5+(3.5*G)) THEN 1330
1310 LET S(U,V)=2*G
1320 GOTO 1340
1330 LET S(U,V)=S(X,Y)
1340 LET S(X,Y)=0
1350 IF ABS(X-U)<>2 THEN 1370
1360 LET S((X+U)/2,(Y+V)/2)=0
1370 REM
1380 REM
1390 REM
1400 GOSUB 1440
1410 RETURN
1420 GOSUB 1670
1430 RETURN
1440 PRINT
1450 FOR Y=8 TO 1 STEP -1
1460 IF Y<>8 THEN 1510
1470 PRINT : FOR X1=1 TO 8
1480 IF X1=1 THEN PRINT "    ";
1490 PRINT X1;
1500 NEXT X1 : PRINT : PRINT
1510 PRINT Y;
1520 FOR X=1 TO 8
1530 LET I=2*X+2
1540 IF S(X,Y)<>0 THEN 1560
1550 PRINT TAB(I); "·";
1560 IF S(X,Y)<>1 THEN 1580
1570 PRINT TAB(I);"O";
1580 IF S(X,Y)<>-1 THEN 1600
1590 PRINT TAB(I);"X";
1600 IF S(X,Y)<>-2 THEN 1620
1610 PRINT TAB(I);"X";TAB(I);"*";
1620 IF S(X,Y)<>2 THEN 1640
1630 PRINT TAB (I);"O";TAB(I);"*";
1640 NEXT X
1650 PRINT
1660 NEXT Y
1670 PRINT
```

```
1680 PRINT "FROM";
1690 INPUT E,H
1700 LET X=E
1710 LET Y=H
1720 IF S(X,Y)<>0 THEN 1760
1730 PRINT "There is no one occupying that space"
1740 PRINT
1750 GOTO 1680
1760 PRINT "TO";
1770 INPUT A,B
1780 LET X=A
1790 LET Y=B
1800 IF S(X,Y)=0 THEN 1840
1810 PRINT "That space is already occupied"
1820 PRINT
1830 GOTO 1760
1840 LET S(A,B)=S(E,H)
1850 LET S(E,H)=0
1860 LET T=(4.5-(3.5*G))
1870 IF ABS(E-A)<>2 THEN 1890
1880 LET S((E+A)/2,(H+B)/2)=0
1890 IF B<>T THEN 1910
1900 LET S(A,B)=-2*G
1910 FOR X=8 TO 8
1920 FOR Y=8 TO 8
1930 RETURN
1940 NEXT Y
1950 NEXT X
1960 PRINT "       Very good, YOU WIN!!!!"
1970 PRINT
1980 PRINT
1990 PRINT "BYE  for now....."
2000 END
```

I'M READY

8

```
6000 REM SINGLE ROOT OF A REAL CONTINUOUS FUNCTION
6002 REM
6005 REM INPUTS, (X0,X2)= INTERVAL, FNA(X)= FUNCTION
6010 REM OUTPUT, X3= ROOT OF FNA(X)
6015 REM
6020 X1= (X0+X2)/2:D= X1-X0
6025 F0= FNA(X0):F1= FNA(X1):F2= FNA(X2):T1= F1/F0:T2= F2/F0
6030 X3= X1+D*(T1/SQR(T1*T1-T2))
6035 IF ABS(X3-X2) < 1E-8 RETURN
6040 F3= FNA(X3):IF F3*F2 < 0 THEN X0= X2:X2= X3:GOTO 6020
6045 IF F3*F1 < 0 THEN X0= X1:X2= X3:GOTO 6020
6050 IF F3*F0 < 0 THEN X2= X3:GOTO 6020
```

REFERENCE,

C.J.F. RIDDERS,

"IEEE TRANSACTIONS ON CIRCUITS & SYSTEMS", NOV. 79

ROOT OF EQUATION.

$F(X) = 0$

ON INTERVAL $[X0, X2]$.

NOTES

CONVERGENCE IS QUADRATIC OR BETTER.

DOES NOT REQUIRE DERIVATIVES OF $F(X)$ AS DOES NEWTON'S METHOD AND OTHERS.

DEFINE FNA(X) IN MAIN PROGRAM BEFORE CALLING SUBROUTINE.

A New Algorithm for Computing a Single Root of a Real Continuous Function

C. J. F. RIDDERS

Abstract—**A fast and simple iterative method is proposed for the determination of a single real root of a real continuous function. The idea is based upon linearizing the original function whereafter the *regula falsi* is applied to this modified function which leads to a very simple algorithm. The rate of convergence is shown to be quadratic or better.**

I. Method

Let the function be represented by $F(x)$. We create a new function $H(x)=F(x)\cdot e^{mx}$ in such a way that for three equidistant x values x_0, x_1 an x_2 the following requirement is met:

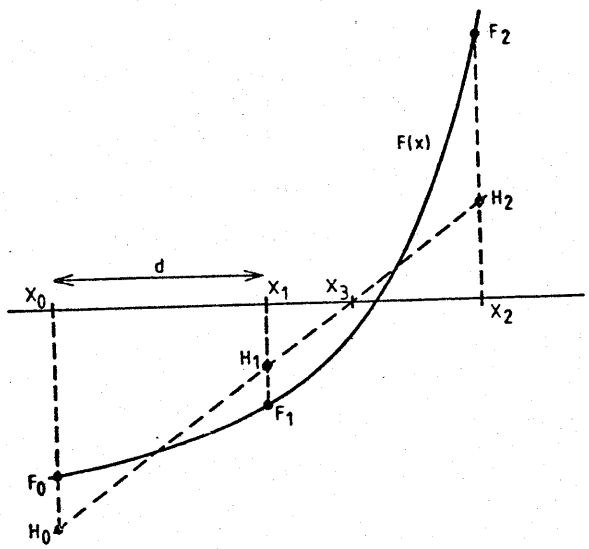


Fig. 1.

$$H_2-2H_1+H_0=0, \quad \text{with} \quad H_n=H(x_n). \tag{1}$$

Let $d=x_2-x_1=x_1-x_0$ and $F_0\cdot F_2<0$, then from (1) it follows

$$F_2\cdot e^{2md}-2F_1\cdot e^{md}+F_0=0 \tag{2}$$

with the analytical solution

$$e^{md}=\frac{F_1-\text{sign}(F_0)\cdot\sqrt{W}}{F_2}, \quad \text{with} \quad W=F_1^2-F_0F_2. \tag{3}$$

The factor sign (F_0) is deduced from the conditions $W>0$ and $e^{md}>0$. The next step is the application of the *regula falsi* to the points (x_1,H_1) and (x_2,H_2), which leads to the expression

$$x_3=\frac{x_1H_2-x_2H_1}{H_2-H_1}=x_1-\frac{d}{H_2/H_1-1} \tag{4}$$

where x_3 is the first approximation of the root of $F(x)$ and $H_2/H_1=F_2\cdot e^{md}/F_1$. Equation (4) can be written in the form

$$x_3=x_1+\text{sign}(F_0)\cdot\frac{F_1\cdot d}{\sqrt{W}}. \tag{5}$$

To avoid the factor sign (F_0) we divide numerator and denominator by F_0 and obtain the final expression for the algorithm:

$$x_3=x_1+d\cdot\frac{F_1/F_0}{\sqrt{(F_1/F_0)^2-F_2/F_0}}. \tag{6}$$

When $F_0\cdot F_2<0$, x_3 will be on the interval $[x_0,x_2]$ so convergence is guaranteed.

After computation of the first iterate x_3 we build up a new interval consisting of x_3 and one of the other remaining x values in such a way that $F_3\cdot F_n<0$ $(n=0,1,2)$ in order to be sure that the next iterate will remain on the starting interval. The procedure is depicted in Fig. 1.

The described method can even be used when $F_0=F_1$ or $F_1=F_2$ as can accidentally happen.

Suppose $F(x)=x^3-x-5$ and we choose $[-1,3]$ as the starting interval.

$$F_0=F_1=-5; \qquad F_2=19.$$

For x_3 we compute the value 1.9128, which is already fairly close to the root 1.904160859$\cdots$.

As $F_3>0$ we decide to take $[x_1,x_3]$ as the next interval of computation (Fig. 2).

The procedure can be terminated when a given accuracy is obtained.

II. Rate of Convergence

Let $e_n=x_n-r$ be the actual error between x_n and the root r. By means of a Taylor expansion in the vicinity of r we get $F_n\sim e_nf+e_n^2g+e_n^3h$ with $f=F'(r)$, $g=\frac{1}{2}F''(r)$, and $h=\frac{1}{6}F'''(r)\cdot d=e_1-e_0=e_2-e_1$ so the error at the first iterate is

$$e_3=e_1-e_1\cdot\frac{(e_1-e_0)(f+e_1g+e_1^2h)}{\sqrt{W}} \tag{7}$$

which can be derived from (5). This expression is valid for all possible shapes of $F(x)$.

$$W=e_1^2(f+e_1g+e_1^2h)^2-e_0e_2(f+e_0g+e_0^2h)(f+e_2g+e_2^2h).$$

After some adequate approximations we get

$$W\sim(e_1-e_0)^2\cdot\left[f^2+g^2(e_1^2+2e_0e_1-e_0^2)+2e_1fg+2fh(e_1-e_0)^2\right].$$

When $F_1\to 0$, $e_1\to 0$, and $e_1^2\ll|e_0e_2|$ so

$$e_3\sim\frac{1}{2}e_0e_1e_2\frac{g^2-2fh}{f^2}. \tag{8}$$

III. Examples

$F(x)=xe^x-10$ on $[-10,10]$

$x_3=0.06\cdots$
$2.75\cdots$
$1.71\cdots$
$1.746\cdots$
$1.74552798\cdots$
1.745528003

on $[-100,100]$

$x_3=6.10^{-21}$
$7.74\cdots$
$2.38\cdots$
$1.709\cdots$
$1.7458\cdots$
$1.745527990\cdots$
1.745528003

$F(x)=(\tan x)^{\tan x}-10^3$ on $[1.3,1.4]$

$x_3=1.352\cdots$
$1.356\cdots$
$1.3547099\cdots$
1.354710442

on $[0,1.5]$

$x_3=0.75\cdots$
$1.12\cdots$
$1.31\cdots$
$1.40\cdots$
$1.357\cdots$
$1.35429\cdots$
1.354710756
$1.354710442.$

$F(x)=\sin x$ on $[10,280]$, x in degrees. A trivial example.

$x_3=254.50\cdots$
$177.09\cdots$
$179.97\cdots$
$179.99995\cdots$
$180.$

IV. Conclusion

The proposed algorithm offers a good rate of convergence and is suitable especially on those cases where $F(x)$ is not strictly monotone. The method can be used when other three-point iterative methods (e.g., exponential or hyperbolic) fail.

ENGINEERING

(1) POSITION OF THE SUN AND THE SOLAR DIAGRAM

(2) HANG GLIDER PERFORMANCE AND IN THE WORKS;

(1) GEODESIC DOME DESIGN

(2) WIND ENERGY DESIGN

(3) SOLAR ENERGY DESIGN

THESE PROGRAMS ARE AVAILABLE FOR THE COST OF DUPLICATION ($5-10) AND A CASSETE TAPE ($5).

SINCERELY,

DAVID EAGLE
3330 S. GARLAND WAY
LAKEWOOD, CO. 80227

```
5000 REM SUBROUTINE CUBIC
5005 REM SOLUTION OF THE CUBIC EQUATION
5010 REM A*X*X*X + B*X*X + C*X + D = 0
5015 REM INPUTS, EQUATION COEFFICIENTS, A,B,C,D
5020 REM OUTPUT, ROOTS OF THE CUBIC EQUATION, X1,X2,X3
5025 REM
5030 P= B/A:Q= C/A:R= D/A:E= 1/3
5035 A1= (3*Q-P*P)/3:B1= (2*P*P*P-9*P*Q+27*R)/27
5040 D1= A1*A1*A1/27+B1*B1/4:IF ABS(D1) < 1E-10 THEN D1=0
5045 ON (2+SGN(D1)) GOTO 5050,5070,5080
5050 E0= 2*SQR(-A1/3):C1=-B1/(2*SQR(-A1*A1*A1/27))
5055 S1= SQR(1-C1*C1):GOSUB 9000
5060 Z1= E0*COS(P1/3):Z2= E0*COS(P1/3+2*PI/3)
5065 Z3= E0*COS(P1/3+4*PI/3):GOTO 5095
5070 Z1= SGN(-B1/2)*(2*ABS(-B1/2)^E)
5075 Z2= SGN(B1/2)*(ABS(B1/2)^E):Z3=Z2:GOTO 5095
5080 T1= -B1/2+SQR(D1):T2= -B1/2-SQR(D1)
5085 Z1= SGN(T1)*(ABS(T1)^E)+SGN(T2)*(ABS(T2)^E)
5090 PRINT "X1 REAL, X2,X3 COMPLEX"
5095 X1= Z1-P/3:X2= Z2-P/3:X3= Z3-P/3:RETURN
5100 REM
9000 REM ATAN3 SUBROUTINE
9005 REM INPUTS, S1= SIN(P1), C1=COS(P1)
9010 REM OUTPUT, P1= ATAN3(S1/C1), 0=< P1 <=2*PI
9015 IF ABS(S1) < 1E-10 THEN P1= 0:RETURN
9020 P1= (2-SGN(S1))*PI/2:IF ABS(C1) < 1E-10 THEN RETURN
9025 P1= P1+SGN(S1)*SGN(C1)*(ABS(ATAN(S1/C1))-PI/2):RETURN
```

SOLUTION OF

$$AX^3 + BX^2 + CX + D = 0$$

FOR REAL ROOTS.

NOTE PRINT OF "COMPLEX ROOTS" AT LINE 5090.

REFERENCE,

P.R. ESCOBAL, "METHODS OF ORBIT DETERMINATION"

NOTE USE OF SGN FUNCTION AT LINES 5070, 5075, 5085 REQUIRED BECAUSE CSS & SWTPC BASIC CANNOT EXPONENTIATE NEGATIVE NUMBERS.

N

```
8000 REM SUBROUTINE NYM4
8005 REM 4TH-ORDER NYSTROM INTEGRATOR
8010 REM INPUTS
8011 REM INITIAL CONDITIONS; TO,XO(I),VO(I)
8012 REM D1= STEP SIZE, N= NUMBER OF EQUATIONS
8013 REM E(I)= SYSTEM OF DIFFERENTIAL EQUATIONS
8014 REM OUTPUTS
8015 REM FINAL CONDITIONS; TO,XO(I),VO(I)
8020 REM INTEGRATOR COEFFICIENTS
8021 A1=.045:A2=.3:A3=13/126:A4=5/18:A5=5/42:B1=7/600:B2=7/30
8022 B3=-7/15:B4=7/6:B5=25/63:B6=.7:C1=19/78:C2=35/312:C3=15/104
8023 C4=64/39:C5=-70/39:C6=15/13:RETURN
8050 T= TO:FOR I=1 TO N:X(I)= XO(I):V(I)=VO(I):NEXT I:GOSUB 8500
8051 T= TO+A2*D1:FOR I=1 TO N:K1(I)= D1*E(I)
8052 X(I)= XO(I)+D1*(A2*VO(I)+A1*K1(I))
8053 V(I)= VO(I)+A2*K1(I):NEXT I:GOSUB 8500
8055 T= TO+B6*D1:FOR I=1 TO N:K2(I)= D1*E(I)
8056 X(I)= XO(I)+D1*(B6*VO(I)+B1*K1(I)+B2*K2(I))
8057 V(I)= VO(I)+B3*K1(I)+B4*K2(I):NEXT I:GOSUB 8500
8060 T= TO+D1:FOR I=1 TO N:K3(I)= D1*E(I)
8061 X(I)= XO(I)+D1*(VO(I)+C1*K1(I)+C2*K2(I)+C3*K3(I))
8062 V(I)= VO(I)+C4*K1(I)+C5*K2(I)+C6*K3(I):NEXT I:GOSUB 8500
8065 TO= TO+D1:FOR I=1 TO N:K4(I)= D1*E(I)
8066 XO(I)= XO(I)+D1*(VO(I)+A3*K1(I)+A4*K2(I)+A5*K3(I))
8067 VO(I)= VO(I)+A3*(K1(I)+K4(I))+B5*(K2(I)+K3(I))
8068 NEXT I:RETURN
```

FOURTH-ORDER NYSTROM INTEGRATION OF

$$\ddot{\underline{x}} = f(\underline{x}, \dot{\underline{x}}, t)$$

MUCH MORE ACCURATE THAN 4th ORDER RUNGE-KUTTA WITH SAME STEP SIZE.

NOTES

- GOSUB 8020 TO READ INTEGRATOR COEFFICIENTS
  CALL ONCE FROM MAIN PROGRAM DRIVER.
- GOSUB 8050 => ACTUAL NYSTROM INTEGRATOR
- GOSUB 8500 => SUBROUTINE WHICH EVALUATES SYSTEM OF DIFFERENTIAL EQUATIONS, E(I).

IF N>10 DIMENSION XØ, VØ, X, V, K1, K2, K3, K4 IN MAIN PROGRAM

TYPICAL DRIVER FOR NYM4

TO INTEGRATE A SYSTEM OF N DIFFERENTIAL EQUATIONS LOCATED AT 8500 FROM TIME Tϕ=0 TO TIME TI= 1.0 AT STEP SIZE DI= 0.1, A DRIVER MIGHT LOOK LIKE,

```
5   HOME
10  DIM  XØ(N), VØ(N), X(N), V(N),
         K1(N), K2(N), K3(N), K4(N)
15  TØ=0.0 : XØ=0.0 : VØ=0.0 : TI=1.0 : DI=.1
20  L= INT ((TI-TØ) * DI)
25  GOSUB  8020
30  FOR K=1 TO L
35  GOSUB  8050
40   FOR  J=1 TO N
45  PRINT  TØ, XØ(J), VØ(J)
50  NEXT J
55  NEXT K
60  END
```

FRIDAY+ 1/4/1980

DEAR JEFF,

GREETINGS AND FELICITATIONS FOR THE NEW YEAR AND NEW DECADE! HOPE YOU AND YOUR FAMILY ARE OFF TO A GOOD START FOR THE YEAR.

ON A FEW OCCASIONS DURING THE PAST YEAR I HAVE NEEDED TO RUN THE RANDOM NUMBER GENERATOR IN PROGRAMS WHERE SEVERAL HUNDRED DIFFERENT NUMBERS WITHIN GIVEN UPPER AND LOWER LIMITS ARE NEEDED. I HAVE FOUND THAT WHATEVER ROUTINE AND WHATEVER SEED QUANTITY IS USED IN RND RUNS TO EXHAUSTION A LITTLE TOO QUICKLY.

SHOWN ABOVE ARE TWO RUNS OF A PROGRAM WHICH PLACES /!'S AT RANDOM LOCATIONS ON THE SCREEN BETWEEN E000 AND E1FF. WITH 600 LOOPS THROUGH THE PROGRAM AND AN EFFICIENT RND FUNCTION, ONE WOULD EXPECT THE SCREEN TO END UP NEARLY FULL, BUT SUCH IS NOT THE CASE. IN FACT, THE SCREEN CEASES TO SHOW CHANGE LONG BEFORE THE 600 REPETITIONS ARE COMPLETE.

MY QUESTION OF YOU, JEFF, IS WHETHER YOU HAPPEN TO KNOW FROM YOUR MANY DISASSEMBLIES OF OUR BASIC WHERE THE SEED NUMBER FOR THE RND FUNCTION IS LOCATED AND WHERE THE RND FUNCTION ITSELF IS LOCATED? IT MIGHT BE POSSIBLE TO IMPROVE THIS FUNCTION EITHER BY USING SOME TRICKS TO CHANGE THE SEED NUMBER OCCASIONALLY OR BY INSERTING ANOTHER ALGORITHM FOR GENERATING RANDOM NUMBERS. IF YOU CAN HELP ME WITH THE APPROPRIATE ADDRESSES, I WOULD BE GRATEFUL.

YOU'LL NOTICE THAT THE PORTION OF THE PROGRAM LISTED ABOVE FROM LINE 50 TO THE END IS A NICE LITTLE ROUTINE WHICH WILL DUPLICATE THE SCREEN (32X16) CONTENTS ON THE PRINTER. FORWARD THIS TO OUR NEWSLETTER IF YOU WISH.

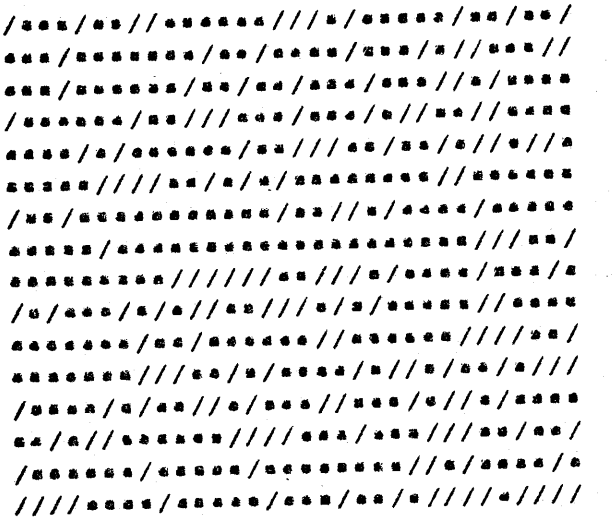
I WROTE THE FELLOW WHO WAS ASKING FOR PIM SCHEMATICS THAT I HAD THEM. I DIDN'T KNOW THEY WERE THAT HARD TO FIND, BUT FOR THE RECORD IN CASE ANYONE ELSE WANTS TO KNOW, I HAVE COMPLETE BLUEPRINTS FOR THE PIM/5 BOARD.

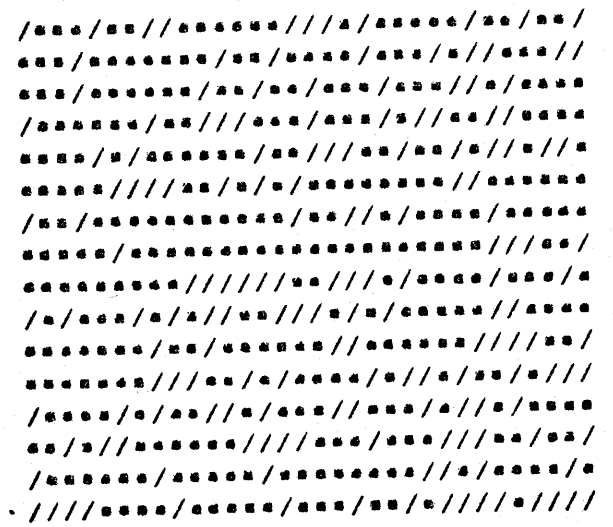
I HAVE MY EYE ON THE MECA HIGH SPEED CASSETTE TAPE UNIT CALLED THE BETA-1 AS A LOWER PRICED SUBSTITUTE FOR DISK. HAVE YOU HEARD ANY INFORMATION ABOUT THIS MACHINE WHICH YOU COULD PASS ON TO ME? FROM WHAT I HAVE READ ABOUT IT, IT LOOKS LIKE IT WOULD BE MORE EASILY INTERFACED TO SPHERE THAN THE EXATRON STRINGY FLOPPY.

BEST REGARDS TO YOU.

SINCERELY, Joe Dawes

```
0010 HOME
0020 FOR X=1 TO 600
0030 POKE( INT(((57855-57344+1)*RND(0)+57344)),47)
0040 NEXT X
0050 LET M=57344
0060 FOR X=1 TO 16
0070 FOR Y=1 TO 31
0080 PRINT #7,CHR$(PEEK(M));
0090 LET M=M+1
0100 NEXT Y
0110 PRINT #7,CHR$(PEEK(M))
0115 LET M=M+1
0120 NEXT X
0130 END
```





THE FOLLOWING BAR-GRAPHING PROGRAM, WRITTEN IN CSS BASIC, WILL GRAPH NUMERIC DATA ON AN 80 COLUMN PRINTER.

EACH BAR CAN BE LABELED WITH A STRING <19 CHARACTERS IN LENGTH.

THE SUMMING OPTION IS FOR TOTALING A LIST OF NUMBERS IN ORDER THAT THE SUM MAY BE GRAPHED. Ø IS THE ESCAPE FLAG.

WHEN THE PROGRAM CALLS FOR THE VALUES TO BE GRAPHED, IT EXPECTS THE LABEL STRING, A COMMA, ANDTHEN THE NUMERIC VALUE TO BE GRAPHED. THE ESCAPE FLAG TO EXIT INPUT AND START GRAPHING IS...END,Ø.

VALUES CAN BE GRAPHED EITHER AS A PERCENTAGE OF THE LARGEST VALUE IN THE LIST OR ON AN ABSOLUTE SCALE WITH EACH VALUE BEING ITS PROPORTIONAL SHARE OF THE AVAILABLE SPACE ON THE PAGE.

THE ACTUAL GRAPHING SUBROUTINE BEGINS AT LINE 9ØØØ. DEDICATED VARIABLE LABELS ARE Z,N,Y,X(N),X$(N).

THE PROGRAM IS JUST THE THING FOR SHOWING THE BAD NEWS ON HOW YOUR HOUSEHOLD BUDGET ITEMS HAVE STEADILY INCREASED OVER ME YEARS! AS IF YOU NEED TO SEE SUCH AGONIZING DATA!

JOSEPH DAWES

A little trick:

Charles Matteson told me that he is able to run keyboards as much as 150 feet from the Sphere Computer using long ribbon cables. The trick is to give the keyboard its own 5 Volt power supply and disconnect the 5 Volt line coming from the Sphere. J.B.

Make certain that DEF FN in your CSS works!
check locations for correct code

18A0 should have 17
18DB should have 36
18FF should have 13
now run this test program:

J.B.

```
0020 PRINT "ENTER THE RADIUS OF A CIRCLE (IN INCHES)";
0030 INPUT R
0040 DEF FNC(X)=2*PI*X
0050 PRINT "THE CIRCUMFERENCE OF A CIRCLE WITH A RADIUS OF ";R;" INCHES IS ";FNC
R);" INCHES"
ENTER THE RADIUS OF A CIRCLE (IN INCHES          4.000
THE CIRCUMFERENCE OF A CIRCLE WITH A RADIUS OF
       4.000  INCHES IS          25.132  INCHES
```

```
0005 REM BARGRAPHER (BG) - CSSBA
0007 REM JOSEPH DAWES
0010 HOME
0020 INPUT "HOW MANY ITEMS",X
0030 DIM X$(X):DIM X(X)
0035 LET N=1
0040 INPUT "SUMMING DESIRED",X$
0050 IF X$="NO" THEN 130
0060 LET X=0
0070 INPUT "OK",Z
0080 IF Z=0 THEN 110
0090 LET X=X+Z
0100 GOTO 70
0110 PRINT "SUM IS ";X
0130 PRINT "INPUT 'LABEL,VALUE'"
0140 INPUT X$(N),X(N)
0150 IF X$(N)="END" THEN 170
0160 LET N=N+1:GOTO 40
0170 LET N=N-1
0180 INPUT "%AGE OR ABSOLUTE SCALE",X$
0190 IF X$="%AGE" THEN 220
0200 GOSUB 9200
0210 END
0220 GOSUB 9000
0230 END
9000 GOSUB 9005
9002 GOTO 9080
9005 LET Z=N:LINE=100
9010 LET N=1
9020 LET Y=X(N)
9030 IF N=Z THEN RETURN
9040 LET N=N+1
9050 IF Y<X(N) THEN 9020
9060 IF N=Z THEN RETURN
9070 GOTO 9040
9080 PRINT #7:PRINT #7,"100% VALUE=";Y
9090 FOR N=1 TO Z
9100 PRINT #7,X$(N);TAB(21);
9110 IF X(N)<0 THEN X(N)=-1*X(N)
9120 IF X(N)=0 THEN 9160
9125 IF INT((X(N)/Y*45)+.5)<1 THEN 9190
9130 FOR C=1 TO INT((X(N)/Y*45)+.5)
9140 PRINT #7,"*";
9150 NEXT C
9160 PRINT #7," ";INT((X(N)/Y*100)+.5);"%"
9170 NEXT N
9180 LINE= 80:RETURN
9190 PRINT #7,"*";:GOTO 9160
9200 GOSUB 9005
9210 PRINT #7:PRINT #7,"SCALE=";Y/45;"/MARK--FULL SCALE=";Y
9220 FOR N=1 TO Z
9230 PRINT #7,X$(N);TAB(21);
9240 IF X(N)<0 THEN X(N)=-1*X(N)
9250 IF X(N)=0 THEN 9300
9260 IF INT((X(N)/(Y/45))+.5)<1 THEN 9330
9270 FOR C=1 TO INT((X(N)/(Y/45))+.5)
9280 PRINT #7,"*";
9290 NEXT C
9300 PRINT #7," ";X(N)
9310 NEXT N
9320 LINE= 80:RETURN
9330 PRINT #7,"*";GOTO 9300
9340 END
```

Additional CSS Commands
By Robert M. Grainger

The following notes describe routines to allow the use of CSS Basic with a Teletype. The other routines represent an alternate method of reading and writing DATA to tape without using TWRITE and TREAD which are admittedly restricting as to size of data block.

COMMANDS

For ASR 33

RDON	52	44	4F	4E	00 2936
RDOF	52	44	4F	46	00 293A
FORM	46	4F	52	4D	00 293E
BELL	42	44	4C	4C	00 2942
ECHO	45	43	48	4F	00 2946
NCHO	4E	43	48	4F	00 294A

For BASIC

TPOS	54	50	4F	53	00 295A
TBKL	54	42	4B	4C	00 2980
TBKS	54	42	4B	53	00 2989
DRV1	44	52	56	31	00 2950
DRV2	44	52	56	32	00 2954

*All is relocatable except byte 2928

To use an ASR 33 at Port 4 put in jump table:
0149 7E 2929 TTY out
014C 7E 291E TTY in
014F 7E 2911 TTY initialize
Now PRINT #4," " will print to TTY

One can run CSS completely from the teletype with the exception ii of the EDIT command which requires the screen and cursors.

To use the teletype, just put in PORT=4 making the screen inactive until you say Port=1

One can also just use the TTY as an extra input leaving the screen at the control pott (1)

To use paper tape
20 RDON
30 INPUT #4,A,B,C
40 RDOF

Paper tape must use commas and carriage return

```
0 005 REM TO ILUSTRATE USE OF TPOS,
 0006 REM TBKS &TBKL AT GO TIME
 0007 REM YOU CAN LOAD ANY BLOCK UP TO 256 BY 256 EG DIM D(255,255)
 0008 REM AS LONG AS IT IS FIRST in D/M statement
 0009 REM BOB GRAINGER ,CANADA
 0 010 DIM D(100)
 0 020 INPUT J
 0 030 INPUT D(J)
 0 040 IF D(J)>50 THEN GOTO 60
  0050 GOTO 20
 0 060 TPOS D0  positions tape after last file
 0 065 TBKS D1  writes new file
 0 070 PRINT "SAVED"
 0 080 INPUT J
 0 090 INPUT D(J)
 0 100 IF D(J)>50 THEN GOTO 120
  0110 GOTO 80
 0 120 TPOS D1 (optional)
 0 125 TBKL D2
 0 130 PRINT D(1);D(2);D(3)
 0 140 END
 0 161 REM USE TPOS TO FIND LAST SPOT
 0 162 REM WITHOUT OVERLOADING CURRENT JOB
```

FOR 3 SUBSCRIPTS
D(10,50) also D$(100,50)
FOR J=1 TO 10
FOR K=1 TO 25
FOR L=1 TO 2
M=((K-1)*25)+L
D(J,M) = To load buffer
A = D(J,M) to unpack

strings are more compact and easier to unpack or pack

Pull in as one a block as you need then break it up or rework it using Jeff & Rogers simulated disk statements.

TTY DRIVERS Block #1

```
        2911> 36 PSH A
INIT    2912> 86 LDA A #$03
        2914> B7 STA A $F050    .
        2917> 86 LDA A #$02
        2919> B7 STA A $F050    .
        291C> 32 PUL A
        291D> 39 RTS
TTY IN  291E> B6 LDA A $F050    .
        2921> 84 AND A #$01
        2923> 27 BEQ   $F9      291E
        2925> B6 LDA A $F051    .
        2928> 01 NOP   ECHO NoECHO = 39
TTY OUT 2929> 36 PSH A
        292A> B6 LDA A $F050    .
        292D> 84 AND A #$02
        292F> 27 BEQ   $F9      292A
        2931> 32 PUL A
        2932> B7 STA A $F051    .
        2935> 39 RTS
RDON    2936> 86 LDA A #$11
        2938> 20 BRA   $EF      2929
RDOF    293A> 86 LDA A #$13
        293C> 20 BRA   $EB      2929
FORM    293E> 86 LDA A #$0C
        2940> 20 BRA   $EF      2931
BELL    2942> 86 LDA A #$07
        2944> 20 BRA   $E3      2929
ECHO    2946> 86 LDA A #$01
        2948> 20 BRA   $02      294C
NOECHO  294A> 86 LDA A #$39
        294C> B7 STA A $2928    .
Block 2 294F> 39 RTS
DRV1    2950> 86 LDA A #$60
        2952> 20 BRA   $02      2956
DRV2    2954> 86 LDA A #$62
        2956> B7 STA A $0112    .
        2959> 39 RTS
```

Block #3

```
TPOS    295A> 8D BSR   $3D      2999
        295C> DE LDX   $20
        295E> DF STX   $3C
        2960> DF STX   $3E
        2962> BD JSR   $FB77    .
        2965> BD JSR   $FBB5    .
        2968> 5F CLR B
        2969> DE LDX   $3C
        296B> DF STX   $00
        296D> BD JSR   $FB7E    .
```

next page

TPOS continued

```
        2970> DE LDX   $00
        2972> 1B ABA
        2973> 16 TAB
        2974> 9C CPX   $3E
        2976> 27 BEQ   $03      297B
        2978> 08 INX
        2979> 20 BRA   $F0      296B
        297B> BD JSR   $FB97
        297E> 20 BRA   $25      29A5
TBKL    2980> 8D BSR   $17      2999
        2982> 8D BSR   $35      29B9
        2984> BD JSR   $FB91
        2987> 20 BRA   $1C      29A5
TBKS    2989> 8D BSR   $0F      2999
        298B> 8D BSR   $2C      29B9
        298D> 09 DEX
        298E> 09 DEX
        298F> EE LDX   00,X
        2991> 09 DEX
        2992> DF STX   $3E
        2994> BD JSR   $FB2D
        2997> 20 BRA   $0C      29A5
        2999> DE LDX   $20
        299B> DF STX   $F0
        299D> DE LDX   $22
        299F> DF STX   $F2
        29A1> BD JSR   $0C2A
        29A4> 39 RTS
        29A5> BD JSR   $0C61
        29A8> DE LDX   $EC
        29AA> DF STX   $3C
        29AC> DE LDX   $EE
        29AE> DF STX   $3E
        29B0> DE LDX   $F0
        29B2> DF STX   $20
        29B4> DE LDX   $F2
        29B6> DF STX   $22
        29B8> 39 RTS
        29B9> DE LDX   $22
        29BB> 08 INX
        29BC> 08 INX
        29BD> 08 INX
        29BE> 08 INX
        29BF> 08 INX
        29C0> 08 INX
        29C1> DF STX   $3C
        29C3> DF STX   $3E
        29C5> 39 RTS
```

```
     DE LDX  2C
     DF STX  6A
     BD JSR  0609          . FIND VARIABLE        MAT SET
     DE LDX  63
     DF STX  6C
     DE LDX  6A
     DF STX  2C
     BD JSR  12B7               BORROW LET CODE FROM BASIC
     DE LDX  6C
  09 DEX
  09 DEX
     EE LDX X00
     DF STX  6E
.... BD JSR  04AC               MOVE NUMBER BUILD BUFFER POINTER
LOOP BD JSR  04AC
     DE LDX  6C
     08 INX
     08 INX                          'SAMPLE PROGRAM
     08 INX
     08 INX                     0010 LET X=9.9
     08 INX                     0020 RJUST= 10
     08 INX                     0030 DIM A(3,3)
     9C CPX  6E                 0040 MATSET A(1,1)=X
     26 BNE  01      STORE      0050 MATPRINT A(1,1)
     39 RTS
STORE DF STX 6C                      9.9        9.9        9.9
     BD JSR  031E                    9.9        9.9        9.9
     20 BRA  E9      LOOP            9.9        9.9        9.9
```

Faster Looping

by Bob Van Valzah

(FASTER LOOPING by Bob Van Valzah appears courtesy of the CACHE Newsletter.)

How many times have you heard that you can't get something for nothing? In many cases, it is possible to increase the execution speed of a loop without taking any more memory! I will use BASIC to illustrate the idea, but it works in any language.

Suppose that you want to execute the body of a loop zero or more times. It is typically coded like this:

```
100 IF I=0 THEN 400        .exit loop if done
200 . . .                  .body of loop
300 GOTO 100               .branch to exit test
400 . . .                  .continue here
```

Compare the example above to this loop which has the same effect, but which runs faster:

```
100 GOTO 300               .branch to exit test
200 . . .                  .body of loop
300 IF I<>0 THEN 200       .loop until done
400 . . .                  .continue here
```

This second loop has the same statements, but the difference is in the loop overhead. Suppose the body of the loop has to be executed 1000 times. In the first example, 1000 GOTOs and 1000 IFs are executed in addition to the body of the loop. In the second example, one GOTO and 1000 IFs are executed. Thus we have eliminated the time it takes to execute the GOTO from all iterations of the loop but the first!

This technique assumes that it takes no longer to test if a condition is false than to test if it is true. This is almost always the case, e.g. I <> 0 is just as fast a I=0. The benefit from using this technique is greatest if the body of the loop is executed a large number of times. I wish that I could take credit for thinking this one up, but I can't. The idea appears in an article written by David Feign of Chapman College in the November issue of SIGPLAN NOTICES.

```
START  BD JSR  109A
       DE LDX  2C
       BD JSR  0609
       8D BSR  76        FIND
       DE LDX  63
       DF STX  6E
       09 DEX
       09 DEX
       09 DEX
       09 DEX
       EE LDX X00
       DF STX  6A
       DF STX  6C
       CE LDX #006A
       7D TST  009C              TEST FOR BASE=
       27 BEQ  0C        BASE1
       6C INC X00
       6C INC X02
       6D TST X01
       27 BEQ  04        BASE1
       6C INC X01
       6C INC X03
BASE   6D TST X01
       26 BNE  0A        PRNT
       A6 LDAAX00
       A7 STAAX01
       86 LDAA#01
       A7 STAAX00
       A7 STAAX02
PRNT   7A DEC  006B
LOOP   DE LDX  63
       D6 LDAB 6C
INX6   08 INX
       08 INX
       08 INX
       08 INX
       08 INX
       08 INX
       5A DECB
       26 BNE  F7        INX6
       DF STX  63
       BD JSR  0353
       8D BSR  3C        PRNT1
       7A DEC  006B
       26 BNE  E7        LOOP
       7A DEC  006A
       26 BNE  08        INX6E
       CE LDX #0001
       DF STX  70
       7E JMP  02C6
INX6E  DE LDX  6E
       08 INX
       08 INX
       08 INX
       08 INX
       08 INX
       08 INX
       DF STX  6E
       BD JSR  111A
       01 NOP
       BD JSR  0353
```

MAT PRINT I

SAMPLE PROGRAM

```
0010 DIM A(6,6)
0020 MATPRINT #0,A(1,1)

 0  0  0  0  0  0
 0  0  0  0  0  0
 0  0  0  0  0  0
 0  0  0  0  0  0
 0  0  0  0  0  0
 0  0  0  0  0  0
```

```
      8D BSR  0A       FIND
      96 LDAA 6D
      97 STAA 6B
      DE LDX  6E
      DF STX  63
      20 BRA  BA
FIND  D6 LDAB 70       PRNT
      27 BEQ  09       PRNT1
      5A DECB
SPACE 86 LDAA#20
      BD JSR  01F1
      5A DECB
      26 BNE  F8       SPACE
PRNT1 7E JMP  11BE
```

MAT PRINT for CSS BASIC

Mat Print actually prints out a list(one dimensioal) or an array (two dimensional). The routine will correctly coordinate with BASE=, RJUST, MAT TAB, MAT SPREAD, DIGITS= and PORT= all of which add flexibility to the appearance of the printed array. In the sample program the printer is at port 0.

Mat Tab works like TAB and thus moves the entire matrix to the right. Mat Spread spreads spaces between the elements and is particularly nice if each element has many digits. Without the spreading, the printout is very tightly packed and is hard to read.

RJUST moves the printing to the right, lines up the decimal points and causes some spreading out also.

In order to generate the Mat tab and Mat spread commands, I had to create two memory pointers and initialize them at hard start. It was possible to do this without needing more bytes in the original CSS pointer intialize routine.

At 0A54

```
4F 97 3E 97 3F 97 48 97 74 97 75 97 58 97 9B 97 9C  old
CE 00 00 DF 3E DF 74 DF 9B 4F 97 48 97 58 08 DF 70  new
```

It was also necessary, for spreading, to have the BASIC's PRINT routine have the ability to put various numbers of spaces after a number is printed. The final version of the routine to allow this responds to the value of location 71. This may be used for other purposes also. For instance USING previously always followed a numeric with a space. If one takes the trouble to add to his USING routine a clear 71 before USING (and a load 71 back with a 1 afterwards) then USING will be a little more perfect routine.

At 11E0 PRINT outputted the single space with 7E 02F6

Change it to 7E XXXX where XXXX is the address of the following little routine.

```
7D TST  0071
26 BNE  01
39 RTS
D6 LDAB 71
BD JSR  02F6
5A DECB
26 BNE  FA
39 RTS
```

The MAT TAB routine is:

```
BD 1808   get number of columns to tab to
D7 70     store it
39        return
```

The MAT SPREAD routine similarly is:

```
BD 1808 get number of spaces to spread out
D7 71   store it
39      return
```

```
------
 6B80 BD JSR  FC37
6B83 BD JSR  FC3D
6B86 86 LDAA#44  D
6B88 BD JSR  FCBC
6B8B 86 LDAA#55  U
6B8D BD JSR  FCBC
6B90 86 LDAA#4D  M
6B92 BD JSR  FCBC
6B95 86 LDAA#50  P
6B97 BD JSR  FCBC
6B9A BD JSR  6CF5
6B9D 86 LDAA#53  S
6B9F BD JSR  FCBC
6BA2 86 LDAA#54  T
6BA4 BD JSR  FCBC
6BA7 86 LDAA#41  A
6BA9 BD JSR  FCBC
6BAC 86 LDAA#52  R
6BAE BD JSR  FCBC
6BB1 86 LDAA#54  T
6BB3 BD JSR  FCBC
6BB6 BD JSR  6CF5
6BB9 86 LDAA#3D  =
6BBB BD JSR  FCBC
6BBE BD JSR  6CF5
6BC1 BD JSR  FEE4
6BC4 F7 STAB 6D04
6BC7 B7 STAA 6D05
6BCA BD JSR  6CF5
6BCD 86 LDAA#45  E
6BCF BD JSR  FCBC
6BD2 86 LDAA#4E  N
6BD4 BD JSR  FCBC
6BD7 86 LDAA#44  D
6BD9 BD JSR  FCBC
6BDC BD JSR  6CF5
6BDF 86 LDAA#3D  =
6BE1 BD JSR  FCBC
6BE4 BD JSR  6CF5
6BE7 BD JSR  FEE4
6BEA F7 STAB 6D02
6BED B7 STAA 6D03
6BF0 B6 LDAA 6D04
6BF3 BD JSR  FCBC
6BF6 B6 LDAA 6D05
6BF9 BD JSR  FCBC
6BFC BD JSR  6CF5
6BFF BD JSR  6CF5
6C02 FE LDX  6D04
6C05 FF STX  6D00
6C08 BD JSR  FC37
6C0B BD JSR  FC3D
6C0E C6 LDAB#0F
6C10 F7 STAB 6CFF
6C13 B6 LDAA 6D00
6C16 BD JSR  FF02
6C19 B6 LDAA 6D01
6C1C BD JSR  FF02
6C1F BD JSR  6CF5
6C22 FE LDX  6D00
6C25 FF STX  6CFD
6C28 C6 LDAB#04
6C2A F7 STAB 6CFB
6C2D C6 LDAB#04
6C2F F7 STAB 6CFC
6C32 A6 LDAAX00
6C34 BD JSR  FF02
6C37 FE LDX  6CFD
6C3A BC CPX  6D02
6C3D 27 BEQ  59
6C3F 08 INX
6C40 FF STX  6CFD
6C43 7A DEC  6CFC
6C46 26 BNE  EA
6C48 BD JSR  6CF5
6C4B FE LDX  6CFD
6C4E 7A DEC  6CFB
6C51 26 BNE  DA
6C53 27 BEQ  02
6C55 20 BRA  DB
6C57 BD JSR  6CF5
6C5A BD JSR  6CF5
6C5D BD JSR  6CF5
6C60 C6 LDAB#10
6C62 F7 STAB 6CFC
6C65 FE LDX  6D00
6C68 FF STX  6CFD
6C6B A6 LDAAX00
6C6D 81 CMPA#20
6C6F 2D BLT  04
6C71 81 CMPA#5F
6C73 2F BLE  02
6C75 86 LDAA#2E  .
6C77 BD JSR  FCBC
6C7A FE LDX  6CFD
6C7D BC CPX  6D02
6C80 27 BEQ  33
6C82 08 INX
6C83 7A DEC  6CFC
6C86 26 BNE  E0
6C88 FF STX  6D00
6C8B 7A DEC  6CFF
6C8E 27 BEQ  5D
6C90 DE LDX  1C
6C92 BD JSR  FD14
6C95 7E JMP  6C13
6C98 F6 LDAB 6CFC
6C9B 58 ASLB
6C9C B6 LDAA 6CFB
6C9F 81 CMPA#04
6CA1 26 BNE  02
6CA3 CB ADDB#05
6CA5 C0 SUBB#01
6CA7 F7 STAB 6CFC
6CAA BD JSR  6CF5
6CAD 7A DEC  6CFC
6CB0 26 BNE  A3
6CB2 7E JMP  6C57
6CB5 BD JSR  6CF5
6CB8 BD JSR  6CF5
6CBB BD JSR  6CF5
6CBE BD JSR  6CF5
6CC1 86 LDAA#44  D
6CC3 BD JSR  FCBC
6CC6 86 LDAA#4F  O
6CC8 BD JSR  FCBC
6CCB 86 LDAA#4E  N
6CCD BD JSR  FCBC
6CD0 86 LDAA#45  E
6CD2 BD JSR  FCBC
6CD5 BD JSR  6CF5
6CD8 86 LDAA#3F  ?
6CDA BD JSR  FCBC
6CDD BD JSR  6CF5
6CE0 BD JSR  FE71
6CE3 81 CMPA#59  Y
6CE5 26 BNE  0B
6CE7 7E JMP  FE64
6CEA 7E JMP  6C08
6CED BD JSR  FC4A
6CF0 26 BNE  F8
6CF2 7E JMP  6B80
6CF5 86 LDAA#20
6CF7 BD JSR  FCBC
6CFA 39 RTS
```

Here is Harry Friedman's utility program to dump blocks of memory contents to a 64 char/line screen in hex and Ascii.

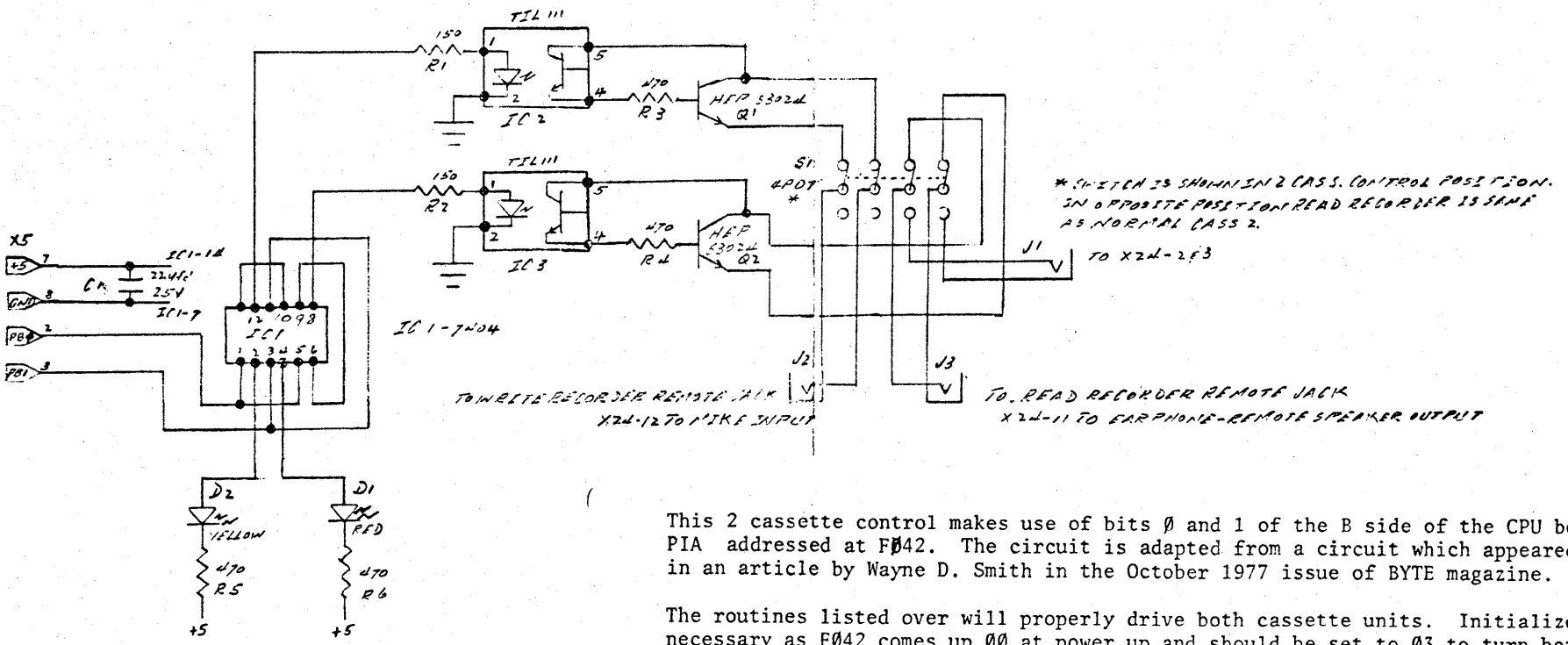
The format is:

XXXX XXXX XXXX XXXX XXXX ASCII

```
A  4  4  4  4  UP
D  B  B  B  B  TO
D  Y  Y  Y  Y  8
R  T  T  T  T  Asci
E  E  E  E  E  Letter
S  S  S  S  S
S
```

For further info send a blank tape & request object or source listing to be recorded.

Harry Friedman
945 Dudley Dr.
Shreveport La.
71104



This 2 cassette control makes use of bits Ø and 1 of the B side of the CPU board PIA addressed at FØ42. The circuit is adapted from a circuit which appeared in in an article by Wayne D. Smith in the October 1977 issue of BYTE magazine.

The routines listed over will properly drive both cassette units. Initialize is necessary as FØ42 comes up ØØ at power up and should be set to Ø3 to turn both drives off. Proper operation of the cassette drives is still necessary to put them in the required operation mode.

I load these routines in high memory (4F1Ø), but since they are only 2AHex bytes long they could be located almost anywhere.

PARTS LIST

IC1	7404 Hex inverter
IC2,3	TIL111 opto coupler
Q1,2	HEP S3024
D1	Red LED
D2	Yellow LED
R1,2	150 ohm
R3/6	470 ohm
C1	22ufd, 25V
J1/3	Miniature or sub-miniature phone jack
S1	4 pole, double throw switch (min)

Regards,

Harry Friedman
945 Dudley Dr., Shreveport, La. 71104

NO. 340-4 DIETZGEN GRAPH PAPER
4 X 4 PER INCH

DIETZGEN CORPORATION
MADE IN U.S.A.

TITLE 2 CASSETTE CONTROL

DATE 2/ /19

PAGE 1 OF 1

	LABEL	OP CODE	OPERAND(S)	ADDRESS	MACHINE CODE
1	MNEMONIC	CODE			
	REWIND	LDAA	#$00	4F10	86 00
4		STAA	$F042	12	B7 F042
		JSR	$FC4A	15	BD FC4A
		JMP	INTLIZ	18	7E 4F1C
		RTS		1B	39
8	INTLIZ	LDAA	#$03	1C	86 03
		STAA	$F042	1E	B7 F042
		RTS		21	39
	TAPSAV	LDAA	#$01	22	86 01
12		STAA	$F042	24	B7 F042
		JSR	$FB2D	27	BD FB2D
		JMP	INTLIZ	2A	7E 4F1C
		RTS		2D	39
16	TPLOAD	LDAA	#$02	2E	86 02
		STAA	$F042	30	B7 F042
		JSR	$F891	33	BD F891
		JMP	INTLIZ	36	7E 4F1C
20		RTS		4F39	39
	INITIZ	EQU	$4F1C (LOAD ADDRESS + 13$_{10}$)		
24	TO EXECUTE REWIND USUALLY KEY AN "R" TO START AND A CARRIAGE RETURN TO STOP				
28					

OVER

For Sale: *One tape and manual of CSS Basic with manual and patched for SPHERE (including renumber and matrix operators). Be certain to describe your system when ordering. $27*

Free: *To those who have purchased DYNASOFT PASCAL If you send a blank tape we will return a copy of the Spherized Pascal including save and load programs to cassette. The master cassette is patched for a 20K V3N system.*

FOR SALE: Sphere Microprocessor * (V3N Proms)

CPU/2
MEM w/4K on board
SIM w/Cass II (FØ6Ø)
CRT w/Composite & RF Video Out

(CRT needs replacement of one ROM)

Power Supply
Keyboard

Misc: Connecting cables; low profile case, modified to accept home-brew card rack; Sphere Manual; Motorola 6800 Application & Programming Manuals; plus misc. tapes and software.

Any reasonable offer will be considered.

Write: J. D. Tregeagle
Box B-35638
CTF-C / F-336
Soledad, CA 93960

*Everything in working condition except as noted on CRT - needs a MCM6810AL

FOR SALE....

ADM 3A TERMINAL
24 LINES X 80 CHARACTERS
UPPER/LOWER CASE

$650

VIC WINTRISS
800-621-1466 EXT 1041

Wanted: any information about the DATA DISC Corp. formerly located at 1275 California Ave., Palo Alto Ca. Gullible old Jeff bought their Model F-3 Hard Disk with electronics. Not to worry. I paid only fifteen dollars for the whole thing.